4-2012

Generalized Disjunctive Programming as a Systematic Modeling Framework to Derive Scheduling Formulations

Pedro M. Castro
National Energy and Geology Laboratory, Lisbon

Ignacio E. Grossmann
Carnegie Mellon University, grossmann@cmu.edu

Published In
Industrial and Engineering Chemistry Research, 51, 16, 5781-5792.

Generalized Disjunctive Programming as a Systematic Modeling Framework to Derive Scheduling Formulations

Pedro M. Castro,[*,†,‡] *and Ignacio E. Grossmann*[‡]

[†] Unidade Modelação e Optimização de Sistemas Energéticos, Laboratório Nacional de Energia e Geologia, 1649-038 Lisboa, Portugal

[‡] Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh PA 15213-3890, USA

Abstract

We propose generalized disjunctive programming models for the short-term scheduling problem of single stage batch plants with parallel units. Three different concepts of continuous-time representation are explored, immediate and general precedence, as well as multiple time grids. The GDP models are then reformulated using both big-M and convex hull reformulations, and the resulting mixed-integer linear programming models compared to the solution of a set of example problems. We show that two general precedence models from the literature can be derived using a big-M reformulation for a set of disjunctions and a convex hull reformulation for another. The best performer is, however, a multiple time grid model which can be derived from the convex hull reformulation followed by simple algebraic manipulations to eliminate the disaggregated variables and reduce the sets of constraints, thus leading to a more compact and efficient formulation.

[*] Corresponding author. Tel.: +351-210924643. E-mail: pedro.castro@lneg.pt.

1. Introduction

The last twenty years have seen a variety of scheduling models by the Process Systems Engineering Community, where major developments have been the unified frameworks for process representation[1-2]. The State and Resource-Task Networks provide a systematic way of converting the real plant entities (e.g. reactors, products, utilities, manpower) into virtual entities that can be used by a mathematical model. They have been proposed with discrete-time optimization models that like most scheduling models feature variables and constraints derived from intuition and by trial and error[18] so as to maximize computational performance. Thus, and in contrast to the derivation of the process model, no attempt has been made to unify the mathematical modeling part.

This aspect becomes particularly relevant when switching from a discrete to a continuous-time representation, which has more freedom from a modeling point of view. In fact, few different approaches have been proposed that can be classified as precedence[4-6], single[7-9] or multiple time grid based[10-18]. Some of the latter unit-specific approaches, when applied to the more general multipurpose production environment, have raised serious concerns over the years with respect to the generality, suggesting that: (i) either the underlying time representation concept does not account for all possibilities, or (ii) not all the required variables or constraints are effectively part of the model. It is thus highly desirable to have a systematic modeling framework that starting with a somewhat simple concept for time representation can generate the constraints that will make it work.

Generalized Disjunctive Programming[19] provides a high level framework for modeling mixed-integer programs starting from a logic representation in which mixed-integer logic is represented through disjunctions and integer logic through propositions. In fact, different formulations can be derived using for example big-M[20] and convex hull reformulations[21], which have complementary strengths. The former has the advantage of being simpler and does not require the definition of additional continuous variables and constraints, but it may compromise computational performance. On the other hand, the convex hull reformulation is tighter, which generally helps to speed up the search procedure. GDP formulations have primarily been proposed for process networks[19,23-25] problems where disjunctions reflect the options of choosing a unit, which enforces mass balances

and is associated to a certain cost, or not. Other examples include strip-packing[26] and multistage scheduling problems featuring a single unit per stage and the general precedence concept[19,26].

In this paper, we propose to examine the relatively simple case of single stage scheduling problems with single and parallel units focusing on continuous-time representations and specifically on the immediate, general precedence and multiple time grids alternatives. Starting with the single unit case, and next for the case of parallel units, GDP models are formulated featuring linear constraints together with logical expressions involving the Boolean variables of the disjunctions. Based on these GDP models, we derive the equivalent mixed-integer linear programming models using the big-M and convex hull reformulations. The main goal will be to show that well-known models from the literature can be traced back to a particular elementary time representation/reformulation pair. Their performance is evaluated on a set of test problems with up to 30 orders and 5 parallel units.

2. Fundamentals

We consider the following generalized disjunctive program[19,22-23]:

$$\begin{aligned} &\min\ f(x) \\ &s.t.\ g(x) \le 0 \\ &\veebar_{j \in D_k} \begin{bmatrix} Y_{j,k} \\ A_{j,k} x \le b_{j,k} \end{bmatrix} \forall k \in K \\ &\Omega(Y) = True \\ &x \in \Re^n, x \ge 0, Y_{j,k} \in \{True, False\}^m \end{aligned} \qquad \text{(GDP)}$$

where $f(x)$ and $g(x)$ are linear functions, and discrete choices are expressed with Boolean variables $Y_{j,k}$ in terms of disjunctions and logic propositions $\Omega(Y)$. It is assumed that the disjunctive set K is proper in that the intersection over $j \in D_k$ of the feasible regions defined by the set of points $x : A_{j,k} x \le b_{j,k}$ is empty. Hence the use of the exclusive OR operator ($\veebar$) separating the disjunctions.

Disjunctions can be transformed into mixed-integer linear form through big-M[20], Beaumont surrogate[27] and convex hull[21] reformulations. Given the fact that for linear constraints, the big-M and Beaumont relaxations are equivalent[27], the latter is not considered in this work.

2.1. Big-M reformulation

The simplest representation of the disjunctions $k \in K$ in mixed-integer linear form are the big-M constraints given in (BM). Notice that binary variables $y_{j,k}$ have a one-to-one correspondence with Boolean variables $Y_{j,k}$ and that the last expression gives the tightest value for parameters $M_{j,k}$, where x^L and x^U are the lower and upper bounds of variable x, respectively. It is well-known that this set of constraints often yields poor relaxations.

$$\begin{aligned} & A_{j,k} x \le b_{j,k} + M_{j,k} \cdot (1 - y_{j,k}) \; \forall k \in K, \, j \in D_k \\ & \sum_{j \in D_k} y_{j,k} = 1 \; \forall k \in K \\ & y_{j,k} = \{0,1\} \\ & m^i_{j,k} = \max \{ a^i_{j,k} x - b^i_{j,k} : x^L \le x \le x^U \} \; \forall k \in K, \, j \in D_k \end{aligned} \tag{BM}$$

where $m^i_{j,k}$ is the i^{th} row of $M_{j,k}$ and $a^i_{j,k}$, $b^i_{j,k}$ are the i^{th} row entries of $A_{j,k}$ and $b_{j,k}$.

2.2. Convex hull reformulation

The convex hull relaxation[21] (CH) has the advantage of being at least as tight as the big-M relaxation[23], thus helping to reduce the search effort in the branch-and-bound procedure. The drawback is that it increases the number of continuous variables and constraints of the original problem, which can make a problem more expensive to solve, especially in larger problems.

$$\begin{aligned} & x = \sum_{j \in D_k} \hat{x}_{j,k} \; \forall k \in K \\ & A_{j,k} \hat{x}_{j,k} \le b_{j,k} y_{j,k} \; \forall k \in K, \, j \in D_k \\ & \hat{x}^L_{j,k} y_{j,k} \le \hat{x}_{j,k} \le \hat{x}^U_{j,k} y_{j,k} \; \forall k \in K, \, j \in D_k \\ & \sum_{j \in D_k} y_{j,k} = 1 \; \forall k \in K \\ & \hat{x}_{j,k} \ge 0, \, y_{j,k} = \{0,1\} \end{aligned} \tag{CH}$$

The critical step is to identify the set of disaggregated variables $\hat{x}_{j,k}$ to employ. The sum of the disaggregated variables over the set of disjunctions D_k needs to be equal to the original variables x. Two sets of constraints are then used to relate the disaggregated variables with the binary variables $y_{j,k}$. While the second reflects the original constraint in the disjunction, the third ensures that the new set of disaggregated variables is different than zero only if the corresponding binary variable is one. The bounds $\hat{x}^L_{j,k}$ and $\hat{x}^U_{j,k}$ feature indices j and k to emphasize that they can change between

disjunctions. Furthermore, they will typically be tighter than the bounds of the original variables x (x^L and x^U).

2.3. Logic Propositions

One way to derive constraints involving 0-1 variables is to first consider the logic expressions of the model, and then transform them into an equivalent equation or inequality with 0-1 variables[28-29]. Since the Boolean variable Y_i is associated to a selection or action in the logic expressions, a binary variable y_i that has a one-to-one correspondence can be assigned to it. Then, the negation of Y_i ($\neg Y_i$) is given by $1-y_i$. The logical value of *True* corresponds to the binary value of 1 and *False* corresponds to the binary variable of 0. The basic operators used in propositional logic and the representation of their relationships are shown in Table 1[28].

Table 1. Constraint Representation of Logic Propositions and Operators

Logical relation	Comments	Boolean Expression	Representation as Linear Inequalities
Logical OR		$Y_1 \vee Y_2 \vee ... \vee Y_n$	$y_1+y_2+...+y_n \geq 1$
Logical AND		$Y_1 \wedge Y_2 \wedge ... \wedge Y_n$	$y_1=1$ $y_2=1$ ... $y_n=1$
Implication	$Y_1 \Rightarrow Y_2$	$\neg Y_1 \vee Y_2$	$1-y_1+y_2 \geq 1$
Equivalence	Y_1 if and only if Y_2 ($Y_1 \Rightarrow Y_2 \wedge Y_2 \Rightarrow Y_1$)	$(\neg Y_1 \vee Y_2) \wedge (\neg Y_2 \vee Y_1)$	$y_1=y_2$
Exclusive OR	Exactly one of the variables is true	$Y_1 \veebar Y_1 \veebar ... \veebar Y_n$	$y_1+y_2+...+y_n=1$

With the relations in Table 1, one can systematically model an arbitrary propositional logic expression that is given in terms of the different operators, as a set of linear equality and inequality constraints. One approach is to convert step by step the logical expression into its equivalent conjunctive normal form representation[29]. The conjunctive normal form is a conjunction of clauses, $Q_1 \wedge Q_2 \wedge \ldots \wedge Q_s$ (i.e. connected by AND operators $\wedge$). Hence, for the conjunctive normal form to be true, each clause Q_i must be true, independent of the others. Also, since a clause Q_i is a disjunction of non-negated or negated literals, $Y_1 \vee Y_2 \vee ... \vee Y_n$ (i.e., connected by OR operators $\vee$), it can be expressed as the linear inequality in the first row of Table 1.

More specifically, the procedure to convert a logical expression into its corresponding conjunctive normal form was formalized by Clocksin and Mellish[30]. It consists of applying recursively three steps. The first one involves the replacement of the implication by its equivalent disjunction, as shown in Table 1. Next, one needs to move the negation inward by applying De Morgan's Theorem:

$$\neg(Y_1 \wedge Y_2) \Leftrightarrow \neg Y_1 \vee \neg Y_2$$

$$\neg(Y_1 \vee Y_2) \Leftrightarrow \neg Y_1 \wedge \neg Y_2$$

Finally, the OR is distributed over the AND by using the following equivalence:

$$(Y_1 \wedge Y_2) \vee Y_3 \Leftrightarrow (Y_1 \vee Y_3) \wedge (Y_2 \vee Y_3)$$

3. Problem Definition

In this paper, we consider short-term scheduling problems for batch plants, starting from the simplest sequencing problem in a single unit and ending with multiple units in parallel for a single stage. Given are a set I of orders characterized by processing time p_i as well as release r_i and due times d_i, which are enforced as hard constraints. When in the presence of multiple units, the processing time becomes unit dependent and p_i is replaced by $p_{i,m}$. For simplicity in the presentation, we assume no changeover times. The objective function considered is makespan minimization.

$$\min MS \tag{1}$$

4. Single Unit Sequencing Problem

The single unit sequencing problem is the basic building block of any scheduling problem. Without the issue of sequence dependent changeovers, every sequence will lead to the same makespan if release and due times are neglected. However, enforcing such constraints will make many sequences infeasible and may also lead to idle times while waiting for an order to be released. The concept of sequence is present on the three different continuous-time models that have been proposed to handle such problem, either explicitly through the definition of sequencing variables, or implicitly, by considering a time grid consisting of time slots and assigning orders to such slots. We now look in detail to each of these possibilities, propose the generalized disjunctive programming models and derive their corresponding big-M and convex hull relaxations.

4.1. General precedence concept

The most widely used concept in sequencing variables based models is the one of general precedence. Given any two orders *i* and *i'* there are just two sequencing possibilities, either *i* before *i'* or *i'* before *i*, and both cannot occur simultaneously. This can be modeled with a proper disjunctive set featuring two disjunctions separated by the exclusive OR operator, see Figure 1, where Boolean variables $Y_{i,i'}$ indicate if order *i* is before *i'*. Notice that the number of Boolean variables can be reduced without loss of generality by making $i<i'$.

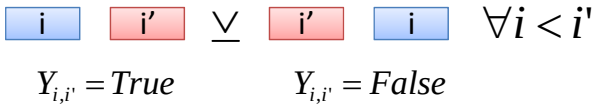

Figure 1. General precedence concept for single unit problem.

Defining x_i as the starting time of order *i*, then the condition inside the first disjunction is for the ending time of order *i* (x_i+ p_i) to be less than the starting time of order *i'*, see equation (2). The remaining sets of constraints enforce release and due times, equations (3-4), and define the makespan as the ending time of the last order, equation (5).

$$\begin{bmatrix} Y_{i,i'} \\ x_i + p_i \le x_{i'} \end{bmatrix} \underline{\vee} \begin{bmatrix} \neg Y_{i,i'} \\ x_{i'} + p_{i'} \le x_i \end{bmatrix} \forall i \in I, i' \in I, i < i' \tag{2}$$

$$x_i \ge r_i \ \forall i \in I \tag{3}$$

$$x_i + p_i \le d_i \ \forall i \in I \tag{4}$$

$$x_i + p_i \le MS \ \forall i \in I \tag{5}$$

4.1.1. Big-M reformulation

According to (BM), we need two sets of constraints for the big-M relaxation of equation (2), see equations (6-7). Note that the MILP equivalent of the negation of Boolean variable $Y_{i,i'}$ is 1-$y_{i,i'}$ and that the tightest big-M parameters are calculated through (8).

$$x_i + p_i \le x_{i'} + M_{i,i'} \cdot (1 - y_{i,i'}) \ \forall i \in I, i' \in I, i < i' \tag{6}$$

$$x_{i'} + p_{i'} \le x_i + M_{i',i} \cdot y_{i,i'} \ \forall i \in I, i' \in I, i < i' \tag{7}$$

$$M_{i,i'} = \max(x_i - x_{i'} + p_i) = \max(x_i + p_i) - \min(x_{i'}) = d_i - r_{i'} \ \forall i \in I, i' \in I, i \ne i' \tag{8}$$

4.1.2. Convex hull reformulation

In order to identify the required set of disaggregated variables, note that if we compare the disjunctions in equation (2) with those in (GDP), index $j \in \{1,2\}$ and set $K=\{(i,i'): i<i'\}$. We thus need two disaggregated variables for each of the variables x appearing in the disjunctions, see equation (9-10). Then, equations (11-12) reflect the constraints inside the disjunctions.

$$x_i = \hat{x}^1_{i,i,i'} + \hat{x}^2_{i,i,i'} \ \forall i \in I, i' \in I, i < i' \tag{9}$$

$$x_{i'} = \hat{x}^1_{i',i,i'} + \hat{x}^2_{i',i,i'} \ \forall i \in I, i' \in I, i < i' \tag{10}$$

$$\hat{x}^1_{i',i,i'} - \hat{x}^1_{i,i,i'} \geq p_i \cdot y_{i,i'} \ \forall i \in I, i' \in I, i < i' \tag{11}$$

$$\hat{x}^2_{i,i,i'} - \hat{x}^2_{i',i,i'} \geq p_{i'} \cdot (1 - y_{i,i'}) \ \forall i \in I, i' \in I, i < i' \tag{12}$$

The next step is to determine the lower and upper bounds of the disaggregated variables. Considering variable $\hat{x}^1_{i,i,i'}$ as an example, its lower bound is the same as that of the original variable x_i: r_i. However, for the upper bound, the knowledge that order i is before order i' leads to a tighter bound whenever $d_{i'}$-$p_{i'}$<d_i, see equation (13). Similar insights can be used to get to the bounds of the remaining variables appearing in equations (14-16).

$$r_i \cdot y_{i,i'} \leq \hat{x}^1_{i,i,i'} \leq \min(d_i - p_i, d_{i'} - p_{i'} - p_i) \cdot y_{i,i'} \ \forall i \in I, i' \in I, i < i' \tag{13}$$

$$\max(r_i, r_{i'} + p_{i'}) \cdot (1 - y_{i,i'}) \leq \hat{x}^2_{i,i,i'} \leq (d_i - p_i) \cdot (1 - y_{i,i'}) \ \forall i \in I, i' \in I, i < i' \tag{14}$$

$$\max(r_{i'}, r_i + p_i) \cdot y_{i,i'} \leq \hat{x}^1_{i',i,i'} \leq (d_{i'} - p_{i'}) \cdot y_{i,i'} \ \forall i \in I, i' \in I, i < i' \tag{15}$$

$$r_{i'} \cdot (1 - y_{i,i'}) \leq \hat{x}^2_{i',i,i'} \leq \min(d_{i'} - p_{i'}, d_i - p_i - p_{i'}) \cdot (1 - y_{i,i'}) \ \forall i \in I, i' \in I, i < i' \tag{16}$$

4.2. Immediate precedence concept

The alternative to general precedence is immediate precedence, which can be quite useful to account for sequence dependent costs in the objective function. One explicitly identifies the orders immediately preceding and following order i, provided that this is neither the first nor the last order in the sequence. To accurately model the problem, four sets of disjunctions are required as well as additional logic constraints. The first is illustrated in Figure 2, and states that a particular order i is either followed by another order or is the last one in the sequence, see equation (17). The second set

of disjunctions, states that order *i* is either preceded by another order or is the first one in the sequence, equation (18). Then, there can only be one first (19) and one last order (20) and it cannot be the same. The latter constraint corresponds to the logic expression in (21). The conditions inside the disjunctions are shared with the general precedence model, together with equations (3-5). Notice however, that if order *i* is the first or the last in the sequence, the conditions are written for every order $i' \neq i$.

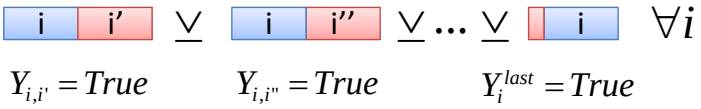

Figure 2. Immediate precedence concept for single unit problem.

$$\underset{i' \neq i}{\vee}\left[\begin{array}{c} Y_{i,i'} \\ x_i + p_i \leq x_{i'} \end{array}\right] \underline{\vee} \left[\begin{array}{c} Y_i^{last} \\ x_{i'} + p_{i'} \leq x_i \; \forall i' \neq i \end{array}\right] \forall i \in I \tag{17}$$

$$\underset{i' \neq i}{\vee}\left[\begin{array}{c} Y_{i',i} \\ x_{i'} + p_{i'} \leq x_i \end{array}\right] \underline{\vee} \left[\begin{array}{c} Y_i^{first} \\ x_i + p_i \leq x_{i'} \; \forall i' \neq i \end{array}\right] \forall i \in I \tag{18}$$

$$\underset{i \in I}{\vee}\left[\begin{array}{c} Y_i^{first} \\ x_i + p_i \leq x_{i'} \; \forall i' \neq i \end{array}\right] \tag{19}$$

$$\underset{i \in I}{\vee}\left[\begin{array}{c} Y_i^{last} \\ x_{i'} + p_{i'} \leq x_i \; \forall i' \neq i \end{array}\right] \tag{20}$$

$$\neg(Y_i^{first} \wedge Y_i^{last}) \; \forall i \in I \tag{21}$$

4.2.1. Big-M reformulation

Three sets of big-M constraints result (note that if the indices are changed in the condition for $Y_{i,i'}$ we get the condition for $Y_{i',i}$), see eqs (22-25) where the $M_{i,i'}$ parameters are also calculated by (8).

$$x_i + p_i \leq x_{i'} + M_{i,i'} \cdot (1 - y_{i,i'}) \; \forall i \in I, i' \in I, i \neq i' \tag{22}$$

$$x_{i'} + p_{i'} \leq x_i + M_{i',i} \cdot (1 - y_i^{last}) \; \forall i \in I, i' \in I, i \neq i' \tag{23}$$

$$x_i + p_i \leq x_{i'} + M_{i,i'} \cdot (1 - y_i^{first}) \; \forall i \in I, i' \in I, i \neq i' \tag{24}$$

The Boolean variables involved in the four sets of disjunctions are converted to binary variables and give rise to equations (25-28). Recall that the disjunctions of a set are mutually exclusive and hence the equality to one.

$$\sum_{i'\neq i} y_{i,i'} + y_i^{last} = 1 \; \forall i \in I \tag{25}$$

$$y_i^{first} + \sum_{i'\neq i} y_{i',i} = 1 \; \forall i \in I \tag{26}$$

$$\sum_{i\in I} y_i^{last} = 1 \tag{27}$$

$$\sum_{i\in I} y_i^{first} = 1 \tag{28}$$

As for the logic expression in (21), according to De Morgan's Theorem and the second row in Table 1, we get equation (29).

$$1 - y_i^{first} + 1 - y_i^{last} \geq 1 \Leftrightarrow y_i^{first} + y_i^{last} \leq 1 \; \forall i \in I \tag{29}$$

4.2.2. Convex hull reformulation

Finding the correct set of disaggregated variables to use is particularly tricky for the immediate precedence case. We know that variables x_i appear in constraints related to $Y_{i,i'}$, $Y_{i',i}$, Y_i^{first}, $Y_{i'}^{first}$, Y_i^{last}, $Y_{i'}^{last}$. Thus, we need to define variables $\hat{x}_{i,i,i'}$ and $\hat{x}_{i,i',i}$, $\forall$ $i\in I$, $i'\in I$, $i\neq i'$, which read disaggregated variable associated to order i and disjunction (i,i') or (i',i), respectively. In addition we need variables $\hat{x}_{i,i'}^{first}$ and $\hat{x}_{i,i'}^{last}$, $\forall$ $i\in I$, $i'\in I$, meaning disaggregated variable linked to order i participating in the disjunction of the first/last order i'. We also know that there are 4 sets of disjunctions (17-20), so there must be 4 sets of constraints relating the original variables with the disaggregated variables. Taking the first set of disjunctions as an example, it is either i before an i' or i is the last order, so we get equation (30). Similarly for the second set of disjunctions, see equation (31).

$$x_i = \sum_{i'\neq i} \hat{x}_{i,i,i'} + \hat{x}_{i,i}^{last} \; \forall i \tag{30}$$

$$x_i = \hat{x}_{i,i}^{first} + \sum_{i'\neq i} \hat{x}_{i,i',i} \; \forall i \tag{31}$$

For generating the remaining two sets of constraints linking the disaggregated variables to the original variables, note that we have to ensure that the disaggregated variable of a given order can only be different than zero in its own last order disjunction or in the last disjunction of an order *i*', see equation (32). Likewise for the first order disjunction, see equation (33).

$$x_i = \sum_{i' \in I} \hat{x}_{i,i'}^{last} \ \forall i \in I \tag{32}$$

$$x_i = \sum_{i' \in I} \hat{x}_{i,i'}^{first} \ \forall i \in I \tag{33}$$

The constraints inside the disjunctions (17-18), featuring the disaggregated rather than the original variables, are given in equations (34-36). Note that disjunctions (19-20) also lead to constraints (35-36).

$$\hat{x}_{i',i,i'} - \hat{x}_{i,i,i'} \geq p_i \cdot y_{i,i'} \ \forall i \in I, i' \in I, i \neq i' \tag{34}$$

$$\hat{x}_{i,i}^{last} - \hat{x}_{i',i}^{last} \geq p_{i'} \cdot y_i^{last} \ \forall i \in I, i' \in I, i \neq i' \tag{35}$$

$$\hat{x}_{i',i}^{first} - \hat{x}_{i,i}^{first} \geq p_i \cdot y_i^{first} \ \forall i \in I, i' \in I, i \neq i' \tag{36}$$

As for the relation between the disaggregation variables and the binary variables, the lower and upper bounds of variables $\hat{x}_{i,i,i'}$ and $\hat{x}_{i,i',i}$ are the ones already given in equations (13-16). In equation (37) we simply use another index to reduce the number of constraints. Similar bounds are used for variables $\hat{x}_{i,i'}^{first}$ and $\hat{x}_{i,i'}^{last}$, the difference being that in the lower bound of equation (38) and in the upper bound of equation (39) we ensure that enough time is given for all orders *i''*≠*i* finish/start before the start/end of order *i* if this is in the fact the last/first order in the sequence.

$$\begin{aligned} &[\max(r_{i'}, r_i + p_i)\big|_{i''=i'} + r_{i''}\big|_{i''\neq i'}] \cdot y_{i,i'} \leq \hat{x}_{i'',i,i'} \leq [\min(d_i - p_i, d_{i'} - p_{i'} - p_i)\big|_{i''=i} + (d_{i''} - p_{i''})\big|_{i''\neq i}] \cdot y_{i,i'} \\ &\forall i \in I, i' \in I, i'' \in I, i \neq i', (i'' = i \vee i'' = i') \end{aligned} \tag{37}$$

$$\begin{aligned} &(\max[r_i, \max_{i''\neq i'}(r_{i''} + p_{i''})]\Big|_{i''=i} + r_{i'}\big|_{i'\neq i}) \cdot y_i^{last} \leq \hat{x}_{i',i}^{last} \leq \\ &[\min(d_{i'} - p_{i'}, d_i - p_i - p_{i'})\big|_{i'\neq i} + (d_i - p_i)\big|_{i'=i}] \cdot y_i^{last} \ \forall i \in I, i' \in I \end{aligned} \tag{38}$$

$$\begin{aligned} &(\max[r_{i'}, r_i + p_i]\big|_{i'\neq i} + r_i\big|_{i'=i}) \cdot y_i^{first} \leq \hat{x}_{i',i}^{first} \leq \\ &(\min[d_i - p_i, \min_{i''\neq i}(d_{i''} - p_{i''} - p_i)]\Big|_{i'=i} + (d_{i'} - p_{i'})\big|_{i'\neq i}) \cdot y_i^{first} \ \forall i \in I, i' \in I \end{aligned} \tag{39}$$

The convex hull reformulation shares with its big-M counterpart equations (25-29).

4.3. Time slots concept

Rather than relying on sequencing variables to locate orders with respect to each other, one can define a grid consisting of $|T|=|I|$ time slots and assign orders to slots. Then, each slot t may only be occupied by an order, see Figure 3. Being x_t the starting time of slot t, then the difference between x_{t+1} (MS if $t=|T|$) and x_t must be greater than the processing time of order i if $Y_{i,t}=True$. Note that the equality is not enforced so that, if necessary, we can wait for the release time of the following order. In addition, there are two extra constraints associated to the disjunction, which ensure that the release and due time constraints are met, see equation (40). The model is complete with an extra set of disjunctions that reflect the fact that every order must be assigned to one slot, equation (41). Note that the constraints associated to such disjunctions are the ones in (40).

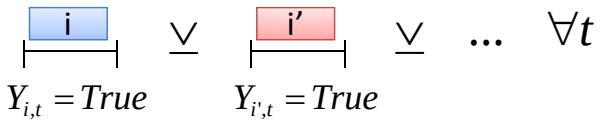

Figure 3. Time slots concept for single unit problem.

$$\bigvee_i \begin{bmatrix} Y_{i,t} \\ MS\big|_{t=|T|} + x_{t+1}\big|_{t\neq|T|} - x_t \geq p_i \\ x_t \geq r_i \\ x_t + p_i \leq d_i \end{bmatrix} \forall t \in T \tag{40}$$

$$\bigvee_t Y_{i,t} \ \forall i \in I \tag{41}$$

4.3.1. Big-M reformulation

According to (BM), the big-M reformulation for the constraints inside the disjunction is given in equations (42-44). Notice that big-M parameters for a given i are p_i, r_i and $\max_{i'}(d_{i'} - p_{i'}) + p_i - d_i$, respectively. Because the firsts are the parameters in the original constraints, equations (42-43) are not big-M constraints. This is the consequence of considering the tightest, equation dependent, big-M parameters, instead of an overall larger value.

$$MS\big|_{t=|T|} + x_{t+1}\big|_{t\neq|T|} - x_t \geq p_i \cdot y_{i,t} \ \forall i \in I, t \in T \tag{42}$$

$$x_t \geq r_i \cdot y_{i,t} \ \forall i \in I, t \in T \tag{43}$$

$$x_t \le (d_i - p_i)\cdot y_{i,t} + \max_{i'}(d_{i'} - p_{i'})\cdot(1 - y_{i,t})\ \forall i \in I, t \in T \quad (44)$$

The constraints representing the exclusive OR in (40-41) over the binary variables are then shown in equations (45-46).

$$\sum_{i\in I} y_{i,t} = 1\ \forall t \in T \quad (45)$$

$$\sum_{t\in T} y_{i,t} = 1\ \forall i \in I \quad (46)$$

4.3.2. Convex hull reformulation

There are three variables appearing in the constraints associated to disjunction $Y_{i,t}$ in equation (40), and so we need to define the following disaggregated variables: $\widehat{MS}_i$, $\widehat{x}_{t,i,t}$ and $\widehat{x}_{t+1,i,t}$. Note that the first does not need the time index t since it only shows up for $t=|T|$. The relation between the disaggregated variables and original variables is given in equations (47-49).

$$MS = \sum_{i\in I} \widehat{MS}_i \quad (47)$$

$$x_t = \sum_{i\in I} \widehat{x}_{t,i,t}\ \forall t \in T \quad (48)$$

$$x_{t+1} = \sum_{i\in I} \widehat{x}_{t+1,i,t}\ \forall t \in T, t \ne |T| \quad (49)$$

Equations (50-52) reflect the constraints associated to the disjunction in (40). Notice that the latter sets also act as the lower and upper bounds of variables $\widehat{x}_{t,i,t}$, respectively. As for the relation of the other two sets of disaggregated variables with binaries $y_{i,t}$, equations (53-54) result. In the latter, we have a very tight lower bound for the disaggregated makespan variable, which is equal to the sum of all processing times plus the minimum release time over all orders. The MILP model is complete with equations (45-46).

$$\widehat{MS}_i\Big|_{t=|T|} + \widehat{x}_{t+1,i,t}\Big|_{t\ne|T|} - \widehat{x}_{t,i,t} \ge p_i \cdot y_{i,t}\ \forall i \in I, t \in T \quad (50)$$

$$\widehat{x}_{t,i,t} \ge r_i \cdot y_{i,t}\ \forall i \in I, t \in T \quad (51)$$

$$\widehat{x}_{t,i,t} \le (d_i - p_i)\cdot y_{i,t}\ \forall i \in I, t \in T \quad (52)$$

$$(r_i + p_i)\cdot y_{i,t} \le \widehat{x}_{t+1,i,t} \le \max_{i'\in I}(d_{i'} - p_{i'})\cdot y_{i,t}\ \forall i \in I, t \in T \quad (53)$$

$$(\min_{i'\in I} r_{i'} + \sum_{i'\in I} p_{i'}) \cdot y_{i,t} \le \widehat{MS}_i \le \max_{i'\in I}(d_{i'}) \cdot y_{i,t} \;\; \forall i \in I, t \in T, t = |T| \tag{54}$$

4.3.3. Eliminating disaggregated variables in convex hull reformulation

As mentioned before, the main disadvantage of the convex hull reformulation over its big-M counterpart is the increased size resulting from the additional continuous variables and constraints. This is often responsible for an increase in the computational effort offsetting the eventual benefits from a tighter formulation. One should therefore check if it is possible to eliminate these extra variables to keep the benefit without the disadvantage. This is fortunately the case with the time slots convex hull formulation.

If for a given *t*, one sums over *i* constraints (50), and replaces the sum of the disaggregated variables with their corresponding original ones, according to equations (47-49), one obtains equation (55). Doing the same thing for equations (51-52), we obtain equations (56-57), which together with (45-46) conclude the much simpler and considerably more efficient formulation, as will be seen later on.

$$MS\big|_{t=|T|} + x_{t+1}\big|_{t\neq|T|} - x_t \ge \sum_{i\in I} p_i \cdot y_{i,t} \;\; \forall t \in T \tag{55}$$

$$x_t \ge \sum_{i\in I} r_i \cdot y_{i,t} \;\; \forall t \in T \tag{56}$$

$$x_t \le \sum_{i\in I} (d_i - p_i) \cdot y_{i,t} \;\; \forall t \in T \tag{57}$$

It should also be noted that the constraints in (55-57) are stronger than in (42-44). This can clearly be seen in constraints (55-56) where the right-hand sides are tighter than in (42-43).

5. Single Stage Problem with Parallel Units

If instead of a single unit there are multiple units in parallel the problem becomes more challenging since we no longer know a priori how many orders are to be allocated to a particular unit. This affects the concept of sequence, with constraints between any pair of orders to be enforced only if both are assigned to the unit, as well as that of time slots, since it is possible to have an insufficient number of slots, which will lead to an infeasible solution, or there may be slots in excess,

which should be handled appropriately in order to reduce solution degeneracy. We will now revisit the general precedence and time slots concepts and, starting with GDP models, discuss the reformulations required to generate some of the MILP models that have been published in the literature. For the general precedence concept there are actually two different alternatives as will be shown below.

5.1. General precedence concept (option 1)

The first option is to look at sequencing from a unit perspective, see Figure 4. Given any two orders (*i*,*i*') with *i*<*i*', the Boolean variable $Y^1_{i,i',m}$ is true if and only if *i* is sequenced before *i*' and both are assigned to unit *m*. Using the same sequencing variables of the single unit problem and a new set of Boolean assignment variables, $\overline{Y}_{i,m}$, yields equation (58). On the other hand, if the two orders are assigned to *m* but *i*' is before *i*, then it is $Y^2_{i,i',m}$ that is true, (59). The third possibility, which is not required for the follow up, is that the two orders are not both assigned to *m*.

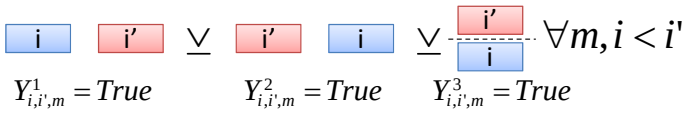

Figure 4. General precedence concept for single stage problem (option 1).

$$Y^1_{i,i',m} \Leftrightarrow Y_{i,i'} \wedge \overline{Y}_{i,m} \wedge \overline{Y}_{i',m} \;\; \forall i \in I, i' \in I, i < i', m \in M \tag{58}$$

$$Y^2_{i,i',m} \Leftrightarrow \neg Y_{i,i'} \wedge \overline{Y}_{i,m} \wedge \overline{Y}_{i',m} \;\; \forall i \in I, i' \in I, i < i', m \in M \tag{59}$$

It is also possible to avoid variables $Y_{i,i'}$ and replace equations (58-59) by (60), which states that if the two orders are assigned to unit *m* it is either *i* before *i*', or *i*' before *i*. However, such alternative leads to problems with significantly more binary variables and constraints than options 1 and 2 (see section 5.2), and failed to produce a better performance so it is not considered.

$$\overline{Y}_{i,m} \wedge \overline{Y}_{i',m} \Rightarrow Y^1_{i,i',m} \vee Y^2_{i,i',m} \;\; \forall i \in I, i' \in I, i < i', m \in M \tag{60}$$

The disjunctions are given in equation (61), where the processing time of order *i*, p_i, is now a variable instead of a parameter. The information given from the problem data is now the processing time of *i* in unit *m*, so $p_i = p_{i,m}$ if *i* is indeed assigned to *m*. Note that we could have used $p_{i,m}$ instead of

p_i in the disjunctions in equation (61), but this would prevent us from reaching our targeted constraint, which is tighter. Since we are still enforcing all orders to be assigned to a unit, another set of disjunctions is required, see equation (62).

$$\begin{bmatrix} Y^1_{i,i',m} \\ x_i + p_i \le x_{i'} \end{bmatrix} \veebar \begin{bmatrix} Y^2_{i,i',m} \\ x_{i'} + p_{i'} \le x_i \end{bmatrix} \veebar Y^3_{i,i',m} \;\; \forall i \in I, i' \in I, i < i', m \in M \tag{61}$$

$$\veebar_m \begin{bmatrix} \bar{Y}_{i,m} \\ p_i = p_{i,m} \end{bmatrix} \forall i \in I \tag{62}$$

The GDP model is completed with constraints (3-5).

5.1.1. MILP reformulation

The MILP reformulation of the GDP model just described gives rise to the general precedence model proposed Méndez et al.[4] if reduced to a single stage. Interestingly, one set of disjunctions is derived from the big-M reformulation, while the other results from the convex hull.

Starting with the first set of disjunctions, equation (61), it is straightforward to derive (63-64), where the *M* parameter is determined from equation (8).

$$x_i + p_i \le x_{i'} + M_{i,i'} \cdot (1 - y^1_{i,i',m}) \;\forall i \in I, i' \in I, i < i', m \in M \tag{63}$$

$$x_{i'} + p_{i'} \le x_i + M_{i',i} \cdot (1 - y^2_{i,i',m}) \;\forall i \in I, i' \in I, i < i', m \in M \tag{64}$$

As for the convex hull of the second disjunction, we need to first define a new set of disaggregated variables, $\hat{p}_{i,m}$, which are related to the known parameters $p_{i,m}$ and the remaining variables through equations (65-67). Clearly, one can add over *i* all constraints in (66) for a given *m*, and use (65) to derive an alternative and better set of constraints that avoids the need for the disaggregated variables, see equation (68).

$$p_i = \sum_{m \in M} \hat{p}_{i,m} \;\forall i \in I \tag{65}$$

$$\hat{p}_{i,m} = p_{i,m} \cdot \bar{y}_{i,m} \;\forall i \in I, m \in M \tag{66}$$

$$\sum_{m \in M} \bar{y}_{i,m} = 1 \;\forall i \in I \tag{67}$$

$$p_i = \sum_{m \in M} p_{i,m} \cdot \bar{y}_{i,m} \;\forall i \in I \tag{68}$$

The next step is to convert the logic in equations (58-59), into an MILP form, see equations (69-70).

$$\begin{aligned} &Y_{i,i'} \wedge \overline{Y}_{i,m} \wedge \overline{Y}_{i',m} \Rightarrow Y^1_{i,i',m} \Leftrightarrow \neg Y_{i,i'} \vee \neg \overline{Y}_{i,m} \vee \neg \overline{Y}_{i',m} \vee Y^1_{i,i',m} \Leftrightarrow \\ &1 - y_{i,i'} + 1 - \bar{y}_{i,m} + 1 - \bar{y}_{i',m} + y^1_{i,i',m} \geq 1 \Leftrightarrow \\ &y^1_{i,i',m} \geq y_{i,i'} + \bar{y}_{i,m} + \bar{y}_{i',m} - 2 \; \forall i \in I, i' \in I, i < i', m \in M \end{aligned} \tag{69}$$

$$\begin{aligned} &\neg Y_{i,i'} \wedge \overline{Y}_{i,m} \wedge \overline{Y}_{i',m} \Rightarrow Y^2_{i,i',m} \Leftrightarrow \\ &y^2_{i,i',m} \geq \bar{y}_{i',m} + \bar{y}_{i,m} - y_{i,i'} - 1 \; \forall i \in I, i' \in I, i < i', m \in M \end{aligned} \tag{70}$$

After replacing (68-70) in (4-5) and (63-64), this yields the final constraints (71-74). Note that (3) and (67) still apply.

$$x_i + \sum_{m' \in M} p_{i,m'} \cdot \bar{y}_{i,m'} \leq x_{i'} + M_{i,i'} \cdot (3 - y_{i,i'} - \bar{y}_{i,m} - \bar{y}_{i',m}) \; \forall i \in I, i' \in I, i < i', m \in M \tag{71}$$

$$x_{i'} + \sum_{m' \in M} p_{i',m'} \cdot \bar{y}_{i',m'} \leq x_i + M_{i',i} \cdot (2 + y_{i,i'} - \bar{y}_{i,m} - \bar{y}_{i',m}) \; \forall i \in I, i' \in I, i < i', m \in M \tag{72}$$

$$x_i + \sum_{m \in M} p_{i,m} \cdot \bar{y}_{i,m} \leq d_i \; \forall i \in I \tag{73}$$

$$x_i + \sum_{m \in M} p_{i,m} \cdot \bar{y}_{i,m} \leq MS \; \forall i \in I \tag{74}$$

5.2. General precedence concept (option 2)

The second option considers Boolean variables of type $Y_{i,i'}$ that are true if both *i* and *i'* are assigned to the same unit and *i* is sequenced before *i'*, see Figure 5. Notice that more variables are now required since even though the set of disjunctions is for *i*<*i'*, we now consider $Y_{i,i'}$ as well as $Y_{i',i}$. Hence, their actual domain is for every *i'*≠*i*. For the third disjunction ($Y^3_{i,i'}$) to be true, both orders cannot be allocated to the same unit. Notice that only the implication is true since due to the absence of the unit index, there may be an actual unit *m* handling units *i* and *i'*, for which the reverse implication would lead to $Y^3_{i,i'}$=False, which is correct, but for all other units *m'* the same implication would lead to an inconsistent outcome $Y^3_{i,i'}$=True.

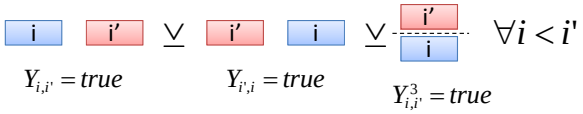

Figure 5. General precedence concept for single stage problem (option 2).

$$\begin{bmatrix} Y_{i,i'} \\ x_i + p_i \le x_{i'} \end{bmatrix} \veebar \begin{bmatrix} Y_{i',i} \\ x_{i'} + p_{i'} \le x_i \end{bmatrix} \veebar Y^3_{i,i'} \;\forall i \in I, i' \in I, i < i' \tag{75}$$

$$Y^3_{i,i'} \Rightarrow \neg(\bar{Y}_{i,m} \wedge \bar{Y}_{i',m}) \;\forall i \in I, i' \in I, i < i', m \in M \tag{76}$$

The GDP model for option 2 also includes equations (3-5) and disjunctions (62).

5.2.1. MILP reformulation

The MILP reformulation of the GDP general precedence model corresponding to option 2 is essentially the model proposed by Jain and Grossmann[5]. The big-M reformulation of the constraints in the first and second disjunctions in equation (75), gives rise to equation (77). Then, we need to ensure that a single term of the disjunction is active, equation (78). As for the conversion of the logic into MILP format, this yields equation (79).

$$x_i + p_i \le x_{i'} + M_{i,i'} \cdot (1 - y_{i,i'}) \;\forall i \in I, i' \in I, i \ne i' \tag{77}$$

$$y_{i,i'} + y_{i',i} + y^3_{i,i'} = 1 \;\forall i \in I, i' \in I, i < i' \tag{78}$$

$$\begin{aligned} &Y^3_{i,i'} \Rightarrow \neg(\bar{Y}_{i,m} \wedge \bar{Y}_{i',m}) \Leftrightarrow \neg Y^3_{i,i'} \vee \neg \bar{Y}_{i,m} \vee \neg \bar{Y}_{i',m} \Leftrightarrow \\ &1 - y^3_{i,i'} + 1 - \bar{y}_{i,m} + 1 - \bar{y}_{i',m} \ge 1 \Leftrightarrow \\ &y^3_{i,i'} \le 2 - \bar{y}_{i,m} - \bar{y}_{i',m} \;\forall i \in I, i' \in I, i < i', m \in M \end{aligned} \tag{79}$$

We now just need to replace (78) in (79) to remove variables $y^3_{i,i'}$ from the formulation, and apply the convex hull reformulation to (62), as described in section 5.1.1, to derive the final model. It consists of equations (3), (67), (73-74) and (80-81).

$$x_i + \sum_{m \in M} p_{i,m} \cdot \bar{y}_{i,m} \le x_{i'} + M_{i,i'} \cdot (1 - y_{i,i'}) \;\forall i \in I, i' \in I, i \ne i' \tag{80}$$

$$y_{i,i'} + y_{i',i} \ge \bar{y}_{i,m} + \bar{y}_{i',m} - 1 \;\forall i \in I, i' \in I, i < i', m \in M \tag{81}$$

5.3. Time slots concept with multiple grids

When relying on the concept of time slots, moving from a single unit to multiple units in a single stage has an important implication: we no longer know a priori how many orders are going to be allocated to a particular unit. Therefore, there is a need to define multiple time grids, one for each unit, and it will be assumed that all grids feature the same number of slots. Note that with multiple time grids, the execution of orders is limited to a single slot, which results in a better performance

compared to the alternative of a single common grid for all units[10], where orders may span across multiple slots. Only by chance will the specified number of slots be equal to the number of allocated orders for every unit, and hence idle slots are highly likely to occur. Naturally, if the number of slots is too low ($|T|<|I|/|M|$), the model will be infeasible.

Figure 6 illustrates the concept, where Boolean variables $Y_{i,m,t}$=*True* if order *i* is allocated to slot *t* of unit *m* and variables $Y_{m,t}^{free}$ identify if the slot *t* of unit *m* has no order assigned to. Clearly, a particular slot of a given unit may either have an order assigned to or none at all, leading to the disjunctions in equation (82). Then, a particular order needs to be allocated to a slot of a certain unit, equation (83). Finally, equation (84) reduces solution degeneracy by keeping idle slots last and next to each other[15]. More specifically, idle slots are the ones further to right, or in other words, if slot *t* in unit *m* is free then slot *t*+1 must also be free of orders.

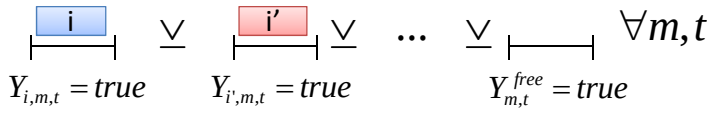

Figure 6. Time slots concept with multiple grids for single stage problem.

$$\bigvee_i \begin{bmatrix} Y_{i,m,t} \\ MS\big|_{t=|T|} + x_{m,t+1}\big|_{t\neq|T|} - x_{m,t} \geq p_{i,m} \\ x_{m,t} \geq r_i \\ x_{m,t} + p_{i,m} \leq d_i \end{bmatrix} \veebar \begin{bmatrix} Y_{m,t}^{free} \\ MS\big|_{t=|T|} + x_{m,t+1}\big|_{t\neq|T|} - x_{m,t} = 0 \\ x_{m,t} \geq 0 \\ x_{m,t} \leq H \end{bmatrix} \quad \forall m \in M, t \in T \tag{82}$$

$$\bigvee_{m,t} Y_{i,m,t} \quad \forall i \in I \tag{83}$$

$$Y_{m,t}^{free} \Rightarrow Y_{m,t+1}^{free} \quad \forall m \in M, t \in T \tag{84}$$

5.3.1. Convex hull reformulation

Similarly to the single unit case, applying the convex hull reformulation and a few algebraic manipulations makes it possible to eliminate the disaggregated variables and reduce the number of constraints to come up with a very tight and efficient model. The final outcome is essentially the model by Castro and Grossmann[10].

Equations (85-87) give the relation between the original and the new set of disaggregated variables.

$$MS = \sum_{i \in I} \widehat{MS}_{i,m} + \widehat{MS}_{m}^{free} \ \forall m \in M \tag{85}$$

$$x_{m,t} = \sum_{i \in I} \widehat{x}_{t,i,m,t} + \widehat{x}_{t,m,t}^{free} \ \forall m \in M, t \in T \tag{86}$$

$$x_{m,t+1} = \sum_{i \in I} \widehat{x}_{t+1,i,m,t} + \widehat{x}_{t+1,m,t}^{free} \ \forall m \in M, t \in T, t \neq |T| \tag{87}$$

Equations (88-91) correspond to the constraints within the disjunctions in (82), while equation (92) is the only bound constraint needed for the follow up.

$$\widehat{MS}_{i,m}\Big|_{t=|T|} + \widehat{x}_{t+1,i,m,t}\Big|_{t\neq|T|} - \widehat{x}_{t,i,m,t} \geq p_{i,m} \cdot y_{i,m,t} \ \forall i \in I, m \in M, t \in T \tag{88}$$

$$\widehat{x}_{t,i,m,t} \geq r_i \cdot y_{i,m,t} \ \forall i \in I, m \in M, t \in T \tag{89}$$

$$\widehat{x}_{t,i,m,t} \leq (d_i - p_{i,m}) \cdot y_{i,m,t} \ \forall i \in I, m \in M, t \in T \tag{90}$$

$$\widehat{MS}_{m}^{free}\Big|_{t=|T|} + \widehat{x}_{t+1,m,t}^{free}\Big|_{t\neq|T|} - \widehat{x}_{t,m,t}^{free} = 0 \cdot y_{m,t}^{free} \ \forall m \in M, t \in T \tag{91}$$

$$0 \cdot y_{m,t}^{free} \leq \widehat{x}_{t,m,t}^{free} \leq \max_{i \in I}(d_i) \cdot y_{m,t}^{free} \ \forall m \in M, t \in T \tag{92}$$

To eliminate the disaggregated variables one only needs to add over *i* all constraints in (88-90), and replace the sum of the disaggregated variables with the original variables according to (85-87). The remaining disaggregated free variables are then eliminated according to (91), leading to equation (93), or replaced by their lower or upper bounds in (92), resulting in equations (94-95), respectively.

$$MS\Big|_{t=|T|} + x_{m,t+1}\Big|_{t\neq|T|} - x_{m,t} \geq \sum_{i \in I} p_{i,m} \cdot y_{i,m,t} \ \forall m \in M, t \in T \tag{93}$$

$$x_{m,t} \geq \sum_{i \in I} r_i \cdot y_{i,m,t} \ \forall m \in M, t \in T \tag{94}$$

$$x_{m,t} \leq \sum_{i \in I} (d_i - p_{i,m}) \cdot y_{i,m,t} + \max_{i \in I}(d_i) \cdot y_{m,t}^{free} \ \forall m \in M, t \in T \tag{95}$$

The model is complete with the constraints relating the binary variables, (96-98), where the latter results from the logic expressed in (84).

$$\sum_{i\in I} y_{i,m,t} + y_{m,t}^{free} = 1 \;\forall m \in M, t \in T \tag{96}$$

$$\sum_{m\in M}\sum_{t\in T} y_{i,m,t} = 1 \;\forall i \in I \tag{97}$$

$$y_{m,t+1}^{free} \geq y_{m,t}^{free} \;\forall m \in M, t \in T, t \neq |T| \tag{98}$$

6. Computational Results

The performance of the different MILP models is illustrated through the solutions of a few example problems with up to 30 orders and 5 parallel units. We consider 10 single unit problems with random generated data (provided as Supporting Information) for comparison of the 7 different formulations derived from the 3 different concepts for continuous-time representation. Then, 10 single stage problems with parallel units corresponding to problems 3.1 to 5.2 from Harjunkoski and Grossmann[6], and problems P7-P10 in Castro and Grossmann[10], are used to evaluate the two general precedence options and the time slots concept alternative.

The models were implemented in GAMS 23.5 and solved using CPLEX 12.2 with a single thread up to a relative optimality tolerance=10^{-6}, or a maximum computational time of 3,600 CPUs. The hardware consisted on a laptop with an Intel Core2 Duo T9300 2.5 GHz processor, with 4 GB of RAM and running Windows Vista Enterprise.

6.1. Single unit problems

The results are given in Table 2, where the acronyms for the models are the following. In terms of the concept for time representation, TS stands for time slots, GP for general precedence and IP for immediate precedence. With respect to the GDP reformulation, CH refers to convex hull and BM for big-M. In the case of TS_CH, we consider the models with (section 4.3.2) and without (TS_CH_C) the disaggregated variables (section 4.3.3).

As seen in Table 2, the compact time slots convex hull model (TS_CH_C) is the best performer by at least one order of magnitude, the exception being the general precedence big-M model (GP_BM) in Ex3, which is roughly five times faster. Even if we use the disaggregated variables, we still get a reasonable performance despite the significant increase in problem size. The exception is again Ex3,

for which TS_CH is unable to prove optimality (best possible solution at time of termination=552). Clearly, the time slots concept is the best for the single unit problem provided that one relies on the very tight convex hull reformulation since its big-M counterpart is very poor, failing to find the optimal solution for EX6 and not even finding a feasible solution for Ex7-10.

Table 2. Computational effort (CPU s) for single unit problems (best performer in bold, maximum resource limit or out of memory termination in italic)

Problem	|I|	Optimum	TS_CH_C	TS_CH	GP_BM	IP_CH	GP_CH	TS_BM	IP-BM
Ex1	15	430	**0.34**	3.77	113	1.1	1375	*3600*	*3600*
Ex2	15	438	**0.22**	0.72	26.6	3.15	412	3599	*3600*
Ex3	20	559	1.27	*3600*	**0.25**	518	1.2	204	92.9
Ex4	20	664	**0.78**	53.1	2186	168	*3600*	*3600*	*771*[a]
Ex5	20	680	**1.02**	36.3	247	210	1415	*3600*	*896*
Ex6	25	739	**1.53**	123	166	1537	*3600*	*3600*[b]	*3600*
Ex7	25	700	**1.07**	21.6	*3600*	*3600*[a]	*3600*	*3600*[a]	*3600*[a]
Ex8	25	772	**1.80**	80.3	*3600*	915	*3600*	*3600*[a]	*635*[a]
Ex9	30	853	**0.59**	63.3	*3600*	479	*3600*	*3600*[a]	*1067*[a]
Ex10	30	929	**3.91**	213	*3600*	*3600*[a]	*3600*	*3600*[a]	*3600*[a]

[a] No feasible solution. [b] Suboptimal solution=797.

Table 3. Relative optimality gap from relaxed model in the initial (I), also known as integrality gap, and final (F) iterations of the branch-and-bound search with respect to known optimal solution

	TS_CH		TS_BM		IP_CH		GP_CH		GP_BM		IP_BM
Problem	I	F	I	F	I	F	I	F	I	F	F
Ex1	0.0%	0%	87%	4%	38%	0%	40%	0%	44%	0%	11%
Ex2	0.7%	0%	88%	0%	46%	0%	46%	0%	47%	0%	42%
Ex3	6.3%	0%	89%	0%	7%	0%	7%	0%	7%	0%	0%
Ex4	4.8%	0%	92%	32%	36%	0%	35%	8%	38%	0%	36%
Ex5	2.9%	0%	91%	15%	38%	0%	35%	0%	38%	0%	31%
Ex6	3.9%	0%	92%	16%	21%	0%	19%	1%	21%	0%	19%
Ex7	0.0%	0%	92%	56%	53%	0%	55%	12%	56%	6%	54%
Ex8	3.0%	0%	92%	38%	33%	0%	33%	7%	33%	3%	33%
Ex9	0.7%	0%	93%	69%	40%	0%	40%	29%	40%	25%	40%
Ex10	1.5%	0%	94%	68%	39%	0%	40%	27%	41%	18%	40%

The results for the two precedence concepts are quite interesting. While it is better to rely on general precedence with the big-M reformulation (GP_BM), it becomes increasingly more difficult to prove optimality as the problem size increases (failures in Ex7-Ex10, see Table 3). On the other hand, for immediate precedence, the convex-hull is much better than the big-M reformulation and despite the failure to find a feasible solution, the best possible solution from the MILP relaxation rapidly becomes equal to the optimal solution. Since GP_BM excels at finding near optimal

solutions very rapidly, GP_BM and IP_CH have complementary strengths. In other words, one can use the upper bound from the former and the lower bound from the latter to prove optimality in under a minute, for all problems solved. However, such hybrid algorithm would still be worse than TS_CH_C.

The convex hull reformulation of the general precedence concept (GP_CH) can still find all optimal solutions, but is restricted to proving optimality in 4 cases. Furthermore, the optimality gap at time of termination is always larger than for GP_BM, despite being normally tighter, see Table 3. Note that the big-M reformulations of the two precedence models have the exact same relaxation (this explains why the I column for IP_BM is not shown). With respect to the convex hull reformulation, the model from general precedence is sometimes tighter and sometimes looser than its immediate precedence counterpart. The worst model by far in terms of relaxation is TS_BM, the second worst performer of the lot. Also note that eliminating the disaggregated variables when going from TS_CH to TS_CH_C leads to one third the number of total variables (see Table 4) and a one order of magnitude reduction in the number of equations (Table 5) but does not affect the relaxation.

Table 4. Computational statistics related to problem size (part 1)

	Binary variables			Continuous variables						
|I|	GP	TS	IP	GP_BM	TS_CH_C	TS_BM	IP_BM	GP_CH	TS_CH	IP_CH
15	105	225	240	16	16	16	16	436	466	886
20	190	400	420	21	21	21	21	781	821	1581
25	300	625	650	26	26	26	26	1226	1276	2476
30	435	900	930	31	31	31	31	1771	1831	3571

Table 5. Computational statistics related to problem size (part 2)

	Total Equations						
|I|	TS_CH_C	GP_BM	IP_BM	TS_BM	TS_CH	GP_CH	IP_CH
15	75	225	692	705	1185	1275	2492
20	100	400	1222	1240	2080	2300	4422
25	125	625	1902	1925	3225	3625	6902
30	150	900	2732	2760	4620	5250	9932

The number of binary variables is dependent on the time representation concept and not on the reformulation. Notice in Table 4 that the general precedence model is the one with fewest binary variables, less than half the number of its immediate precedence counterpart, which together with the

lower number of equations (see Table 5) can be used to explain its better performance. With respect to the number of continuous variables, it is the same for all big-M reformulations (one starting time per order plus the makespan) and the compact time slot convex hull reformulation. The number increases significantly for the convex hull reformulations being lower for GP and higher for IP.

Together with the integrality gap, the number of equations is the other statistic that explains the better performance of TS_CH_C, with the difference to other models becoming larger with an increase in problem size. While predicting the computational performance of the MILP based solely on problem size seems to work for IP_CH, since it requires the highest number of binary, continuous variables and equations and it has the worst overall performance, it is misleading when comparing TS_CH and GP_BM because the former has a larger number of entities but a much better performance, primarily due to a considerably lower integrality gap.

6.2. Single stage, parallel units problems

The better performance of the time slots and general precedence concepts in the single unit problem, explains why it was chosen to discard the immediate precedence concept when going for the more complex, multiple parallel units scenario. With respect to the time slots approach, we consider two alternatives: (i) TS_H is a so called heuristic approach[11], where the given number of slots is determined from the standard global search procedure[8] (keep increasing |T| while observing a change in the value of objective function then stop and report the result from the previous iteration, which also led to the same value and thus defines the minimum number of time slots); (ii) TS_F the full approach with |T|=|I|, featuring enough slots to guarantee global optimal solutions in a single run, thus providing a fairer comparison with the general precedence methods. The latter are named GP_1 and GP_2, respectively from modelling options 1 (see section 5.1) and 2 (see section 5.2).

The results in Table 6 offer no doubt about the best performer, the time slots approach for the minimum number of slots, TS_H. Notice that the optimal number of slots listed in column 6 is at most the minimum number that potentially leads to a feasible solution, resulting from evenly distributed the orders amongst the available units, $\lceil |I|/|M| \rceil$, plus two (in H&G5.2). More than the doubling the number of slots up to the maximum (TS_F) degrades the performance by at least one

order of magnitude (notice the increase in integrality gap in Table 7), but confirms the superiority of the time slots concept, which is only surpassed by the general precedence models in H&G4.2.

Table 6. Results for single stage problems (best performer in bold, maximum resource limit or out of memory termination in italic)

		Optimum	Suboptimal solutions		Time slots |T|		CPUs			
Problem	(|I|,|M|)	TS_F	GP_1	GP_2	TS_H	TS_F	TS_H	TS_F	GP_1	GP_2
H&G3.1	(12,3)	365	369	-	5	12	**0.41**	2.41	*3600*	898
H&G3.2	(12,3)	161	-	-	5	12	**0.30**	0.68	*3600*	55.0
H&G4.1	(15,5)	259	-	-	4	15	**1.73**	12.4	*3600*	*3600*
H&G4.2	(15,5)	147	-	-	3	15	**0.43**	2.07	1.33	0.64
H&G5.1	(20,5)	302	315	309	6	20	**3.22**	23.0	*3600*	*3600*
H&G5.2	(20,5)	148	-	158	5	20	**82.5**	611	*3600*	*3600*
C&G7	(25,5)	188	190	204	5	25	**3.65**	47.8	*3600*	*3600*
C&G8	(25,5)	197	-	217	5	25	**2.6**	70.8	*3600*	*541*
C&G9	(30,5)	178	181	193	7	30	**3.48**	102	*3600*	*3600*
C&G10	(30,5)	220	228	263	7	30	**2.95**	408	*3600*	*925*

Table 7. Relative optimality gap from relaxed model in the initial (I), also known as integrality gap, and final (F) iterations of the branch-and-bound search with respect to known optimal solution

	TS_H		TS_F		GP_1		GP_2	
Problem	I	F	I	F	I	F	I	F
H&G3.1	1.7%	0%	6.2%	0%	44%	19%	44%	0%
H&G3.2	6.2%	0%	11%	0%	8.7%	8.1%	8.7%	0%
H&G4.1	5.4%	0%	9.0%	0%	22%	22%	22%	14%
H&G4.2	15%	0%	29%	0%	0%	0%	0%	0%
H&G5.1	3.2%	0%	5.3%	0%	33%	33%	33%	29%
H&G5.2	6.4%	0%	12%	0%	0.7%	0.7%	0.7%	0.7%
C&G7	0.5%	0%	11%	0%	30%	30%	30%	26%
C&G8	0.5%	0%	11%	0%	32%	32%	32%	32%
C&G9	6.6%	0%	8.9%	0%	30%	30%	30%	30%
C&G10	5.9%	0%	7.8%	0%	40%	40%	40%	40%

The comparison between the two general precedence models is not as conclusive. We can argue that GP_1, linked to the approach by Méndez et al.[4], has a slight edge since it can find the optimal solutions in 5 vs. 4 cases and normally generates better schedules. Furthermore, it seems to be more robust to an increase in problem size, since it returns better solutions for the four largest problems (C&G7-C&G10). On the other hand, GP_2, linked to the approach by Jain & Grossmann[5], is able to prove optimality in two more cases, corresponding to the smallest problems H&G3.1 and H&G3.2. Provided that the solver does not run out of memory, it also leads to a smaller optimality gap at time

of termination (considering the real optimum and not the best feasible solution up to that point), see Table 7. The integrality gap of both formulations is the same, however.

Finally, Table 8 provides the statistics related to problem size. Similarly to the single unit problem, the general precedence concept involves the fewest number of binary and continuous variables. Notice, however, that option 2 generates considerably more binary variables, which is compensated by the smaller number of equations. In terms of continuous variables, the same number results from all three approaches if the number of time slots happens to be equal to the number of orders (e.g. (25,5) row, corresponding to C&G7). Using the time slots concept leads to the smallest number of equations, even for the maximum number of slots (TS_F). For the heuristic approach (TS_H), the difference to the general precedence concept is roughly one order of magnitude.

Table 8. Computational statistics related to problem size for single stage problems

	Binary variables				Continuous variables				Total equations			
(|I|,|M|)	GP_1	GP_2	TS_H	TS_F	GP_1	GP_2	TS_H	TS_F	TS_H	TS_F	GP_2	GP_1
(12,3)	102	168	195	468	13	13	16	37	84	189	366	432
(15,5)	180	285	320	1200	16	16	21	76	110	385	780	1095
(20,5)	290	480	630	2100	21	21	31	101	165	515	1390	1960
(25,5)	425	725	650	3250	26	26	26	126	145	645	2175	3075
(30,5)	585	1020	1085	4650	31	31	36	151	200	775	3135	4440

7. Conclusions

This paper has shown that well-known MILP formulations from the literature can be derived from elementary time representation concepts and their corresponding Generalized Disjunctive Programming models, featuring disjunctions to consider all operational possibilities, and simple logic. By first proposing a GDP model, the burden of finding the appropriate constraints for the MILP has been significantly reduced by taking advantage of reformulation techniques such as the big-M and convex hull to systematically generate such constraints. The overall modeling process has thus effectively been divided into two steps, where choosing the right time representation is at least as important as choosing the best reformulation method. More specifically, and while the time slots convex hull reformulation proved to be the best performer for the single stage problems considered, going for a time slots big-M model is much worse than using a general precedence big-M

formulation. Another interesting result was that the two similar precedence concepts (general and immediate) gave opposite results with respect to the most efficient reformulation. Finally, we have also seen that by using the knowledge that is implicit in the disjunctions, tighter bounds can be derived for the big-M constraints.

Future work will be concerned about more complex scheduling problems involving multistage and multipurpose plants, as well as sequence dependent changeovers and different storage policies.

Acknowledgments

The authors gratefully acknowledge financial support from the Luso-American and National Science Foundations, under the 2011 Portugal – U.S. Research Networks Program, and from the NSF under grant OCI-0750826.

References

(1) Kondili, E.; Pantelides, C. C.; Sargent, R. A General Algorithm for Short-Term Scheduling of Batch Operations - I. MILP Formulation. *Comput. Chem. Eng.* **1993**, *17*, 211.

(2) Pantelides, C.C. Unified Frameworks for the Optimal Process Planning and Scheduling. In *Proceedings of the Second Conference on Foundations of Computer Aided Operations*; Cache Publications: New York, 1994; pp 253.

(3) Shah, N.; Pantelides, C.C.; Sargent, R.W.H. A general algorithm for short-term scheduling of batch operations—II. Computational issues. *Comput. Chem. Eng.* **1993**, *17*, 229.

(4) Méndez, C.A.; Henning, G.P.; Cerdá, J. An MILP continuous-time approach to short-term scheduling of resource-constrained multistage flowshop batch facilities. *Comput. Chem. Eng.* **2001**, *25*, 701.

(5) Jain, V.; Grossmann, I.E. Algorithms for Hybrid MILP/CP Models for a Class of Optimization Problems. *Informs Journal on Computing* **2001**, *13* (4), 258.

(6) Harjunkoski, I.; Grossmann; I. E. Decomposition techniques for multistage scheduling problems using mixed-integer and constraint programming methods. *Comput. Chem. Eng.* **2002**, *26*, 1533.

(7) Maravelias, C.T.; Grossmann, I.E. A New General Continuous-Time State Task Network Formulation for Short Term, Scheduling of Multipurpose Batch Plants. *Ind. Eng. Chem. Res.* **2003**, *42*, 3056.

(8) Castro, P.M.; Barbosa-Póvoa, A.P.; Matos, H.A.; Novais, A.Q. Simple Continuous-time Formulation for Short-Term Scheduling of Batch and Continuous Processes. *Ind. Eng. Chem. Res.* **2004**, *43*, 105.

(9) Sundaramoorthy, A.; Karimi, I.A.. A simpler better slot-based continuous-time formulation for short-term scheduling in multipurpose batch plants. *Chem. Eng. Sci*. **2005**, *60*, 2679.

(10) Castro, P.; Grossmann, I.E. An Efficient MILP Model for the Short-term Scheduling of Single Stage Batch Plants. *Comput. Chem. Eng.* **2006**, *30*, 1003.

(11) Liu, Y.; Karimi, I.A. Scheduling multistage, multiproduct batch plants with nonidentical parallel units and unlimited intermediate storage. *Chemical Engineering Science* **2007**, *62*, 1549.

(12) Castro, P.M.; Grossmann, I.E.; Novais, A.Q. Two New Continuous-Time Models for the Scheduling of Multistage Batch Plants with Sequence Dependent Changeovers. *Ind. Eng. Chem. Res.* **2006**, *45*, 6210.

(13) Ierapetritou, M.G.; Floudas, C.A. Effective continuous-time formulation for short-term scheduling. 1. Multipurpose batch processes. *Ind. Eng. Chem. Res.* **1998**, *37*, 4341.

(14) Castro P.M.; Grossmann I.E. New Continuous-time MILP model for the short-term scheduling of multistage batch plants. *Ind. Eng. Chem. Res.* **2005**, *44*, 9175.

(15) Castro P.M.; Novais A.Q. Short-term scheduling of multistage batch plants with unlimited intermediate storage. *Ind. Eng. Chem. Res.* **2008**, *47*, 6126.

(16) Shaik, M.; Floudas, C. Novel Unified Modeling Approach for Short-Term Scheduling. *Ind. Eng. Chem. Res.* **2009**, *48*, 2947.

(17) Susarla, N.; Li, J.; Karimi, I.A. A Novel Approach to Scheduling Multipurpose Batch Plants Using Unit-Slots. *AIChE J.* **2010**, *56*, 1859.

(18) Seid, R.; Majozi, T. A robust mathematical formulation for multipurpose batch plants. *Chem. Eng. Sci*. **2012**, *68*, 36.

(19) Raman, R; Grossmann, I.E. Modeling and Computational techniques for Logic Based Integer Programming. *Comput. Chem. Eng.* **1994**, *18*, 563.

(20) Nemhauser, G.L.; Wolsey, L.A. Integer and Combinatorial Optimization. Wiley 1998.

(21) Balas, E. Disjunctive Programming and a Hierarchy of Relaxations for Discrete Optimization Problems. *SIAM Journal of Algebraic and Discrete Mathematics* **1985**, *6* (3), 466.

(22) Türkay, M.; Grossmann, I.E. Logic-based Algorithms for the Optimal Synthesis of Process Networks. *Comput. Chem. Eng.* **1996**, *20*, 959.

(23) Vecchietti, A.; Lee, S.; Grossmann, I.E. Modeling of Discrete/Continuous Optimization Problems: Characterization and Formulation of Disjunctions and their Relaxations. *Comput. Chem. Eng.* **2003**, *27*, 433.

(24) Ropotar, M.; Kravanja, Z. Translation of Variables and Implementation of Efficient Logic-Based Techniques in the MINLP Process Synthesizer MIPSYN. *AIChE J.* **2009**, *55*, 2896.

(25) Montagna, J.M.; Iribarren, O.A.; Vecchietti, A. Synthesis of Biotechnological Processes Using Generalized Disjunctive Programming. *Ind. Eng. Chem. Res.* **2004**, *43*, 4220.

(26) Sawaya, N.W.; Grossmann, I.E. A Cutting Plane Method for Solving Linear Generalized Disjunctive Programming Problems. *Comput. Chem. Eng.* **2005**, *29*, 1891.

(27) Beaumont, N. An algorithm for Disjunctive Programs. *European Journal of Operation Research* **1990**, *48*, 362.

(28) Biegler, L.T.; Grossmann, I.E.; Westerberg, A.W. Systematic Methods of Chemical Processing Design. Prentice Hall 1997.

(29) Raman, R.; Grossmann, I.E. Relation Between MILP Modelling and Logical Inference for Chemical Process Synthesis. *Comput. Chem. Eng.* **1991**, *15*, 73.

(30) Clocksin, W.F.; Mellish, C.S. Programming in Prolog. New York: Springer-Verlag 1981.