New General Continuous-Time State−Task Network Formulation for Short-Term Scheduling of Multipurpose Batch Plants

Christos T. Maravelias and Ignacio E. Grossmann*

Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213

A new continuous-time MILP model for the short-term scheduling of multipurpose batch plants is presented. The proposed model relies on the state−task network (STN) approach and addresses the general problem of batch scheduling, accounting for resource (utility) constraints, variable batch sizes and processing times, various storage policies (UIS, FIS, NIS, ZW), batch mixing/splitting, and sequence-dependent changeover times. The key features of the proposed model are the following: (a) a continuous-time representation is used, common for all units; (b) assignment constraints are expressed using binary variables that are defined only for tasks, not for units; (c) start times of tasks are eliminated, so that time-matching constraints are used only for the finish times of tasks; and (d) a new class of valid inequalities that improves the LP relaxation is added to the MILP formulation. Compared to other general continuous time STN formulations, the proposed model is faster. Compared to event-driven formulations, it is more general, as it accounts for resources other than equipment and gives solutions in comparable computational times. The application of the model is illustrated through four example problems.

1. Introduction

The problem of short-term scheduling of multipurpose batch plants has received considerable attention during the past decade. Kondili et al.[1] introduced the state−task network (STN) and proposed a discrete-time MILP model, where the time horizon is divided into time periods of equal duration. Because the resulting models have many binary variables, Shah et al.[2] developed a reformulation and specific techniques to reduce the computational times for discrete-time STN models. Pantelides[3] proposed the alternative representation and formulation of the resource−task network (RTN); Schilling and Pantelides[4] developed a continuous-time MILP model, based on the RTN representation, and a novel branch-and-bound algorithm that branches on both continuous and discrete variables. Zhang and Sargent[5] and Mockus and Reklaitis[6] proposed MINLP continuous-time representations for the scheduling of batch and continuous processes. Ierapetritou and Floudas[7] proposed a new MILP formulation based on event points for the scheduling of batch and continuous multipurpose plants.

Despite the improved formulations, the specialized algorithms, and the recent improvements in computer hardware and optimization software, the short-term scheduling of STN multipurpose batch plants in continuous time remains a difficult problem to solve. In an effort to reduce problem sizes and computational times, several authors have proposed various approaches during the past two years.[8−12] In most of these approaches, however, the authors make specific assumptions that, on one hand, lead to more compact formulations but, on the other hand, address only specific cases of the general short-term scheduling problem. Some of the most common assumptions that result in significant reductions in problem size are (a) no batch splitting and (b) no resource constraints other than those on equipment units. As will be illustrated later, even weaker assumptions give rise to formulations that cannot account for all conditions that might arise in a multipurpose batch plant.

In this work, we propose a new general state−task network MILP model for the short-term scheduling of multipurpose batch plants in continuous time that accounts for resource constraints other than equipment (utilities); variable batch sizes and processing times; various storage policies (UIS, FIS, NIS, ZW), including shared storage; and sequence-dependent changeover times and that allows for batch mixing/splitting. The paper is structured as follows. In section 2, we briefly discuss the different time representation approaches proposed in the literature, and we outline the proposed approach. The problem statement is presented in section 3. The mathematical formulation and its derivation are described in section 4, and some remarks are presented in section 5. Four examples are presented in the last section.

2. Outline of Proposed Approach

Several time representation schemes have been proposed for the scheduling of multipurpose batch plants. Kondili et al.[1] introduced the STN formulation using a discrete-time representation (Figure 1a), where the time horizon is divided into H intervals of equal duration, common for all units, and where the tasks must begin and finish exactly at a time point. This means that the discrete-time representation can be used only when the processing times are constant and, furthermore, that the duration of the intervals must be equal to the greatest common factor of the processing times. The assumption of constant processing times is not always realistic, and the length of the intervals might be so small that it either leads to a prohibitive number of intervals, rendering the resulting model unsolvable, or

* To whom all correspondence should be addressed. Tel.: 412-268-3642. Fax: 412-268-7139. E-mail: grossmann@cmu.edu.

10.1021/ie020923y CCC: $25.00
Published on Web 05/17/2003

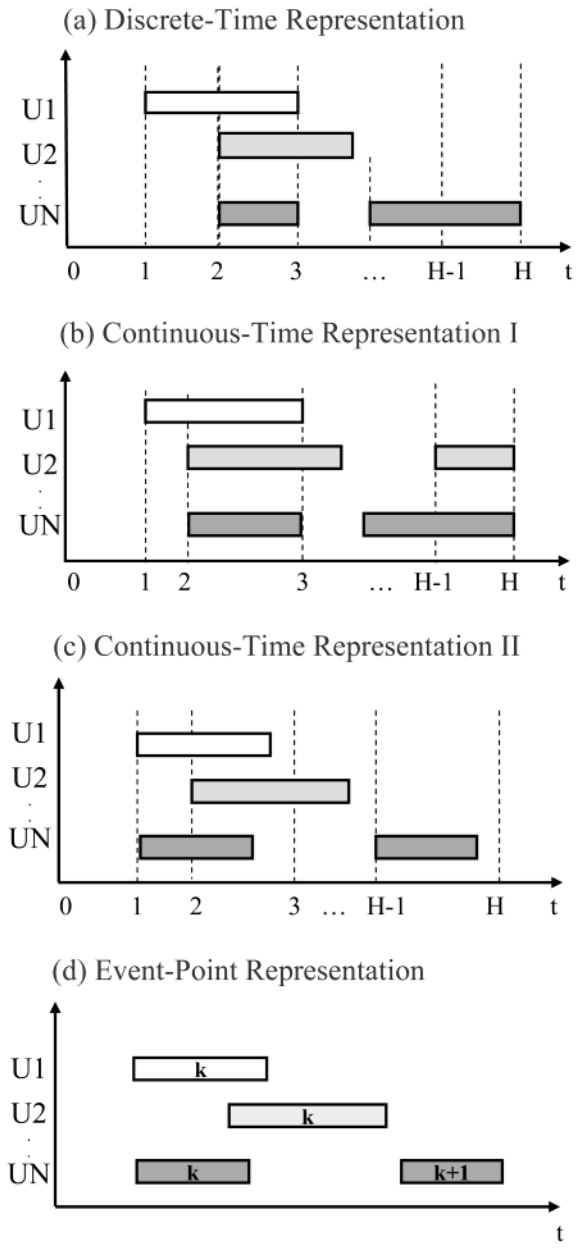


Figure 1. Alternative time-domain representations.

else requires approximations that might compromise the feasibility and optimality of the solution.

To circumvent the above-cited difficulties, two different continuous-time representations were proposed in which the time horizon is divided into time intervals of unequal and unknown duration, common for all units. In continuous-time representation I (Figure 1b), each task must start and finish exactly at a time point,[4−6] whereas in representation II (Figure 1c), each task must start at a time point but it might not finish at a time point.[8] In both representations, the number of time points is determined with an iterative procedure, during which the number of time points is increased by 1 until there is no improvement in the objective function. Because the time points are not fixed, constraints that match a time point with the start (or finish) of a task are necessary. These constraints are big-M constraints that result in poor LP relaxations. On the other hand, the continuous-time representation accounts for variable processing times and is more realistic than the discrete

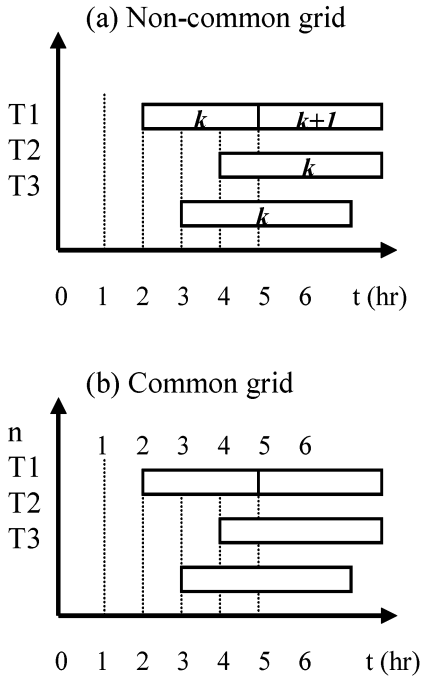


Figure 2. Common vs noncommon time points.

representation. It also requires significantly fewer time intervals and, hence, leads to smaller problems.

An alternative approach is the event-point representation (Figure 1d) proposed by Ierapetritou and Floudas,[7] in which the time intervals are not common for all units. In this approach, the time horizon is divided into a number of events that is different for each unit, subject to some sequencing constraints. In the schedule depicted in Figure 2, the continuous-time approach (II) requires four common intervals (Figure 2b), whereas the event-point approach (Figure 2a) requires two events (k and $k + 1$), as the maximum number of tasks assigned to any unit is two. This leads to smaller formulations that are up to 2 orders of magnitude faster than other continuous-time formulations, as illustrated by Ierapetritou and Floudas.[7] As shown in Appendix A, however, event-driven approaches are not as general as the continuous-time approaches and might, in fact, eliminate feasible solutions. Furthermore, they cannot be readily used when utilities need to be taken into account.

Other assumptions that have led to formulations that are easier to solve (e.g., Rodrigues et al.[9]) are (i) no batch splitting and mixing and (ii) no resource constraints other than those on equipment. Assumption i is usually coupled with the assumption of constant batch sizes and processing times, which, in turn, implies that the amounts of raw materials and intermediates needed for the production of one batch of final product can be calculated given the assignment of units to tasks. This means that mass balance equations need not be included in the formulation. When both assumptions i and ii are considered, the level of states and resources need not be monitored; thus, a common time coordinate (time discretization for all units) is not necessary. This, in turn, removes the (big-M) time-matching constraints of continuous-time formulations between grid time points and task time points (start and finish times). In addition, assignment binaries are indexed by tasks and unit time slots (instead of tasks and time periods), and because the total number of time periods is larger than

the number of slots for each unit, the number of binary variables is reduced.

In the proposed MILP model, we use continuous-time representation I for the tasks that produce at least one state for which a zero-wait policy is applied and continuous-time representation II for all other tasks. As will be shown, this mixed representation does not compromise feasibility or optimality. To reduce the number of binary variables, we use the idea of task decoupling, and we eliminate the binaries for unit assignment. The first idea was proposed by Ierapetritou and Floudas,[7] while the second is achieved by expressing the assignment constraints using only task binaries. Furthermore, we eliminate start times, thus reducing the number of big-M time-matching constraints. Finally, we develop a new class of valid inequalities that significantly tighten the LP relaxation. The result of these actions is to develop a general continuous-time MILP model that is computationally effective.

3. Problem Statement

We assume that we are given the following items: (i) a fixed or variable time horizon; (ii) the available units and storage tanks, along with their capacities; (iii) the available utilities and their upper limits; (iv) the production recipe (mass balance coefficients and utility requirements); (v) the processing time data; (vi) the amounts of available raw materials; and (vii) the prices of raw materials and final products.

The goal is then to determine: (i) the sequence and timing of tasks taking place in each unit, (ii) the batch sizes of tasks, (iii) the allocated resources, and (iv) the amounts of raw materials purchased and final products sold.

The proposed model can accommodate various objectives, such as the maximization of income or profit (if tasks incur a cost) or the minimization of the makespan for a specified demand. For simplicity, we first assume no changeover times, but later, we discuss how changeovers can be addressed.

4. Mathematical Formulation

To clarify the derivation of the proposed MILP model, we first formulate it as a hybrid generalized disjunctive/mixed-integer programming (GDP/MILP) model (Raman and Grossmann,[13] Vecchietti and Grossmann[14]) that involves 0−1 and Boolean variables and mixed-integer constraints as well as disjunctions and implications.

4.1. Hybrid GDP/MILP Model. To decouple units from tasks, we use the following rule: If a task i can be performed in both units j and j', then two tasks i (performed in unit j) and i' (performed in unit j') are defined; note that unit-dependent processing times are also handled by decoupling. The time horizon is divided into intervals that are common for all units and utilities. The two main assumptions for utilities are that (a) the plant has an upper bound on the availability of utilities that cannot be exceeded at any time and (b) a task consumes the same amount of utility(ies) throughout its execution. Time point n occurs at time T_n, and N is the set of time points. The first time point corresponds to the start ($T_1 = 0$) and the last to the end ($T_{|N|} = H$) of the scheduling horizon. The ordering of time points is enforced trough constraint 3.

$$T_{n=1} = 0 \tag{1}$$

$$T_{n=|N|} = H \tag{2}$$

$$T_{n+1} \geq T_n \qquad \forall\, n \tag{3}$$

For each task i and time point n, three binary variables (Ws_{in}, Wp_{in}, and Wf_{in}) are defined as follows:

$Ws_{in} = 1$ if task i starts at time point n.

$Wp_{in} = 1$ if task i is being processed at time point n (i.e., starts before and finishes after time point n).

$Wf_{in} = 1$ if task i finishes at or before time point n.

To derive the assignment constraints, the following three auxiliary binary variables are defined [$I(j)$ is the set of tasks that can be performed in equipment unit j]:

$Zs_{jn} = 1$ if a task in $I(j)$ is assigned to start in unit j at time point n.

$Zp_{jn} = 1$ if a task in $I(j)$ is being processed in unit j at time point n (i.e., starts before and finishes after n).

$Zf_{jn} = 1$ if a task in $I(j)$ assigned to unit j, finishes at or before time point n.

Equivalently, Zs_{jn} is equal to 1 if and only if one of the tasks that can be assigned to unit j is assigned to start in unit j at time point n. Note that, for any given time, at most one of these tasks can be assigned to unit j. This condition is expressed by the following logic expression in which the binary variables are treated as Boolean variables:

$$Zs_{jn} \leftrightarrow \bigvee_{i \in I(j)} Ws_{in} \;\forall j, \forall n \tag{A}$$

Similarly, Zf_{jn} is equal to 1 if and only if one task that can be assigned to unit j finishes processing in j at or before time point n, which is logically expressed as

$$Zf_{jn} \leftrightarrow \bigvee_{i \in I(j)} Wf_{in} \qquad \forall\, j, \forall\, n \tag{B}$$

Additionally, at most one task can start (finish) at unit j at any time point n

$$\sum_{i \in I(j)} Ws_{in} \leq 1 \qquad \forall\, j, \forall\, n \tag{4}$$

$$\sum_{i \in I(j)} Wf_{in} \leq 1 \qquad \forall\, j, \forall\, n \tag{5}$$

Also, to enforce the condition that all tasks that start must finish, we have

$$\sum_{n} Ws_{in} = \sum_{n} Wf_{in} \qquad \forall\, i \tag{6}$$

The integer expression for binary Zp_{jn} is given by eq C (see derivation in Appendix B)

$$Zp_{jn} = \sum_{n' < n} Zs_{jn'} - \sum_{n' \leq n} Zf_{jn'} \qquad \forall\, j, \forall\, n \tag{C}$$

The following necessary logic condition is the core for the proposed assignment constraint: A task can be assigned to start in unit j at time point n only if there is no other task being processed in equipment j at time point n. This condition can be expressed in logic form as

$$Zs_{jn} \Rightarrow \neg Zp_{jn} \tag{D}$$

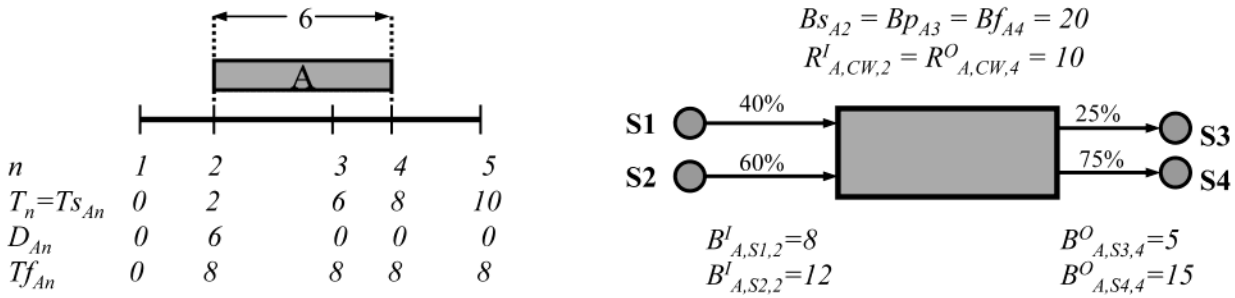


Figure 3. Example of variables T_n, Ts_{in}, D_{in}, Tf_{in}, R^{I}_{irn}, R^{O}_{irn}, B^{I}_{isn}, and B^{O}_{isn}.

For the batch size of each task i, we define three continuous variables that correspond to the batch size of task i when it starts at time point n (Bs_{in}), when it is being processed at time point n (Bp_{in}), and when it finishes at or before time point n (Bf_{in}). At any time point n, at most one of these variables is nonzero. For each task i, we also define the start time, Ts_{in}; the duration, D_{in}; and the finish time, Tf_{in}. The amount of state s consumed/produced by task i at time point n is denoted by $B^{\mathrm{I}}_{isn}/B^{\mathrm{O}}_{isn}$ and the amount of utility r needed for task i that starts/finishes at time point n is denoted by $R^{\mathrm{I}}_{irn}/R^{\mathrm{O}}_{irn}$. Finally, S_{sn}, SS_{sn}, and R_{rn} are the level of state s, the amount of state s sold, and the level of utility r in use, at time point n, respectively. In the proposed formulation, the start time of a task i, Ts_{in}, is always equal to T_n, and hence, it can be eliminated.

An example of how the variables referring to a task A vary is given in Figure 3, where a horizon of 10 h is divided into four intervals. Reactants S1 and S2 are mixed in a 2/3 proportion to produce products S3 and S4 in a 1/3 proportion. The batch size of task A is 20 kg, and 10 kg/min of cooling water is required throughout the processing of A. Task A starts at $t = 2$ and finishes at $t = 8$, i.e., starts at $n = 2$, is being processed at $n = 3$, and finishes at $n = 4$. As shown in Figure 3, the duration of a task is nonzero only at the time point that the task starts ($n = 2$: $D_{\mathrm{A2}} = 6$), and the finish time of a task changes only when a task starts ($n = 2$) and remains unchanged during the remaining time points ($n > 2$). Variables Bs_{in}, Bp_{in}, and Bf_{in} are nonzero only at the time points that the task starts ($n = 2$, $\mathrm{Bs}_{\mathrm{A2}} = 20$), is being processed ($n = 3$, $\mathrm{Bp}_{\mathrm{A3}} = 20$), and finishes ($n = 4$, $\mathrm{Bf}_{\mathrm{A4}} = 20$), respectively. Similarly, the amount of utility reserved for task A (made available after task A is completed) is nonzero at the beginning (end) of the task at $n = 2$ ($n = 4$). In the example of Figure 3, 10 kg/min of cooling water is reserved ("engaged") for task A at $n = 2$ ($R^{\mathrm{I}}_{\mathrm{A,CW,2}} = 10$) and becomes available again ("released") at $n = 4$ ($R^{\mathrm{O}}_{\mathrm{A,CW,4}} = 10$). Finally, the amount of a state consumed/produced by a task is nonzero when the task starts/finishes.

Regarding the time representation, we use continuous-time representation I for tasks that produce at least one ZW state; i.e., we require that the produced states of such a task are immediately transferred to another unit or to a storage tank. For all other tasks, we use continuous-time representation II; i.e., we allow such tasks to finish within a period. For states with unlimited storage, this is obviously not a restriction because the storage capacity and the timing of the transfer is not an issue. However, this is also possible for FIS and NIS states because we assume that non-ZW states can be temporarily stored in an equipment unit. If task A assigned to unit U finishes at $t = \mathrm{Tf}_{\mathrm{A}}$ (with $T_{n-1} < \mathrm{Tf}_{\mathrm{A}} < T_n$), for example, and the storage tanks for the output states are full, unit U can be used for storage from $t = \mathrm{Tf}_{\mathrm{A}}$ until $t = T_n$. When the states are actually transferred, at $t = T_n$, mass balance and storage capacity constraints are enforced, and hence, feasibility is guaranteed. If a solution where one unit is utilized as storage is suboptimal, a better solution can be obtained when additional time points are postulated, allowing for all tasks performed in this unit to finish exactly at a time point. The proposed hybrid time representation, therefore, allows us to find the same solutions as continuous-time representation I using fewer time points whenever this is possible. Because the performance of continuous-time models depends heavily on the number of time points, the proposed mixed time representation leads to more efficient models.

For each state s and time point n, the mass balance and the storage constraints are

$$S_{sn} + \mathrm{SS}_{sn} = S_{sn-1} + \sum_{i \in O(s)} B^{\mathrm{O}}_{isn} - \sum_{i \in I(s)} B^{\mathrm{I}}_{isn} \qquad \forall\, s, \forall\, n > 1 \quad (7)$$

$$S_{sn} \le C_s \qquad \forall\, s, \forall\, n \quad (8)$$

where C_s is the capacity of the storage tank of state s. The continuous variable SS_{sn} offers the possibility of removing final products before the end of the time horizon. This might be necessary if, for example, there is limited storage capacity for the final products. If this is not true or if sales can occur only at the end of the horizon, as is usually the case, variables SS_{sn} can be fixed to zero for $n < |N|$. Constraint 7 can be modified to account for raw material purchases.

The total amount of utility r used at time point n is given by eq 9 and bounded not to exceed the maximum availability, R^{MAX}_r, by eq 10

$$R_{rn} = R_{rn-1} - \sum_i R^{\mathrm{O}}_{irn-1} + \sum_i R^{\mathrm{I}}_{irn} \qquad \forall\, r, \forall\, n \quad (9)$$

$$R_{rn} \le R^{\mathrm{MAX}}_r \qquad \forall\, r, \forall\, n \quad (10)$$

Because the batch size variables, Bs_{in}, Bp_{in}, and Bf_{in}; the start time Ts_{in}; the duration D_{in}; the finish time Tf_{in}; the utility requirements $R^{\mathrm{I}}_{irn}/R^{\mathrm{O}}_{irn}$; and the consumption/production $B^{\mathrm{I}}_{isn}/B^{\mathrm{O}}_{isn}$ of state s are defined for a task i and a time point n, we will first present the constraints

for the calculation of those variables in disjunctive form, where each disjunction is expressed for all (i, n) pairs. For each (i, n) pair there are three different disjunctive cases: (a) task i starts (Ws_{in}) at n, (b) task i is being processed (Wp_{in}) at n, and (c) task i finishes (Wf_{in}) at or before time point n.

The disjunction in terms of the start of task i at time point n is given by disjunction E, which is expressed for all $n < |N|$, since a task cannot start at the last time point

$$\begin{bmatrix} Ws_{in} \\ D_{in} = \alpha_i + \beta_i Bs_{in} \\ Tf_{in} = Ts_{in} + D_{in} \\ B_i^{MIN} \le Bs_{in} \le B_i^{MAX} \\ Bs_{in} = Bp_{in+1} + Bf_{in+1} \\ B_{isn}^{I} = \rho_{is} Bs_{in} \\ R_{irn}^{I} = \gamma_{ir} + \delta_{ir} Bs_{in} \\ Bp_{in} = 0 \end{bmatrix} \vee \begin{bmatrix} \neg Ws_{in} \\ D_{in} = 0 \\ Tf_{in} = Tf_{in-1} \\ Bs_{in} = B_{isn}^{I} = R_{irn}^{I} = 0 \end{bmatrix}$$

$$\forall\, i, \forall\, n < |N| \quad \text{(E)}$$

As explained in Figure 3, if task i starts at time point n (Ws_{in} is true), its duration D_{in}, which is a linear function of Bs_{in}, is nonzero, whereas its finish time Tf_{in}, changes by D_{in}. The batch size, Bs_{in}, lies within lower and upper bounds, and it is (a) equal to Bp_{in+1} if i is being processed at $n + 1$ and (b) equal to Bf_{in+1} if i finishes at or before $n + 1$. The amount of utility r reserved for task i, R_{irn}^{I}, and the amount of state s consumed by task i, B_{isn}^{I}, are also nonzero. If Ws_{in} is false ($\neg Ws_{in}$), the condition that the finish time, Tf_{in}, remains unchanged is enforced. This condition is not necessary, but our computational experience shows that it reduces the size of the branch and bound tree.

The disjunction in terms of the processing of task i at time point n is given by eq F. Note that, if a task is processed at time point n, it needs to start before n and finish after n, which, in turn, implies that it cannot be processed in the first and the last time periods.

$$\begin{bmatrix} Wp_{in} \\ Tf_{in} = Tf_{in-1} \\ B_i^{MIN} \le Bp_{in} \le B_i^{MAX} \\ Bp_{in} = Bs_{in-1} + Bp_{in-1} \\ Bp_{in} = Bp_{in+1} + Bf_{in+1} \\ Bs_{in} = Bf_{in} = B_{isn}^{I} = B_{isn}^{O} = 0 \\ D_{in} = R_{irn}^{I} = R_{irn}^{O} = 0 \end{bmatrix} \vee \begin{bmatrix} \neg Wp_{in} \\ Tf_{in} \ge Tf_{in-1} \\ Bp_{in} = 0 \end{bmatrix}$$

$$\forall\, i, 1 < n < |N| \quad \text{(F)}$$

When task i is being processed at time point n (i.e., Wp_{in} is true), the finish time, Tf_{in}, remains unchanged, and the batch size, Bp_{in}, lies within lower and upper bounds and equals the batch size of the previous and the next time points. If task i is not processed at n ($\neg Wp_{in}$), the finish time, Tf_{in}, remains unchanged if i does not start at n ($\neg Ws_{in}$) and increases if task i starts at n (Ws_{in}).

The disjunction in terms of the finishing of task i at or before time point n is given by eq G, which is expressed for all $n > 1$, as a task cannot finish at the start of the scheduling horizon

$$\begin{bmatrix} Wf_{in} \\ Tf_{in} \le T_n,\ i \notin ZW \\ Tf_{in} = T_n,\ i \in ZW \\ B_i^{MIN} \le Bf_{in} \le B_i^{MAX} \\ Bf_{in} = Bs_{in-1} + Bp_{in-1} \\ B_{isn}^{O} = \rho_{is} Bf_{in} \\ R_{irn}^{O} = \gamma_{ir} + \delta_{ir} Bf_{in} \\ Bp_{in} = 0 \end{bmatrix} \vee \begin{bmatrix} \neg Wf_{in} \\ Tf_{in} \ge Tf_{in-1} \end{bmatrix}$$

$$\forall i, \forall n < |N| \quad \text{(G)}$$

Here, the batch size Bf_{in} lies within the lower and upper bounds if task i finishes at or before time point n (Wf_{in}), and it is equal to the batch size of the previous time period. The amount of state s produced and the level of utility r becoming available are also nonzero.

Finally, the proposed model, as most continuous-time models for the short-term scheduling of batch plants, appears to be effective when the objective function is the maximization of income or a profit-/income-related function

$$\max Z = \sum_s \sum_n \zeta_s SS_{sn} \tag{11}$$

The minimization of makespan (for fixed demand) can also be accommodated. The problem given by the mixed-integer constraints 1−11 and C; the logic constraints A, B, and D; and the disjunctions in E−G corresponds to a hybrid GDP/MILP model, which will be transformed into an MILP model, as shown in the next section.

4.2. MILP Model. Using the definitions of the binaries Ws_{in}, Wp_{in}, Wf_{in}, Zs_{in}, Zp_{in}, and Zf_{in}, we can transform the logic conditions A, B, and D and the disjunctions in E−G into mixed-integer constraints. As explained in Appendix B, the binaries Zs_{jn}, Zp_{jn}, Zf_{jn}, and Wp_{in} can be expressed through binaries Ws_{in} and Wf_{in}, which are the only ones used in the mixed-integer constraints.

Assignment Constraints. The basic assignment constraint 12 is derived in Appendix B and follows from the constraints in A−D. Constraints 13 and 14 enforce the conditions that no task can finish at $t = 0$ or start at the end of the horizon. Constraints 4−6 are repeated for completeness.

$$\sum_{i \in I(j)} Ws_{in} \le 1 \qquad \forall\, j, \forall\, n \tag{4}$$

$$\sum_{i \in I(j)} Wf_{in} \le 1 \qquad \forall\, j, \forall\, n \tag{5}$$

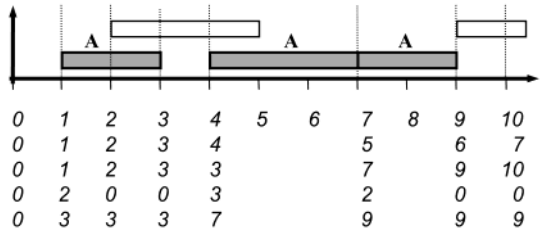
$$\sum_n Ws_{in} = \sum_n Wf_{in} \qquad \forall\, i \tag{6}$$

$$\sum_{i \in I(j)} \sum_{n' \le n} (Ws_{in'} - Wf_{in'}) \le 1 \qquad \forall\, j, \forall\, n \tag{12}$$

$$Wf_{i0} = 0 \qquad \forall\, i \tag{13}$$

$$Ws_{in} = 0 \qquad \forall\, i, n = |N| \tag{14}$$

Duration, Finish Time, and Time-Matching Constraints. From the disjunction in E, the duration of a task is calculated by eq 15 using convex hull reformulation (Balas,[15] Raman and Grossmann[16]) and eliminating variables, while the finish time is expressed with big-M constraints 16 and 17, which are active only if



t	0	1	2	3	4	5	6	7	8	9	10
n	0	1	2	3	4			5		6	7
$T_n=Ts_{An}$	0	1	2	3	3			7		9	10
D_{An}	0	2	0	0	3			2		0	0
Tf_{An}	0	3	3	3	7			9		9	9

Figure 4. Relationships among variables T_n, Ts_{in}, Tf_{in}, and D_{in}.

task i starts at time point n ($Ws_{in} = 1$)

$$D_{in} = \alpha_i Ws_{in} + \beta_i Bs_{in} \qquad \forall\, i, \forall\, n \qquad (15)$$

$$Tf_{in} \le Ts_{in} + D_{in} + H(1 - Ws_{in}) \qquad \forall\, i, \forall\, n \qquad (16)$$

$$Tf_{in} \ge Ts_{in} + D_{in} - H(1 - Ws_{in}) \qquad \forall\, i, \forall\, n \qquad (17)$$

As explained in Figure 3 and enforced in the disjunction in E, the finish time, Tf_{in}, of a task i remains unchanged until the next occurrence of task i. This condition is enforced by the big-M constraint in 18. Constraint 19 is used to enforce the conditions that (a) Tf_{in} is always greater or equal to Tf_{in-1} and (b) the "step" in the finish time, Tf_{in}, when a task occurs at time n must be at least as large as the duration of task i. The latter condition is not necessary, and it is not derived from disjunction E, but its addition leads to smaller branch-and-bound trees and shorter computational times

$$Tf_{in} - Tf_{in-1} \le H \cdot Ws_{in} \qquad \forall\, i, \forall\, n \qquad (18)$$

$$Tf_{in} - Tf_{in-1} \ge D_{in} \qquad \forall\, i, \forall\, n \qquad (19)$$

Because the start time, Ts_{in}, is equal to T_n, variables Ts_{in} are eliminated (constraint 20). Constraints 21 and 22 are the time-matching constraints for the finish time, Tf_{in}, of a task and result from disjunction G using the big-M formulation. Note that, in the general case, a task i might finish at or before time point n (constraint 21), but when it produces a material for which a zero-wait (ZW) storage policy applies, the finish time of task i should coincide with time point n (effect of constraints 21 and 22)

$$Ts_{in} = T_n \qquad \forall\, i, \forall\, n \qquad (20)$$

$$Tf_{in-1} \le T_n + H(1 - Wf_{in}) \qquad \forall\, i, \forall\, n \qquad (21)$$

$$Tf_{in-1} \ge T_n - H(1 - Wf_{in}) \qquad \forall\, i \in ZWI, \forall\, n \qquad (22)$$

An example of how variables T_n, Ts_{in}, and Tf_{in} vary for a task A is given in Figure 4. Note that task A is assumed to produce a state for which a ZW storage policy applies, so the finish time must coincide with a time point. Although integer processing times have been used, the same principles apply for real constant or variable processing times. As shown, start and finish times are defined for seven time points, and the former are always equal to T_n, so they are eliminated. Because of constraint 18, Tf_{A2} and Tf_{A3} are equal to 3 (i.e., equal to Tf_{A1}), and Tf_{A6} and Tf_{A7} are equal to 9 (i.e., equal to Tf_{A5}), while constraint 19 is trivially satisfied: $Tf_{A4} - Tf_{A3} = 7-3 = 4 \ge 3 = D_{A4}$ and $Tf_{A5} - Tf_{A4} = 9-7 = 2 \ge 2 = D_{A5}$.

Batch Size Constraints. Constraints 23 and 24 impose minimum and maximum bounds on the batch size of a task and result from the convex hull reformulation of disjunctions E−G. Note that the terms multiplied by B_i^{MIN} and B_i^{MAX} in constraint 25 are equal to the auxiliary binary Wp_{in} (Appendix B). Constraint 26 enforces the requirement that variables Bs_{in}, Bp_{in}, and Bf_{in} are equal for a given task, and it is also derived from disjunctions E−G using the convex hull reformulation and eliminating the disaggregated variables

$$B_i^{MIN} Ws_{in} \le Bs_{in} \le B_i^{MAX} Ws_{in} \qquad \forall\, i, \forall\, n \qquad (23)$$

$$B_i^{MIN} Wf_{in} \le Bf_{in} \le B_i^{MAX} Wf_{in} \qquad \forall\, i, \forall\, n \qquad (24)$$

$$B_i^{MIN}\Big(\sum_{n'<n} Ws_{in'} - \sum_{n'\le n} Wf_{in'}\Big) \le Bp_{in} \le B_i^{MAX}\Big(\sum_{n'<n} Ws_{in'} - \sum_{n'\le n} Wf_{in'}\Big) \qquad \forall\, i, \forall\, n \qquad (25)$$

$$Bs_{in-1} + Bp_{in-1} = Bp_{in} + Bf_{in} \qquad \forall\, i, \forall\, n \qquad (26)$$

The amount of state s consumed/produced by task i at time point n is calculated through eq 27/29 and bounded by eq 28/30, where SI(i) and SO(i) are the set of input and output states of task i, respectively. Constraints 28 and 30 are not necessary but result in shorter computational times. The convex hull reformulation was used for the derivation of constraints 27−30.

$$B_{isn}^{I} = \rho_{is} Bs_{in} \qquad \forall\, i, \forall\, n, \forall\, s \in SI(i) \qquad (27)$$

$$B_{isn}^{I} \le B_i^{MAX} \rho_{is} Ws_{in} \qquad \forall\, i, \forall\, n, \forall\, s \in SI(i) \qquad (28)$$

$$B_{isn}^{O} = \rho_{is} Bf_{in} \qquad \forall\, i, \forall\, n, \forall\, s \in SO(i) \qquad (29)$$

$$B_{isn}^{O} \le B_i^{MAX} \rho_{is} Wf_{in} \qquad \forall\, i, \forall\, n, \forall\, s \in SO(i) \qquad (30)$$

Mass Balance/Storage Constraints. As shown above, constraints 7 and 8 express the mass balance and capacity constraints for state s at time point n

$$S_{sn} + SS_{sn} = S_{sn-1} + \sum_{i \in O(s)} B_{isn}^{O} - \sum_{i \in I(s)} B_{isn}^{I} \qquad \forall\, s, \forall\, n > 1 \qquad (7)$$

$$S_{sn} \le C_s \qquad \forall\, s, \forall\, n \qquad (8)$$

Utility Constraints. The amount of utility r consumed by task i that starts at time point n is calculated through eq 31; the amount of utility r released at the end of task i is calculated through eq 32. Constraints 31 and 32 are derived from disjunctions E and G, respectively, using the convex hull reformulation. The total amount of utility r consumed by various tasks during period n is calculated by eq 9 and bounded not to exceed R_r^{MAX} by constraint 10

$$R_{irn}^{I} = \gamma_{ir} Ws_{in} + \delta_{irs} Bs_{in} \qquad \forall\, i, \forall\, r, \forall\, n \qquad (31)$$

$$R_{irn}^{O} = \gamma_{ir} Wf_{in} + \delta_{irs} Bf_{in} \qquad \forall\, i, \forall\, r, \forall\, n \qquad (32)$$

$$R_{rn} = R_{rn-1} - \sum_i R_{irn-1}^{O} + \sum_i R_{irn}^{I} \qquad \forall\, r, \forall\, n \qquad (9)$$

$$R_{rn} \le R_r^{MAX} \qquad \forall\, r, \forall\, n \qquad (10)$$

Objective Function. As in the hybrid GDP/MILP model, the objective is given by

$$\max Z = \sum_s \sum_n \zeta_s SS_{sn} \tag{11}$$

$$Ws_{in}, Wf_{in} \in \{0, 1\}; Bs_{in}, Bp_{in}, Bf_{in}, SS_{sn}, S_{sn}, T_n, Tf_{in}, D_{in}, B^{I}_{isn}, B^{O}_{isn}, R^{I}_{irn}, R^{O}_{irn}, R_{rn} \geq 0 \tag{33}$$

Constraints 1−33 comprise the MILP model M for the short-term scheduling of multipurpose batch plants. This model can be tightened by adding the following valid inequalities that lead to model M*.

Tightening Constraints. Constraint 12 is sufficient to enforce feasibility, but it leads to weak relaxations. In the relaxed solution of this formulation, specifically, the summation of processing times of tasks that are scheduled to a unit j is larger than the time horizon (see Figure 5a, where the time horizon is divided into three time periods). Constraint 34 tightens the formulation by enforcing the condition that the summation of the durations of the tasks assigned to a specific equipment unit should be smaller than or equal to the time horizon (Figure 5b)

$$\sum_{i \in I(j)} \sum_n D_{in} \leq H \qquad \forall\, j \tag{34}$$

However, although this condition is enforced for the entire time horizon, it is not enforced for each time period; i.e., the summation of processing times of tasks assigned to start on equipment unit j at time point n and finish at time point $n + 1$ is larger than $T_{n+1} - T_n$ (e.g., the first period of Figure 5b). The result of the addition of constraints 35 and 36 is that the same condition is satisfied for each time period (Figure 5c). Constraint 35 restricts the sum of the processing times of all tasks staring on unit j after T_n to be less than the amount of time left (i.e., $H - T_n$), and constraint 36 restricts the sum of processing times of tasks finishing in unit j before T_n to be less than T_n.

$$\sum_{i \in I(j)} \sum_{n' \geq n} D_{in'} \leq H - T_n \qquad \forall\, j, \forall\, n \tag{35}$$

$$\sum_{i \in I(j)} \sum_{n' \leq n} (\alpha_i Wf_{in'} + \beta_i Bf_{in'}) \leq T_n \qquad \forall\, j, \forall\, n \tag{36}$$

The addition of constraints 34−36 leads to relaxed solutions with smaller durations, and this, in turn, leads to smaller values of the Ws_{in} variables, since D_{in} is related to Ws_{in} via eq 15 (Figure 6). Moreover, because of eq 6, the binaries Wf_{in} have also smaller values. Finally, because the batch sizes of tasks (Bs_{in}, Bp_{in}, and Bf_{in}) are bounded by Ws_{in} and Wf_{in}, and because the objective function depends on the amount of final products produced (i.e., on the batch sizes), the addition of these constraints results in tighter relaxations.

The proposed model M* consists of constraints 1−36, and as will be shown in section 6.1.2, it is significantly faster than model M.

5. Remarks

5.1. Sequence-Dependent Changeover Times. Sequence-dependent changeovers can be easily incorporated within the proposed framework. To allow temporary storage of produced materials in the equipment unit, in this case, we express constraint 17 only for tasks

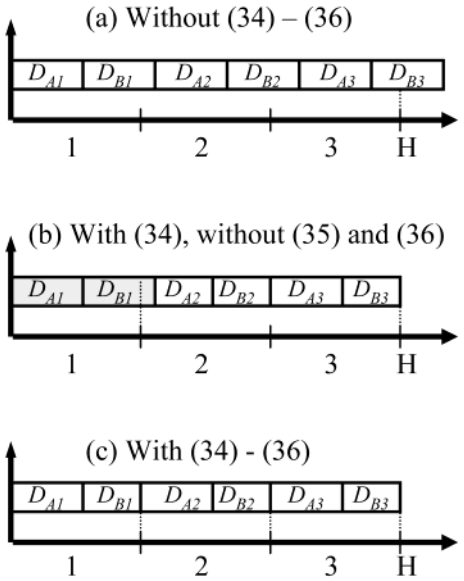


Figure 5. Tightening of LP relaxation through constraints 34−36.

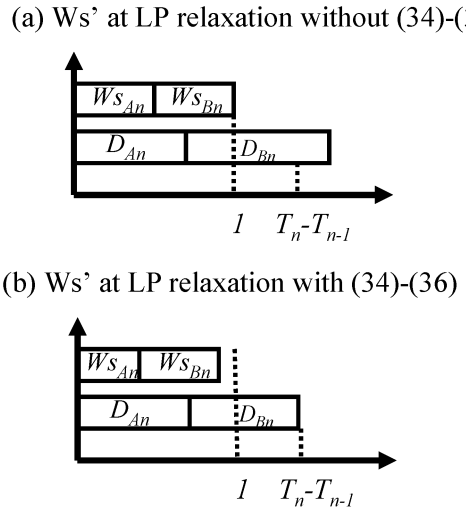


Figure 6. Effect of tightening constraints

in ZWI, but we enforce eq 22 for all tasks. This implies that the finish time, Tf_{in}, of task i can be greater than the sum $Ts_{in} + D_{in}$ (i.e., we allow storage of material from $t = Ts_{in} + D_{in}$ until $t = Tf_{in}$), but it must coincide with a time point (i.e., produced states have to be transferred to storage tanks or other units at $t = Tf_{in}$), and thus, the equipment becomes available for cleaning/setup after Tf_{in}. Assuming that changeovers are shorter than task processing times, it is sufficient to add constraint 37, where $s_{ii'}$ is the sequence-dependent setup time when task i is followed by task i' in unit j

$$Ts_{i'n} \geq Tf_{in-1} + s_{ii'} \qquad \forall\, j, \forall\, i \in I(j), \forall\, i' \in I(j) | s_{ii'} > 0, \forall\, n > 1 \tag{37}$$

As explained in section 4.2, the finish time Tf_{in} of task i remains unchanged, as a result of constraint 18, until the next batch of task i occurs, and thus, there are three cases for the last batch of task i performed in unit j before task i' starts at $T_n = Ts_{i'n}$:

(a) Task i finishes exactly at $T_n = Ts_{i'n}$. Task i is obviously the last task processed in unit j before task i', and in this case, $Tf_{in-1} = T_n = Ts_{i'n}$; clearly, this can happen only if no changeover time is required ($s_{ii'} = 0$).

If $s_{ii'} > 0$, constraint 37 is violated, and hence, this solution is excluded.

(b) Task i finishes at T_{n-k}, where $k \geq 1$, and it is the last task processed in unit j before task i' starts at $T_n = \mathrm{Ts}_{i'n}$. Because task i is not carried out again until time point n, its finish time remains unchanged, i.e., $\mathrm{Tf}_{n-k} = \mathrm{Tf}_{n-1}$, and thus, constraint 37 is a valid constraint that correctly enforces the changeover.

(c) Task i finishes at T_{n-k} ($k \geq 1$), but it is not the last task processed in unit j before task i'. This implies that there is another task $i'' \neq i$ performed after i. Hence, there is task i'' processed between the end of task i (at $t = \mathrm{Tf}_{n-k} = \mathrm{Tf}_{n-1}$) and the start of task i' at $t = \mathrm{Ts}_{i'n}$, and constraint 37 is expressed for the (i'', i') pair as well. Because we have assumed that the setup times are shorter than the processing times, constraint 37 for the (i'',i') pair is tighter than constraint 37 for the (i',i) pair, and the latter is redundant.

Note that no additional variables are needed to model sequence-dependent changeover times. The requirement to apply continuous-time representation I for all tasks, however, results in models that are difficult to solve because of the increased number of time points.

It should also be mentioned that, if, in addition to changeover times, one would like to include changeover costs, this would require the introduction of the new variable $Y_{ii'n}$ that is equal to 1 if task i, finishing at or before time point n, is followed by task i', starting at time point n, where $i \in I(j)$ and $i' \in I(j)$ for some unit j. Variable $Y_{ii'n}$ is defined only for pairs of tasks that take place in the same unit and have a nonzero changeover cost; it can be treated as a continuous variable because it will always be equal to 0 or 1 in an integer solution and is activated through the following constraint (assuming that changeover times are shorter than processing times)

$$Y_{ii'n} \geq \mathrm{Wf}_{in-1} + \mathrm{Ws}_{i'n} - 1 \quad \forall\, j, \forall\, i \in I(j), \forall\, i' \in I(j) | \kappa_{ii'} > 0,\ 1 < n < |N| \qquad (38)$$

The objective function is modified as follows

$$\max Z = \sum_s \sum_n \zeta_s \mathrm{SS}_{sn} - \sum_{1<n<|N|} \sum_j \sum_{i \in I(j)} \sum_{i' \in I(j)} \kappa_{ii'} Y_{ii'n} \qquad (11^*)$$

where $\kappa_{ii'}$ is the changeover cost for the changeover from task i to task i'.

If variable $Y_{ii'n}$ is included, we can also tighten valid inequalities 34−36 as follows

$$\sum_{i \in I(j)} \sum_n D_{in} + \sum_{1<n<|N|} \sum_{i \in I(j)} \sum_{i' \in I(j)} s_{ii'} Y_{ii'n} \leq H \quad \forall\, j \qquad (34^*)$$

$$\sum_{i \in I(j)} \sum_{n' \geq n} D_{in'} + \sum_{n'>n} \sum_{i \in I(j)} \sum_{i' \in I(j)} s_{ii'} Y_{ii'n'} \leq H - T_n \quad \forall\, j, \forall\, n \qquad (35^*)$$

$$\sum_{i \in I(j)} \sum_{n' \leq n} (\alpha_i \mathrm{Wf}_{in'} + \beta_i \mathrm{Bf}_{in'}) + \sum_{n' \leq n} \sum_{i \in I(j)} \sum_{i' \in I(j)} s_{ii'} Y_{ii'n'} \leq T_n \quad \forall\, j, \forall\, n \qquad (36^*)$$

5.2. Shared Storage Tanks. The proposed model can also be extended to account for storage tanks shared among many states, a feature very common in chemical plants. To do so, we need to treat a shared storage tank s as a unit j. For each state that can be stored in tank j [i.e., $s \in S(j)$], we define a new binary variable V_{jsn} that is 1 if state s is stored in tank j during period n. If JT is the set of shared storage tanks and C_j is the capacity of storage tank j, the following two constraints are added

$$\sum_{s \in S(j)} V_{jsn} \leq 1 \quad \forall\, j \in \mathrm{JT}, \forall\, n \qquad (38)$$

$$S_{sn} \leq C_j V_{jsn} \quad \forall\, j \in \mathrm{JT}, \forall\, s \in S(j), \forall\, n \qquad (39)$$

Constraint 38 ensures that at most one state is stored in tank j at any time, and constraint 39 ensures that the inventory of state s at time point n is 0 if binary V_{jsn} is 0. In some cases, it is computationally more efficient to model variables V_{jsn} as special ordered sets of type I (SOS1) variables and express constraint 39 as an equality. Finally, if state s can be stored in more than one tank, constraint 39 is expressed for variable S_{sjn} which denotes the amount of state s at time n stored in tank j, and constraint 40 is added

$$S_{sn} = \sum_{j \in \mathrm{JT}(s)} S_{sjn} \quad \forall\, s \in S^*, \forall\, n \qquad (40)$$

where S^* is the set of states for which there is no dedicated storage and JT(s) is the set of storage tanks in which state s can be stored. It is important to note that the addition of binary variables V_{jsn} and constraints 38 and 39 results in an MILP model that is larger and more difficult to solve.

5.3. Number of Time Points. The choice of number of intervals is an important issue for all continuous-time STN/RTN models. A rigorous approach was proposed by Zhang and Sargent,[5] but as the authors indicate, the bounds on the number of intervals are very loose. In this work, we use the approach used in practically all continuous STN models, where we start with a small number of intervals and we iteratively increase the number of intervals by one until there is no improvement in the objective function for a fixed number of iterations (usually 1 or 2).

5.4. Minimization of Makespan. If the objective is to minimize the makespan, MS, for fixed demand, the following changes need to be made in the MILP model M*:

(a) The objective function is

$$\min \mathrm{MS} \qquad (11')$$

(b) Constraint 41 is added to ensure that the demand is met

$$\sum_n \mathrm{SS}_{sn} \geq d_s \quad \forall\, s \quad \text{or} \quad \sum_n \mathrm{SS}_{sn} = d_s \quad \forall\, s \qquad (41)$$

where d_s is the demand for state s at the end of the horizon.

(c) The length of the fixed time horizon, H, is replaced by the makespan, MS, in constraints 2, 34, and 35. The parameter H is used in constraints 16−18, 21, and 22 as an overestimate of the makespan.

The model for minimization of makespan consists of equations 1−10, 11′, 12−36, and 41.

As in all STN models, the computational efficiency of the proposed model decreases when the objective is the minimization of the makespan. The minimization of the makespan for a fixed demand, however, is more common in short-term scheduling. We are currently working on the development of a scheduling framework that combines mixed-integer programming and constraint pro-

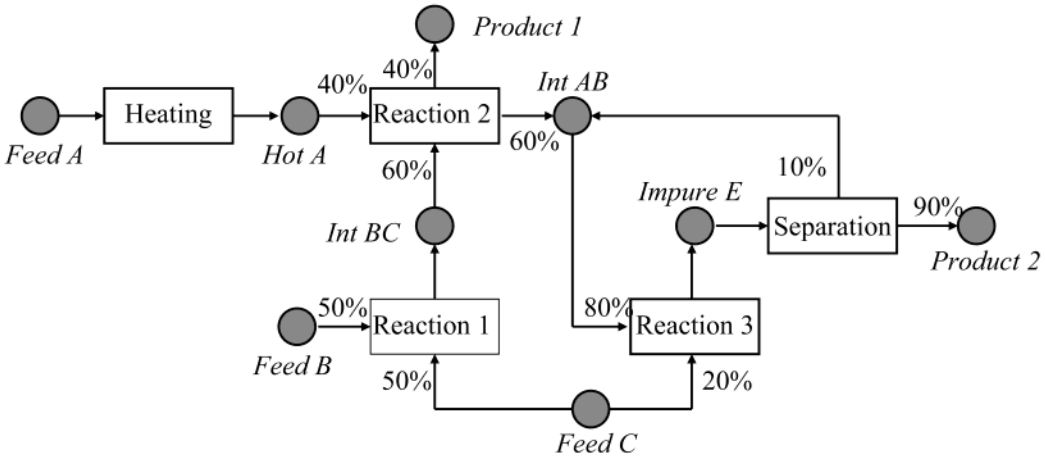


Figure 7. State−task network of example 1.

gramming (CP) techniques that successfully addresses the problem of makespan minimization.[17] Finally, it should be noted that the proposed model can be extended to handle delivery and due dates.[18]

6. Examples

In this section, we use the proposed model M* to solve a well-studied example[1] of a multipurpose batch plant, and we perform computational comparisons with the continuous-time models of Schilling and Pantelides,[4] Castro et al.[8] and Lee et al.,[11] and the event-driven model of Ierapetritou and Floudas.[7] As discussed in Appendix A, the model of Ierapetritou and Floudas[7] is not as general as the proposed model, but it seems to be among the most efficient, if not the most efficient, batch scheduling model. In the comparison reported by Castro et al.,[7] for instance, where the effect of hardware was eliminated, the Ierapetritou and Floudas model turns out to be 40−200 times faster than the general RTN model of Schilling and Pantelides[4] and, on average, twice as fast as the more recent continuous-time RTN model of Castro et al.[8] It also seems to be faster than the model of Lee et al.[11]

The example used by Kondili et al.[1] is first solved assuming constant processing times for two time horizons. Then, it is solved assuming variable processing times that are functions of the batch sizes of the tasks. The second example illustrates the handling of utility constraints and storage policies. The third example (modified from Papageorgiou and Pantelides[19]) is a medium-scale process network with all types of storage policies and utility requirements, solved in a reasonable computational time. A fourth example with sequence-dependent setup times and shared storage tanks is solved to illustrate the generality of the proposed model. The data for all examples are given in Appendix C. For the computational comparisons, we used the same hardware (a Linux workstation at 667 MHz) and software (GAMS 20.7/CPLEX 7.5) for the proposed model and the model of Ierapetritou and Floudas.[7] The CPLEX option for emphasis to feasibility, a relative optimality tolerance equal to 0.01% was used as the termination criterion, and the default GAMS/CPLEX options were used in all runs. For all instances, we solved the MILP model increasing the number of time points until there was no improvement in the objective function. The computational statistics refer to the first time grid that gave the best solution. For the models of Schilling and Pantelides[4] (S&P) and Castro et al.[8] (CBM), we have used the results reported in Castro et al.,[8] where the authors eliminate the effect of hardware differences. While for the model of Lee et al.[11] (LPL) we cite the results of the authors.

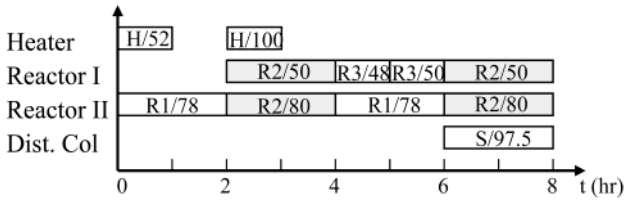


Figure 8. Gantt chart for the optimal solution of the proposed model for $H = 8$ h (six time points).

6.1. Example 1. 6.1.1. Constant Processing Times. The state−task network of the first example is depicted in Figure 7 (modified from Kondili et al.[1]). The available units are one heater for heating (H), two reactors (RI and RII) suitable for reactions R1−R3, and one distillation column suitable for separation (S). Unlimited dedicated storage is available for raw materials and final products, and dedicated FIS is available for all intermediates. Time horizons of 8 and 12 h were used to solve the corresponding problems.

The Gantt chart of the units and the batch sizes of the solution found by the proposed model M* for a time horizon of 8 h and five intervals (six time points) is shown in Figure 8. The Gantt chart of the optimal solution for the time horizon of 12 hours is given in Figure 9.

The model and solution statistics are given in Table 1. Note that, because we have implemented the model of Ierapetritou and Floudas[7] and reproduced their results, the model and solution statistics of this model are not the same as the ones reported in the original paper. Although solution statistics (CPU time and number of nodes) are expected to be different, the model statistics (number of variables and constraints) are surprisingly different. This is probably due to the fixing of some variables or the elimination of some redundant constraints that the authors performed in their original work. Because similar enhancements can be made for all STN/RTN models, here, we do not fix any variables or eliminate redundant constraints for any of the models.

When the time horizon is 8 h, the optimal solution of the proposed model is found when six time points (five intervals) are used. The optimal solution of the model of Ierapetritou and Floudas[7] is found when six event

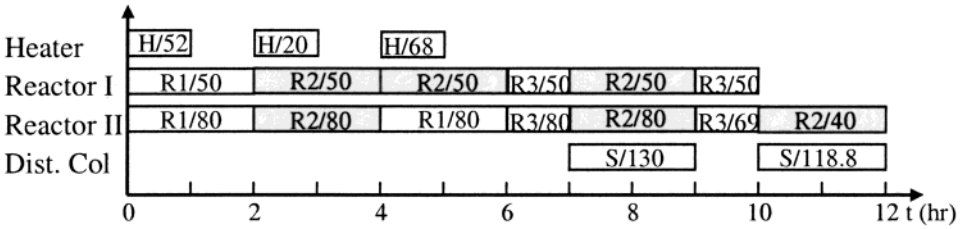


Figure 9. Gantt chart for the optimal solution of the proposed model for H = 12 h (eight time points).

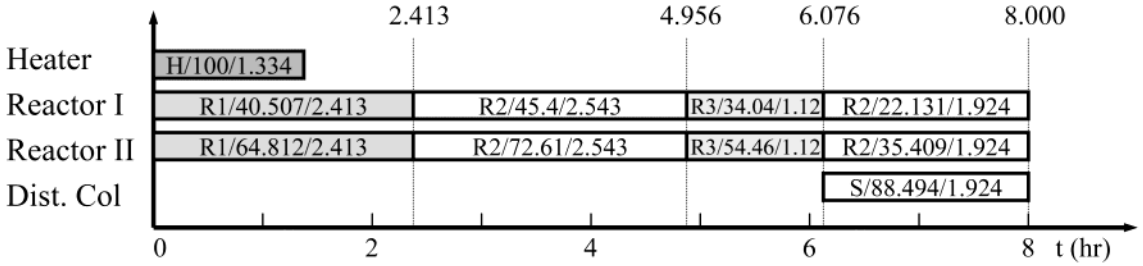


Figure 10. Equipment Gantt chart of example 1 with variable processing times.

Table 1. Model and Solution Statistics of Example 1 for Constant Processing Times

	$H = 8$				$H = 12$	
	proposed model M*		Ierapetritou and Floudas		proposed model M*	Ierapetritou and Floudas
time points (events)	5	6	5	6	8	8
binary variables	80	96	60	72	128	96
continuous variables	547	656	211	253	874	337
constraints	1194	1430	463	567	1902	775
LP relaxation	1933.8	2473.8	1933.8	2473.8	3799.4	3799.4
objective ($)	1,760.0	1,917.5	1,760.5	1,917.5	3,638.8	3,638.8
nodes	4	196	0	266	325	465
CPU time (s)	0.19	1.23	0.04	0.9	4.10	2.33

Table 2. Model and Solution Statistics of Example 1 for Variable Processing Times

	$H = 8$				
	M*	S&P	I&F	CBM	LPL
time points (events)	5	6	5	5	5
binary variables	80	89	60	80	56
continuous variables	547	375	211	226	209
constraints	1194	507	463	297	376
LP relaxation	1730.9	2191.1	1730.9	1804.4	1704.2
objective ($)	1,498.6	1,480.1	1,498.6	1480.1	1480.5
nodes	25	747	44	60	22
CPU time (s)	0.31	117	0.25	0.32	0.66

Table 3. Computational Impact of Tightening Constraints 34−36

	constant processing times				variable processing times			
	$H = 8$		$H = 12$		$H = 8$		$H = 12$	
	M	M*	M	M*	M	M*	M	M*
time points	6		8		5		7	
binary variables	96		128		80		112	
continuous variables	656		874		547		765	
constraints	1378	1430	1834	1902	1150	1194	1606	1666
LP relaxation	2541.9	2473.8	3813.2	3799.4	1931.6	1730.9	3190.5	3002.5
objective ($)	1917.5	1917.5	3638.8	3638.8	1498.6	1498.6	2610.1	2610.1
nodes	460	196	1272	465	60	23	3437	676
CPU time (s)	2.04	1.23	17.85	2.33	0.34	0.34	23.83	8.06

points are used. When the time horizon is 12 h, the two models need the same number of time points (events), and they find the same solution. Note that the proposed model needs fewer nodes, but because of its size, it requires more time.

6.1.2. Variable Processing Times. The previous example was also solved assuming variable processing times. The optimal solution for the 8-h time horizon is $1,498.6 and is found using five time points. The equipment Gantt chart of the optimal solution is shown in Figure 10 (where the batch size and duration of each task are reported). Compared to the optimal solution with constant processing times (Figure 8), we observe that, although the batch sizes are smaller, the processing times are longer; this is due to the fact that, for all tasks, the processing time for the maximum capacity corresponds to a processing time that is 33% longer than the constant processing time of Figure 8. The batch of reaction R1 in reactor RII, for example, has a batch size equal to 64.812 and a processing time equal to 2.413 h,

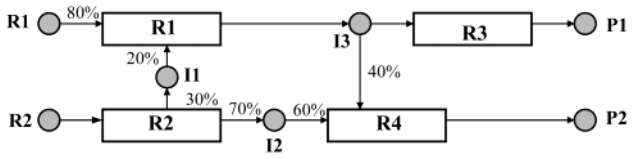


Figure 11. State−task network of example 2.

whereas in Figure 8, it has a larger batch size (78 kg) and a shorter processing time (2 h). This is the reason the optimal solution with variable processing times ($1,498.6) is lower than the one with constant processing times ($1,917.5). Model and solution statistics are given in Table 2.

Considering computational efficiency, the proposed model M* is much faster than the general models of S&P[4], CBM[8] and LPL[11] and almost as fast as the model of Ierapetritou and Floudas.[7] Regarding the quality of the solutions, we see that the solutions of all models are the same (the discrepancy between the objective values is due to small differences in the data, as discussed in Castro et al.[8] and the note by Ierapetritou and Floudas[20]).

The solution statistics of Table 3 demonstrate the effect of the addition of tightening constraints 34−36. When constant processing times are used, the effect is noticeable (especially for the larger problem), but the computational enhancement becomes even more important when variable processing times are used. In the first case, with the addition of the tightening constraints, the gap between the LP relaxation and the optimal solution is reduced by 8−11%, while for variable processing times the gap is closed by 32−46%. The number of nodes is reduced by a factor of 2−5. Note also that the improvement in computational time and number of nodes becomes greater as the problem size increases.

6.2. Example 2. To illustrate the handling of utility constraints, we consider the state−task network of Figure 11 (data in Appendix C). There are two types of reactors available for the process (type I and II) with different numbers of corresponding units available: two reactors (RI1 and RI2) of type I but only one reactor (RII) of type II. Reactions R1 and R2 require a type I reactor, whereas reactions R3 and R4 require a type II reactor. Furthermore, reactions R1 and R3 require heat, provided by steam (HS) produced in limited amounts in the plant, whereas reactions R2 and R4 are exothermic and require cooling water (CW), also available in limited amounts. For safety reasons and because of temperature restrictions, the heat integration of the process is not possible. Utility requirements include a fixed term as well as a variable term proportional to the batch size. The minimum batch size for each task is one-half of the capacity of the unit where it takes place. Processing times for all tasks are functions of the batch size. Specifically, the minimum batch size is processed in 60% of the time needed for the maximum batch size. There is unlimited storage for the raw materials and final products and finite capacity for intermediates I1−I3.

6.2.1. Maximization of Profit. To illustrate how the availability of such resources can alter the solution of a problem, we consider two cases. We first solve this problem assuming that the availability of both steam and cooling water is 40 kg/min (case 1). The optimal profit in this case is $5,904.0. The Gantt chart of units and the plot showing the level of utilization of steam and cooling water at the optimal solution are given in Figure 12. Then, we solve the problem assuming that the availability of utilities is 30 kg/min (case 2), which yields the solution shown in Figure 12 with a profit of $5,227.8.

Note that we have plotted the resource consumption level calculated by the model ,which is not necessarily equal to the actual consumption level. In Figure 12, for example, the consumption of cooling water is not constant throughout the second period, because task R2 finishes before the end of the second period. This discrepancy occurs because we use the second continuous-time representation for tasks that do not produce ZW states, allowing for a task to finish within a time period while the resource consumption is constant throughout a period (constraints 9, 10, 31, and 32). This assumption, however, does not compromise feasibility or optimality. Specifically, if a task finishes before a time point, the *calculated* resource consumption for this period will be an overestimation and is obviously feasible. Moreover, if another solution (that uses the extra resource amount) was better than the existing one, this solution would be found if one or more additional time points are used; if no better solution can be obtained, this means that the existing *overestimated* solution corresponds to the best feasible solution. For the problem of Figure 12, for instance, no better solution is found when we re-solve the problem with eight or nine time points; i.e., the utilization of the *extra* resource at the end of the second period does not yield a better solution.

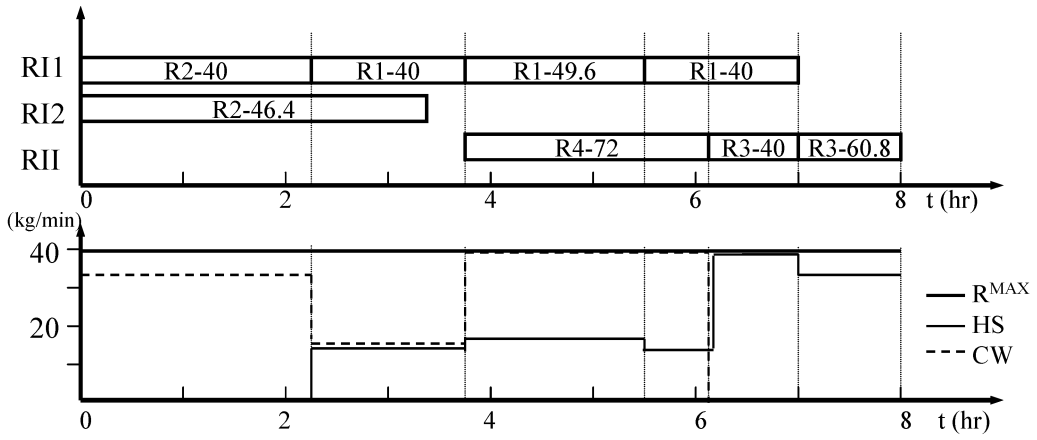


Figure 12. Solution of example 2 for maximization of profit (case 1).

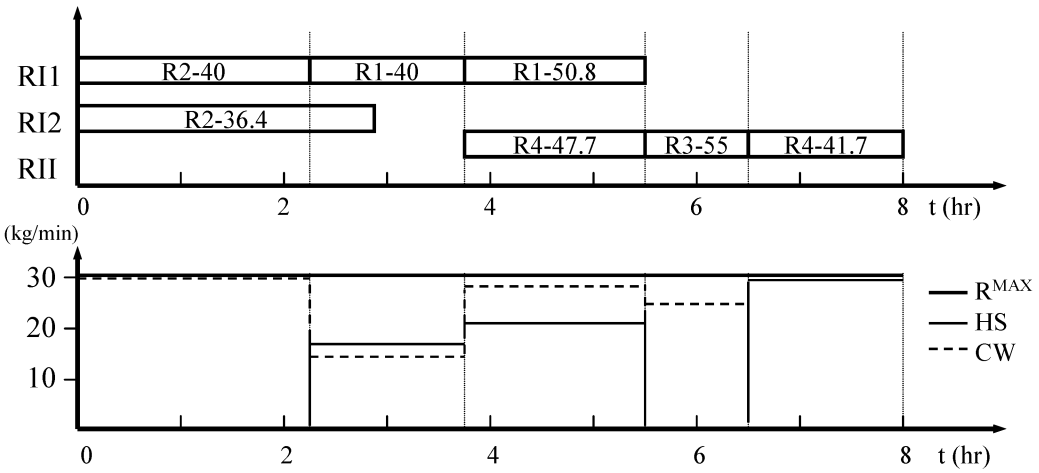


Figure 13. Solution of example 2 for maximization of profit (case 2).

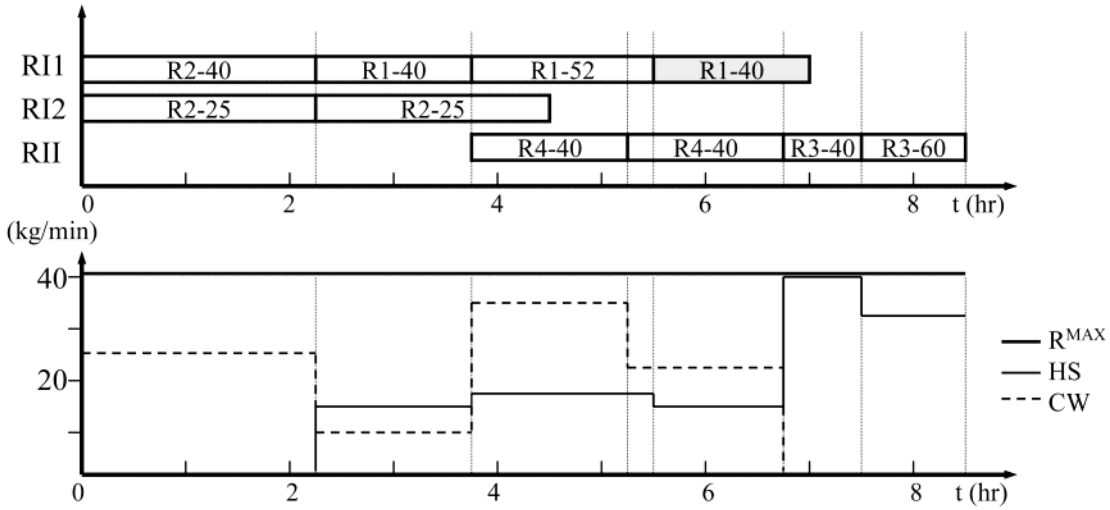


Figure 14. Solution of example 2 for minimization of makespan (case 1).

Table 4. Model and Solution Statistics of Example 2

	max profit		min makespan	
	case 1	case 2	case 1	case 2
time points	7	6	8	7
binary variables	84	72	96	84
continuous variables	661	567	753	659
constraints	1335	1146	1528	1339
LP relaxation	8875.4	7685.7	5.077	5.077
objective	$5,904.0	$5,227.8	8.5 h	9.025 h
nodes	1173	117	3411	509
CPU time (s)	9.37	1.57	36.7	6.3

In case 1, 100.8 kg of P1 and 72.0 kg of P2 are produced. As shown in Figure 12, the level of utilization of steam during the fifth interval is 40 kg/min, and the levels of consumption of cooling water during the first, third, and fourth interval are above 30 kg/min. Thus, when the maximum availability of steam and water is 30 kg/min, this solution is not feasible. As shown in Figure 13 for case 2, to keep the level of utilization of cold water bellow 30 kg/min during the first interval, the batch size of task R1 that is performed in reactor RI2 is reduced to 36.7 kg. Similarly, during the third period, the batch size of reaction R4 (performed in reactor RII) is reduced to 47.7 kg (compared to 72 kg in case 1). Furthermore, to keep the consumption of steam bellow 30 kg/min, only two batches of reaction R1 (instead of three) are performed in reactor RI1, as well as a second batch of R4 rather than two batches of reaction R3. In the optimal solution of case 2, 55 kg of P1 and 89.4 kg of P2 are produced. Note that, if the availability of both utilities is reduced to 25 kg/min, the optimal profit decreases to $4,537, and if the utility availability is further decreased to 20 kg/min, no final products can be produced. Model and solution statistics for both cases are given in Table 4.

6.2.2. Minimization of Makespan. Here, the two above cases are resolved for a fixed demand of 100 kg of P1 and 80 kg of P2, using the minimization of the makespan as the objective function. An upper bound on the makespan of 15 h ($H = 15$) is used for constraints 16−18, 21, and 22.

When the availability of resources is 40 kg/min, the makespan is 8.5 h. The equipment Gantt chart and the utility consumption graph are shown in Figure 14. When the availability of resources is reduced to 30 kg/min, the optimal makespan is 9.025 h, and the equipment Gantt chart and the utility consumption graph for this case are shown in Figure 15. Comparing the two Gantt charts, we observe that, in the second case, several tasks are delayed because of the reduced utility availability. Model and solution statistics for both cases are given in Table 4, where it can be seen that the minimization of the makespan requires greater computational effort.

6.3. Example 3. In this example, the proposed model is used for the scheduling of the STN shown in Figure 16, which is a modification of an example of Papageor-

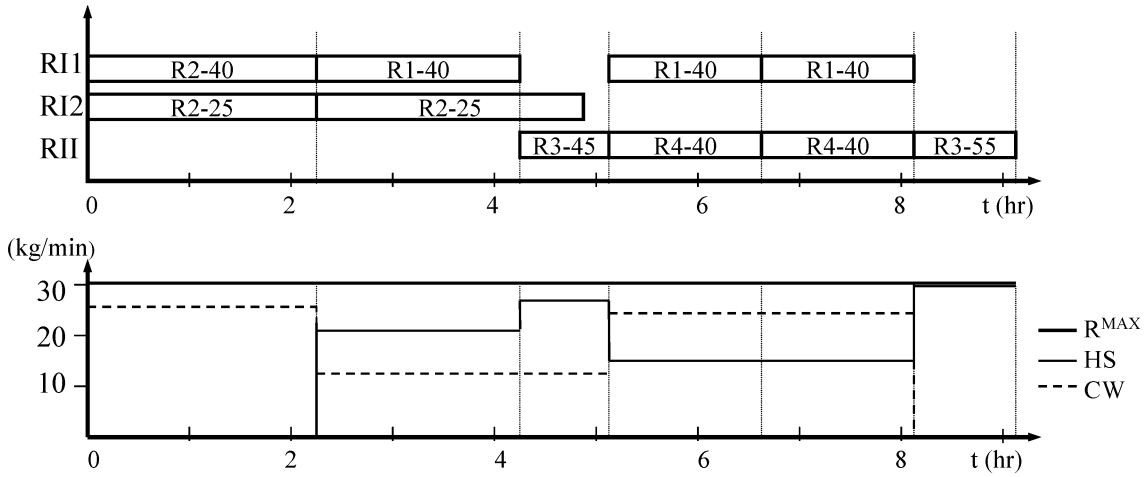


Figure 15. Solution of example 2 for minimization of makespan (case 2).

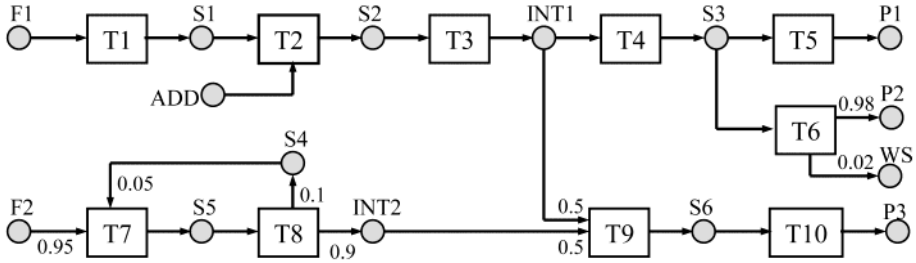


Figure 16. State−task network of example 3.

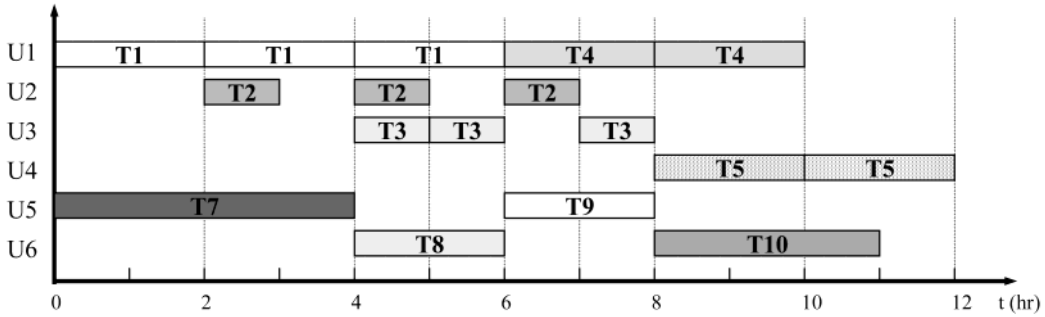


Figure 17. Equipment Gantt chart for example 3

giou and Pantelides[19] (data in Appendix C). A time horizon of 12 h is used for profit maximization. The plant consists of 6 units: unit 1 for tasks T1 and T4, unit 2 for T2, unit 3 for T3, unit 4 for T5 and T6, unit 5 for T7 and T9, and unit 6 for T8 and T10. Unlimited storage is available for raw materials F1 and F2, intermediates Int1 and Int2, and final products P1−P3; finite storage is available for states S3 and S4; no intermediate storage is available for states S2 and S6; and a zero-wait policy applies for states S1 and S5. Tasks T2, T7, T9, and T10 require cooling water (CW); tasks T1, T3, T5, and T8 require low-pressure steam (LPS); and tasks T4 and T6 require high-pressure steam (HPS). No heat integration is possible. The maximum availabilities of cooling water and low- and high-pressure steam are 25, 40 and 20 kg/min, respectively. Solving the proposed MILP model M* yields the equipment Gantt chart shown in Figure 17 and the graph of resource utilization shown in Figure 18.

The optimal solution is $13,000, and eight intervals (nine time points) are needed to obtain it. As shown in the Gantt chart of Figure 17, tasks T1 and T7 are immediately followed by tasks T2 and T8, respectively,

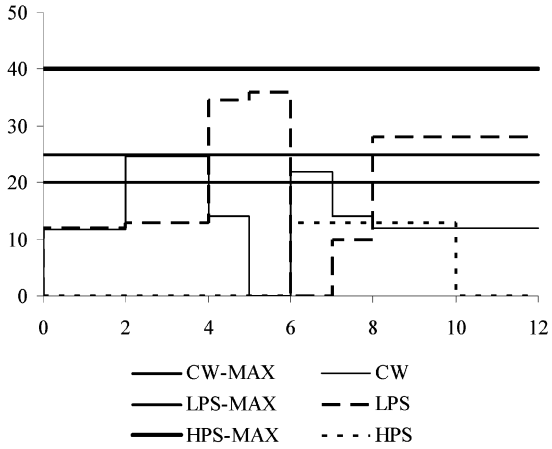


Figure 18. Resource utilization for example 3.

because they produce states S1 and S5 for which a zero-wait policy applies. Note that, for task T2, this is not necessary because S2 can remain in the unit before it is sent to the next task (NIS), whereas, for the final product P3 produced by task T10, unlimited storage is

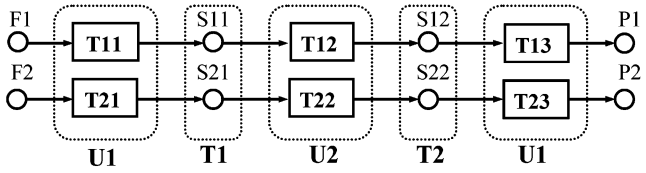


Figure 19. STN of example 4.

available. The problem consists of 3067 constraints, 180 binary, and 1587 continuous variables. Its LP relaxation is $19,500, and its optimal solution is $13,000. The optimal solution was found in 62.8 s and 2107 nodes.

Note that, if continuous-time representation I had been used, two additional time points would have been needed to capture the finish of the first batch of T2 at $t = 3$ and the finish of task T10 at $t = 11$. The resulting MILP model would have had 3833 constraints, 220 binary and 1939 continuous variables, and an LP relaxation of $20,308, and it would have obtained the optimal solution of Figure 17 ($13,000) in 2373.8 CPU s. Compared to the proposed mixed time representation, this is an increase of more than 1 order of magnitude and illustrates the effectiveness of the proposed time representation.

When the time horizon is 14 h, the optimal solution, with an objective value of $16,350, is obtained in 1,548.6 CPU s and 60 070 nodes using nine time intervals. This sharp increase in the computational effort (1548.6 vs 62.8 CPU s) is common in MILP models and implies that MILP formulations might not be used effectively for the scheduling of medium-sized plant networks with utility requirements when more than 10−12 time intervals are needed. We are currently developing an MILP/constraint programming hybrid algorithm that can potentially overcome this barrier.[17]

6.4. Example 4. Finally, the proposed model was used for the scheduling of the STN of Figure 19, which exhibits sequence-dependent setup times, shared storage tanks, and variable processing times (data in Appendix C). Specifically, there is one common storage tank T1 for states S11 and S21 and one common storage tank T2 for states S12 and S22. Tasks T11, T21, T13, and T23 take place in unit U1, and tasks T12 and T22 are performed in unit U2. The capacities of units U1 and U2 are 5 and 3 tons, respectively. The objective is to maximize the production over a fixed time horizon of 12 h while meeting a minimum demand of 2 tons for each product.

The optimal solution with an objective function of 5.019 tons is obtained using nine time points, and the equipment Gantt chart of the optimal solution is shown in Figure 20a, where, for each task, we report the batch size and its duration. For the production of P1, 2.019 tons of raw material F1 are converted into S11, which is immediately transferred to U2 and converted into S12, which is stored in T2 from $t = 5.0$ until $t = 5.3$ before it converted into 2.019 tons of final product P1. For the production of P2, 3 tons of F2 are converted into S21, which is stored in T1 from $t = 4.058$ until $t = 5.3$; intermediate S11 is converted into S22 via T22, and the latter is immediately transferred to U1 to give final product P2 through T23. In Figure 20b, we show an alternative solution with the same objective function and 10 time points, where 3 tons of raw material F1 are converted into intermediate S11, which is immediately transferred to unit U2 for task T12, and intermediate S12 is stored for 0.6 h (equal to the setup time $s_{T21,T13}$) in storage tank T2 and then is converted via task T13 into final product P1. For the production of P2, 2.019 tons of raw material F2 are converted into S21, which is stored in unit U1 from $t = 3.862$ until $t = 6.7$ and in storage tank T1 from $t = 6.7$ until $t = 7.0$, before it is transferred to U2 for task T22 and finally to unit U1 for task T23. Note that intermediate S21 could be transferred to storage tank T2 upon or at any time after the finishing of task T21, i.e., stored in tank T1 beginning at or any time after $t = 3.862$.

The model and solution statistics of the two MILP models that yield the solutions of Figure 20 are reported in Table 5. If there were no changeover times and UIS were available for intermediate states S11, S21, S12, and S22, the optimal solution of 5.192 would be obtained with five time points. As seen in Table 5, the computational cost with no changeover times is significantly lower.

7. Conclusions

A new general continuous-time MILP formulation for the short-term scheduling of STN multipurpose batch plants has been proposed. The proposed formulation is general as it accounts for batch splitting and mixing,

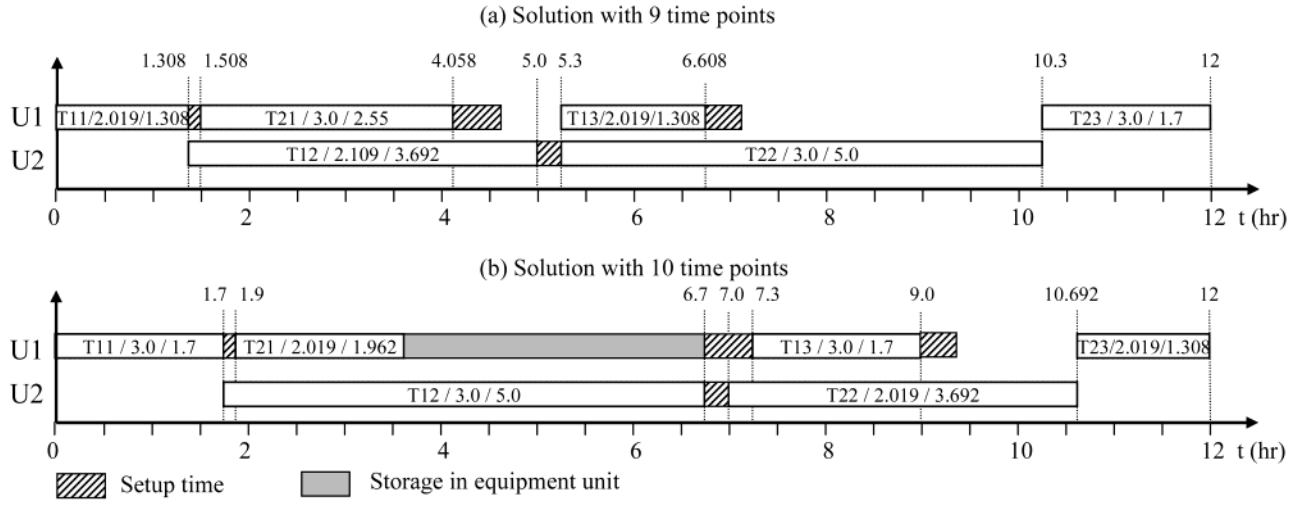


Figure 20. Equipment Gantt chart of example 4.

Table 5. Model and Solution Statistics of Example 4

	changeovers/ shared storage		no changeovers/ UIS	
	solution a	solution b	solution a	solution b
time points	9	10	5	6
binary variables	144	160	60	72
continuous variables	705	783	393	471
constraints	1627	1808	825	988
LP relaxation	7.131	7.160	6.000	6.652
objective (tons)	5.019	5.019	5.192	5.192
nodes	3117	23850	19	226
CPU time (s)	62.43	546.40	0.44	1.95

variable processing times, different storage policies (including shared storage tanks), resources other than equipment units, and sequence-dependent changeover times and costs. As shown in the examples, the proposed model is significantly faster than other general STN models. Compared to event-driven models, it is more general, and thus, better solutions might be obtained in comparable computational times.

Acknowledgment

The authors gratefully acknowledge financial support from the National Science Foundation under Grant ACI-0121497.

Nomenclature

Indices

n = time points
i = tasks
j = equipment units
r = resource categories (utilities)
s = states

Sets

$I(j)$ = set of tasks that can be scheduled on equipment unit j
$I(s)$ = set of tasks that use state s as input
JT = set of shared storage tanks
JT(s) = set of shared storage tanks in which state s can be stored
$O(s)$ = set of tasks that produce state s
$S(j)$ = set of states that can be stored in shared storage tank j
SI(i) = set of states consumed in task i
SO(i) = set of states produced from task i
ZWI = set of tasks that produce at least one ZW state

Parameters

H = time horizon
α_i = fixed duration of task i
β_i = variable duration of task i
γ_{ir} = fixed amount or utility r required for task i
δ_{ir} = variable amount of utility r required for task i
ρ_{is} = mass balance coefficient for the consumption/production of state s in task i
$C0_s$ = initial amount of state s
C_s/C_j = storage capacity for state s/shared tank j
R_r^{MAX} = upper bound for utility r
B_i^{MIN}/B_i^{MAX} = lower/upper bounds on the batch size of task i
ζ_s = price of state s
d_s = demand of state s at the end of the time horizon
$s_{ii'}$ = sequence-dependent changeover time when task i is followed by task i'
$\kappa_{ii'}$ = sequence-dependent changeover cost when task i is followed by task i'

Binary Variables

Zs_{jn} = 1 if a task in $I(j)$ is assigned to start in unit j at time point n
Zp_{jn} = 1 if a task in $I(j)$ is being processed in unit j at time point n
Zf_{jn} = 1 if a task in $I(j)$ assigned to unit j finishes at or before time point n
Ws_{in} = 1 if task i starts at time point n
Wp_{in} = 1 if task i is being processed at time point n
Wf_{in} = 1 if task i finishes at or before time point n
V_{jsn} = 1 if state s is stored in shared tank j during time period n

Continuous Variables

MS = makespan
T_n = time that corresponds to time point n (i.e., start of period n; end of period $n-1$)
Ts_{in} = start time of task i that starts at time point n
Tf_{in} = finish time of task i that starts at time point n
D_{in} = duration of task i that starts at time point n
Bs_{in} = batch size of task i that starts at time point n
Bp_{in} = batch size of task i that is being processed at time point n
Bf_{in} = batch size of task i that finishes at or before time point n
B_{isn}^{I} = amount of state s used as input for task i at time point n
B_{isn}^{O} = amount of state s produced from task i at or before time point n
S_{sn} = amount of state s available at time point n
SS_{sn} = sales of state s at point n
R_{irn}^{I} = amount of utility r consumed at time point n by task i
R_{irn}^{O} = amount of utility r released at or before time point n by task i
R_{rn} = amount of utility r utilized at time point n
$Y_{ii'n}$ = 1 if task i is immediately before task i' (starting at time point n)

Appendix A: Limitations of Event-Driven Models

As mentioned in the main body of the paper, some of the models proposed in the literature do not account for all possible configurations. To illustrate the limitations of event-driven approaches, consider the example of Figure A1. Raw material A is heated in a heater and then converted into intermediate IB through reaction R1 or reaction R2. Intermediate IB is finally purified into final product B through separation sep. A heater, two reactors, and a filter are available. Reaction R1 can take place in reactor 1 and reaction R2 in reactor 2. The maximum batch sizes and the (constant) task durations are given in Table A1. Unlimited storage is available for all states. The time horizon is 6 h. The demand for final product A is 10 kg at the due date of 6 h. It is easy to verify that the optimal schedule is the one depicted in Figure A2, which is actually the only schedule that can meet both the demand and the due date. Inventory levels for states hA and IB are also shown in Figure A2.

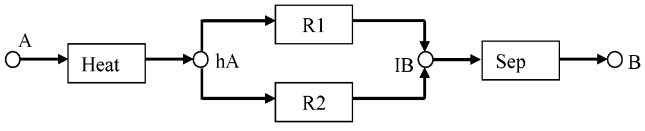


Figure A1. State−task network for the motivating example.

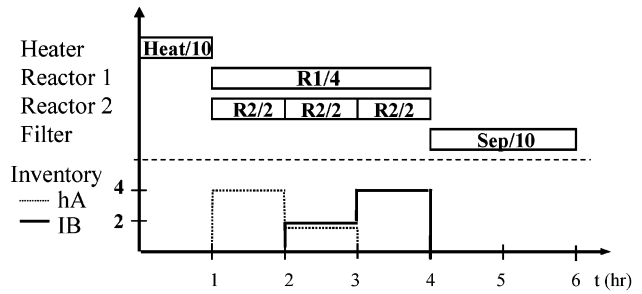


Figure A2. Gantt chart of the optimal (and uniquely feasible) solution of the motivating example.

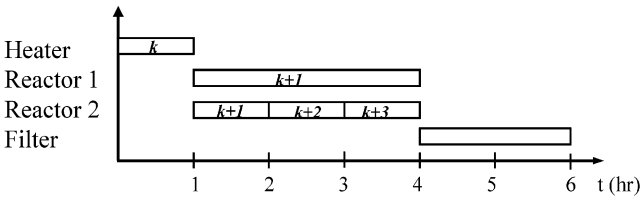


Figure A3. Numbering of event points of the motivating example.

Table A1. Data for the Motivating Example

task/unit	duration (h)	max batch size (kg)
heat/heater	1	10
R1/reactor 1	3	4
R2/reactor 2	1	2
sep/filter	2	10

In event-point approaches, the mass balance equation for state s at event point n is one of the following form

$$\mathrm{ST}_{sn} = \mathrm{ST}_{sn-1} - d_{sn} + \sum_{i\in I_s}\rho^{\mathrm{p}}_{si}\sum_{j\in J_i} B_{ij,n-1} + \sum_{i\in I_s}\rho^{\mathrm{c}}_{si}\sum_{j\in J_i} B_{ijn} \quad \forall\, s\in S,\ n\in N \quad \text{(MB)}$$

where ST_{sn} is the amount of state s at event point n; ρ^{p}_{si} and ρ^{c}_{si} are mass balance coefficients for the production and consumption of state s in task i, respectively; B_{ijn} is the batch size of task i performed in unit j at event point n; and d_{sn} is the amount of state s sold at event time n.

The schedule of Figure A2 cannot be represented by event-based formulations because there is no feasible numbering of event points that simultaneously satisfies the mass balance equation and the inventory level of IB. This important observation is further explained using Figure A3, where the numbering of event points for the heater and the two reactors is shown.

Assume that the first event point for the heater is k. To satisfy eq MB, the event points for reactors 1 and 2 that start at $t = 1$ must be numbered as $n = k + 1$. Thus, neglecting d_{sn}, the mass balance equation for state s = hA at time point $n = k + 1$ reads

$$\mathrm{ST}_{\mathrm{hA},k+1} = \mathrm{ST}_{\mathrm{hA},k} + B_{\mathrm{ht,H},k} - B_{\mathrm{rxn1,R1},k+1} - B_{\mathrm{rxn2,R2},k+1} \Rightarrow 4 = 0 + 10 - 4 - 2$$

which holds true.

The event points for the second and the third batches of reaction 2 are naturally numbered as $k + 2$ and $k + 3$, respectively, and the corresponding mass balance equations for s = hA are

$n = k + 2$:

$$\mathrm{ST}_{\mathrm{hA},k+2} = \mathrm{ST}_{\mathrm{hA},k+1} - B_{\mathrm{rxn2,R2},k+2} \Rightarrow 2 = 4 - 2$$

$n = k + 3$:

$$\mathrm{ST}_{\mathrm{hA},k+3} = \mathrm{ST}_{\mathrm{hA},k+2} - B_{\mathrm{rxn2,R2},k+3} \Rightarrow 0 = 2 - 2$$

which also hold true.

Given this numbering of Figure A3, there is no numbering for the event point of separation that simultaneously satisfies eq MB and the inventory levels of Figure A2. To prove this, let us first assume that the separation takes place at event point $n = k + 4$. Then, the mass balance equations for state IB at times $t = 2$ and $t = 4$ correspond to points $n = k + 2$ and $n = k + 4$, respectively, and read

$n = k + 2$:

$$\mathrm{ST}_{\mathrm{IB},k+2} = \mathrm{ST}_{\mathrm{IB},k+1} + B_{\mathrm{rxn1,R1},k+1} + B_{\mathrm{rxn2,R2},k+1} \Rightarrow 2 = 0 + 4 + 2$$

$n = k + 4$:

$$\mathrm{ST}_{\mathrm{IB},k+4} = \mathrm{ST}_{\mathrm{IB},k+3} + B_{\mathrm{rxn2,R2},k+3} - B_{\mathrm{sep,F},k+4} \Rightarrow 0 = 4 + 2 - 10$$

both of which are false.

Next, assume that the separation takes place at event point $k + 2$, whose start corresponds to time $t = 4$ (although, for reactor 2, the start of event $k + 2$ corresponds to $t = 2$), and that the start of event $k + 1$ corresponds to $t = 1$. The mass balance for state IB is

$$\mathrm{ST}_{\mathrm{IB},k+2} = \mathrm{ST}_{\mathrm{IB},k+1} + B_{\mathrm{rxn1,R1},k+1} + B_{\mathrm{rxn2,R2},k+1} - B_{\mathrm{sep,F},k+2} \Rightarrow 0 = 0 + 4 + 2 - 10$$

which is also false. Thus, from this example, it is clear that there is no feasible event numbering that simultaneously satisfies the mass balances and inventory levels of the optimal solution.

To address this problem, Ierapetritou and Floudas[21] proposed a modification in which the storage of a state is treated as an additional task and storage tanks are treated as units. In addition, for every storage task and event time, three constraints must be added in the formulation, two of which are big-M constraints. This modification introduces $O(|S| \times |N|)$ new binary variables, where $|S|$ is the cardinality of states and $|N|$ the cardinality of event points, and thus leads to models whose computational performance can be potentially expensive.

Another complication caused by the noncommon time grid is the monitoring of the utility usage level. Consider the case depicted in Figure 2, for example, where 10 kg/h of steam is required by tasks T1, T2, and T3. If a common grid is used, the calculation of required steam is straightforward: 10 kg/h is required during the second time interval ($t = 1 \rightarrow 2$), 20 kg/h is required during the third interval ($t = 2 \rightarrow 3$), 30 kg/h is required during the fourth interval ($t = 3 \rightarrow 4$), etc. If the event point representation is used, then the kth period is different for each task: it corresponds to $t = 1 \rightarrow 4$ for T1, $t = 3 \rightarrow 7$ for T2, and $t = 2 \rightarrow 6$ for T3. This gives rise to two problems: (a) the start and finish of the kth period are not uniquely defined, and (b) the resource consumption during the kth period has many different values. Because of these difficulties, resource constraints cannot be rigorously incorporated within the proposed event-driven models, as they have been presented

Although the method of doing so has never been shown, utility constraints can potentially be addressed by event-driven models by introducing a new state for each resource and expressing the corresponding mass balances and, more importantly, by increasing the

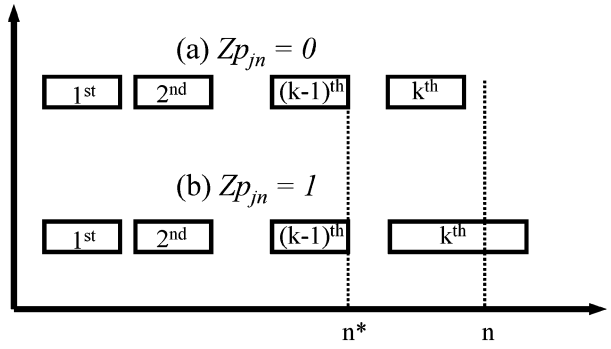


Figure B1. Alternative cases for binary Zp_{jn}.

number of events to overcome the two above-mentioned difficulties. By increasing the number of events, however, the main advantage of event-driven models, namely, the small number of events compared to models with common time partitioning, disappears.

Appendix B: Derivation of Mixed-Integer Constraints C and 12

Derivation of Equation C. For the derivation of eq C, consider the two cases depicted in Figure B1, where we have assumed that k tasks have been assigned to unit j and start before time n. Obviously, $k - 1$ of these tasks will have finished before time n. If the $(k - 1)$st task finishes at $n^* < n$, then the following equation holds true

$$\sum_{n'<n^*} Zs_{jn'} - \sum_{n'\le n^*} Zf_{jn'} = (k-1) - (k-1) = 0$$

In the case where no task is being processed in unit j at time n (Figure B1a), the kth task starts at or after n^* and finishes at or before n

$$\sum_{n^*\le n'<n} Zs_{jn'} - \sum_{n^*<n\le n} Zf_{jn'} = 1 - 1 = 0$$

and thus, eq C holds true

$$Zp_{jn} = \sum_{n'<n} Zs_{jn'} - \sum_{n'\le n} Zf_{jn'} = \left(\sum_{n'<n^*} Zs_{jn'} - \sum_{n'\le n^*} Zf_{jn'}\right) + \left(\sum_{n^*\le n'<n} Zs_{jn'} - \sum_{n^*<n'\le n} Zf_{jn'}\right) = 0 + 0 = 0$$

In the case where a task is being processed in unit j at time n (Figure B1b), the kth task starts at or after n^* but does not finish until n

$$\sum_{n^*\le n'<n} Zs_{jn'} - \sum_{n^*<n'\le n} Zf_{jn'} = 1 - 0 = 1$$

which makes eq C correct again

$$Zp_{jn} = \sum_{n'<n} Zs_{jn'} - \sum_{n'\le n} Zf_{jn'} = \left(\sum_{n'<n^*} Zs_{jn'} - \sum_{n'\le n^*} Zf_{jn'}\right) + \left(\sum_{n^*\le n'<n} Zs_{jn'} - \sum_{n^*<n'\le n} Zf_{jn'}\right) = 0 + 1 = 1$$

In a similar way, it can be shown that binary Wp_{in} is calculated by the following equation

$$Wp_{jn} = \sum_{n'<n} Ws_{jn'} - \sum_{n'\le n} Wf_{jn'} \qquad \forall j, \forall n \qquad (C^*)$$

Using eq C*, binary Wp_{in} can be eliminated in disjunction F.

Derivation of Assignment Constraint 12. Logical expressions A and B can be converted into integer equations A* and B*, respectively

$$Zs_{jn} = \sum_{i\in I(j)} Ws_{in} \qquad \forall j, \forall n \qquad (A^*)$$

$$Zf_{jn} = \sum_{i\in I(j)} Wf_{in} \qquad \forall j, \forall n \qquad (B^*)$$

Replacing implication D by its equivalent disjunction and using the definitions of binaries Zs_{jn}, Zp_{jn}, and Zf_{jn} (eqs A*, B* and C, respectively), we obtain constraint 12, which is the basic assignment constraint

$$(Zs_{jn} \Rightarrow \neg Zp_{jn}) \Leftrightarrow$$

$$(\neg Zs_{jn} \vee \neg Zp_{jn}) \Leftrightarrow$$

$$[(1 - Zs_{jn}) + (1 - Zp_{jn}) \ge 1] \Leftrightarrow$$

$$\{(1 - Zs_{jn}) + [1 - (\sum_{n'<n} Zs_{jn'} - \sum_{n'\le n} Zf_{jn'})] \ge 1\} \Leftrightarrow$$

$$(2 - \sum_{n'\le n} Zs_{jn'} + \sum_{n'\le n} Zf_{jn'} \ge 1) \Leftrightarrow$$

$$[\sum_{n'\le n} (Zs_{jn'} - Zf_{jn'}) \le 1] \Leftrightarrow$$

$$\sum_{n'\le n} (\sum_{i\in I(j)} Ws_{in'} - \sum_{i\in I(j)} Wf_{in'}) \le 1 \Leftrightarrow$$

$$\sum_{i\in I(j)} \sum_{n'\le n} (Ws_{in'} - Wf_{in'}) \le 1 \qquad \forall j, \forall n \qquad (12)$$

Appendix C: Data for Examples

The data for examples 1−4 are listed in Tables C1−C11 below.

Table C1. Data for Example 1

	A	B	C	HotA	IntAB	IntBC	ImE	P1	P2
C_s (kg)	1000	1000	1000	100	200	150	200	1000	1000
$C0_s$ (kg)	1000	1000	1000	0	0	0	0	0	0
g_s ($/kg)	0	0	0	0	0	0	0	10	10

Table C2. Data for Example 1 (Constant Processing Times)

task	duration (h)	B^{MAX}			
		heater	reactor I	reactor II	column
heating	1	100	−	−	−
reaction 1	2	−	50	80	−
reaction 2	2	−	50	80	−
reaction 3	1	−	50	80	−
separation	2	−	−	−	200

Table C3. Data for Example 1 (Variable Processing Times)

		heating		reaction 1		reaction 2		reaction 3		separation	
task	B^{MAX}	α	β	α	β	α	β	α	β	α	β
heater	100	0.667	0.006 67	–	–	–	–	–	–	–	–
reactor I	50	–	–	1.334	0.026 64	1.334	0.026 64	0.667	0.013 32	–	–
reactor II	80	–	–	1.334	0.016 65	1.334	0.016 65	0.667	0.008 325	–	–
column	200	–	–	–	–	–	–	–	–	1.3342	0.006 66

Table C4. Data for Example 2

	R1	R2	I1	I2	I3	P1	P2
C_s (kg)	1000	1000	200	100	500	1000	1000
$C0_s$ (kg)	400	400	0	0	0	0	0
g_s ($/kg)	10	15	25	0	0	30	40

Table C5. Data for Example 2[a]

min/ mean/ max d			R1 1.5/2.0/2.5		R2 2.25/3.0/3.75		R3 0.75/1.0/1.25		R4 1.5/2.0/2.5	
task	B^{MIN}	B^{MAX}	α	β	α	β	α	β	α	β
RI1	40	80	0.5	0.025	0.75	0.0375	–	–	–	–
RI2	25	50	0.5	0.04	0.75	0.06	–	–	–	–
RII	80	80	–	–	–	–	0.25	0.0125	0.5	0.025

[a] B^{MIN}/B^{MAX} in kg, α in h, β in h/kg.

Table C6. Resource Requirements for Example 2[a]

	R1		R2		R3		R4	
	γ_{iHS}	δ_{iHS}	γ_{iCW}	δ_{iCW}	γ_{iHS}	δ_{iHS}	γ_{iCW}	δ_{iCW}
RI1	6	0.25	4	0.3	–	–	–	–
RI2	4	0.25	3	0.3	–	–	–	–
RII	–	–	–	–	8	0.2	4	0.5

[a] γ in kg/min, d in kg/min per kilogram of batch.

Table C7. Data for Example 3

	F1	F2	S1	S2	S3	S4	S5	S6	INT1	INT2	P1	P2	P3
C_s (kg)	∞	∞	0	0	15	40	0	0	∞	∞	∞	∞	∞
$C0_s$ (kg)	100	100	0	0	0	10	0	0	0	0	0	0	0
g_s (10^3 $/kg)	–	–	–	–	–	–	–	–	–	–	1	1	1

Table C8. Data for Example 3[a]

task	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
unit	U1	U2	U3	U1	U4	U4	U5	U6	U5	U6
B^{MAX}	5	8	6	5	8	8	3	4	3	4
α	2	1	1	2	2	2	4	2	2	3
utility	LPS	CW	LPS	HPS	LPS	HPS	CW	LPS	CW	CW
γ	3	4	4	3	8	4	5	5	5	3
δ	2	2	3	2	4	3	4	3	3	3

[a] B^{MAX} in kg, α in h, γ in kg/min, δ in kg/min per kilogram of batch.

Table C9. Data for Example 4

	F1	F2	S11	S12	S12	S22	P1	P2
C_s (kg)	100	100	5 in T1	5 in T1	3 in T2	3 in T2	100	100
$C0_s$ (kg)	100	100	0	0	0	0	0	0
g_s ($10^3$$/kg)	–	–	–	–	–	–	1	1

Table C10. Data for Example 4[a]

task	T11	T21	T12	T22	T13	T23
unit	U1	U1	U2	U2	U3	U3
B^{MAX}	5	5	3	3	3	3
B^{MIN}	2	2	1.2	1.2	1.2	1.2
α	1	0.5	0.5	0.75	0.75	0.5
β	0.8	0.667	0.667	0.6		

[a] B^{MAX}, B^{MIN} in tons; α in h; β in hr/ton of batch.

Table C11. Setup Times for Example 4 (h)

	T11	T21	T12	T22	T13	T23
T11	0	0.2			0.1	0.5
T21	0.3	0			0.6	0.4
T12			0	0.3		
T22			0.2	0		
T13	0.2	0.6			0	0.5
T23	0.5	0.3			0.4	0

Note Added after ASAP

An incorrect image of Figure A2 appeared in the original version of this article posted ASAP on 05/17/2003. The correct Figure A2 was posted 05/19/2003.

Literature Cited

(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*, 211−227.

(2) Shah, N.; E.; Pantelides, C. C.; Sargent, R. A General Algorithm for Short-Term Scheduling of Batch Operations II. Computational Issues. *Comput. Chem. Eng.* **1993**, *17*, 229−244.

(3) Pantelides, C. C. Unified Frameworks for the Optimal Process Planning and Scheduling. In *Proceedings on the Second Conference on Foundations of Computer Aided Operations*; AIChE: New York, 1994; pp 253−274.

(4) Schilling, G.; Pantelides, C. C. A Simple Continuous-Time Process Scheduling Formulation and a Novel Solution Algorithm. *Comput. Chem. Eng.* **1996**, *20*, S1221−1226.

(5) Zhang, X.; Sargent, R. W. H. The Optimal Operation of Mixed Production Facilities—General Formulation and Some Approaches for the Solution. *Comput. Chem. Eng.* **1996**, *20*, 897−904.

(6) Mockus, L.; Reklaitis, G. V. Continuous Time Representation Approach to Batch and Continuous Process Scheduling. 1. MINLP Formulation. *Ind. Eng. Chem. Res.* **1999**, *38*, 197−203.

(7) Ierapetritou, M. G.; Floudas, C. A. Effective Continuous-Time Formulation for Short-Term Scheduling. 1. Multipurpose Batch Processes. *Ind. Eng. Chem. Res.* **1998**, *37*, 4341−4359.

(8) Castro, P.; Barbosa-Povoa, A. P. F. D.; 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.

(9) Rodrigues, M. T.; Latre, L. G.; Rodrigues, L. C. A. Short-Term Planning and Scheduling in Multipurpose Batch Chemical Plants: A Multilevel Approach. *Comput. Chem. Eng.* **2000**, *24*, 2247−2258.

(10) Mendez, C. A.; Henning, G. P.; Cerda, J. Optimal Scheduling of Batch Plants Satisfying Multiple Product Orders with Different Due Dates. *Comput. Chem. Eng.* **2000**, *24*, 2223−2245.

(11) Lee, K.-H.; Park, H. I.; Lee, I.-B. A Novel Nonuniform Discrete Time Formulation for Short-Term Scheduling of Batch and Continuous Processes. *Ind. Eng. Chem. Res.* **2001**, *40*, 4902−4911.

(12) Giannelos, N. F.; Georgiadis, M. C. A Simple New Continuous-Time Formulation for Short-Term Scheduling of Multipurpose Batch Processes. *Ind. Eng. Chem. Res.* **2002**, *41*, 2178−2184.

(13) Raman, R.; Grossmann, I. E. Modeling and Computational Techniques for Logic-Based Integer Programming. *Comput. Chem. Eng.* **1994**, *18*, 563−578

(14) Vecchietti, A.; Grossmann, I. E. LOGMIP: A Disjunctive 0−1 Non-Linear Optimizer for Process System Models. *Comput. Chem. Eng.* **1999**, *23* (4−5), 555−556.

(15) Balas, E. Disjunctive Programming and Hierarchy of Relaxations for Discrete Optimization Problems. *SIAM J. Alg. Discuss. Math.* **1985**, *35*, 2611−2623.

(16) Raman, R.; Grossmann, I. E. Relation between MILP Modeling and Logical Inference for Chemical Process Synthesis. *Comput. Chem. Eng.* **1991**, *15* (2), 73−84.

(17) Maravelias, C. T.; Grossmann, I. E. A Hybrid MILP/CP Decomposition Approach for the Short-Term Scheduling of Multipurpose Batch Plants, Submitted for publication.

(18) Maravelias, C. T.; Grossmann, I. E. A New Continuous-Time State−Task Network Formulation for Short-Term Scheduling of Multipurpose Batch Plants with Due Dates. To be presented at PSE 2003, Kunming, China, Jun 22−27, 2003.

(19) Papageorgiou, L. G.; Pantelides, C. C. Optimal Campaign Planning/Scheduling of Multipurpose Batch/Semicontinuous Plants. 2. Mathematical Decomposition Approach. *Ind. Eng. Chem. Res.* **1996**, *35*, 510−529.

(20) Ierapetritou, M. G.; Floudas, C. A. Comments on "An Improved RTN Continuous-Time Formulation for the Short-Term Scheduling of Multipurpose Batch Plants". *Ind. Eng. Chem. Res.* **2001**, *40*, 5040−5041.

(21) Ierapetritou, M. G.; Floudas, C. A. Effective Continuous-Time Formulation for Short-Term Scheduling. 2. Continuous and Semicontinuous Processes. *Ind. Eng. Chem. Res.* **1998**, *37*, 4360−4374.

Received for review November 18, 2002
Revised manuscript received March 19, 2003
Accepted April 2, 2003

IE020923Y