Mostafaei, Hossein; Harjunkoski, Iiro

Continuous-time scheduling formulation for multipurpose batch plants

Published in:
AIChE Journal

DOI:
10.1002/aic.16804

Published: 01/02/2020

Document Version
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

Continuous-time scheduling formulation for multipurpose batch plants

Hossein Mostafaei[1] | Iiro Harjunkoski[1,2]

[1]Department of Chemical and Metallurgical Engineering, School of Chemical Engineering, Aalto University, Espoo, Finland

[2]ABB Corporate Research Germany, Ladenburg, Germany

Correspondence
Iiro Harjunkoski, ABB Corporate Research, Wallstadter Str. 59, 68526 Ladenburg, Germany
Email: iiro.harjunkoski@de.abb.com

Funding information
Academy of Finland, Grant/Award Number: 313466

Abstract

Short-term scheduling of batch processes is a complex combinatorial problem with remarkable impact on the total revenue of chemical plants. It consists of the optimal allocation of limited resources to tasks over time in order to manufacture final products following given batch recipes. This article addresses the short-term scheduling of multipurpose batch plants, using a mixed integer linear programming formulation based on the state-task network representation. It employs both single-grid and multi-grid continuous-time representations, derived from generalized disjunctive programming. In comparison to other multigrid scheduling models in the literature, the proposed multi-grid model uses no big-M constraints and leads to more compact mathematical models with strong linear relaxations, which often results in shorter computational times. The single-grid counterpart of the formulation is not as favorable, as it leads to weaker linear relaxations than the multi-grid approach and is not capable of handling changeover time constraints.

KEYWORDS

batch plant, generalized disjunctive programming, MILP, scheduling, state-task network

1 | INTRODUCTION

Short-term scheduling of multipurpose batch processes is a challenging problem and has received increasing attention over the last 20 years. It consists of allocating limited resources to tasks over time in order to produce final products following given batch recipes in terms of some specific performance criteria (e.g., minimum cost, makesapn or maximum profit). Different types of rigorous optimization approaches have been proposed to study this problem, with most of them consisting of solving a discrete- or continuous-time mixed integer linear programming (MILP) formulation. Reviews can be found in References 1-5.

In discrete-time approaches, the time horizon of interest is divided into a number of time slots of equal and fixed duration,[6,7] with the beginning and ending of tasks being associated with the boundaries of these time intervals. In contrast, in - continuous-time representations the length of time slots is selected by the optimization,[8-11] thus leading to a significant reduction in the number of decision variables. Though continuous-time approaches will result in better solutions than their discrete counterparts, they inevitably need tuning parameters (e.g., number of time slots), thus requiring an iterative procedure to find the optimal solutions.[12] This is a big drawback because the initial guess on the minimum number of time slots is hardly trivial, thus involving, in most cases, solving a particular instance at least five times (see, e.g., Table 6 in Reference 13). However, if the initial guess is accurate enough, no more than two iterations are usually needed.

For all of the above, the time representation becomes a critical issue for scheduling problems,[14] with discrete-time being a particular case of continuous-time.[15] The main reason why the large majority of scheduling models for batch plants have adopted a continuous-time representation is that it is easier to cope with variable processing times. Two types of continuous-time - models have been developed for multipurpose batch plants: single-grid[16-20] and multi-grid.[21-29,33] In the former, events and time slots are common for all units, while in the latter each unit has its own set of time intervals, which is not shared with other units. Both formulations have their pros and cons, but the remarkable advantage of multi-grid continuous-time models is that they need fewer event points to generate optimal solutions. In turn, the single-grid models can consider resource constraints other than equipment, together with various storage policies.

Most of the scheduling models for multipurpose batch plants can be classified into two categories according to the process network representation: state-task network (STN)[6,18,21-24,26-28] and resource-task network (RTN),[2,16,17,19,25,31] both facilitating the modeling by converting the real plant entities into virtual entities. In contrast to the STN, the RTN representation has been shown to be very versatile due to the unified treatment of resources, meaning that all resources are the same, regardless if they are equipments, materials, and utilities. The first MILP scheduling model based on the STN for batch plants was presented by Kondili et al.[6] who relied on the discrete-time representation where the length of the time slots is determined from the greatest common factor among the processing times involved in the problem. For the same problem, Ierapetritou and Floudas[21] developed an efficient multi-grid continuous-time MILP formulation that was then generalized to tackle continuous and semicontinuous processes with multiple intermediate due dates.[22,23] The models need fewer events to discover the optimal solutions and can be orders of magnitude faster than the single-grid models.[17-20]

In another work, Castro et al.[19] presented a single-grid continuous-time formulation based on the RTN that is suitable for both continuous and batch plants. Compared to the earlier attempt by the authors,[17] the formulation uses different sets of timing constraints that apart from providing the optimal solutions in lower CPU times, lead to strong linear relaxations. Shaik and co-workers[26-28] introduced multi-grid continuous-time models based on the work by Ierapetritou and Floudas,[21] which can handle both batch and continuous processes. The main disadvantage of these models comes from allowing tasks to be taken place over multiple events, which is achieved with the model parameter Δn (=0, 1, 2, …), a limit on the maximum number of events over which a task can span. For a new instance, it is not known which value of Δn should be adopted, and, for example, switching Δn from zero to one doubles the number of binary variables and constraints (see, e.g., Table 7 in Reference 26). This is not the case for the multi-grid continuous-time formulation by Susarla et al.[29] who generalized the model by Sundaramoorthy and Karimi[20] to allow various types of storage and wait policies for material states.

In addition to developing a wide range of MILP models, research on the scheduling of multipurpose batch plants has also studied a variety of solution methods. Velez et al.[32] developed a constraint propagation algorithm to account for the calculation of lower bounds on the number and size of tasks necessary to meet given demand, which leads to reductions in the computational requirements by orders of magnitude. Wu and Ierapetritou[33] used a decomposition method based on a heuristic approach and Lagrangian relaxation to provide lower and upper bounds for the original problem first introduced by Ierapetritou and Floudas.[21] More recently, Lee and Maravelias[30] devised a three-stage optimization -framework, in which the first stage uses a discrete-time model to find an approximate solution, which then is mapped onto a continuous-time model. The final stage uses a continuous-time linear programming (LP) model to obtain a fairly accurate solution by refining the timing of events and batch sizes.

On the other hand, generalized disjunctive programming (GDP) is becoming more popular, providing a high-level framework for modeling complex mixed integer programs.[34-40] It is generally followed by two formulations with complementary strengths: convex hull and big-M reformulations. The former has been shown to always be at least as tight as the big-M reformulation, but needs a couple more decision variables and constraints. Maravelias and Grossmann[18] are the first to partly rely on a GDP-based formulation for the scheduling of batch plants. The advantage is a tight formulation when minimizing the makespan (see, e.g., tables 15–17 in Reference 13), which is a more difficult performance criterion for scheduling problems.[18]

In this article, we develop a continuous-time MILP approach based on the STN for the optimal scheduling of multipurpose batch plants. Both single-grid and multi-grid approaches are addressed, which differ only by their timing constraints. The proposed model uses a slot-based approach, always leading to one more event point in our multi-grid model compared with unit-specific event-based approaches developed by Floudas and co-workers.[21-26] This is because, in the slot-based approaches when a task starts at a time point it does not finish at the same time point. In order to ensure an efficient MILP formulation by design, we rely on GDP for deriving the model constraints, applying convex hull reformulations and converting logical propositions into integer inequalities. Results show that the proposed multi-grid model is very efficient computationally and leads to tighter linear relaxations.

2 | PROBLEM STATEMENT

This work addresses the short-term scheduling of multipurpose batch plants with no resources other than materials and equipment units. Given the problem data (including the processing time of tasks in units, material inventories, and the available units for each task) and the production recipe through the STN, we aim to develop a continuous-time MILP formulation to optimally determine the sequence, timing, and size of tasks in each equipment unit. We will use the idea of task decoupling, in which a task that can be processed in several units is replaced by several tasks that represent unique task–unit combinations (see Figure 1).

We adopt a continuous-time representation with both single and multiple reference grids, featuring a set $r \in R$ time points and $|R| - 1$ time slots. In the single-grid case, the time slots are common for all units whereas in multigrid option, each unit has its own set of time intervals, which is not shared with other units. In other words, in the multi-grid option the occurrences of each time point can vary across different units. An example is event point r_5 in the left hand side of Figure 2. The time point r_5 across unit $J2$ occurs at time 7.85 hr whereas it locates at time 7.21 hr along the time axis of unit $J3$.

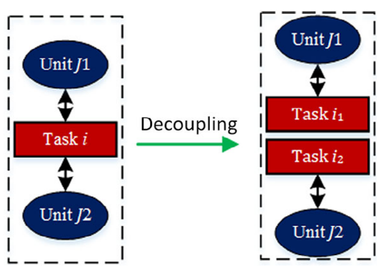

FIGURE 1 Task decoupling idea: Task ***i*** is replaced by two Tasks $\boldsymbol{i_1}$ and $\boldsymbol{i_2}$ [Color figure can be viewed at wileyonlinelibrary.com]

We use the binary variable $X_{i,r,r'}$ to indicate that task i is performed during time interval $[r,r']|_{r<r'\leq r+\Delta r}$, where Δr is a limit on the maximum number of time slots over which task i can span (if Δr = 1, then tasks can only last one time slot). For example, in the single-grid case in Figure 2, task I8 spans over three time slots $[r_4, r_5]$, $[r_5, r_6]$ and $[r_6, r_7]$, starting at time point r_4(6 hr) and ending at time point r_7 (8 hr).

3 | MATHEMATICAL MODEL

In the mathematical formulation to be presented next, we rely on GDP to derive the new MILP scheduling model for the multipurpose batch plants. We first present the common constraints for the single-grid (- SG) and the multi-grid (MG) models and then separately device timing constraints. Finally, we incorporate intermediate due date constraints into the SG.

3.1 | Common constraints for the SG and MG models

3.1.1 | Allocation, batch size, and processing time

Let $X_{i,\ r,\ r'}$ be a binary variable indicating that task $i \in I_j$ is being processed during time interval $[r,r']|_{r<r'\leq r+\Delta r}$. According to - disjunction in Equation (1), in each equipment unit at most a single task can be processed during the time interval $[r,r']|_{r<r'\leq r+\Delta r}$. Let $V_{i,\ r,\ r'}$ represent the batch size of task i being processed during time interval $[r,r']|_{r<r'\leq r+\Delta r}$. The lower and upper limits on the batch size must be taken into account in each unit. The processing time of task i during time interval $[r,r']|_{r<r'\leq r+\Delta r}$, that is, $LR_{i,\ r,\ r'}$ is a linear function of batch size, where cp_i and vp_i are the constant and variable terms for the processing time of task i in unit j. We have thus the following disjunction:

$$\underset{i\in I_j}{\veebar}\begin{bmatrix} X_{i,r,r'} \\ v_i^{\min}\leq V_{i,r,r'}\leq v_i^{\max} \\ LR_{i,r,r'}=cp_i+vp_iV_{i,r,r'}\end{bmatrix}\veebar\begin{bmatrix} X^{\text{no task}}_{r,r'} \\ V_{i,r,r'}=0,\quad \forall i\in I_j \\ LR_{i,r,r'}=0,\quad \forall i\in I_j\end{bmatrix}\forall j\in J,r,r'|_{r<r'\leq r+\Delta r} \tag{1}$$

The convex hull reformulation of disjunction (1) gives rise to Equations (2–4).

$$\sum_{i\in I_j} X_{i,r,r'}\leq 1,\quad \forall j\in J,r,r'|r<r'\leq r+\Delta r \tag{2}$$

$$v_i^{\min}X_{i,r,r'}\leq V_{i,r,r'}\leq v_i^{\max}X_{i,r,r'}\quad \forall i\in I_j,j\in J,r,r'|_{r<r'\leq r+\Delta r} \tag{3}$$

$$LR_{i,r,r'}=cp_iX_{i,r,r'}+vp_iV_{i,r,r'}\quad \forall i\in I_j,j\in J,r,r'|_{r<r'\leq r+\Delta r} \tag{4}$$

From Figure 3, for each unit only one task can start and finish at each time point, as stated by the logic propositions (5) and (6). Note that due to Equations (5) and (6), constraint (2) becomes redundant. If task $i\in I_j$ is being performed during time interval $[r,r']|_{r+1<r'\leq r+\Delta r}$, the same or other tasks suitable in unit j cannot be processed in any time intervals $[k,r'']|_{r<k\leq r''<t'}$ (Figure 4a) or $[k,r'']|_{r<k<r''<r' \text{or} k<r<r''<r'}$ (Figure 4b) or $[k,r'']|_{r<k<r'<r''}$(Figure 4c), as imposed by the logic proposition (7). Note that if a task is performed in unit j during time intervals $[k,r'']|_{r<k\leq r''<r}$ or $[k,r'']|_{r<k<r''<r' \text{or} k<r<r''<r'}$ or $[k,t'']|_{r<k<r'<r''}$, the same or other tasks cannot be performed in unit j during time interval $[r,r']|_{r+1<r'\leq r+\Delta r}$. These are also satisfied by Equation (7), since from the discrete mathematics we have $(P\Rightarrow \neg Q)\equiv(Q\Rightarrow \neg P)$.

$$\underset{i\in I_j}{\veebar}\ \underset{\substack{r'\in R\\ r<r'\leq r+\Delta r}}{\veebar} X_{i,r,r'}\veebar X^{\text{no }i}_{r,r'}\quad \forall j\in J,r\in R \tag{5}$$

$$\underset{i\in I_j}{\veebar}\ \underset{\substack{r\in R\\ r<r'\leq r+\Delta r}}{\veebar} X_{i,r,r'}\veebar X^{\text{no }i}_{r,r'}\quad \forall j\in J,r'\in R \tag{6}$$

$$X_{i,r,r'}\Rightarrow\neg\left(\bigvee_{\substack{r''\in R\\ k<r''\leq k+\Delta r}} X_{i',k,r''}\right)\forall i,i'\in I_j,r,r',k\in R|r+1<r'\leq r+\Delta r,r+1\leq k,r'\geq k+1 \tag{7}$$

The above logic propositions lead to Equations (8–9).

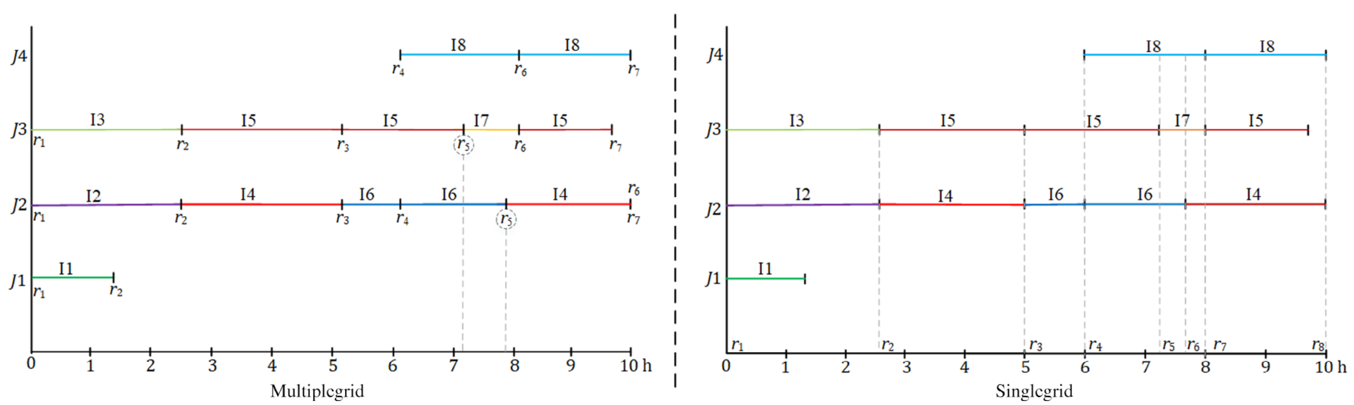

FIGURE 2 Single-grid and multi-grid representations for the proposed continuous-time approach [Color figure can be viewed at wileyonlinelibrary.com]

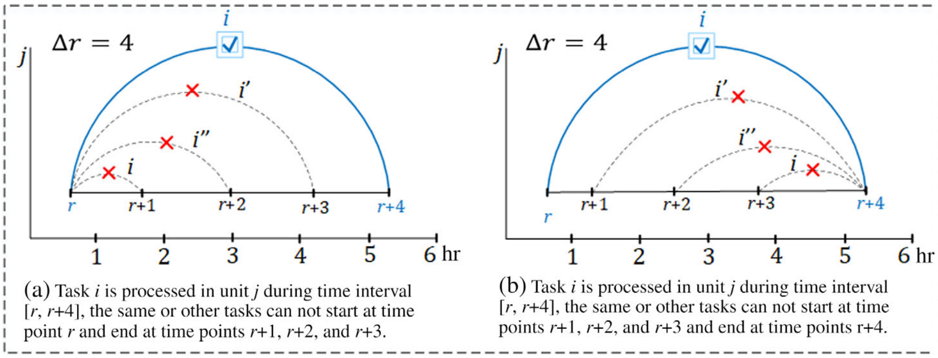

FIGURE 3 In each unit, only one task can start and finish at each time point [Color figure can be viewed at wileyonlinelibrary.com]

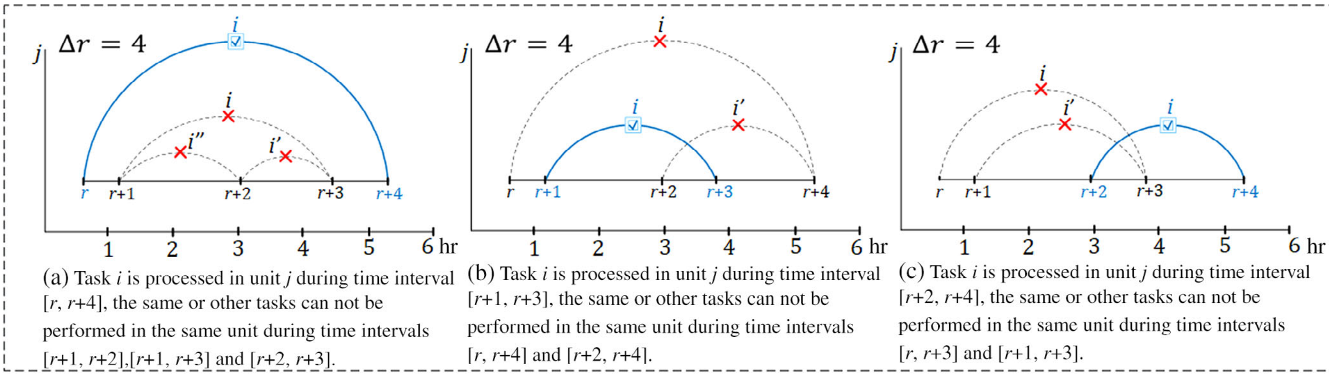

FIGURE 4 Simple example illustrating the need for Equation (10) [Color figure can be viewed at wileyonlinelibrary.com]

$$\sum_{\substack{r\in R\\ r<r'\le r+\Delta r}} \sum_{i\in I_j} X_{i,r,r'} \le 1 \quad \forall j\in J, r'\in R \tag{8}$$

$$\sum_{\substack{r'\in R\\ r<r'\le r+\Delta r}} \sum_{i\in I_j} X_{i,r,r'} \le 1 \quad \forall j\in J, r\in R \tag{9}$$

$$X_{i,r,r'} + \sum_{k<r''\le k+\Delta r} X_{i',k,r''} \le 1 \quad \forall i,i'\in I_j, r,r',k\in R \,|\, r+1<r'\le r+\Delta r, r+1\le k, r'\ge k+1 \tag{10}$$

3.1.2 | Mass balance constraints

Let $F_{s,\,r}$ be the excess amount of state s at time point r. Due to Equation (11), $F_{s,\,r}$ will be equal to $F_{s,\,r-1}$ adjusted by the amounts produced/ consumed by all tasks starting or ending at time point r. The initial inventory of state s is a known parameter, f_s^0. Equation (12) imposes lower and upper bounds on the storage capacity of state s.

$$F_{s,r} = f_s^0|_{r=1} + F_{s,r-1}|_{r>1} + \sum_{i\in I_s^p} \rho^p_{i,s} \sum_{\substack{r'\in R\\ r'<r\le r'+\Delta r}} V_{i,r',r} + \sum_{i\in I_s^c} \rho^c_{i,s} \sum_{\substack{r'\in R\\ r<r'\le r+\Delta r}} V_{i,r,r'} \quad \forall s\in S, r\in R \tag{11}$$

$$f_s^{\min} \le F_{s,r} \le f_s^{\max} \quad \forall s\in S, r\in R \tag{12}$$

3.1.3 | Meeting demand

Demands at states $s \in SM$ storing final products are enforced as a hard constraint in Equation (13) indicating that the amount in the state s at time point $r = |R|$ must be at least as much as d_s, the minimum amount required by the market.

$$F_{s,|R|} \ge d_s, \quad \forall s\in SM \tag{13}$$

3.1.4 | Tightening constraint

Constraints (14) lead to generate tight LP-relaxations for both SG and MG formulations, stating that the summation of the durations of the tasks assigned to a specific unit should be smaller than or equal to the time horizon.

$$\sum_{\substack{r' \in R \\ r < r' \le r + \Delta r}} \sum_{r \in R} \sum_{i \in I_j} LR_{i,r,r'} \le h_{\max} \forall j \in J \tag{14}$$

3.2 | Timing constraints

Timing constraints are very crucial for continuous-time formulations. They can have profound impact on the CPU time and even on the solution quality. These will be demonstrated later in the result section.

3.2.1 | Timing constraints for the MG model

Let $SR_{j,r}$ be the time that unit j starts to perform a task at time point r. As stated by Equation (15), the difference between the start time of unit j at two time points r and r' ($r < r' \le r + \Delta r$) should be either equal to or greater than the processing time of all tasks starting and ending at those same time points in unit j. For different units linked to the same state s, the start of unit j consuming state s, at time point r' should be after the activity duration of unit j' feeding the same state during the time interval $[r,r']|_{r < r' \le r + \Delta r}$, as stated in Equation (16).

$$SR_{j,r'} \ge SR_{j,r} + \sum_{i \in I_j} LR_{i,r,r'} \quad \forall j \in J, r, r'|_{r < r' \le r + \Delta r} \tag{15}$$

$$SR_{j,r'} \ge SR_{j',r} + \sum_{i \in I_{j'}} LR_{i,r,r'} \quad \forall j \in J_s^c, j' \in J_s^p (j \ne j'), s \in S, r, r'|_{r < r' \le r + \Delta r} \tag{16}$$

In short-term scheduling problems, the optimal production is to be obtained for a given time horizon. Thus, the following time horizon constraint should be considered.

$$SR_{j,r} + \sum_{i \in I_j} LR_{i,r,r'} \le h_{\max} \quad \forall j \in J, r, r' \in R|_{r < r' \le r + \Delta r} \tag{17}$$

Similar timing constraints can be defined for tasks instead of units, as imposed by Equations (15') and (16'). Note, however, that these alternative constraints lead to relatively weaker LP relaxations in most cases than Equations (15) and (16).

$$SR_{i,r'} \ge SR_{i',r} + LR_{i',r,r'} \quad \forall i, i' \in I_j, j \in J, r, r'|_{r < r' \le r + \Delta r} \tag{15}$$

$$SR_{i,r'} \ge SR_{i',r} + LR_{i',r,r'}, \quad \forall i \in I_j, i' \in I_{j'}, i \in I_s^c, i' \in I_s^p (i \ne i, j \ne j), s \in S, r, r'|_{r < r' \le r + \Delta r} \tag{16}$$

3.2.2 | Timing constraints for the SG model

Let CR_r be the absolute time of time point r. The difference between CR_r and $CR_{r'}$ ($r < r' \le r + \Delta r$) should be either equal to or greater than the processing time of all tasks starting and ending at those same event points. In addition, the absolute time of each time point must equal to or less than the length of time horizon.

$$CR_{r'} - CR_r \ge LR_{i,r,r'} \quad \forall i, r, r' \in R|_{r < r' \le r + \Delta r} \tag{18}$$

$$CR_r \le h_{\max} \quad \forall r \in R \tag{19}$$

3.3 | Sequence-dependent cleaning for the MG model

Changeovers occur when two different tasks are processed in the same unit in consecutive process operations. The changeovers are often associated with changing the operating conditions or with the cleaning of the equipment. If unit j is processing two different tasks i and i' during time intervals $[r,r']|_{r < r' \le r + \Delta r}$ and $[r'',k]|_{r' \le r'' < k \le r' + \Delta r}$, respectively, there will be a changeover time of $\tau_{i',i}$ at time point r'' in unit j. We have thus the following proposition and disjunction:

$$X_{i,r,r'} \wedge \bigvee_{\substack{k \in R \\ r'' < k \le r'' + \Delta r}} X_{i',r'',k} \Rightarrow X^{\text{changeover}}_{i,i',r,r',r''} \forall i, i' \in I_j (i \ne i'), j \in JM, r, r', r'' \in R|_{r < r' \le r + \Delta r, r' \le r''} \tag{20}$$

$$\bigvee_{i'} \bigvee_{i} \begin{bmatrix} X^{\text{changeover}}_{i,i',r,r',r''} \\ SR_{j,r''} - (SR_{j,r} + LR_{i,r,r'}) \ge \tau_{i',i} \end{bmatrix} \forall i, i' \in I_j (i \ne i'), j \in JM, r, r', r'' \in R|_{r < r' \le r + \Delta r, r' \le r''} \tag{21}$$

The convex hull reformulation of the disjunction and the conversion of the logic expression give rise to the following constraints:

$$X^{\text{changeover}}_{i,i',r,r',r''} \ge X_{i,r,r'} + \sum_{\substack{k \in R \\ r'' < k \le r'' + \Delta r}} X_{i',r'',k} - 1 \forall i, i' \in I_j (i \ne i'), j \in JM, r, r', r'' \in R|_{r < r' \le r + \Delta r, r' \le r''} \tag{22}$$

$$SR_{j,r''} - (SR_{j,r} + LR_{i',r,r'}) \ge \tau_{i',i} X^{\text{changeover}}_{i,i',r,r',r''} \quad \forall i, i' \in I_j (i \ne i'), j \in JM, r, r', r'' \in R|_{r < r' \le r + \Delta r, r' \le r''} \tag{23}$$

We can now eliminate variables $X^{\text{changeover}}_{i,i',r,r',r''}$ from the formulation by combining Equations (22) and (23), giving rise to the following constraint (24):

$$SR_{j,r''} \ge (SR_{j,r} + LR_{i,r,r'}) + \tau_{i',i} \left(X_{i,r,r'} + \sum_{\substack{k \in R \\ r'' < k \le r'' + \Delta r}} X_{i',r'',k} - 1 \right) \forall i, i' \in I_j (i \ne i'), j \in JM, r, r', r'' \in R|_{r < r' \le r + \Delta r, r' \le r''} \tag{24}$$

3.4 | Objective function

Two alternative objective functions will be considered:

1. Maximum revenue. If we assume that all final products are sold at the end of time horizon, the total revenue of the plant is calculated by Equation (25)

$$\max z = \sum_{s \in SM} v_s F_{s,|R|} \tag{25}$$

The SG model for revenue maximization consists of Equations (3, 4, 8–14, 18–19, and 25). The MG model for revenue maximization consists of Equations (3–4, 8–17, 24, and 25).

2. Minimum makespan. The plant must satisfy the amount required by the market in a shorter time.

$$\min z = H \tag{26}$$

where the continuous variable H satisfies the following equations.

$$SR_{j,r} + \sum_{i \in I_j} LR_{i,r,r'} \le H \quad \forall j \in J, r, r' \in R|_{r < r' \le r + \Delta r} \tag{27}$$

$$CR_r \le H \quad \forall r \in R \tag{28}$$

Note that when minimizing the schedule makespan, the time horizon parameter $h_{\max}$ has to be replaced by the continuous variable H in the tightening constraint (14).

The SG model for makespan minimization consists of Equations (3–4, 8–14, 18, 26, and 28). The MG model for makespan minimization consists of Equations (3–4, 8–16, 24, and 26–27).

3.5 | Due date constraints

In this section, we will incorporate intermediate due date constraints into the SG model. The same constraints can be easily extended for the MG model. Let us assume that state requirements linked to the markets ($s \in SM$) need to be met within a few predefined time periods $t \in T$, with h_t standing for the end of time period t (h_0 = 0 h and $h_{|T|} = h_{\max}$). The exclusive disjunction in Equation (31) states that every time point in the grid must locate in exactly one time period $t \in T$. If time point r locates in period t, its absolute time (CR_r) should be within the time interval $[h_{t-1}, h_t]$.

$$\underset{t \in T}{\underline{\bigvee}} \begin{bmatrix} Y_{r,t} \\ h_{t-1} \le CR_r \le h_t \end{bmatrix} \forall r \in R \tag{29}$$

Equations (30 and 31) are obtained using the convex hull reformulation of disjunction (29).

$$\sum_{t \in T} Y_{r,t} = 1, \quad \forall r \in R \tag{30}$$

$$\sum_{t \in T} Y_{r,t} h_{t-1} \le CR_r \le \sum_{t \in T} Y_{r,t} h_t, \forall r \in R \tag{31}$$

If time point r is the last one located in time period t (i.e., $Y_{r,t} = 1$ and $Y_{r+1,t} = 0$), the amount of material stored in state s should be as large as $d_{s,t}$, the minimum amount that the plant should produce during the period t. Otherwise, the slack variables $BR_{s,t}$, representing the backorder of product at state s during the time interval $[h_{t-1}, h_t]$, is activated in disjunction (33) and results in penalty costs. Note that the term $BR_{s,t-1}$ inside the disjunction denotes the shortage of state s during period $t-1$ to be tardily met at period t.

$$Y_{r,t} \wedge \neg Y_{r+1,t} \Rightarrow Y_{r,t}^{\text{last}} \quad \forall t \in T, r \in R \tag{32}$$

$$\underset{t}{\underline{\bigvee}} \begin{bmatrix} Y_{r,t}^{\text{last}} \\ F_{s,r} + BR_{s,t} - BR_{s,t-1} \ge \sum_{t' \in T}^{t} d_{s,t'} \end{bmatrix} \forall s \in SM, r \in R \tag{33}$$

The convex hull reformulation of the disjunction (33) and the conversion of the logic expression (32) give rise to the following constraints:

$$Y_{r,t}^{\text{last}} \ge Y_{r,t} - Y_{r+1,t}, \quad \forall t \in T, r \in R \tag{34}$$

$$F_{s,r} + BR_{s,t} - BR_{s,t-1} \ge \sum_{t' \in T}^{t} d_{s,t'} Y_{r,t}^{\text{last}}, \quad \forall s \in SM, t \in T, r \in R \tag{35}$$

Combining Equations (34) and (35), we have the following constraint (36):

$$F_{s,r} + BR_{s,t} - BR_{s,t-1} \ge \sum_{t' \in T}^{t} d_{s,t'} (Y_{r,t} - Y_{r+1,t}), \forall s \in SM, t \in T, r \in R \tag{36}$$

From constraint (36), one can simply derive Equation (37), leading to a slight reduction in the CPU time.

$$F_{s,r} \ge \sum_{t \in T} (d_{s,t} - BR_{s,t}) \quad \forall s \in SM, r = |R| \tag{37}$$

Equation (38) maximizes the total revenue of the plant while respecting the intermediate due dates needs.

$$\max z = \sum_{s \in SM} v_s F_{s,|R|} - \sum_{t \in T} \sum_{s \in SM} bc_s BR_{s,t} \tag{38}$$

The SG model with the intermediate due date constraints consists of Equations (3, 4, 8–14, 18–19, 30, 31, and 36–38).

TABLE 1 Effect of *optcr* and threads option in GAMS on solution quality and CPU time

	Case 1 (Makespan minimization)					
	S&F[a] ($\|N\|$ = 26, Δn = 0)			MG ($\|R\|$ = 27, Δr = 1)		
Optcr	CPU (s)	MILP (hr)	RMILP (hr)	CPU (s)	MILP (hr)	RMILP (hr)
Option threads = 0						
10^{-1}	3.28	47.01	44.4	1.01	47.49	44.8
10^{-2}	135.46	47.01	44.4	2.67	47.01	44.8
10^{-3}	133.01	47.01	44.4	5.21	47.01	44.8
10^{-4}	221.28	47.01	44.4	7.82	47.01	44.8
10^{-5}	224.86	47.01	44.4	7.23	47.01	44.8
10^{-6}	226.26	47.01	44.4	7.03	47.01	44.8
Option threads = 1						
10^{-1}	1.00	47.64	44.4	0.76	48.55	44.8
10^{-2}	247.56	47.01	44.4	2.95	47.01	44.8
10^{-3}	260.26	47.01	44.4	50.89	47.01	44.8
10^{-4}	275.01	47.01	44.4	64.98	47.01	44.8
10^{-5}	281.82	47.01	44.4	65.29	47.01	44.8
10^{-6}	366.63	47.01	44.4	64.64	47.01	44.8
	Case 2 (profit maximization)					
	S&F[a] ($\|N\|$ = 8, Δn = 1)			**MG ($\|R\|$ = 9, Δr = 2)**		
Optcr	**CPU (s)**	**MILP ($)**	**RMILP ($)**	**CPU (s)**	**MILP ($)**	**RMILP ($)**
Option threads = 0						
10^{-1}	10.96	2,338.7	3,618.6	9.35	2,331.3	3,618.6
10^{-2}	549.5	2,345.3	3,618.6	102.7	2,345.3	3,618.6
10^{-3}	353.7	2,358.2	3,618.6	119.6	2,358.2	3,618.6
10^{-4}	462.4	2,358.2	3,618.6	146.6	2,358.2	3,618.6
10^{-5}	462.7	2,358.2	3,618.6	145.7	2,358.2	3,618.6
10^{-6}	459.1	2,358.2	3,618.6	147.3	2,358.2	3,618.6
Option threads = 1						
10^{-1}	31.9	2,292.5	3,618.6	51.65	2,330.9	3,618.6
10^{-2}	572.6	2,345.3	3,618.6	655.1	2,358.2	3,618.6
10^{-3}	875.4	2,358.2	3,618.6	947.6	2,358.2	3,618.6
10^{-4}	714.8	2,358.2	3,618.6	979.4	2,358.2	3,618.6
10^{-5}	710.7	2,358.2	3,618.6	950.5	2,358.2	3,618.6
10^{-6}	708.4	2,358.2	3,618.6	950.0	2,358.2	3,618.6

[a]Shaik and Floudas.[26]

4 | COMPUTATIONAL RESULTS

The performance of the proposed models (the SG and MG) is now compared to two similar approaches from the literature, that is, to the single-grid model of Maravelias and Grossmann,[18] and to the multi-grid model of Shaik and Floudas.[26] To ensure a fair comparison, all models have been implemented in GAMS 24.9.1 and solved using CPLEX 12.7.1 (using four threads in parallel, i.e., option threads = 0) and the computations are performed without prefixing any binary or continuous variables. The hardware consists of a laptop with an Intel i5-7300U (2.60 GHz, 8 GB of RAM), running Windows 10, 64-bit operating system. The termination criteria were either a relative optimality tolerance of 10^{-3} (based on Table 1) or a maximum computational time of 7,200 s.

We consider two benchmark example problems (Ex1-Ex2) tackled in Shaik et al.[13]The STN representations of these examples can be found in Figure 5. For these example problems we consider two different scheduling horizons (H = 8 hr and H = 10 hr) and three different demand scenarios.

4.1 | Choosing *optcr* and threads options

Optcr or optimality tolerance in GAMS specifies a relative termination tolerance for a global solver.[41] The default option for *optcr* is 10^{-1}

indicating that the objective value will be within 10% of the true objective value. With different adjustments for the *optcr* value, different objective values may be found and the solver will stop as soon as it has found a feasible solution proven to be within the tolerance specified by *optcr*. The *threads* option controls the number of CPU cores to be used by a solver. The default value for *threads* is 1, which means that the solver will use only one core and leave the rest free for other tasks. The solver will use all available cores if *threads* option is set to zero (*option threads* = 0).

To show the effect of *optcr* and *threads* on the CPU time and the solution quality, we solve two cases, both -sharing the STN representation of Ex 2. In the first case, the aim is to meet - product demands at states S12 and S13 (d_{S12}= 750 mass unit (mu), d_{S13}= 750 mu) at a minimum makespan, while the second case aims at maximizing the revenue during the next 10 hr. As can be observed from Table 1, different *optcr* values lead to different CPU times and almost in all cases the smaller *optcr* value, the higher CPU time. Table 1 also indicates that the *threads* option has profound impact on CPU times and can affect the solution quality too. For instance, the proposed model finds a solution of $2,345.3 for Case 2 in 102.7 s with *option threads* = 0 and *option optcr* = 10^{-2}, while it returns the global optimum (2,358.2 $) in 655.1 s with the same optimality tolerance and *option threads* = 1.

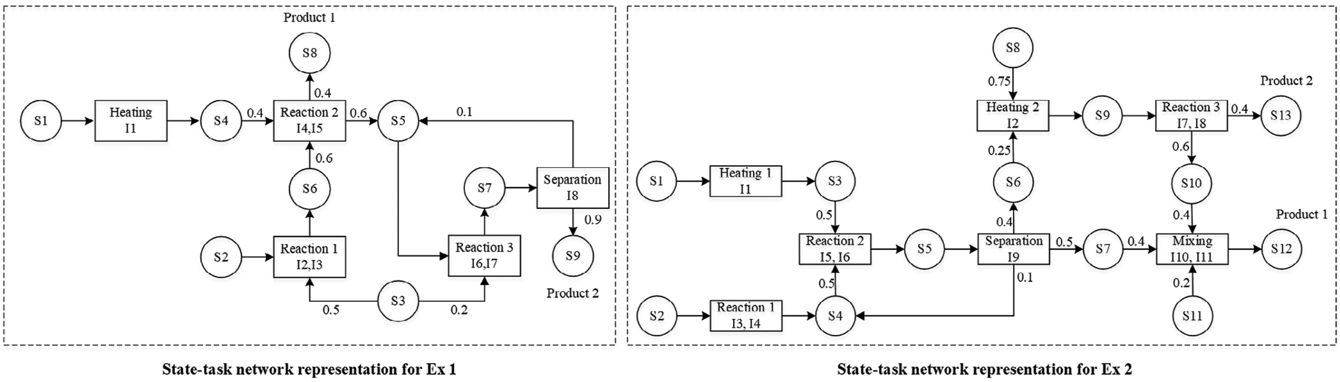

FIGURE 5 State task network representations for Ex1 and Ex2

TABLE 2 Computational results for Ex 1 using the SG model under revenue maximization

	Event points	CPU (s)	Nodes	Nonzeros	Binary variables	Total variables	Eqs.	MILP ($)	RMILP ($)
Ex 1a (*H* = 8 hr)									
M&G	5	0.23	0	3,157	80	496	1,095	1,498.56	1,730.8
SG	5 (Δr = 1)	0.04	0	580	32	147	212	1,498.56	1,730.8
	5 (Δr = 2)	0.06	0	1,112	56	219	368	1,498.56	1,730.8
Ex 1b (*H* = 10 hr)									
M&G	8	25.01	41,023	5,752	128	793	1,719	1,962.69	2,690.5
SG	8 (Δr = 1)	0.17	6,654	1,006	56	249	359	1,860.72	2,775.4
	8 (Δr = 2)	6.50	31,511	2,090	104	393	671	1,958.99	2,775.6
	8 (Δr = 3)	9.45	30,266	3,510	144	513	1,031	1,962.69	2,775.6
	8 (Δr = 4)	10.87	41,209	5,166	176	609	1,399	1,962.69	2,775.6
Ex 2a (*H* = 8 hr)									
M&G	7	7.50	14,730	6,728	154	946	2,076	1,583.44	2,560.6
SG	7 (Δr = 1)	0.50	743	1,198	66	297	436	1,274.48	2,750.9
	7 (Δr = 2)	0.70	1,421	2,403	121	462	781	1,583.44	2,750.9
	7 (Δr = 3)	0.76	1,421	1,379	165	594	1,157	1,583.44	2,750.9
Ex 2b (*H* = 10 hr)									
M&G	10	7,200[a]	3,256,256	10,856	220	1,351	2,934	2,307.66	3,473.9
SG	10 (Δr = 1)	5.87	22,129	1,789	99	438	643	1,963.88	3,618.6
	10 (Δr = 2)	326.92	1,453,542	3,732	187	702	1,195	2,156.36	3,618.6
	10 (Δr = 3)	148.4	509,111	6,379	264	933	1,853	2,307.66	3,618.6
	10 (Δr = 4)	188.29	457,597	9,755	330	1,131	2,567	2,307.66	3,618.6

[a]Relative gap (RG) = 8.67%.

TABLE 3 Computational results for Ex 1 using the SG model under makespan minimization

	Event points	CPU (s)	Nodes	Nonzeros	Binary variables	Total variables	Eqs.	MILP (hr)	RMILP (hr)
Ex 1c (d_{S8}= 200 mu, d_{S9}= 200 mu)									
M&G	10	7.06	3,687	7,968	160	992	2,137	19.34	18.68
SG	10 (Δr = 1)	8.65	29,164	1,368	72	318	459	19.78	18.68
	10 (Δr = 2)	1.39	3,451	2,884	136	510	875	19.34	18.68
	10 (Δr = 3)	1.79	3,585	4,968	192	678	1,379	19.34	18.68
Ex 1d (d_{S8}= 500 mu, d_{S9}= 400 mu)									
M&G	20	7,200[a]	342,956	21,578	320	1,982	4,217	46.52	45.57
SG	20 (Δr = 1)	7,200[b]	4,777,385	2,868	152	658	949	46.52	45.57
Ex 1e (d_{S8}= 600 mu, d_{S9}= 600 mu)									
M&G	25	7,200[c]	231,955	30,483	400	2,477	5,257	56.81	56.05
SG	25 (Δr = 1)	7,200[c]	985,756	3,618	192	828	1,194	56.81	56.05
Ex 2c (d_{S12}= 100 mu, d_{S13}= 200 mu)									
M&G	11	71.1	44,575	12,226	242	1,487	2,980	13.36	11.33
SG	11 (Δr = 1)	0.62	2,034	1,994	110	486	714	14.61	11.25
	11 (Δr = 2)	6.09	20,582	4,183	209	783	1,335	13.53	11.25
	11 (Δr = 3)	11.07	31,419	7,226	297	1,048	2,087	13.36	11.25
	11 (Δr = 4)	13.03	32,315	11,223	374	1,278	2,920	13.36	11.25
Ex 2d (d_{S12}= 250 mu, d_{S13}= 250 mu)									
M&G	12	23.60	5,665	14,357	264	1,622	3,508	17.02	14.40
SG	12 (Δr = 1)	0.79	3,083	2,191	121	533	783	18.97	14.27
	12 (Δr = 1)	1.36	2,475	4,626	231	863	1,473	17.02	14.27
	12 (Δr = 3)	1.78	2,529	8,065	330	1,160	2,319	17.02	14.27
Ex 2e (d_{S12}= 930 mu, d_{S13}= 840 mu)									
M&G	29	691.4	32,276	54,749	638	3,917	8,370	51.82	50.92
SG	29 (Δr = 1)	12.43	17,780	5,540	308	1,332	1,956	59.74	49.92
	29 (Δr = 2)	49.28	29,507	12,157	605	2,223	3,819	51.82	49.92
	29 (Δr = 3)	110.21	38,830	22,328	891	3,081	6,263	51.82	49.92

[a]RG = 2.01%.
[b]RG = 0.79%.
[c]RG = 1.15%.

4.2 | Comparison of the SG to a previous single-grid approach

In this section, we compare the performance of the proposed SG model to the single-grid model of Maravelias and Grossmann,[18] hereafter referred to as M&G. From Tables 2 and 3, one can see that both formulations require the same number of time points to find the same optimal solution. However, the proposed model always requires fewer continuous variables (but slightly more binary variables), constraints and often spends lower CPU times finding the optimal solutions. In turn, M&G exhibits tighter LP relaxations (RMILP) in most cases. Table 2 gives the model and solution statistics of the SG and M&G models for the case of revenue maximization. In Ex 1a (H = 8 hr), Ex 2a (H = 10 hr) and Ex 2a (H = 8 hr), both the SG and M&G models perform equally well. The proposed SG model excels for Ex 3b, being 10 times faster to find a solution worth 2,307.66 $ in 669.48 s (5.87 + 326.92 + 148.4 + 188.29) with 10 event points. Note that unlike M&G that needs only a single tuning parameter (i.e., the number of time points in the grid), the proposed model requires two tuning parameters, i.e., the number of time points and Δr. At each event point, we need to iterate over Δr to discover the global optimal solution.

When minimizing the schedule makespan, we need to specify the big-M value for those constraints in M&G involving big-M parameters. Similar to the literature, we use M = 50 in Ex 1c and Ex 2c and M = 100 in Ex 1d, Ex 1e, Ex 2d, and Ex 2e (these are also applied to the -multi-grid model of Shaik and Floudas[26]). Table 3 summarizes the model and solution statistics of the SG and M&G models for the case of makespan minimization. In Ex 2c, our SG model solves slightly slower than M&G (8.65 + 1.39 + 1.79 = 11.83 s vs. 7.06 s). In Ex 1d and Ex 1e, both SG and M&G perform equally and cannot close the gap in 2 hr while generating the same optimal solution at termination. For Ex 2c, Ex 2d, and Ex 2e, though the SG model requires more

TABLE 4 Computational results for Ex 2 using the MG model under revenue maximization

	Event points	CPU (s)	Nodes	Nonzeros	Binary variables	Total variables	Eqs.	MILP ($)	RMILP ($)
Ex 1a (H = 8 hr)									
S&F	4 (Δn = 0)	0.14	0	722	32	165	307	1,498.56	1,730.8
	5 (Δn = 0)	0.32	418	921	40	206	390	1,498.56	2,083.9
	5 (Δn = 1)	0.54	886	2,217	72	270	754	1,498.56	2,123.3
MG	5 (Δr = 1)	0.10	0	704	32	162	250	1,498.56	1,730.8
	6 (Δr = 1)	0.26	0	877	40	199	309	1,498.56	2,019.3
	6 (Δr = 2)	0.42	541	1,717	72	295	557	1,498.56	2,091.1
Ex 1b (H = 10 hr)									
S&F	6 (Δn = 0)	0.76	1,997	1,120	48	247	473	1,943.16	2,727.1
	7 (Δn = 0)	6.29	36,996	1,319	56	288	556	1,943.16	2,775.4
	7 (Δn = 1)	61.23	256,043	3,233	104	384	1,088	1,962.69	2,780.2
	8 (Δn = 0)	27.39	140,520	1,518	64	329	639	1,943.16	2,804.9
	8 (Δn = 1)	1,829.87	3,500,606	3,741	120	441	1,255	1,962.69	2,805.4
MG	7 (Δr = 1)	0.46	978	1,066	48	236	368	1,912.87	2,628.4
	8 (Δr = 1)	0.82	5,211	1,242	56	273	427	1,912.87	2,732.0
	8 (Δr = 2)	17.67	75,429	2,530	104	417	799	1,962.69	2,770.9
	9 (Δr = 1)	4.81	25,696	1,418	64	310	486	1,912.87	2,775.3
	9 (Δr = 2)	302.12	908,315	2,881	120	478	920	1,962.69	2,804.2
Ex 2a (H = 8 hr)									
S&F	5 (Δn = 0)	0.21	310	1,280	55	286	545	1,583.44	2,100.0
	6 (Δn = 0)	1.15	4,911	1,557	66	343	661	1,583.44	2,750.9
	6 (Δn = 1)	8.82	41,801	3,727	121	453	1,280	1,583.44	2,750.9
MG	6 (Δr = 1)	0.28	402	1,289	55	280	455	1,583.44	2,100.0
	7 (Δr = 1)	0.64	2,725	1,544	66	332	542	1,583.44	2,563.2
	7 (Δr = 2)	3.73	21,764	3,039	121	497	977	1,583.44	2,682.0
Ex 2b (H = 10 hr)									
S&F	6 (Δn = 0)	0.57	1,161	1,557	66	343	661	2,292.54	2,871.8
	7 (Δn = 0)	3.87	14,793	1,834	77	400	777	2,293.46	3,369.6
	7 (Δn = 1)	17.04	51,931	4,222	143	532	1,512	2,345.30	3,369.6
	8 (Δn = 0)	25.01	117,362	2,111	88	457	893	2,293.46	3,618.6
	8 (Δn = 1)	384.12	1,328,510	5,117	165	611	1,744	2,358.20	3,618.6
	8 (Δn = 2)	1,570.12	2,896,294	7,075	231	743	2,008	2,358.20	3,618.6
MG	7 (Δr = 1)	0.35	1,035	1,544	66	332	542	2,292.54	2,853.9
	8 (Δr = 1)	1.67	6,886	1,799	77	384	629	2,292.54	3,204.6
	8 (Δr = 2)	5.46	22,490	3,598	143	582	1,151	2,345.30	3,325.6
	9 (Δr = 1)	11.17	33,576	2,054	88	436	716	2,292.54	3,424.58
	9 (Δr = 2)	118.56	525,135	4,157	165	667	1,325	2,358.20	3,618.64
	9 (Δr = 3)	321.20	839,726	6,756	231	865	1,997	2,358.20	3,618.64

binary variables, it performs better and solves a few times faster than M&G.

4.3 | Comparison of MG to a previous -multi-grid model

In this section, we compare the performance of the proposed MG model to perhaps one of the most efficient -multi-grid scheduling models in the literature, that is, to the event-based model by Shaik and Floudas,[18] hereafter referred to as S&F. For the sake of a fair comparison, we have removed the big-M term in constraint (16) in S&F, which makes it exactly equivalent to the model by Vooradi and Shaik.[27] In general, MG always needs one more event point to find the optimal solutions, but roughly the same number of total variables and fewer constraints and often leads to tighter LP relaxations. Similar to the proposed model, S&F also uses a parameter Δn

TABLE 5 Computational results for Ex 2 using the MG model under makespan minimization

	Event points	CPU (s)	Nodes	Nonzeros	Binary variables	Total variables	Eqs.	MILP (hr)	RMILP (hr)
Ex 1c (d_{S8}= 200 mu, d_{S9}= 200 mu)									
S&F	9 (Δn = 0)	1.10	3,984	1,722	72	371	652	19.34	18.68
	10 (Δn = 0)	1.54	2,511	1,921	80	412	727	19.34	18.68
	10 (Δn = 1)	1.93	3,890	4,762	152	556	1,511	19.34	18.68
MG	10 (Δr = 1)	1.09	4,112	1,635	72	348	547	19.34	18.68
	11 (Δr = 1)	1.20	4,044	1,815	80	385	606	19.34	18.68
	11 (Δr = 2)	1.64	4,013	3,793	152	601	1,164	19.34	18.68
Ex 1d (d_{S8}= 500 mu, d_{S9}= 400 mu)									
S&F	20 (Δn = 0)	0.59	0	3,911	160	822	1,477	46.11	45.57
	21 (Δn = 0)	2.34	2,670	4,110	168	863	1,552	46.11	45.57
	21 (Δn = 1)	6.14	2,610	10,350	328	1,183	3,260	46.11	45.57
MG	21 (Δr = 1)	1.48	2,631	3,615	160	755	1,196	46.11	45.57
	22 (Δr = 1)	1.92	3,840	3,795	168	792	1,255	46.11	45.57
	22 (Δr = 2)	3.62	2,741	8,215	328	1,272	2,495	46.11	45.57
Ex 1e (d_{S8}= 600 mu, d_{S9}= 600 mu)									
S&F	25 (Δn = 0)	3.17	2,806	4,906	200	1,027	1,852	56.68	56.05
	26 (Δn = 0)	2.61	2,606	5,105	208	1,068	1,927	56.68	56.05
	26 (Δn = 1)	10.37	3,244	12,890	408	1,468	4,055	56.68	56.05
MG	26 (Δr = 1)	2.25	2,949	4,515	200	940	1,491	56.68	56.05
	27 (Δr = 1)	2.70	2,688	4,695	208	977	1,550	56.68	56.05
	27 (Δr = 2)	4.73	2,565	10,225	408	1,577	3,100	56.68	56.05
Ex 2c (d_{S12}= 100 mu, d_{S13}= 200 mu)									
S&F	7 (Δn = 0)	0.21	130	1,841	77	401	702	13.36	11.25
	8 (Δn = 0)	0.64	1,580	2,118	88	458	807	13.36	11.25
	8 (Δn = 1)	5.21	9,015	5,124	165	612	1,658	13.36	11.25
MG	8 (Δr = 1)	0.34	80	1,848	77	385	631	13.36	12.31
	9 (Δr = 1)	0.54	760	2,109	88	437	718	13.36	11.62
	9 (Δr = 2)	1.18	3,621	4,254	165	668	1,327	13.36	11.25
Ex 2d (d_{S12}= 250 mu, d_{S13}= 250 mu)									
S&F	10 (Δn = 0)	0.46	238	2,672	110	572	1,017	17.02	14.27
	11 (Δn = 0)	0.90	627	2,949	121	629	1,122	17.02	14.27
	11 (Δn = 1)	6.64	9,739	7,209	231	849	2,321	17.02	14.27
MG	11 (Δr = 1)	0.45	56	2,631	110	541	892	17.02	14.53
	12 (Δr = 1)	1.12	1,303	2,892	121	593	979	17.02	14.39
	12 (Δr = 2)	2.29	3,274	5,967	231	923	1,849	17.02	14.27
Ex 2e (d_{S12}= 930 mu, d_{S13}= 840 mu)									
S&F	28 (Δn = 0)	73.18	54,090	7,658	308	1,598	2,907	51.82	49.92
	29 (Δn = 0)	1,013	704,068	7,935	319	1,655	3,012	51.82	49.92
	29 (Δn = 1)	7,200[a]	1,739,494	19,716	627	2,271	6,299	51.82	49.92
MG	29 (Δr = 1)	0.90	54	7,329	308	1,477	2,458	51.82	50.92
	30 (Δr = 1)	5.35	3,783	7,590	319	1,529	2,545	51.82	50.26
	30 (Δr = 2)	22.89	14,579	16,245	627	2,453	4,981	51.82	49.92

[a]RG = 3.68%.

(=0, 1, 2,...) that defines a limit on the maximum number of events over which a task can span. However, as the proposed model is a slot-based approach and not a event-based model, a task that starts at a time point does not end at the same point, leading to a minimum value of one for the parameter Δr (=1, 2, 3,...) in our model.

TABLE 6 Results for Ex 1b and Ex 2b with two alternative timing constraints for the MG

	Event points	CPU (s)	Nodes	Nonzeros	Binary variables	Total variables	Eqs.	MILP ($)	RMILP ($)
Ex 1b (H = 10 hr)									
S&F	7 (Δn = 0)	6.29	36,996	1,319	56	288	556	*1,943.16*	2,775.4
	7 (Δn = 1)	61.23	256,043	3,233	104	384	1,088	1,962.69	2,780.2
	7 (Δn = 2)	89.87	291,704	4,154	144	464	1,244	1,962.64	2,780.2
MGT[a]	8 (Δr = 1)	5.95	26,491	1,578	56	305	567	*1,943.16*	2,775.4
	8 (Δr = 2)	56.64	184,206	3,154	104	449	1,059	1,962.64	2,775.4
	8 (Δr = 3)	46.96	103,278	4,984	144	569	1,569	1,962.64	2,775.4
MGU[b]	8 (Δr = 1)	0.82	5,211	1,242	56	273	427	*1,912.87*	2,732.0
	8 (Δr = 2)	17.67	75,429	2,530	104	417	799	1,962.69	2,770.9
	8 (Δr = 3)	28.03	77,534	4,120	144	537	1,209	1,962.69	2,770.9
Ex 2b (H = 10 hr)									
S&F	8 (Δn = 0)	25.01	117,362	2,111	88	457	893	2,293.46	3,618.6
	8 (Δn = 1)	384.12	1,328,510	5,117	165	611	1,744	2,358.20	3,618.6
	8 (Δn = 2)	1,570.12	2,896,294	7,075	231	743	2,008	2,358.20	3,618.6
MGT[a]	9 (Δr = 1)	24.61	133,841	2,510	88	481	908	2,293.46	3,618.64
	9 (Δr = 2)	202.12	864,354	5,012	165	712	1,685	2,358.20	3,618.64
	9 (Δr = 3)	610.40	1,531,723	7,953	231	910	2,501	2,358.20	3,618.64
MGU[b]	9 (Δr = 1)	11.17	33,576	2,054	88	436	716	2,292.54	3,424.58
	9 (Δr = 2)	118.56	525,135	4,157	165	667	1,325	2,358.20	3,618.64
	9 (Δr = 3)	321.20	839,726	6,756	231	865	1,997	2,358.20	3,618.64

[a]MG with constraints (15') and (16').
[b]MG with constraints (15) and (16).

TABLE 7 Sequence-dependent changeover times used in Ex1

	Changeover time (hr)					
Task	I2	I3	I4	I5	I6	I7
I2	0	-	0.2	-	0.4	-
I3	-	0		0.2	-	0.3
I4	0.2	-	0	-	0.5	-
I5	-	0.1	-	0	-	0.2
I6	0.1	-	0.4	-	0	-
I7	-	0.3	-	0.5	-	0

Table 4 shows the results for the MG and S&F models when maximizing the revenue. For Ex 1a, both MG and S&F perform equally well, but the proposed model needs fewer nodes and nonzero elements. In Ex 1b, the MG confirms the solution (with nine events point and Δr = 2) in 302.12 s of CPU while S&F proves the optimality in 1,829 s. Results for Ex 2a are roughly similar, but the MG leads to a tighter LP relaxation. The proposed MG model excels for Ex 2b, being able to prove the optimum in 321.20 s versus 1,570.12 s required by S&F.

Table 5 gives the computational results for MG and S&F when minimizing the schedule makespan. For examples Ex 1c, Ex 1d, Ex 1e, Ex 2c, and Ex 2d, both the MG and S&F perform roughly similar in terms of CPU time, but the proposed model requires fewer nonzero elements and constraints. In Ex 3e, the proposed MG model surprisingly outperforms S&F. When tasks cannot span, S&F spends 73.18 s finding a solution worth 51.82 hr with 28 event points whereas the proposed model finds the same solution in 0.90 s. Considering one more time point, S&F requires 1,013 s to confirm the solution obtained with 28 time points, while the proposed model proves the solution in 5.35 s. For the case that tasks can last more than one time slot (Δn = 1 and Δr = 2) the proposed model finds the same makespan of 51.82 hr in 22.89 s with 30 time points whereas S&F cannot close gap in 2 hr with 29 events.

4.4 | Alternative timing constraints (Ex 1b, Ex 2b)

As already mentioned, timing constraints are very crucial for continuous-time formulations and they can have profound impact on the CPU time and even on the solution quality. As can be observed from Tables 4 and 5, the MG model always needs one more event point compared to S&F. In other words, if S&F discovers the global optimum with n event points, the proposed MG model will require n + 1 event points to generate the same optimum. However, with the same task spanning limits, they may result in different solutions. This can be - seen from the results of Ex 1b in Table 4, where with seven event points and when tasks cannot span (Δn = 0), S&F generates a solution worth $1,943.16, whereas the MG model yields a solution of $1,912.87 with eight event points and Δr = 1 (the same happens for Ex 2b). Considering the timing constraints for tasks instead of units (i.e., Equations (15') and (16')), the MG model results in the same solution as S&F generates with the same task spanning limits (see

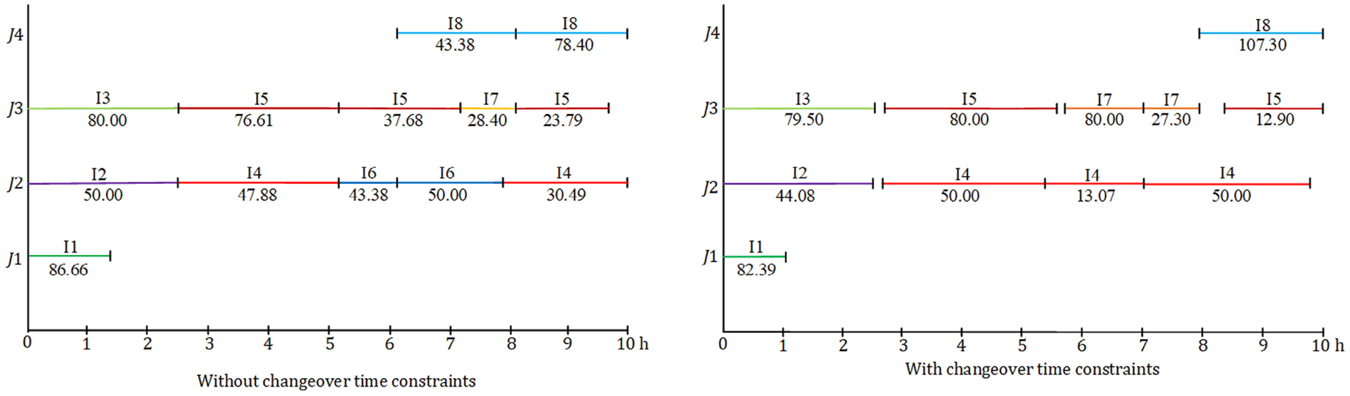

FIGURE 6 Gantt chart for Ex 1b with and without changeover time constraints using seven event points (**Δr = 2**) [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 8 Results for Ex 2a with and without changeover time constraints

	Event points	CPU (s)	Nodes	Nonzeros	Binary variables	Total variables	Eqs.	MILP ($)	RMILP ($)
Ex 1b (*H* = 10 hr)									
MG1[a]	6 (Δr = 1)	0.31	266	1,730	40	199	489	1,765.75	2,397.16
	6 (Δr = 2)	0.51	474	3,242	72	295	857	1,765.75	2,419.69
	7 (Δr = 1)	0.70	1,693	2,254	48	236	620	1,765.78	2,584.49
	7 (Δr = 2)	3.50	12,355	4,356	88	356	1,110	1,789.70	2,717.17
	7 (Δr = 3)	4.79	14,511	6,319	120	452	1,558	1,789.70	2,721.21
	8 (Δr = 1)	4.10	15,370	2,838	56	273	763	1,765.75	2,691.46
	8 (Δr = 2)	44.90	153,694	1,242	104	417	1,387	1,789.70	2,769.02
MG2[b]	6 (Δr = 1)	0.15	0	890	40	199	309	1,912.87	2,436.69
	6 (Δr = 2)	0.26	157	1,742	72	295	557	1,912.87	2,436.69
	7 (Δr = 1)	0.46	978	1,066	48	236	368	1,912.87	2,628.45
	7 (Δr = 2)	1.15	2,989	2,136	88	356	678	1,962.69	2,722.23
	7 (Δr = 3)	0.96	2,930	3,388	120	452	1,006	1,962.69	2,742.19
	8 (Δr = 1)	0.82	5,211	1,242	56	273	427	1,912.87	2,732.06
	8 (Δr = 2)	17.67	75,429	2,530	104	417	799	1,962.69	2,770.92

[a]MG with changeover time constraints.
[b]MG without changeover constraints.

Table 6). However, these constraints lead to a slight increment in the number of continuous variables and constraints and to relatively poor LP relaxations in most cases. As can be seen from Table 6, replacing constraints (15′) and (16′) by (15) and (16) roughly doubles the CPU times in all cases.

4.5 | Changeover time (Ex 1b)

Here we solve Ex 1b with changeover time constraints to show how they can affect the solution quality and - CPU time. Table 7 shows the sequence-dependent changeover times considered in Ex 1. The changeover time constraints are very important in practice and all scheduling models should consider them rigorously; otherwise, the returned solutions may not be acceptable to the users. This can be

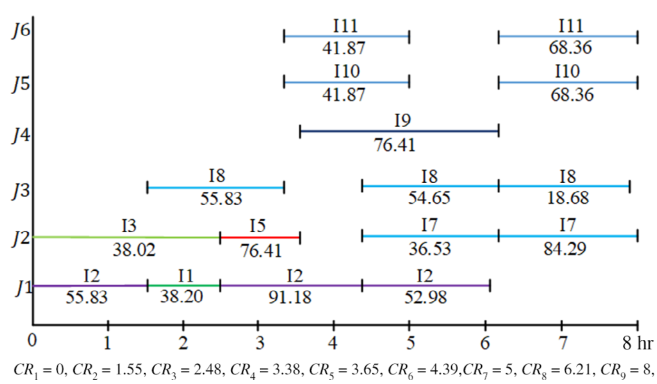

FIGURE 7 Gantt chart for Ex 2b with intermediate due dates using nine event points (**Δr = 3**) [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 9 Results for Ex 2a with intermediate due dates constraints

$\|R\|$	CPU (s)	Nodes	Nonzero	Binary variables	Total variables	Eqs	MILP ($)	RMILP ($)
9(Δr = 1)	12.26	48,784	1,794	106	416	639	604.1	2,940.1
9(Δr = 2)	122.92	506,609	3,463	183	647	1,122	986.7	2,940.1
9(Δr = 3)	101.96	465,398	5,690	249	845	1,686	1,152.9	2,940.1
10(Δr = 1)	34.92	122,887	2,012	119	465	715	604.1	2,940.1
10(Δr = 2)	676.43	1,890,415	3,923	207	729	1,267	986.7	2,940.1
10(Δr = 3)	2,469.5	4,428,925	6,542	284	960	1,925	1,152.9	2,940.1

realized from the right-hand side of Figure 6. Reaction 3 (tasks I6, I7) that has been processed twice in unit *J*2 and once in *J*3, now with the changeover time constraints is processed only twice in unit *J*3. The computational results for Ex 1b with and without the changeover time constraints are all summarized in Table 8. It can be seen that apart from poor solutions, with the changeover time constraints the MG model becomes slower, which can be caused due to almost 40% increases in the number of constraints.

4.6 | Ex 2a with intermediate due dates

Ex 2a is tackled again here, but this time the requirements of states S12 and S13 should be met within two due dates T1 = 5 hr and T2 = 8 hr. We consider $d_{S12,\ T1}$ = 80 mu, $d_{S13,\ T1}$ = 60 mu, $d_{S12,\ T2}$ = 200 mu and $d_{S13,\ T2}$ = 60 mu, and $bc_{s,\ t}$ = 2 $/mu. The Gantt chart schedule for Ex 2a using nine time points (Δr = 3) is given in Figure 7. It can be observed that time point 7 (CR_7 = 5 hr) is the last one located in the first period. Therefore, at this time point the amount of material stored at states S12 and S13 should be as large as 80 and 60 mu, respectively. Otherwise, backordered costs will be raised. Analyzing Figure 7, one can simply derive that, tasks I10 and I11 produce 83.75 mu (= 41.87 + 41.87) for state S12 from time 3.38 to 5 hr. At the end of period 2, the total amount stored at state S12 should be at least 280 (80 + 200) mu. However, accodring to Figure 7, state S12 faces 59.47 mu shortages of P1 in the second period. This is because the total amount that tasks I10 and I11 produce for the state S12 during the time horizon is 220.52 mu (=41.87 + 41.87 + 68.36 + 68.36). Backorders also cannot be avoided at state S13, that is, 37.66 mu in the first period and 40 mu in the second period. Model statistics and computational results for Ex 2a with intermediate due dates are all summarized in Table 9.

5 | CONCLUSIONS

This article presented an MILP formulation based on the STN for the short-term scheduling of multipurpose batch plants with changeover time and intermediate due dates constraints. It employed a continuous-time representation with both single and multiple reference grids. We used the high-level construct of GDP to associate simple linear constraints to each decision variable, which were then converted into MILP format using convex hull reformulations. We also rely on logical propositions to formulate constraints featuring solely binary variables. The proposed MILP formulations have been tested using a set of benchmark problems from the literature and results were compared to two previous published formulations. The results demonstrate that the proposed MG model leads to substantial reductions in solution times, problem sizes, and to tighter linear relaxations compared to a state-of-the-art event-driven approach. The single-grid counterpart of the formulation is not as favorable, as it exhibits relatively weaker linear relaxations and is not capable of handling the changeover time constraints. The main disadvantage of the proposed approach is that it requires two tuning parameters that may hinder finding the global optimal solution.

ACKNOWLEDGMENT

Financial support is gratefully acknowledged from the Academy of Finland project “SINGPRO,” Decision No. 313466.

NOTATION

Sets/Indices

I/i	tasks
J/j	units
S/s	states
R/r	time points
T/t	time periods
I_j	tasks that can be processed in unit j
I_s^p	tasks which produce state s
I_s^c	tasks which consume state s
J_s^p	units which feed state s
J_s^c	units which is fed by state s
SM	states that feed local markets (final states)
JM	units that can process more than one task

Parameters

$h_{\max}$	time horizon (hr)
h_t	time at which time period t ends (hr)
cp_i	fixed duration of task i (hr)
vp_i	variable duration of task i (hr)
$\tau_{i,i'}$	sequence-dependent changeover time when task i is followed by task i' (hr)
$\rho^p_{i,s(>0)}$	proportion of state s produced by task i
$\rho^c_{i,s(<0)}$	proportion of state s consumed by task i

v_i^{max}	maximum batch size for task *i* (mu)
v_i^{min}	minimum batch size for task *i* (mu)
f_s^0	initial amount of state *s* (mu)
f_s^{max}	maximum storage capacity of state *s* (mu)
f_s^{min}	minimum storage capacity of state *s* (mu)
d_s	minimum amount needed by state *s* at the end of the time horizon (mu)
$d_{s,\,t}$	minimum demand of state *s* due at period *t* (mu)
v_s	price of state *s* ($)
$cs_{i,i'}$	changeover cost when task *i* is followed by task *i′* ($)
$bc_{s,\,t}$	unit backorder cost in state *s* during time period *t* ($)
Variables	
$X_{i,\,r,\,r'}$	binary variable indicating that task *i* is being processed during time interval [*r*, *r′*]
$Y_{r,\,t}$	binary variable indicating that time point *r* locates in time period *t*
$Y_{r,t}^{last}$	true if time point *r* is the last one located in time period *t*
$V_{i,r,r'}$	batch size of task *i* during time interval [*r*, *r′*]
$F_{s,\,r}$	excess amount of state *r* at time point *r*
$SR_{j,\,r}$	time at which task *j* starts at time point *r*
$SR_{i,\,r}$	time at which task *i* starts at time point *r*
$LR_{i,r,r'}$	processing time of task *i* during time interval [*r*, *r′*]
CR_r	absolute time of time point *r*
$BR_{s,\,t}$	backorder in state $s \in SM$ during time period *t*
H	makespan

ORCID

Iiro Harjunkoski https://orcid.org/0000-0003-0428-0751

REFERENCES

1. Reklaitis GV. Overview of scheduling and planning of batch process operations. Paper presented at: NATO Advanced Study Institutes-Batch Processing Systems Engineering, Antalya, Turkey; 1992.
2. Pantelides CC. Unified frameworks for the optimal process planning and scheduling. Paper presented at: Proceedings on the Second Conference on Foundations of Computer Aided Process Operations; AIChE, 1994; New York:253-274.
3. Floudas CA, Lin X. Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review. *Comput Chem Eng*. 2004;28:2109-2129.
4. Mendez CA, Cerda J, Grossmann IE, Harjunkoski I, Fahl M. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Comput Chem Eng*. 2006;30:913-946.
5. Maravelias CT. General framework and modeling approach classification for chemical production scheduling. *AIChE J*. 2012;58:1812-1828.
6. Kondili E, Pantelides CC, Sargent R. A general algorithm for short-term scheduling of batch operations-I. MILP formulation. *Comput Chem Eng*. 1993;17:211-227.
7. Shah N, Pantelides CC, Sargent RWH. A general algorithm for short-term scheduling of batch operations. Part II. Computational issues. *Comput Chem Eng*. 1993;17:229-244.
8. Pinto JM, Grossmann IE. Optimal cyclic scheduling of multistage continuous multiproduct plants. *Comput Chem Eng*. 1994;18:797-816.
9. Pinto JM, Grossmann IE. A continuous time mixed integer linear programming model for short term scheduling of multistage batch plants. *Ind Eng Chem Res*. 1995;34:3037-3051.
10. Karimi LA, McDonald CM. Planning and scheduling of parallel semicontinuous processes. Part 2. Short-term scheduling. *Ind Eng Chem Res*. 1997;36:2701-2714.
11. Lamba N, Karimi LA. Scheduling parallel production lines with resource constraints. Part 1. Model formulation. *Ind Eng Chem Res*. 2002;41:779-789.
12. Mostafaei H, Castro PM. Continuous-time scheduling formulation for straight pipelines. *AIChE J*. 2017;63:1923-1936.
13. Shaik MA, Janak SL, Floudas CA. Continuous-time models for short-term scheduling of multipurpose batch plants: a comparative study. *Ind Eng Chem Res*. 2006;45:6190-6209.
14. Harjunkoski I, Maravelias C, Bongers P, et al. Scope for industrial applications of production scheduling models and solution methods. *Comput Chem Eng*. 2014;62:161-193.
15. Maravelias CT, Grossmann IE. On the relation of continuous- and discrete–time state–task network formulations. *AIChE J*. 2006;52:843-849.
16. Schilling G, Pantelides CC. A simple continuous-time process scheduling formulation and a novel solution algorithm. *Comput Chem Eng*. 1996;20:1221-1226.
17. Castro PM, Barbosa-Povoa AP, Matos H. An improved RTN continuous-time formulation for the short-term scheduling of multipurpose batch plants. *Ind Eng Chem Res*. 2001;40:2059-2068.
18. Maravelias CT, Grossmann IE. New general continuous-time state-task network formulation for short-term scheduling of multipurpose batch plants. *Ind Eng Chem Res*. 2003;42:3056-3074.
19. Castro PM, Barbosa-Povoa AP, Matos HA, Novais AQ. Simple continuous-time formulation for short-term scheduling of batch and continuous processes. *Ind Eng Chem Res*. 2004;43:105-118.
20. Sundaramoorthy A, Karimi IA. A simpler better slot-based continuous time formulation for short-term scheduling in multipurpose batch plants. *Chem Eng Sci*. 2005;60:2679-2702.
21. Ierapetritou MG, Floudas CA. Effective continuous-time formulation for short-term scheduling. 1. Multipurpose batch processes. *Ind Eng Chem Res*. 1998;37:4341-4359.
22. Ierapetritou MG, Floudas CA. Effective continuous- time formulation for short-term scheduling. 2. Continuous and semicontinuous processes. *Ind Eng Chem Res*. 1998;37:4360-4374.
23. Ierapetritou MG, Hene TS, Floudas CA. Effective continuous-time formulation for short-term scheduling. 3. Multiple intermediate due dates. *Ind Eng Chem Res*. 1999;38:3446-3461.
24. Janak SL, Floudas CA. Improving unit-specific event based continuous time approaches for batch processes: Integrality gap and task splitting. *Comput Chem Eng*. 2008;32:913-955.
25. Shaik MA, Floudas CA. Unit-specific event-based continuous-time approach for short-term scheduling of batch plants using RTN framework. *Comput Chem Eng*. 2008;32:260-274.
26. Shaik MA, Floudas CA. Novel unified modeling approach for short-term scheduling. *Ind Eng Chem Res*. 2009;48:2947-2964.
27. Vooradi R, Shaik MA. Improved three-index unit-specific event-based model for short-term scheduling of batch plants. *Comput Chem Eng*. 2012;43:148-172.
28. Shaik MA, Vooradi R. Short-term scheduling of batch plants: reformulation for handling material transfer at the same event. *Ind Eng Chem Res*. 2017;56:11175-11185.
29. Susarla N, Li J, Karimi IA. A novel approach to scheduling multipurpose batch plants using unit slots. *AIChE J*. 2010;56:1859.
30. Lee H, Maravelias CT. Combining the advantages of discrete- and continuous-time scheduling models: part 1. Framework and mathematical formulations. *Comput Chem Eng*. 2018;116:176-190.

31. Castro PM, Novais AQ. Scheduling multistage batch plants with sequence-dependent changeovers. *AIChE J*. 2009;55:2122-2137.
32. Velez S, Sundaramoorthy A, Maravelias CT. Valid inequalities based on de- mand propagation for chemical production scheduling MIP models. *AIChE J*. 2013;59:872-887.
33. Wu D, Ierapetritou MG. Decomposition approaches for the efficient solution of short-term scheduling problems. *Comput Chem Eng*. 2003; 27:1261-1276.
34. Balas E. Disjunctive programming. *Ann Discrete Math*. 1979;5: 3-51.
35. Castro PM, Grossmann IE. Generalized disjunctive programming as asystematic modeling framework to derive scheduling formulations. *Ind Eng Chem Res*. 2012;51:5781-5792.
36. Grossmann IE, Trespalacios F. Systematic modeling of discretecontinuous optimization models through generalized disjunctive programming. *AIChE J*. 2013;59:3276-3295.
37. Castro PM, Mostafaei H. Batch-centric scheduling formulation for treelike pipeline systems with forbidden product sequences. *Comput Chem Eng*. 2019;122:2-18.
38. Castro PM, Grossmann IE, Veldhuizen P, Esplin D. Optimal maintenance scheduling of a gas engine power plant using generalized disjunctive programming. *AIChE J*. 2014;60:2083-2097.
39. Castro PM, Grossmann IE, Zhang Q. Expanding scope and computational challenges in process scheduling. *Comput Chem Eng*. 2018;114:14-42.
40. Castro PM, Harjunkoski I, Grossmann IE. Discrete and continuous-time formulations for dealing with break periods: Preemptive and non-preemptive scheduling. *Eur J Oper Res*. 2019;278(2):563-577.
41. Gams Documentation, August 2009. https://www.gams.com/latest/docs/gams.pdf.