Université Catholique de Louvain
Faculté des Sciences Appliquées

Scheduling of Mixed Batch-Continuous Production Lines

Thèse présentée en vue de l'obtention du grade
de Docteur en Sciences Appliquées par

François Warichet

Acknowledgements

I would like to thank many people who have helped me to achieve this long-term work.

Firstly, I would like to thank my supervisor Yves Pochet for his unlimited encouragement to keep my work progressing; his kindness is recognised here at CORE by everyone. His expertise in his field and his support were indispensable. He has always been open and available and had time to read my numerous drafts even when he was very busy with other tasks.

I really appreciate the kindness of Georges Bastin, my second supervisor. His stimulating comments, remarks and ideas helped me to progress in my research and develop better links between the areas of Control and Optimisation.

I would also like to thank the other members of my jury. I would like to thank Laurence Wolsey, the supervisor of my Master's thesis, who introduced me to the area of Operations Research. He provides many opportunities to the researchers here at CORE to improve their knowledge of the field by meeting people in the domain. I also appreciate the comments and remarks of Denis Dochain. I would like to thank Ignacio Grossmann for accepting to come from the US to Belgium in order to participate to my pre-defense. His comments and suggestions have helped me to improve and clarify previous versions of the thesis. Finally, I would like to thank Vincent Blondel for having accepted to be the president of the jury.

During four years, I was financed by the Solvay Research fund. I really appreciated working on real-size industrial problems and tried to provide an adequate solution method for the specific problem identified. The topic of my research was subdivided in two parts - control and optimisation. I therefore had the pleasure to work with Iliyana Simeonova from the INMA departement. She is very open and always ready to help. I particularly appreciate the interaction we have had. I gratefully acknowledge the support of the European Network in Algorithmic Discrete Optimization (ADONET 504438) and the support of the engineering department at the Université Catholique de Louvain.

I want to say some words about CORE. The atmosphere at CORE is really wonderful. You meet a lot of people from everywhere and I really appreciate this international environment. I particularly enjoyed the friendship of Peter and his daily complaints about Belgian administration. I also enjoyed our numerous badminton or squash games, even if I could not win enough. I enjoyed the very good company of Robert, Ruslan, Joël, Ayse, Jean-Sébastien, Géraldine, Giorgia, Xenia, Alain, Gilles, Nicolas, Elina, Yu, Santanu, Michel, Quentin, Diego, Joao, Alexander, Kent, Bharath, Helena and François. I also want to especially thank Benoît Gailly for his encouragements and advice. I also want to thank Alphonse Magnus for his help during my doctorate.

I would like to thank all the administrative staff at CORE for their wonderful work and permanent kindness.

Finally, I would like to thank all of my family for their support during all these years. What you have done for me is invaluable. My wife Fabienne was for me an enormous source of motivation and support. I would like to thank my two sons Simon and Victor for their patience with their father at the very end of the thesis.

Contents

1 Introduction **1**
- 1.1 Scheduling of a mixed plant : Models & Algorithms 2
- 1.2 Mixed integer programming optimization 8
  - 1.2.1 Branch-and-Bound algorithm 8
  - 1.2.2 Solution of the basic example 10
- 1.3 Improving MIP formulations 12
  - 1.3.1 Cutting plane algorithm 12
  - 1.3.2 Strengthening techniques 17
  - 1.3.3 Application to the basic example 19
- 1.4 MIP heuristics 21
- 1.5 Outline and contribution of the thesis 23

2 Continuous time formulation **27**
- 2.1 Problem description 29
- 2.2 Models Classification 31
- 2.3 One batch task 33
  - 2.3.1 Model formulation 34
  - 2.3.2 Valid Inequalities : variable lower bounds on the cycle duration 52
- 2.4 Multiple batch tasks 54
  - 2.4.1 Initial and strengthened formulation 54
  - 2.4.2 Successive Batch Tasks 58
  - 2.4.3 Valid inequalities : lower bounds on the cycle duration . . . 60
- 2.5 Mixed plant : batch and continuous tasks 61
  - 2.5.1 Model formulation 61
  - 2.5.2 Strengthened and valid inequalities 63
- 2.6 Conclusion 66

3 Computational Experiments **67**
- 3.1 Basic case 68
  - 3.1.1 One batch task 69
  - 3.1.2 One batch and one continuous task 74
- 3.2 Multiple Batch Task Case 78
  - 3.2.1 Description 78
  - 3.2.2 Multiple batch tasks, no continuous task 82
  - 3.2.3 Multiple batch tasks and one continuous task 88

3.2.4 Comparison with the formulation of Schilling and Pantelides [44] 100
3.2.5 The choice of the values of T and d 109
3.3 Industrial case 111
3.3.1 Problem description 111
3.3.2 Comparison of Heuristics 111
3.3.3 Bounds on the maximal productivity 113
3.3.4 Solution of the industrial case 115
3.4 Conclusion 117

4 Modeling the process dynamics **121**
4.1 Modeling the process dynamics 123
4.1.1 Description of the process dynamics 124
4.1.2 Approximated model formulation 124
4.2 Application to a single batch task 129
4.3 Application to parallel batch tasks 144

5 Conclusion **149**

Chapter 1

Introduction

We address in this thesis the scheduling problem of production lines of a plant. We want to obtain the best production program of these lines according to a given criterion.

The scheduling problem considered in this thesis is composed of tasks that have to be processed taking into account that some capacity restrictions have to be satisfied for some resources shared among the tasks. We consider two types of tasks : batch and continuous. A batch task is processed during a certain amount of time and produces at the end a fixed amount of product. A continuous task is processed continuously and its decision variable is the speed or processing rate at which it is performed. Both types of tasks consume resources with limited capacity or availability. The objective is to obtain a schedule optimizing a specific productivity criterion.

In order to model and solve such a problem, we use the mixed integer programming (MIP) approach, i.e. the approach based on the optimization of a mixed integer program defined by linear constraints and both continuous and discrete variables.

In Section 1.1, we describe the scheduling problem that we have to solve and we illustrate how to model a special case of the general problem. In Section 1.2, we outline the mixed integer programming algorithm used to solve such a problem. In this approach, the quality or tightness of the formulation is crucial in order to be able to solve large scale instances to optimality. In Section 1.3, we give the main methods used in the thesis in order to improve or tighten model formulations. Even with good or tight formulations, it remains sometimes difficult to solve large instance to optimality. Therefore, heuristic methods have to be used in order to obtain good solutions quickly. In Section 1.4, we present MIP-based heuristic methods and finally, in Section 1.5, we give the outline of the thesis.

1.1 Scheduling of a mixed plant : Models and Algorithms

We consider a production process where the resources are the processing units and the utilities shared by the tasks, and the storage tanks containing the intermediate products produced or consumed by the tasks. In this process, there are both batch and continuous tasks. Each batch task has a processing time (fixed or variable), can be processed on a subset of the reactors or processing units, and can be repeated several times. The starting time of each instance of a batch task, as well as the processing unit used, have to be decided by the scheduler. Precedence and zero waiting time constraints exist between some of the batch tasks. Each continuous task has a processing rate that has to be decided by the scheduler, and falls between some lower and upper technological limits. The batch and the continuous tasks consume and produce resources, for which we have some capacity restrictions. Moreover, some of the continuous tasks cannot be interrupted. The objective is to obtain a schedule of the mixed plant optimizing a given criterion. The criterion used in this thesis is the productivity.

Two types of formulations are typically proposed in the literature to model such scheduling problems as mixed integer programs : discrete time formulations and continuous time formulations.

Initially, scheduling problems were modeled by discrete time formulations (see for example Kondili et al. [25]) using time intervals of fixed duration and a state task network representation to model the process network. To better model the various types of resources used, the resource task network was introduced (see Pantelides [35]) to generalize the concept of state task network. Typically discrete time formulations require a large number of small time intervals to model the problem accurately and obtain realistic solutions. This gives rise to large size models in terms of number of variables. However, given that the formulation is usually strong (i.e. the duality gap is rather small), one can hope to solve moderate size problems to near optimality.

Even though the number of variables is very large, the number of time intervals in which an event occurs in an optimal solution is usually quite small. Therefore, continuous time formulations were proposed to reduce the size of the formulation, see among others Zhang and Sargent [51], Pinto and Grossmann [36] and Mockus and Reklaitis [32]. The characteristic of these types of formulations is that time intervals have variable duration, and the end of a time interval corresponds to an event (start or end of a batch, modification of the availability of a resource, ...) where the status of the scheduling system changes. Consequently, the number of time intervals required to model the scheduling problem accurately can be much smaller and is close to the number of events that really occur. The continuous time formulations can be based on time slots or on events.
For the slot-based formulation, time is decomposed into a set of consecutive time slots of variable duration and a batch task is assigned to a set of consecutive time slots. The representation of time can be global, i.e synchronized for all pro-

cessing units (see for example Schilling and Pantelides [43]) or unit specific, i.e asynchronous (see for example Karimi and McDonald [24]).
For the event-based formulations, the variables correspond to the starting and the ending times of each batch task expressed in absolute time units, and model events that occur at different moments. Here also, the representation of time can be global (see for example Zhang and Sargent [51] and Mockus and Reklaitis [32]) or unit specific (see for example Ierapetritou and Floudas [22]).
The main interest of continuous time formulations is that the number of variables is very small. The mixed integer program is compact but typically weak in the sense that the duality gap is large, and therefore the number of Branch and Bound nodes needed to obtain an optimal solution is large. This is due, very often, to the necessary introduction of so-called big M type constraints to obtain a correct model formulation.

For recent literature reviews about scheduling formulations for chemical processing systems, see Floudas and Lin [21] and Mendez et al. [30].

In order to illustrate the difference between discrete and continuous time formulations, we show how to model a special case of the general problem that we want to address.

Consider the scheduling of a mixed plant composed of one batch and one continuous task. The batch task has a fixed processing time p. This batch task can be performed on *nbr_unit* reactors. At the end of the batch task, a fixed quantity (B $[ru]$(resource units, for generality)) of an intermediate product is discharged in a storage tank. The continuous task pumps continuously this intermediate product from the storage tank either to distribute it or to feed into a next processing stage. The speed of the continuous task must be in the interval $[\underline{\rho}, \overline{\rho}]$ $[ru/h]$. The capacity of the storage tank is limited between $[0, C]$ $[ru]$. The objective is to maximize for a fixed time horizon of length T the quantity processed by the continuous task. In Figure 1.1, this problem is represented.

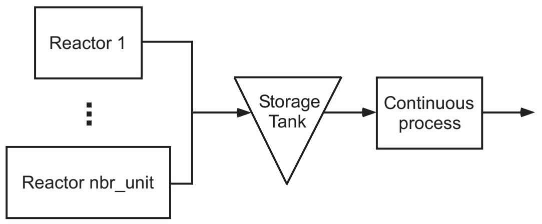

Figure 1.1: A simple production process

We first formulate this problem as a discrete time MIP formulation based on the paper by Kondili et al. [25]. The scheduling time horizon is divided into $T1$ time intervals ($t = 1, \ldots, T1$) of fixed and equal duration (Δt such that $T = T1.\Delta t$) that is a divisor of the processing time of the batch task p. Below, $p1 = \frac{p}{\Delta t}$ is integral and corresponds to the number of discrete time periods needed in order to perform the batch task. The parameters $T1$ and Δt are critical because they determine the size of the formulation, and have to be fixed before solving the problem instance. These parameters fix the length of the time horizon.

The variables of the problem are

$W_{j,t}$	: equals 1 if processing unit j starts processing a batch task at the beginning of time period t, and equals 0 otherwise.
q_t	: is the quantity $[ru]$ processed by the continuous task during time period t, $q_t \geq 0$
s_t	: is the quantity $[ru]$ of intermediate product in the storage tank at the beginning of time period t, after addition of the output of batches finished at the end of time period $t-1$.
sf_t	: is the quantity $[ru]$ of intermediate product in the storage tank at the end of time period t, just before adding the output of batches finishing at the end of time period t.

The MIP discrete time formulation for this example is

$$\max \sum_{t=1}^{T1} q_t \tag{1.1}$$

$$st \quad \sum_{t'=t-p1+1|t'\geq 1}^{t} W_{j,t'} \leq 1 \ \forall j \in \{1,\ldots,nbr\_unit\}, t \in \{1,\ldots,T1\} \tag{1.2}$$

$$s_t = sf_{t-1} + B \sum_{j=1|t-p1\geq 1}^{nbr\_unit} W_{j,t-p1} \ \forall t \in \{2,\ldots,T1\} \tag{1.3}$$

$$sf_t = s_t - q_t \ \forall t \in \{1,\ldots,T1\} \tag{1.4}$$

$$\underline{\rho}\Delta t \leq q_t \leq \overline{\rho}\Delta t \ \forall t \in \{1,\ldots,T1\} \tag{1.5}$$

$$0 \leq s_t, sf_t \leq C \ \forall t \in \{1,\ldots,T1\} \tag{1.6}$$

$$W_{j,t} \in \{0,1\} \ \forall j \in \{1,\ldots,nbr\_unit\}, t \in \{1,\ldots,T1\} \tag{1.7}$$

$$s_1 = ST0 \tag{1.8}$$

where the objective (1.1) is to maximize the quantity processed by the continuous task over the time horizon, constraint (1.2) imposes that for each processing unit and during each time period, at most one batch task can be performed, constraint (1.3) and (1.4) are the balance constraint for the product in the storage tank at the beginning and at the end of the time period t, where the output of a batch task finishing at the end of time period $t-1$ is added to sf_{t-1} to obtain s_t, and where the continuous discharge q_t during time period t is subtracted from s_t to obtain sf_t. Constraint (1.5) limits the speed of the continuous task, constraint (1.6) restricts the storage tank level. Constraint (1.7) defines W as binary variables, and finally constraint (1.8) are the specific initial conditions considered for this example, i.e. we impose an initial storage tank level $ST0$.

We then formulate the problem with a continuous time formulation as a MIP based on a formulation similar to the one proposed by Schilling and Pantelides

[43]. The time horizon is divided into $T2$ time intervals $t = 1, \dots, T2$ of variable duration. To avoid confusion, we call such time intervals of variable duration time slots. The main difference between the discrete and continuous time formulations is that, because we do not know in advance the moments in time when new batches start their processing, $T1$ has to be large in order to model these start times with enough accuracy. Whereas $T2$ can be as small as the number of batches starting over the entire scheduling horizon. Therefore, $T1 >> T2$ in a typical application. The parameter $T2$ is critical and has to be fixed before solving the problem instance. There is no way to decide the best value of $T2$ for a given problem. $T2$ is therefore part of the problem formulation.

The variables of the continuous time formulation of this problem are

τ_t :	is the duration of time slot t $[h]$, $\tau_t \geq 0$.
$z_{t,t'}$:	$= 1$ if a batch task starts at the beginning of time slot t and finishes at the end of time slot t', $= 0$ otherwise.
q_t :	is the quantity $[ru]$ processed by the continuous task during time slot t, $q_t \geq 0$.
r_t :	is the quantity $[ru]$ of intermediate product in the storage tank at the beginning of time slot t, just after the event occurring at the end of time slot $t-1$ (end of batches), $r_t \geq 0$.
rf_t :	is the quantity $[ru]$ of intermediate product in the storage tank at the end of time slot t, just before the event occurring at the end of time slot t (end of batches), $rf_t \geq 0$.

The MIP continuous time formulation for this example is

$$\max \sum_{t=1}^{T2} q_t \tag{1.9}$$

$$st \qquad p\ z_{t,l} \leq \sum_{k=t}^{l} \tau_k \qquad \forall t, l : 1 \leq t \leq T2, t \leq l \leq T2 \tag{1.10}$$

$$\sum_{k=t}^{l} \tau_k \leq p z_{t,l} + T(1 - z_{t,l}) \forall t, l : 1 \leq t \leq T2, t \leq l \leq T2 \tag{1.11}$$

$$\sum_{l=t}^{T2} z_{t,l} \leq 1 \qquad \forall t : 1 \leq t \leq T2 \tag{1.12}$$

$$\sum_{t=1}^{l} z_{t,l} \leq 1 \qquad \forall l : 1 \leq l \leq T2 \tag{1.13}$$

$$\sum_{t=1}^{t''} \sum_{t'=t''}^{T2} z_{t,t'} \leq nbr\_unit \qquad \forall t'' : 1 \leq t'' \leq T2 \tag{1.14}$$

$$r_t = rf_{t-1} + B \sum_{t'=1}^{t-1} z_{t',t-1} \forall t \in \{2, \ldots, T2\} \tag{1.15}$$

$$rf_t = r_t - q_t \qquad \forall t : 1 \leq t \leq T2 \tag{1.16}$$

$$\underline{\rho}\tau_t \leq q_t \leq \overline{\rho}\tau_t \qquad \forall t : 1 \leq t \leq T2 \tag{1.17}$$

$$0 \leq r_t, rf_t \leq C \ \forall t \in \{1, \ldots, T2\} \tag{1.18}$$

$$z_{t,l} \in \{0,1\} \forall t, l : 1 \leq t \leq T2, t \leq l \leq T2$$

$$\tau_t \geq 0 \forall t : 1 \leq t \leq T2 \tag{1.19}$$

$$r_1 = ST0, \sum_{t=1}^{T2} \tau_t \leq T \tag{1.20}$$

where the objective (1.9) is to maximize the quantity processed by the continuous task over the time horizon, constraints (1.10)-(1.11) impose that if a batch task starts at time slot t and finishes at time slot l (i.e., $z_{t,l} = 1$) then the sum of the time slot durations from t to l equals p. In particular, in (1.11), $\sum_{k=t}^{l} \tau_k \leq p$ if $z_{t,l} = 1$, and $\sum_{k=t}^{l} \tau_k \leq T$ otherwise because we do not have any bound on the time slots durations when $z_{t,l} = 0$. This constraint (1.11) is a so called big M constraint and usually leads to weak model formulations. Constraints (1.12)-(1.13) impose that at each time event, at most one batch task can start and one can finish. Two tasks starting (or finishing) at the same time are modeled as one task starting (finishing) at time slot t, the other starting (finishing) at time slot $t + 1$, and $\tau_t = 0$ ($\tau_{t+1} = 0$). In Figure 1.2, we represent how we model two tasks starting at the same time. So, when adding constraints (1.12)-(1.13), one needs to increase

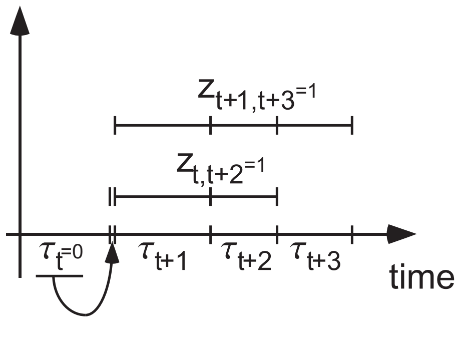

Figure 1.2: Two tasks starting at the same time

the size of the formulation (larger $T2$) in order to obtain an equivalent formulation because we limit to one the number of tasks starting or finishing at each time event. But we do not limit the number of tasks starting or finishing at the same time. However, as observed in practice, this new formulation is stronger, i.e. the duality gap is smaller, and allows one to strengthen the formulation of the timing constraints (see Chapter 2, and tests in Chapter 3). Constraint (1.14) imposes during each time slot t that the number of batch tasks processed in parallel is less than the number of processing units available. Constraint (1.15) and (1.16) are the balance constraints for the product in the storage tank at the beginning and at the end of the time slot t, constraint (1.17) limits the speed of the continuous task and constraint (1.18) restricts the storage tank level. Constraint (1.19) defines the z variables as binary and τ are positive continuous variables. Finally constraint (1.20) defines the specific conditions considered for this example, i.e. we impose an initial condition on the storage tank level, and the sum of time slots durations is less or equal than the scheduling time horizon.

In the literature, various algorithms are proposed for solving scheduling problems. We describe two important classes : the enumeration and the heuristic algorithms.

The enumeration algorithms are exact (i.e., they provide an optimal solution of the problems considered) but are usually not running in polynomial time. The drawback is that we cannot estimate beforehand the running time of the algorithm for a particular instance. Two well-known enumeration approaches for solving scheduling problems are the *branch-and-bound algorithm* and *constraint programming*. The branch-and-bound algorithm will be explained in Section 1.2.1. Constraint programming (CP) is a method for formulating and solving combinatorial problems. As opposed to the branch-and-bound method, CP focuses on the construction of a feasible solution to a scheduling problem. In order to formulate combinatorial problems with CP, one (or several) Constraint Satisfaction Problem(s) (CSP) is (are) defined. A CSP consists of a set of variables, a set of possible discrete values for each variable and a set of constraints on the set of variables. A solution of a CSP instance is an assignment of values to variables for which the set of constraints is satisfied. To solve a CSP, or to find a feasible solution, we can enumerate all possible combinations of values of variables. In order to reduce the enumeration, it is possible to use Constraint Propagation. The idea of Constraint Propagation is to use the constraints of the problem in an active way in order to reduce the domain of the variables. More information about Constraint Programming for solving scheduling problems can be found in Baptiste et al. [4].

The heuristic algorithms provide at best a feasible solution. The main advantage of these methods is that they can provide hopefully good solutions quickly. Whereas, the main drawback is that it is not possible to estimate how good the solution obtained is. The first type of heuristic algorithms are basic local search heuristics and improved local search heuristics (also called metaheuristics) such as Tabu Search, Simulated Annealing or Genetic algorithms, see Reeves [42] for a first introduction to such heuristic methods. For the basic local search heuristic, the idea is to define a neighborhood of solutions close to any given feasible solution, and an algorithm able to find the best solution in this neighborhood (which is easy if the neighborhood is of small size). Then, a first feasible solution is constructed, and the best solution in its neighborhood is sought. If the best feasible solution in the neighborhood is better than the initial feasible solution, then it replaces it and the procedure is repeated. Otherwise, we have found a local optimal solution that corresponds to the best solution in this neighborhood and this is the end of the heuristic. The improved local search heuristics try to do better. The idea is to escape from local optimal solution by accepting feasible solutions with worse objective value. The main difficulty of such methods is to prevent cycling.

There are also MIP-based heuristic algorithms that are based on the mixed integer programming formulation of the scheduling problem to be solved. The idea is to obtain a good feasible solution quickly, with some guarantee on the quality of the solution, see Section 1.4 for more information about such heuristic methods.

In this thesis, we focus only on MIP scheduling algorithms which are the branch-and-bound exact algorithm used to solve the MIP formulation of the scheduling problem, and the MIP-based heuristics. In the next Sections, we show how to solve such MIP problem formulations.

1.2 Mixed integer programming optimization

A mixed integer programming (MIP) model formulation is the minimization or the maximization of a linear objective function subject to linear constraints over a set of decision variables, such that a subset of it has to be integral. A general MIP problem can be written as

$$\begin{aligned} \max \quad & cx + hz && (1.21)\\ st \quad & Ax + Gz \leq b && (1.22)\\ & x \in R^n_+, z \in Z^q_+ && (1.23)\end{aligned}$$

where c and h are $n-$ and $q-$ dimensional row vector, A is a $m \times n$ matrix, G is a $m \times q$ matrix and b is a m-dimensional column vector. The variables z have to be integral.

In order to solve such a MIP problem, a classical approach is to use a branch-and-bound algorithm.

1.2.1 Branch-and-Bound algorithm

The branch-and-bound algorithm combines the concepts of branching and bounding in order to solve MIP problems.

In order to obtain an **upper bound** on the objective for a maximization MIP problem, we solve a linear programming (LP) relaxation of this problem, i.e. we relax continuously the integrality restriction, $z \in R^q_+$. We obtain first a solution (x^*, z^*) for this LP relaxation.

If the solution (x^*, z^*) is feasible for the MIP problem (i.e. z^* is integral) then it must be optimal for the MIP problem. This holds because the feasible set of the LP relaxation contains the feasible set of the corresponding MIP problem. So, if the best LP solution has integral values for the z variables, this solution must also be the best MIP solution. In this case, we can stop because the MIP problem is solved.

Otherwise, the solution of the LP is not feasible for the MIP problem (i.e., some z_j variable takes a fractional value z_j^*), and the optimal LP objective value

$cx^* + hz^*$ defines an upper bound on the optimal objective value for the MIP problem because we have solved a relaxation (larger feasible set) of the original MIP. Then, we **branch** on one fractional decision variable that is not integral, say z_i^* for some $i \in \{1, \ldots, q\}$, and we divide the problem into two subproblems (two nodes). For the first subproblem (Node 1), we impose the branching constraint $z_i \leq \lfloor z_i^* \rfloor$ and for the second subproblem (Node 2), we impose $z_i \geq \lfloor z_i^* \rfloor + 1$. Doing so, we replace the original problem by the union of the two subproblems, and we eliminate the infeasible solution (x^*, z^*) from the two subproblems.

Next, we select one of the two Nodes (suppose we select Node 1), we solve the LP relaxation of this subproblem (initial LP relaxation + one branching constraint) and we obtain an upper bound of the optimal MIP objective value over the current node. We can branch again on one fractional decision variable in the optimal solution to the LP relaxation of Node 1, and the subproblem is again divided in two subproblems (Nodes 3-4, replacing Node 1).

We have now to select a node in the set of Nodes {2,3,4}. We reoptimize the LP relaxation of this node, and so on.

The evolution of this algorithm is represented by an enumeration tree, where nodes are repeatedly selected and replaced by other nodes using bounding and branching. At each node processed, one of the following situations may occur :

- the optimal solution of the LP relaxation is a feasible solution for the MIP problem. In this case, we have obtained the best MIP solution for this subproblem (node) and we can stop processing that node (pruning the node by optimality).
- there are no feasible solutions for the LP relaxation. In this case, there are no MIP solutions for the subproblem at the current node (because MIP solutions are a subset of the LP solutions), and we can stop processing that node. (pruning by infeasibility)
- the optimal solution (x^*, z^*) of the LP relaxation is not feasible for the MIP problem, but the upper bound obtained ($cx^* + hz^*$) is lower than the value of the best MIP feasible solution found so far (at other nodes). In this case, we can stop processing this node because it will never produce better MIP solutions than the best found so far (pruning by bound).
- the optimal solution (x^*, z^*) of the LP relaxation of the node is fractional, but not dominated by bound. In this case, we select a variable z_j with a fractional value z_j^* and we branch (replace the current node by two new nodes).

The algorithm terminates when all nodes have been treated and pruned. The optimal solution of the MIP problem is the best solution among the MIP feasible solutions encountered during the enumeration of the nodes.

During the branch-and-bound algorithm, three main choices can affect the resolution of the problems :

1. The node selection
   At each step of the algorithm, the choice of the next node to visit will have an impact on the performance of the algorithm (number of nodes to process).

2. The generation of cuts
   At each node, we can improve the bound (decrease the upper bound in the case of a maximization problem) obtained by introducing cutting planes, i.e. new constraints in the formulation that are valid for all MIP solutions to (1.22)-(1.23). Doing so, the total number of nodes in the branch-and-bound tree and the CPU solution time will be reduced (see Section 1.3). A branch-and-bound algorithm in which cuts are used to improve the bounds is called a branch-and-cut algorithm.

3. The variable selection
   When a fractional LP solution is obtained, we have to decide on which fractional decision variable we branch on.

More information and details regarding branch-and-bound and branch-and-cut algorithms can be found for example in Nemhauser and Wolsey [33]. Several software systems (XPRESS-MP, CPLEX, . . .) are available for solving MIP problems by using branch-and-bound and branch-and-cut algorithms.

1.2.2 Solution of the basic example

By using the branch-and-bound algorithm, we solve an instance of the simple production process presented in Section 1.1 with the discrete and the continuous time formulations. This is a trivial example for pedagogy.

We suppose that there are 2 processing units and that the speed of the continuous task is in the interval $[\underline{\rho}; \overline{\rho}] = [0.5; 1.5] ru/h$. The capacity of the storage tank is $C = 15\ ru$. The processing time of the batch task is $p = 3$ hours (h), the batch size is $B = 8\ ru$ and the initial condition for the storage tank is $S0 = C$.

We consider first the discrete time formulation with $\Delta t = 1\ h$, $T1 = 40$, and $T = T1\Delta t = 40\ h$.

The formulation consists of 157 constraints, 80 binary variables and 120 continuous variables. The optimal solution obtained is $60ru$ and is obtained after 18 nodes and less than 1 second. In Figure 1.3 on the left, we represent when the batch tasks are in process by a solid line. We can observe that over the 40 time periods, only 6 time periods correspond to the start of a batch task. Observe also that, because of the storage tank capacity and maximum speed of the continuous task, the second production line never processes. So, we only need one processing unit for this optimal solution. Taking into account this information, we will still need 40 binary variables for the model with the discrete time formulation. In Figure 1.3 on the right, we represent the evolution over time of the level of the intermediate product in the storage tank.

The solution of this instance is not unique. All other feasible solutions for which the continuous task can be processed at maximal speed ($q_t = 1.5\ \forall t \in \{1, \dots, T1\}$) are also optimal for this problem instance. The multiple possible optimal solutions for this problem instance shows that symmetry breaking would be necessary.

To model this problem instance with the continuous time formulation, we consider $T2 = 9$ and we assume that the batch task can last for at most one time slot. Therefore, the z binary variables considered here are only the $z_{t,t}$ variables for $t \in \{1, \dots, 9\}$. The formulation consists of only 9 binary variables, 36 continuous

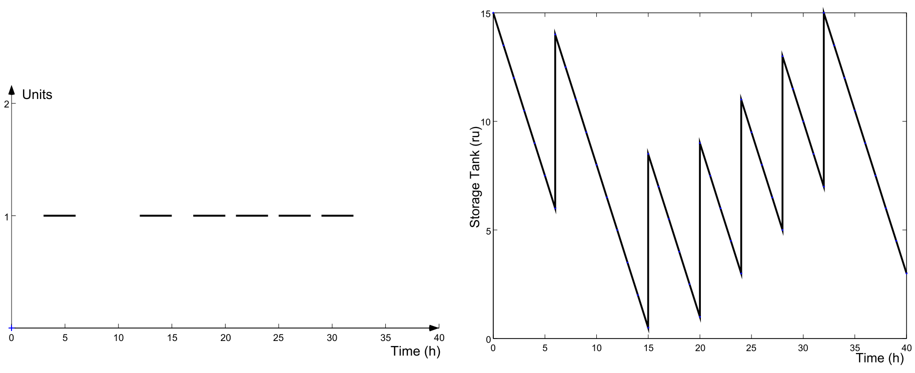

Figure 1.3: Solution of a special case with the discrete time formulation

variables and 54 constraints. The optimal objective value is 60 ru and is equal to the one obtained when using the discrete time formulation. This optimal solution is also obtained after 18 nodes and less than 1 second. In Figure 1.4 on the left, we represent the batch tasks in process by a solid horizontal line. The idle times are represented by the horizontal dashed lines. The solid line between two bullets correspond to the batch task that is performed during this time interval. In Figure 1.4 on the right, we represent the evolution of the level of the intermediate product in the storage tank over time. This optimal solution is equivalent in terms of total production (objective function) to the one obtained by using the discrete time formulation.

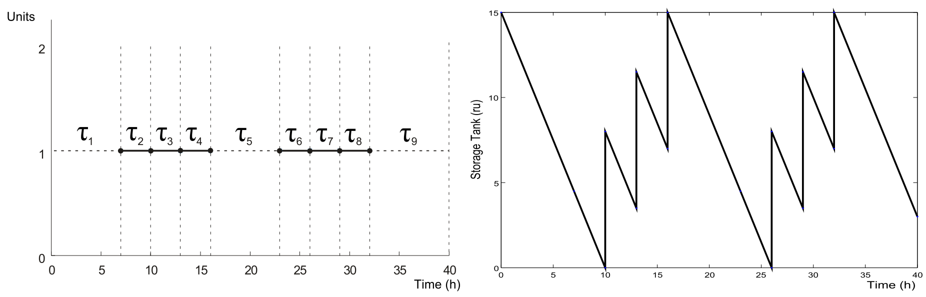

Figure 1.4: Solution of the special case with the continuous time formulation

We can conclude that by using continuous time formulations, we can reduce the size of model formulations. However, continuous time formulations are compact but weak in the sense that typically such model formulations need many (or more) branch-and-bound nodes to prove optimality for large or real life instances.

In the next Section, we discuss how to improve a model formulation in order to make it stronger or tighter. This will be a central topic in this thesis.

1.3 Improving MIP formulations

In this section, we analyze the quality of a model formulation. We explain how to measure it and how to improve a model formulation. A good model formulation is crucial in order to solve MIP optimization problems, because good formulations allow one to reduce the number of branch-and-bound nodes needed to solve the MIP (there are as many LPs to solve as the number of nodes to explore). Note that the scheduling formulations we are considering need to be improved, partly because of big M constraints like (1.11). When using such constraints, the integer variables are often fractional in the optimal LP relaxation solution, and a lot of nodes need to be enumerated before pruning.

One possible way to avoid weak formulations with big M constraints is to get rid of or to avoid them by decomposing the problem. This can be done for scheduling problems by decomposing the problem into an assignment master problem and a sequencing subproblem. In the paper by Maravelias and Grossmann [28], for example, the assignment of production units to tasks is done in the master problem and modeled by a mixed integer linear programming (MIP) formulation, and the sequencing subproblem, to be solved separately for each production unit, is modeled and solved by constraint programming (CP). Another related decomposition approach is proposed by Maravelias in [27]. The assignment master problem is also modeled as a mixed integer linear program, but the feasibility check in the subproblem, and the construction of feasible schedules for each production unit, are performed by combinatorial sequencing algorithms.

Another way is to improve directly the model formulations with big M constraints by tightening the formulation. In general, to tighten a formulation , we can use strong or facet defining valid inequalities for the problem studied (see Nemhauser and Wolsey [33]). This can be done either in the original variable space, or in some larger (extended) space of variables (see Pochet and Wolsey [38]). In these approaches, one tries to obtain a better formulation by adding "tighter" valid constraints (and possibly new variables) to the formulation. In contrast, the strengthening techniques try to improve the original constraints of a formulation by changing the coefficients of some variables (see Andersen and Pochet [1]).

In the next sections, we give more details about the cutting plane algorithm, used to add valid inequalities to the formulation, and strengthening techniques used to improve the initial constraints (without adding new ones).

1.3.1 Cutting plane algorithm

We consider the general MIP problem (1.21)-(1.23) written as follows :

$$\max \quad cx + hz \tag{1.24}$$
$$st \quad (x, z) \in X \tag{1.25}$$

where $X = \{(x, z) : Ax + Gz \leq b,\, x \in R^n_+, z \in Z^q_+\}$.

It is possible to find alternative formulations for a problem. We explain here that some formulations are better than others. First, we need two definitions.

Definition 1.3.1. *A polyhedron P is a subset of R^n and is described by a finite set of linear constraints $\{x \in R^n : Ax \leq b\}$.*

Definition 1.3.2. *A polyhedron $P \subseteq R^{n+q}$ is a formulation for $X \subseteq R^n \times Z^q$ if and only if $X = P \cap (R^n \times Z^q)$.*

Various polyhedra can define formulations for the same mixed integer set X. However, an interesting polyhedron (and formulation) is the convex hull of the set of feasible points of X.

Definition 1.3.3. *The convex hull of X (conv(X)) is defined as : conv(X) = $\{(x, z) : (x, z) = \sum_{i=1}^{t} \lambda_i (x_i, z_i),\, \sum_{i=1}^{t} \lambda_i = 1,\, \lambda_i \geq 0$ for $i = 1, \ldots, t$ and for all finite subsets $\{(x_1, z_1), \ldots, (x_t, z_t)\} \subseteq X\}$.*

For a MIP with rational data, conv(X) is a polyhedron, i.e., conv(X) can be defined by a finite set of linear inequalities. So, suppose that conv(X)=$\tilde{X} = \{(x, z) : \tilde{A}x + \tilde{G}z \leq b, x \in R^n_+, z \in R^q_+\}$.

The MIP problem (1.24)-(1.25) can be reformulated equivalently as

$$\max \quad cx + hz \tag{1.26}$$
$$st \quad \tilde{A}x + \tilde{G}z \leq b, x \in R^n_+, z \in Z^q_+ \tag{1.27}$$

This means that the linear description of conv(X) defines an alternative equivalent formulation for the MIP problem (1.24)-(1.25). The interest of using this latter formulation is that the linear programming relaxation of (1.26)-(1.27) (obtained by replacing (1.27) by $(x, z) \in \tilde{X}$, i.e. $z \in R^q_+$) has extreme points that belong to X. Consequently, for each c and each h, there exists an optimal solution of this LP relaxation which is feasible for the corresponding MIP problem, and therefore is the optimal solution of the MIP. So, the MIP problem can be solved as an LP, without any branching.

This is a theoretical but not practical result because, in general, it is not possible to compute the complete linear description of the convex hull in a reasonable amount of time. Based on this result, the idea for improving an initial model formulation is to find a formulation whose LP relaxation better approximates or comes closer to the convex hull of solutions of the problem. If we have two formulations for X (say P_1, P_2), since $X \subseteq conv(X) \subseteq P_i$ for $i \in \{1, 2\}$, we say that P_1 is a better formulation than P_2 if $P_1 \subset P_2$.

As an illustration, in Figure 1.5, we consider a problem composed of two variables (x (a continuous variable) and z (an integer variable)). The feasible mixed integer solutions are represented by the three horizontal normal lines and the corresponding feasible set is denoted by X. We suppose that we have two formulations for the problem, P_1 (bold lines) and P_2 (dashed lines). One can observe that the feasible set P_2 is included in the feasible set P_1 and therefore the formulation P_2

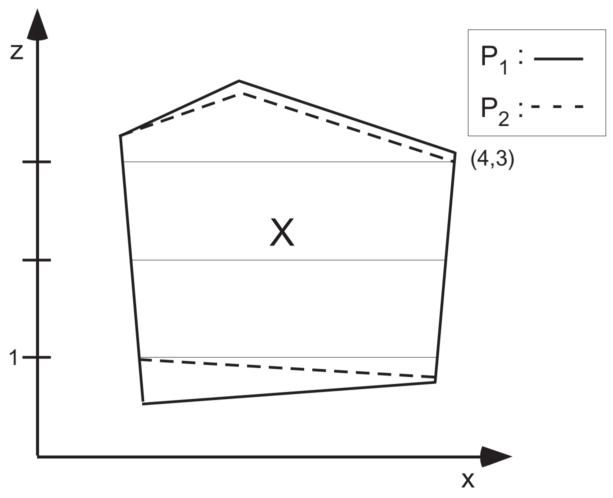

Figure 1.5: Comparison of two formulations

is better than P_1. Geometrically, we can observe that the formulation P_2 is better than the formulation P_1 since it is closer to the convex hull of the set X of feasible solutions.

We can improve a formulation P to get a formulation that is closer to the convex hull of X, by using the notion of valid inequality.

Definition 1.3.4. *An inequality $\pi_1 x + \pi_2 z \leq \pi_0$ is a valid inequality for $X \subseteq R^{n+q}$ if $\pi_1 x + \pi_2 z \leq \pi_0$ for all $(x, z) \in X$.*

Valid inequalities can be added to any formulation P, to obtain a better formulation. So, we want to compute the best possible valid inequalities. We give below four definitions and then we explain which valid inequalities are important to improve formulations and to approximate the convex hull of X.

Definition 1.3.5. *The face F of X induced by a valid inequality $\pi_1 x + \pi_2 z \leq \pi_0$ of X is $F = \{(x, z) \in X : \pi_1 x + \pi_2 z = \pi_0\}$.*

Definition 1.3.6. *$x^1, \ldots, x^n \in R^{T+Td}$ are affinely independent if $\sum_{i=1}^n \alpha^i x^i = 0$ and $\sum_{i=1}^n \alpha^i = 0$ implies $\alpha^i = 0$ for $i = 1, \ldots, n$, or equivalently if the directions $x^2 - x^1, \ldots, x^n - x^1$ are linearly independent.*

Definition 1.3.7. *The dimension of X ($dim(X)$) is equal to the maximum number of affinely independent points in X minus 1.*

Definition 1.3.8. *A face F of X is a facet of X if dim(F)=dim(X)*-1.

For any set X, it is possible to show that a valid inequality $\pi_1 x + \pi_2 z \leq \pi_0$ is needed for the linear description of the polyhedron conv(X), and therefore important in the description of X, if and only if this valid inequality defines a facet of X. In other words, the linear description of conv(X) is made of all facet defining valid inequalities for X plus the linear equalities that are satisfied by all feasible solutions.

The next question to address is how to generate all facet defining valid inequalities for a given set X. For a given problem instance (with fixed data), by using the POlyhedron Representation Transformation Algorithm (PORTA, see Christof and Loebel [16]), starting from the extreme points and the extreme rays of the problem instance, we can generate the complete linear inequality representation of the convex hull of the instance. The algorithm implemented in PORTA in order to compute this, is the Fourier-Motzkin elimination algorithm.

In Figure 1.6-1.7, we show the relation between the extreme points of a set X and its convex hull, for a problem instance with the two variables x and z. The polyhedron ($\text{conv}(X)$) is bounded and the convex hull of the feasible solutions can be implicitly represented by its extreme points (large dots) (see Figure 1.6) or explicitly by linear inequalities (see Figure 1.7).

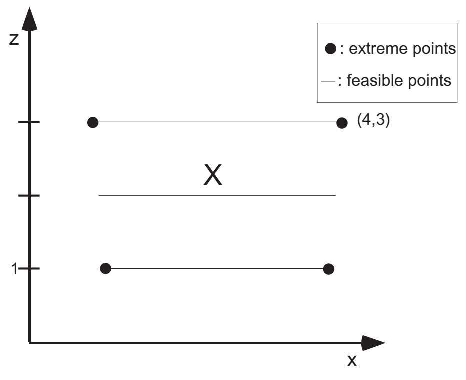

Figure 1.6: Convex hull : extreme points

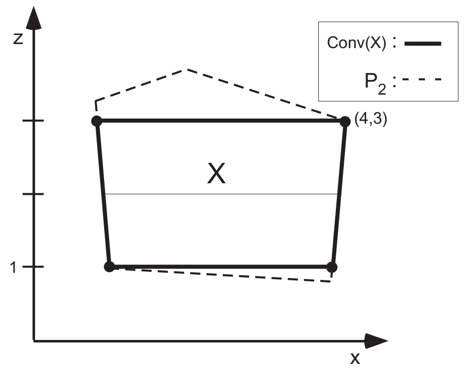

Figure 1.7: Convex hull : linear inequalities

Although PORTA can be used for any instance X, it is very time consuming and can only be used for small polyhedra. Moreover, PORTA provides only the linear description of the convex hull of a problem instance with fixed data (matrix coefficients and right-hand-side coefficients), but not the complete linear description of all instances in a problem class. Therefore, PORTA can be used to analyze small instances, and its results have to be analyzed and generalized to suggest (facet defining) valid inequalities for all instances of a problem.

There are other methods to identify valid inequalities for a MIP problem X. Pochet and Wolsey in [38] classify general valid inequalities in two classes. First, there are so called high level valid inequalities based on the complete formulation of a problem, or based on the structure of some relaxations of it. There are also low level valid inequalities based only on some more local information in the formulation (like knapsack or Gomory cuts derived from a single constraint in the original formulation). These low level constraints are generated automatically by MIP solvers by using the coefficient matrix of the constraints. But such constraints do not exploit the global structure of the formulation, and therefore the generation of high level valid inequalities remains a problem specific task that is not performed automatically. This will be done in this thesis for some specific scheduling formulations.

When one or several classes of valid inequalities have been identified for a problem X, the final question to address is how to add these valid inequalities to the original formulation, in order to improve or to tighten it. Very often, the number of valid inequalities grows exponentially in the problem size. It is therefore not practical to add all these valid inequalities a priori (i.e., before the optimization starts) to the formulation.

For a specific problem instance and objective function, we are not interested in computing the whole convex hull. We want to have a good approximation of it in the neighborhood of the optimal solution. For instance, in Figure 1.7, if the optimal solution is $(x, z) = (4, 3)$, then we only need to add $z \leq 3$ to the original formulation in order to solve the integer program as an LP (i.e., by branch-and-bound, without any branching). The aim of cutting plane algorithm is to generate such valid inequalities during the optimization, and only when they are needed.

The branch-and-bound and cutting plane algorithm works as follows. We start by solving the LP relaxation of the initial formulation. At the root branch-and-bound node, if the solution (x^*, z^*) of the linear relaxation is fractional ($z_j^* \notin Z$ for some j), we solve a so-called separation problem where we try to find a valid inequality $\pi_1 x + \pi_2 z \leq \pi_0$ that is violated (not satisified, i.e. $\pi_1 x^* + \pi_2 z^* > \pi_0$) by the solution (x^*, z^*) of the relaxation. If we find such a valid inequality, we can cut off the optimal solution of the linear relaxation by adding the valid inequality $\pi_1 x + \pi_2 z \leq \pi_0$ to the initial formulation. The inequality $\pi_1 x + \pi_2 z \leq \pi_0$ is called a cutting plane, or simply a cut. By resolving this augmented linear relaxation, we obtain a solution closer to the convex hull of the problem, for which we can again try to identify a cut. The cutting plane algorithm is iterated until no violated valid inequality can be found.

If the final solution (x^*, z^*) is still fractional, we can continue with the classical branch-and-bound algorithm. This combined branch-and-bound and cutting plane algorithm is also called *cut-and-branch*.

The variant where cuts (violated valid inequalities) are added at all nodes of the branch-and-bound tree (and not just the root node) is called *branch-and-cut* (see for example for more details Wolsey [49] and Pochet and Wolsey [38]).

We present in the next section another way to improve model formulations, the so-called strengthening technique. The main difference with respect to the cutting plane technique is that one tries to improve the initial constraints of the formulation by changing some coefficients of the variables, but not by adding new constraints.

1.3.2 Strengthening techniques

Andersen and Pochet [1] propose to analyze the quality of MIP formulations by considering whether or not some of the coefficients in the constraints can be strengthened. The idea is to create a tighter model formulation P' (i.e., closer to the convex hull of the solutions of the problem than the initial model formulation P, i.e. $P' \subset P$) without increasing the number of constraints.

They describe and show how to solve the optimization problem corresponding to this strengthening problem, where the objective is to strengthen the coefficients as much as possible. It is a procedure that tightens iteratively the coefficients of integer variables in existing constraints.

We introduce now the general concept of coefficient strengthening in existing constraints. Suppose that we want to improve the inequality $a_k^T x + g_k^T z \leq b_k$, where a_k^T and g_k^T are the k^{th} line of the matrices A and G, defining the initial formulation $Ax + Gz \leq b$, respectively. We select another constraint of the matrix, for example the l^{th} one, $a_l^T x + g_l^T z \leq b_l$. The idea is to find $w_{k,l} > 0$ such that

$$(a_k^T x + g_k^T z) - w_{k,l}(a_l^T x + g_l^T z - b_l) \leq b_k \tag{1.28}$$

is valid for X. This new inequality is stronger than $a_k^T x + g_k^T z \leq b_k$ because $a_l^T x + g_l^T z - b_l \leq 0$ for all feasible solutions $(x, z) \in X$, and therefore (1.28) implies that $a_k^T x + g_k^T z \leq b_k$ is also satisfied.

To find the best or strongest inequality (1.28), one has to solve the following strengthening problem

$$w_{k,l}^* := \max\{w_{k,l} : (1.28) \text{ is valid for } \bar{X}\}, \tag{1.29}$$

where $X \subseteq \bar{X} \subseteq P = \{(x, z) : Ax + Gz \leq b, x \in R_+^n, z \in R_+^q\}$.

The formulation is strengthened by solving a sequence of strengthening problems for various relaxations $\bar{X}$ of X (in the simplest case, $\bar{X} = P$), and for various combinations of constraints k and l. A similar strengthening problem can be defined for improving the right-hand-side b_k of any constraint.

Andersen and Pochet show how to model this general strengthening problem. In particular, they focus on strengthening the coefficients of binary variables. Suppose z_l is a binary variable and we want to introduce this variable in the constraint $a_k^T x + g_k^T z \leq b_k$. We have to find the best δ for which the constraint

$a_k^T x + g_k^T z + \delta z_l \leq b_k$ remains valid for X. We only have to consider the case when $z_l = 1$, since otherwise there is no condition on δ.

In this particular case, they show that the strengthening problem reduces to the MIP problem

$$\begin{aligned} & \max a_k^T x + g_k^T z - b_k \\ st \quad & (x, z) \in X \\ & z_l = 1 \end{aligned}$$

If (x^*, z^*) is the optimal solution of this MIP problem, then $\delta^* = b_k - (a_k^T x^* + g_k^T z^*)$. If $\delta^* > 0$, then $a_k^T x + g_k^T z + \delta^* z_l \leq b_k$ is stronger than $a_k^T x + g_k^T z \leq b_k$, and we have strengthened the constraint. This problem is solved sequentially for every binary variable in every constraint k.

In Figure 1.8, we illustrate the effect of coefficient strengthening on a problem with the two variables x (continuous variable) and z (integer variable), and initial formulation P. We want to improve the inequality $I1 := -1/3x + z \leq 3$. We select another constraint $I2 := -x \leq 0$. It is possible to show that the optimal solution of the problem (1.29) for $\bar{X}$=X is $w_{I1,I2} = 1/3$. The new inequality is $I3 := (-1/3x + z) - 1/3(-x - 0) \leq 3$, or equivalently $z \leq 3$ and is represented in solid line. We can observe that the improved formulation becomes closer to the

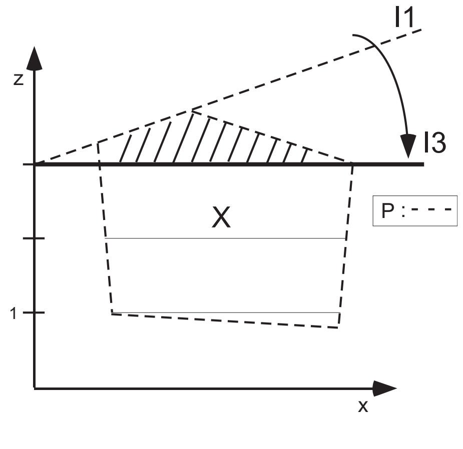

Figure 1.8: Coefficient Strengthening

convex hull of X and is strictly included in P.

Taking into account the fact that the computation of the strengthening coefficients is heavy (one has to solve a MIP over X to improve each single coefficient), we will use this coefficient strengthening technique as a tool in order to produce tight model formulations in the following way. First, we start with some small instances of the problem to reformulate. We strengthen the coefficients as much as possible. Then, we analyze the improved coefficients for the small instances and try to generalize the improvements obtained in order to make them valid for all instances of the problem.

In the next section, we illustrate how it is possible to improve the model formulation of the special case of the general problem presented in Section 1.1.

1.3.3 Application to the basic example

We propose valid and strengthened inequalities for the special case presented in Section 1.1 and tested in Section 1.2.2. All the corresponding proofs are presented in Chapter 2.

In order to compare the quality of the initial and the strengthened continuous time formulations, we consider a larger instance than the one tested in Section 1.2.2, with the following parameters : $T2 = 40$, $nbr\_unit = 4$, $\underline{\rho} = 5\ ru/h$ and $\overline{\rho} = 8\ ru/h$. We assume that the batch task can last for at most 10 time slots. Therefore, the z binary variables considered here are only the $z_{t,t},\ z_{t,t+1},\ldots,\ z_{t,\min(t+9,40)}$ variables for $t \in \{1,\ldots,40\}$.

The initial continuous time formulation is described by (1.9)-(1.20). We describe now a valid inequality, as well as strengthened versions of inequalities (1.10)-(1.11). The variables are not affected by the reformulations.

The batch task can be performed on each reactor. The valid inequality (1.30) below takes into account the fact that the time needed to complete all the tasks is greater or equal to the time needed if the batch tasks and the total processing time are evenly allocated among all the nbr_unit reactors.

The new (high level) valid inequality expressing this observation can be written as

$$\frac{1}{nbr\_unit}\left(\sum_{t=1}^{T2}\sum_{t'=t}^{T2} p z_{t,t'}\right) \leq \sum_{t=1}^{T2} \tau_t. \tag{1.30}$$

We present now two strengthened inequalities of (1.10) and (1.11) for this special case. The analysis of these inequalities and proofs of validity will be given in Chapter 2. The main reason of this strengthening is that at most one batch task can start (see (1.12)) and at most one can finish (see (1.13)) at each time slot. We also impose without loss of generality for this strengthened formulation that the duration of a time slot is limited by the maximum processing time of the batch task, i.e., $\tau_t \leq p$ for all t.

The initial timing constraint (1.10) can be strengthened to

$$\sum_{k=t}^{l} \tau_k \geq \sum_{k=t|t\neq l}^{l} p z_{t,k} + p z_{l,l} \tag{1.31}$$

for all $t, l : 1 \leq t \leq T2, t \leq l \leq T2$.

Similarly, the timing constraint (1.11) can be strengthened to

$$\sum_{k=t}^{l} \tau_k + p\sum_{k=1}^{l-1} \min\{l-t, l-k\} z_{k,l} \leq p(l-t+1) \tag{1.32}$$

for all $t, l : 1 \leq t \leq T2, t \leq l \leq T2$.

The problem instance is first solved with the initial continuous time formulation and without adding any cut during the branch-and-bound algorithm. Then, we solve the problem with the valid and strengthened inequalities presented above, and with low level cuts generated automatically by the MIP solver. We call it the improved (continuous time) formulation. The results obtained are presented in Table 1.1.

T2=40, d=10		
nbr unit=4	Initial formulation	Improved formulation
Constraints	948	949
Nodes	37512	8134
Time	81 s	49 s
Objective	311 ru	311 ru

Table 1.1: The improvement of a model formulation

There is only one optimal solution for the quantity processed by the continuous task for this instance. The optimal solution is the following : $q_t = 5$ for $t \in \{1, 2, 3\}$, $q_t = 8$ for $t \in \{4, \ldots, 40\}$ and $\tau_t = 1$ for $t \in \{1, \ldots, 40\}$. The evolution of the level of the intermediate product in the storage tank is represented in Figure 1.9.

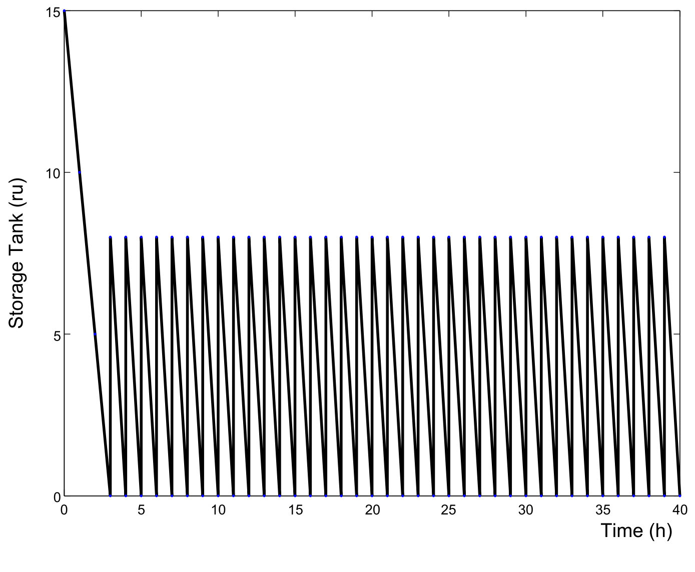

Figure 1.9: The level of the intermediate product

On this small example, we can already observe the effects of taking into account valid and strengthened inequalities. The total number of branch-and-bound nodes for solving this problem to optimality is reduced as well as the CPU time.

In Figure 1.10, we observe first that the LP relaxation of the improved formulation (316.33 ru) provides a better upper bound for the MIP problem than the one provided by the initial formulation (320 ru). Then, by using the low level cuts generated automatically by the MIP solver for the initial formulation, we get as upper bound 314.27 ru. In order to obtain this upper bound, 391 knapsack cuts and 145 Gomory cuts were generated and 47 cuts were kept in the formulation after processing the top node. The others were generated during the cutting plane phase but were removed from the final formulation obtained at the root node because they become inactive. By using the low level cuts generated automatically by the MIP solver for the improved formulation, the value of the upper bound at the root node (314.25 ru) is improved furthermore. For the improved formulation, 131 knapsack cuts and 122 Gomory cuts were generated and 35 cuts were kept in the formulation at the top node.

By improving the upper bound at the root node, one can hope to reduce the number of nodes in the branch-and-bound tree. However, a trade-off exists since if many cuts or dense cuts (i.e., cuts with many non zero coefficients) are added to the formulation then the resolution at each node can be much slower than before.

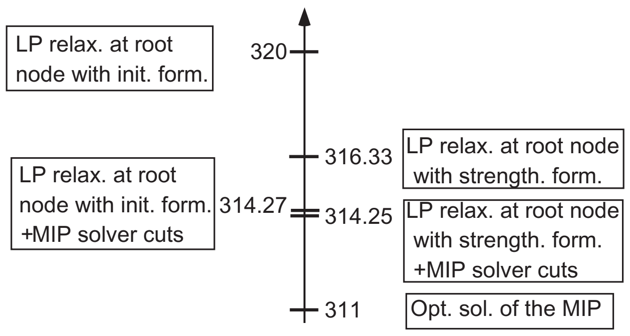
LP relax. at root node with init. form.
320
316.33
LP relax. at root node with strength. form.
LP relax. at root node with init. form. +MIP solver cuts
314.27
314.25
LP relax. at root node with strength. form. +MIP solver cuts
311
Opt. sol. of the MIP

Figure 1.10: Representation of the solution of the LP relaxation and of the MIP

1.4 MIP heuristics

Even if we use an improved model formulation, the time needed to solve some larger instances to optimality remains very long. For the scheduling problems tackled in this thesis, this difficulty arises because of the size of the problems and because it seems difficult for the MIP solver to find feasible solutions even though the duality gap (=LP relaxation objective value-MIP objective value) is small. Therefore, we consider MIP heuristic techniques in order to obtain good feasible solutions quickly. MIP heuristics are heuristics based on MIP formulations. We have tried to use such heuristics in order to take advantage of the improved formulations obtained. In other words, their performance usually improves when they are applied on improved formulations of a problem. The MIP heuristic methods can be subdivided into two groups. The first type are the construction heuristic methods that construct a feasible solution from scratch (Truncated MIP, LP-and-

Fix or Cut-and-Fix, Relax-and-Fix (see Stadtler [46]), ...), and the second type are the improvement heuristic methods that try to improve some initial feasible solution (Relaxation Induced Neighborhood Search (RINS) (see Danna et al. [18]), Local Branching (LB) (see Fischetti and Lodi [20]), Exchange (EXCH)). For more details about these heuristic methods, see for instance Pochet and Wolsey [38].

As an illustration, we use first the construction heuristic called truncated MIP on the problem instance solved in the previous section. It consists in solving the problem by branch-and-bound for a fixed CPU time. The best solution obtained at the end is the solution of the heuristic.

For the special case instance presented above, we use the improved formulation and we solve the problem by branch-and bound during 6 *sec.* and the truncated MIP heuristic provides a feasible solution (a lower bound on the optimal objective value) with an objective function of 231 *ru*. The best upper bound obtained after 6 *s* is 314.25 *ru*. This upper bound is the maximum LP relaxation objective value among all branch-and-bound nodes that are still active (not pruned) after 6 *sec.* The remaining duality gap ($\frac{BestBound-BestSolution}{BestBound}$) is of 26.5 % and is a measure of the quality of the feasible solution obtained.

Then, we test the same construction heuristic on the initial formulation. We solve the problem by branch-and-bound during 13 *sec.* and the truncated MIP heuristic provides a first feasible solution with an objective function of 207 *ru*. The best upper bound obtained after 13 *s* is 317.85 *ru*. The remaining duality gap is of 34.87 %. This feasible solution is worse than the one obtained when using the improved formulation and needs more CPU time to be computed.

The combination of formulation strengthening and MIP-based heuristic methods seems important.

We continue with the improved formulation and we look at the improvement heuristics. We propose to use Local Branching (LB). Based on an initial feasible solution (x^*, z^*), the aim of this heuristic is to look for a better solution in the neighborhood of the current solution. The neighborhood is defined by allowing at most k binary variables to change their value with respect to the currently best solution. This condition can be imposed by adding the following linear constraint to the problem formulation :

$$\sum_{(t,l):1\leq t\leq T2, t\leq l\leq T2 | z^{\star}_{t,l}=0} z_{t,l} + \sum_{(t,l):1\leq t\leq T2, t\leq l\leq T2 | z^{\star}_{t,l}=1} (1 - z_{t,l}) \leq k.$$

This heuristic can be applied iteratively in order to improve the objective function further.

Other related neighborhoods could be defined. For instance, another neighborhood is defined by allowing at most k binary variables to change their values from 1 to 0 with respect to the currently best solution. This neighborhood was not tested in this thesis.

Based on the initial feasible solution obtained in less than 6 *sec.* by the truncated MIP with an objective function of 231 *ru*, we use the Local Branching heuristic. We allow at most $k = 5$ binary variables to change their value. We apply the Local Branching heuristic and in 2 *sec.*, a better feasible solution is obtained with an objective of 255 *ru*. The total CPU time needed when combining the two heuristics is 8 *sec.* The same feasible solution cannot be obtained by the branch-and-bound algorithm in less than 35 *sec.* This illustrates that such a combination of heuristics can be important in practice in order to obtain quickly a better feasible solution than the one obtain by the truncated branch-and-bound algorithm for the same amount of CPU time.

1.5 Outline and contribution of the thesis

The aim of this thesis is to solve scheduling problems for mixed production lines, i.e., involving batch and continuous processes, and some resource restrictions. Discrete time formulations were initially proposed in the literature in order to solve such problems, but their drawback is the large size of formulations for solving real industrial cases. This is why continuous time formulations were then proposed. To the best of our knowledge, in the literature, there is no study about the strengthening of the continuous time formulations used to solve scheduling problems. Various authors did compare different types of continuous time formulations, but did not try to improve or tighten such formulations.

In Chapter 2, we study a specific continuous time formulation for the cyclic scheduling problem of a mixed plant, where we suppose that all the processing times of batch processes are known and fixed before the scheduling phase. For a subcase of the general cyclic scheduling problem composed of one batch process and no resource restriction, we propose first a strengthened formulation, prove its validity and prove that our strengthened continuous time formulation has no duality or integrality gap. We also prove that some of the strengthened inequalities found are facet defining for this subcase. Moreover, when the number of processing units is limited, we prove by adding one valid inequality that the integrality gap remains 0. We then extend the tightened formulation to make it valid for multiple batch processes and, by using strengthening techniques, we show how to additionally improve the model formulation. Finally, we extend the tightened formulation obtained for these subcases in order to model and optimize the general cyclic scheduling problem composed of batch and continuous processes and again we show, by using strengthening techniques, how to additionally tighten the basic constraints limiting the processing rate of the continuous processes.

In Chapter 3, by using the tightened formulations proposed in Chapter 2 for various subcases of the general problem, and for the general one, we show that we can solve problem instances quicker than with the initial formulation. For the large instances that we cannot solve to optimality, we propose and test a number of MIP based heuristic methods. These heuristics find feasible solutions quicker than exact solution methods. MIP heuristic methods are based on model formulations and we show by computational experiments that they also take advantage of the improvements obtained by tightening the continuous time formulations. Finally, we show that one of the MIP heuristic method can provide for an industrial case a better feasible solution than truncated exact branch-and-bound methods. Therefore we show that the combination of formulation strengthening and MIP-based heuristic methods seem to be important in order to get good solutions quickly for such industrial problems.

Chapter 3 except Section 3.1, and Chapter 2 except Section 2.2 and 2.3 have been accepted for publication in the Special issue of Computers & Chemical Engineering on Enterprise-wide optimization [37]. Moreover, this paper has been first presented in November 2006 at the annual meeting of the American Institute of Chemical Engineers (AIChE) in San Francisco, California and also in August 2007 at the third Multidisciplinary International Scheduling Conference : Theory and Applications (MISTA) in Paris, France.

In contrast to the earlier models, in Chapter 4, we consider a scheduling problem in which we model the dynamics of the process. The processing times of the batch tasks are therefore considered to be variable in this case. These processing times are determined as the solution of the system of differential equations describing the process dynamics, and influenced by process parameters that have to be optimized. Here we consider a single process step, and our objective is to minimize the time needed to perform this step by optimizing the process parameters. Two classical time discretization approaches (the trapezoidal method and the collocation method) are proposed and tested in order to model the dynamics of such a process and to schedule it. The corresponding model formulation is composed of non linear equations and we solve this problem by sequential mixed integer programming (SMILP). We also model the dynamics of such a process by two piecewise linear approximation methods. In these models, both the time and the state space of the system are discretized. The idea of the first piecewise linear approach is to decompose the space of variables involved in the process dynamics into a set of hypercubes and in every hypercube, we approximate the solutions of the system of nonlinear differential equations by a linear approximation around the point in the middle of the hypercube. For the second piecewise linear approximation, we decompose the space of variables involved in the process dynamics into a set of simplices. Every point in the state space can be expressed as a convex combination of extreme points of a simplex. The values of nonlinear functions describing the process dynamics are then linearized by taking the corresponding convex combination of the values of the nonlinear functions at the extreme points of the simplex. This piecewise linear approximation is then integrated by using the euler explicit method. We show that, by using the classical approaches (trapezoidal rule and the collocation method), the optimal solutions obtained for the optimization problem approximate very well the dynamics of the process, but are obtained slower than with the two piecewise linear formulations proposed. We show that the second piecewise linear approximation can give an optimal solution quicker than the other methods with a good approximation of the process dynamics.

Chapter 4 except Section 4.3 was presented in March 2005 at the third FNRS

Cycle in Mathematical Programming Seminar in Han-sur-Lesse, Belgium.

Finally, in Chapter 5, we conclude and propose some future works.

Chapter 2

Continuous time formulation for the cyclic schedule optimization

The problem that we address in this chapter is the optimization of the cyclic schedule of a general mixed plant in order to maximize its productivity. A mixed plant is composed of batch and continuous tasks. A batch task has a fixed processing time and produces at the end a fixed amount of product. A continuous task is processed continuously and its decision variable is the speed or processing rate at which this continuous task is performed. Both types of tasks consume resources with limited capacity or availability.

In order to solve such a scheduling problem, continuous time formulations were proposed in the literature. The characteristic of continuous time formulation is that the time is discretized in a finite number of time slots of variable duration that are decision variables of the optimization problem. Therefore, time is decomposed into a set of consecutive time slots of variable duration. The end of a time slot corresponds to an event where the status of the process is changed (start or end of a batch task, modification of the speed of a continuous task). The execution of each batch task is assigned to a set of consecutive time slots, whose global duration is equal to the processing time of the batch.

Schilling and Pantelides in [44] propose a mixed integer non linear programming (MINLP) formulation of this scheduling problem. They solve the problem by developing a special Branch and Bound algorithm that branches on both continuous and discrete variables. Then, Castro et al. [12] propose a continuous time formulation where the sum of the durations of the consecutive time slots to which a batch task is assigned can be greater or equal to the processing time of the batch task. This relaxation, in which the end of a batch task is not modeled explicitly, allows one to obtain better results (reduced running times thanks to a tighter formulation), but does not allow to model cases with zero waiting time constraints between successive processing stages. They also show that a discrete time formulation, i.e., a formulation where the time is discretized in a larger but finite number of time slots of fixed duration, seems more appropriate to solve their case study. Finally, Wu and Ierapetritou [50] propose a continuous time formulation with a reduced number of binary variables obtained by using a time representation (decomposition into time slots) that is specific for each processing unit or reactor. Unfortunately, this model does not allow to treat cases where resources are shared by the different processing units.

The formulation used in this chapter is similar to the one proposed by Schilling and Pantelides [44] in order to be able to model shared resources and some zero waiting time constraints between successive process stages. Therefore we cannot take advantage of the continuous time formulations proposed by Castro et al. [12] or Wu and Ierapetritou [50]. The number of binary variables corresponding to such continuous time formulations is drastically reduced in comparison with the corresponding discrete time formulation. The drawback is that such continuous time formulations are usually weak because of the introduction of so called big M types of constraints in order to build a correct model formulation. The tightness of the continuous time formulation has to be improved in order to solve larger instances. To the best of our knowledge, there are no study of the strength of those continuous time formulations, or results showing how to strengthen them, published in the literature.

In this chapter, we study the quality and strengthen the mathematical programming formulation of three special cases of a batch-plant and mixed-plant scheduling problem.

In the first special case, we maximize (a measure of) the productivity of a plant performing only one batch task with (Capacitated case) and without (Uncapacitated case) restrictions on the number of processing units available to perform this batch task. For the uncapacitated case, we prove that some of the strengthened inequalities are facet defining and that the duality gap of the strengthened formulation is zero. By adding a valid inequality for the capacitated case, we also prove that the duality gap is zero in that case.

In the second special case, we maximize (a measure of) the productivity of a batch plant performing multiple batch tasks. We show how to extend the strengthened formulation obtained in the first special case and how to additionally tighten some inequalities.

In the third special case, we maximize (a measure of) the productivity of a plant performing both batch and continuous tasks. We derive some new valid inequalities.

The outline of the chapter is the following. In section 2.1, we describe the general scheduling problem addressed in this chapter and we give the definitions of the sets and the variables used in our formulations. In section 2.3, we show how to strengthen the initial continuous time formulation for the first special case with only one batch task. We consider both the uncapacitated (no limit on the number of reactors) and capacitated cases. In section 2.4, we study the formulation for the second special case that takes into account multiple batch tasks. In section 2.5, we propose valid inequalities for the third special case that takes into account batch and continuous tasks. Finally, in section 2.6, we conclude.

2.1 Problem description

We consider a general production process modeled by a resource task network (see Pantelides [35]), where the resources are the processing units, the utilities shared by the tasks, and the storage tanks containing the intermediate products produced or consumed by the tasks. In Figure 2.1, we represent a resource task network with 2 tasks and 5 resources. In this case, task 1 produces the set of resources $\{2,4\}$ and consumes resources $\{1,4\}$. Task 2 produces the set of resources $\{3,5\}$ and consumes resources $\{2,5\}$, where resources 1 to 3 model the storage tanks for raw material, intermediate product and finished product, respectively. Resources 4,5 are the processing units performing tasks 1 and 2, respectively.

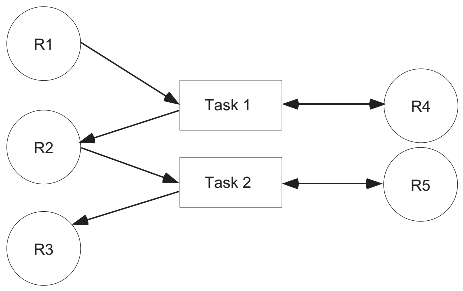

Figure 2.1: Resource Task Network

In this process, there are both batch and continuous tasks. Each batch task has a fixed processing time, can be processed on a subset of reactors and can be repeated several times. The main decision for a batch task is to determine the starting times of the corresponding instances of the task. Precedence and zero waiting time constraints exist between some of the batch tasks. For each continuous task, the processing rate has to be determined over time. This rate has to satisfy some given lower and upper limits. The batch and the continuous tasks consume and produce resources, for which we have some capacity restrictions. Moreover, the continuous tasks cannot be interrupted. The objective is to obtain a cyclic schedule of the mixed plant maximizing its productivity, where productivity is defined as the quantity of finished product produced over one cycle, divided by the cycle duration.

We consider a cyclic scheduling problem since the demand of product is relatively stable. Therefore, the objective of the scheduling problem is to obtain a schedule that can be repeated over time, and that maximizes long term productivity. This is also the reason why we do not include inventory costs and change over costs in the model. We also do not consider change over times between different products processed in a same reactor. This would deserve further investigations.

In this section, we propose a general definition of the sets and variables used in order to model this scheduling problem. We are given a set of batch tasks BT, a set of continuous tasks CT and a maximum number T of time slots.

An event is here defined as the beginning or end of a batch task, and a time slot is the time between two events. The number of time slots $t \in \{1, \ldots, T\}$ in the cycle is limited. Events and time slots are numbered from 1 up to T. In cyclic scheduling, the event at the end of time slot T coincides with the event occurring at the beginning of time slot 1, and is numbered as event 1. In Figure 2.2 we represent the time decomposition into time slots.

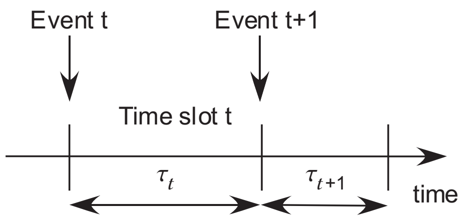

Figure 2.2: Event and Time slot

For generality, we suppose that each resource has its own resource unit that we call ru. The size of a batch of task i is denoted by $BS_i[ru]$. The processing time of a batch task $i \in BT$ is constant and is given by $p_i[h]$. The continuous tasks $j \in CT$ are performed continuously. The lower and upper bound on the speed or processing rate of the continuous task $j \in CT$ are $\underline{\rho}_j[ru/h]$ and $\overline{\rho}_j[ru/h]$, respectively. There are some precedence constraints between specific batch tasks and some resources $r \in \{1, \ldots, R\}$ are shared between the tasks.

The four types of sets are the following :

i :	is the index of a batch task, $i \in BT$
j :	is the index of a continuous task, $j \in CT$
t :	is the index of a time slot, $t \in \{1, \ldots, T\}$
r :	is the index of a resource, $r \in \{1, \ldots, R\}$

The five types of decision variables are the following :

τ_t :	is the duration of time slot $t.[h]$
$z_{i,t,t'}$:	$= 1$ if a batch of task i starts at the beginning of time slot t and finishes at the end of time slot t'
	$= 0$ otherwise
$q_{j,t}$:	is the quantity processed by the continuous task j during time slot t $[ru]$; (at rate $\dfrac{q_{j,t}}{\tau_t}$)

$w_{r,t}$:	is the rate $[ru/h]$ or the quantity $[ru]$ of resource r available at the beginning of time slot t, after adding (resp. removing) the resources released (resp. consumed) at the event placed at the beginning of time slot t.
$wf_{r,t}$:	is the rate $[ru/h]$ or the quantity $[ru]$ of resource r available at the end of time slot t, before adding (resp. removing) the resources released (resp. consumed) at the event at the end of time slot t.

As proposed in Castro et al. [12], to reduce the number of binary variables $z_{i,t,t'}$, we impose the following rule. A batch task i can only be performed during a maximum number of consecutive time slots denoted by d. This means that if task i starts at time slot t, this task has to end at or before time slot $t + d - 1$. This implies that $z_{i,t,t'}$ exists for all $i \in BT, t \in \{1, \dots, T\}$ and $t' \in \{t, \dots, t + d - 1\}$. The choice of the value for T and d induces very restrictive condition not related to the physical description of the plant. There is no way to decide the best value of T and d for a given problem. Therefore, we take into account that T and d are part of the problem formulation. This will be illustrated in Section 3.2.5.

In general, it is possible that a task starts in the current cycle and finishes in the next cycle. As observed in Shah et al. [45], it is possible to optimize the schedule over one cycle by using the concept of task 'wrap-around', because the end of the task in the next cycle has a corresponding end in the current cycle.

Definition 2.1.1. *$\Omega(t)$ is the 'wrap-around' time operator defined in Schilling and Pantelides [44] as*

$$\begin{aligned} &\Omega(t) = t \qquad \forall t : 1 \leq t \leq T, \\ &\Omega(t) = \Omega(t - T) \textit{ for } t > T, \\ &\Omega(t) = \Omega(t + T) \textit{ for } t < 1. \end{aligned}$$

So, a batch task i starting in time slot t, $1 \leq t \leq T$, and finishing in time slot $t + k$ corresponds to $z_{i,t,\Omega(t+k)} = 1$ as illustrated in Figure 2.3.

Typically, continuous time formulations are weak (i.e. implying a large number of branch-and-bound nodes to solve the scheduling problem to optimality) and our goals are to improve the tightness of the formulation for special cases of the general scheduling problem that we want to solve, and to show how to extend and use these strengthened formulations for the general case.

2.2 Models Classification

In this thesis, we are addressing various cyclic scheduling problems. We define a classification scheme of such cyclic scheduling problems composed of four fields:

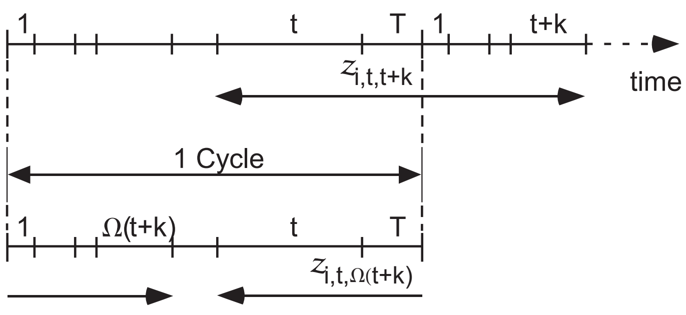

Figure 2.3: The 'wrap-around' time operator $\Omega(t)$

BT/CT/REACTORS/RESOURCES.
We also add one field F describing the formulation used to model each cyclic scheduling problem.
The general classification proposed for our cyclic scheduling problems is thus expressed as follows :

BT/CT/REACTORS/RESOURCES//F

In each field, the notation $[x, y, z]^1$ means that we select exactly one element from the set $\{x, y, z\}$, and the notation $[x, y, z]^*$ means that we select any subset of $\{x, y, z\}$. Empty fields are dropped.

We start by describing the four fields related to the problem description.

The first field BT defines the number of batch tasks of the scheduling problem.
BT=$[1B,nB]^1$.
1B : the problem contains one batch task.
nB : the problem contains multiple batch tasks.

The second field CT defines the number of continuous tasks of the scheduling problem.
CT=$[0C,1C,nC]^1$.
0C : there is no continuous task in the problem.
1C : the problem contains one continuous task.
nC : the problem contains multiple continuous tasks.

The third field REACTORS characterizes the restriction on the number of reactors for the scheduling problem.
REACTORS=$[CAP,UNCAP]^1$.
CAP : the number of reactors is limited for the problem.
UNCAP: the number of reactors is not limited for the problem.

Note that the parameter d defines implicitly a capacity constraint because at most d batches can be in process at any time. This parameter is present in all formulations used, CAP or UNCAP.

The fourth field RESOURCES indicates whether or not the problem contains restriction on the consumption or production of additional resources (other than the number of reactors to process the batch tasks) in the scheduling problem.
RESOURCES=[Res]*.
Res : the restricted resources in the problem can be : a storage tank, utilities, ...

The field F describes the model formulation.
F=[INIT,STR]^1.
INIT : we use the initial formulation in order to model the problem
STR : we use the strengthened formulation, or variants of the strengthened formulation, in order to model the problem.

To illustrate the classification scheme proposed for cyclic scheduling problems, we give below two examples :
1) 1B/0C/UNCAP//INIT means that the scheduling problem is composed of one batch task, the number of reactors is not limited and we use the initial formulation in order to model the problem.
2) nB/1C/CAP/Res//STR means that the scheduling problem is composed of multiple batch tasks and of one continuous task. The number of reactors is restricted and some other resources are taken into account. The problem is modeled by using the strengthened formulation.

2.3 One batch task

First we analyze a continuous time formulation for special case one, i.e., the cyclic scheduling of a batch plant performing one batch task of fixed duration p. Multiple instances or realizations of this batch task can be processed in parallel on several processing units or reactors. For this special case of the general problem, the process contains no continuous task and no resource restrictions. The objective is to maximize the productivity of the process. This problem reduces to the maximization of the number of batches produced per unit of time. This objective function is non linear. It was shown by Isbell and Marlow [23], and also extended by Dinkelbach [19], that this nonlinear objective function can be optimized for continuous problems by solving a sequence of linear optimization problems where a parameter of the objective function (the fixed cost μ per unit of time) has to be updated at each iteration. Their proof carries over for mixed integer linear problems (see for example in Megiddo [29]) and we give more details about this in Section 3.2.1. We study the continuous time formulation for this special case and we use as objective function the linearized objective corresponding to a single iteration of this objective linearization procedure.

We divide this problem in two parts : the capacitated case, where we consider that there is a limited number of processing units available to perform the batch task in parallel, and the uncapacitated case with no restriction. The binary variable $z_{i,t,t'}$ becomes $z_{t,t'}$ in this case since we consider only one batch task i.

2.3.1 Model formulation

The initial MIP formulation for this cyclic scheduling problem is the following :

1B/0C/CAP//INIT

$$\max \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)} - \mu \sum_{t=1}^{T} \tau_t \tag{2.1}$$

$$p z_{t,\Omega(l)} \leq \sum_{k=t}^{l} \tau_{\Omega(k)} \qquad \forall t, l : 1 \leq t \leq T, t \leq l \leq t+d-1 \tag{2.2}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} \leq p z_{t,\Omega(l)} + p(l-t+1)(1-z_{t,\Omega(l)}) \qquad \forall t, l : 1 \leq t \leq T, t \leq l \leq t+d-1 \tag{2.3}$$

$$\sum_{l=t}^{t+d-1} z_{t,\Omega(l)} \leq 1 \qquad \forall t : 1 \leq t \leq T \tag{2.4}$$

$$\sum_{t=l-d+1}^{l} z_{\Omega(t),l} \leq 1 \qquad \forall l : 1 \leq l \leq T \tag{2.5}$$

$$\sum_{t=1}^{T} \sum_{\substack{l=t \mid t \leq t' \leq l \\ \text{or } t \leq t'+T \leq l}}^{t+d-1} z_{t,\Omega(l)} \leq nbr\_unit \qquad \forall t' : 1 \leq t' \leq T \tag{2.6}$$

$$z_{t,\Omega(l)} \in \{0,1\} \forall t, l : 1 \leq t \leq T, t \leq l \leq t+d-1, \quad \tau_t \geq 0 \forall t : 1 \leq t \leq T \tag{2.7}$$

where $\mu \geq 0$ is a constant and where the 'wrap-around' time operator $\Omega(t)$ is defined in Section 2.1.

The linear objective function (2.1) corresponds to a measure of the productivity, corresponding to total cycle production minus μ times the cycle duration. In this linearized objective, μ corresponds to a fixed production cost per unit of time. The constraints (2.2)-(2.3) are the timing constraints and ensure that the duration of the batch task is well respected, i.e. $\sum_{k=t}^{l} \tau_{\Omega(k)} = p$ when $z_{t,\Omega(l)} = 1$. In particular (2.3) for $t = l$ reduces to $\tau_t \leq p$. The constraints (2.4)-(2.5) specify the fact that at most one batch task can begin and finish at each time event. The constraints (2.6) take into account the fact that the number of units available for performing the batch task is limited. In particular, one unit is performing an instance of the batch task at time slot t', if an instance of the batch task starts at a time slot t before time slot t' and finishes after t' at a time slot l, i.e. if $t \leq t' \leq l$, or if an instance of the batch task starts at a time slot t before time slot t' in the next cycle (i.e. before $t' + T$) and finishes after $t' + T$ at time slot l, i.e. if $t \leq t' + T \leq l$. Indeed, by using the 'wrap-around' time operator $\Omega(t)$, if $z_{t,\Omega(l)} = 1$ with $t \leq t' + T \leq l$, this implies that an instance of the batch task is performed on one unit at time slot $\Omega(t' + T)$, and we have that $\Omega(t' + T) = t'$.

Finally, the constraints (2.7) ensure that z are binary variables and time slot duration variables (τ) are non negative.

To remove the redundancy in the solution set, we can fix the initial starting time slot of one batch task that has to be processed during the cyclic schedule. When there is only one batch task, this can be achieved by adding the constraint: $\sum_{t'=1}^{d} z_{1,t'} = 1$.

We study first a special case of this problem where we suppose that the number of processing units is not restricted (Uncapacitated case), i.e. $nbr\_unit \geq d$.

Uncapacitated case

We study first the quality of the initial formulation without the constraint (2.6) limiting the number of units. Using PORTA, we can analyze polytopes and polyhedra of small dimensions, only one instance at a time. Given the extreme points and extreme rays of a small instance of a given problem, PORTA can compute the "numerical" linear representation of the convex hull of this single small instance. Then we have to generalize the results of PORTA in order to have a correct general "symbolic" form for the linear representation of the convex hull that is valid for every instance of the given problem, or at least to obtain some improved formulation for the general problem at hand.

Following this approach, we were able to suggest a tighter formulation for the constraints (2.2) and (2.3).

We prove below that the constraints (2.2) can be tightened as follows :

$$p \sum_{k=t|t\neq l}^{l} z_{t,\Omega(k)} + p z_{\Omega(l),\Omega(l)} \leq \sum_{k=t}^{l} \tau_{\Omega(k)} \tag{2.8}$$

$$p \sum_{k=t|t\neq l}^{l} z_{\Omega(k),\Omega(l)} + p z_{t,t} \leq \sum_{k=t}^{l} \tau_{\Omega(k)} \tag{2.9}$$

for all $t, l : 1 \leq t \leq T,\ t \leq l \leq t+d-1$

To illustrate the difference between the initial and the strengthened formulation, we write the initial constraint (2.2) for $t = 1$ and $l = 3$:

$$\tau_1 + \tau_2 + \tau_3 \geq p z_{1,3}.$$

The corresponding strengthened expression (2.8) can be written as

$$\tau_1 + \tau_2 + \tau_3 \geq p\,(z_{1,1} + z_{1,2} + z_{1,3}) + p z_{3,3}$$

and intuitively explained as follows. In Figure 2.4, taking into account the fact that we can start or finish at most one batch task at each time slot, we can see that we may add the variables corresponding to the dashed intervals in the right hand side, and keep the inequality valid. This holds because $z_{1,3} + z_{1,2} + z_{1,1} \leq 1$, and if $z_{1,3} + z_{1,2} + z_{1,1} = 1$ then $\tau_1 + \tau_2 + \tau_3 \geq p$. Similarly, if $z_{1,3} + z_{3,3} = 1$ then we must also have $\tau_1 + \tau_2 + \tau_3 \geq p$. Finally, when $z_{1,1} + z_{1,2} + z_{1,3} + z_{3,3} = 2$ (for instance when $z_{1,1} = z_{3,3} = 1$) then the two batches do not overlap and we must have $\tau_1 + \tau_2 + \tau_3 \geq 2p$. Constraint (2.8) is clearly stronger than (2.2) because non negative terms are added to the left hand side.

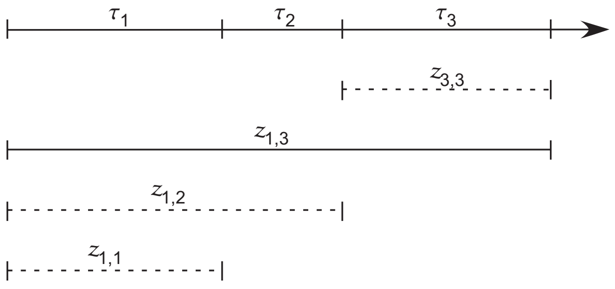

Figure 2.4: Strengthened constraint (2.8) : an example

The second strengthened inequality (2.9) is closely related to the first and can be explained in the same way.

We can mention here that the constraints (2.8) and (2.9) are equivalent for $l = t$ and $l = t + 1$.

Constraint (2.3) reduces to $\tau_t \leq p$ when $l = t$. We prove below that the constraints (2.3) when $t < l$ can be tightened as follows :

$$\sum_{k=t}^{l} \tau_{\Omega(k)} + p \sum_{k=l-d+1}^{l-1} \min\{l-t, l-k\} z_{\Omega(k),\Omega(l)} \leq p(l-t+1) \quad (2.10)$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} + p \sum_{k=t+1}^{t+d-1} \min\{l-t, k-t\} z_{t,\Omega(k)} \leq p(l-t+1) \quad (2.11)$$

for all $t, l : 1 \leq t \leq T,\ t \leq l \leq t + d - 1$.

The two strengthened inequalities are closely related and we explain here intuitively how to interpret one of them.

To illustrate the difference between the initial and the strengthened formulation, we write the initial constraint (2.3) for $t = 1$, $l = 3$ and $d = 4$:

$$\tau_1 + \tau_2 + \tau_3 \leq p z_{1,3} + 3p(1 - z_{1,3}) = p(3 - 2z_{1,3}). \quad (2.12)$$

The corresponding strengthened expression (2.11), can be written as

$$\tau_1 + \tau_2 + \tau_3 \leq p(3 - z_{1,2} - 2z_{1,3} - 2z_{1,4})$$

In Figure 2.5, taking into account the fact that we can start at most one batch task at each time slot, we can see that we may subtract in the right hand side of (2.12) a positive multiple of the variables corresponding to the dashed intervals, and keep the inequality valid. This holds because $Z = z_{1,2} + z_{1,3} + z_{1,4} \leq 1$ and if $Z = 0$, then $\tau_1 + \tau_2 + \tau_3 \leq 3p$ by the initial upper bound on τ_t. Similarly, if $Z = 1$ and $z_{1,2} = 1$ then $\tau_1 + \tau_2 = p$ and $\tau_3 \leq p$ and finally if $Z = 1 = z_{1,3} + z_{1,4}$ then $\tau_1 + \tau_2 + \tau_3 \leq p$ because the batch task starts in $t = 1$ and finishes at or after $l = 3$.

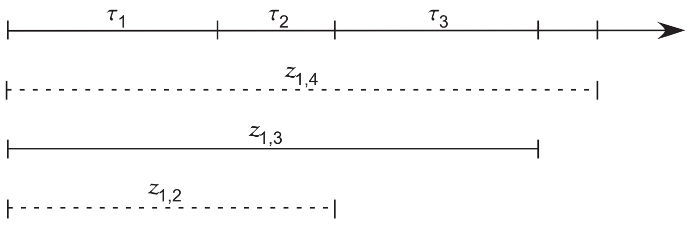

Figure 2.5: Strengthened constraint (2.11) : an example

We prove also that the constraints (2.8)-(2.9) and (2.10)-(2.11) are facet-defining for the problem defined by constraints (2.2)-(2.5) and (2.7).

a. Valid and facet defining inequalities

The approach used for proving that the constraints (2.8)-(2.9) and (2.10)-(2.11) are facet-defining is given in Wolsey [49].

We begin with the inequality (2.8). This inequality can be denoted as $\pi x \leq \pi_0$, where x is composed of the variables τ and z such that :

$$x = \begin{pmatrix} \tau_1 \\ \dots \\ \tau_T \\ z_{1,1} \\ \dots \\ z_{1,d} \\ z_{2,2} \\ \dots \\ z_{T,\Omega(T+d-1)} \end{pmatrix}$$

First we to prove that the inequality (2.8) is valid, and then that it is facet-defining. Let $X^U = \{(\tau, z) \in R^{T+Td} : (\tau, z) \text{ satisfies constraints } (2.2)-(2.5), (2.7)\}$.

Proposition 2.3.1. *Constraint (2.8) is valid for X^U.*

Proof. If $z_{\Omega(l),\Omega(l)} = 0$, then because of constraints (2.4), only one of the $z_{t,\Omega(k)}$ binary variable ($t \leq k \leq l$) can take the value 1. If $z_{t,\Omega(k')} = 1$ ($t \leq k' \leq l$), then by constraints (2.2)-(2.3), $\tau_t + \dots \tau_{\Omega(k')} = p$ and therefore $\sum_{t'=t}^{l} \tau_{\Omega(t')} \geq p$. Otherwise, $\sum_{t'=t}^{l} \tau_{\Omega(t')} \geq 0$ and this is valid.

Suppose now that $z_{\Omega(l),\Omega(l)} = 1$, then because of constraints (2.4)-(2.5), only one of the $z_{t,\Omega(k')}$ with $t \le k' \le l-1$ can be equal to 1. If $z_{t,\Omega(k')} = 1$ for some k' with $t \le k' \le l-1$, then $\sum_{k=t}^{k'} \tau_{\Omega(k)} = p$ and $\tau_{\Omega(l)} = p$. Therefore $2p \le \sum_{k=t}^{l} \tau_{\Omega(k)}$ is valid because the batch tasks do not overlap. If all the $z_{t,\Omega(k')}$ with $t \le k' \le l-1$ are equal to 0, then $\tau_{\Omega(l)} = p$ and the constraint becomes $p \le \sum_{k=t}^{l} \tau_{\Omega(k)}$ and is valid. □

We prove below that X^U is full dimensional.

We need two definitions.

Definition 2.3.2. *$x^1, \dots, x^n \in R^{T+Td}$ are affinely independent if $\sum_{i=1}^{n} \alpha^i x^i = 0$ and $\sum_{i=1}^{n} \alpha^i = 0$ implies $\alpha^i = 0$ for $i = 1, \dots, n$, or equivalently if the directions $x^2 - x^1, \dots, x^n - x^1$ are linearly independent.*

Definition 2.3.3. *The dimension of X^U ($dim(X^U)$) is equal to the maximum number of affinely independent points in X^U minus 1.*

Proposition 2.3.4. *$dim(X^U) = T(d+1)$, i.e., X^U is full dimensional.*

Proof. By taking $\tau_t = p$ for $t \in \{1, \dots, T\}$ and $z = 0$, we generate T feasible points $x^1, \dots, x^T$ for X^U. By taking $z_{t,\Omega(l)} = 1$ and $\tau_t = p$ for $t, l : t \in \{1, \dots, T\}, l \in \{t, \dots, t+d-1\}$, we generate Td feasible points $x^{T+1}, \dots, x^{T+Td}$ for X^U. Since $x^{T+Td+1} = 0 \in X^U$, it remains to prove that the $T+Td$ generated directions $x^i - x^{T+Td+1} = x^i$, for $i = 1, \dots, T+Td$, are linearly independent. By observing the structure of the points, the unique solution of $\sum_{i=1}^{T+Td} \lambda_i x^i = 0$ is $\lambda_i = 0$, $\forall i \in \{1, \dots, T+Td\}$, and therefore the result follows. □

Proposition 2.3.5. *Constraint (2.8) is facet defining for X^U.*

Proof. (i) For given t and l ($t \le l \le t+d-1$), we select first s points, $s \ge dim(X) = T(d+1)$ satisfying the inequality at equality. We denote them as follows : $x^1, \dots, x^{Td+T}$. By default, for each generated point, the variables that are not mentioned take the value 0.

a. $\tau_{t_1} = 0 \ \forall \ t_1 : 1 \le t_1 \le T, z_{t_1,\Omega(t_2)} = 0 \ \forall \ t_1, t_2 : 1 \le t_1 \le T, t \le t_2 \le t+d-1$. This solution (x=0) satisfies the inequality at equality.

b. $\tau_{\Omega(t_1)} = p \ \forall \ t_1 : l+1 \le t_1 \le t+T-1$. These $T-(l-t+1)$ points satisfy the inequality at equality.

c. For each (t_1, t_2): $1 \le t_1 \le T$, $t_1 \le t_2 \le t_1+d-1$, such that $(t_1, t_2) \ne (l, l)$ and $t_1 \ne t$, or $t_1 = t$ and $t_2 > l$, we select $z_{\Omega(t_1),\Omega(t_2)} = 1$ and we generate a feasible point satisfying the inequality (2.8) at equality as follows :

   (1) if $\{t_1, \dots, t_2\} \subseteq \{t+1, \dots, l\}$ and $t_1 \ne t_2$ then we choose $\tau_{\Omega(t_1)} = p$ ($\tau_t = \ldots = \tau_{\Omega(t_1-1)} = 0$), $z_{t,\Omega(t_1)} = 1$ and $z_{\Omega(t_1),\Omega(t_2)} = 1$.

   (2) if $t_1 = t_2$ and $t_1 \in \{t+1, \dots, l-1\}$ then we take $\tau_{\Omega(t_1)} = p$, $z_{t,\Omega(l)} = 1$ and $z_{\Omega(t_1),\Omega(t_2)} = 1$.

   (3) if $\{t_1, \dots, t_2\} \not\subseteq \{t, \dots, l\}$, then in this case , $\exists t' \in \{1, \dots, t-1\} \cup \{l+1, \dots, t+T-1\}$ such that $t' \in \{t_1, \dots, t_2\}$ and we choose $\tau_{\Omega(t')} = p$ and $z_{\Omega(t_1),\Omega(t_2)} = 1$.

These feasible points satisfying the inequality at equality correspond to every variables $z_{\Omega(t_1),\Omega(t_2)}$ excepts the one starting in time period t and finishing in $\{t,\ldots,l\}$ and the one starting and finishing in time period l. Therefore, there are $Td-(l-t+1)-1$ such points.

d. For each $t_1 \in \{t,\ldots,l\}$, we generate one point with $z_{t,\Omega(l)} = 1$ and $\tau_{\Omega(t_1)} = p$. This generates $(l-t+1)$ points.

e. If $t < l$, for each $t_1 \in \{t,\ldots,l-1\}$, we generate one point with $\tau_t = p$ and $z_{t,\Omega(t_1)} = 1$. There are $(l-t)$ such points.

f. Finally, we generate one point with $z_{\Omega(l),\Omega(l)} = 1$ and $\tau_{\Omega(l)} = p$ and this point satisfies the inequality at equality.

It is easy to check that all these points are feasible for (2.2)-(2.5),(2.7). The total number of points generated satisfying the inequality at equality is $Td+T$.

(ii) In order to prove that (2.8) (written as $\pi x \leq \pi_0$) is facet defining for given t and l, it suffices to prove that all points $x^1,\ldots,x^{Td+T}$ generated above are affinely independent. This can be done by showing that the system of linear equalities in the unknowns $(\mu,\mu_0) \in R^{Td+T} \times R$,

$$\sum_{j=1}^{T\ d+T} \mu_j x_j^k = \mu_0 \ \forall k = 1,\ldots,T\ d+T \tag{2.13}$$

has a unique non zero solution (μ,μ_0), up to a constant multiplier, with $\mu_0 = \pi_0 = 0$ and μ_j equal to the coefficient π_j of variable x_j in (2.8).(see Wolsey [49])

By using the various points previously defined, we solve the system of equation as follows :

By considering the feasible point generated in a. (i.e., $x = 0$), we can deduce that $\mu_0 = 0$. For ease of notation, we denote by μ_{τ_t} (resp. $\mu_{z_{t,l}}$) the coefficient of μ corresponding to variable τ_t (resp. $z_{t,l}$).

From the feasible points in b., we observe that $\mu_{\tau_{\Omega(t_1)}} = \mu_0 = 0$ for all $t_1 \in \{l+1,\ldots,t+T-1\}$.

By considering and comparing the points generated in d., we can deduce that $\mu_{\tau_t} = \ldots = \mu_{\tau_{\Omega(l)}}$ and that $\mu_{z_{t,\Omega(l)}} = -p\mu_{\tau_t}$ because $\mu_0 = 0$.

Then, by considering the point generated in e. and combining the results obtained with the points in d., we have that $\mu_{z_{t,t}} = \mu_{z_{t,\Omega(t+1)}} = \ldots = \mu_{z_{t,\Omega(l-1)}} = -p\mu_{\tau_t} = \mu_{z_{t,\Omega(l)}}$ and by considering also the point generate in f., we have that $\mu_{z_{\Omega(l),\Omega(l)}} = -p\mu_{\tau_{\Omega(l)}} = -p\mu_{\tau_t} = \mu_{z_{t,t}} = \ldots = \mu_{z_{t,\Omega(l)}}$.

Finally, we consider the points generated in c. It is not difficult to observe that for each (t_1,t_2), $t_1 \leq t_2 \leq t_1+d-1$, such that $t_1 \neq t$ or $t_1 = t$ and $t_2 > l$, and $(t_1,t_2) \neq (l,l)$, we must have $\mu_{z_{\Omega(t_1),\Omega(t_2)}} = 0$. For instance, in case (1), as $\mu_{z_{t,\Omega(t_1)}} = -p\mu_{\tau_{\Omega(t_1)}}$ and $\mu_0 = 0$, we must have $\mu_{z_{\Omega(t_1),\Omega(t_2)}} = 0$. The other cases are similar.

So far, we have shown that the solution (μ, μ_0) of system (2.13) is defined by :

$$\begin{aligned}
\mu_{\tau_{\Omega(t_1)}} &= \mu_{\tau_t} \text{ for } t_1 \in \{t+1, \dots, l\} \\
\mu_{z_{t,\Omega(t_1)}} &= -p\mu_{\tau_t} \text{ for } t_1 \in \{t, \dots, l\} \\
\mu_{z_{\Omega(l),\Omega(l)}} &= -p\mu_{\tau_t}
\end{aligned}$$

and the other components of μ are equal to zero. This shows that $(\mu, \mu_0) = \lambda(\pi, \pi_0)$ for some $\lambda \in R$. $(\lambda = -\mu_{\tau_t})$

This proves that the constraint (2.8) is facet defining for the problem defined by the equation (2.2)-(2.5) and (2.7).

□

Proposition 2.3.6. *Constraint (2.9) is valid and facet defining for X^U.*

Proof. The proof is similar to the proof given for the inequality (2.8). □

We now prove that the inequality (2.10) is valid and facet defining for X^U.

Proposition 2.3.7. *Constraint (2.10) is valid for X^U.*

Proof. When $l = t$, constraint (2.10) reduces to $\tau_t \leq p$ which is valid by (2.3). We consider now the general case with $l > t$.

If $z_{\Omega(k),\Omega(l)} = 0 \ \forall k \in \{l-d+1, \dots, l-1\}$, then the constraint becomes $\sum_{k=t}^{l} \tau_{\Omega(k)} \leq p(l-t+1)$ and this is clearly valid since we know that for every $t \in \{1, \dots, T\}$, $\tau_t \leq p$.

Otherwise, $\sum_{k=l-d+1}^{l-1} z_{\Omega(k),\Omega(l)} = 1$ because of constraints (2.5). Assuming that $z_{\Omega(k'),\Omega(l)} = 1$ for some $k' \in \{l-d+1, \dots, l-1\}$, constraint (2.10) becomes $\sum_{k=t}^{l} \tau_{\Omega(k)} + p \min\{l-t, l-k'\} z_{\Omega(k'),\Omega(l)} \leq p(l-t+1)$.

Suppose first that $t \leq k' \leq l-1$, then $\tau_{\Omega(k')} + \dots \tau_{\Omega(l)} = p$ and the inequality becomes : $\sum_{k=t}^{k'-1} \tau_{\Omega(k)} \leq p(k'-t)$ and this is clearly valid because $\tau_{\Omega(k)} \leq p$ for all k.

Suppose now that $t > k'$, then $\sum_{k=t}^{l} \tau_{\Omega(k)} \leq \sum_{k=k'}^{l} \tau_{\Omega(k)} = p$ and the inequality is valid because $\sum_{k=t}^{l} \tau_{\Omega(k)} + p(l-t) \leq p + p(l-t) = p(l-t+1)$. □

Proposition 2.3.8. *Constraint (2.10) is facet defining for X^U.*

Proof. We use the same technique for the proof as for constraint (2.8).
Again, by default, for each generated point, the variables that are not mentioned take the value 0.

(i) For fixed t and l $(t \leq l \leq t+d-1)$, we enumerate $Td+T$ feasible solutions of X^U satisfying the corresponding inequality (2.10) at equality.

We assume first that $t < l$.

a. $\tau_t = \ldots = \tau_{\Omega(l)} = p$, and in addition :

(1) All other decision variables are 0. This generates one feasible solution.

(2) For each $t_1 \notin \{t, \ldots, l\}$, we generate one point with $\tau_{\Omega(t_1)} = p$. This generates $T - (l - t + 1)$ feasible points.

(3) For each $t_1 \in \{t, \ldots, l\}$, we generate one point with $z_{\Omega(t_1),\Omega(t_1)} = 1$. This generates $l - t + 1$ feasible points.

(4) For each $t_2 \notin \{t, \ldots, l\}$, we generate one point with $z_{\Omega(t_2),\Omega(t_2)} = 1$ and $\tau_{\Omega(t_2)} = p$. This generates $T - (l - t + 1)$ feasible points.

b. For each $t_1 \in \{t, \ldots, l\}$, we generate one point with $z_{t,\Omega(l)} = 1$ and $\tau_{\Omega(t_1)} = p$. This generates $l - t + 1$ points.

c. For each t_1, t_2 with $1 \leq t_1 \leq T$, $t_1 < t_2 \leq t_1 + d - 1$, $t_1 \neq t$ and $t_2 \neq l$, we generate one point with $z_{\Omega(t_1),\Omega(t_2)} = 1$ and in addition :

(1) if $t_1 \in \{t + 1, \ldots, l\}$, then we take $\tau_{\Omega(t_1)} = p$ and $z_{t,\Omega(l)} = 1$.

(2) If $t_1 \notin \{t + 1, \ldots, l\}$ and $t_2 \in \{t, \ldots, l - 1\}$ then we take $\tau_{\Omega(t_2)} = p$ and $z_{t,\Omega(l)} = 1$.

(3) If $t_1, t_2 \notin \{t, \ldots, l\}$ and $t_1 \leq t \leq l \leq t_2$, we take $z_{t,\Omega(l)} = 1$ and $\tau_t = p$.

(4) Finally, if $t_1, t_2 \notin \{t, \ldots, l\}$ and either $t_1, t_2 < t$ or $t_1, t_2 > l$, by taking $\tau_t = \ldots = \tau_{\Omega(l)} = p$ and $\tau_{\Omega(t_1)} = p$, we get a feasible point.

The total number of points generate here is Td - $\underbrace{T}_{t_1 \neq t_2}$ - $\underbrace{(d-1)}_{t_1 \neq t}$ - $\underbrace{(d-1)}_{t_2 \neq l}$ + $\underbrace{1}_{z_{t,\Omega(l)} \text{ counted two times}}$.

d. For each $t_1 \in \{t + 1, \ldots, t + d - 1\}$, $t_1 \neq l$, we generate one feasible point with $z_{t,\Omega(t_1)} = 1$ and

(1) if $t_1 > l$, we have that $t \neq \Omega(l - d + 1)$, we take $z_{\Omega(l-d+1),\Omega(l)} = 1$ and $\tau_{\Omega(l)} = p$.

(2) if $t_1 < l$ then we take $z_{\Omega(t+1),\Omega(l)} = 1$, $\tau_t = p$ and $\tau_{\Omega(l)} = p$.

We generate $d - 2$ points in this way.

e. For each $t_1 \in \{l - d + 1, \ldots, l - 1\}$, $t_1 \neq t$, we generate one feasible point with $z_{\Omega(t_1),\Omega(l)} = 1$ and

(1) if $t_1 > t$, we take $\tau_t = \ldots = \tau_{\Omega(t_1-1)} = p$ and $\tau_{\Omega(l)} = p$ and this point satisfies the inequality at equality because $p(t_1 - t) + p + p(l - t_1) = p(l - t + 1)$.

(2) if $t_1 < t$, we take $\tau_{\Omega(l)} = p$ and this is a feasible point.

We generate $d - 2$ points in this way.

The total number of points generated is $T + Td$.

If $t = l$ then (2.10) becomes $\tau_t \leq p$. We enumerate $Td + T$ feasible solutions for (2.2)-(2.5), (2.7) satisfying the corresponding inequality (2.10) at equality.

$\tau_t = p$, and in addition :

a. All other decision variables are 0. This generates one feasible point.

b. For each $t_1 \neq t$, we generate one point with $\tau_{\Omega(t_1)} = p$. This generates $T-1$ feasible points.

c. We generate one point with $z_{t,t} = 1$.

d. For each t_1, t_2 with $1 \leq t_1 \leq T$, $t_1 \leq t_2 \leq t_1 + d - 1$, $(t_1, t_2) \neq (t, t)$, we generate one point with $z_{t_1,\Omega(t_2)} = 1$ and in addition :
(1) if $t_1 \leq t \leq t_2$ or if $t_1 \leq t + T - 1 \leq t_2$, we get a feasible point.
(2) Otherwise, we take $\tau_{\Omega(t_1)} = p$ and we get a feasible point.
The total number of points generate here is $Td - 1$.

The total number of points generated is $T + Td$.

(ii) We have to solve the system of linear equations defined by (2.13) for the points defined in (i) satisfying the inequality (2.10) at equality, and prove that the unique solution up to a constant is $(\mu, \mu_0) = \lambda(\pi, \pi_0)$ for some $\lambda \in R$, where $\pi x \leq \pi_0$ denotes inequality (2.10). These points impose conditions on the solution of the linear system of equation.

First, we consider the case $t < l$.
By considering the feasible point generated in a(1), we can deduce that $\sum_{t_1=t}^{l} \mu_{\tau_{\Omega(t_1)}} p = \mu_0$, and therefore that $\mu_{\tau_{\Omega(t_1)}} = 0 \; \forall t_1 \notin \{t, \dots, l\}$ by a(2) and that $\mu_{z_{\Omega(t_1),\Omega(t_1)}} = 0 \; \forall t_1 \in \{1, \dots, T\}$ by a(3) and a(4).
By looking now at the feasible points generated in b., we have that $\mu_{\tau_t} = \dots = \mu_{\tau_{\Omega(l)}}$, and by the condition obtained in a., we have that

$$\mu_{\tau_{\Omega(t_1)}} = \left(\frac{1}{p(l-t+1)}\right) \mu_0 \forall t_1 \in \{t, \dots, l\}.$$

Therefore, we can deduce that

$$\mu_{z_{t,\Omega(l)}} = \mu_0 - \mu_0 \left(\frac{1}{p(l-t+1)}\right) p = \left(\frac{l-t}{l-t+1}\right) \mu_0.$$

By considering the results obtained above, the feasible points generated in c. imply that $\mu_{z_{\Omega(t_1),\Omega(t_2)}} = 0$ for $t_1 \neq t, t_2 \neq l$ and $t_1 \neq t_2$.
The feasible points generated in e. give the following conditions :
If $t_1 > t$ and $t_1 \neq l$ then
$\sum_{t_2=t}^{t_1-1} \mu_{\tau_{\Omega(t_2)}} p + \mu_{\tau_{\Omega(l)}} p + \mu_{z_{\Omega(t_1),\Omega(l)}} = \mu_0$, and this implies that $\mu_{z_{\Omega(t_1),\Omega(l)}} = \mu_0 - (t_1 - t + 1) p \frac{1}{p(l-t+1)} \mu_0 = \left(\frac{l-t_1}{l-t+1}\right) \mu_0$.
if $t_1 < t$ and $t_1 \neq l$ then
$\mu_{\tau_{\Omega(l)}} p + \mu_{z_{\Omega(t_1),\Omega(l)}} = \mu_0$ and this implies that $\mu_{z_{\Omega(t_1),\omega(l)}} = \mu_0 - p \left(\frac{1}{p(l-t+1)}\right) \mu_0 = \left(\frac{l-t}{l-t+1}\right) \mu_0$.

We can thus conclude from b. and e. that if $t_1 \neq l$:

$$\mu_{z_{\Omega(t_1),\Omega(l)}} = \min\left(\frac{l-t_1}{l-t+1}, \frac{l-t}{l-t+1}\right) \mu_0.$$

Finally, by looking at the point generated in d., we can deduce that $\mu_{z_{t,\Omega(t_1)}} = 0\ \forall t_1 \neq t$ and $t_1 \neq l$.

For the case $t = l$, by considering the point generated in a., we can deduce that first $\mu_{\tau_t} p = \mu_0$ and that all others μ's are equal to zero by b,c and d.

(iii) We now check that the above conditions on (μ, μ_0) allow one to obtain $(\mu, \mu_0) = \lambda(\pi, \pi_0)$ for some constant λ. The conditions for the case $t < l$ are the following :

$$\mu_{\tau_{\Omega(t_1)}} = \mu_0 \frac{1}{p(l-t+1)} = \lambda \text{ for } t_1 \in \{t, \ldots, l\}$$

and

$$\mu_{z_{\Omega(t_1),\Omega(l)}} = \mu_0 \min\left(\frac{l-t_1}{l-t+1}, \frac{l-t}{l-t+1}\right) = \lambda p \min\left(l-t_1, l-t\right)$$

for $t_1 \in \{l-d+1, \ldots, l\}$, and the other components of μ are equal to zero, and take the same value as the corresponding component of π.

By taking : $\lambda = \mu_0 \frac{1}{p(l-t+1)}$, these conditions show that the system (2.13) has a unique solution, up to the constant λ.

We have also that μ_0 has to be equal to $\lambda\pi_0$, i.e $\mu_0 = \lambda(l-t+1)p$ and this is clearly valid for the solution proposed.

By taking into account the conditions above for the case $t = l$, this proves that $(\mu, \mu_0) = \lambda(\pi, \pi_0)$ with $\lambda = \mu_0/p$.

This completes the proof that constraint (2.10) is facet defining for X^U.

□

Proposition 2.3.9. *Constraint (2.11) is valid and facet defining for X^U.*

Proof. The proof is similar to the proof given for the inequality (2.10). □

We prove below that the initial inequality (2.4) is facet defining for X^U.

Proposition 2.3.10. *Constraint (2.4) is facet defining for X^U.*

Proof. We use the same technique for the proof. Again, by default, for each generated point, the variables that are not mentioned take the value 0.

(i) For a given t and l ($t \leq l \leq t+d-1$), we enumerate $T\ d + T$ feasible points satisfying the corresponding inequality (2.4) at equality :

a. $z_{t,t} = 1$ and $\tau_t = p$, and, in addition,
(1) all other variables are 0. We generate one such feasible point.
(2) for each $t_1 \in \{1, \ldots, T\} \setminus \{t\}$, we generate one point with $\tau_{t_1} = p$. We generate $T-1$ feasible points in this way.

b. $z_{t,\Omega(t+1)} = 1$ and $\tau_{\Omega(t+1)} = p$. We generate one such feasible point.

c. For each $t_1 \in \{t+1, \ldots, t+d-1\}$, we generate one feasible point with $z_{t,\Omega(t_1)} = 1$ and $\tau_t = p$. We generate $d-1$ such feasible points.

d. For each $t_1 \in \{t-d+1,\ldots,t-1\}$, we generate one feasible point with $z_{\Omega(t_1),t}=1$, $z_{t,\Omega(t+1)}=1$ and $\tau_t=p$. We generate $d-1$ feasible points, in this way.

e. For each $t_1 \in \{1,\ldots,T\}$ and $t_2 \in \{t,\ldots,t+d-1\}$ with $t_1 \neq t$ and $\Omega(t_2) \neq t$, we generate one feasible point with $z_{\Omega(t_1),\Omega(t_2)}=1$ and, in addition,
(1) if $t \in \{t_1,\ldots,t_2\}$, we choose $z_{t,t}=1$ and $\tau_t=p$.
(2) if $t \notin \{t_1,\ldots,t_2\}$, we choose $z_{t,t}=1$, $\tau_t=p$ and $\tau_{\Omega(t_1)}=p$.

The number of points generated here is $Td-2d+1$

We can check that we have generated $T+Td$ feasible points that satisfy the inequality at equality.

(ii) We have to solve the system of linear equations defined by the equation (2.13) for the points defined in (i). By using these points, we obtain conditions on the solution of the linear system of equations.

By comparing the feasible points generated in a(1) and a(2), we can deduce that $\mu_{\tau_{t_1}}=0\ \forall\ t_1 \in \{1,\ldots,T\}\setminus\{t\}$. By looking then at c., we can deduce that $\mu_{z_{t,t}}=\mu_{z_{t,\Omega(t+1)}}=\ldots=\mu_{z_{t,\Omega(t+d-1)}}$ and by combining the results of a. and c., and by considering the point in b., we can deduce that $\mu_{\tau_t}=0$, and therefore $\mu_{z_{t,t}}=\mu_{z_{t,\Omega(t+1)}}=\ldots=\mu_{z_{t,\Omega(t+d-1)}}=\mu_0$. Finally, by considering d. and e., we can deduce that $\mu_{z_{\Omega(t_1),t}}=0$ for $t_1 \in \{t-d+1,\ldots,t-1\}$ and that $\mu_{z_{\Omega(t_1),\Omega(t_2)}}=0$ for all t_1,t_2 such that $t_1 \neq t$ and $t_2 \neq t$.

(iii) It remains to check that the solution (μ,μ_0) obtained is $(\mu,\mu_0)=\lambda(\pi,\pi_0)$ for some constant λ, where $\pi x \leq \pi_0$ denotes inequality (2.4). The solution (μ,μ_0) obtained satisfies the following conditions :

$$\mu_{z_{t,\Omega(t_1)}}=\mu_0=\lambda \text{ for } t_1 \in \{t,\ldots,t+d-1\}$$

and

$$\mu_0=\lambda$$

and all other components of μ and π are equal to zero.

Therefore the unique solution of (2.13) up to the constant λ is $(\mu,\mu_0)=\lambda(\pi,\pi_0)$ with $\lambda=\mu_0$.

This completes the proof that constraint (2.4) is facet defining for the problem defined by the equation (2.2)-(2.5) and (2.7).

□

Proposition 2.3.11. *Constraint (2.5) is facet defining for X^U.*

Proof. The proof is similar to the proof given for the inequality (2.4). □

We have proved that inequalities (2.4)-(2.5), (2.8)-(2.11) and (2.7) provide an improved or tightened formulation of (2.2)-(2.5), (2.7), and that inequalities (2.4)-(2.5), (2.8)-(2.11) are facet defining inequalities for the uncapacitated case.

b. Duality gap

To characterize the tightness of the reformulation, we now prove that the improved formulation obtained for the uncapacitated model leads to a formulation with a zero duality gap when considering objective function in the form (2.1).

The improved formulation for this cyclic scheduling problem composed of one batch task is recalled

1B/0C/UNCAP//STR

$$\max \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)} - \mu \sum_{t=1}^{T} \tau_t \tag{2.14}$$

$$p \sum_{k=t|t\neq l}^{l} z_{t,\Omega(k)} + p z_{\Omega(l),\Omega(l)} \leq \sum_{k=t}^{l} \tau_{\Omega(k)}$$

$$\forall t,l : 1 \leq t \leq T, t \leq l \leq t+d-1 \tag{2.15}$$

$$p \sum_{k=t|t\neq l}^{l} z_{\Omega(k),\Omega(l)} + p z_{t,t} \leq \sum_{k=t}^{l} \tau_{\Omega(k)}$$

$$\forall t,l : 1 \leq t \leq T, t \leq l \leq t+d-1 \tag{2.16}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} + p \sum_{k=l-d+1}^{l-1} \min\{l-t, l-k\} z_{\Omega(k),\Omega(l)} \leq p(l-t+1)$$

$$\forall t,l : 1 \leq t \leq T, t \leq l \leq t+d-1 \tag{2.17}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} + p \sum_{k=t+1}^{t+d-1} \min\{l-t, k-t\} z_{t,\Omega(k)} \leq p(l-t+1)$$

$$\forall t,l : 1 \leq t \leq T, t \leq l \leq t+d-1 \tag{2.18}$$

$$\sum_{l=t}^{t+d-1} z_{t,\Omega(l)} \leq 1 \qquad \forall t : 1 \leq t \leq T \tag{2.19}$$

$$\sum_{t=l-d+1}^{l} z_{\Omega(t),l} \leq 1 \qquad \forall l : 1 \leq l \leq T \tag{2.20}$$

$$z_{t,\Omega(l)} \in \{0,1\} \forall t,l : 1 \leq t \leq T, t \leq l \leq t+d-1,$$

$$\tau_t \geq 0 \forall t : 1 \leq t \leq T \tag{2.21}$$

We consider a relaxation of the problem (2.14)-(2.21). Then, we give the formulation of the dual of the LP relaxation of the relaxed problem. Finally, we prove that the duality gap is zero, i.e. the difference between the optimal objective value of the mixed integer program (2.14)-(2.21) and of the dual of the LP relaxation of this relaxed problem is zero. This implies that problem (2.14)-(2.21) and its LP relaxation have the same optimal objective value. This holds because of the specific objective function considered.

We construct a relaxation of the problem (2.14)-(2.21), with the facet defining inequalities (2.15) and (2.19). Its LP relaxation is the following

$$\max \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)} - \mu \sum_{t=1}^{T} \tau_t \tag{2.22}$$

$$p \sum_{k=t|t\neq l}^{l} z_{t,\Omega(k)} + p z_{\Omega(l),\Omega(l)} \leq \sum_{k=t}^{l} \tau_{\Omega(k)}$$

$$\forall t, l : 1 \leq t \leq T, t \leq l \leq t+d-1 \tag{2.23}$$

$$\sum_{l=t}^{t+d-1} z_{t,\Omega(l)} \leq 1 \qquad \forall t : 1 \leq t \leq T \tag{2.24}$$

$$z \geq 0, \tau \geq 0 \tag{2.25}$$

Then, we give the dual of the LP relaxation (2.22)-(2.25).

The dual variables associated with the constraints (2.23) are $\alpha_{t,\Omega(l)}$, for $1 \leq t \leq l \leq t+d-1$, $t \leq T$. The dual variables associated with the constraints (2.24) are u_t, for $1 \leq t \leq T$.

The dual problem is formulated as :

$$\min \sum_{t=1}^{T} u_t \tag{2.26}$$

$$st \quad p \sum_{t_1=l-d+1|t_1\neq t, t=l}^{l-1} \alpha_{\Omega(t_1),\Omega(l)} + p \sum_{t_2=l}^{t+d-1} \alpha_{t,\Omega(t_2)} + u_t \geq 1$$

$$\text{for all } t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\} \tag{2.27}$$

$$-\sum_{t=1}^{T} \sum_{l=t|t\leq k\leq l \text{ or } k\leq\Omega(l)<t}^{t+d-1} \alpha_{t,\Omega(l)} \geq -\mu \qquad \forall k \in \{1,\ldots,T\} \tag{2.28}$$

$$\alpha, u \geq 0 \tag{2.29}$$

In order to prove that the duality gap is zero, we take a feasible solution for the mixed integer program (2.14)-(2.21) and a feasible solution for the dual of the relaxation of this problem (2.26)-(2.29) and we show that the two objective function values are equal.

Proposition 2.3.12. *The duality gap is zero, i.e. problems (2.14)-(2.21) and (2.26)-(2.29) have the same optimal objective value.*

Proof. We have to distinguish two cases :

$$\frac{p\mu}{d} > 1 \tag{2.30}$$

and

$$\frac{p\mu}{d} \leq 1 \tag{2.31}$$

We define a feasible solution of the primal mixed integer problem (2.14)-(2.21) for both cases.

If the condition (2.30) is satisfied, the feasible MIP solution considered here is the zero solution, i.e. $z_{t,\Omega(t_1)} = 0$ for all $t \in \{1,\dots,T\}, t_1 \in \{t,\dots,t+d-1\}$ and $\tau_t = 0$ for all $t \in \{1,\dots,T\}$. This solution is trivially feasible and has objective value 0. Note that the optimality of this solution is a direct consequence of our proof that the duality gap is zero.

If the condition (2.31) holds, we propose a feasible solution : $z_{t,\Omega(t+d-1)} = 1\ \forall t \in \{1,\dots,T\}$ and $\tau_t = p/d\ \forall t \in \{1,\dots,T\}$.

The corresponding objective function value of the primal MIP is then

$$|T| - \mu\frac{p|T|}{d}. \tag{2.32}$$

We show first that this solution is feasible. Its optimality is a consequence of our proof that the duality gap is zero.

The constraints (2.4), (2.5) and (2.7) are clearly satisfied. For the constraints (2.2), if $l \neq t+d-1$ then the constraint reduces to $\sum_{k=t}^{l} \tau_{\Omega(k)} \geq 0$ and this is clearly valid. When $l = t+d-1$, the inequality (2.2) is satisfied at equality. Finally, for the constraints (2.3), if $l < t+d-1$, then the constraint reduces to $(l-t+1)p/d \leq p(l-t+1)$, which is valid because $d \geq 1$. Otherwise, if $l = t+d-1$, then the inequality (2.3) is satisfied at equality.

The proposed solution is therefore feasible for the primal MIP problem (2.2)-(2.5), (2.7). As (2.15)-(2.21) is a valid reformulation of the problem, the proposed solution is also feasible for (2.15)-(2.21).

We define now a feasible solution for the dual (2.26)-(2.29). We distinguish again the two cases (2.30) and (2.31).

When condition (2.30) is satisfied, we take the following dual solution : For all $t \in \{1,\dots,T\}, l \in \{t,\dots,t+d-1\}$: $\alpha_{t,\Omega(l)} = 1/p$ if $l = t+d-1$ and otherwise $\alpha_{t,\Omega(l)} = 0$, and $u_t = 0$ for all $t \in \{1,\dots,T\}$. We show that this solution is feasible.

Constraint (2.29) is trivially satisfied. For the constraints (2.27), if $t = l$ then the left hand-side equal 2 and is greater than the right hand-side. Otherwise, the left and right hand-side equal 1. For the constraints (2.28), the left hand-side is $-d/p \geq -\mu$ because (2.30) holds.

Therefore, the dual solution proposed is feasible when (2.30) holds.

When the other condition (2.31) is satisfied, we take the dual solution : For all $t \in \{1,\dots,T\}, l \in \{t,\dots,t+d-1\}$: $\alpha_{t,\Omega(l)} = \mu/d$ if $l = t+d-1$ and otherwise $\alpha_{t,\Omega(l)} = 0$, and $u_t = 1 - \frac{p\mu}{d}$ for all $t \in \{1,\dots,T\}$. We show that this dual solution is feasible.

Constraint (2.29) is satisfied since condition (2.31) holds. For the constraints (2.27), if $t = l$ then the left hand-side equal $2p\mu/d + (1 - p\mu/d) = 1 + p\mu/d$ and is greater or equal to the right hand-side ($0 \leq p\mu/d \leq 1$). Otherwise, the left and right hand sides equal 1. For the constraints (2.28), we have that $-d\ \mu/d = -\mu$ which is clearly $\geq -\mu$.

Therefore, the dual solution proposed is feasible when (2.31) holds.

For both cases ((2.30) and (2.31)) the primal feasible solutions proposed are optimal because the objective values of the primal MIP solution and of the solution of the dual of the relaxation coincide.

When (2.30) holds, the objective function value of the dual is equal to 0 and is the same as the corresponding primal MIP one.

When (2.31) holds, the objective function of the dual is $|T|\left(1-\frac{p\mu}{d}\right)$, and this is also the objective value of the primal MIP given in (2.32).

This proves the optimality for both solutions, and that the duality gap is zero. $\square$

Remark: If (2.30) holds, it can be shown that the zero solution, i.e. $z_{t,\Omega(t_1)} = 0$ for all $t \in \{1,\ldots,T\}, t_1 \in \{t,\ldots,t+d-1\}$ and $\tau_t = 0$ for all $t \in \{1,\ldots,T\}$, is optimal for the MIP problem. We give here another direct argument of optimality for the zero solution.

The objective of the maximization problem is greater or equal to 0, simply because the zero solution is feasible

$$\max \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)} - \mu \sum_{t=1}^{T} \tau_t \geq 0. \tag{2.33}$$

Moreover, at each time slot, only one batch task can begin and finish and a batch task can last for at most d time slots. Therefore, at most d batch tasks can be active during one time slot. Therefore, a lower bound for the cycle duration can be expressed as follows :

$$\frac{p}{d} \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)} \leq \sum_{t=1}^{T} \tau_t. \tag{2.34}$$

This inequality is valid for the problem formulation, because $p \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)}$ is the total processing time over the cycle, and at most d batches can be processed in parallel at any moment. By combining (2.33) and (2.34), we have that :

$$\frac{p}{d} \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)} \leq \sum_{t=1}^{T} \tau_t \leq \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)}/\mu.$$

must be satisfied by any solution with a non negative objective function value. Because of the condition (2.30), the only possibility to obtain a feasible solution with a non negative objective value is that $z_{t,\Omega(t_1)} = 0$ for all $t \in \{1,\ldots,T\}, t_1 \in \{t,\ldots,t+d-1\}$ and therefore because of the equation (2.33), we have also that $\tau_t = 0$ for all $t \in \{1,\ldots,T\}$ in the optimal solution.

In conclusion, we have shown that we need only the two constraints (2.23) and (2.24) of the relaxation of the MIP problem (2.14)-(2.21) in order to obtain a zero duality gap. However, the other facet defining inequalities are important in order to obtain a feasible solution for the initial problem and to reduce the number of fractional components in the solution of the LP relaxation of (2.14)-(2.21). They are also important for more complex problems involving more tasks and resource constraints. We address this question in the computational experiments in Chapter 3.

Capacited case

In this section, we look at the quality of the initial formulation when we limit the number of units on which multiple batches of the single task can be processed in parallel. We show that by adding an appropriate lower bound on the cycle duration, it is possible to keep a linear formulation with zero duality gap.

The initial formulation for this cyclic scheduling problem composed of one batch task and a limited number of units is recalled below :

1B/0C/CAP//INIT

$$\max \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)} - \mu \sum_{t=1}^{T} \tau_t \tag{2.35}$$

$$p z_{t,\Omega(l)} \leq \sum_{k=t}^{l} \tau_{\Omega(k)} \qquad \forall t, l : 1 \leq t \leq T, t \leq l \leq t+d-1 \tag{2.36}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} \leq p z_{t,\Omega(l)} + p(l-t+1)(1-z_{t,\Omega(l)}) \qquad \forall t, l : 1 \leq t \leq T, t \leq l \leq t+d-1 \tag{2.37}$$

$$\sum_{l=t}^{t+d-1} z_{t,\Omega(l)} \leq 1 \qquad \forall t : 1 \leq t \leq T \tag{2.38}$$

$$\sum_{t=l-d+1}^{l} z_{\Omega(t),l} \leq 1 \qquad \forall l : 1 \leq l \leq T \tag{2.39}$$

$$\sum_{t=1}^{T} \sum_{\substack{l=t | t \leq t' \leq l \\ \text{or } t \leq t'+T \leq l}}^{t+d-1} z_{t,\Omega(l)} \leq nbr\_unit \qquad \forall t' : 1 \leq t' \leq T \tag{2.40}$$

$$z_{t,\Omega(l)} \in \{0,1\} \forall t, l : 1 \leq t \leq T, t \leq l \leq t+d-1, \quad \tau_t \geq 0 \forall t : 1 \leq t \leq T \tag{2.41}$$

In this formulation, and in this Section, we assume that $nbr\_unit < d$. This means that d must be large enough to allow to use the full capacity. Because otherwise constraint (2.40) is implied by (2.38)-(2.39), (2.40) is redundant, and the capacitated case is identical to the uncapacitated one.

Since the batch task can be performed on every processing unit, the following constraint (2.42) defines a lower bound on the cycle duration by taking into account the fact that the time needed to complete all the batches (i.e. the duration of the cycle) is greater or equal to the time needed if the batches and processing time are evenly allocated among the nbr_unit units.

We add to the initial formulation (2.35)-(2.41) the constraint :

$$\frac{p}{nbr\_unit}\sum_{t=1}^{T}\sum_{t'=t}^{t+d-1} z_{t,\Omega(t')} - \sum_{t=1}^{T}\tau_t \leq 0. \tag{2.42}$$

Let X^C denote the set of feasible solutions to (2.36)-(2.41).

Proposition 2.3.13. *Constraint (2.42) is valid for X^C*

Proof. This constraint defines a valid lower bound on the cycle duration because this lower bound assumes 100 % utilization of the reactors in order to perform the total processing time $p\sum_{t=1}^{T}\sum_{t'=t}^{t+d-1} z_{t,\Omega(t')}$. □

We choose a relaxation of the problem (2.35)-(2.42). Then, we give the formulation of the dual of its LP relaxation. Finally, we prove that the duality gap is zero.

We construct a relaxation of the problem (2.35)-(2.42) composed of (2.35), (2.38), (2.41) and (2.42). Its LP relaxation is

$$\max \sum_{t=1}^{T}\sum_{l=t}^{t+d-1} z_{t,\Omega(l)} - \mu\sum_{t=1}^{T}\tau_t \tag{2.43}$$

$$st \quad \sum_{l=t}^{t+d-1} z_{t,\Omega(l)} \leq 1 \qquad \forall t : 1 \leq t \leq T \tag{2.44}$$

$$\frac{p}{nbr\_unit}\sum_{t=1}^{T}\sum_{t'=t}^{t+d-1} z_{t,\Omega(t')} - \sum_{t=1}^{T}\tau_t \leq 0 \tag{2.45}$$

$$z \geq 0, \tau \geq 0 \tag{2.46}$$

Then, we give the dual of the LP relaxation (2.43)-(2.46). The dual variables associated with the constraints (2.44) are u_t, for all t, and with the constraint (2.45) is w.

The dual problem is formulated as :

$$\min \sum_{t=1}^{T} u_t \tag{2.47}$$

$$st \quad u_t + \Big(\frac{p}{nbr\_unit}\Big) w \geq 1 \;\forall t \in \{1, \dots, T\} \tag{2.48}$$

$$-w \geq -\mu \tag{2.49}$$

$$u, w \geq 0 \tag{2.50}$$

In order to prove that the duality gap is zero, we take a feasible solution of the mixed integer program (2.35)-(2.42) and a feasible solution of the dual (2.47)-(2.50) and we show that the two objective function values are equal.

Proposition 2.3.14. *If $nbr\_unit < d$, the duality gap is zero, i.e., problems (2.35)-(2.42) and (2.47)-(2.50) have the same optimal objective value.*

Proof. We distinguish two cases : $p\mu \leq nbr\_unit$ and $p\mu > nbr\_unit$.

If $p\mu \leq nbr\_unit$, then the primal solution that we propose is the following: for all $t \in \{1, \ldots, T\}$, $l \in \{t, \ldots, t+d-1\}$: $z_{t,\Omega(l)} = 1$ if $l = t + nbr\_unit - 1$ and $z_{t,\Omega(l)} = 0$ otherwise and for all $t \in \{1, \ldots, T\}$: $\tau_t = p/nbr\_unit$. The corresponding objective value is :

$$|T| - \mu \frac{p|T|}{nbr\_unit} \tag{2.51}$$

We show that this mixed integer solution is feasible for the MIP capacitated problem (2.36)-(2.42).

For the constraint (2.36), if $l \neq t + nbr\_unit - 1$ then the validity condition reduces to : $\sum_{k=t}^{l} \tau_{\Omega(k)} \geq 0$, and this is clearly always valid. Otherwise, the validity condition is $p \leq (l-t+1)p/nbr\_unit = p$, and this is also valid. The constraints (2.38)-(2.42) are trivially satisfied. Finally, for the constraints (2.37), if $l \neq t + nbr\_unit - 1$, then the validity condition becomes : $(l-t+1)p/nbr\_unit \leq p(l-t+1)$ and this inequality is valid. If $l = t + nbr\_unit - 1$ then the validity condition becomes $(l-t+1)p/nbr\_unit = p \leq p$ and this is clearly valid.

We propose the following dual solution : $w = \mu$, $u_t = 1 - \frac{p\mu}{nbr\_unit}$ $\forall t \in \{1, \ldots, T\}$. It can easily be checked that this solution is feasible for the dual problem (2.47)-(2.50) if $p\mu \leq nbr\_unit$.

We can check that the dual objective value for the dual feasible solution proposed is : $|T|(1 - \frac{p\mu}{nbr\_unit})$ and has the same objective function value as the primal one (2.51).

Therefore, there is no duality gap when using formulation (2.43)-(2.46). This implies that the primal solution proposed is optimal for the problem.

Finally, if $p\mu > nbr\_unit$ the primal solution that we propose is $z_{t,\Omega(t')} = 0 \forall t \in \{1, \ldots, T\}$, $t' \in \{t, \ldots, t+d-1\}$ and $\tau_t = 0 \forall t \in \{1, \ldots, T\}$. This zero solution is trivially valid, and has objective value 0.

If $p\mu > nbr\_unit$, a feasible solution for the dual is the following : $w = \frac{nbr\_unit}{p}$ and $u = 0$. The objective function values corresponding to the primal and the dual solutions defined above are both zero.

This proves that there is no duality gap when using formulations (2.35)-(2.42) or (2.43)-(2.46) for solving the capacitated problem in both cases, $p\mu \leq nbr\_unit$ and $p\mu > nbr\_unit$. □

Remark 1: If $p\mu > nbr\_unit$, the zero solution, i.e. $z_{t,\Omega(t')} = 0 \forall t \in \{1, \ldots, T\}$, $t' \in \{t, \ldots, t+d-1\}$ and $\tau_t = 0 \forall t \in \{1, \ldots, T\}$ is optimal for the MIP problem by Proposition 2.3.14. We give here another direct argument of optimality for the zero solution. As we have a maximization problem, the optimal objective function value has to be at least 0.

$$\max \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)} - \mu \sum_{t=1}^{T} \tau_t \geq 0. \tag{2.52}$$

By combining the constraint (2.42) and the constraint (2.52), we get :

$$\frac{p}{nbr\_unit}\sum_{t=1}^{T}\sum_{l=t}^{t+d-1} z_{t,\Omega(l)} \le \sum_{t=1}^{T}\tau_t \le \sum_{t=1}^{T}\sum_{l=t}^{t+d-1} z_{t,\Omega(l)}/\mu.$$

Because of the condition $p\mu > nbr\_unit$, the only possibility to obtain a feasible solution with non negative objective value in that case is that $z_{t,\Omega(t')} = 0 \forall t \in \{1,\ldots,T\}$, $t' \in \{t,\ldots,t+d-1\}$ and therefore, because of the equation (2.52), we must also have that $\tau_t = 0 \forall t \in \{1,\ldots,T\}$.

Remark 2: Although there is no duality gap in the capacitated case when using the initial formulation, plus the valid inequality (2.42), the strengthened inequalities found in the uncapacitated case remain valid in the capacitated case, and may be useful to reduce the number of fractional components of the LP relaxation solution for more complex capacitated problems.

2.3.2 Valid Inequalities : variable lower bounds on the cycle duration

The idea presented in the valid inequality (2.42) can be generalized for the uncapacitated and capacitated cases. We prove this result only for the capacitated case since the uncapacitated case is a special case with a number of units equals to d.

For every $t \in \{1,\ldots,T\}$ and $l \in \{t+2,\ldots,t+T+d-3\}$, we define a vector $coeff$ whose components are defined by

$$\begin{aligned} coeff_k \quad &= \min(k-t+1; l-k+1) \qquad \forall k \in \{t,\ldots,l\} \\ &= 0 \text{ otherwise} \end{aligned}$$

For fixed t and l, the following valid inequality can be obtained :

$$\sum_{k=t}^{l} \min(coeff_k, nbr\_unit, d)\tau_{\Omega(k)} \ge p\left(\sum_{t_1=t}^{l}\sum_{t_2=t_1}^{\min(l,t_1+d-1)} z_{\Omega(t_1),\Omega(t_2)}\right). \tag{2.53}$$

Proposition 2.3.15. *Constraint (2.53) is valid for X^C.*

Proof. We consider only tasks that start and end within the interval given by time slot t to time slot l. Since at most one task can start at each time slot, then for every $k \in \{t,\ldots,l\}$, the maximum number of tasks that can start in the interval $\{t,\ldots,l\}$ at or before slot k is $k-t+1$. Moreover, since at most one task can finish at each time slot, the maximum number of tasks that can end at or after slot $k \in \{t,\ldots,l\}$ in the interval $\{t,\ldots,l\}$ is $l-k+1$. By taking the minimum of these two values, we get the maximum number of tasks, starting and finishing in the time slot interval $\{t,\ldots,l\}$, that can be active during time slot $k \in \{1,\ldots,T\}$. Moreover, the maximum number of batch tasks that can be active at a given time slot has to be smaller than the number of units, otherwise we do not satisfy the capacity restrictions. Finally, since a batch task can last only for d time slots and since at each time slot at most one batch task can start and at most one batch task can finish, this implies that at each time slot at most d batch tasks can be processed in parallel. This is why in a given interval $\{t, \dots, l\}$, the number of batches produced during a given time slot cannot be more than the minimum value between $coeff_k$, nbr_unit and d. Then, the inequality (2.53) is valid because the left-hand-side is the maximum amount of processing time available in slots t up to l for batches that are completely processed from t to l, and the right-hand-side is the amount of processing time used in $\{t, \dots, l\}$ for such batches. □

We show on an example that this valid inequality is useful.

Example 2.3.16. *We considered the cyclic scheduling problem composed of one batch task and a limited number of units. The MIP formulation used is composed of (2.35)-(2.42) for which constraints (2.36) are remplaced by (2.8)-(2.9) and constraints (2.37) are remplaced by (2.10)-(2.11). The characteristics of the instance are $T = 4$, $d = 4$, $nbr\_unit = 3$, $\mu = 0.5$ and $p = 3$. An optimal solution of the linear relaxation of this problem for the tasks that start and finish within the interval given by time slot 1 to time slot 4 is :*

$$z^* = \begin{pmatrix} 0.083 & 0.166 & 0.5 & 0.25 \\ / & 0.083 & 0.166 & 0.416 \\ / & / & 0 & 0.166 \\ / & / & / & 0.166 \end{pmatrix}$$

The optimal time slot durations are $\tau^* = (1.5, 0.75, 0, 1.75)$. The valid inequality (2.53) for $t = 1$ and $l = 4$ is the following :

$$\begin{aligned} \tau_1 + 2\tau_2 + 2\tau_3 + \tau_4 \quad \geq \quad & 3(z(1,1) + z(1,2) + z(1,3) + z(1,4) + z(2,2) + z(2,3)) \\ & +3(z(2,4) + z(3,3) + z(3,4) + z(4,4)) \qquad (2.54) \end{aligned}$$

The optimal solution of the linear relaxation given above violates the valid inequality (2.54) since the left-hand-side equal 4.75 and the right-hand-side equal 6.

Although the duality gap is zero without (2.53), this example shows that the valid inequality (2.53) is useful since the fractional optimal LP solution is cut off by (2.53).

We can generalize the inequality (2.53). For all $t \in \{1, \dots, T\}, l \in \{t+2, \dots, t+T+d-3\}$, and $c1 \in \{1, \dots, \min(d, nbr\_unit)\}$, we have that

$$p \sum_{t_1=t}^{l} \sum_{t_2=t_1 | t_2-t_1+1 \leq c1}^{\min(l, t_1+d-1)} z_{\Omega(t_1),\Omega(t_2)} \leq \sum_{k=1}^{T} \min(coeff_k, c1)\tau_k. \qquad (2.55)$$

The proof of validity of (2.55) is similar to the one given for (2.53). We restrict here the set of binary variables z that start and end in the interval $\{t, \dots, l\}$ by considering only the variables that can last for maximum $c1$ time slots. Therefore, the maximum number of such batches that can be active at a given time slot in $\{t, \dots, l\}$ has to be smaller or equal to $c1$. This inequality and its use deserve further investigation.

2.4 Multiple batch tasks

We consider now the special case of the general problem composed of several batch tasks where the objective is to maximize a measure of the productivity of the plant. We first extend the initial formulation proposed for the one batch task problem in order to have a correct model formulation for this problem. Then, we show how to strengthen the formulation, and finally we extend some valid inequalities obtained for the one batch task problem.

We consider here a direct extension of the single task model, in which several batch tasks $i \in BT$ must be performed in parallel on *nbr_units*, in order to maximize the weighted sum of the number of batch tasks produced over a cycle, minus μ times the cycle duration. Each batch task i has its own processing time p_i.

2.4.1 Initial and strengthened formulation

As a direct extension of the single task model (2.1)-(2.7), the initial formulation proposed for the multiple batch task problem is the following :

nB/0C/CAP//INIT

$$\max \sum_{i\in BT}\sum_{t=1}^{T}\sum_{l=t}^{t+d-1} w_i z_{i,t,\Omega(l)} - \mu\sum_{t=1}^{T}\tau_t \tag{2.56}$$

$$p_i z_{i,t,\Omega(l)} - \sum_{k=t}^{l}\tau_{\Omega(k)} \le 0 \qquad \forall i\in BT, t\in\{1,\dots,T\}, l\in\{t,\dots,t+d-1\} \tag{2.57}$$

$$\sum_{k=t}^{l}\tau_{\Omega(k)} \le p_i z_{i,t,\Omega(l)} + p^{\max}(l-t+1)(1-z_{i,t,\Omega(l)}) \qquad \forall i\in BT, t\in\{1,\dots,T\}, l\in\{t,\dots,t+d-1\} \tag{2.58}$$

$$\sum_{i\in BT}\sum_{l=t}^{t+d-1} z_{i,t,\Omega(l)} \le 1 \qquad \forall t\in\{1,\dots,T\} \tag{2.59}$$

$$\sum_{i\in BT}\sum_{t=l-d+1}^{l} z_{i,\Omega(t),l} \le 1 \qquad \forall l\in\{1,\dots,T\} \tag{2.60}$$

$$\sum_{i\in BT}\sum_{t=1}^{T}\sum_{\substack{l=t|t\le t'\le l\\ \text{or } t\le t'+T\le l}}^{t+d-1} z_{i,t,l} \le nbr\_unit \qquad \forall t'\in\{1,\dots,T\} \tag{2.61}$$

$$z_{i,t,\Omega(l)} \in\{0,1\}\forall i\in BT, t\in\{1,\dots,T\}, l\in\{t,\dots,t+d-1\}, \quad \tau_t\ge 0 \qquad \forall t\in\{1,\dots,T\} \tag{2.62}$$

where $p^{\max} = \max_{i\in BT} p_i$ is the maximum processing time of a task and w_i corresponds to the weight of batch task i in the objective function. Here, $\tau_t \le p^{\max}$

is implied by (2.58) with $l = t$, for all t.

By considering the improvement obtained for the formulation of the one batch task problem, and by using strengthening techniques, see Andersen and Pochet [1], on small instances of such a problem, we have been able to strengthen the constraints (2.57) and (2.58).

We denote by X^M the set of feasible solutions to (2.57)-(2.62).

Strengthened formulation

By using the strengthening techniques presented in Section 1.3.2, we have obtained that the constraint (2.57) can be strengthened as follows :

$$\sum_{i \in BT} \sum_{k=t|t \neq l}^{l} p_i z_{i,t,\Omega(k)} + \sum_{i \in BT} p_i z_{i,\Omega(l),\Omega(l)} - \sum_{k=t}^{l} \tau_{\Omega(k)} \leq 0, \tag{2.63}$$

and

$$\sum_{i \in BT} \sum_{k=t|t \neq l}^{l} p_i z_{i,\Omega(k),\Omega(l)} + \sum_{i \in BT} p_i z_{i,t,t} - \sum_{k=t}^{l} \tau_{\Omega(k)} \leq 0 \tag{2.64}$$

for all $t \in \{1, \ldots, T\}, l \in \{t, \ldots, t + d - 1\}$.

The constraints (2.63)-(2.64) are straightforward extensions of the single task constraints (2.8) and (2.9). Since only one batch task can start at each time slot and only one can finish at each time slot, we can aggregate the inequalities (2.57), and sum the processing times, over all the batch tasks. This allows one to strengthen inequality (2.57) and to reduce the number of constraints by a factor $|BT|$.

Proposition 2.4.1. *The constraints (2.63)-(2.64) are valid for X^M.*

Proof. The proofs of validity are similar to the proofs of (2.8)-(2.9). $\square$

We extend the example proposed for the problem composed of one batch task, in order to illustrate the inequality (2.63) for the problem composed of two (or more) batch tasks. The inequalities (2.57) for $t = 1$, $l = 3$ are the following,

$$\begin{aligned} \tau_1 + \tau_2 + \tau_3 &\geq p_1 z_{1,1,3} \\ \tau_1 + \tau_2 + \tau_3 &\geq p_2 z_{2,1,3} \end{aligned}$$

the strengthened single item inequalities (2.8) are

$$\begin{aligned} \tau_1 + \tau_2 + \tau_3 &\geq p_1 (z_{1,1,1} + z_{1,1,2} + z_{1,1,3}) + p_1 z_{1,3,3} \\ \tau_1 + \tau_2 + \tau_3 &\geq p_2 (z_{2,1,1} + z_{2,1,2} + z_{2,1,3}) + p_2 z_{2,3,3} \end{aligned}$$

and the corresponding strengthened inequality (2.63) is

$$\begin{aligned} \tau_1 + \tau_2 + \tau_3 \geq\; & p_1 (z_{1,1,1} + z_{1,1,2} + z_{1,1,3}) + p_1 z_{1,3,3} \\ & + p_2 (z_{2,1,1} + z_{2,1,2} + z_{2,1,3}) + p_2 z_{2,3,3}, \end{aligned}$$

which is stronger because the processing times are summed in the right hand-side.

Again, by using the strengthening techniques presented in Section 1.3.2, we have obtained that the constraint (2.58) can be strengthened as follows :

$$\begin{aligned}\sum_{k=t}^{l} \tau_{\Omega(k)} \leq & \sum_{i\in BT} \sum_{k=l-d+1}^{l} p_i z_{i,\Omega(k),\Omega(l)} \\ & + p^{\max}\Big((l-t+1) - \sum_{i\in BT} \sum_{k=l-d+1}^{l} \min\{l-t+1, l-k+1\} z_{i,\Omega(k),\Omega(l)}\Big) \\ & + \sum_{i\in BT} (p_i - p^{\max}) \sum_{t'=t \mid l=t+d-1}^{l-1} z_{i,t,\Omega(t')} \qquad (2.65)\end{aligned}$$

for all $t, l : 1 \leq t \leq T, t \leq l \leq t+d-1$, and

$$\begin{aligned}\sum_{k=t}^{l} \tau_{\Omega(k)} \leq & \sum_{i\in BT} \sum_{k=t}^{t+d-1} p_i z_{i,t,\Omega(k)} \\ & + p^{\max}\Big((l-t+1) - \sum_{i\in BT} \sum_{k=t}^{t+d-1} \min\{l-t+1, k-t+1\} z_{i,t,\Omega(k)}\Big) \\ & + \sum_{i\in BT} (p_i - p^{\max}) \sum_{t'=t+1 \mid l=t+d-1}^{l} z_{i,\Omega(t'),\Omega(l)} \qquad (2.66)\end{aligned}$$

for all $t, l : 1 \leq t \leq T, t \leq l \leq t+d-1$.

The constraints (2.65)-(2.66) are also extensions of the single task constraints (2.10) and (2.11), plus a last strengthened term for the case where $l = t+d-1$. Again, by summing over all batch tasks, the inequalities (2.58) can be strengthened, and the number of inequalities reduced by a factor $|BT|$.

Proposition 2.4.2. *Constraint (2.65) is valid for X^M.*

Proof. Let $t \in \{1, \ldots, T\}$ and $l \in \{t, \ldots, t+d-1\}$. First observe that, $\sum_{i\in BT} \sum_{t'=t}^{l-1} z_{i,t,\Omega(t')} \leq 1$ by (2.59), and $\sum_{i\in BT} \sum_{k=l-d+1}^{l} z_{i,\Omega(k),\Omega(l)} \leq 1$ by (2.60).

(i) We first consider the case where $l = t+d-1$ and $z_{i,t,\Omega(t')} = 1$ for some $i \in BT$ and $t' \in \{t, \ldots, l-1\}$. We decompose this case into two subcases.

Case 1 : $z_{j,\Omega(t_1),\Omega(l)} = 1$ for some $j \in BT$ and $t_1 \in \{l-d+1, \ldots, l\}$.

Note that $t_1 \neq t$ because at most one batch can start in t. We know that $t_1 > t$ since $l = t+d-1$ and $t_1 \neq t$. This implies $\min\{l-t+1, l-t_1+1\} = l-t_1+1$. There are no condition on the relative position of time slots t' and t_1.

Since constraint (2.58) with $l = t$ imposes that the duration of a time slot t is bounded by $p^{\max}$, i.e., $\tau_t \leq p^{\max}$, this implies that $\sum_{k=t}^{l} \tau_{\Omega(k)} \leq p_i + p_j +$

$p^{\max}\max(t_1 - t' - 1; 0)$ has to be satisfied, whatever the relative position of t_1 and t', see the two possible cases (a) and (b) in Figure 2.6. Therefore we bound this

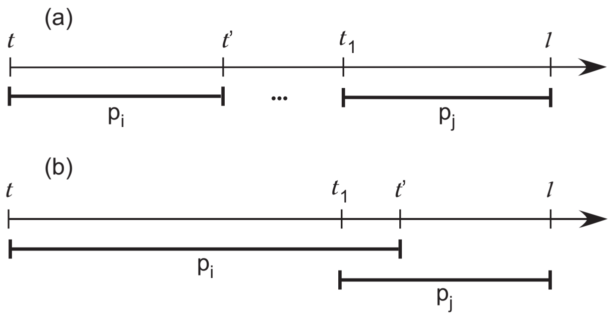

Figure 2.6: The two possible cases for t'

expression as follows : $\sum_{k=t}^{l} \tau_{\Omega(k)} \leq p_i + p_j + p^{\max}(t_1 - t - 1)$ and this is precisely the constraint (2.65) for that case.

Case 2 : $z_{j,\Omega(t_1),\Omega(l)} = 0$ $\forall j \in BT$ and $\forall t_1 \in \{l - d + 1, \dots, l\}$.

Constraint (2.65) becomes $\sum_{k=t}^{l} \tau_{\Omega(k)} \leq p_i + p^{\max}(l - t)$ and this is valid in this case.

(ii) Then, we consider the case where $l < t + d - 1$ and $z_{i,t,\Omega(t')} = 1$ for some $i \in BT$ and $t' \in \{t, \dots, l - 1\}$.

Again, at most one task finishing in time period l can start. Suppose that $z_{j,\Omega(t_1),\Omega(l)} = 1$ for some $j \in BT$ and $t_1 \in \{l - d + 1, \dots, l\}$. If $t_1 < t$ then Constraint (2.65) reduces to $\sum_{k=t}^{l} \tau_{\Omega(k)} \leq p_j$ and this is clearly valid since in that case $\sum_{k=t_1}^{l} \tau_{\Omega(k)} = p_j$. If $t_1 \geq t$ then Constraint (2.65) reduces to $\sum_{k=t}^{t_1-1} \tau_{\Omega(k)} + p_j \leq p_j + p^{\max}((l - t + 1) - (l - t_1 + 1))$ and this is equivalent to $\sum_{k=t}^{t_1-1} \tau_{\Omega(k)} \leq p^{\max}(t_1 - t)$, which is valid since $\tau_t \leq p^{\max}$ for each $t \in \{1, \dots, T\}$.

If $z_{j,\Omega(t_1),\Omega(l)} = 0$ $\forall t_1 \in \{l - d + 1, \dots, l\}$, then the constraint becomes $\sum_{k=t}^{l} \tau_{\Omega(k)} \leq p^{\max}(l - t + 1)$ and this is clearly valid since we know that for every $t \in \{1, \dots, T\}$, $\tau_t \leq p^{\max}$.

(iii) Finally, we consider the case where $l \leq t + d - 1$ and $z_{i,t,\Omega(t')} = 0$ for all $i \in BT$ and for all $t' \in \{t, \dots, l - 1\}$.

This case is equivalent to the case (ii) and the validity of the constraint in this case is proved by using exactly the same arguments as the ones used for (ii). □

Proposition 2.4.3. *Constraint (2.66) is valid for X^M.*

Proof. The proof is similar to the one given for the validity of constraint (2.65). □

We illustrate inequalities (2.65)-(2.66) with an example. Based on the example proposed in order to illustrate the inequality (2.3), we assume that there are two (or more) batch tasks. The inequalities (2.58) with $t = 1$, $l = 2$, $d = 3$ are the following

$$\begin{aligned} \tau_1 + \tau_2 \leq & \quad p_1 z_{1,1,2} + p^{\max}(2 - 2z_{1,1,2}) \\ \tau_1 + \tau_2 \leq & \quad p_1 z_{2,1,2} + p^{\max}(2 - 2z_{2,1,2}), \end{aligned}$$

and the strengthened inequality (2.66) is

$$\begin{aligned} \tau_1 + \tau_2 \leq & \quad p_1(z_{1,1,1} + z_{1,1,2} + z_{1,1,3}) + p^{\max}(2 - z_{1,1,1} - 2z_{1,1,2} - 2z_{1,1,3}) \\ & \quad + p_2(z_{2,1,1} + z_{2,1,2} + z_{2,1,3}) + p^{\max}(-z_{2,1,1} - 2z_{2,1,2} - 2z_{2,1,3}) \end{aligned}$$

which is stronger than the two inequalities above because the terms added in the right hand side are non positive.

Finally, if we assume that we have two batch tasks and that $d = 3, t = 1$ and $l = 3$ $(= t + d - 1)$, the strengthened inequality (2.66) for that example is written as

$$\begin{aligned} \tau_1 + \tau_2 + \tau_3 \leq & \quad p_1(z_{1,1,1} + z_{1,1,2} + z_{1,1,3}) + p^{\max}(3 - z_{1,1,1} - 2z_{1,1,2} - 3z_{1,1,3}) \\ & \quad + (p_1 - p^{\max})(z_{1,2,3} + z_{1,3,3}) \\ & \quad + p_2(z_{2,1,1} + z_{2,1,2} + z_{2,1,3}) + p^{\max}(-z_{2,1,1} - 2z_{2,1,2} - 3z_{2,1,3}) \\ & \quad + (p_2 - p^{\max})(z_{2,2,3} + z_{2,3,3}). \end{aligned}$$

2.4.2 Successive Batch Tasks

We model here the case where the different batch tasks have to be performed one after the other in a fixed sequence. Such sequence restrictions may involve only a subset of the batch tasks, or all of them. For simplicity, we assume here that there is a single sequence involving all batch tasks. That is, the tasks must be performed in the sequence $i = 1, 2, \ldots, |BT|$. Such a situation occurs typically when the tasks are successive production stages of a single process, where the final product is produced at the end of the last task $|BT|$. The sequence restriction between batch tasks is modeled by resource constraints. We define for every batch task i an associated resource r_i, $i \in BT$, as well as a variable $w_{r_i,t}$ to indicate the availability of resource r_i at the beginning of time slot t for all $i \in BT$ and all $t \in \{1, \ldots, T\}$. When a batch of task i starts in time slot t, one unit of available resource r_i is taken from $w_{r_i,t-1}$. Similarly, when a batch of task i finishes at the end of time slot l, one unit of resource r_{i+1} is added to $w_{r_{i+1},l}$, and becomes available to perform the next task $i + 1$, from time slot $l + 1$ on, or later.

Example 2.4.4. *We consider the following precedence between three batch tasks i_1,i_2 and i_3. If batch task i_1 starts, a unit of the associated resource (r_{i_1}) is consumed and at the end of the batch task i_1, since batch task i_2 has to be processed after batch task i_1, the unit of resource is released in the resource associated to the batch task i_2 (r_{i_2}). At the end of batch task i_2, since batch task i_3 has to be processed after batch task i_2, the unit of resource is released in the resource* *associated to batch task i_3 (r_{i_3}). Finally, at the end of batch task i_3, since batch task i_1 has to be process after batch task i_3, the unit of resource is released in the resource associated to batch task i_1 (r_{i_1}). We represent the resource task network of this example in Figure 2.7.*

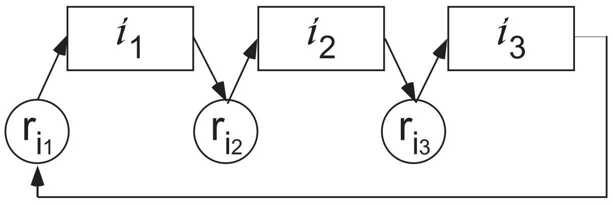

Figure 2.7: The resource task network for modeling the fixed sequence between batch tasks

We use the classical way of modeling resources and tasks, by using a resource task network, see Pantelides [35]. The material balance constraints used to model the availability of resources can be written as follows :

$$w_{r_1,t} = w_{r_1,\Omega(t-1)} + \sum_{t'=t-d}^{t-1} z_{|BT|,\Omega(t'),\Omega(t-1)} - \sum_{t'=t}^{t+d-1} z_{1,t,\Omega(t')} \tag{2.67}$$

$\forall t : 1 \leq t \leq T$, and

$$w_{r_i,t} = w_{r_i,\Omega(t-1)} + \sum_{t'=t-d}^{t-1} z_{i-1,\Omega(t'),\Omega(t-1)} - \sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')} \tag{2.68}$$

$\forall i, t : 2 \leq i \leq |BT|, 1 \leq t \leq T$.

We suppose that each batch task can be performed on every processing unit and has to be processed after its preceding batch task (where the predecessor of batch task 1 is batch task BT).

At each time slot, there are at most *nbr_unit* units available for performing the batch tasks.

$$0 \leq w_{r_i,t} \leq \textit{nbr\_unit} \qquad \forall i, t : 1 \leq i \leq |BT|, 1 \leq t \leq T. \tag{2.69}$$

At each time slot, a batch task is performed on a unit or the unit is in stand by. This is imposed as follows :

$$\sum_{i \in BT} \sum_{t=1}^{T} \sum_{\substack{l=t|t\leq t'\leq l \\ \text{or } t\leq t'+T\leq l}}^{t+d-1} z_{i,t,\Omega(l)} + \sum_{i=1}^{|BT|} w_{r_i,t'} = \textit{nbr\_unit} \ \forall t' \in \{1, \ldots, T\} \tag{2.70}$$

Finally, if each batch task appears once in the resource task network and has to be processed after another batch task, then each batch task has to be performed the same number of times since the scheduling problem is cyclic. Therefore, the maximum number of times that each batch task can be performed is limited by the total number of time slots. We can express this as follows :

$$\sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{i,t,\Omega(l)} \leq \left\lfloor \frac{T}{|BT|} \right\rfloor \forall i \in BT \tag{2.71}$$

No waiting time between batch tasks

In some cases, some batch tasks have to be performed directly after others, without any waiting time. For instance, in a chemical reactor, regulation has to start immediately after heating. If task $i+1$ has to be performed after task i, without any waiting, this restriction can be modeled as follows :

$$\sum_{l=t}^{t+d-1} z_{i+1,t,\Omega(l)} = \sum_{l=t-d}^{t-1} z_{i,\Omega(l),\Omega(t-1)}, \tag{2.72}$$

and for all $t \in \{1, \ldots, T\}$.

This constraint is the reason why we need to know or to model exactly the duration of a batch task as the sum of time slot durations.

2.4.3 Valid inequalities : lower bounds on the cycle duration

By aggregating over all the batch tasks, we can extend the valid inequality (2.42) as

$$\sum_{i \in BT} \frac{p_i}{nbr\_unit} \sum_{t=1}^{T} \sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')} - \sum_{t=1}^{T} \tau_t \leq 0. \tag{2.73}$$

Again, by aggregating over all the batch tasks, we can also extend the valid inequality (2.53). For every $t \in \{1, \ldots, T\}$ and $l \in \{t+2, \ldots, t+T+d-3\}$, we define the vector $coeff$ whose components are defined by

$$\begin{aligned} coeff_k \quad & = \min(k-t+1; l-k+1) \qquad \forall k \in \{t, \ldots, l\} \\ & = 0 \text{ otherwise} \end{aligned}$$

and the extended constraint is

$$\sum_{k=t}^{l} \min(coeff_k, nbr\_unit, d)\tau_k \geq \sum_{i \in BT} p_i \left(\sum_{t_1=t}^{l} \sum_{t_2=t_1}^{\min(l,t_1+d-1)} z_{i,\Omega(t_1),\Omega(t_2)} \right) \tag{2.74}$$

Finally, in the same way, we can extend the generalization of the inequality (2.53). For all $t \in \{1, \ldots, T\}, l \in \{t+2, \ldots, t+T+d-3\}$, and $c1 \in \{1, \ldots, \min(d, nbr\_unit)\}$, we have that

$$\sum_{i \in BT} \sum_{t_1=t}^{l} \sum_{t_2=t_1 | t_2-t_1+1 \leq c1}^{\min(l,t_1+d-1)} p_i \; z_{i,\Omega(t_1),\Omega(t_2)} \leq \sum_{k=t}^{l} \min(coeff_k, c1)\tau_k. \tag{2.75}$$

Proposition 2.4.5. *Constraints (2.73)-(2.75) are valid for X^M.*

Proof. The proof of validity for these three extended constraints is a trivial extension of the proof of validity given for the one batch task problem. □

2.5 Mixed plant : batch and continuous tasks with resource restrictions

We introduce now the continuous tasks in the model formulation, as well as the general resource restrictions. We study the special case composed of multiple batch tasks with fixed sequences as defined in Section 2.4.2, multiple continuous tasks and the sharing of some resources. The batch and the continuous tasks consume and produce some resources r such as materials (stored in tanks) ($r \in R_m$), utilities ($r \in R_u$), and processing equipment ($r \in R_e$) involved in the process.

We consider here an extension of the multiple task model, in which several continuous tasks $j \in CT$ must be performed and several resource restrictions must be satisfied, and the objective is to maximize the output products of the plant, over a cycle, minus μ times the cycle duration. Each continuous task j has its own lower and upper processing rate $\underline{\rho}_j$ and $\overline{\rho}_j$, respectively.

2.5.1 Model formulation

In order to model such a case, we introduce two new indices

j: is the index of a continuous task, $j \in CT$

r: is the index of a resource, $r \in R_m \cup R_u \cup R_e$

and three new types of decision variables :

$q_{j,t}$: is the quantity processed by the continuous task j during time slot t $[ru]$, $q_{j,t} \geq 0$.

$w_{r,t}$: is the rate $[ru/h]$ or the quantity $[ru]$ of resource r available at the beginning of time slot t, $w_{r,t} \geq 0$.

$wf_{r,t}$: is the rate $[ru/h]$ or the quantity $[ru]$ of resource r available at the end of time slot t, $wf_{r,t} \geq 0$.

The continuous tasks

When the continuous task is processing, this task cannot be stopped.

In a cyclic scheduling optimization problem, each continuous task has thus to be active all the time.

The processing constraints for continuous tasks can be written as follows :

$$q_{j,t} \leq \overline{\rho}_j \tau_t \qquad \forall j,t : j \in CT, 1 \leq t \leq T \tag{2.76}$$

$$q_{j,t} \geq \underline{\rho}_j \tau_t \qquad \forall j,t : j \in CT, 1 \leq t \leq T \tag{2.77}$$

where $\underline{\rho}_j[ru/h]$ and $\overline{\rho}_j[ru/h]$ are the lower and the upper bounds for the rate of material that is processed by the continuous task j, respectively.

The resource constraints

The general material balance constraints (constraints (2.67)-(2.68) are a special case of (2.78) used to model the sequence restrictions) can be written as follows:

$$w_{r,t} = wf_{r,\Omega(t-1)} + \sum_{i \in BT} \bar{\mu}_{i,r} \sum_{t'=t-d}^{t-1} z_{i,\Omega(t'),\Omega(t-1)} - \sum_{i \in BT} \mu_{i,r} \sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')} \quad (2.78)$$

$\forall r, t : 1 \leq r \leq R, 1 \leq t \leq T$ and where

$\bar{\mu}_{i,r}$: is the rate $[ru/h]$ of renewable resource $r \in R_u$ released at the end of task i, or
the quantity $[ru]$ of non renewable resource $r \in R_m$ produced at the end of task i, or
the number of units $[ru]$ of non renewable resource $r \in R_e$ released at the end of task i.

$\mu_{i,r}$: is the rate $[ru/h]$ of renewable resource $r \in R_u$ consumed at the beginning of task i, or
the quantity $[ru]$ of non renewable resource $r \in R_m$ consumed at the beginning of task i, or
the number of units $[ru]$ of non renewable resource $r \in R_e$ allocated to begin task i.

Continuous tasks are in process during time slots and therefore we have to control the levels of all the resources $r \in R_m$ related to the continuous tasks at the end of every time slot τ_t just before receiving possible resource releases at the beginning of the next time slot. The resource level at the end of every time slot can be expressed as follows :

$$wf_{r,t} = w_{r,t} + \sum_{j \in CT} \lambda_{j,r} q_{j,t} \qquad \forall r, t : r \in R_m, 1 \leq t \leq T \quad (2.79)$$

where

$$\begin{aligned} \lambda_{j,r} &= 1 \text{ if the continuous task } j \text{ produces resource } r \in R_m \\ &= -1 \text{ if the continuous task } j \text{ consumes resource } r \in R_m \\ &= 0 \text{ otherwise.} \end{aligned}$$

The levels of resources $r \in R_e \cup R_u$ are not changed during time slots, therefore

$$wf_{r,t} = w_{r,t} \forall t,\ \forall r \in R_e \cup R_u \quad (2.80)$$

We have to verify at the beginning and at the end of each time slot that the resource capacity limitations are satisfied.

$$Rmin_r \leq w_{r,t} \leq Rmax_r \qquad \forall r,t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.81}$$

$$Rmin_r \leq wf_{r,t} \leq Rmax_r \qquad \forall r,t : r \in R_m, 1 \leq t \leq T \tag{2.82}$$

where $Rmin_r, Rmax_r$ represent the lower and upper limits on the level of resource r, respectively, in $[ru/h]$ for $r \in R_u$ and in $[ru]$ for $r \in R_m \cup R_e$.

However if there exists a time event t at which there exists both a release of resource r at the end of time slot $t-1$ and a consumption of r at the start of time slot t, then we need an additional set of constraints in order to guarantee that the maximum resource capacity usage is satisfied. This is why we have to also impose the following condition :

$$wf_{r,t} + \sum_{i \in BT} \bar{\mu}_{i,r} \sum_{t'=t-d+1}^{t} z_{i,\Omega(t'),t} \leq Rmax_r \tag{2.83}$$

for all $r,t : 1 \leq r \leq R, 1 \leq t \leq T$.

Finally, if for a certain resource $\tilde{r} \in R$, $\bar{\mu}_{i,\tilde{r}} = \mu_{i,\tilde{r}}$, for all $i \in BT$, then we do not use the constraints (2.78)-(2.83) for that resource $\tilde{r}$. We only impose the following set of constraints :

$$Rmin_{\tilde{r}} \leq \sum_{i \in BT} \sum_{t_1=1}^{T} \sum_{\substack{t_2=t_1 | t_1 \leq t \leq t_2 \\ \text{or } t_1 \leq t+T \leq t_2}}^{t_1+d-1} \bar{\mu}_{i,\tilde{r}} z_{i,\Omega(t_1),\Omega(t_2)} \leq Rmax_{\tilde{r}} \tag{2.84}$$

for all $t \in \{1, \ldots, T\}$. This occurs typically for the modeling of the availability of the utilities (i.e $\tilde{r} \in R_u$).

The objective function

We consider the maximization of the following objective function :

$$\sum_{j \in OUT} \sum_{t=1}^{T} q_{j,t} - \mu \sum_{t=1}^{T2} \tau_t \tag{2.85}$$

where μ is a constant processing cost per unit of time and $OUT \subseteq CT$ is the set of output products of the plant. It is here assumed that the output products are produced by some of the continuous tasks. This restriction can be relaxed easily.

Let X^G denote the set of feasible solutions to (2.59)-(2.62), (2.63), (2.65), (2.71)-(2.73), (2.76)-(2.84).

2.5.2 Strengthened and valid inequalities

By using the strengthening techniques presented in Section 1.3.2, we have obtained that if the continuous tasks $j \in CT$ consume or produce a product stored into a resource $r \in R_m$, i.e. if $\lambda_{j,r} \neq 0$ in (2.79), then the constraint (2.76) can be strengthened to

$$q_{j,t} \leq \overline{\rho}_j \tau_t - \sum_{i \in BT} \max(p_i \overline{\rho}_j - (Rmax_{r_j} - Rmin_{r_j}); 0) z_{i,t,t} \qquad \forall t \in \{1, \ldots, T\} \tag{2.86}$$

where r_j is the resource that stores the product produced or consumed by the continuous task j, and where $Rmin_{r_j}$ and $Rmax_{r_j}$ represent the lower and upper limits on the level of such a resource r_j, respectively.

Proposition 2.5.1. *Constraint (2.86) is valid for X^G.*

Proof. The continuous task j has a maximum speed ($\overline{\rho}_j[ru/h]$) and the resource r_j has a limited capacity ($[Rmin_{r_j}; Rmax_{r_j}][ru]$). If a batch task i starts at time slot t and finishes at time slot t, the duration τ_t of the time slot t is equal to the processing time of the batch task p_i. In that case, if $p_i \overline{\rho}_j > Rmax_{r_j} - Rmin_{r_j}$ then the continuous task cannot be performed at maximum speed and $q_{j,t} \leq Rmax_{r_j} - Rmin_{r_j}$. Constraint (2.86) imposes this additional restriction. □

We illustrate that this constraint is useful on a small example.

Example 2.5.2. *We consider the cyclic scheduling problem composed of one batch task with a limited number of units for performing the batch task, one continuous task and a storage tank. This test case is represented in Figure 1.1.*

The MIP formulation used is composed of (2.4)-(2.11), (2.42), (2.53) and (2.76)-(2.82). The objective function is given by (2.85). The characteristics of the instance are $T = 4$, $d = 4$, $nbr\_unit = 3$, $\mu = 3$, $p = 3$, $\underline{\rho} = 1$, $\overline{\rho} = 6$ and the minimum and maximum storage tank capacity are 0 ($R_{\min}$) and 15 ($R_{\max}$), respectively. The optimal solution $x^\star = (\tau^\star, z^\star, q^\star)$ of the linear relaxation of this problem at time slot $t = 2$ is : $\tau_2^* = 2.5$, $z_{2,2}^* = 0.361$ and $q_2^* = 15$ (where we have dropped the indices for batch and continuous tasks). The valid inequality (2.86) for $t = 2$ is :

$$q_2 \leq 6\tau_2 - 3z_{2,2}$$

and this inequality clearly cuts off the optimal solution of the linear relaxation since the left-hand-side equal 15 and the right-hand-side 13.917.

In order to limit further the amount processed by the continuous task at each time slot, the previous valid inequalities can be extended as follows :

$$\begin{aligned} \sum_{t'=t}^{l} q_{j,\Omega(t')} \leq & \sum_{i \in BT} \sum_{l'=t}^{t+d-1} \min\Big(p_i \overline{\rho}_j + Rmax_{r_j} - Rmin_{r_j} + BS \\ & \max(l - l' - 1, 0); \overline{\rho}_j (p_i + p^{\max} \max(l - l', 0)); Rmax_{r_j} - Rmin_{r_j} + BS(l - t)\Big) \\ & z_{i,t,\Omega(l')} + \min\Big(p^{\max}(l - t + 1)\overline{\rho}_j; Rmax_{r_j} - Rmin_{r_j} + BS(l - t)\Big) \\ & \Big(1 - \sum_{i \in BT} \sum_{l'=t}^{t+d-1} z_{i,t,\Omega(l')}\Big) \end{aligned} \tag{2.87}$$

for all $t \in \{1, \dots, T\}, l \in \{t, \dots, t+d-1\}$ where BS is the maximum batch size that can be discharged (output of a batch task) at each time slot in the storage tank corresponding to resource r_j.

A similar extension of this can be written as :

$$\sum_{t'=t}^{l} q_{j,\Omega(t')} \leq \sum_{i \in BT} \sum_{t'=l-d+1}^{l} \min \Big(p_i \overline{\rho}_j + Rmax_{r_j} - Rmin_{r_j} + BS \\ \max(t'-t-1,0); \overline{\rho}_j (p_i + p^{\max} \max(t'-t,0)); Rmax_{r_j} - Rmin_{r_j} + BS(l-t) \Big) \\ z_{i,\Omega(t'),\Omega(l)} + \min \Big(p^{\max}(l-t+1)\overline{\rho}_j; Rmax_{r_j} - Rmin_{r_j} + BS(l-t) \Big) \\ \Big(1 - \sum_{i \in BT} \sum_{l'=l-d+1}^{l} z_{i,\Omega(t'),\Omega(l)} \Big) \tag{2.88}$$

for all $t \in \{1, \dots, T\}, l \in \{t, \dots, t+d-1\}$.

Proposition 2.5.3. *Constraint (2.87) is valid for X^G.*

Proof. (sketch)

a. If $z_{i,t,\Omega(l')} = 1$ for some $l' \leq l-1$, then $\sum_{t'=t}^{l'} \tau_{\Omega(t')} = p_i$. Three possible cases have to be satisfied.

   The $\sum_{t'=t}^{l} q_{j,\Omega(t')} \leq p_i \overline{\rho}_j + Rmax_{r_j} - Rmin_{r_j} + BS(l-l'-1)$ since the storage tank is at the value $Rmax_{r_j}$ at the beginning of time $l'+1$, we can have at most $(l-l'-1)$ discharge before the end of time slot l and therefore the continuous task can process at most such amount of material.

   The $\sum_{t'=t}^{l} q_{j,\Omega(t')} \leq \overline{\rho}_j (p_i + p^{\max}(l-l'))$ since $\sum_{t'=l'+1}^{l} \tau_{t'} \leq p^{\max}(l-l')$ and the maximum rate for performing the continuous task j is $\overline{\rho}_j$.

   Finally, the $\sum_{t'=t}^{l} q_{j,\Omega(t')} \leq Rmax_{r_j} - Rmin_{r_j} + BS(l-t)$ since at the beginning of time slot t, the storage tank is at most at level $Rmax_{r_j}$ and between time slot t and the end of time slot l, at most $l-t$ discharge can happen and this corresponds to the maximum amount of material that the continuous task can process.

b. If $z_{i,t,\Omega(l')} = 1$ for some $l' \geq l$, then $\sum_{t'=t}^{l} \tau_{\Omega(t')} \leq p_i$. Clearly $\sum_{t'=t}^{l} q_{j,\Omega(t')} \leq \overline{\rho}_j p_i$ since $\sum_{t'=t}^{l} \tau_{t'} \leq p_i$.

   We have also that $\sum_{t'=t}^{l} q_{j,\Omega(t')} \leq Rmax_{r_j} - Rmin_{r_j} + BS(l-t)$ since at the beginning of time slot t, the storage tank is at most at level $Rmax_{r_j}$ and between time slot t and time slot l, at most $l-t$ discharge can happen and this corresponds to the maximum amount of material that the continuous task can process.

c. If $z_{i,t,\Omega(l')} = 0$ for all $l' \geq t$, then clearly $\sum_{t'=t}^{l} q_{j,\Omega(t')} \leq p^{\max}(l-t+1)\overline{\rho}_j$ since the maximum time slot duration is $p^{\max}$.

   And we have that $\sum_{t'=t}^{l} q_{j,\Omega(t')} \leq Rmax_{r_j} - Rmin_{r_j} + BS(l-t)$ since at the beginning of time slot t, the storage tank is at most at level $Rmax_{r_j}$ and between time slot t and time slot l, at most $l - t$ discharge can happen and this corresponds to the maximum amount of material that the continuous task can process.

$\square$

Proposition 2.5.4. *Constraint (2.88) is valid for X^G.*

Proof. By using the same technique as for the proof of validity of (2.87), we can prove that constraint (2.88) is valid. $\square$

The two previous valid inequalities can clearly be strengthened for specific test cases.

We illustrate on a small example that these constraints are useful.

Example 2.5.5. *Based on the same MIP formulation and same instance as for Example 2.5.2, where we only have added constraint (2.86), we obtain the following optimal solution of the linear relaxation : for the z variables starting at time slot 4 : $z^*_{4,1} = 0.44$, $z^*_{4,3} = 0.56$ and $z^*_{4,4} = z^*_{4,2} = 0$ and for the quantity processed by the continuous task at each time slot, we obtain : $q^*_1 = 8, q^*_2 = 6, q^*_3 = 10$ and $q^*_4 = 8$. The constraint (2.87) for $t = 4$ and $l = 7$ is the following :*

$$\sum_{t'=1}^{4} q_{t'} \leq 39z_{4,4} + 39z_{4,1} + 33z_{4,2} + 18z_{4,3} + 39(1 - \sum_{t'=1}^{4} z_{4,t'})$$

and this inequality clearly cuts off the optimal solution of the linear relaxation since the left-hand-side equal 32 and the right-hand-side 27.24.

Unfortunately, for more general cases, these two valid inequalities (2.87)-(2.88) have been tested and do not improve much the tightness of the model formulation.

2.6 Conclusion

In this Chapter, we have studied a continuous time formulation in order to model the cyclic scheduling of a mixed plant composed of batch and continuous processes. By improving the initial continuous time formulation of various special cases of the general problem, we have obtained a tighter model formulation for these special cases. In the next Chapter, we test these three improved model formulations in order to illustrate the efficiency of such model formulations.

Chapter 3

Computational Experiments

In this Chapter, we report on tests on the efficiency of exact and heuristic methods in order to solve special cases of the general cyclic scheduling problem that are described in Chapter 2. By solving various instances of these special cases, we show that the corresponding improved formulations given in Chapter 2 can solve these problem instances faster than the original continuous time formulations derived from Schilling and Pantelides in [44]. We solve the improved formulations of such problems by using a standard Branch and Bound system. In order to improve the Branch and Bound performance for the multiple batch tasks case studied, we set priorities to branch first on the $z_{i,t,\Omega(l)}$ binary variables. The reason is that the other integer variables (units of resources available corresponding to machines and equipment) depend only on the binary variables $z_{i,t,\Omega(l)}$, and automatically take integer values when the $z_{i,t,\Omega(l)}$ variables take integral values. For the special cases, we impose to branch up first because this influences more the objective function value of the corresponding subproblem.

All the results in this Chapter have been obtained by using the Xpress MP software on a pentium 4, running at 3 GHz.

We look at basic, multiple batch task and industrial cases and we illustrate the resolution of the corresponding problem instances when using the improved formulation.

In order to summarize the various cases tested in this Chapter, we present in Figure 3.1 a map of the test cases.

The basic case is addressed in Section 3.1, the multiple batch task case in Section 3.2 and the industrial case in Section 3.3.

For example, Figure 3.1 indicates for the multiple batch task case in Section 3.2 that in Subsection 3.2.2, the test case contains multiple batch tasks, a restricted number of reactors and some resource capacity restrictions. Moreover, this figure indicates that in Subsection 3.2.3, the test case contains the one described in subsection 3.2.2 and additionally a continuous task.

In this Chapter, we will compare the efficiency of various strengthened formu-

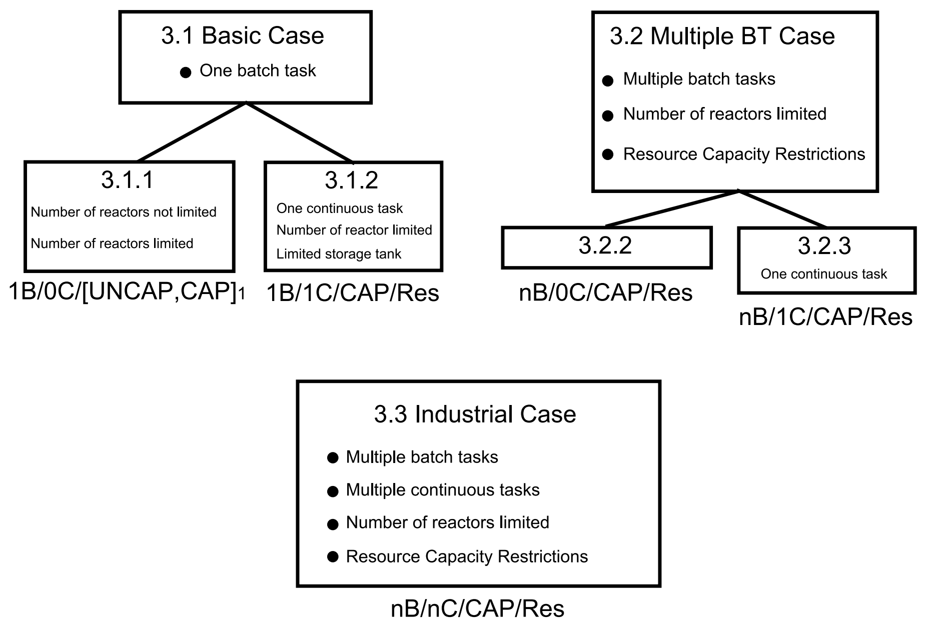

Figure 3.1: Map of the Test Cases

lations. Therefore, we want to further subdivide the field F defined in Section 2.2 for the classification of the formulation of scheduling problems. The new definition for F is the following :

F=[INIT,STR1,STR2,STRM][1].

INIT : we use the initial formulation in order to model the problem

STR1 : we use the basic strengthened formulation in order to model the problem

STR2 : we use the complete strengthened formulation in order to model the problem

STRM : we use the complete strengthened formulation in order to model the problem but some of the constraints are defined as model cuts. These constraints are removed from the complete strengthened formulation, are added to the cutpool, and generated as cuts when they are violated.

3.1 Basic case

The basic test case is composed of one continuous task and of a finite number of units (reactors) denoted by *nbr_unit* on which one batch task is performed. The reactors are identical and produce the batches in parallel. The batches produced are stored in a limited capacity tank before being processed by the continuous task. The test case is represented in Figure 1.1.

The size of a batch is fixed and is equal to 8 ru and the processing time of the batch task is $3h$. The lower and upper limits on the level of product in the storage tank are 0 ru and 15 ru, respectively. The lower and upper bounds for the rate of material that is processed by the continuous task are $\underline{\rho} = 1[ru/h]$ and $\overline{\rho} = 6[ru/h]$, respectively.

In this section, we consider first a special case of this problem composed of only one batch task. Then, we consider the basic case defined above.

3.1.1 One batch task

This case corresponds to a special case of the basic case presented above. We do not consider the continuous task and the storage tank after the batch task. For this special case that corresponds to the model presented in Section 2.3, we use the data of the basic case given above.

We consider first the performance of the initial (F1) and of three strengthened formulations (F2)-(F3)-(F4).

For the uncapacitated case, the initial formulation (F1) is based on the constraints (2.2)-(2.5) and (2.7). The basic strengthened formulation (F2) is based on the constraints (2.4)-(2.5), (2.7), (2.8) and (2.10). The complete strengthened formulation (F3) is composed of (F2) and in addition the constraints (2.9), (2.11) and (2.53). The last strengthened formulation (F4) considers the additional constraints (2.9), (2.11) and (2.53) as model cuts, i.e these constraints are removed from the initial formulation, are added to the cut pool, and generated as cuts when they are violated. The number of units (*nbr_unit*) in the constraint (2.53) is unlimited (infinite).

For the capacitated case, the formulations (F1)-(F4) are composed of the constraints of the uncapacitated case and the constraint (2.6). The valid inequality (2.42) is added to the formulations (F2)-(F4).

The objective function for both cases is the following ($\mu = 1$) :

$$\max \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)} - \sum_{t=1}^{T} \tau_t$$

and corresponds to a measure of the productivity, corresponding to total cycle production minus the cycle duration.

For clarity we recall these formulations below.

F1 uncapacitated (without constraint (2.6)) / capacitated (with constraint (2.6)) :

1B/0C/[CAP,UNCAP]^1//INIT

$$\max \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)} - \sum_{t=1}^{T} \tau_t$$

$$p z_{t,\Omega(l)} \leq \sum_{k=t}^{l} \tau_{\Omega(k)} \qquad \forall t,l : 1 \leq t \leq T, t \leq l \leq t+d-1 \tag{2.2}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} \leq p z_{t,\Omega(l)} + p(l-t+1)(1 - z_{t,\Omega(l)})$$
$$\forall t,l : 1 \leq t \leq T, t \leq l \leq t+d-1 \tag{2.3}$$

$$\sum_{l=t}^{t+d-1} z_{t,\Omega(l)} \leq 1 \qquad \forall t : 1 \leq t \leq T \tag{2.4}$$

$$\sum_{t=l-d+1}^{l} z_{\Omega(t),l} \leq 1 \qquad \forall l : 1 \leq l \leq T \tag{2.5}$$

$$\sum_{t=1}^{T} \sum_{\substack{l=t|t\leq t'\leq l \\ \text{or } t\leq t'+T\leq l}}^{t+d-1} z_{t,\Omega(l)} \leq nbr\_unit \qquad \forall t' : 1 \leq t' \leq T \tag{2.6}$$

$$z_{t,\Omega(l)} \in \{0,1\} \qquad \forall t,l : 1 \leq t \leq T, t \leq l \leq t+d-1,$$
$$\tau_t \geq 0 \qquad \forall t : 1 \leq t \leq T. \tag{2.7}$$

F2 uncapacitated (without constraints (2.6) and (2.42)) / capacitated (with constraints (2.6) and (2.42)) :

1B/0C/[CAP,UNCAP]^1//STR1

$$\max \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)} - \sum_{t=1}^{T} \tau_t$$

$$p \sum_{k=t|t\neq l}^{l} z_{t,\Omega(k)} + p z_{\Omega(l),\Omega(l)} \leq \sum_{k=t}^{l} \tau_{\Omega(k)}$$
$$\forall t,l : 1 \leq t \leq T,\ t \leq l \leq t+d-1 \tag{2.8}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} + p \sum_{k=l-d+1}^{l-1} \min\{l-t, l-k\} z_{\Omega(k),\Omega(l)} \leq p(l-t+1)$$
$$\forall t,l : 1 \leq t \leq T,\ t \leq l \leq t+d-1 \tag{2.10}$$

1B/0C/[CAP,UNCAP]^1//STR1 (Contd)

$$\sum_{l=t}^{t+d-1} z_{t,\Omega(l)} \leq 1 \qquad \forall t : 1 \leq t \leq T \tag{2.4}$$

$$\sum_{t=l-d+1}^{l} z_{\Omega(t),l} \leq 1 \qquad \forall l : 1 \leq l \leq T \tag{2.5}$$

$$\sum_{t=1}^{T} \sum_{\substack{l=t|t\leq t'\leq l \\ \text{or } t\leq t'+T\leq l}}^{t+d-1} z_{t,\Omega(l)} \leq nbr\_unit \qquad \forall t' : 1 \leq t' \leq T \tag{2.6}$$

$$\frac{p}{nbr\_unit} \sum_{t=1}^{T} \sum_{t'=t}^{t+d-1} z_{t,\Omega(t')} - \sum_{t=1}^{T} \tau_t \leq 0 \tag{2.42}$$

$$z_{t,\Omega(l)} \in \{0,1\} \qquad \forall t,l : 1 \leq t \leq T, t \leq l \leq t+d-1,$$

$$\tau_t \geq 0 \qquad \forall t : 1 \leq t \leq T \tag{2.7}$$

F3/F4 uncapacitated (without constraints (2.6) and (2.42), with nbr_unit in constraint (2.53) infinite) / capacitated (with constraints (2.6) and (2.42)) :

1B/0C/[CAP,UNCAP]^1//[STR2,STRM]^1

$$\max \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{t,\Omega(l)} - \sum_{t=1}^{T} \tau_t$$

$$p \sum_{k=t|t\neq l}^{l} z_{t,\Omega(k)} + p z_{\Omega(l),\Omega(l)} \leq \sum_{k=t}^{l} \tau_{\Omega(k)}$$

$$\forall t,l : 1 \leq t \leq T,\ t \leq l \leq t+d-1 \tag{2.8}$$

$$p \sum_{k=t|t\neq l}^{l} z_{\Omega(k),\Omega(l)} + p z_{t,t} \leq \sum_{k=t}^{l} \tau_{\Omega(k)}$$

$$\forall t,l : 1 \leq t \leq T,\ t \leq l \leq t+d-1 \tag{2.9}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} + p \sum_{k=l-d+1}^{l-1} \min\{l-t, l-k\} z_{\Omega(k),\Omega(l)} \leq p(l-t+1)$$

$$\forall t,l : 1 \leq t \leq T,\ t \leq l \leq t+d-1 \tag{2.10}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} + p \sum_{k=t+1}^{t+d-1} \min\{l-t, k-t\} z_{t,\Omega(k)} \leq p(l-t+1)$$

$$\forall t,l : 1 \leq t \leq T,\ t \leq l \leq t+d-1 \tag{2.11}$$

1B/0C/[CAP,UNCAP]^1//[STR2,STRM]^1 (Contd)

$$\sum_{k=t}^{l} \min(coeff_k, nbr\_unit, d)\tau_{\Omega(k)} \geq p \left(\sum_{t_1=t}^{l} \sum_{t_2=t_1}^{\min(l,t_1+d-1)} z_{\Omega(t_1),\Omega(t_2)} \right)$$

$$\forall t, l : 1 \leq t \leq T, t+2 \leq l \leq t+T+d-3 \qquad (2.53)$$

$$\sum_{l=t}^{t+d-1} z_{t,\Omega(l)} \leq 1 \qquad \forall t : 1 \leq t \leq T \qquad (2.4)$$

$$\sum_{t=l-d+1}^{l} z_{\Omega(t),l} \leq 1 \qquad \forall l : 1 \leq l \leq T \qquad (2.5)$$

$$\sum_{t=1}^{T} \sum_{\substack{l=t | t \leq t' \leq l \\ \text{or } t \leq t'+T \leq l}}^{t+d-1} z_{t,\Omega(l)} \leq nbr\_unit \qquad \forall t' : 1 \leq t' \leq T \qquad (2.6)$$

$$\frac{p}{nbr\_unit} \sum_{t=1}^{T} \sum_{t'=t}^{t+d-1} z_{t,\Omega(t')} - \sum_{t=1}^{T} \tau_t \leq 0 \qquad (2.42)$$

$$z_{t,\Omega(l)} \in \{0,1\} \qquad \forall t, l : 1 \leq t \leq T, t \leq l \leq t+d-1,$$

$$\tau_t \geq 0 \qquad \forall t : 1 \leq t \leq T. \qquad (2.7)$$

Uncapacitated Case

The characteristics of the small (resp. large) instance, i.e. the instance with the smallest (resp. the highest) amount of binary variables, in Table 3.1 are the following : $T = 30$ (resp. 20), $d = 5$ (resp. 10). There are 150 (resp. 200) binary variables and 30 (resp. 20) continuous variables.

In Tables 3.1-3.3, "Nodes" represents the total number of Branch-and-Bound nodes needed in order to solve the special case to optimality, "Time" represents the corresponding total CPU time, "Optimal number of batches" represents the optimal number of times that the batch task is performed during the cycle and "Optimal cycle duration" represents the optimal length of the scheduling cycle.

In Table 3.1, for the small and the large instances, the formulation (F1) cannot solve the problem to optimality within a CPU time of 1000 *sec*. For such cases, a star is added to the CPU solution time of the corresponding instance in Table 3.1 and we calculate the remaining duality gap after 1000 *sec*. for that case as $\frac{\text{Best bound - Best solution}}{\text{Best bound}}$.

For both instances in Table 3.1, the strengthened formulations (F2)-(F4) obtain more quickly the optimal solution in fewer nodes than the initial formulation (F1). The strengthened formulation (F4) solves both instances quicker than the other three formulations. Not surprisingly, the strengthened formulations (F3)-(F4) taking into account all valid inequalities need fewer nodes than the other two formulations even though formulation (F2) has no duality gap remaining after the root branch-and-bound node.

Small Instance ($T = 30, d = 5$)				
	F1	F2	F3	F4
Constraints	332	332	1502	1502
Nodes	149800	190	1	**1**
Time	1000 s*	2.7 s	2 s	**1.5 s**
Remaining Duality gap	28.18 %	0 %	0 %	**0 %**
Optimal number of batches	/	30	30	30
Optimal cycle duration (h)	/	18	18	18
Large Instance ($T = 20, d = 10$)				
	F1	F2	F3	F4
Constraints	422	422	1302	1302
Nodes	198600	1817	1	**1**
Time	1000 s*	17.1 s	2 s	**1.5 s**
Remaining Duality gap	18.45 %	0 %	0 %	**0 %**
Optimal number of batches	/	20	20	20
Optimal cycle duration (h)	/	6	6	6

Table 3.1: Comparison between the initial (F1) and the strengthened formulations (F2)-(F4); One batch task, Uncapacitated.

For each instance, the optimal number of batches and the optimal cycle duration are the same for the formulations (F2)-(F4). Note that the objective function is the maximization of a measure of the productivity.

Capacitated Case

We suppose now that the number of units is limited to 4 ($nbr\_unit = 4$).

The characteristics of the small (resp. large) instance in Table 3.2 are the following : $T = 30$ (resp. 20), $d = 5$ (resp. 10). There are 150 (resp. 200) binary variables and 30 (resp. 20) continuous variables.

The formulation (F1*) is nothing else than the initial formulation (F1) where we have added the valid inequality (2.42).

For both instances in Table 3.2, the strengthened formulations (F2)-(F4) obtain the optimal solution quicker in fewer nodes than the initial formulations (F1)-(F1*). The strengthened formulation (F4) solves the small instance quicker than the other formulations. The strengthened formulations (F3)-(F4) taking into account all valid inequalities need fewer nodes than the other three formulations. We can observe that, when we add (2.42) in the initial formulation (F1), the resolution of the problem is much faster than with the initial formulation (F1).

For each instance, the optimal number of batches and the optimal cycle duration are the same for the formulations (F1*)-(F4). Again, the objective function is the maximization of a measure of the productivity.

Note that the difference between the optimal solutions obtained for the un-

Small Instance ($T = 30, d = 5$)					
nbr_unit=4	F1	F1*	F2	F3	F4
Constraints	362	363	363	1533	1533
Nodes	196200	6627	1123	29	**1**
Time	1000 s*	36.4 s	8.1 s	22.8 s	**7 s**
Remaining Duality gap	53.34 %	0 %	0 %	0 %	**0 %**
Optimal number of batches	/	30	30	30	30
Optimal cycle duration (h)	/	22.5	22.5	22.5	22.5
Large Instance ($T = 20, d = 10$)					
nbr_unit=4	F1	F1*	F2	F3	F4
Constraints	442	443	443	1323	1323
Nodes	232900	663	**1**	**1**	**1**
Time	1000 s*	2.6 s	**0 s**	**0 s**	**0 s**
Remaining Duality gap	55.42 %	0 %	**0 %**	**0 %**	**0 %**
Optimal number of batches	/	20	20	20	20
Optimal cycle duration (h)	/	15	15	15	15

Table 3.2: Comparison between the initial (F1)-(F1*) and the strengthened formulations (F2)-(F4); One batch task, Capacitated.

capacited case and the ones obtained for the capacitated case, is the duration of the cycle. We can observe for the two instances tested that for the same optimal number of batch task performed, the length of the scheduling cycle is larger for the capacitated case than for the uncapacitated one.

3.1.2 One batch and one continuous task

We consider the basic case where there is only one batch and one continuous tasks, and one storage tank between the batch and the continuous tasks with limited capacity. The objective is to maximize a measure of the productivity of this process.

We consider first the performance of the initial (F1*) and of three strengthened formulations (F5)-(F6)-(F7). The initial formulation (F1*) is based on the constraints (2.2)-(2.7), (2.42), the constraints for the continuous task (2.76)-(2.77) and the resource restrictions given by (2.78)-(2.79) and (2.81)-(2.82). The completed strengthened formulation (F5) is based on the same set of constraints as (F1*) except that (2.2) is replaced by (2.8)-(2.9), (2.3) is replaced by (2.10)-(2.11) and (2.76) is replaced by (2.86). Moreover, we have added the valid inequalities (2.53). The strengthened formulation (F6) is composed of (F5) and of the constraints (2.87)-(2.88). The last strengthened formulation (F7) considers the additional constraints (2.87)-(2.88) as model cuts, i.e these constraints are removed from the initial formulation, are added to the cut pool, and generated as cuts when they are violated.

The objective function is the following :

$$\max \sum_{t=1}^{T} q_t - \mu \sum_{t=1}^{T} \tau_t.$$

where q_t is the quantity process by the continuous task at time slot t.

For clarity we recall these formulations below.
F1* :

1B/1C/CAP/Res//INIT

$$\max \sum_{t=1}^{T} q_t - \mu \sum_{t=1}^{T} \tau_t$$

$$p z_{t,\Omega(l)} \leq \sum_{k=t}^{l} \tau_{\Omega(k)} \qquad \forall t, l : 1 \leq t \leq T, t \leq l \leq t+d-1 \tag{2.2}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} \leq p z_{t,\Omega(l)} + p(l-t+1)(1 - z_{t,\Omega(l)})$$
$$\forall t, l : 1 \leq t \leq T, t \leq l \leq t+d-1 \tag{2.3}$$

$$\sum_{l=t}^{t+d-1} z_{t,\Omega(l)} \leq 1 \qquad \forall t : 1 \leq t \leq T \tag{2.4}$$

$$\sum_{t=l-d+1}^{l} z_{\Omega(t),l} \leq 1 \qquad \forall l : 1 \leq l \leq T \tag{2.5}$$

$$\sum_{t=1}^{T} \sum_{\substack{l=t \mid t \leq t' \leq l \\ \text{or } t \leq t'+T \leq l}}^{t+d-1} z_{t,\Omega(l)} \leq nbr\_unit \qquad \forall t' : 1 \leq t' \leq T \tag{2.6}$$

$$\frac{p}{nbr\_unit} \sum_{t=1}^{T} \sum_{t'=t}^{t+d-1} z_{t,\Omega(t')} - \sum_{t=1}^{T} \tau_t \leq 0 \tag{2.42}$$

$$q_t \leq \overline{\rho} \tau_t \qquad \forall t : 1 \leq t \leq T \tag{2.76}$$

$$q_t \geq \underline{\rho} \tau_t \qquad \forall t : 1 \leq t \leq T \tag{2.77}$$

$$w_t = wf_{\Omega(t-1)} + \bar{\mu} \sum_{t'=t-d}^{t-1} z_{\Omega(t'),\Omega(t-1)} - \mu \sum_{t'=t}^{t+d-1} z_{t,\Omega(t')}$$
$$\forall t : 1 \leq t \leq T \tag{2.78}$$

$$wf_t = w_t + \lambda q_t \qquad \forall t : 1 \leq t \leq T \tag{2.79}$$

$$Rmin \leq w_t \leq Rmax \qquad \forall t : 1 \leq t \leq T \tag{2.81}$$

$$Rmin \leq wf_t \leq Rmax \qquad \forall t : 1 \leq t \leq T \tag{2.82}$$

$$z_{t,\Omega(l)} \in \{0,1\} \qquad \forall t, l : 1 \leq t \leq T, t \leq l \leq t+d-1,$$
$$\tau_t \geq 0 \qquad \forall t : 1 \leq t \leq T. \tag{2.7}$$

F5 :

1B/1C/CAP/Res//STR1

$$\max \sum_{t=1}^{T} q_t - \mu \sum_{t=1}^{T} \tau_t$$

$$p \sum_{k=t|t\neq l}^{l} z_{t,\Omega(k)} + p z_{\Omega(l),\Omega(l)} \leq \sum_{k=t}^{l} \tau_{\Omega(k)}$$
$$\forall t, l : 1 \leq t \leq T,\ t \leq l \leq t+d-1 \tag{2.8}$$

$$p \sum_{k=t|t\neq l}^{l} z_{\Omega(k),\Omega(l)} + p z_{t,t} \leq \sum_{k=t}^{l} \tau_{\Omega(k)}$$
$$\forall t, l : 1 \leq t \leq T,\ t \leq l \leq t+d-1 \tag{2.9}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} + p \sum_{k=l-d+1}^{l-1} \min\{l-t, l-k\} z_{\Omega(k),\Omega(l)} \leq p(l-t+1)$$
$$\forall t, l : 1 \leq t \leq T,\ t \leq l \leq t+d-1 \tag{2.10}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} + p \sum_{k=t+1}^{t+d-1} \min\{l-t, k-t\} z_{t,\Omega(k)} \leq p(l-t+1)$$
$$\forall t, l : 1 \leq t \leq T,\ t \leq l \leq t+d-1 \tag{2.11}$$

$$\sum_{k=t}^{l} \min(coeff_k, nbr\_unit, d) \tau_{\Omega(k)} \geq p \left(\sum_{t_1=t}^{l} \sum_{t_2=t_1}^{\min(l, t_1+d-1)} z_{\Omega(t_1),\Omega(t_2)} \right)$$
$$\forall t, l : 1 \leq t \leq T, t+2 \leq l \leq t+T+d-3 \tag{2.53}$$

$$\sum_{l=t}^{t+d-1} z_{t,\Omega(l)} \leq 1 \qquad \forall t : 1 \leq t \leq T \tag{2.4}$$

$$\sum_{t=l-d+1}^{l} z_{\Omega(t),l} \leq 1 \qquad \forall l : 1 \leq l \leq T \tag{2.5}$$

$$\sum_{t=1}^{T} \sum_{\substack{l=t|t\leq t'\leq l \\ \text{or } t\leq t'+T\leq l}}^{t+d-1} z_{t,\Omega(l)} \leq nbr\_unit \qquad \forall t' : 1 \leq t' \leq T \tag{2.6}$$

$$\frac{p}{nbr\_unit} \sum_{t=1}^{T} \sum_{t'=t}^{t+d-1} z_{t,\Omega(t')} - \sum_{t=1}^{T} \tau_t \leq 0 \tag{2.42}$$

$$q_t \leq \overline{\rho} \tau_t - \max(p\overline{\rho} - (Rmax - Rmin); 0) z_{t,t}$$
$$\forall t : 1 \leq t \leq T \tag{2.86}$$

$$q_t \geq \underline{\rho} \tau_t \qquad \forall t : 1 \leq t \leq T \tag{2.77}$$

1B/1C/CAP/Res//STR1 (Contd)

$$w_t = wf_{\Omega(t-1)} + \bar{\mu} \sum_{t'=t-d}^{t-1} z_{\Omega(t'),\Omega(t-1)} - \mu \sum_{t'=t}^{t+d-1} z_{t,\Omega(t')}$$

$$\forall t : 1 \leq t \leq T \qquad (2.78)$$

$$wf_t = w_t + \lambda q_t \qquad \forall t : 1 \leq t \leq T \qquad (2.79)$$

$$Rmin \leq w_t \leq Rmax \qquad \forall t : 1 \leq t \leq T \qquad (2.81)$$

$$Rmin \leq wf_t \leq Rmax \qquad \forall t : 1 \leq t \leq T \qquad (2.82)$$

$$z_{t,\Omega(l)} \in \{0,1\} \qquad \forall t,l : 1 \leq t \leq T, t \leq l \leq t+d-1,$$

$$\tau_t \geq 0 \qquad \forall t : 1 \leq t \leq T. \qquad (2.7)$$

F6/F7 : The formulations F6 and F7 are composed of F5 and of the following two constraints :

1B/1C/CAP/Res//[STR2,STRM][1]

1B/1C/CAP/Res//STR1

$$\sum_{t'=t}^{l} q_{\Omega(t')} \leq \sum_{l'=t}^{t+d-1} \min\Big(p\overline{\rho} + Rmax - Rmin + BS\max(l-l'-1,0);$$
$$\overline{\rho}(p + p\max(l-l',0)); Rmax - Rmin + BS(l-t)\Big) z_{t,\Omega(l')}$$
$$+ \min\Big(p(l-t+1)\overline{\rho}; Rmax - Rmin + BS(l-t)\Big)$$
$$\Big(1 - \sum_{l'=t}^{t+d-1} z_{t,\Omega(l')}\Big) \qquad \forall t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\} \ (2.87)$$

$$\sum_{t'=t}^{l} q_{\Omega(t')} \leq \sum_{t'=l-d+1}^{l} \min\Big(p\overline{\rho} + Rmax - Rmin + BS\max(t'-t-1,0);$$
$$\overline{\rho}(p + p\max(t'-t,0)); Rmax - Rmin + BS(l-t)\Big) z_{\Omega(t'),\Omega(l)}$$
$$+ \min\Big(p(l-t+1)\overline{\rho}; Rmax - Rmin + BS(l-t)\Big)$$
$$\Big(1 - \sum_{l'=l-d+1}^{l} z_{\Omega(t'),\Omega(l)}\Big) \qquad \forall t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\} \ (2.88)$$

The characteristics of the small (resp. large) instance in Table 3.3 are the following : $T = 10$ (resp. 15), $d = 8$, $nbr\_unit = 4$ and $\mu = 3$. There are 80 (resp. 120) binary variables and 41 (resp. 61) continuous variables.

In Table 3.3, the formulations (F6)-(F7) need fewer nodes to solve the two problem instances. For the small instance, the initial formulation (F1*) solves the problem quicker. However, for the large instance, the formulation (F6) provides the optimal solution quicker. By using all valid inequalities found, we reduce the CPU time and the number of nodes to solve the large problem instance.

Small Instance ($T = 10, d = 8$)				
	F1*	F5	F6	F7
Constraints	223	503	663	663
Nodes	**1061**	2317	267	293
Time	**3 s**	11 s	4 s	4 s
Large Instance ($T = 15, d = 8$)				
	F1*	F5	F6	F7
Constraints	333	828	1068	1068
Nodes	610379	93619	**2709**	3359
Time	1214 s	744 s	**57 s**	59 s

Table 3.3: Comparison between the initial (F1*) and the strengthened formulations (F5)-(F7); One batch and one continuous task

3.2 Multiple Batch Task Case

In this section, we address a multiple batch task problem where there are multiple batch tasks, one continuous task and some capacity restrictions. We describe first the multiple batch task case in Section 3.2.1. Then, in Section 3.2.2, we solve two instances of the special case of the multiple batch task problem composed of multiple batch tasks and of some resource restrictions (i.e. no continuous task). Finally, in Section 3.2.3, in addition to the multiple batch tasks, we consider a continuous task and a storage tank between the batch and the continuous processes. We solve various instances of this problem by using exact and heuristic methods.

3.2.1 Description

We describe first the multiple batch task case. Then we show that in order to optimize the productivity of such a problem, i.e. a nonlinear objective function, it is possible to solve a sequence of problems with a linear objective function that gives at the end the solution of the problem with the nonlinear objective function. Finally, we illustrate a typical optimal solution for the multiple batch task case.

Characteristics of the multiple batch task case

The multiple batch task test case is composed of one continuous task and of a finite number of units (reactors) denoted by *nbr_unit* on which a set of batch tasks are performed. The reactors are identical and produce the batches in parallel. The batches produced are stored in a limited capacity tank before being processed by the continuous task. The test case is represented in Figure 1.1. The size of a batch is fixed and is equal to 8 *ru*.

The polymerization process (I) that takes place in the reactors is subdivided in five consecutive batch tasks (there are thus precedence constraints between the 5 batch tasks) : the filling of the reactor (I,1), the heating of the raw material (I,2,h), the exothermic reaction (I,2,r), the cooling (I,2,c) and the discharge (I,3). In our model, the exothermic reaction (I,2,r) is further subdivided in 4 batch subtasks because the consumption of resources varies too much during the temperature regulation task. We assume that the filling of the reactor takes 0.166 h, the heating of the raw material takes 0.4522 h, the exothermic reaction is subdivided in 4 subtasks taking 0.5 h, 0.5 h, 1 h and 1.44125 h respectively, the cooling takes 0.919 h and the discharge 0.166 h.

The resources are the intermediate storage tank (IS), the hot water (H), the cold water (C) and a stock attached to each batch task (a1-a5) in order to model the number of reactors available to perform the corresponding batch task, i.e nbr_unit of resources a_1 to a_5 are available in total.
The rate of hot water needed to perform the heating task is 3 $[ru/h]$. The rates of cold water needed to perform the four subtasks of the exothermic reaction are 3.7 $[ru/h]$, 1.64 $[ru/h]$, 0.92 $[ru/h]$ and 0.41 $[ru/h]$ respectively. The rate of cold water needed for the cooling task is 2 $[ru/h]$.

The lower and upper limits on the level of product in the storage tank, on the rate of hot water and cold water are $[0, 15]$ ru,$[0, 3]$ $[ru/h]$ and $[0, 4.2]$ $[ru/h]$, respectively.

The lower and upper bounds for the rate of material that is processed by the continuous task are $\underline{\rho} = 1[ru/h]$ and $\overline{\rho} = 6[ru/h]$, respectively.

The resource task network of the test case problem is represented in Figure 3.2, where the continuous task is denoted by II.

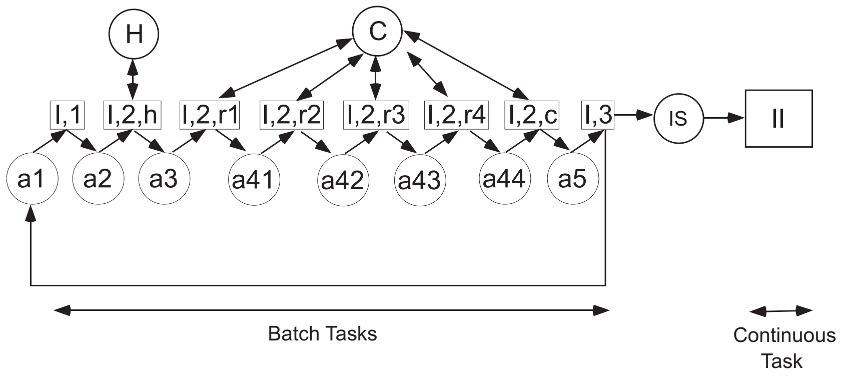

Figure 3.2: The resource task network representation for the test case

The two main characteristics relative to this test case are that the continuous task cannot be interrupted because we try to find a cyclic schedule and the second is that we cannot wait between the heating, the exothermic reaction and the cooling tasks.

The objective function

The objective is to maximize the average production per unit of time over the entire cycle, or the productivity. The non linear objective function is the following

$$\max \frac{\sum_{j \in OUT} \sum_{t=1}^{T} q_{j,t}}{\sum_{t=1}^{T} \tau_t}, \tag{3.1}$$

where $OUT \subseteq CT$ is the set of output products of the plant. It is here assumed that the output products are produced by some of the continuous tasks. This restriction can be relaxed easily.

It was shown by Isbell and Marlow [23], and also extended by Dinkelbach [19], that this nonlinear objective function can be optimized for continuous problems by solving a sequence (iterations $p = 1, 2, \ldots$) of linear optimization problems where the objective function at iteration p is

$$\max \sum_{j \in OUT} \sum_{t=1}^{T} q_{j,t} - \mu^p \sum_{t=1}^{T} \tau_t, \tag{3.2}$$

and where μ^p is a constant which is computed before iteration p as

$$\mu^p = \frac{\sum_{j \in OUT} \sum_{t=1}^{T} q_{j,t}^{\star,p-1}}{\sum_{t=1}^{T} \tau_t^{\star,p-1}},$$

where $q_{j,t}^{\star,p-1}$, for $j \in OUT$ and for all t, and $\tau_t^{\star,p-1}$, for all t, are the optimal solution of the problem at iteration $p-1$. The initialization is $\mu^1 = \sum_{j \in OUT} \underline{\rho}_j$. This iterative process stops at the end of iteration p when $\mu^{p+1} - \mu^p < \epsilon$, and here typically we consider that $\epsilon = 10^{-6}$. In that case, the optimal objective function (3.2) is equal to 0. As explained and proved in Isbell and Marlow [23], the corresponding optimal solution is also optimal for the problem with the nonlinear objective function (3.1). Their proof carries over for mixed integer linear problems (see for example in Megiddo[29]).

Another way to prove this result for the mixed integer linear case is presented below :

Proof. Suppose that for the problem composed of the linear objective function (3.2) and a given set of constraints, the optimal solution at iteration p is $q_{j,t}^{\star,p}$, for all t and $j \in OUT$, and $\tau_t^{\star,p}$, for all t. Suppose also that the $\sum_t \tau_t^{\star,p} > 0$, and that

$$\mu^p = \mu^{p+1} = \frac{\sum_{j \in OUT} \sum_t q_{j,t}^{\star,p}}{\sum_t \tau_t^{\star,p}}.$$

The optimal objective value at iteration p for the objective function (3.2) is then zero because the solutions of the optimization problems at iterations $p-1$ and p have the same productivity.

Suppose also that for the problem composed of the same constraints and of the non-linear objective function (3.1), the optimal solution is $\tilde{q}_{j,t}$, for all $j \in OUT$ and t, and $\tilde{\tau}_t$ for all t. Suppose also that the $\sum_t \tilde{\tau}_t > 0$.

Let $\tilde{\mu}$ be the optimal productivity

$$\tilde{\mu} = \frac{\sum_{j \in OUT} \sum_t \tilde{q}_{j,t}}{\sum_t \tilde{\tau}_t}$$

We have then that : $\sum_{j \in OUT} \sum_t \tilde{q}_{j,t} - \tilde{\mu} \sum_t \tilde{\tau}_t = 0$.

We cannot have $\mu^p > \tilde{\mu}$, because $\tilde{q}$ and $\tilde{\tau}$ define the optimal productivity and $q^{\star,p}$ and $\tau^{\star,p}$ are part of a feasible schedule.

Suppose now the $\tilde{\mu} > \mu^p$, i.e :

$$\tilde{\mu} = \frac{\sum_{j \in OUT} \sum_t \tilde{q}_{j,t}}{\sum_t \tilde{\tau}_t} > \frac{\sum_{j \in OUT} \sum_t q^{\star,p}_{j,t}}{\sum_t \tau^{\star,p}_t} = \mu^p$$

Then, $\sum_{j \in OUT} \sum_t \tilde{q}_{j,t} - \mu^p \sum_t \tilde{\tau}_t > \sum_{j \in OUT} \sum_t q^{\star,p}_{j,t} - \mu^p \sum_t \tau^{\star,p}_t = 0$

This is a contradiction because the optimal solution at iteration p of the problem with the objective function (3.2) is $q^{\star,p}$ and $\tau^{\star,p}$, and $\tilde{q}$, $\tilde{\tau}$ is feasible for this problem.

Therefore $\tilde{\mu} = \mu^p$ and the optimal solution of the problem with the objective function (3.2), $q^{\star,p}$ and $\tau^{\star,p}$, is also optimal for the problem with the nonlinear objective function (3.1).

The result is proved. □

This sequence of μ^p converges because the value of μ^p increases monotonically and is bounded from above. The sequence of μ^p increases monotonically because if $\mu^{p+1} < \mu^p$ then

$$\sum_{j \in OUT} \sum_{t=1}^{T} q^{*,p}_{j,t} - \mu^p \sum_{t=1}^{T} \tau^{*,p}_t < 0$$

and this implies

$$\sum_{j \in OUT} \sum_{t=1}^{T} q^{*,p}_{j,t} - \mu^p \sum_{t=1}^{T} \tau^{*,p}_t < 0 = \sum_{j \in OUT} \sum_{t=1}^{T} q^{*,p-1}_{j,t} - \mu^p \sum_{t=1}^{T} \tau^{*,p-1}_t$$

and therefore $(q^{*,p}, \tau^{*,p})$ cannot be optimal for iteration p because it is dominated by $(q^{*,p-1}, \tau^{*,p-1})$, and this is a contradiction.
Moreover, the value of the μ's is bounded from above by the upper bound on the rate of material that is processed by the continuous tasks in the set OUT, that is $\sum_{j \in OUT} \overline{\rho}_j$.

Finally, as explained in Isbell and Marlow [23], this algorithm converges in a finite number of iterations. The optimal solution of each problem with the linear objective function (3.2) is a vertex of the feasible set and there are only a finite number of vertices for this set. If the same vertex is returned for two successive iterations, the corresponding μ's are equivalent and the algorithm terminates.

For the multiple batch task case, there is only one continuous task and the output products of the plant are produced by this continuous task. The μ used at the first iteration (μ^1) equal to the lower bound on the speed of the continuous task ($\mu^1 = \underline{\rho}$) since the continuous task has to be active all the time and has a processing rate greater or equal to $\underline{\rho}$.

A typical solution for the multiple batch task case

Suppose that we have 2 reactors ($nbr\_unit = 2$), 17 time slots ($T = 17$) and a batch task can last for 4 time slots ($d = 4$). A typical evolution of the resource level of the storage tank, of the hot water rate used and of the cold water rate used over the scheduling cycle, and the corresponding values of the $z_{i,t,\Omega(l)}$ batch variables are represented in Figure 3.3. The dots represent the event times over the scheduling cycle and the two horizontal dashed lines for each resource represent the lower and upper limits available for that resource.

3.2.2 Multiple batch tasks, no continuous task

This case corresponds to a special case of the multiple batch task test case presented above. We do not consider the continuous task and the storage tank after the batch tasks. As explained in the multiple batch task test case description given at the beginning of the section, the various batch tasks share some resources (utilities and processing equipments) and some capacity restrictions have to be satisfied. The precedence and the zero waiting constraints between some batch tasks have also to be satisfied. The objective is to maximize a measure of the productivity of the process. For this special case that corresponds to the model presented in Section 2.4, we use the data of the multiple batch task case given above.

We consider first the performance of the initial (F1*) and of three strengthened formulations (F2)-(F3)-(F4). The initial formulation (F1*) is based on the constraints (2.57)-(2.62), (2.71)-(2.73) and the general resource restriction given by (2.78), (2.81) and (2.83)-(2.84) with $wf_{r,t} = w_{r,t}\ \forall r, t$ since we do not consider a continuous task for this special case. We also consider the constraint (2.70) in the model formulation. The basic strengthened formulation (F2) is based on the same set of constraints that the one of (F1*) except that (2.57) is replaced by (2.63) and (2.58) is replaced by (2.65). The complete strengthened formulation (F3) is composed of (F2) and in addition the constraints (2.64), (2.66) and (2.74). The last strengthened formulation (F4) considers the additional constraints (2.64), (2.66) and (2.74) as model cuts, i.e these constraints are removed from the initial formulation, are added to the cut pool, and generated as cuts when they are violated.

The objective function is the following ($\mu = 1$):

$$\max \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} BS z_{|BT|,t,\Omega(l)} - \sum_{t=1}^{T} \tau_t.$$

For clarity we recall these formulations below.

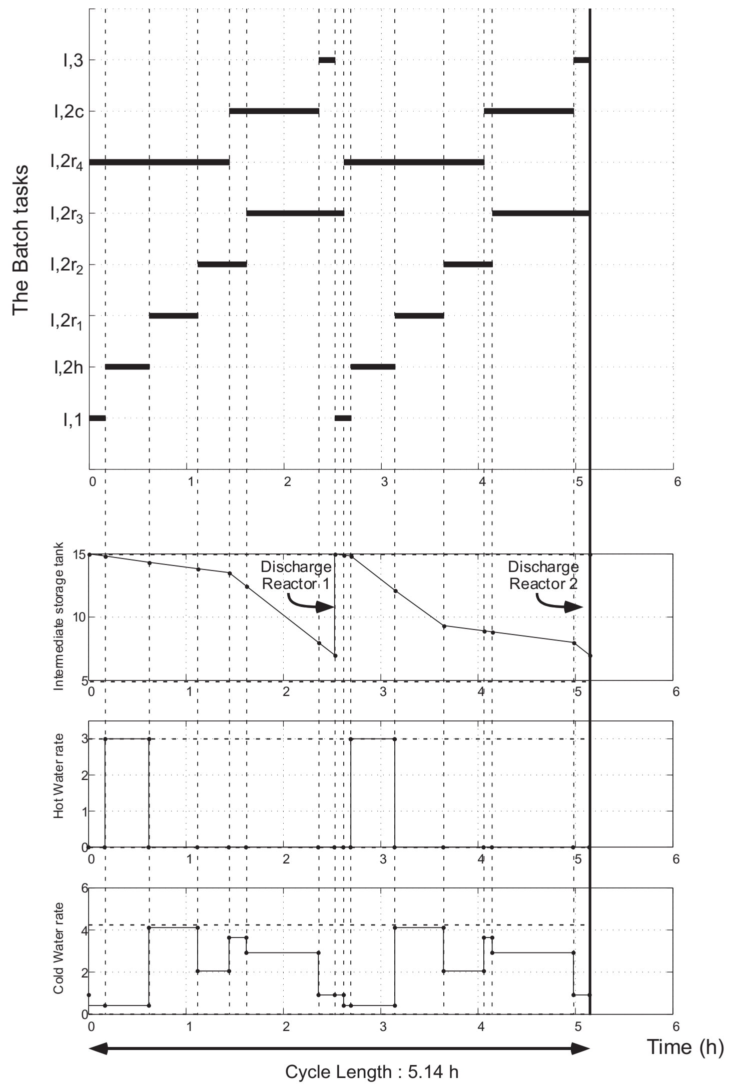

Figure 3.3: The scheduling of the batch tasks (2 reactors) and the evolution of the resources over the scheduling cycle

F1* :

nB/0C/CAP/Res//INIT

$$\max \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} BS z_{|BT|,t,\Omega(l)} - \sum_{t=1}^{T} \tau_t$$

$$p_i z_{i,t,\Omega(l)} - \sum_{k=t}^{l} \tau_{\Omega(k)} \leq 0$$

$$\forall i \in BT, t \in \{1,\dots,T\}, l \in \{t,\dots,t+d-1\} \tag{2.57}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} \leq p_i z_{i,t,\Omega(l)} + p^{\max}(l-t+1)(1-z_{i,t,\Omega(l)})$$

$$\forall i \in BT, t \in \{1,\dots,T\}, l \in \{t,\dots,t+d-1\} \tag{2.58}$$

$$\sum_{i\in BT} \sum_{l=t}^{t+d-1} z_{i,t,\Omega(l)} \leq 1 \qquad \forall t \in \{1,\dots,T\} \tag{2.59}$$

$$\sum_{i\in BT} \sum_{t=l-d+1}^{l} z_{i,\Omega(t),l} \leq 1 \qquad \forall l \in \{1,\dots,T\} \tag{2.60}$$

$$\sum_{i\in BT} \sum_{t=1}^{T} \sum_{\substack{l=t|t\leq t'\leq l \\ \text{or } t\leq t'+T\leq l}}^{t+d-1} z_{i,t,l} \leq nbr\_unit \qquad \forall t' \in \{1,\dots,T\} \tag{2.61}$$

$$\sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{i,t,\Omega(l)} \leq \left\lfloor \frac{T}{|BT|} \right\rfloor \forall i \in BT \tag{2.71}$$

$$\sum_{l=t}^{t+d-1} z_{i+1,t,\Omega(l)} = \sum_{l=t-d}^{t-1} z_{i,\Omega(l),\Omega(t-1)} \qquad \forall t \in \{1,\dots,T\},$$

$\forall i \in BT :$ task $i+1$ has to be performed after task i, without any waiting. (2.72)

$$\sum_{i\in BT} \frac{p_i}{nbr\_unit} \sum_{t=1}^{T} \sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')} - \sum_{t=1}^{T} \tau_t \leq 0 \tag{2.73}$$

$$w_{r,t} = w_{r,\Omega(t-1)} + \sum_{i\in BT} \bar{\mu}_{i,r} \sum_{t'=t-d}^{t-1} z_{i,\Omega(t'),\Omega(t-1)}$$

$$- \sum_{i\in BT} \mu_{i,r} \sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')} \qquad \forall r,t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.78}$$

$$Rmin_r \leq w_{r,t} \leq Rmax_r \qquad \forall r,t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.81}$$

nB/0C/CAP/Res//INIT (Contd)

$$w_{r,t} + \sum_{i\in BT} \bar{\mu}_{i,r} \sum_{t'=t-d+1}^{t} z_{i,\Omega(t'),t} \leq Rmax_r$$

$$\forall r,t : 1 \leq r \leq R, 1 \leq t \leq T \qquad (2.83)$$

$$Rmin_{\tilde{r}} \leq \sum_{i\in BT} \sum_{t_1=1}^{T} \sum_{\substack{t_2=t_1|t_1\leq t\leq t_2 \\ \text{or } t_1\leq t+T\leq t_2}}^{t_1+d-1} \bar{\mu}_{i,\tilde{r}} z_{i,\Omega(t_1),\Omega(t_2)} \leq Rmax_{\tilde{r}}$$

$$\forall \tilde{r} \in R : \bar{\mu}_{i,\tilde{r}} = \mu_{i,\tilde{r}}, \forall i \in BT \qquad (2.84)$$

$$\sum_{i\in BT} \sum_{t=1}^{T} \sum_{\substack{l=t|t\leq t'\leq l \\ \text{or } t\leq t'+T\leq l}}^{t+d-1} z_{i,t,\Omega(l)} + \sum_{i=1}^{|BT|} w_{r_i,t'} = nbr\_unit \ \forall t' \in \{1,\ldots,T\} \qquad (2.70)$$

$$z_{i,t,\Omega(l)} \in \{0,1\} \forall i \in BT, t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\},$$
$$\tau_t \geq 0 \forall t \in \{1,\ldots,T\} \qquad (2.62)$$

F2 :

nB/0C/CAP/Res//STR1

$$\max \sum_{t=1}^{T} \sum_{l=t}^{t+d-1} BS z_{|BT|,t,\Omega(l)} - \sum_{t=1}^{T} \tau_t$$

$$\sum_{i\in BT} \sum_{k=t|t\neq l}^{l} p_i z_{i,t,\Omega(k)} + \sum_{i\in BT} p_i z_{i,\Omega(l),\Omega(l)} - \sum_{k=t}^{l} \tau_{\Omega(k)} \leq 0$$

$$\forall t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\} \qquad (2.63)$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} \leq \sum_{i\in BT} \sum_{k=l-d+1}^{l} p_i z_{i,\Omega(k),\Omega(l)}$$

$$+ p^{\max}\Big((l-t+1) - \sum_{i\in BT} \sum_{k=l-d+1}^{l} \min\{l-t+1, l-k+1\} z_{i,\Omega(k),\Omega(l)}\Big)$$

$$+ \sum_{i\in BT} (p_i - p^{\max}) \sum_{t'=t|l=t+d-1}^{l-1} z_{i,t,\Omega(t')}$$

$$\forall t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\} \qquad (2.65)$$

nB/0C/CAP/Res//STR1 (Contd)

$$\sum_{i \in BT} \sum_{l=t}^{t+d-1} z_{i,t,\Omega(l)} \leq 1 \qquad \forall t \in \{1,\ldots,T\} \tag{2.59}$$

$$\sum_{i \in BT} \sum_{t=l-d+1}^{l} z_{i,\Omega(t),l} \leq 1 \qquad \forall l \in \{1,\ldots,T\} \tag{2.60}$$

$$\sum_{i \in BT} \sum_{t=1}^{T} \sum_{\substack{l=t \mid t \leq t' \leq l \\ \text{or } t \leq t'+T \leq l}}^{t+d-1} z_{i,t,l} \leq nbr\_unit \qquad \forall t' \in \{1,\ldots,T\} \tag{2.61}$$

$$\sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{i,t,\Omega(l)} \leq \left\lfloor \frac{T}{|BT|} \right\rfloor \forall i \in BT \tag{2.71}$$

$$\sum_{l=t}^{t+d-1} z_{i+1,t,\Omega(l)} = \sum_{l=t-d}^{t-1} z_{i,\Omega(l),\Omega(t-1)} \qquad \forall t \in \{1,\ldots,T\},$$

$\forall i \in BT:$ task $i+1$ has to be performed after task i, without any waiting. (2.72)

$$\sum_{i \in BT} \frac{p_i}{nbr\_unit} \sum_{t=1}^{T} \sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')} - \sum_{t=1}^{T} \tau_t \leq 0 \tag{2.73}$$

$$w_{r,t} = w_{r,\Omega(t-1)} + \sum_{i \in BT} \bar{\mu}_{i,r} \sum_{t'=t-d}^{t-1} z_{i,\Omega(t'),\Omega(t-1)} - \sum_{i \in BT} \mu_{i,r} \sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')}$$

$$\forall r,t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.78}$$

$$Rmin_r \leq w_{r,t} \leq Rmax_r \qquad \forall r,t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.81}$$

$$w_{r,t} + \sum_{i \in BT} \bar{\mu}_{i,r} \sum_{t'=t-d+1}^{t} z_{i,\Omega(t'),t} \leq Rmax_r$$

$$\forall r,t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.83}$$

$$Rmin_{\tilde{r}} \leq \sum_{i \in BT} \sum_{t_1=1}^{T} \sum_{\substack{t_2=t_1 \mid t_1 \leq t \leq t_2 \\ \text{or } t_1 \leq t+T \leq t_2}}^{t_1+d-1} \bar{\mu}_{i,\tilde{r}} z_{i,\Omega(t_1),\Omega(t_2)} \leq Rmax_{\tilde{r}}$$

$$\forall \tilde{r} \in R : \bar{\mu}_{i,\tilde{r}} = \mu_{i,\tilde{r}}, \forall i \in BT \tag{2.84}$$

$$\sum_{i \in BT} \sum_{t=1}^{T} \sum_{\substack{l=t \mid t \leq t' \leq l \\ \text{or } t \leq t'+T \leq l}}^{t+d-1} z_{i,t,\Omega(l)} + \sum_{i=1}^{|BT|} w_{r_i,t'} = nbr\_unit \; \forall t' \in \{1,\ldots,T\} \tag{2.70}$$

$$z_{i,t,\Omega(l)} \in \{0,1\} \qquad \forall i \in BT, t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\}$$

$$\tau_t \geq 0 \qquad \forall t : 1 \leq t \leq T \tag{2.62}$$

F3/F4 : The formulations F3 and F4 are composed of F2 and of the following constraints :

nB/0C/CAP/Res//[STR2,STRM][1]

nB/0C/CAP/Res//STR1

$$\sum_{i \in BT} \sum_{k=t|t\neq l}^{l} p_i z_{i,\Omega(k),\Omega(l)} + \sum_{i \in BT} p_i z_{i,t,t} - \sum_{k=t}^{l} \tau_{\Omega(k)} \leq 0$$

$$\forall t, l : 1 \leq t \leq T, t \leq l \leq t+d-1 \qquad (2.64)$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} \leq \sum_{i \in BT} \sum_{k=t}^{t+d-1} p_i z_{i,t,\Omega(k)}$$

$$+ p^{\max}\Big((l-t+1) - \sum_{i \in BT} \sum_{k=t}^{t+d-1} \min\{l-t+1, k-t+1\} z_{i,t,\Omega(k)}\Big)$$

$$+ \sum_{i \in BT} (p_i - p^{\max}) \sum_{t'=t+1|l=t+d-1}^{l} z_{i,\Omega(t'),\Omega(l)}$$

$$\forall t, l : 1 \leq t \leq T, t \leq l \leq t+d-1 \qquad (2.66)$$

$$\sum_{k=t}^{l} \min(coeff_k, nbr\_unit, d)\tau_k \geq \sum_{i \in BT} p_i \left(\sum_{t_1=t}^{l} \sum_{t_2=t_1}^{\min(l,t_1+d-1)} z_{i,\Omega(t_1),\Omega(t_2)} \right)$$

$$\forall t \in \{1, \ldots, T\}, l \in \{t+2, \ldots, t+T+d-3\} \qquad (2.74)$$

The characteristics of the small (resp. large) instance in Table 3.4 are the following : $T = 17$, $d = 4$ and $nbr\_unit = 2$ (resp. 3). There are 544 binary variables, 136 integer variables and 52 continuous variables.

Small Instance ($T = 17, nbr\_unit = 2$)				
	F1*	F2	F3	F4
Constraints	1430	495	886	886
Nodes	5954	**785**	190	141
Time	135.5 s	**23 s**	51 s	35 s

Large Instance ($T = 17, nbr\_unit = 3$)				
	F1*	F2	F3	F4
Constraints	1430	495	886	886
Nodes	61375	**19895**	13260	13627
Time	1712 s	**367** s	1589 s	1926 s

Table 3.4: Comparison between the initial (F1*) and the strengthened formulations (F2)-(F4); Multiple batch tasks, resource restrictions.

In Table 3.4, the strengthened formulation (F2) provides the optimal solution quicker for the small and the larger instances. The formulation (F3)-(F4) needs fewer nodes to solve the problem instances. A better cutting plane strategy should be developed in order to take advantage of the valid inequalities found and reduce the computing time.

3.2.3 Multiple batch tasks and one continuous task

We consider the multiple batch task case described above. We show first that exact solution methods can only be used for limited size problem instances. Then, we introduce MIP heuristic methods taking advantage of the strengthened formulation in order to solve some larger instances. We show that, with theses MIP heuristic methods, we can find good feasible solutions quickly.

Formulations

We pay attention first to the performance of the initial (F1) and of three strengthened formulations (F2)-(F3)-(F4). The initial formulation (F1) is based on the constraints (2.57)-(2.62), (2.70)-(2.73) and (2.76)-(2.84). The basic strengthened formulation (F2) is based on the constraints (2.59)-(2.62), (2.63), (2.65), (2.70)-(2.73), (2.77)-(2.84) and (2.86). The complete strengthened formulation (F3) is composed of (F2) and in addition the constraints (2.64), (2.66) and (2.74). The last strengthened formulation (F4) considers the additional constraints (2.64), (2.66) and (2.74) as model cuts, i.e these constraints are removed from the initial formulation, are added to the cut pool, and generated as cuts when they are violated.

The initial objective function is the following ($\mu^1 = \underline{\rho} = 1$) :

$$\max \sum_{t=1}^{T} q_t - \sum_{t=1}^{T} \tau_t.$$

For clarity we recall these formulations below.

F1 :

nB/1C/CAP/Res//INIT

$$\max \sum_{t=1}^{T} q_t - \sum_{t=1}^{T} \tau_t$$

$$p_i z_{i,t,\Omega(l)} - \sum_{k=t}^{l} \tau_{\Omega(k)} \leq 0$$

$$\forall i \in BT, t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\} \tag{2.57}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} \leq p_i z_{i,t,\Omega(l)} + p^{\max}(l-t+1)(1-z_{i,t,\Omega(l)})$$

$$\forall i \in BT, t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\} \tag{2.58}$$

$$\sum_{i \in BT} \sum_{l=t}^{t+d-1} z_{i,t,\Omega(l)} \leq 1 \qquad \forall t \in \{1,\ldots,T\} \tag{2.59}$$

$$\sum_{i \in BT} \sum_{t=l-d+1}^{l} z_{i,\Omega(t),l} \leq 1 \qquad \forall l \in \{1,\ldots,T\} \tag{2.60}$$

$$\sum_{i \in BT} \sum_{t=1}^{T} \sum_{\substack{l=t | t \leq t' \leq l \\ \text{or } t \leq t'+T \leq l}}^{t+d-1} z_{i,t,l} \leq nbr\_unit \qquad \forall t' \in \{1,\ldots,T\} \tag{2.61}$$

$$\sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{i,t,\Omega(l)} \leq \left\lfloor \frac{T}{|BT|} \right\rfloor \forall i \in BT \tag{2.71}$$

$$\sum_{l=t}^{t+d-1} z_{i+1,t,\Omega(l)} = \sum_{l=t-d}^{t-1} z_{i,\Omega(l),\Omega(t-1)} \qquad \forall t \in \{1,\ldots,T\}, \forall i \in BT :$$

task $i+1$ has to be performed after task i, without any waiting. (2.72)

$$\sum_{i \in BT} \frac{p_i}{nbr\_unit} \sum_{t=1}^{T} \sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')} - \sum_{t=1}^{T} \tau_t \leq 0 \tag{2.73}$$

$$q_t \leq \overline{\rho}\tau_t \qquad \forall t : 1 \leq t \leq T \tag{2.76}$$

$$q_t \geq \underline{\rho}\tau_t \qquad \forall t : 1 \leq t \leq T \tag{2.77}$$

$$w_{r,t} = wf_{r,\Omega(t-1)} + \sum_{i \in BT} \bar{\mu}_{i,r} \sum_{t'=t-d}^{t-1} z_{i,\Omega(t'),\Omega(t-1)}$$

$$- \sum_{i \in BT} \mu_{i,r} \sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')} \qquad \forall r,t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.78}$$

$$wf_{r,t} = w_{r,t} + \lambda_r q_t \qquad \forall r,t : r \in R_m, 1 \leq t \leq T \tag{2.79}$$

$$wf_{r,t} = w_{r,t} \forall t,\ \forall r \in R_e \cup R_u \tag{2.80}$$

nB/1C/CAP/Res//INIT (Contd)

$$Rmin_r \leq w_{r,t} \leq Rmax_r \qquad \forall r,t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.81}$$

$$Rmin_r \leq wf_{r,t} \leq Rmax_r \qquad \forall r,t : r \in R_m, 1 \leq t \leq T \tag{2.82}$$

$$wf_{r,t} + \sum_{i \in BT} \bar{\mu}_{i,r} \sum_{t'=t-d+1}^{t} z_{i,\Omega(t'),t} \leq Rmax_r$$
$$\forall r,t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.83}$$

$$Rmin_{\tilde{r}} \leq \sum_{i \in BT} \sum_{t_1=1}^{T} \sum_{\substack{t_2=t_1 | t_1 \leq t \leq t_2 \\ \text{or } t_1 \leq t+T \leq t_2}}^{t_1+d-1} \bar{\mu}_{i,\tilde{r}} z_{i,\Omega(t_1),\Omega(t_2)} \leq Rmax_{\tilde{r}}$$
$$\forall \tilde{r} \in R : \bar{\mu}_{i,\tilde{r}} = \mu_{i,\tilde{r}}, \forall i \in BT \tag{2.84}$$

$$\sum_{i \in BT} \sum_{t=1}^{T} \sum_{\substack{l=t | t \leq t' \leq l \\ \text{or } t \leq t'+T \leq l}}^{t+d-1} z_{i,t,\Omega(l)} + \sum_{i=1}^{|BT|} w_{r_i,t'} = nbr\_unit \ \forall t' \in \{1,\ldots,T\} \tag{2.70}$$

$$z_{i,t,\Omega(l)} \in \{0,1\} \forall i \in BT, t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\},$$
$$\tau_t \geq 0 \forall t \in \{1,\ldots,T\} \tag{2.62}$$

F2 :

nB/1C/CAP/Res//STR1

$$\max \sum_{t=1}^{T} q_t - \sum_{t=1}^{T} \tau_t$$

$$\sum_{i \in BT} \sum_{k=t | t \neq l}^{l} p_i z_{i,t,\Omega(k)} + \sum_{i \in BT} p_i z_{i,\Omega(l),\Omega(l)} - \sum_{k=t}^{l} \tau_{\Omega(k)} \leq 0$$
$$\forall t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\} \tag{2.63}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} \leq \sum_{i \in BT} \sum_{k=l-d+1}^{l} p_i z_{i,\Omega(k),\Omega(l)}$$
$$+ p^{\max}\Big((l-t+1) - \sum_{i \in BT} \sum_{k=l-d+1}^{l} \min\{l-t+1, l-k+1\} z_{i,\Omega(k),\Omega(l)}\Big)$$
$$+ \sum_{i \in BT} (p_i - p^{\max}) \sum_{t'=t | l=t+d-1}^{l-1} z_{i,t,\Omega(t')}$$
$$\forall t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\} \tag{2.65}$$

$$\sum_{i \in BT} \sum_{l=t}^{t+d-1} z_{i,t,\Omega(l)} \leq 1 \qquad \forall t \in \{1,\ldots,T\} \tag{2.59}$$

nB/1C/CAP/Res//STR1 (Contd)

$$\sum_{i\in BT}\sum_{t=l-d+1}^{l} z_{i,\Omega(t),l} \leq 1 \qquad \forall l \in \{1,\ldots,T\} \tag{2.60}$$

$$\sum_{i\in BT}\sum_{t=1}^{T}\sum_{\substack{l=t|t\leq t'\leq l\\ \text{or } t\leq t'+T\leq l}}^{t+d-1} z_{i,t,l} \leq nbr\_unit \qquad \forall t' \in \{1,\ldots,T\} \tag{2.61}$$

$$\sum_{t=1}^{T}\sum_{l=t}^{t+d-1} z_{i,t,\Omega(l)} \leq \left\lfloor \frac{T}{|BT|} \right\rfloor \forall i \in BT \tag{2.71}$$

$$\sum_{l=t}^{t+d-1} z_{i+1,t,\Omega(l)} = \sum_{l=t-d}^{t-1} z_{i,\Omega(l),\Omega(t-1)} \qquad \forall t \in \{1,\ldots,T\}, \forall i \in BT: \text{ task}$$

$i+1$ has to be performed after task i, without any waiting. (2.72)

$$\sum_{i\in BT}\frac{p_i}{nbr\_unit}\sum_{t=1}^{T}\sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')} - \sum_{t=1}^{T}\tau_t \leq 0 \tag{2.73}$$

$$q_t \leq \overline{\rho}\tau_t - \sum_{i\in BT}\max(p_i\overline{\rho} - (Rmax_{\tilde{r}} - Rmin_{\tilde{r}}); 0) z_{i,t,t}$$

$$\text{for } \tilde{r} \in R : \lambda_{\tilde{r}} = -1, \forall t \in \{1,\ldots,T\} \tag{2.86}$$

$$q_t \geq \underline{\rho}\tau_t \qquad \forall t : 1 \leq t \leq T \tag{2.77}$$

$$w_{r,t} = wf_{r,\Omega(t-1)} + \sum_{i\in BT}\bar{\mu}_{i,r}\sum_{t'=t-d}^{t-1} z_{i,\Omega(t'),\Omega(t-1)} - \sum_{i\in BT}\mu_{i,r}\sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')}$$

$$\forall r,t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.78}$$

$$wf_{r,t} = w_{r,t} + \lambda_r q_t \qquad \forall r,t : r \in R_m, 1 \leq t \leq T \tag{2.79}$$

$$wf_{r,t} = w_{r,t}\forall t,\ \forall r \in R_e \cup R_u \tag{2.80}$$

$$Rmin_r \leq w_{r,t} \leq Rmax_r \qquad \forall r,t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.81}$$

$$Rmin_r \leq wf_{r,t} \leq Rmax_r \qquad \forall r,t : r \in R_m, 1 \leq t \leq T \tag{2.82}$$

$$wf_{r,t} + \sum_{i\in BT}\bar{\mu}_{i,r}\sum_{t'=t-d+1}^{t} z_{i,\Omega(t'),t} \leq Rmax_r$$

$$\forall r,t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.83}$$

$$Rmin_{\tilde{r}} \leq \sum_{i\in BT}\sum_{t_1=1}^{T}\sum_{\substack{t_2=t_1|t_1\leq t\leq t_2\\ \text{or } t_1\leq t+T\leq t_2}}^{t_1+d-1} \bar{\mu}_{i,\tilde{r}} z_{i,\Omega(t_1),\Omega(t_2)} \leq Rmax_{\tilde{r}}$$

$$\forall \tilde{r} \in R : \bar{\mu}_{i,\tilde{r}} = \mu_{i,\tilde{r}}, \forall i \in BT \tag{2.84}$$

$$\sum_{i\in BT}\sum_{t=1}^{T}\sum_{\substack{l=t|t\leq t'\leq l\\ \text{or } t\leq t'+T\leq l}}^{t+d-1} z_{i,t,\Omega(l)} + \sum_{i=1}^{|BT|} w_{r_i,t'} = nbr\_unit\ \forall t' \in \{1,\ldots,T\} \tag{2.70}$$

nB/1C/CAP/Res//STR1 (Contd)

$$z_{i,t,\Omega(l)} \in \{0,1\} \qquad \forall i \in BT, t \in \{1,\dots,T\}, l \in \{t,\dots,t+d-1\}$$
$$\tau_t \geq 0 \qquad \forall t : 1 \leq t \leq T \qquad (2.62)$$

F3/F4 : The formulations F3 and F4 are composed of F2 and of the following constraints :

nB/1C/CAP/Res//[STR2,STRM][1]

nB/1C/CAP/Res//STR1

$$\sum_{i\in BT}\sum_{k=t|t\neq l}^{l} p_i z_{i,\Omega(k),\Omega(l)} + \sum_{i\in BT} p_i z_{i,t,t} - \sum_{k=t}^{l}\tau_{\Omega(k)} \leq 0$$
$$\forall t,l : 1 \leq t \leq T, t \leq l \leq t+d-1 \qquad (2.64)$$

$$\sum_{k=t}^{l}\tau_{\Omega(k)} \leq \sum_{i\in BT}\sum_{k=t}^{t+d-1} p_i z_{i,t,\Omega(k)}$$
$$+ p^{\max}\Big((l-t+1) - \sum_{i\in BT}\sum_{k=t}^{t+d-1}\min\{l-t+1,k-t+1\} z_{i,t,\Omega(k)}\Big)$$
$$+ \sum_{i\in BT}(p_i - p^{\max}) \sum_{t'=t+1|l=t+d-1}^{l} z_{i,\Omega(t'),\Omega(l)}$$
$$\forall t,l : 1 \leq t \leq T, t \leq l \leq t+d-1 \qquad (2.66)$$

$$\sum_{k=t}^{l}\min(coeff_k, nbr\_unit, d)\tau_k \geq \sum_{i\in BT} p_i \left(\sum_{t_1=t}^{l}\sum_{t_2=t_1}^{\min(l,t_1+d-1)} z_{i,\Omega(t_1),\Omega(t_2)}\right)$$
$$\forall t \in \{1,\dots,T\}, l \in \{t+2,\dots,t+T+d-3\} \qquad (2.74)$$

Exact solution methods

The characteristics of the small (resp. large) instance in Table 3.5 are the following : $T = 10$(resp. 17), $d = 4$, $nbr\_unit = 2$, There are 80 (resp. 136) integer variables, 320 (resp. 544) binary variables and 61 (resp. 103) continuous variables. The number of constraints in the strengthened formulations (F2)-(F4) is smaller because the timing constraints have been aggregated over all batch tasks.

Table 3.5 reports on the solution of the multiple batch task case using the four formulations and a standard MIP solver. For both instances and all formulations, the maximal productivity for the test case is obtained by solving two mixed integer optimization problems with the objective function (3.2). We start with $\mu^1 = \underline{\rho}$. In Table 3.5, "Nodes" represents the total number of Branch-and-Bound nodes needed in order to solve the two iterations to optimality, "Time" represents the

Small Instance (T=10)				
nbr unit=2	F1	F2	F3	F4
Constraints	889	339	499	499
Nodes	118	21	3	**1**
Time	2.3 s	1.4 s	2 s	**0.7 s**
Productivity	3.11	3.11	3.11	**3.11**
Large Instance (T=17)				
nbr unit=2	F1	F2	F3	F4
Constraints	1498	563	954	954
Nodes	**274**	334	174	644
Time	**13.3 s**	13.8 s	44 s	153.7 s
Productivity	**3.11**	3.11	3.11	3.11

Table 3.5: First comparison between the initial (F1) and the strengthened formulations (F2)-(F4)

corresponding total CPU time and "Productivity" is the maximal productivity obtained for the test case.

In Table 3.5, we can see that the small and the large instances are solved easily by the four formulations. The small instance is solved by using the formulation (F4) at the root node. However, for the large instance, (F1) and (F2) give the best results and are quite comparable. In order to really test the quality of the two formulations, we need to solve more instances.

In Table 3.6, we consider a second case where the instances characteristics and the number of variables are the same as those in Table 3.5, except that $nbr\_unit = 3$. In the next Tables, the star (*) after the productivity measure means that the iterative procedure (that gives the optimal solution of the problem with the nonlinear objective function) is stopped at the end of some iteration before obtaining the optimal productivity solution. This occurs when the CPU solution time is too large.

For both instances in Table 3.6, the strengthened formulations (F2)-(F4) obtain the optimal solution in fewer nodes. The strengthened formulation (F2) solves both instances quicker than the other three formulations. Not surprisingly, the strengthened formulations (F3)-(F4) taking into account all valid inequalities need fewer nodes than the other two formulations.

Finally, the characteristics and the number of variables of the small and the large instances in Table 3.7 are the same as the one in Table 3.5, except that $nbr\_unit = 4$. For the large instance and for all formulations except (F2), we are not even able to solve the first iteration of the linearized objective to optimality. For such cases, a star is added to the CPU solution time of the corresponding instance in Table 3.7 and we calculate the remaining duality gap for that iteration as $\frac{\text{Best bound - Best solution}}{\text{Best bound}}$ and the productivity is the one corresponding to the best

Small Instance (T=10)				
nbr unit=3	F1	F2	F3	F4
Constraints	889	339	499	499
Nodes	1342	1186	700	630
Time	26.1 s	**14.5 s**	20.4 s	18 s
Productivity	3.27	**3.27**	3.27	3.27
Large Instance (T=17)				
nbr unit=3	F1	F2	F3	F4
Constraints	1498	563	954	954
Nodes	48524	24009	12169	16271
Time	1744 s	**496 s**	1617 s	2275 s
Productivity	3.21*	**3.21***	3.21*	3.21*
Nbr of iterations	1	1	1	1

Table 3.6: Second comparison between the initial (F1) and the strengthened formulations (F2)-(F4), where Nbr of iterations is the number of iterations of the objective linearization procedure

schedule obtained with respect to the linearized objective.

Small Instance (T=10)				
nbr unit=4	F1	F2	F3	F4
Constraints	889	339	499	499
Nodes	1365	**766**	772	1114
Time	20.61 s	**8.9 s**	18 s	25 s
Productivity	3.27	**3.27**	3.27	3.27
Nbr of iterations	2	2	2	2
Large Instance (T=17)				
nbr unit=4	F1	F2	F3	F4
Constraints	1498	563	954	954
Nodes	60900	**25609**	13000	14000
Time	2000 s*	**516 s**	2000 s*	2000 s*
Productivity	3.21*	**3.21***	3.21*	3.21*
Nbr of iterations	1	1	1	1
Remaining Duality gap	10.72 %	**0 %**	5.56 %	2.5 %

Table 3.7: Third comparison between the initial (F1) and the strengthened formulations (F2)-(F4)

For the small instance in Table 3.7, the strengthened formulation (F2) gets the optimal solution quicker and with less nodes than the other three formulations. For the large instance, only the strengthened formulation (F2) gives the optimal solution for the first iteration of the linearization procedure in a reasonable amount of time.

In Tables 3.5 - 3.7, we can observe that for small instances, the four formulations give good results. For the large instances, the strengthened formulation (F2) provides almost always the optimal solution with respect to the linearized objective in less computing time and is more efficient than the initial one (F1). The strengthened formulations (F3)-(F4) solve the problem instances with almost always fewer nodes but are on average slower than (F2) in term of CPU time. A better cutting plane strategy should be developed in order to take advantage of the valid inequalities found in a reduced amount of computing time.

We now comment the evolution of the results obtained in Tables 3.5-3.7 with respect to the number of units. In Table 3.5, we consider two units and we can solve the productivity maximization problem for the two instances to optimality in a reasonable amount of time by using the four model formulations in two iterations of the objective linearization procedure. In Table 3.6, we consider three units and in this case, only the first iteration of the linearization procedure for the large instance can be solved to optimality in a reasonable amount of time by using the four model formulations. In Table 3.7, we consider four units and in this case, only the basic strengthened formulation (F2) can solve to optimality the first iteration of the linearization procedure for this larger instance in a reasonable amount of time.

However, we can observe that for larger instances, the exact methods cannot solve the problems in a reasonable amount of time. Therefore, we consider heuristic methods in order to obtain good feasible solutions for these larger instances quickly.

Heuristic solution methods

We introduce below a heuristic method that combines the use of various well known MIP based heuristic methods. Our objective in using MIP based heuristics is to find good feasible solutions quickly by taking advantage of the improved formulations described above. We outline here the well known heuristic methods and we explain how we combine them. All these heuristics are described in Pochet and Wolsey [38].

The main heuristic method used is **Relax-and-Fix**, see Stadtler [46]. The various steps of our specific implementation are the following :

1. We decompose the set of binary variables $z_{i,t,\Omega(l)}$ in various non-disjoint sets $S_1, S_2, \ldots, S_T$. In our case, the set S_t is composed of the binary variables that can be active during time slot t. Every variable will be part of at least one set. In other words, for each $t \leq k \leq l \leq t+d-1$, the variable $z_{i,t,\Omega(l)} \in S_{\Omega(k)}$.

2. We start by imposing the integrality restriction for the binary variables in the sets S_1 and S_2 and we relax this constraint (i.e. $0 \leq z \leq 1$) for variables in the sets $(S_3 \cup \ldots \cup S_T) \setminus (S_1 \cup S_2)$. We solve the corresponding relaxed MIP problem and we obtain an upper bound for the original problem (for

the maximization of the linear objective function).
We fix the binary variables in the set S_1 at their optimal values.
This is the end of the first iteration of Relax-and-Fix.

3. Then, we impose the integrality condition for the binary variables in the sets S_2 and S_3 and we relax the integrality constraints for the rest of the variables, i.e. for variables in the sets $(S_4 \cup \ldots \cup S_T) \setminus (S_1 \cup S_2 \cup S_3)$. We solve the corresponding relaxed MIP problem and we fix the binary variables in the set S_2 at their optimal value. This is the end of the second iteration.

4. And so on, up to the last iteration where variables in $(S_{T-1} \cup S_T) \setminus (S_1 \cup \ldots \cup S_{T-2})$ are binary, the other variables being fixed by previous iterations.

The difficulty is that at each step, it is possible that the MIP problem becomes infeasible because of previous variables being fixed at inconsistent values. In such a case, we combine this first heuristic method with a neighborhood search method similar to the **Local Branching** heuristic method (see Fischetti and Lodi [20]). More specifically, if the MIP problem is infeasible at iteration p of Relax-and-Fix, we solve a relaxation of this MIP problem where we impose that the variables previously fixed ($z_{i,t,\Omega(l)} = z^{\star,f}_{i,t,\Omega(l)}$ for $(i,t,l) \in F$) have to remain binary but we allow a limited number k of them to change their values.
This can be modeled by adding the following constraint :

$$\sum_{(i,t,l)\in F | z^{\star,f}_{i,t,\Omega(l)}=0} z_{i,t,\Omega(l)} + \sum_{(i,t,l)\in F | z^{\star,f}_{i,t,\Omega(l)}=1} (1 - z_{i,t,\Omega(l)}) \leq k$$

This additional Local Branching step is performed to improve the robustness of the Relax-and-Fix heuristic method.

An additional way to improve the quality of the feasible solutions obtained by the heuristic is to use the Local Branching heuristic method at the end of the Relax-and-Fix algorithm. It consists in allowing k variables $z_{i,t,\Omega(l)}$ to change their values compared to the final Relax-and-Fix solution. This Local Branching step is repeated until no significant improvement to the objective is obtained. This final Local Branching step defines an improvement heuristic starting from the Relax-and-Fix solution.

Given the superiority of reformulation (F2), the 5 heuristic methods used and compared to solve larger instances are the following :

1. Truncated B&B (F1) : Based on the initial formulation (F1), we solve the problem and we stop the branch and bound algorithm before the end of the resolution.

2. Truncated B&B (F2) : Based on the basic strengthened formulation (F2), we solve the problem and we stop the branch and bound algorithm before the end of the resolution.

3. Relax-and-Fix (F1) : The heuristic method described above based on the initial formulation (F1). The parameter k for Local Branching, during and

at the end of Relax-and-Fix, takes a value in the set $\{6, 9, 12\}$. We always start with $k = 6$. During Relax-and-Fix, we increase k only if we could not find a feasible solution, otherwise we stop Local Branching. At the end of Relax-and-Fix, we increase k as long as a significant improvement is observed.

4. Relax-and-Fix (F2*) : The heuristic method described above based on the formulation (F2) plus constraints (2.64) and (2.66). The parameter k follows the same rules as for method 3.

5. Relax-and-Fix/LB (F2*) : It is a variant of method 4. The differences are that, at each iteration p of Relax-and-Fix : (i) we allow to change $k1$ values of variables in S_{p-1} $(p \geq 2)$, (ii) the binary variables in the set $\{z_{i,t,p-2} : i \in BT, t \in \{p-d-1, \ldots, p-2\}\}$(if $p \geq 3$) are fixed (for all subsequent iterations of Relax-and-Fix) at the optimal value obtained at iteration $p-1$, (iii) the z variables in the set $(S_p \cup S_{p+1})$ have to be binary, and finally (iv) the integrality condition is relaxed on the others z variables. Here, we set $k1 = 3$ and the parameter k follows the same rules as for methods 3 and 4.

For the last three heuristic methods, we have imposed a maximum time for solving each MIP optimization subproblem in the heuristic. We choose to set this parameter to 500 sec. If at the end of the 500 sec., we have obtained a feasible solution, we stop the resolution of the current problem and proceed to the next step of the heuristic method. Otherwise, we continue to solve the current subproblem until a first feasible solution is obtained.

For the small instance reported on in Table (3.8), we solve two iterations of the linearization of the objective function except for the heuristic method 3. For the large instance of Table (3.8) and for Tables (3.9)-(3.10), we only try to solve the first iteration of the linearization of the objective function. Therefore, we put a star (*) after the productivity obtained because we could not prove with one iteration of the linearization that the productivity is optimal. Moreover, for the heuristic 1 or 2, the resolution of the first linearized objective problem was sometimes stopped before optimality was proved. For such cases, a star (*) is added to the CPU solution time of the corresponding instance in the Table. The duality gap indicates the remaining gap with respect to the linearized objective and is defined as $\frac{\text{Best bound - Best solution}}{\text{Best bound}}$. For the Relax-and-Fix methods, the best bound is the optimal solution obtained at the first iteration (before any fixing), and the best solution is the final Relax-and-Fix solution obtained.

The first heuristic comparison is proposed in Table 3.8. The characteristics of the small (resp. large) instance in Table 3.8 are the following : $T = 18(\text{resp. } 26), d = 7, \text{nbr unit} = 2, \underline{\rho} = 1[ru/h], \overline{\rho} = 6[ru/h]$. There are 144 (resp. 208) integer variables, 1008 (resp. 1456) binary variables and 109 (resp. 157) continuous variables.

In Table 3.8, and for the small instance, the heuristic method 2 provides the optimal solution quicker and with fewer nodes than the other heuristic methods. It is interesting to note for the small instance that the heuristic methods 4 and 5 are also able to solve the problem to optimality. Regarding the large instance, the

2 units		B&B (F1)	B&B (F2)	R&F (F1)	R&F (F2*)	R&F/LB (F2*)
Small Instance T=18	# Constr.	2449	703	2449	919	919
	Nodes	1643	**1**	18524	10	23
	D. Gap (%)	0	**0**	50.4	0	0
	Time	97 s	**4 s**	484 s	9 s	21 s
	Prod.	3.11	**3.11**	3.05*	3.11	3.11
	Nbr of iterations	2	2	1	2	2
Large Instance T=26	# Constr.	3529	1007	3529	1319	1319
	Nodes	6200	9300	15192	493	**48**
	D. Gap (%)	14.8	0.74	40	1	**0**
	Time	1000 s*	1000 s*	556 s	105 s	**51 s**
	Prod.	2.37*	3.06*	2.51*	3*	**3.11***

Table 3.8: First heuristic comparison, *nbr_unit* = 2

heuristic method 5 provides the optimal solution of the first linearization quicker and with fewer nodes.

The second heuristic comparison is proposed in Table 3.9. The characteristics of the small (resp. large) instance in Table 3.9 are the same as the one given for Table 3.8, except that *nbr_unit* = 3. Again, for every instance of this table, we only perform one linearization iteration.

3 units		B&B (F1)	B&B (F2)	R&F (F1)	R&F (F2*)	R&F/LB (F2*)
Small Instance T=18	# Constr.	2449	703	2449	919	919
	Nodes	13400	7490	12918	10647	2575
	D. Gap (%)	11	8.13	11	7.4	8.13
	Time	1000 s*	517 s*	589 s	403 s	196 s
	Prod.	3.32*	3.6*	3.32*	3.67*	3.6*
Large Instance T=26	# Constr.	3529	1007	3529	1319	1319
	Nodes	4300	5400	2019	3762	2558
	D. Gap (%)	48.9	40.67	48.9	11.56	22.2
	Time	1000 s*	1000 s*	109 s	700 s	332 s
	Prod.	2.51*	3.32*	2.51*	3.27*	2.57*

Table 3.9: Second heuristic comparison, *nbr_unit* = 3

The heuristic methods 4 or 5 seem to outperform the other three methods, a better solution in terms of remaining duality gap is obtained quicker. The heuristic 4 gives for the two instances a solution with a smaller remaining duality gap and a higher productivity than heuristic 5 but needs more running time to compute these better results. Therefore, both heuristic methods 4 and 5 can be interesting in order to compute good feasible solutions quickly.

The third heuristic comparison is proposed in Table 3.10. The characteristics of the small (resp. large) instance in Table 3.9 are the same as the one of Table 3.8

except that *nbr_unit* $= 4$. Again, for every instance of this table, we only perform one linearization iteration.

4 units		B&B (F1)	B&B (F2)	R&F (F1)	R&F (F2*)	R&F/LB (F2*)
Small Instance T=18	# Constr.	2449	703	2449	919	919
	Nodes	**489**	2545	1901	11960	13969
	D. Gap (%)	**16.1**	16.1	16.1	17.33	16.1
	Time	**58 s***	270 s*	95 s	1167 s	1265 s
	Prod.	**3.32***	3.32*	3.32*	3.21*	3.32*
Large Instance T=26	# Constr.	3529	1007	3529	1319	1319
	Nodes	8800	9300	2302	6420	5816
	D. Gap (%)	44.06	42.63	51.8	**16.3**	42.2
	Time	2000 s*	2000 s*	256.4 s	**1421 s**	1490.3 s
	Prod.	3.32*	3.53*	2.51*	**3.3***	3.6*

Table 3.10: Third heuristic comparison, *nbr_unit* $= 4$

In Table 3.10, we can see that for the small instance, the heuristic method 1 gives a good feasible solution quicker. For the large instance, the heuristic method 4 gives a good solution with a smaller duality gap quicker than the other heuristic methods. We can observe in Table 3.10 that the heuristic methods 4 and 5 give good solutions for the small instance but are quite slow. For the large instance, the heuristic method 5 finds a good solution in terms of productivity, but a solution with a large duality gap in terms of the linear objective. For heuristic method 5, in order to obtain a reduced duality gap (and a better feasible solution) with respect to the linear objective, we change two parameters. The maximum time for solving each optimization subproblem is now set to 100 sec. in order to obtain quicker a feasible solution and we introduce more flexibility in the resolution of each subproblem by using the parameter $k1 = 5$ in order to obtain a better feasible solution.

The results obtained for the modified heuristic method 5 are presented in Table 3.11.

In Table 3.11, we can observe that for the small instance, the solution obtained with the heuristic method 5 and the new parameters is the same as the one obtained with the initial parameters but the CPU time is reduced. For the large instance, we get quicker a very good solution with a small remaining duality gap. With these new parameters, the heuristic method 5 provides a very good solution quicker than the other heuristic methods.

To summarize, we can conclude that for the small instances in Tables 3.8-3.10, the heuristic method 4 does not (except once) obtain a better feasible solution than the heuristics 1 and 2, with a smaller duality gap. However for the large instances in the same Tables, the heuristic methods 1 and 2 do not provide good solutions quickly anymore. For large instances, the use of the heuristic method 4 or 5 is quite interesting because we obtain on average better feasible solutions in less computing time. Moreover, heuristic method 5 can be more interesting

4 units		R&F/LB(F2*)
Small Instance T=18	# Constr.	919
	Nodes	5584
	D. Gap (%)	16.1
	Time	551 s
	Prod.	3.32*
Large Instance T=26	# Constr.	1319
	Nodes	4388
	D. Gap (%)	16.5
	Time	823 s
	Prod.	3.28*

Table 3.11: Modified heuristic method 5, $nbr\_unit = 4$

than heuristic method 4, because, on average, this heuristic provides good feasible solutions quicker and with a small duality gap. However, for one instance, the results obtained by the heuristic method 5 are not satisfactory. For this case, we have seen that the parameters of the algorithm can be adapted in order to provide better feasible solutions quicker.

3.2.4 Comparison with the formulation of Schilling and Pantelides [44]

We consider the multiple batch task problem described in Section 3.2.1 that is classified as follows : nB/1C/CAP/Res. We want to compare the performances of the basic strengthened formulation (F2) and of the formulation proposed by Schilling and Pantelides (SP) in [44] adapted to our case.

The only difference between their formulations and ours is that we do not allow to perform several batches of a task in parallel with the same starting and ending time slots.

For the multiple batch task case, this has no impact because a same batch task is never performed in parallel on two reactors at the same starting and ending times.

In general, our formulation of Schilling and Pantelides would require a zero duration time slot to start and finish several batches of a task at the same time. So, the only consequence would be an increase in the number of time slots. This is illustrated in Figure 3.4.

The formulation (F2) is based on the constraints : (2.59)-(2.62), (2.63), (2.65), (2.70)-(2.73), (2.77)-(2.84) and (2.86). The formulation (SP) is based on the constraints (2.57)-(2.58), (2.61)-(2.62), (2.70)-(2.73) and (2.76)-(2.84). The difference between the formulation (F1) and the formulation (SP) is that the formulation (SP) does not limit the number of batch tasks starting or finishing at a time slot. The formulation (SP) is composed of all the constraints of the formulation (F1) except constraints (2.59)-(2.60).

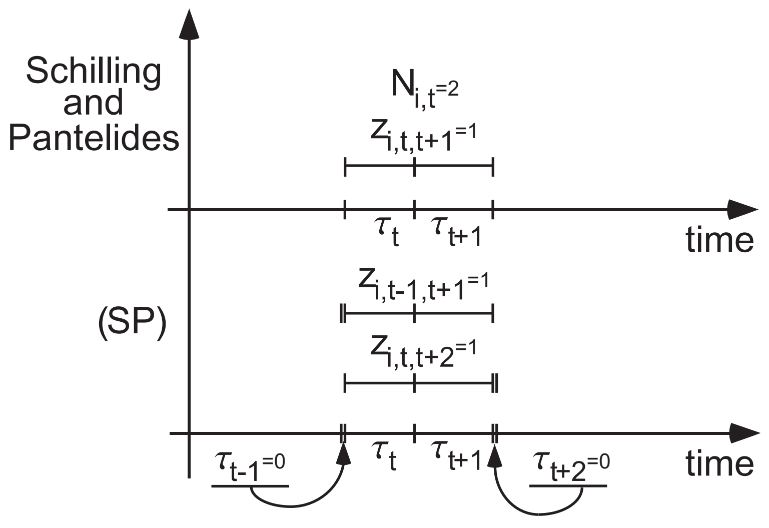

Figure 3.4: Difference between the formulation of Schilling and Pantelides [44] and their formulation adapted to our case (SP), $N_{i,t} = 2$ means that two batches of task i are started at time slot t.

To represent a solution with several events occurring at the same time, (F2) requires zero duration time slots, and therefore requires a larger number T of time slots than formulation (SP). This is illustrated in Figure 3.5.

The initial objective function is the following ($\mu^1 = \underline{\rho} = 1$) :

$$\max \sum_{t=1}^{T} q_t - \sum_{t=1}^{T} \tau_t.$$

For clarity we recall these formulations below.

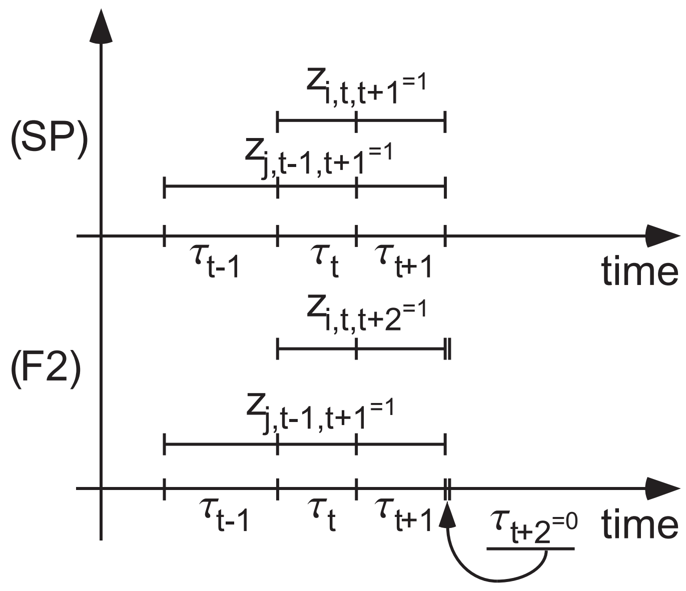

Figure 3.5: Difference between the formulation of Schilling and Pantelides adapted to our case (SP) and the formulation (F2).

F2 :

nB/1C/CAP/Res//STR1

$$\max \sum_{t=1}^{T} q_t - \sum_{t=1}^{T} \tau_t$$

$$\sum_{i\in BT} \sum_{k=t|t\neq l}^{l} p_i z_{i,t,\Omega(k)} + \sum_{i\in BT} p_i z_{i,\Omega(l),\Omega(l)} - \sum_{k=t}^{l} \tau_{\Omega(k)} \leq 0$$

$$\forall t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\} \tag{2.63}$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} \leq \sum_{i\in BT} \sum_{k=l-d+1}^{l} p_i z_{i,\Omega(k),\Omega(l)}$$

$$+ p^{\max}\Big((l-t+1) - \sum_{i\in BT} \sum_{k=l-d+1}^{l} \min\{l-t+1, l-k+1\} z_{i,\Omega(k),\Omega(l)}\Big)$$

$$+ \sum_{i\in BT} (p_i - p^{\max}) \sum_{t'=t|l=t+d-1}^{l-1} z_{i,t,\Omega(t')}$$

$$\forall t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\} \tag{2.65}$$

nB/1C/CAP/Res//STR1 (Contd)

$$\sum_{i\in BT}\sum_{l=t}^{t+d-1} z_{i,t,\Omega(l)} \le 1 \qquad \forall t\in\{1,\dots,T\} \tag{2.59}$$

$$\sum_{i\in BT}\sum_{t=l-d+1}^{l} z_{i,\Omega(t),l} \le 1 \qquad \forall l\in\{1,\dots,T\} \tag{2.60}$$

$$\sum_{i\in BT}\sum_{t=1}^{T}\sum_{\substack{l=t|t\le t'\le l\\ \text{or } t\le t'+T\le l}}^{t+d-1} z_{i,t,l} \le nbr\_unit \qquad \forall t'\in\{1,\dots,T\} \tag{2.61}$$

$$\sum_{t=1}^{T}\sum_{l=t}^{t+d-1} z_{i,t,\Omega(l)} \le \left\lfloor \frac{T}{|BT|}\right\rfloor \forall i\in BT \tag{2.71}$$

$\displaystyle\sum_{l=t}^{t+d-1} z_{i+1,t,\Omega(l)} = \sum_{l=t-d}^{t-1} z_{i,\Omega(l),\Omega(t-1)} \qquad \forall t\in\{1,\dots,T\}, \forall i\in BT:$ task $i+1$ has to be performed after task i, without any waiting. (2.72)

$$\sum_{i\in BT}\frac{p_i}{nbr\_unit}\sum_{t=1}^{T}\sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')} - \sum_{t=1}^{T}\tau_t \le 0 \tag{2.73}$$

$$q_t \le \overline{\rho}\tau_t - \sum_{i\in BT}\max(p_i\overline{\rho} - (Rmax_{\tilde r} - Rmin_{\tilde r}); 0) z_{i,t,t}$$
$$\text{for } \tilde r\in R : \lambda_{\tilde r} = -1, \forall t\in\{1,\dots,T\} \tag{2.86}$$

$$q_t \ge \underline{\rho}\tau_t \qquad \forall t : 1\le t\le T \tag{2.77}$$

$$w_{r,t} = wf_{r,\Omega(t-1)} + \sum_{i\in BT}\bar\mu_{i,r}\sum_{t'=t-d}^{t-1} z_{i,\Omega(t'),\Omega(t-1)} - \sum_{i\in BT}\mu_{i,r}\sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')}$$
$$\forall r,t : 1\le r\le R, 1\le t\le T \tag{2.78}$$

$$wf_{r,t} = w_{r,t} + \lambda_r q_t \qquad \forall r,t : r\in R_m, 1\le t\le T \tag{2.79}$$

$$wf_{r,t} = w_{r,t}\forall t,\ \forall r\in R_e\cup R_u \tag{2.80}$$

$$Rmin_r \le w_{r,t} \le Rmax_r \qquad \forall r,t : 1\le r\le R, 1\le t\le T \tag{2.81}$$

$$Rmin_r \le wf_{r,t} \le Rmax_r \qquad \forall r,t : r\in R_m, 1\le t\le T \tag{2.82}$$

$$wf_{r,t} + \sum_{i\in BT}\bar\mu_{i,r}\sum_{t'=t-d+1}^{t} z_{i,\Omega(t'),t} \le Rmax_r$$
$$\forall r,t : 1\le r\le R, 1\le t\le T \tag{2.83}$$

$$Rmin_{\tilde r} \le \sum_{i\in BT}\sum_{t_1=1}^{T}\sum_{\substack{t_2=t_1|t_1\le t\le t_2\\ \text{or } t_1\le t+T\le t_2}}^{t_1+d-1} \bar\mu_{i,\tilde r} z_{i,\Omega(t_1),\Omega(t_2)} \le Rmax_{\tilde r}$$
$$\forall \tilde r\in R : \bar\mu_{i,\tilde r} = \mu_{i,\tilde r}, \forall i\in BT \tag{2.84}$$

nB/1C/CAP/Res//STR1 (Contd)

$$\sum_{i \in BT} \sum_{t=1}^{T} \sum_{\substack{l=t \mid t \le t' \le l \\ \text{or } t \le t'+T \le l}}^{t+d-1} z_{i,t,\Omega(l)} + \sum_{i=1}^{|BT|} w_{r_i,t'} = nbr\_unit \ \forall t' \in \{1,\ldots,T\} \quad (2.70)$$

$$z_{i,t,\Omega(l)} \in \{0,1\} \qquad \forall i \in BT, t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\}$$

$$\tau_t \ge 0 \qquad \forall t : 1 \le t \le T \quad (2.62)$$

SP :

nB/1C/CAP/Res

$$\max \sum_{t=1}^{T} q_t - \sum_{t=1}^{T} \tau_t$$

$$p_i z_{i,t,\Omega(l)} - \sum_{k=t}^{l} \tau_{\Omega(k)} \le 0$$

$$\forall i \in BT, t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\} \quad (2.57)$$

$$\sum_{k=t}^{l} \tau_{\Omega(k)} \le p_i z_{i,t,\Omega(l)} + p^{\max}(l-t+1)(1-z_{i,t,\Omega(l)})$$

$$\forall i \in BT, t \in \{1,\ldots,T\}, l \in \{t,\ldots,t+d-1\} \quad (2.58)$$

$$\sum_{i \in BT} \sum_{t=1}^{T} \sum_{\substack{l=t \mid t \le t' \le l \\ \text{or } t \le t'+T \le l}}^{t+d-1} z_{i,t,l} \le nbr\_unit \qquad \forall t' \in \{1,\ldots,T\} \quad (2.61)$$

$$\sum_{t=1}^{T} \sum_{l=t}^{t+d-1} z_{i,t,\Omega(l)} \le \left\lfloor \frac{T}{|BT|} \right\rfloor \forall i \in BT \quad (2.71)$$

$$\sum_{l=t}^{t+d-1} z_{i+1,t,\Omega(l)} = \sum_{l=t-d}^{t-1} z_{i,\Omega(l),\Omega(t-1)} \qquad \forall t \in \{1,\ldots,T\}, \forall i \in BT: \text{ task}$$

$i+1$ has to be performed after task i, without any waiting. (2.72)

$$\sum_{i \in BT} \frac{p_i}{nbr\_unit} \sum_{t=1}^{T} \sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')} - \sum_{t=1}^{T} \tau_t \le 0 \quad (2.73)$$

$$q_t \le \overline{\rho} \tau_t \qquad \forall t : 1 \le t \le T \quad (2.76)$$

$$q_t \ge \underline{\rho} \tau_t \qquad \forall t : 1 \le t \le T \quad (2.77)$$

$$w_{r,t} = wf_{r,\Omega(t-1)} + \sum_{i \in BT} \bar{\mu}_{i,r} \sum_{t'=t-d}^{t-1} z_{i,\Omega(t'),\Omega(t-1)} - \sum_{i \in BT} \mu_{i,r} \sum_{t'=t}^{t+d-1} z_{i,t,\Omega(t')}$$

$$\forall r, t : 1 \le r \le R, 1 \le t \le T \quad (2.78)$$

$$wf_{r,t} = w_{r,t} + \lambda_r q_t \qquad \forall r, t : r \in R_m, 1 \le t \le T \quad (2.79)$$

nB/1C/CAP/Res (Contd)

$$wf_{r,t} = w_{r,t} \forall t,\ \forall r \in R_e \cup R_u \tag{2.80}$$

$$Rmin_r \leq w_{r,t} \leq Rmax_r \qquad \forall r, t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.81}$$

$$Rmin_r \leq wf_{r,t} \leq Rmax_r \qquad \forall r, t : r \in R_m, 1 \leq t \leq T \tag{2.82}$$

$$wf_{r,t} + \sum_{i \in BT} \bar{\mu}_{i,r} \sum_{t'=t-d+1}^{t} z_{i,\Omega(t'),t} \leq Rmax_r$$
$$\forall r, t : 1 \leq r \leq R, 1 \leq t \leq T \tag{2.83}$$

$$Rmin_{\tilde{r}} \leq \sum_{i \in BT} \sum_{t_1=1}^{T} \sum_{\substack{t_2=t_1 | t_1 \leq t \leq t_2 \\ \text{or } t_1 \leq t+T \leq t_2}}^{t_1+d-1} \bar{\mu}_{i,\tilde{r}} z_{i,\Omega(t_1),\Omega(t_2)} \leq Rmax_{\tilde{r}}$$
$$\forall \tilde{r} \in R : \bar{\mu}_{i,\tilde{r}} = \mu_{i,\tilde{r}}, \forall i \in BT \tag{2.84}$$

$$\sum_{i \in BT} \sum_{t=1}^{T} \sum_{\substack{l=t | t \leq t' \leq l \\ \text{or } t \leq t'+T \leq l}}^{t+d-1} z_{i,t,\Omega(l)} + \sum_{i=1}^{|BT|} w_{r_i,t'} = nbr\_unit\ \forall t' \in \{1, \ldots, T\} \tag{2.70}$$

$$z_{i,t,\Omega(l)} \in \{0,1\} \forall i \in BT, t \in \{1, \ldots, T\}, l \in \{t, \ldots, t+d-1\},$$
$$\tau_t \geq 0 \forall t \in \{1, \ldots, T\} \tag{2.62}$$

First Comparison (SP) versus (F2)

We consider first an instance of the formulation (SP). The characteristics of this instance are the following : $T = 9$, $d = 9$ and $nbr\_unit = 2$. There are 72 integer variables, 648 binary variables and 55 continuous variables. The optimal solution obtained is represented in Figures 3.6-3.7.

The same optimal solution can be obtained by using the formulation (F2). In Figure 3.6, we can observe that no task is starting or finishing at the same time, only eight time slot durations are non zero and the batch tasks last for at most four time slots. Therefore, the optimal solution obtained by using the formulation (SP) above is feasible for the formulation (F2) if the parameters used are $T = 8$, $d = 4$ and $nbr\_unit = 2$. The characteristics of the corresponding instance are the following : 64 integer variables, 256 binary variables and 49 continuous variables.

However, it is clear for the same reason given above for the parameter used in formulation (F2) that the optimal solution presented in Figures 3.6-3.7 can also be obtained by using the formulation (SP) with the parameters $T = 8$, $d = 4$ and $nbr\_unit = 2$. This will lead to a reduced formulation (SP*) with 64 integer variables, 256 binary variables and 49 continuous variables.

In Table 3.12, we present the results obtained for this first comparison between the three formulations.

We observe first that the number of constraints in the formulation (F2) is

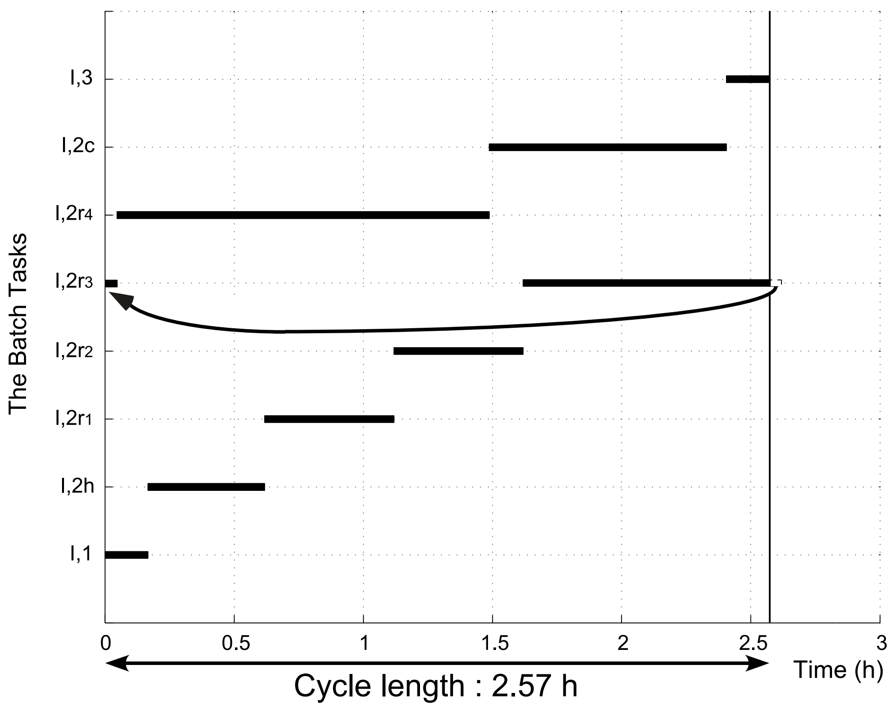

Figure 3.6: The optimal scheduling of the batch tasks using formulation (SP)

First comparison (SP) versus (F2)			
nbr unit=2, p=1	SP	SP*	F2
Binary var.	648	256	256
Cont. var.	55	49	49
Constraints	1496	691	275
LP relax at root node	11.63	10.55	5.42
LP relax at root node + MIP solver cuts	9.02	6.56	5.42
OPT sol of the MIP	5.42	5.42	5.42
Nodes	516	387	9
Time	10 s	4 s	$<$ 1 s
Productivity	3.11	3.11	3.11
The cycle length	2.57 h	2.57 h	2.57 h

Table 3.12: First comparison between the formulations (SP), (SP*) and (F2).

smaller than in the formulations (SP) and (SP*). The reason is that in the formulation (F2) the timing constraints have been aggregated over all batch tasks. We can also observe, for the first iteration of the linearization of the objective function (p=1), that the formulation (F2) is tighter than the other two formulations since the difference between the optimal solution and the LP relaxation at the root node

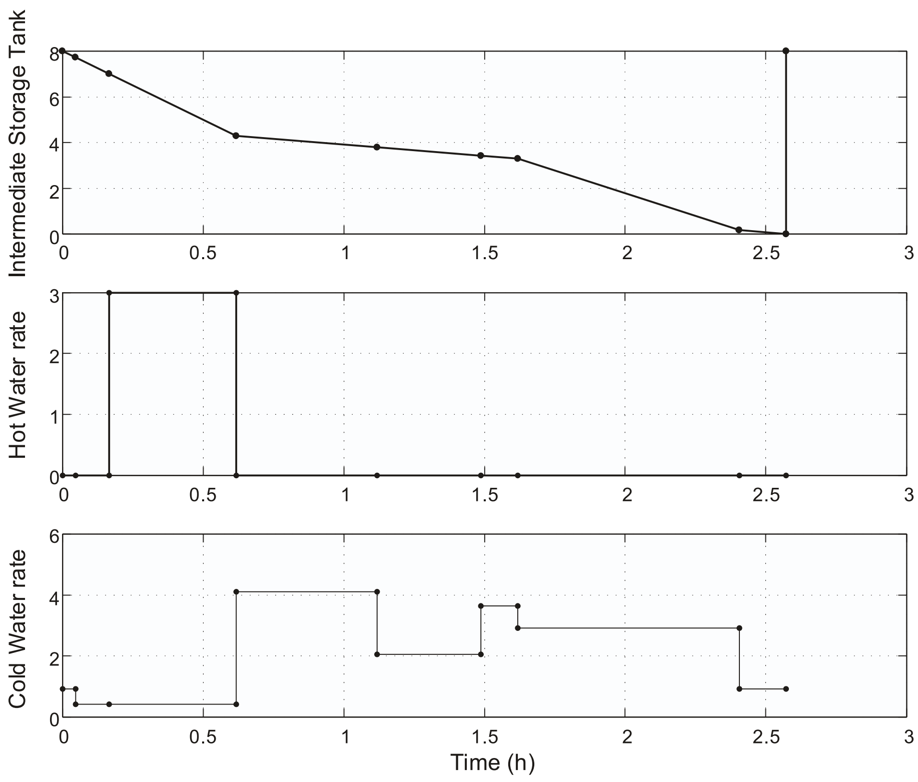

Figure 3.7: Optimal evolution of the resources over the scheduling cycle using the formulation (SP)

is zero. This means that the duality gap is 0. However, the formulations (SP) and (SP*) have large duality gap. This is mainly due to big M constraints in the model formulation that lead to weak model formulation with large duality gap. The consequences of large duality gap for formulations (SP) and (SP*) are first the increase of the number of branch-and-bound nodes needed in order to solve the first iteration to optimality and the increase of CPU solution time. As shown in Table 3.12, the number of branch-and-bound nodes and the CPU solution time for solving the first linearization iteration to optimality are smaller by using the formulation (F2).

The formulation (F2) is tighter and has less constraints (more compact) than the formulations (SP) and (SP*).

We have checked with the three formulations (SP), (SP*) and (F2) that the productivity 3.11 obtained at the first iteration is optimal. This is proved in less than 1 second with the three formulations by observing that the LP relaxation objective value is zero at iteration 2.

Second Comparison (F2) versus (SP)

We consider another instance of the formulation (F2). The characteristics of this instance are the following : $T = 17$, $d = 4$, $nbr\_unit = 2$. There are 136 integer variables, 544 binary variables and 103 continuous variables. The optimal solution obtained is represented in Figure 3.3.

The same optimal solution can be obtained by using the formulation (SP). In Figure 3.3, we observe that only fifteen time slots have non zero durations and that the batch tasks last for at most four time slots (Two time slots have zero duration because there are twice two events occurring simultaneously).

Therefore, the optimal solution obtained by using the formulation (F2) above is feasible for the formulation (SP) if the parameters used are $T = 15$, $d = 4$ and $nbr\_unit = 2$. The characteristics of the corresponding instance are the following : 120 integer variables, 480 binary variables and 91 continuous variables.

In Table 3.13, we present the results obtained for the second comparison between these two formulations.

Second comparison (F2) versus (SP)		
p=1	F2	SP
Binary var.	544	480
Cont. var.	103	91
Constraints	563	1286
LP relax at root node	10.85	16.28
LP relax at root node + MIP solver cuts	10.85	15.82
OPT sol of the MIP	10.85	10.85
Nodes	334	5707
Time	12 s	90 s
Productivity	3.11	3.11
The cycle length	5.14 h	5.14 h

Table 3.13: Second comparison between the formulations (F2) and (SP).

In this case, we have seen that the number of time slots for the formulation (F2) is larger than one for (SP). This implies that the number of integer, binary and continuous variables increases when we use the formulation (F2) in comparison with the formulation (SP). However, the number of constraints is reduced when using the formulation (F2) since the timing constraints are aggregated over all the batch tasks.

In Table 3.13, we can observe that the number of branch-and-bound nodes and the CPU solution time for solving the first linearization iteration to optimality is reduced when we use the formulation (F2). The formulation (F2) has a smaller duality gap than the formulation (SP) and is therefore tighter than the formulation (SP). Although formulation (F2) consists in a larger number of decision variables, we have shown that its tightness leads to a more efficient resolution of the problem instance.

Again, we have checked with both formulations (SP) and (F2) that the productivity 3.11 obtained at the first iteration is optimal. This is proved in less than 1 second with the two formulations by observing that the LP relaxation objective value is zero at iteration 2.

To conclude, the formulation (F2) is a good compromise between size of the model formulation and tightness. These are the two central questions for solving efficiently optimization problems. By using the formulation (F2), we have shown that we can solve quicker such a type of scheduling problem and with less branch-and-bound nodes than when using the formulation (SP).

3.2.5 The choice of the values of T and d

There is no way to decide the best value of T and d for a given problem. By increasing T and or d, the number of decision variables increases and the number of constraints as well. This leads to a larger model formulation and therefore the CPU solution and the number of Branch-and-Bound nodes needed to solve the problem instances usually increase as well. In general, apart from a complete enumeration over all the values of T and d, there is no guarantee that the solution obtained for a given problem is globally optimal. Therefore, we consider that T and d are part of the problem formulation.

In order to illustrate this statement, we consider the multiple batch task problem described in Section 3.2.1 (nB/1C/CAP/Res) where we set the upper limits on the level of product in the storage tank, on the rate of hot water and cold water to $24ru$, $3[ru/h]$ and $6.4[ru/h]$, respectively.

We consider three instances of the formulation (F2). The characteristics of these instances are presented in Table 3.14 and $nbr\_unit = 3$.

3 reactors	T=8 d=8	T=9 d=9	T=10 d=10
# Constr.	339	397	459
Bin. var.	512	648	800
Int. var.	64	72	80
Cont. var.	49	55	61
Nodes	634	4722	19876
CPU sol time	16 s	83 s	288.3 s
Prod. [ru/h]	3.11	4	4

Table 3.14: The characteristics and the solutions of the three instances

Table 3.14 reports on the solution of this multiple batch task case using the formulations (F2) and a standard MIP solver. For the three instances tested, the maximal productivity is obtained by solving two iterations of the linearization procedure using objective function (3.2).

For the first instance with $T = 8$ and $d = 8$, the maximal productivity obtained is 3.11 [ru/h]. We can observe in Table 3.14 that by increasing T and d to 9, we get a better productivity of $4[ru/h]$. Finally, the maximal productivity obtained for the last instance with $T = 10$ and $d = 10$ is also $4[ru/h]$. The solution of the instance with $T = 9$ and $d = 9$ is obtained more quickly and with less nodes, and gives the best productivity.

However, as shown in Table 3.15, it is possible for this problem to obtain a better productivity than $4[ru/h]$ by using the following parameters : $T = 30$ and $d = 10$. Table 3.15 reports on the characteristics and the solution of this instance.

3 reactors	T=30 d=10
# Constr.	1339
Bin. var.	2400
Int. var.	240
Cont. var.	181
Nodes	194
CPU sol time	85.1 s
Prod. [ru/h]	4.66

Table 3.15: Characteristics and solution of the instance with $T = 30$ and $d = 10$ using the formulation (F2)

In Table 3.15, we have shown that by taking $T = 30$ and $d = 10$, we have improved the productivity of this problem to $4.66[ru/h]$. This improvement is significant and we can also observe that the CPU solution time remains reasonable in comparison with the ones obtained for the three other instances in Table 3.14.

In order to get this better solution, we have simply chosen a different value for T and d. However, there is no way to decide for a general problem how to choose values for T and d.

In this case, we can go one step further and prove that the solution obtained by the instance with $T = 30$ and $d = 10$ is globally optimal for the problem considered. The maximal productivity obtained is 4.66 $[ru/h]$ and is equal to the theoretical maximal productivity of the batch tasks, i.e. the maximal possible quantity produced by the sequence of batch tasks per hour, which in this case can be expressed as follows : $\frac{nbr\_unit * BS}{\sum_{i \in BT} p_i}$. The reason is that the sequence of batch tasks is performed in 5.14 h ($\sum_{i \in BT} p_i = 5.14\ h$) and produces at the end a batch of 8 ru of product ($BS = 8\ ru$). Since we have 3 reactors ($nbr\_unit = 3$) and since the sequence of batch tasks can be performed in parallel on the 3 reactors, the maximal productivity of the sequence of batch tasks on the 3 reactors is $(3 * 8)/5.14$=4.66 $[ru/h]$.

In general, apart from a complete enumeration over all the values of T and d, there is no guarantee that the solution obtained for a given problem is globally optimal.

In the next section, we try to solve a larger industrial scheduling instance and we compare the efficiency of heuristic method 5 with truncated branch and bound corresponding to heuristic methods 1 and 2.

3.3 Industrial case

In this section, we describe a basic industrial scheduling problem where the data were invented. We use some of the heuristic methods defined previously and we compare the quality of the resolution for one instance. Then, we show how it is possible to bound the productivity improvement between two iterations when we use linear objective functions. Finally, we illustrate a typical optimal solution for the industrial case.

3.3.1 Problem description

There are three polymerization lines (denoted by I-II-III) with three, three and one units respectively. The volume of the reactors for the first two lines ($V1$) is 27 ru. The volume of the reactor of line three ($V2$) is 140 ru. The polymerization task performed in the reactors is decomposed into the same number of batch tasks as in Section 3.2.1. Moreover, the same resources as for the test case are shared among the processes of a same line, but not shared between lines. The maximum rate of hot water for each line is 3 [ru/h]. The maximum rate of cold water is 7.4 [ru/h] for line 1, 8 [ru/h] for line 2 and 3.7 [ru/h] for line 3.

After each reactor, there is one tank for each line where the product is discharged. The capacity of the tanks are 35 ru, 35 ru and 140 ru, respectively. The discharge of these tanks into the common buffer to all lines is modeled as a continuous task. We assume that the valve is processing like a continuous task and that the rate of material processed by the valves of the two first lines is in the interval $[0ru/h, 15ru/h]$ and by the valves of the third line in the interval $[0ru/h, 10ru/h]$. The valves of the three lines do not have to process material all the time. The capacity of the common buffer is 40 ru and the continuous task after the buffer is a stripping task that cannot be stopped. For this task, the rate of material processed is in the interval $\rho \in [10ru/h, 55ru/h]$.

The initial objective function is the following ($\mu^1 = 10$) :

$$\max \sum_{t=1}^{T} q_t - 10 \sum_{t=1}^{T} \tau_t.$$

This process is represented in Figure 3.8.

3.3.2 Comparison of Heuristics

We first model the problem with the initial formulation (F1) and with the strengthened formulation (F2) that were proposed in Section 3.2.3 and we try to solve it up to optimality. We decide to take 20 time slots (i.e., $T = 20$) and a batch task can last for 10 time slots (i.e, $d = 10$). The problem has 10973 constraints for the initial formulation and 1833 for the strengthened one. For both formulations, we have 4800 binary variables, 480 integer variables, and 381 continuous variables. We stopped the resolution of the problem modeled by the initial formulation at the first linearization iteration after 20000 sec. and 21500 nodes, we got a remaining

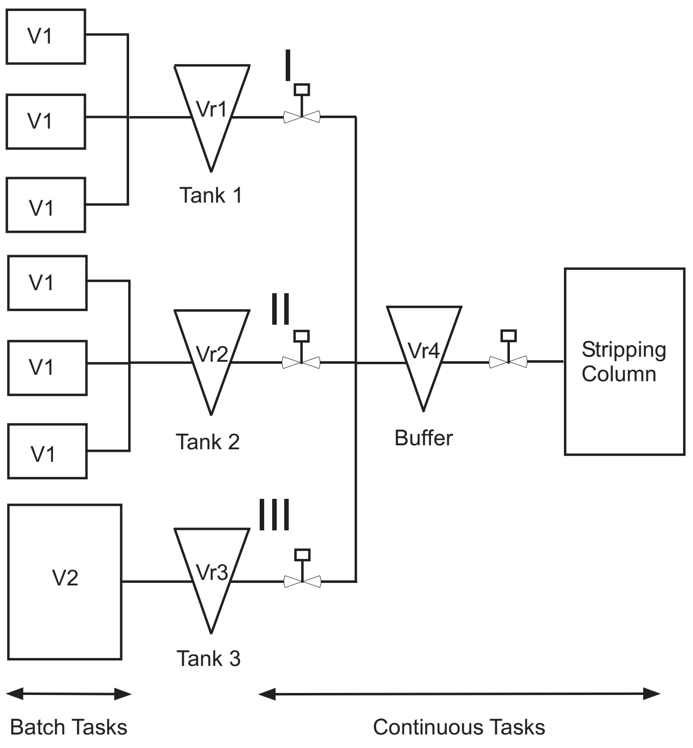

Figure 3.8: The industrial case

duality gap of 31.3%. The productivity corresponding to the best feasible solution is 20.6 $[ru/h]$. This solution method corresponds to heuristic method 1.
Also, we stopped the resolution of the problem modeled by the strengthened formulation at the first linearization iteration after 20000 sec. and 57600 nodes, we got a remaining duality gap of 27.2%. The productivity corresponding to the best feasible solution is 22 $[ru/h]$. This solution method corresponds to heuristic method 2.

For the same problem, we used the heuristic method 5 defined previously with the parameter $k1 = 8$ and the maximum time for solving each optimization subproblem is now set to 200 sec. The feasible solution obtained has a remaining duality gap of 26.9%. The corresponding number of nodes is 3126 and the CPU solution time is 1232 *sec.*. The productivity corresponding to this feasible solution is 22.12 $[ru/h]$.

After 20000 sec. of CPU time, the two methods proposed, based on the initial (F1) and the strengthened (F2) formulation, were not able to provide a better feasible solution than the one given by the heuristic method 5 in 1232 sec. of CPU time. Therefore, heuristic methods such as heuristic 5 can be interesting for large instances of the linearized objective problem, both in terms of solution quality and running time.

3.3.3 Bounds on the maximal productivity

However, the final objective is to maximize productivity. As explained in section 3.2.1, the optimal productivity is obtained by solving a sequence of mixed integer linear programs where the value of μ is updated at each iteration.

Therefore, in order to obtain a good productivity for this basic industrial case, if the feasible solution obtained at the first iteration has a productivity significantly higher than the one fixed by default for iteration 1 (μ^1), we can solve a second iteration where μ is updated. We will obtain a feasible solution with a productivity at least as good as the one obtained at the first iteration. We can continue this procedure until no significant improvement of the productivity is achieved.

In this case, $\mu^1 = \underline{\rho} = 10$ and the heuristic 5 gives, for the first iteration, a feasible solution with a duality gap of 26.9% (for the linearized objective) and a productivity of $22.12[ru/h]$.

Actually, we cannot prove that this solution is optimal for the problem with the nonlinear objective function (3.1) maximizing productivity because the solution of the linear relaxation of the second linearized problem with objective function (3.2) and with $\mu^2 = 22.12$ is 14.18, and is not 0.

To measure the quality of the solution obtained for the problem with the linearized objective function in term of productivity (i.e. with respect to the nonlinear objective), we show that we can bound the maximal improvement of productivity that can be obtained by the optimal solution at each iteration, and that this bound on the maximal improvement per iteration is monotonically non increasing over the linearization iterations.

The objective value for the optimal solution of the problem with the linear objective function at iteration p $(q^{*,p}, \tau^{*,p})$ is bounded by :

$$\sum_{j \in OUT} \sum_{t=1}^{T} q_{j,t}^{*,p} - \mu^p \sum_{t=1}^{T} \tau_t^{*,p} \leq UB^p$$

where UB^p is the LP relaxation bound obtained at the root node of the Branch-and-Bound algorithm. We show now how an upper bound on the maximal productivity improvement can be obtained for this iteration.

We divide every term of the previous inequality by the unknown $\sum_{t=1}^{T} \tau_t^{*,p} (> 0)$, that is by the optimal cycle duration at iteration p for the problem with the linear objective function, and we get :

$$\frac{\sum_{j \in OUT} \sum_{t=1}^{T} q_{j,t}^{*,p}}{\sum_{t=1}^{T} \tau_t^{*,p}} - \mu^p = \mu^{p+1} - \mu^p \leq \frac{UB^p}{\sum_{t=1}^{T} \tau_t^{*,p}}$$

This defines an upper bound on the productivity improvement $(\mu^{p+1} - \mu^p)$. As

$\sum_{t=1}^{T} \tau_t^{*,p}$ is unknown, we look for a known lower bound on $\sum_{t=1}^{T} \tau_t^{*,p}$ in order to obtain a known upper bound on $\mu^{p+1} - \mu^p$.

First note that $\sum_{t=1}^{T} \tau_t^{*,p} \geq p_{i_1}$ because we know that in order to break the symmetry, we have imposed that at least one batch task i_1 of the line I' has to be performed during the cycle. Moreover, in our case, we have some precedence constraints between batch tasks and therefore we know that if a batch task i_1 starts on line I', at least a set of batch tasks on line I' ($BT_{I'}$) has to be processed during the cycle. The minimum time required, in order to complete all the batch tasks in the set $BT_{I'}$, is obtained when all the possible reactors of line I' are performing the tasks in $BT_{I'}$ in parallel. This is why $(\sum_{t=1}^{T} \tau_t) \geq \frac{1}{nbr\_unit\_I'} \sum_{i \in BT_{I'}} p_i$ where nbr_unit_I' is the number of reactors available on line I' able to perform a batch task of the set $BT_{I'}$.

This lower bound on the cycle length can actually be improved. If a feasible solution at iteration p exists for the problem instance, it must satisfy $\sum_{t=1}^{T} \tau_t \geq LB^{CL,p}$ where

$$LB^{CL,p} = \min \sum_{t=1}^{T} \tau_t$$

$$st \quad \sum_{j \in OUT} \sum_{t=1}^{T} q_{j,t} - \mu^p \sum_{t=1}^{T} \tau_t \geq 0$$

and all the constraints of the basic strengthened formulation (F2) are satisfied

Therefore, the upper bound on the maximal improvement of productivity that can be obtained by the optimal solution at iteration p can be bounded by

$$\mu^{p+1} - \mu^p \leq \frac{UB^p}{\sum_{t=1}^{T} \tau_t^{*,p}} \leq \frac{UB^p}{LB^{CL,p}} \tag{3.3}$$

Now we prove why the upper bound on the maximal productivity improvement that can be obtained by the optimal solution at each iteration is monotonically non increasing over the iterations.

By starting with $\mu^1 = \sum_{j \in OUT} \underline{\rho}_j$, the value of μ^p is monotonically non decreasing (see Section 3.2.1).

Let $\tilde{q}$ and $\tilde{\tau}$ be the solution of the linear relaxation at iteration $p+1$ that defines the upper bound UB^{p+1} on the optimal objective value. Then, this solution can also be a relaxed solution for iteration p. $UB^{p+1} = \sum_{j \in OUT} \sum_{t=1}^{T} \tilde{q}_{j,t} - \mu^{p+1} \sum_{t=1}^{T} \tilde{\tau}_t \leq \sum_{j \in OUT} \sum_{t=1}^{T} \tilde{q}_{j,t} - \mu^p \sum_{t=1}^{T} \tilde{\tau}_t$ because $\mu^p \leq \mu^{p+1}$ and $\sum_{t=1}^{T} \tilde{\tau}_t \geq 0$. As $\tilde{q}$ and $\tilde{\tau}$ are feasible for iteration p, we have $\sum_{j \in OUT} \sum_{t=1}^{T} \tilde{q}_{j,t} - \mu^p \sum_{t=1}^{T} \tilde{\tau}_t \leq UB^p$. This proves that $UB^{p+1} \leq UB^p$.

Moreover, since the value of μ^p is monotonically non decreasing, the value of $LB^{CL,p}$ is also monotonically non decreasing since the constraint $\sum_{j \in OUT} \sum_{t=1}^{T} q_{j,t} - \mu^p \sum_{t=1}^{T} \tau_t \geq 0$ will restrict more and more the feasible solution set.

Then, by dividing UB^{p+1} by $LB^{CL,p+1}$ and UB^p by $LB^{CL,p}$, and by using (3.3), the upper bound on the maximal productivity improvement that can be obtained by the optimal solution at iteration $p+1$ is less or equal than the one at iteration p.

Therefore, if the upper bound on the maximal improvement of productivity that can be obtained by the optimal solution between two iterations is small, we know that for the next step this bound will be even smaller, and we can stop when we find that the improvement will not be significant enough.

In our industrial case, we tried to solve the optimization problem in order to determine $LB^{CL,1}$ but we could not solve it to optimality. The lower bound at the root node obtained on the minimum duration of the cycle is 1.8 h. The upper bound obtained for the first linearization iteration is 40.5 for the three heuristic methods. Therefore, the maximal improvement of productivity that can be obtained by the optimal solution for the first linearization iteration is $\frac{40.5}{1.8} = 22.5[ru/h]$. So the maximal productivity that can be obtained by the optimal solution at iteration 1 is 32.5 $[ru/h]$ $(\mu^1 + 22.5)$. Heuristic 5 produces a solution whose productivity is 22.12 ru/h $(= \mu^2)$.

The upper bound obtained for the second linearization iteration is 14.18 and the lower bound at the root node obtained on the minimum duration of the cycle at iteration 2 is $LB^{CL,2} \geq 1.8$ h. The maximal productivity improvement that we can obtain by solving the second iteration up to optimality is $\frac{14.18}{1.8} = 7.87[ru/h]$. Therefore, the maximal productivity that can be obtained by the optimal solution at iteration 2 is 30 $[ru/h]$ $(\mu^2(22.12) + 7.87)$. We observe that the maximal improvement per iteration is non-increasing.

We can also mention here the fact that during the Branch and Bound algorithm, every time a feasible solution is obtained with a larger productivity than the best current one, we could add a cut valid for the whole formulation imposing that the productivity of the next feasible solution has to be greater or equal to this larger productivity. This could give a better value of μ for the next iteration and therefore speed up the convergence of the μ's. Unfortunately, we have observed that by adding this type of constraints, the resolution of the problem with the linear objective function was slower.

3.3.4 Solution of the industrial case

To conclude, we describe the best solution obtained in terms of productivity by heuristic method 5. The schedule obtained for line I and II is given in Figure 3.9. We can observe that the three reactors of line I and the three reactors of line II are processing during the cycle. Line III is not used.

The schedule obtained for the three reactors of line I and for the three reactors of line II is represented in Figure 3.10.

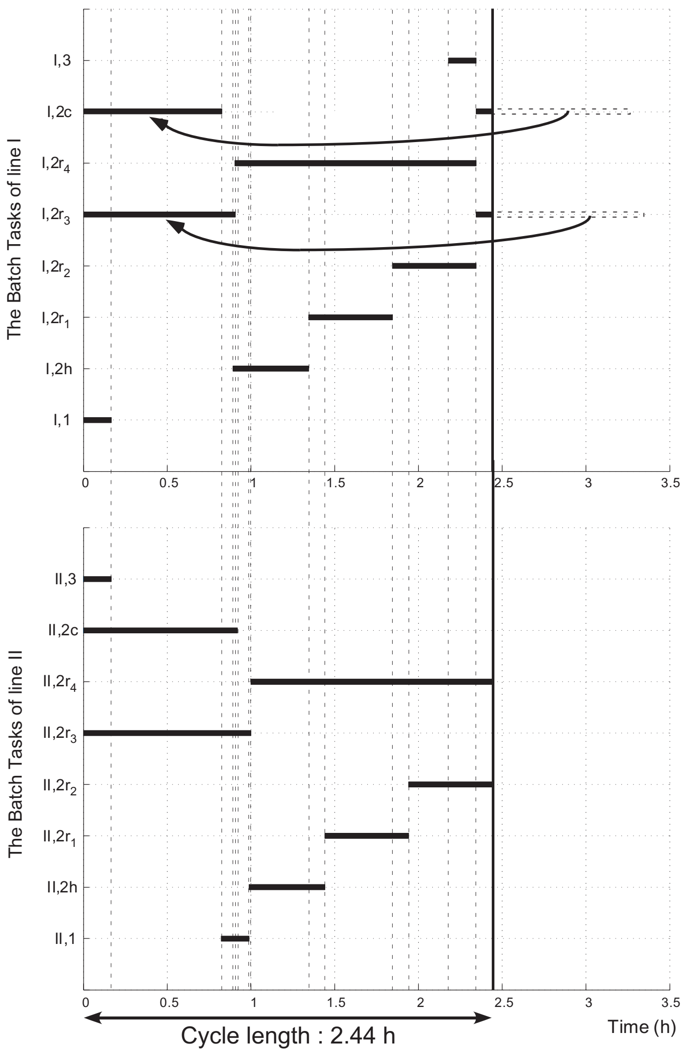

Figure 3.9: The schedule of the batch tasks of line I and of line II

For line 1 and 2, the resource levels are represented in Figures 3.11-3.12, respectively.

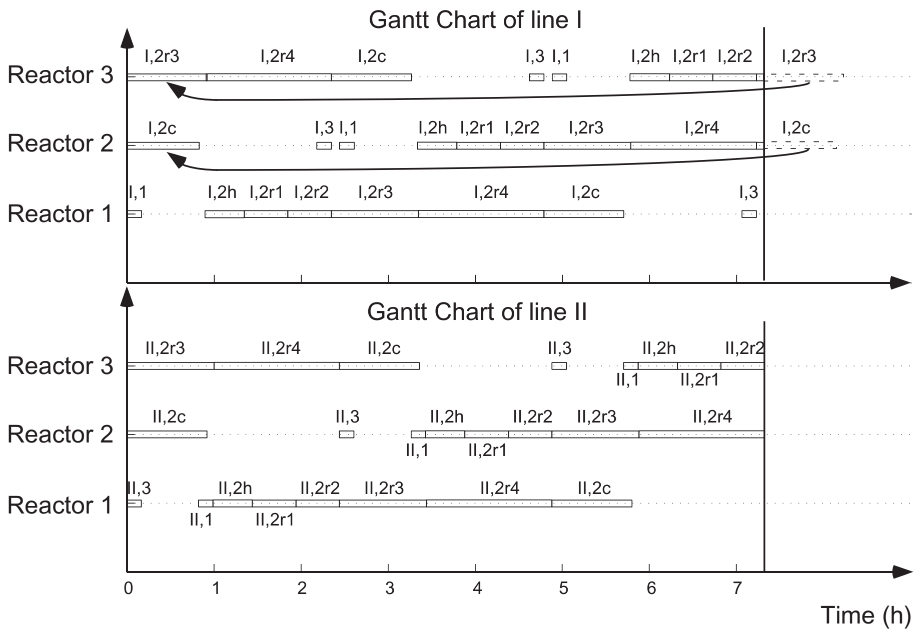

Figure 3.10: The schedule of the reactors of line I and II

The evolution of the storage tank level of the common buffer (Vr4) is represented in Figure 3.13.

3.4 Conclusion

We showed for all special cases of the general problem that the improved formulations give better results (quality and/or running times) than the initial one but the resolution of large instances remains difficult. We were still not able to solve realistic industrial cases with exact Branch-and-Bound methods. So, we investigated MIP based heuristic methods in order to obtain good feasible solutions quickly. We showed that, for some large instances, the heuristic solutions given by the exact methods (truncated Branch-and-Bound) were not better than the feasible solutions given by the MIP based heuristic methods, and the latter use less CPU solution time. Finally, we solved a basic industrial case using the initial and the strengthened formulations by truncated branch and bound, and also by a MIP based heuristic method. We got quicker and better feasible solutions by using the MIP based heuristic method, showing that such heuristic methods seem important for large instances.

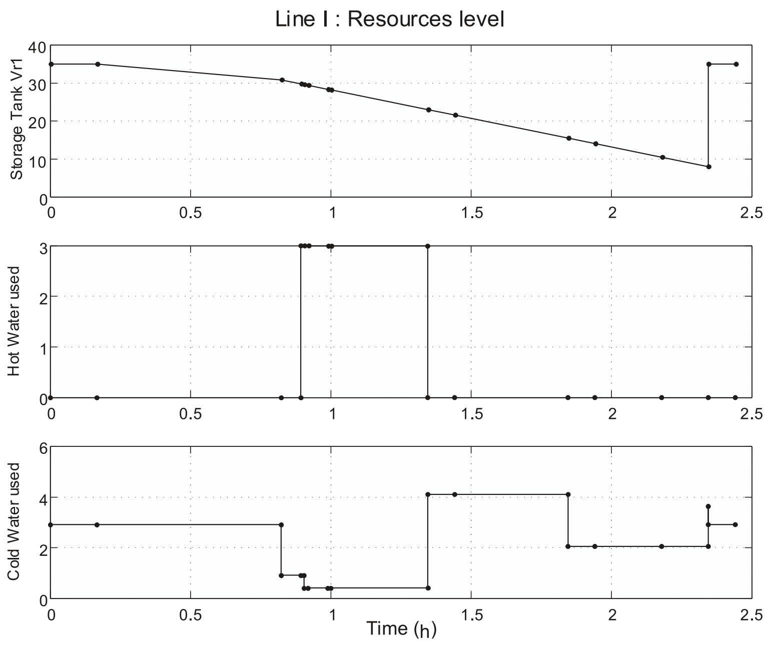

Figure 3.11: The resources for line I

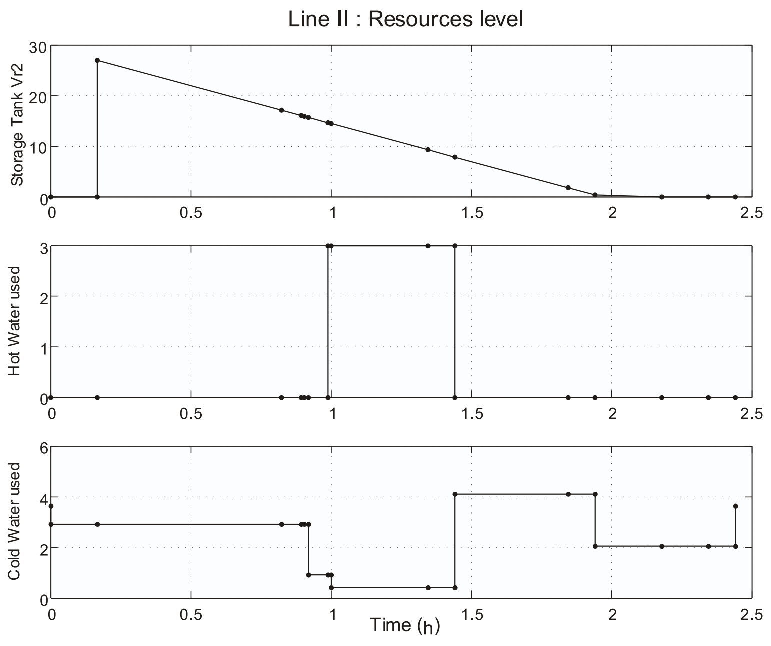

Figure 3.12: The resources for line II

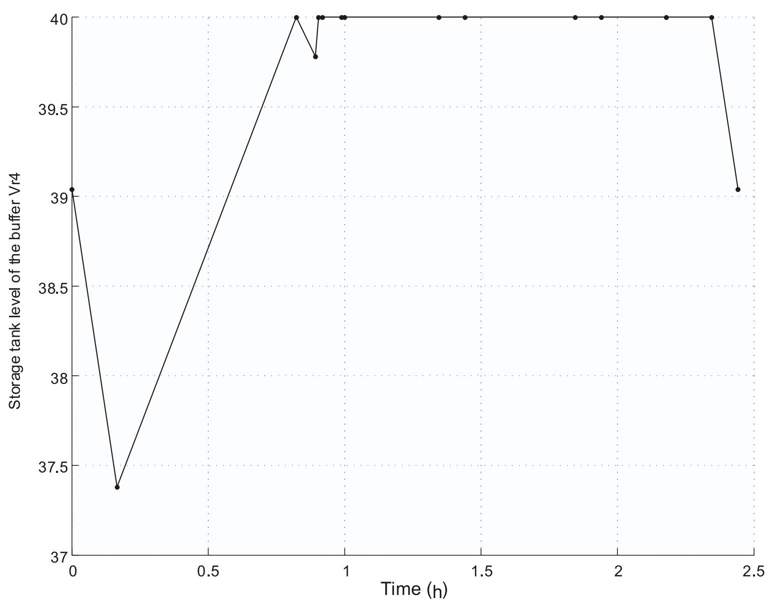

Figure 3.13: The evolution of the storage tank level of the buffer

Chapter 4

Modeling the process dynamics in the scheduling problem

In the previous Chapters, we have solved scheduling problems with static processes, i.e., where processing times of batch tasks and resource usages are constants of the model. However, the processes involved in the scheduling problems can be modeled dynamically with differential and algebraic equations (DAE) in order to obtain a more flexible model, and possibly to increase productivity. The differential equations express the dynamics of the processes but, in general, explicit solutions of differential equations are not available. Therefore, in this Chapter, various methods are proposed and investigated in order to compute numerical solutions of the differential equations within the resolution of the scheduling optimization problem.

In the literature, two approaches are proposed in order to solve problems with DAE. We can use a variational approach or an approach using a non linear programming (NLP) solver. The variational approach is based on the solution of the first order necessary conditions for optimality given by the Pontryagin's maximum principle (see Pontryagin et al. [39]). If the problem is composed of inequality constraints, it becomes often very difficult to solve the problem by using this method. The other approach that uses the NLP solver can be further subdivided into a sequential or a simultaneous approach, see Cervantes and Biegler [13] for more details. More recently, in Mishra et al. [31], the interest of simultaneous scheduling and control problems was highlighted.

In the sequential approach, the idea is to discretize the control variables, i.e. the input variables of the system, and to consider them as the variables of the optimization problem. Then, based on the optimal solution obtained for the control variables, we are able to compute, via the simulation of the DAE, the state variables.

In the simultaneous approach, various possibilities presented below were proposed in the literature. Cuthrell and Biegler in [17] compute the discretized solution of the differential equations by using polynomial approximations and orthogonal collocation. The system of equations is then solved by using a NLP solver. In Cervantes and Biegler [14], the discretization of the solution of the differential equations is based on a monomial basis representation that implies smaller rounding errors, see Bader and Ascher [3]. This leads to a large-scale NLP that is solved efficiently by using the structure of the equations. In Biegler et al. [9], an improved nonlinear programming strategy for the simultaneous approach using collocation formulations is proposed based on interior point methods. In Terrazas-Moreno et al. [47], a simultaneous solution for the optimal sequencing and optimal dynamic transition of two multi-grade polymerization reactors was presented. This simultaneous formulation leads to a Mixed-Integer Dynamic Optimization (MIDO) problem that was transformed into a Mixed-Integer Nonlinear Program by approximating the dynamics using orthogonal collocation equations. The problem was then solved using an outer-approximation method. In Nystrom et al. [34], the MIDO formulation is decomposed in a primal problem with the dynamic part (a Dynamic Optimization Problem) and a master problem with the scheduling formulation (a Mixed-Integer Linear Problem). Another idea for integrating scheduling and control problem was proposed by Prata et al. in [40], the MIDO is solved by using single and multiple shooting methods.

We can mention that collocation methods for dynamic optimization have been used in design and scheduling optimization of batch plants, see Bhatia and Biegler [8]. Finally, we can also indicate that in Charalambides et al. [15], the synthesis of processes, i.e. the design of processes, including batch reaction and distillation tasks was considered and they provide the optimal operating strategies for entire batch processes by using a simultaneous approach.

A review on dynamic optimization methods can be found in Biegler and Grossmann [10].

In this Chapter, we focus on so-called simultaneous approaches that solve the differential equations within the optimization problem. The production process considered here is composed of a sequence of consecutive batch processes, where the evolution of each batch task is modeled by differential equations, as a function of some control variables (heating,. . .), and the transitions between batch tasks occur when specific condition on the system state are met. This is a hybrid process described by a network composed of states and transitions. The associated scheduling problem consists in selecting the control variables in order to minimize the time required to produce the sequence of tasks. As in Avraam et al. [2], we discretize the state and the control variables and we use first a collocation method in order to solve numerically the differential equations within the scheduling optimization problem. The problem to solve is a mixed integer non linear program (MINLP). In order to solve such a MINLP problem, a sequential mixed integer linear programming (SMILP) algorithm is used.

However, in order to solve such MINLP problem, we can also use a sequential mixed integer quadratic programming (SMIQP) algorithm instead of the SMILP one. We refer to the review of Boggs and Tolle in [11] for more information on Sequential Quadratic Programming (SQP). The main advantage of SQP is that it is possible to ensure under some assumptions that the algorithm converges to a point satisfying the KKT conditions from any starting point. We can also mention here that SQP is already implemented in several software systems (CONOPT,MINOS,...). This is clearly an important future research direction.

Then, two other non linear methods are proposed in order to numerically solve the differential equations within the optimization problem, namely the explicit euler method and the trapezoidal method. For more details about these methods, see for example Ralston and Rabinowitz [41]. This leads for both cases to MINLP problems that are solved by the SMILP algorithm.

However, even if the non linear numerical integration of the differential equations leads to a more accurate model of the process, it has the drawback that the solution obtained, by using the SMILP algorithm, has no guarantee of optimality since the corresponding optimization problem is nonlinear and generally non convex.

Therefore, we also propose to solve the differential equations by using two piecewise linear approximation methods. The idea of the first method is to discretize the space of the state and control variables in hypercubes. In every hypercube, we build a linear approximation of the solutions of the system of differential equations around the point in the middle of the hypercube. For the second method proposed, we decompose the state and control variables involved in the process dynamics into a set of simplices. Every point in the state/control space can be expressed as a convex combination of extreme points of a simplex. Then the values of the non linear functions describing the process dynamics at a point of a simplex are expressed as the corresponding convex combination of the values of the nonlinear function at the extreme points of the simplex. These two methods were inspired by finite element methods. We refer to Thomée [48] for a review on finite element methods.

The outline of the Chapter is the following. First, we present four numerical ways of approximating numerically the solution of the differential equations within an optimization problem. Then, we illustrate on an example, composed of a single batch task, the efficiency of these four methods. Finally, we solve a more general case composed of parallel batch tasks.

4.1 Modeling the process dynamics

The solution of optimization problems with general differential equations is a very difficult problem. Only very small instances can be succesfully solved when the numerical solution of the differential equations has to be computed within the optimization problem. In this section, we present four methods to compute numerically the solution of the differential equations that can be used within the resolution of an optimization problem.

4.1.1 Description of the process dynamics

The very general process dynamics that we address in this Chapter is the following :

$$\frac{dx(t)}{dt} = f(x(t), u(t)) \tag{4.1}$$

$$x(0) \text{ is fixed} \tag{4.2}$$

$$x^L \leq x(t) \leq x^U, u^L \leq u(t) \leq u^U \tag{4.3}$$

where $f(x(t), u(t))$ is a general continuous and differentiable function, $x(t)$ is the vector of the state variables of the system studied and $u(t)$ is the vector of the control variables of the system.

4.1.2 Approximated model formulation

Next, we present four general discretization methods in order to model the dynamics of the system represented by (4.1).

◇ The Euler explicit and the trapezoidal method

We begin first with the Euler explicit method. We approximate the value of the state variables by approximating the solution of the differential equation (4.1) by the following set of discrete time equations:

$$x_0 = x(0)$$

$$x_i = x_0 + h\left(\sum_{p=0}^{i-1} f(x_p, u_p)\right) \qquad \forall i \in \{1, \dots, K\}$$

where $K+1$ is the number of discrete points in the time interval for approximating the solution of the differential equation and h is the constant time step between two discrete points, i.e., the discrete times t_i, $i \in \{0, \dots, K\}$, are $t_i = ih$. We set that $x(t_i) = x_i$ and $u(t_i) = u_i \; \forall \; i \in \{0, \dots, K\}$.

However, for such problems, there is a much more stable implicit method (see for example Ralston and Rabinowitz [41]) than the Euler explicit one and that is called the trapezoidal method. The value of the state variables is approximated as follows :

$$x_0 = x(0)$$

$$x_i = x_0 + \frac{h}{2}\left(\sum_{p=0}^{i} f(x_p, u_p) + \sum_{p=1}^{i-1} f(x_p, u_p)\right) \; \forall \; i \in \{1, \dots, K\}$$

where again $K+1$ is the discrete number of points in the time interval and h is the constant time step between two discrete points.
Since the trapezoidal method is more stable than the euler explicit one and the number of constraints and variables for both methods are the same, we decide to only use the trapezoidal method for the rest of the Chapter.

◇ The collocation method

The main idea of the method is to assume that the solution of the set of differential equations (4.1) is a set of $K + 1$ polynomials of given degree K. The derivative with respect to t of these polynomials has to be equal to the right hand side of the differential equation (4.1), i.e. $f(x(t), u(t))$.

We assume that the solution of the differential equations (4.1) is the Lagrange interpolation polynomials of degree K written as follows :

$$x(t) = \sum_{p=0}^{K} c_p \phi_p(t) \tag{4.4}$$

where

$$\phi_p(t) = \prod_{k=0,k\neq p}^{K} \frac{(t - t_k)}{(t_p - t_k)} \tag{4.5}$$

for the $K + 1$ collocation points t_i, $i \in \{0, \ldots, K\}$, such that $t_i = ih$.

We impose that the derivative of these polynomials is equal to $f(x(t), u(t))$:

$$\frac{dx(t)}{dt} = \sum_{p=0}^{K} c_p \phi'_p(t) = f(x(t), u(t))$$

where

$$\phi'_p(t) = \frac{\sum_{k=0|k\neq p}^{K} \left(\prod_{l=0|l\neq k,l\neq p}^{K} (t - t_l)\right)}{\prod_{k=0|k\neq p}^{K} (t_p - t_k)}$$

for the $K + 1$ collocation points.

We discretize the proposed solution of the set of differential equations at the $K + 1$ collocation points.

We set $x(t_i) = x_i \forall i \in \{0, \ldots, K\}$ and $\phi_p(t_i) = \phi_{p,i} \forall p \in \{0, \ldots, K\}, i \in \{0, \ldots, K\}$. Then we evaluate theses polynomials at the discretized time points t_i for $i = 0, \ldots, K$, and we have

$$x_i = \sum_{p=0}^{K} c_p \phi_{p,i} \ \forall i \in \{0, \ldots, K\} \tag{4.6}$$

where

$$\phi_{p,i} = \prod_{k=0,k\neq p}^{K} \frac{(t_i - t_k)}{(t_p - t_k)} = \delta_{p,i} \ \forall p \in \{0, \ldots, K\}, i \in \{0, \ldots, K\} \tag{4.7}$$

and $\delta_{p,i}$ is the kronecker delta.

Therefore, we have :

$$x_i = \sum_{p=0}^{K} c_p \phi_{p,i} = \sum_{p=0}^{K} c_p \delta_{p,i} = c_i \ \forall i \in \{0, \ldots, K\} \tag{4.8}$$

The variables for the control ($u(t)$) are discretized and are constant over each time period and are expressed as follows : $u(t_i) = u_i$ for $i \in \{0, \ldots, K\}$.

The discretized conditions imposed on the approximation of the solution of the differential equations (4.1) are written as

$$x_0 = x(0)$$
$$\sum_{p=0}^{K} x_p \phi'_{p,i} = f(x_i, u_i) \; \forall i \in \{0, \ldots, K\}$$

For both methods (the trapezoidal and the collocation), if $f(x(t), u(t))$ is nonlinear and non convex, we have nonlinear equality constraints in the model formulation and therefore the optimization problem is non convex. Therefore, it is not possible to estimate the global quality of the best solution obtained, in general. We solve these non linear optimization problems by using a sequential linear programming (SLP) algorithm.

◇ A first piecewise linear approximation

We suppose that in this case the function $f(x(t), u(t))$ is nonlinear. The objective is to transform or approximate the general system of nonlinear differential equations (4.1) by multiple systems of linear differential equations for which explicit solutions are known. We decompose the (x, u) space in hypercubes and the nonlinear dynamics of the system is linearly approximated around the middle point of the each hypercube. For each linear approximation, an explicit solution exists. In the formulation, at each discrete time, we have to model first in which hypercube we are and then use the explicit solution of the corresponding linear dynamic.

In order to model in which hypercube we are, we have to introduce a new variable : $\delta_{j,t}$ which is equal to 1 if the system is in the hypercube j of the space (x, u) at time t, and 0 otherwise.
We divide the (x, u) space in $\mid J \mid$ hypercubes. Each component of the vector x of state variables is discretized in $Nbrdiv1$ elements and each component of the vector u of control variables is discretized in $Nbrdiv2$ elements. Suppose that $\underline{X}_j$ and $\overline{X}_j$ are the vectors of the lower and upper bounds on the values of vector x in hypercube j, respectively. Similarly, suppose that $\underline{U}_j$ and $\overline{U}_j$ are the lower and upper bounds on the values of vector u in hypercube j.

By using the next set of inequalities, we can model in which hypercube we are at time $i \in \{0, \ldots, K\}$ in the following way.

$$(x^L, u^L)(1 - \delta_{j,i}) + (\underline{X}_j, \underline{U}_j)\delta_{j,i} \leq (x_i, u_i) \leq (\overline{X}_j, \overline{U}_j)\delta_{j,i} + \left(x^U, u^U\right)(1 - \delta_{j,i}) \qquad \forall j \in \{1, \ldots, \mid J \mid\}, i \in \{0, \ldots, K\} \tag{4.9}$$

$$\sum_{j \in J} \delta_{j,i} = 1 \qquad \forall i \in \{0, \ldots, K\} \tag{4.10}$$

$$\delta_{j,i} \in \{0, 1\} \qquad \forall j \in \{1, \ldots, \mid J \mid\}, i \in \{0, \ldots, K\} \tag{4.11}$$

We have to approximate the solution of the nonlinear differential system around the middle point of every hypercube j $(\tilde{x}_j, \tilde{u}_j)$ of the (x, u) space. This linear differential equation for the hypercube j can be expressed as follows :

$$\dot{x} = A_j x + B_j u + Cste_j \tag{4.12}$$

where

$$A_j = \frac{\partial f}{\partial x}\Big|_{(\tilde{x}_j, \tilde{u}_j)}, B_j = \frac{\partial f}{\partial u}\Big|_{(\tilde{x}_j, \tilde{u}_j)}, Cste_j = f(\tilde{x}_j, \tilde{u}_j) - A_j \tilde{x}_j - B_j \tilde{u}_j$$

The discrete time solution of this differential equation is the following if we suppose that (x_{i-1}, u_{i-1}) is in the hypercube j :

$$\begin{aligned} x_i &= e^{A_j \Delta t} x_{i-1} + \left(\int_0^{\Delta t} e^{A_j(\Delta t - \tau)} d\tau \right) K_{j,i-1} \\ &= e^{A_j \Delta t} x_{i-1} + \left(I\Delta t - A_j \frac{(\Delta t)^2}{2!} + A_j^2 \frac{(\Delta t)^3}{3!} - \dots \right) e^{A_j \Delta t} K_{j,i-1} \\ &\approx e^{A_j \Delta t} x_{i-1} + \left(I\Delta t - A_j \frac{(\Delta t)^2}{2!} + A_j^2 \frac{(\Delta t)^3}{3!} \right) e^{A_j \Delta t} K_{j,i-1} \end{aligned}$$

where $\Delta t = t_i - t_{i-1}$ $\forall i \in \{1, \dots, K\}$ is a predefined fixed constant and $K_{j,i} = B_j u_i + Cste_j$.

For the whole area composed of the $\mid J \mid$ hypercubes, the approximation of the general discrete time solution is :

$$x_i = \sum_{j=1}^{|J|} \delta_{j,i-1} [e^{A_j \Delta t} x_{i-1} + \left(I\Delta t - A_j \frac{(\Delta t)^2}{2!} + A_j^2 \frac{(\Delta t)^3}{3!} \right) e^{A_j \Delta t} K_{j,i-1}]$$

for all $i \in \{0, \dots, K\}$.

This equation is nonlinear but can be transformed into a linear form, see for example in Bemporad and Morari [6], as follows :
for all $i \in \{0, \dots, K\}$,

$$x_i = \sum_{j \in J} z_{j,i}$$

and for all $i \in \{1, \dots, K\}$, $j \in J$

$$\begin{aligned} & z_{j,i} \leq M_j \delta_{j,i-1} \\ & z_{j,i} \geq m_j \delta_{j,i-1} \\ & z_{j,i} \leq [e^{A_j \Delta t} x_{i-1} + \left(I\Delta t - A_j \frac{(\Delta t)^2}{2!} + A_j^2 \frac{(\Delta t)^3}{3!} \right) e^{A_j \Delta t} K_{j,i-1}] \\ & \qquad - m_j (1 - \delta_{j,i-1}) \\ & z_{j,i} \geq [e^{A_j \Delta t} x_{i-1} + \left(I\Delta t - A_j \frac{(\Delta t)^2}{2!} + A_j^2 \frac{(\Delta t)^3}{3!} \right) e^{A_j \Delta t} K_{j,i-1}] \\ & \qquad - M_j (1 - \delta_{j,i-1}) \end{aligned}$$

with

$$M_j \geq max_{(\underline{X}_j,\underline{U}_j)\leq(x,u)\leq(\overline{X}_j,\overline{U}_j)} \Big[e^{A_j\Delta t}x + \left(I\Delta t - A_j\frac{(\Delta t)^2}{2!} + A_j^2\frac{(\Delta t)^3}{3!}\right)$$
$$e^{A_j\Delta t}(B_j u + Cste_j)\Big]$$
$$m_j \leq min_{(\underline{X}_j,\underline{U}_j)\leq(x,u)\leq(\overline{X}_j,\overline{U}_j)} \Big[e^{A_j\Delta t}x + \left(I\Delta t - A_j\frac{(\Delta t)^2}{2!} + A_j^2\frac{(\Delta t)^3}{3!}\right)$$
$$e^{A_j\Delta t}(B_j u + Cste_j)\Big]$$

To estimate M_j and m_j, we discretize the (x, u) space in the hypercube j with a very fine grid in order to compute an accurate value for M_j and m_j by complete enumeration.

It is well known that such model formulations are very poor since big M constraints are usually not tight at all. Therefore, for such model formulation, reformulation is crucial in order to be able to solve large instances.

$\diamond$ A second piecewise linear approximation

We suppose here that $f(x(t), u(t))$ is nonlinear. The idea is to approximate $f(x(t), u(t))$ by a piecewise linear function using integer linear programming, as proposed in Lee and Wilson [26].

We first triangle the space (x, u) with a finite set of simplices $\triangle_j$ having vertex set $\nu(\triangle_j)$ where $j \in \{1, \ldots, |J|\}$. Each variable $[x(t); u(t)]$ is a convex combinaison of predefined discrete constants X_k, U_k with $k \in \{1, \ldots, m\}$, which are the values of (x, u) at the extreme points of the simplices, and $m = \left|\bigcup_{j\in J} \nu(\triangle_j)\right|$. We discretize the variables $[x(t); u(t)]$ in $T + 1$ time periods and we express the convex combination as follows :

$$x_i = \sum_{k=1}^{m} \lambda_{k,i} X_k \forall i \in \{0, \ldots, T\}$$
$$u_i = \sum_{k=1}^{m} \lambda_{k,i} U_k \forall i \in \{0, \ldots, T\}$$
$$\sum_{k=1}^{m} \lambda_{k,i} = 1, \lambda_{k,i} \geq 0 \ \forall k \in \{1, \ldots, m\}, \forall i \in \{0, \ldots, T\}$$

We define a binary variable $y_{j,i}$ for each simplex $j \in \{1, \ldots, |J|\}$ and time $i \in \{0, \ldots, T\}$ to impose the adjacency condition, or membership of simplex j, as follows :

$$\begin{aligned} y_{j,i} &= 0 \text{ if } (x_i, u_i) \notin \triangle_j \\ &= 1 \text{ if } (x_i, u_i) \in \triangle_j \end{aligned}$$

By adding the following set of constraints,

$$\sum_{j=1}^{|J|} y_{j,i} = 1 \qquad \forall i \in \{0,\dots,T\} \tag{4.13}$$

$$\lambda_{k,i} \geq 0 \qquad \forall k \in \{1,\dots,m\}, i \in \{0,\dots,T\} \tag{4.14}$$

$$\sum_{k=1}^{m} \lambda_{k,i} = 1 \qquad \forall i \in \{0,\dots,T\} \tag{4.15}$$

$$x_i = \sum_{k=1}^{m} \lambda_{k,i} X_k \forall i \in \{0,\dots,T\} \tag{4.16}$$

$$u_i = \sum_{k=1}^{m} \lambda_{k,i} U_k \forall i \in \{0,\dots,T\} \tag{4.17}$$

$$\lambda_{k,i} - \sum_{j=1 | k \in \nu(\triangle_j)}^{|J|} y_{j,i} \leq 0 \qquad \forall k \in \{1,\dots,m\}, \forall i \in \{0,\dots,T\} \tag{4.18}$$

$$y_{j,i} \in \{0,1\} \qquad \forall j \in \{1,\dots,|J|\}, i \in \{0,\dots,T\}, \tag{4.19}$$

we can model the simplex in which we are at time i and express (x_i, u_i) as a convex combination of extreme points of the simplex. The solution of the differential equation (4.1) can be approximated using the explicit Euler method as follows :

$$x_{i+1} = x(t+\Delta t) = \sum_{k=1}^{m} \lambda_{k,i} \left(f(X_k, U_k)\Delta t + X_k \right) \forall i \in \{0,\dots,T-1\} \tag{4.20}$$

Note that this solution satisfies the initial condition. By setting $\Delta t = 0$, we can observe that

$$x_{i+1} = \sum_{k=1}^{m} \lambda_{k,i} X_k = x_i \forall i \in \{0,\dots,T-1\}.$$

This corresponds exactly to the initial condition.

A more stable implicit method, such as the trapezoidal presented above, could be used as well in order to approximate the solutions of the set of differential equations.

Lee and Wilson in [26] propose a valid inequality that can help to speed up the resolution of the problem. Let $\triangle$ be the set of simplices having vertex set $\nu(\triangle)$. For $\beta \subset \triangle$, with vertex set $\nu(\beta) = \bigcup_{j \in \beta} \nu(\triangle_j)$,

$$\sum_{j \in \beta} y_{j,i} - \sum_{v \in \nu(\beta)} \lambda_{v,i} \leq 0 \qquad \forall i \in \{0,\dots,T\} \tag{4.21}$$

These cuts improve the model formulation and can be used in a branch-and-cut algorithm.

4.2 Application to a single batch task

We have proposed two nonlinear non convex models for the approximation of the solutions of the differential equations which are the trapezoidal method and the collocation method. We have also presented two linear approximation models for modeling the solutions of the differential equations.

In this Section, we test the quality and the efficiency of these four models on a simple test case composed of only one heating task. For this heating task, the process dynamics are described by two non linear differential equations, one for the concentration of the reactant (Ca) and one for the temperature (T). The objective of this simple scheduling problem is to minimize the time needed for reaching the temperature $450.15K$ and the initial temperature is $298.15K$.
We fix as initial conditions $Ca(0) = Ca_0 = 1.1 mole/l$ and $T(0) = T_0 = 298.15K$. The two differential equations are the following :

$$\frac{dCa(t)}{dt} = -2k_0 e^{-\frac{E}{R\,T(t)}} Ca(t)^2 \tag{4.22}$$

$$\frac{dT(t)}{dt} = -\frac{\lambda}{\rho\, C_p} k_0 e^{-\frac{E}{R\,T(t)}} Ca(t)^2 + qh(t)(T_h - T(t)) \tag{4.23}$$

We fixed the other parameters of these two differential equations as follows : $k_0 = 3.512 * 10^{-2} * 3600[l/mole.h]$, $E = 10560[J/mole]$, $R = 8.314[J/mole.K]$, $\lambda = -41.85[J/mole]$, $\rho = 0.9342[kg/l]$, $C_p = 3.01[J/kg.K]$ and $T_h = 493.15[K]$.

The vector of state variables is here

$$x(t) = \begin{pmatrix} Ca(t) \\ T(t) \end{pmatrix}.$$

The control variable $u(t)$ is the rate $qh(t)$ ($[1/h]$) of hot water given to the system. In order to detect the end of the heating task, we add a binary variable $z(t)$ which is 1 when the heating task is finished and 0 otherwise. No resource is shared.

The state, the control and the binary variables are evaluated at discretized time points t_i for $i \in \{0, \ldots, T\}$ and are expressed as follows : $Ca(t_i) = Ca_i$, $T(t_i) = T_i$, $qh(t_i) = qh_i$ and $z(t_i) = z_i$. The conditions on these variables can be written as follows :

$$\begin{aligned}
&0 \leq qh_i \leq 3.6 \ \forall i \in \{0, \ldots, T\} \\
&0 \leq Ca_i \leq 1.1 \ \forall i \in \{0, \ldots, T\} \\
&298.15 \leq T_i \leq 600 \ \forall i \in \{0, \ldots, T\} \\
&z_i \in \{0, 1\} \ \forall i \in \{0, \ldots, T\}
\end{aligned}$$

The global formulation of the optimization problem is the following :

$$\min \sum_{i=0}^{T} i\ z_i \tag{4.24}$$

$$s.t. \quad \text{the discrete solution of the process dynamics (4.22)-(4.23)} \tag{4.25}$$

$$T_i \geq 450.15\ z_i \forall i \in \{0, \dots, T\} \tag{4.26}$$

$$\sum_{i=0}^{T} z_i = 1 \tag{4.27}$$

$$0 \leq qh_i \leq 3.6\ \forall i \in \{0, \dots, T\} \tag{4.28}$$

$$0 \leq Ca_i \leq 1.1\ \forall i \in \{0, \dots, T\} \tag{4.29}$$

$$298.15 \leq T_i \leq 600\ \forall i \in \{0, \dots, T\} \tag{4.30}$$

$$z_i \in \{0, 1\}\ \forall i \in \{0, \dots, T\} \tag{4.31}$$

$$Ca_0 = 1.1, T_0 = 298.15 \tag{4.32}$$

This is a minimum time optimal control problem for the system composed of the two nonlinear differential equations (4.22)-(4.23). We refer to Berkovitz [7] for a review of some important aspects of optimal control theory. The optimal solution for this simple test case is trivial. By using the maximum rate of hot water available, i.e. $qh(t) = 3.6$ ($[1/h]$), we will minimize the time to reach the temperature $450.15K$. In Figure 4.1, we illustrate the simulated solution of the original nonlinear differential equations using the optimal values given above for the control variables. The simulated solution is computed by using the Simulink toolbox of Matlab. The temperature 450.15 K is reached after 0.4028 h.

For the four methods, the software used is Xpress-MP and the computer is a Pentium 4, running at 3 GHz.

For solving the two nonlinear non convex mixed integer programming problems, we use the sequential mixed integer linear programming algorithm of Xpress-MP. For the other two linear approximation methods, we use the standard MIP solver implemented in Xpress-MP.

The results obtained with the four methods are presented in Figures 4.2-4.5. The solid lines represent the simulated solution of the original nonlinear differential equations using the optimal values obtained for the control variables. For the four methods, we consider the following time interval : $t \in [0; 0.83]$ $[h]$. The discretized solution, computed when solving the optimization problem, is represented by discrete stars.

Figure 4.2 represents the solution obtained with the collocation method. We select a time step of 0.083 h and 11 discrete time points.

Figure 4.3 presents the results with the trapezoidal method. We select the same time step and the same number of discrete time points as for the collocation method.

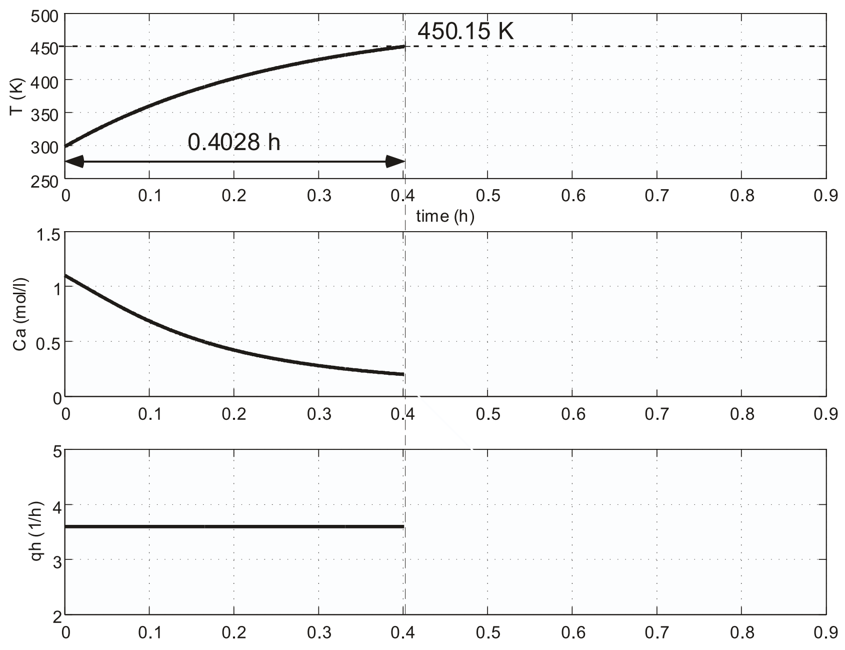

Figure 4.1: The optimal solution. Evolution of the Temperature, the concentration and the heating rate for the heating task

With the first piecewise linear approximation, we obtain the results in Figure 4.4. With this kind of approximation, the number of divisions of the space in hypercubes has to be small. Otherwise, the size of the problem will be too big and the solution time increases very fast. We discretize the space in three for each state and control variables and this leads to a decomposition of the space in 8 hypercubes. Moreover, an important feature is to reduce sufficiently the time step in order to obtain a solution of good quality. The time step is 0.055 h and the number of discrete time points is 16.

With the second piecewise linear approximation method, we obtain the results presented in Figure 4.5. As for the previous method, the most important design feature for this method is to use a small time step and a small number of simplices. We decomposed the space into 5 simplices by using the Delaunay triangulation algorithm proposed by Barber et al. in [5], the time step is of 0.055 h and the number of discrete time points is 16.

We have tried to use the trapezoidal method in the second piecewise linear approximation method but the solution obtained for this example was not much better than the one obtained when using the explicit euler method.

In Table 4.1, we summarize the number and type of constraints and variables for the four methods.

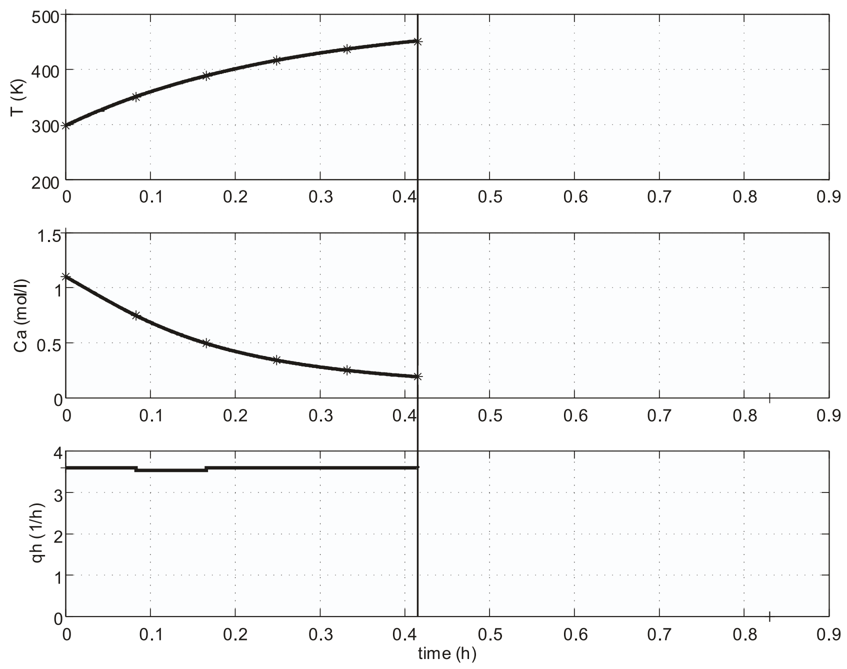

Figure 4.2: Collocation method. Evolution of the Temperature, the concentration and the heating rate for the heating task

	Lin. Constr.	Nonlin. Constr.	Cont. var.	Bin. var.
Collocation	31	22	31	11
Trapezoidal	31	20	31	11
P. lin. approx I	1407	0	288	144
P. lin. approx II	255	0	176	96

Table 4.1: Number of constraints and variables for the four approximation methods

We can observe that the collocation and the trapezoidal methods need the same number of constraints and variables. We can also observe that the piecewise linear approximation method II is more compact than the piecewise linear approximation method I.

In Table 4.2, we summarize the results of the optimization problem obtained for the four methods.

We observe first that the profiles of the control variables obtained for the trapezoidal method and for the two piecewise linear approximation methods are not very close to the global optimal one represented in Figure 4.1. This will be further discussed below and also in the Section “General comments about the results”.

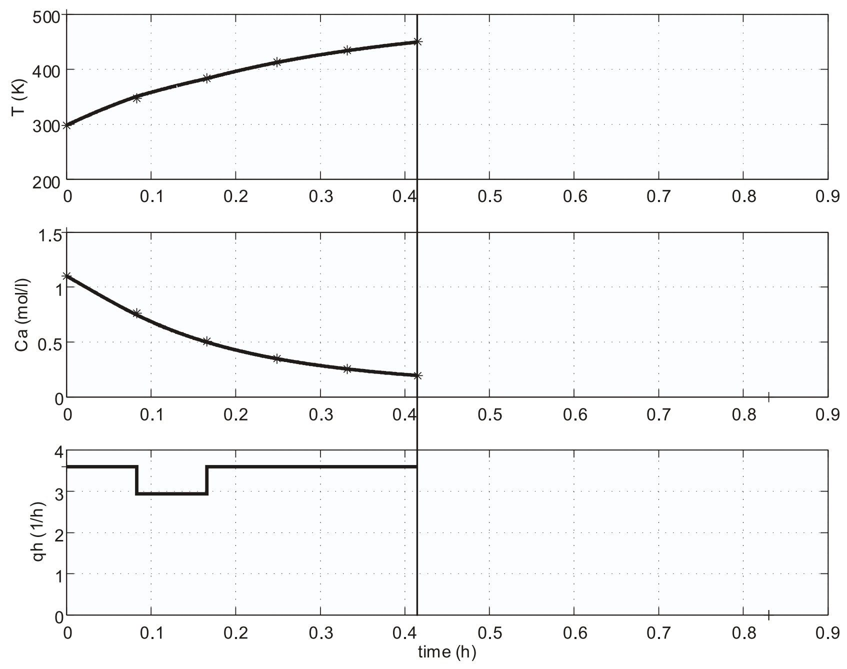

Figure 4.3: Trapezoidal method. Evolution of the Temperature, the concentration and the heating rate for the heating task

	Time points	CPU time (s)	Nodes	Object. (h)
Collocation	11	1.81	11	0.415
Trapezoidal	11	13.37	11	0.415
Piecew. lin. approx. I	16	9	2288	0.387
Piecew. lin. approx. II	16	0.3	69	0.387

Table 4.2: Solution of the optimization problem for the four approximation methods

The piecewise linear approximation method II provides the optimal objective value in less CPU time. However, the quality of the approximation of the non linear process dynamics has to be checked.

Note for the collocation method that the difference in the optimal objective value obtained is partly due to the difference in the discrete time points used. By using 16 time points for the collocation method, we get 0.387 h as optimal objective value.

However, this was not observed using the trapezoidal method. By increasing the number of time points to 16, the optimal objective function obtained is $0.44h$.

In Table 4.3, we compute three types of error for the four approximations of the non linear process dynamics. We compare the state variables values obtained in the simulated solutions using the control variables obtained by optimization, with

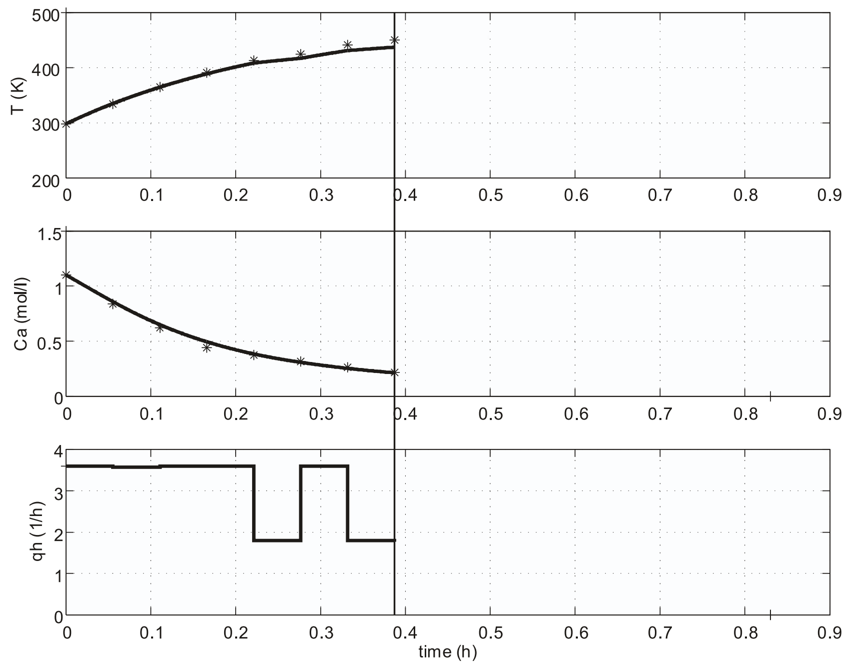

Figure 4.4: First piecewise linear approximation method. Evolution of the Temperature, the concentration and the heating rate for the heating task

the approximate solutions of the state variables obtained by optimization. The error is defined as $e_i =$ simulated value$_i -$ approx. value$_i$ at the different discrete time points $i \in \{0, \ldots, \text{time points} - 1\}$. We define the three types of error below.

The *Average Error* is equal to $(\sum_{i=0}^{\text{time points opt}} e_i/(\text{time points opt} + 1))$, the *Maximum Error* is equal to $(\max_{i\in\{0,\ldots,\text{time points opt}\}} |e_i|)$ and the *Average Absolute Error* is equal to $(\sum_{i=0}^{\text{time points opt}} |e_i|/(\text{time points opt} + 1))$ where for each method *time points opt* corresponds to the minimum number of time points necessary to complete the heating task, i.e. the optimal objective value of the corresponding optimization problem.

In Table 4.4, we give the simulated values of the temperature obtained at the end of the heating task by the four methods.

It is important to compare the best solutions obtained by the four methods represented in Figures 4.2-4.5 with the trivial global optimal solution for this simple test case represented in Figure 4.1. The solution obtained for the collocation method is very close to the global optimal solution of the problem, i.e. $qh(t) = 3.6$ ($[1/h]$). The best solutions obtained by the three other methods have different profile for the control variables than the optimal one represented in Figure 4.1. However, we can observe that the best solutions obtained by the four methods have a corresponding objective value very close to the global optimal objective

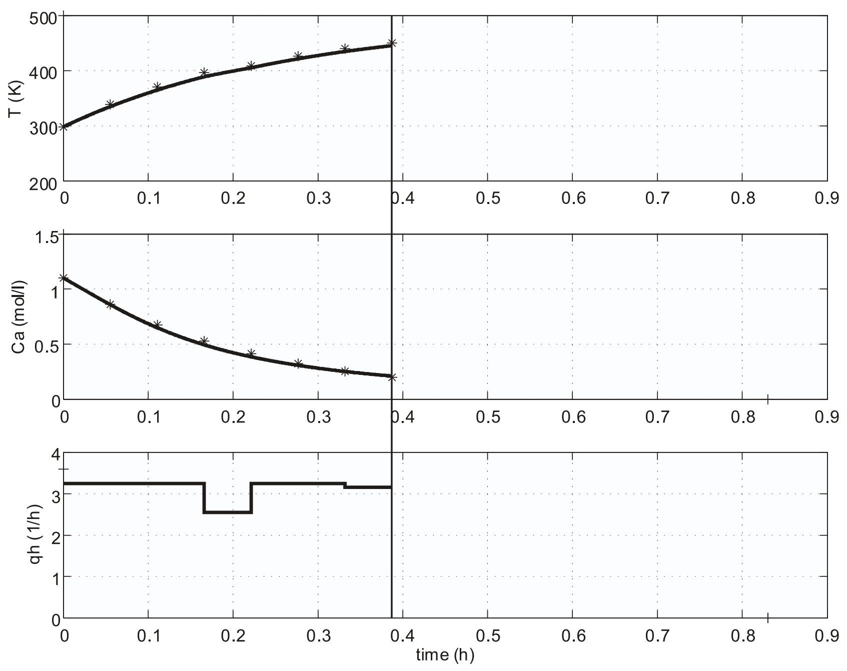

Figure 4.5: Second piecewise linear approximation method. Evolution of the Temperature, the concentration and the heating rate for the heating task

	Average Error		Maximum Error		Av. Abs. Error	
	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)
Collocation	0.62	0	1.67	0	0.62	0
Trapezoidal	0.21	0	3.1	0.01	0.82	0
PL Approx I	-4.6	0.02	12.81	0.05	4.81	0.02
PL Approx II	-4.37	-0.01	7.15	0.03	4.37	0.01

Table 4.3: Comparison of the average, the maximum and the average absolute errors of approximation of the non linear dynamics of the system by the four approximation methods

	Final Temperature
Collocation	451.8 K
Trapezoidal	449.6 K
PL Approx I	437.3 K
PL Approx II	445.5 K

Table 4.4: Final simulated temperature values for the four methods

one.

Many reasons can explain this difference. First of all, the four approximation methods used for solving the general problem cannot ensure the global optimality of the best solution obtained. The solution obtained by using the two first methods (the collocation and the trapezoidal method) is at best locally optimal. As mentioned earlier in this Chapter, it could be interesting for these first two methods to compare the best solutions obtained here by using the SMILP algorithm with the ones obtained by using the SMIQP algorithm. We can recall here also that by using the two piecewise linear approximation methods, the selection of their corresponding parameters is very important in order to approximate well the nonlinear dynamics of the process.

Finally, the discretization of time can also have a strong impact on the quality of the best solutions obtained by the four methods. The profile of the control variables can be far from the optimal one if the discretization of time is not appropriate.

In order to further improve the solutions obtained by the four methods, we discuss below the choice of their corresponding parameters.

General comments about the results

- ◇ We can observe that the collocation method is clearly the best for this application. The approximation of the solution of the nonlinear differential set of equations is quite accurate and the final local optimal solution is obtained quickly and is similar to the other methods. However, in terms of robustness, we can observe that the collocation method is not the best. When we increase the number of collocation points, the quality of the solution obtained can vary much and therefore this method is not reliable.

  If, for example, we increase by three the number of time points, we can observe in Table 4.5 that the solution time and the total number of branch-and-bound nodes do not increase too much in comparison with the initial case composed of 11 time points.

	Time points	CPU time (s)	Nodes	Object. (h)
Collocation (11)	11	1.81	11	0.415
Collocation (14)	14	3	15	0.447

Table 4.5: Solution of the optimization problem for the collocation method with 11 and 14 time points

However, in Table 4.6 and in Figure 4.6, we can observe that the quality of the approximation of the non linear dynamics of the system is very bad. We can also mention that the best solution obtained for this case is not close to the global optimal solution of the problem represented in Figure 4.1.

By increasing the number of time points, the approximation of the nonlinear dynamics of the system provided by the collocation method can be strongly affected and become extremely bad.

In fact, by increasing the number of time points, the degree of the polynomials increases also and therefore the collocation method can become in-

	Average Error		Maximum Error		Av. Abs. Error	
	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)
Collocation (14)	-21.51	0.06	48	0.11	21.51	0.06

Table 4.6: The average, the maximum and the average absolute errors of approximation of the non linear dynamics of the system by the collocation method with 14 time points

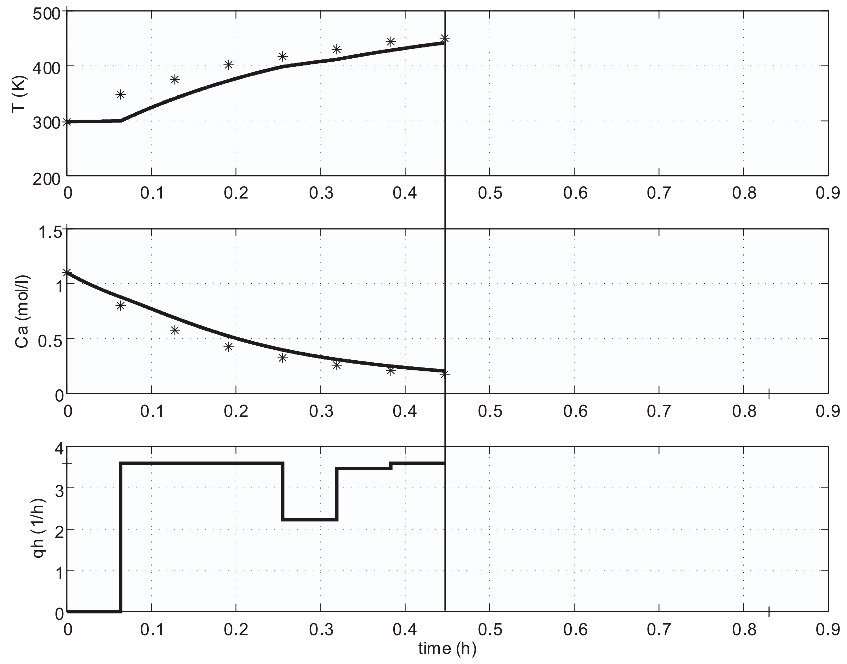

Figure 4.6: Collocation method (14 time points). Evolution of the Temperature, the concentration and the heating rate for the heating task

convenient for numerical purpose because unstable. As shown in Cuthrell and Biegler [17], by introducing finite elements, it is possible to reduce the degree of the polynomials for the collocation method and therefore improve its robustness. In the future, this should be tested on the problem addressed in this Section.

◇ The trapezoidal method provides a quite accurate approximation of the solution of the nonlinear differential equations and a good local optimal solution but is slow in terms of CPU time. The profile of the control variable represented in Figure 4.3 is not exactly the global optimal one given in Figure 4.1. This is due to the discretization of time since if we use the maximum quantity available of hot water at each time period, i.e. if $qh_i = 3.6$ for $i \in \{0, \dots, 10\}$, we are not able to get a better objective function than the one obtained in Figure 4.3, i.e. 0.415 *h*. The best solution for this problem is therefore not unique and is quite dependent of the choice of the time discretization.

We can show that this method is more robust than the collocation one. We illustrate this property below on an example.

If we increase by three the number of time points, we can observe in Table 4.7 that the solution time and the total number of branch-and-bound nodes do not increase too much in comparison with the initial case composed of 11 time points.

	Time points	CPU time (s)	Nodes	Object. (h)
Trapezoidal (11)	11	13.37	11	0.415
Trapezoidal (14)	14	37.6	15	0.447

Table 4.7: Solution of the optimization problem for the trapezoidal method with 14 time points

In Table 4.8 and in Figure 4.7, we can observe that the quality of the approximation of the non linear dynamics of the system is still quite accurate except the approximated value of the temperature for one time point. The same difference between the profile of the control variable obtained here and the global optimal one can be observed in Figure 4.7 but an equivalent explanation as the one given for Figure 4.3 can be provided. The best solution for this problem is therefore not unique and is quite dependent of the choice of the time discretization.

	Average Error		Maximum Error		Av. Abs. Error	
	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)
Trapezoidal (14)	1.02	0	9.3	0.01	1.31	0

Table 4.8: The average, the maximum and the average absolute errors of approximation of the non linear dynamics of the system by the trapezoidal method with 14 time points

The approximation of the solution of the nonlinear dynamics of the system provided by the trapezoidal method remains accurate when we increase the number of time points. This seems to indicate that the robustness of the trapezoidal method is better than the one of the collocation method.

◇ The first piecewise linear approximation method provides a not too bad approximation of the solution of the differential equations and a local optimal solution similar to the one obtained by the other methods. However, the CPU solution time is large. For improving the quality of the approximation, the idea is to increase the number of hypercubes and to decrease the time

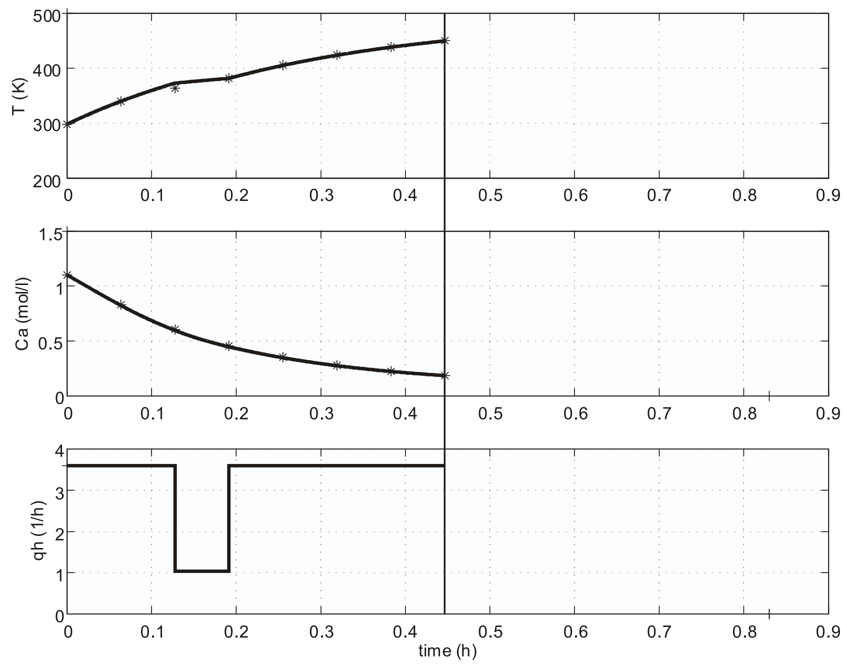

Figure 4.7: Trapezoidal method (14 time points). Evolution of the Temperature, the concentration and the heating rate for the heating task

step. But, rapidly, the size of the problem increases and the CPU solution time becomes quite large. Again, we can observe in Figure 4.4 that the profile obtained for the control variable is not close to the global optimal one represented in Figure 4.1. In this case, we could not show that the bad approximation of the profile of the control variables is due to the time discretization.

In order to obtain a better solution, we try to use another set of parameters for the model formulation. We divide the space not in 8 hypercubes but in 27 by discretizing the space in 3 intervals for each state and command variables, and if we consider only 14 time points, the approximated solution of the non linear dynamics of the system obtained, presented in Table 4.9 and in Figure 4.8, is better for the temperature than the initial one with 8 hypercubes. However, in order to obtain this result, we need a CPU solution of 196 s and 36959 branch-and-bound nodes. We can observed that the profile of the control variables is still not close to the global optimal one but in this case if we use the maximum quantity available of hot water at each time period, i.e. if $qh_i = 3.6$ for $i \in \{0, \dots, 13\}$, we are not able to get a better objective function than the one obtained in Figure 4.8, i.e. $0.447\ h$. The best solution for this problem is therefore not unique and is dependent of the choice of the time discretization.

	Average Error		Maximum Error		Av. Abs. Error	
	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)
PL Approx I (8 hy.)	-4.6	0.02	12.81	0.05	4.81	0.02
PL Approx I (27 hy.)	-0.46	0.03	2.23	0.09	0.6	0.03

Table 4.9: The average, the maximum and the average absolute errors of approximation of the non linear dynamics of the system by the first piecewise linear approximation method with 8 and 27 hypercubes

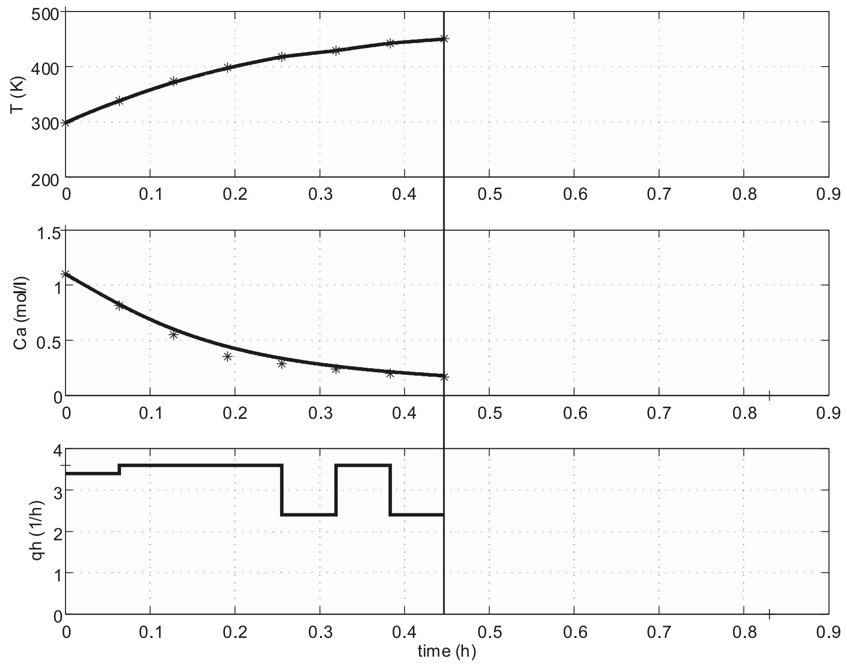

Figure 4.8: The first piecewise approximation method (27 hypercubes). Evolution of the Temperature, the concentration and the heating rate for the heating task

If we also increase the number of time points, the quality of the approximated solutions of the nonlinear dynamics of the system should be further improved. However, it was not possible to solve such a large problem instance in a reasonable amount of computing time.

◇ For the second piecewise linear approximation, the quality of the approximated solution of the differential equations is better than the one obtained by the first piecewise approximation method in terms of average, maximum and average absolute errors, and the CPU solution time is much smaller. The local optimal solution obtained is similar to the one obtained by the other methods. The profile of the control variable represented in Figure 4.5 is not exactly the global optimal one given in Figure 4.1. This is due to the discretization of time since if we use the maximum quantity available of hot water at each time period, i.e. if $qh_i = 3.6$ for $i \in \{0, \ldots, 15\}$, we are not able to get a better objective function than the one obtained in Figure 4.5, i.e. 0.387 h. The best solution for this problem is therefore not unique and is quite dependent of the choice of the time discretization.

In order to increase the quality of the approximated solution of the differential equations, we have to discretize with a finer grid and with more simplices.

For example, we consider 56 time points for the same time interval and the same number of simplices as for the previous piecewise linear approximation method 2. For this case, the formulation is composed of 895 constraints, 336 binary variables and 616 continuous variables.

The quality of the approximated solution of the non linear dynamics of the system obtained, presented in Table 4.10 and in Figure 4.9, is better for the temperature than the previous one. In Figure 4.9, the deviation from the optimal control profile presented in Figure 4.1 is reduced because we have chosen a finer time grid.

We could also further improve the quality of the approximation by increasing the number of simplices. But this leads to a larger problem instance that is more difficult to solve.

	Average Error		Maximum Error		Av. Abs. Error	
	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)
PL Approx II (16)	-4.37	-0.01	7.15	0.03	4.37	0.01
PL Approx II (56)	-1.43	-0.04	1.99	0.06	1.43	0.04

Table 4.10: The average, the maximum and the average absolute errors of approximation of the non linear dynamics of the system by the second piecewise linear approximation method with 16 and 56 time points

However, in order to obtain such a good approximation, we need a CPU solution of 341 s and 79065 branch-and-bound nodes.

It is possible to speed up the resolution of such large instances by improving the model formulation. In order to improve the tightness of the piecewise linear approximation II, we add, at the top node of the branch-and-bound tree after that the cuts were generated automatically by the MIP solver, all the violated valid inequalities of the form (4.21) to the matrix of the constraints.

In Table 4.11, we compare the initial piecewise linear approximation II and the improved one.

We can observe for this instance that the improved formulation provides better results than the initial one since the total number of nodes and the CPU solution time are reduced. For the improved formulation, we have added, at

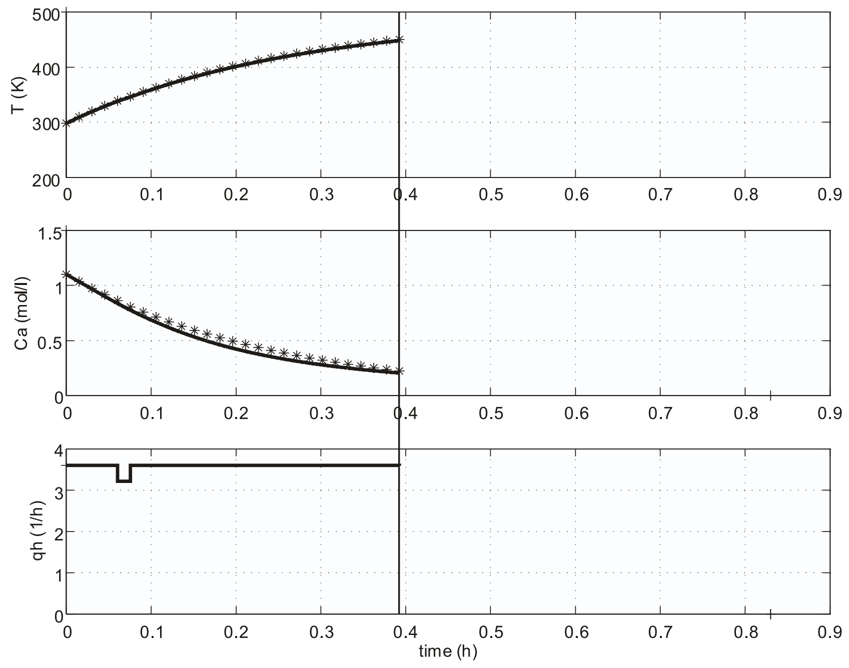

Figure 4.9: The second piecewise approximation method (56 time points). Evolution of the Temperature, the concentration and the heating rate for the heating task

	Time points	CPU time (s)	Nodes	Object. (h)
PL approx II	56	341	79065	0.39
PL approx II improved	56	150	28419	0.39

Table 4.11: Comparison of the initial and improved piecewise linear approximation method 2

the top node of the branch-and-bound tree, 18 violated valid inequalities of the form (4.21) to the matrix of constraints.

To conclude, the results show that the second piecewise linear approximation method makes a good compromise between quality of the solution and CPU time needed for solving the problem. It is clear that the more accurate and robust solution method is the trapezoidal one. The drawback of such method is that it is difficult to solve large mixed integer problems with nonlinear constraints and that the quality of the local optimal solution obtained cannot be measured since the problem is not convex.

4.3 Application to parallel batch tasks

In this Section, we consider a test case composed of a heating task that has to be performed on two reactors. We test the quality and the efficiency of the trapezoidal and of the second piecewise linear approximation methods on this test case. The process dynamics of the heating task and the parameters used were given in the previous section by the differential equations (4.22)-(4.23) and the parameters given below these differential equations. The objective of the scheduling problem is to minimize the time needed for reaching the temperature 400 K in the two reactors and the initial temperature in each reactor is 298.15 K. For this heating task, the concentration of the reactant and the temperature in reactor j are denoted by Ca_j and T_j, respectively. The control variable in each reactor is the rate qh_j of hot water and the total rate of hot water is shared among the two reactors. The binary variables z_j is 1 when the heating task in reactor j is finished and 0 otherwise. Finally, the binary variables z^f is 1 when both reactors have finished the heating task, and is 0 otherwise.

As explained in the previous section, the state, the control and the binary variables are evaluated at discretized time points t_i for $i \in \{0, \ldots, T\}$ and are expressed as : $Ca_j(t_i) = Ca_{j,i}$, $T_j(t_i) = T_{j,i}$, $qh_j(t_i) = qh_{j,i}$, $z_j(t_i) = z_{j,i}$ and $z(t_i)^f = z_i^f$. The initial conditions are : $Ca_{j,0} = 1.1\ \forall j\{1,2\}$ and $T_{j,0} = 298.15\ K$ $\forall j\{1,2\}$.

The global formulation of the optimization problem is the following :

$$\min \sum_{i=0}^{T} i\ z_i^f \tag{4.33}$$

$$\begin{aligned}
s.t. \quad & \text{the discrete solutions of the process dynamics (4.22)-(4.23)} \\
& \text{for reactor 1 and for reactor 2.} && (4.34) \\
& T_{j,i} \leq (400 + \Delta T) z_{j,i} + (400 - \Delta T)(1 - z_{j,i}) \\
& +(100 + \Delta T) \left(\sum_{ii=0}^{i} z_{j,ii} - z_{j,i} \right) \forall j \in \{1,2\}, \forall i \in \{0, \ldots, T\} && (4.35) \\
& T_{j,i} \geq (400 - \Delta T) z_{j,i} + 298.15(1 - z_{j,i}) \\
& \forall j \in \{1,2\}, \forall i \in \{0, \ldots, T\} && (4.36) \\
& \sum_{i=0}^{T} z_{j,i} = 1 \forall j \in \{1,2\} && (4.37) \\
& \sum_{i=0}^{T} z_i^f = 1 && (4.38) \\
& z_i^f \leq \sum_{ii=0}^{i} z_{j,ii} \forall j \in \{1,2\}, \forall i \in \{0, \ldots, T\} && (4.39)
\end{aligned}$$

$$qh_{j,i} \geq 0 \ \forall j \in \{1,2\}, \forall i \in \{0,\ldots,T\} \tag{4.40}$$

$$\sum_{j\in\{1,2\}} qh_{j,i} \leq 3.6 \ \forall i \in \{0,\ldots,T\} \tag{4.41}$$

$$0 \leq Ca_{j,i} \leq 1.1 \ \forall j \in \{1,2\}, \forall i \in \{0,\ldots,T\} \tag{4.42}$$

$$298.15 \leq T_{j,i} \leq 500 \ \forall j \in \{1,2\}, \forall i \in \{0,\ldots,T\} \tag{4.43}$$

$$z_{j,i} \in \{0,1\} \ \forall j \in \{1,2\}, \forall i \in \{0,\ldots,T\} \tag{4.44}$$

$$z_i^f \in \{0,1\} \forall i \in \{0,\ldots,T\} \tag{4.45}$$

$$Ca_{j,0} = 1.1, T_{j,0} = 298.15 \ \forall j \in \{1,2\} \tag{4.46}$$

where constraints (4.35)-(4.36) impose that if $z_{j,i} = 1$, i.e. if the heating task is finished on reactor j at time period i, then $(400 - \Delta T) \leq T_{j,i} \leq (400 + \Delta T)$, where ΔT is a tolerance value on the temperature of 400 K (Here, $\Delta T = 1$ K). Constraint (4.39) imposes that the heating tasks are finished on both reactors at time period i ($z_i^f = 1$) if the heating task is finished on each of the reactors j at or before time period i.

The results obtained with the two methods are represented in Figure 4.10-4.11. Again, the solid line represents the simulated solution of the original nonlinear differential equations using the optimal values obtained for the control variables. The discretized solution obtained by the optimization is represented by discrete stars. For both methods, we consider the following time interval : $t \in [0; 0.86]$ $[h]$.

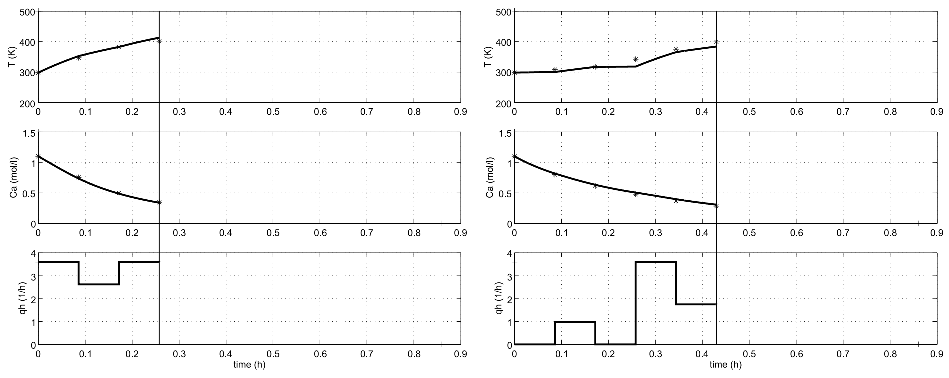

Figure 4.10: The trapezoidal method. Evolution of the Temperature, the concentration and the heating rate for the heating task on reactor 1 (left) and on reactor 2 (right).

Figure 4.10 represents the solution obtained with the trapezoidal method. We select a time step of 0.086 h and 11 discrete time points.

Figure 4.11 represents the solution obtained with the second piecewise linear approximation method. We select a time step of 0.066 h and 14 discrete time

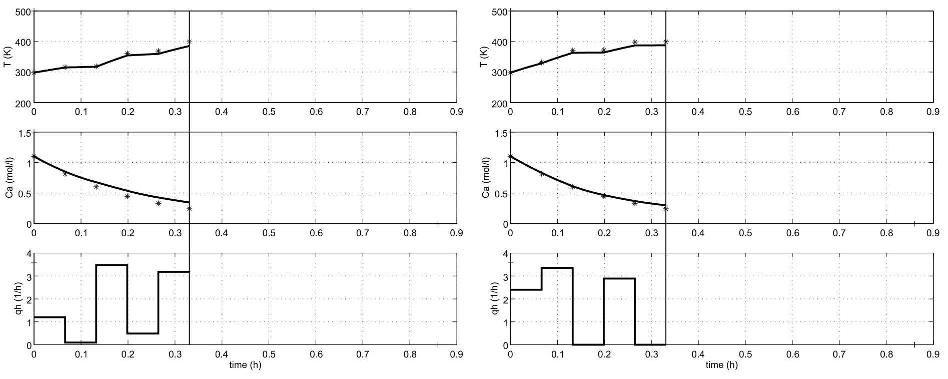

Figure 4.11: The second piecewise linear approximation method. Evolution of the Temperature, the concentration and the heating rate for the heating task on reactor 1 (left) and on reactor 2 (right).

points. We decomposed the space (Ca,T,qh) into 18 simplices by using the Delaunay triangulation algorithm proposed by Barber et al. in [5]. The interval $0 \leq qh \leq 3.6$ is discretized in 3 smaller interval : $0 \leq qh \leq 1.2$, $1.2 \leq qh \leq 2.4$ and $2.4 \leq qh \leq 3.6$. Each interval for $0 \leq Ca \leq 1.1$ and $298.15 \leq T \leq 500$ is discretized in only 1 interval.

In Table 4.12, we summarize the number and type of the constraints and of the variables for the two approximation methods.

	Lin. Constr.	Nonlin. Constr.	Cont. var.	Bin. var.
Trapezoidal	80	40	66	33
PL approx II	741	0	532	546

Table 4.12: Number of constraints and variables for the two approximation methods.

In Table 4.13, we summarize the results of the optimization problem.

	Time points	CPU time (s)	Nodes	Object. (h)
Trapezoidal	11	38.7	25	0.43
PL Approx II	14	3.4	147	0.33

Table 4.13: Solution of the optimization problem for the trapezoidal and the second piecewise linear approximation methods.

We have observed that in order to perform this heating task on one of the reactor by using the maximum rate of hot water available, we will need 0.193 h. Therefore, by performing the heating task on each of the reactors (one after the other) using the maximum rate of hot water available, we will need 0.386 h. This is a feasible solution for our problem. We show in Table 4.13 that the optimal solution obtained by the trapezoidal method is not better than the one we just proposed above with an objective function of 0.386 h. This is mainly due to the discrete time points used since by using the trapezoidal method with 20 time points, we get a feasible solution with an objective value of 0.387 h.

However, we can observe in Table 4.13 that the second piecewise linear approximation method provides the best objective value in less CPU solution time. The quality of the approximation of the non linear process dynamics has to be checked.

In order to estimate the quality of the two approximation methods, we compute, for the trapezoidal method in Table 4.14, and for the second piecewise linear approximation method in Table 4.15, the three types of error defined in the previous section.

	Average Error		Maximum Error		Av. Abs. Error	
	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)
Trapezoidal (1)	4.16	-0.01	11.78	0.02	4.29	0.01
Trapezoidal (2)	-9.45	0.02	23.68	0.03	9.45	0.02

Table 4.14: The average, the maximum and the average absolute errors of approximation of the non linear dynamics of the system by the trapezoidal method for reactor 1 (1) and reactor 2 (2).

	Average Error		Maximum Error		Av. Abs. Error	
	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)	T(K)	Ca(mol/l)
PL Approx II (1)	-5.36	0.07	13.74	0.1	5.36	0.07
PL Approx II (2)	-6.91	0.02	11.4	0.05	6.91	0.02

Table 4.15: The average, the maximum and the average absolute errors of approximation of the non linear dynamics of the system by the second piecewise linear approximation method for reactor 1 (1) and reactor 2 (2)

The second piecewise linear approximation method provides a better approximation of the temperature than the trapezoidal method. However, the concentration of the reactant is better approximated by the trapezoidal method.

In order to significantly improve the accuracy of the approximation methods, we have to discretize furthermore the time and/or the space (Ca,T,qh). This leads to larger problem instances that cannot be solved in a reasonable amount of time.

In Table 4.16, we give the simulated value of the temperature obtained for both methods at the end of the heating task on each reactor.

To conclude, the second piecewise linear approximation method provides quicker the optimal solution of the optimization problem with an approximation of the process dynamics comparable to the one obtained by the trapezoidal method.

	Final Temperature
Trapezoidal (1)	412 K
Trapezoidal (2)	383.8 K
PL Approx II (1)	385.3 K
PL Approx II (2)	387.7 K

Table 4.16: Final simulated temperature values for both methods on each line

However, the adequate or best discretization of the time and of the space in a reasonable amount of simplices, in order to approximate accurately the process dynamics but also be able to solve the optimization problem in a reasonable amount of time, is not easy to compute. Therefore, in order to improve the efficiency of the discretization procedure, an appropriate method should be studied.

Chapter 5

Conclusion

We now review the main results presented in the thesis and propose some directions and perspectives for future research.

In Chapter 2, three special cases of a general cyclic scheduling problem are studied. The general cyclic scheduling problem considered is a general production process modeled by a resource task network, where the resources are the processing units, the utilities shared by the tasks, and the storage tanks containing the intermediate products produced or consumed by the tasks. In this process, there are both batch and continuous tasks. Each batch task has a fixed processing time, can be processed on a subset of reactors and can be repeated several times. The main decision for a batch task is to determine the starting times of the corresponding batches. Precedence and zero waiting time constraints exist between some of the batch tasks. For each continuous task, the processing rate has to be determined over time. This rate has to satisfy some given lower and upper limits. The batch and the continuous tasks consume and produce resources, for which we have some capacity restrictions. Moreover, the continuous tasks cannot be interrupted. The objective is to obtain a cyclic schedule of the mixed plant maximizing its productivity, where productivity is defined as the quantity of finished product produced over one cycle, divided by the cycle duration. We study and tighten the mathematical programming continuous time formulation of three special cases of a batch-plant and mixed-plant scheduling problem.

We show for the first special case composed of one batch task without restrictions on the number of processing units available to perform this batch task (uncapacitated case) that the timing constraints can be strengthened and that these strengthened inequalities are facet-defining for this case. Moreover, we prove that the initial constraints specifying that at most one batch task can begin and finish at each time event are facet-defining for this case as well. Finally, we prove for this case that the duality gap is zero. Then, for the same special case, we consider a restriction on the number of processing units available (capacitated case). We prove by adding a valid inequality that the duality gap is zero as well. Finally, we found a general valid inequality providing a variable lower bound on time slot durations for the uncapacitated and capacitated first special case and we show that this valid inequality is useful in practice.

Then, we consider the second special case composed of several batch tasks where the objective is to maximize a measure of the productivity of the plant. We have shown that by extending the results obtained for the first special case, and by using strengthening techniques, how to strengthen the timing constraints. We also have shown how to formulate the case where different batch tasks have to be performed one after the other in a fixed sequence. Finally, we have shown how to extend the valid inequalities providing a variable lower bound on time slot durations for the first special case.

Finally, we study the third special case composed of multiple batch tasks with fixed sequences, multiple continuous tasks and the sharing of some resources. The batch and the continuous tasks consume and produce some resources. Several resource restrictions must be satisfied, and the objective is to maximize a measure of the productivity of the plant. We have shown that it is possible to strengthen one of the processing constraints for the continuous tasks and that this strengthening can be useful practically. We have also found two valid inequalities limiting the amount processed by the continuous tasks at each time slot and we have shown that these valid inequalities are useful in practice.

For the uncapacitated and capacitated first special case, a research direction concerns an improved model formulation in order to obtain a solution of the LP relaxation that is integral for the kind of objective functions studied. Moreover, for this first special case, the description of the convex hull should be also further studied. For the second special case, the strengthening of the timing constraints should be further examined in order to improve the tightness of the constraints furthermore. For the third special case, an important research direction concerns the improvement of the model formulation for the coordination of the batch and continuous tasks. Some valid inequalities found so far do not help for solving larger instances. Therefore, some other valid inequalities must be found in order to improve the tightness of the formulation. A lot of work must still be done here since the introduction of the continuous tasks in the model formulation leads to a weak model formulation.

In Chapter 3, we test the improved model formulations for basic, multiple batch task and industrial cases. For the basic case, we solve various instances and we show that the corresponding improved formulations can solve these problem instances faster than the original continuous time formulations derived from Schilling and Pantelides in [44]. For the multiple batch task case, we have shown also that the corresponding improved formulation can solve the problem instances faster than the original formulation. However, the resolution of larger instances remains difficult. Therefore, we propose to use MIP based heuristic methods in order to obtain a good feasible solution quickly by taking advantage of the improved formulations. By using a combination of the two MIP based heuristic methods proposed, we have shown that we can get better feasible solutions than when using truncated branch-and-bound methods or other combinations of MIP based heuristic methods for the same computing time. For the multiple batch task case, we also compare the continuous time formulation proposed by Schilling and Pantelides in [44] with the improved formulation that we have proposed. By using the improved formulation, we have shown that we can solve quicker such type of scheduling problems with less branch-and-bound nodes than when using the formulation of Schilling and Pantelides. Finally, we study an industrial case. We first prove that we can bound the maximal improvement of productivity that can be obtained by the optimal solution at each iteration of the linearization of the nonlinear objective function, and that this bound on the maximal improvement per iteration is monotonically non increasing over the iterations. Then, we show that it is not possible to solve realistic industrial cases using an exact branch-and-bound algorithm in a reasonable amount of time. Therefore, we solved the industrial case using the initial and the strengthened formulations by truncated branch and bound, and also by a combination of MIP based heuristic methods. We got quicker and better feasible solutions by using the combination of MIP based heuristic methods, showing that such heuristic methods seem important for large instances.

A first research direction concerns the implementation of an efficient cutting plane strategy for the basic, the multiple batch task and the industrial cases in order to fully take advantage of all the valid inequalities found so far and reduce furthermore the CPU solution time. Another research direction concerns other combinations of MIP based heuristic methods and also other parameter usages for the various combination of MIP based heuristic methods in order to quicker obtain good feasible solutions for larger size instances.

In Chapter 4, we study scheduling problems where the processes involved can be modeled dynamically with differential equations in order to obtain a more flexible model, and possibly to increase productivity. The differential equations express the dynamics of the processes but, in general, explicit solutions of differential equations are not available. Therefore, in this Chapter, four methods are proposed and investigated in order to compute numerical solutions of the differential equations within the resolution of the scheduling optimization problem, namely two non linear methods (the collocation and the trapezoidal methods) and two piecewise linear approximation methods (the first and the second piecewise linear approximation methods).

The first problem studied is composed of one heating task that has to be performed on one reactor. The behaviour of the system during the heating task is represented by two differential equations, namely one for the concentration of the reactant and one for the temperature. The objective of this simple scheduling problem is to minimize the time needed for reaching the temperature 450.15 K, given the initial temperature of 298.15 K. The two non linear methods, the collocation and the trapezoidal methods, are non linear method that approximate closely the dynamics of the system but that lead to non linear mixed integer optimization problem for which at best a local optimal solution can be found. The two other methods approximate the solution of the differential equations by using piecewise linear approximations of the functions involved in the dynamics. We show that the second piecewise linear approximation method, based on the discretization of the space of state and command variables into simplices and on the discretization of time, gives a good feasible solution with the best compromise between quality of the approximation of the solution of the differential equations and CPU solution time. The trapezoidal method is the second best method and approximates the dynamics of the system quite accurately but the CPU solution time is larger.

The second problem studied is composed of one heating task that has to be performed on two reactors. The resource that is shared among the two reactors is the hot water. The objective is to minimize the time needed for both reactors for reaching the temperature 400 K and the initial temperature is 298.15 K. We test the quality and the efficiency of the trapezoidal and of the second piecewise linear approximation methods on this test case. The results have shown that the second piecewise linear approximation method provides quicker a better optimal objective value with an approximation of the process dynamics comparable to the one given by the trapezoidal method. The main difficulty for this second piecewise linear approximation method is that an appropriate discretization of the space of the state and command variables into simplices and of the time is essential in order to quickly obtain an optimal solution with an accurate approximation of the process dynamics.

A first future research direction concerns the discretization of the time for the trapezoidal method and the discretization of the time and of the space for the state and command variables for the second piecewise linear approximation method. A systematic way for choosing an appropriate discretization should be found for the two methods in order to have a good compromise between the CPU solution time of the optimization problem and the approximation of the process dynamics.

Moreover, an efficient cutting plane strategy for the second piecewise linear approximation method should be implemented in order to reduce furthermore the CPU solution time.

A second future research direction concerns the resolution of more general scheduling problems composed of a general hybrid system with states and transitions. In each state, we have various differential and algebraic equations and the transitions between two states occur when some logical conditions are satisfied. The idea is to use the second piecewise linear approximation method in order to solve such more general problems.

Finally a more general extension concerns the fact that the best cyclic schedule calculated here is independent of the initial state of the production lines. In order to be able to apply a good cyclic schedule for the plant processes, we need a transient schedule that brings the system from some initial state (possibly bad in terms of long term productivity) to a state from which a cyclic schedule with good productivity can be applied. Our model formulation should be adapted to contain a transient schedule before the cyclic one.

Bibliography

[1] K. Andersen and Y. Pochet. Coefficient strengthening : a tool for formulating mixed integer programs. *CORE Discussion Paper*, (2007/24), 2007.

[2] M.P. Avraam, N. Shah, and C.C. Pantelides. Modelling and optimisation of general hybrid systems in the continuous time domain. *Computers and Chemical Engineering*, 22:S221–S228, 1998.

[3] G. Bader and U. Ascher. A new basis implementation for a mixed order boundary value ode solver. *Siam journal on scientific and statistical computing*, 8(4):483–500, 1987.

[4] P. Baptiste, C. Le Pape, and W. Nuijten. *Constraint-Based Scheduling : Applying Constraints Programming to Scheduling Problems.* Kluwer Academics Publishers, London, 2001.

[5] C.B. Barber, D.P. Dobkin, and H.T. Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, 22(4):469–483, 1996.

[6] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35:407–427, 1999.

[7] L. D. Berkovitz. *Optimal Control Theory.* SPRINGER-VERLAG, New York, 1974.

[8] T. Bhatia and L.T. Biegler. Dynamic optimization in the design and scheduling of multiproduct batch plants. *Ind. Eng. Chem. Res.*, 35:2234–2246, 1996.

[9] L.T. Biegler, A.M. Cervantes, and A. Wachter. Advances in simultaneous strategies for dynamic process optimization. *Chemical Engineering Science*, 57:575–593, 2002.

[10] L.T. Biegler and I.E. Grossmann. Retrospective on optimization. *Computers and Chemical Engineering*, 28:1169–1192, 2004.

[11] P.T. Boggs and J.W. Tolle. Sequential quadratic programming. *Acta Numerica*, 4:1–51, 1996.

[12] P.M. Castro, A.P. Barbosa-Povoa, and H.A. Matos. Optimal periodic scheduling of batch plants using RTN-based discrete and continuous-time formulations : a case study approach. *Ind. Eng. Chem. Res.*, 42:3346–3360, 2003.

[13] A. Cervantes and L.T. Biegler. Optimization strategies for dynamic systems. *In C. Floudas, P. Pardalos (Eds.), Encyclopedia of Optimization*, 1999.

[14] A. Cervantes and L.T. Biegler. A stable elemental decomposition for dynamic process optimization. *Journal of Computational and Applied Mathematics*, 120:41–57, 2000.

[15] M.S. Charalambides, N. Shah, and C.C. Pantelides. Synthesis of batch reaction/distillation processes using detailed dynamic models. *Computers and Chemical Engineering*, 19(Suppl.):S167–S174, 1995.

[16] T. Christof and A. Loebel. Porta - a polyhedron representation transformation algorithm. *available via http://www.zib.de/Optimization/Software/Porta/*, 1997.

[17] J.E. Cuthrell and L.T. Biegler. On the optimization of differential-algebraic process systems. *AIChE Journal*, 33:1257–1270, 1987.

[18] E. Danna, E. Rothberg, and C. Le Pape. Exploring relaxation induced neighborhoods to improve mip solutions. *Mathematical Programming*, 102:71–90, 2005.

[19] W. Dinkelbach. On nonlinear fractional programming. *Management Science*, 13:492–498, 1967.

[20] M. Fischetti and A. Lodi. Local branching. *Mathematical Programming*, 98:23–48, 2003.

[21] C.A. Floudas and X. Lin. Mixed integer linear programming in process scheduling : modeling, algorithms, and applications. *Annals of Operations Research*, 139:131–162, 2005.

[22] M.G. Ierapetritou and C.A. Floudas. Effective continuous-time formulation for short-term scheduling. 1. multipurpose batch processes. *Ind. Eng. Chem. Res.*, 37:4341–4359, 1998.

[23] J.R. Isbell and W.H. Marlow. Attrition games. *Naval Research Logistics Quarterly*, 3:71–93, 1956.

[24] I.A. Karimi and C.M. McDonald. Planning and scheduling of parallel semicontinuous processes. 2. short-term scheduling. *Ind. Eng. Chem. Res.*, 36:2701–2714, 1997.

[25] E. Kondili, C.C. Pantelides, and R.W.H. Sargent. A general algorithm for short-term scheduling of batch operations - I. MILP formulation. *Computers chem. Engng*, 17:211–227, 1993.

[26] J. Lee and D. Wilson. Polyhedral methods for piecewise-linear functions. i : the lambda method. *Discrete applied mathematics*, 108(3):269–285, 2001.

[27] C.T. Maravelias. A decomposition framework for the scheduling of single- and multi-stage processes. *Computers chem. Engng*, 30:407–420, 2006.

[28] C.T. Maravelias and I.E. Grossmann. A hybrid MILP/CP decomposition approach for the continuous time scheduling of multipurpose batch plants. *Computers chem. Engng*, 28:1921–1949, 2004.

[29] N. Megiddo. Combinatorial optimization with rational objective functions. *Mathematics of Operations Research*, 4:414–424, 1979.

[30] C.A. Mendez, J. Cerda, I.E. Grossmann, I. Harjunkoski, and M. Fahl. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers chem. Engng*, 30:913–946, 2006.

[31] B.V. Mishra, E. Mayer, J. Raisch, and A. Kienle. Short-term scheduling of batch processes : a comparative study of different approaches. *Ind. Eng. Chem. Res.*, 44:4022–4034, 2005.

[32] L. Mockus and G.V. Reklaitis. Mathematical programing formulation for scheduling of batch operations based on nonuniform time discretization. *Computers chem. Engng*, 21:1147–1156, 1997.

[33] G.L Nemhauser and L.A Wolsey. *Integer and combinatorial optimization.* John Wiley & Sons, 1988.

[34] R.H. Nystrom, R. Franke, I Harjunkoski, and A. Kroll. Production campaign planning including grade transition sequencing and dynamic optimization. *Computers and Chemical Engineering*, 29:2163–2179, 2005.

[35] C.C. Pantelides. Unified frameworks for the optimal process planning and scheduling. *Proceedings on the second conference on foundations of computer aided operations*, pages 253–274, 1994.

[36] J.M. Pinto and I.E. Grossmann. A continuous time mixed integer linear programming model for short term scheduling of multistage batch plants. *Ind. Eng. Chem. Res.*, 34:3037–3051, 1995.

[37] Y. Pochet and F. Warichet. A tighter continuous time formulation for the cyclic scheduling of a mixed plant. *Computers and Chemical Engineering*, 2007. (Accepted for Publication).

[38] Y. Pochet and L.A. Wolsey. *Production planning by mixed-integer programming.* Springer, 2006.

[39] V. Pontryagin, V. Boltyanskii, R. Gamkrelidge, and E. Mishchenko. *The mathematical theory of optimal processes.* New York : Interscience Publishers Inc., 1962.

[40] A. Prata, J. Oldenburg, A. Kroll, and W. Marquardt. Integrated scheduling and dynamic optimization of grade transitions for a continuous polymerization reactor. *Computers and Chemical Engineering*, 2007. (Accepted for publication).

[41] A. Ralston and P. Rabinowitz. *A first course in numerical analysis.* International Series in Pure and Applied Mathematics, 1978.

[42] C.R. Reeves. *Modern Heuristic Techniques for Combinatorial Problems.* Blackwell Scientific Publications, Oxford, 1993.

[43] G. Schilling and C.C. Pantelides. A simple continuous-time process scheduling formulation and a novel solution algorithm. *Computers chem. Engng*, 20:S1221–S1226, 1996.

[44] G. Schilling and C.C. Pantelides. Optimal periodic scheduling of multipurpose plants. *Computers chem. Engng*, 23:635–655, 1999.

[45] N. Shah, C.C. Pantelides, and R.W.H. Sargent. Optimal periodic scheduling of multipurpose batch plants. *Annals of Operations Research*, 42:193–228, 1993.

[46] H. Stadtler. Multilevel lot sizing with setup times and multiple constrained resources: Internally rolling schedules with lot-sizing windows. *Operations Research*, 51:487–502, 2003.

[47] S. Terrazas-Moreno, A. Flores-Tlacuahuac, and I.E. Grossmann. Simultaneous cyclic scheduling and optimal control of multi-grade polymerization reactors. Submitted for publication (2005).

[48] V. Thome. From finite differences to finite elements : A short history of numerical analysis of partial differential equations. *Journal of Computational and Applied Mathematics*, 128:1–54, 2001.

[49] L.A. Wolsey. *Integer programming.* John Wiley & Sons, New York, 1998.

[50] D. Wu and M. Ierapetritou. Cyclic short-term scheduling of multiproduct batch plants using continuous-time representation. *Computers chem. Engng*, 28:2271–2286, 2004.

[51] X. Zhang and R.W.H. Sargent. The optimal operation of mixed production facilities - a general formulation and some approaches for the solution. *Computers chem. Engng*, 20:897–904, 1996.