Enhanced Continuous-Time Unit-Specific Event-Based Formulation for Short-Term Scheduling of Multipurpose Batch Processes: Resource Constraints and Mixed Storage Policies

Stacy L. Janak, Xiaoxia Lin, and Christodoulos A. Floudas*

Department of Chemical Engineering, Princeton University, Princeton, New Jersey 08544-5263

An enhanced continuous-time formulation is presented for the short-term scheduling of multipurpose batch plants with intermediate due dates. The proposed formulation is based on the continuous-time formulation introduced by Ierapetritou and Floudas (*Ind. Eng. Chem. Res.* **1998**, *37*, 4341 and 4360) and incorporates and elaborates on several features that include (i) various storage policies (UIS, FIS, NIS, and ZW), (ii) resource constraints, (iii) variable batch sizes and processing times, (iv) batch mixing and splitting, and (v) sequence-dependent changeover times. The key features of the proposed formulation include a continuous-time representation utilizing a necessary number of *event points* of unknown location corresponding to the activation of a task. Also, tasks are allowed to continue over several event points, enabling resource quantities to be correctly determined at the beginning of each resource utilization. Several examples are presented to illustrate the effectiveness of the proposed formulation, and comparisons with other approaches are provided.

1. Introduction

The problem of short-term scheduling for multiproduct/multipurpose batch plants has received a considerable amount of attention during the last two decades. Extensive reviews were written by Reklaitis[1] and Pantelides[2] and more recently by Floudas and Lin.[3,4] Most of the proposed approaches can be classified into two main groups based on time representation: discrete-time models and continuous-time models.

Early attempts relied on the discrete-time approach, in which the time horizon is discretized into a number of time intervals of equal duration and events such as the beginning and ending of a task are associated with the boundaries of these time intervals (Kondili et al.[5]). Specific solution techniques were developed to exploit the characteristics of the short-term scheduling problem in order to reduce the model size and improve the computational efficiency exhibited by the resulting smaller models (Shah et al.[6]). The main advantage of the discrete-time approach is that it provides a reference grid of time for all operations competing for shared resources, allowing the various constraints involved in the problem to be formulated in a relatively straightforward and simple manner. The main limitations of these discrete-time models are that (i) they correspond to an approximation of the time horizon and (ii) they result in an unnecessary increase in the number of binary variables and, consequently, in the overall size of the model.

To address the inherent limitations of the discrete-time models, methods based on continuous-time representations have been developed. All continuous-time approaches can be classified into two categories based on the type of process considered: sequential processes and general network-represented processes. The major difference between these two types of processes is that sequential processes are order- or batch-oriented and do not require the explicit consideration of mass balances. General network-represented processes correspond to the more general case in which batches can merge and/or split and material balances must be taken into account explicitly. Kondili et al.[5] proposed the general framework of the state-task network (STN) for the ambiguity-free representation of such processes. Pantelides[2] extended the STN to the resource-task network (RTN) framework, which describes processing equipment, storage, material transfer, and utilities as resources in a unified way.

One of the first methods used to formulate continuous-time models for the scheduling of single or multiple stage sequential processes is based on the concept of time slots. These time slots represent the sequence of the processing of the specified products at each stage where the product assignment to each slot and its associated processing times are variables to be determined. At each stage in the process, there can be single or multiple parallel units, and when multiple units are involved, time slots are defined for each unit. Research contributions following this direction include those presented by Pinto and Grossmann,[7−10] Karimi and McDonald,[11] and Lamba and Karimi.[12,13]

Because of the batch- or order-oriented characteristics of sequential processes, it is possible to define continuous variables directly to represent the timings of the batches without the use of time slots. This alternate direction has also been pursued to formulate continuous-time scheduling models for sequential processes and, when compared to slot-based formulations, can be more accurate and lead to better solutions. Models utilizing continuous variables for task timings are presented by Cerdá et al.,[14] Méndez and Cerdá,[15,16] Méndez et al.,[17,18] and Lee et al.[19]

For general network-represented processes, two types of approaches have been developed to formulate continuous-time scheduling models: *global event*-based models and *unit-specific event*-based models. *Global event*-based models use a set of events or time slots that are common for all tasks and all units, while *unit-specific event*-based models define events on a unit basis,

* To whom correspondence should be addressed. Tel.: (609) 258-4595. Fax: (609) 258-0211. E-mail: floudas@titan.princeton.edu.

10.1021/ie0341597 CCC: $27.50
Published on Web 04/10/2004

allowing tasks corresponding to the same event point but in different units to take place at different times. The latter is considered the most general and most rigorous representation of time used in short-term scheduling models. The earliest efforts utilizing *global event*-based models were presented by Zhang and Sargent,[20,21] Mockus and Reklaitis,[22−24] and Schilling and Pantelides.[25] Recent developments include the work presented by Castro et al.,[26,27] Majozi and Zhu,[28] Lee et al.,[29] Wang and Guignard,[30] and Maravelias and Grossmann.[31] Most of these formulations have been based on either the STN or RTN process representations.

Unit-specific event-based models have been developed by Ierapetritou and Floudas,[32−34] Ierapetritou et al.,[35] and Lin and Floudas.[36] They proposed a novel continuous-time formulation for short-term scheduling of batch, semicontinuous, and continuous processes. This formulation introduces the original concept of event points, which are a sequence of time instances located along the time axis of a unit, each representing the beginning of a task or the utilization of the unit. The location of the event points is different for each unit, allowing different tasks to start at different times in each unit for the same event point. The timings of tasks are then accounted for through special sequencing constraints. Because of the heterogeneous locations of the event points for different units, as well as the definition of an event as only the starting of a task (where, in global event-based models, events correspond to the start or finish of a task), for the same scheduling problem, the number of event points required in the *unit-specific event*-based formulation is smaller than that of a *global event*-based model. This results in a substantial reduction of the number of binary variables.

In this work, we propose an enhanced STN mixed-integer linear programming model for the short-term scheduling of multiproduct/multipurpose batch plants. The proposed approach extends the work of Ierapetritou and Floudas[32] and Lin and Floudas[36] to account for resource constraints, various storage policies (UIS, FIS, NIS, and ZW), variable batch sizes and processing times, batch mixing and splitting, and sequence-dependent changeover times. The rest of the paper is organized as follows. In section 2, the problem statement is presented, and in section 3, the mathematical formulation is described. Next, in section 4, the proposed approach is tested with example problems that appeared in the literature including both network-represented and sequential processes. Finally, in section 5, results and comparisons are provided.

2. Problem Statement

The short-term scheduling problem of multipurpose batch chemical processes is defined as follows. Given (i) the production recipe (i.e., the processing times for each task at the suitable units and the amount of the materials required for the production of each product), (ii) the available equipment and their capacity limits, (iii) the material storage policy, (iv) the required utilities and their availabilities, (v) initial raw materials and orders of final products (amounts and time), and (vi) time horizon under consideration, determine (i) the optimal sequence of tasks taking place in each unit, (ii) the amount of material being processed at each time in each unit, and (iii) the processing time of each task in each unit so as to optimize a performance criterion, for example, to minimize the makespan or to maximize the overall profit.

3. Mathematical Formulation

The proposed formulation requires the indices, sets, parameters, and variables given in the Nomenclature section.

On the basis of this notation, the mathematical model for the short-term scheduling of batch plants with mixed storage policy and resource constraints involves the following constraints:

Allocation Constraints.

$$\sum_{i\in I_j} w(i,n) \le 1, \quad \forall\, j \in J,\ n \in N \tag{1}$$

These constraints express the requirement that, for each unit j and at each event point n, only one of the tasks that can be performed in this unit (i.e., $i \in I_j$) should take place.

$$w(i,n) = \sum_{n'\le n} \mathrm{ws}(i,n') - \sum_{n'<n} \mathrm{wf}(i,n'), \quad \forall\, i \in I,\ n \in N \tag{2}$$

Constraints (2) relate the continuous variable, $w(i,n)$, to the binary variables, $\mathrm{ws}(i,n)$ and $\mathrm{wf}(i,n)$ so that $w(i,n)$ will take on a value of 1 if task i is activated at event point n. Thus, if task i has started at or before event point n but not finished before event point n, $w(i,n) = 1$, but if task i has started and finished before event point n, then $w(i,n) = 0$. In this way, tasks can occur over several event points instead of starting and finishing at the same event point.

$$\sum_{n\in N} \mathrm{ws}(i,n) = \sum_{n\in N} \mathrm{wf}(i,n), \quad \forall\, i \in I \tag{3}$$

Constraints (3) express that each processing task i must both start and finish during the time horizon. If $\sum_{n\in N}\mathrm{ws}(i,n) = 1$, then the task starts once in the horizon so $\sum_{n\in N}\mathrm{wf}(i,n) = 1$ and, subsequently, the task must finish once in the horizon.

$$\mathrm{ws}(i,n) \le 1 - \sum_{n'<n} \mathrm{ws}(i,n') + \sum_{n'<n} \mathrm{wf}(i,n'), \quad \forall\, i \in I,\ n \in N \tag{4}$$

Constraints (4) express that processing task i cannot start at event point n if it has started at an earlier event point n' and has not finished by event point n. Thus, if task i has started and finished before n, then $\mathrm{ws}(i,n) \le 1$, while if task i has started but not finished before n, then $\mathrm{ws}(i,n) \le 0$ and task i cannot start at event point n.

$$\mathrm{wf}(i,n) \le \sum_{n'\le n} \mathrm{ws}(i,n') - \sum_{n'<n} \mathrm{wf}(i,n'), \quad \forall\, i \in I,\ n \in N \tag{5}$$

Constraints (5) express that processing task i cannot finish at event point n unless is has started at an earlier event point n' and has not finished by event point n. Thus, if task i has started but has not finished before event point n, then $\mathrm{wf}(i,n) \le 1$, while if task i has started and has finished before event point n, then $\mathrm{wf}(i,n) \le 0$ and task i cannot finish at event point n.

Capacity Constraints: Processing Tasks.

$$\mathrm{cap}_{ij}^{\min} w(i,n) \le B(i,j,n) \le \mathrm{cap}_{ij}^{\max} w(i,n), \quad \forall\, i \in I,\ j \in J_i,\ n \in N \tag{6}$$

These constraints express the requirement for the batch size of a task i processing at a unit j, $B(i,j,n)$, to be greater than the minimum amount of material, $\text{cap}_{ij}^{\min}$, and less than the maximum amount of material, $\text{cap}_{ij}^{\max}$, that can be processed by task i in unit j. If $w(i,n) = 1$, then the constraints (6) correspond to lower and upper bounds on the batch sizes, $B(i,j,n)$. If $w(i,n) = 0$, then all of the $B(i,j,n)$ variables become zero.

Capacity Constraints: Storage Tasks.

$$B_{\text{st}}(i^{\text{st}},n) \le \text{cap}_s^{\text{st}}, \quad \forall i^{\text{st}} \in I_s^{\text{st}}, n \in N \tag{7}$$

These constraints represent the maximum available storage capacity for each storage task i^{st} at each event point n. They simply represent an upper bound on the amount of material of state s that can be stored through storage task i^{st}.

Batch-Size Matching Constraints: Processing Tasks.

$$B(i,j,n) \le B(i,j,n-1) + \text{cap}_{ij}^{\max}[1 - w(i,n-1) + \text{wf}(i,n-1)], \quad \forall\, i \in I, j \in J_i, n \in N, n > 1 \tag{8}$$

$$B(i,j,n) \ge B(i,j,n-1) - \text{cap}_{ij}^{\max}[1 - w(i,n-1) + \text{wf}(i,n-1)], \quad \forall\, i \in I, j \in J_i, n \in N, n > 1 \tag{9}$$

These constraints represent the relationship between the batch size of task i in unit j at two consecutive event points $n - 1$ and n. These constraints are required because tasks can extend over several event points, and the batch sizes at these consecutive event points must be consistent. For instance, if a task is active and finishes at $n - 1$, then the batch sizes at event points n and $n - 1$ are not related. However, if a task is active and does not finish at $n - 1$, meaning that the processing task extends to the next event point n, then we write $B(i,j,n) \le B(i,j,n-1)$ and $B(i,j,n) \ge B(i,j,n-1)$ so that there is the same amount of material present at both event points.

$$B^{\text{s}}(i,j,n) \le B(i,j,n), \quad \forall\, i \in I, j \in J_i, n \in N \tag{10}$$

$$B^{\text{s}}(i,j,n) \le \text{cap}_{ij}^{\max}\text{ws}(i,n), \quad \forall\, i \in I, j \in J_i, n \in N \tag{11}$$

$$B^{\text{s}}(i,j,n) \ge B(i,j,n) - \text{cap}_{ij}^{\max}[1 - \text{ws}(i,n)], \quad \forall\, i \in I, j \in J_i, n \in N \tag{12}$$

Constraints (10)−(12) relate the variables $B(i,j,n)$ and $B^{\text{s}}(i,j,n)$, where $B^{\text{s}}(i,j,n)$ is the amount of material starting processing at event point n. If task i starts in unit j at event point n, then they force $B^{\text{s}}(i,j,n) = B(i,j,n)$, and if task i does not start in unit j at event point n, then they force $B^{\text{s}}(i,j,n) = 0$.

$$B^{\text{f}}(i,j,n) \le B(i,j,n), \quad \forall\, i \in I, j \in J_i, n \in N \tag{13}$$

$$B^{\text{f}}(i,j,n) \le \text{cap}_{ij}^{\max}\text{wf}(i,n), \quad \forall\, i \in I, j \in J_i, n \in N \tag{14}$$

$$B^{\text{f}}(i,j,n) \ge B(i,j,n) - \text{cap}_{ij}^{\max}[1 - \text{wf}(i,n)], \quad \forall\, i \in I, j \in J_i, n \in N \tag{15}$$

Similar to the previous set of constraints, constraints (13)−(15) relate the variables $B(i,j,n)$ and $B^{\text{f}}(i,j,n)$, where $B^{\text{f}}(i,j,n)$ is the amount of material finishing processing at event point n. If task i finishes in unit j at event point n, then they force $B^{\text{f}}(i,j,n) = B(i,j,n)$, and if task i does not finish in unit j at event point n, then they force $B^{\text{f}}(i,j,n) = 0$.

Batch-Size Matching Constraints: Utility Tasks.

$$\text{BU}(i,u,n) = \gamma_{iu} w(i,n) + \delta_{iu} B(i,j,n), \quad \forall u \in U, i \in I_u, j \in J_i, n \in N \tag{16}$$

where γ_{iu} and δ_{iu} are the constant and variable terms of the amount of utility u consumed by task i in unit j at event point n. These constraints represent the amount of utility required by the unit to process $B(i,j,n)$ of material while performing task i. Thus, the amount of utility u required, $\text{BU}(i,u,n)$, depends on the batch size of the task, $B(i,u,n)$. If $w(i,n) = 1$, then $\text{BU}(i,u,n)$ equals the sum of the two terms, and if $w(i,n) = 0$, then $\text{BU}(i,u,n)$ equals zero.

$$\sum_{i \in I_u} \text{BU}(i,u,n) + B_{\text{ut}}(u,n) = \sum_{i \in I_u} \text{BU}(i,u,n-1) + B_{\text{ut}}(u,n-1), \quad \forall\, u \in U, n \in N, n > 1 \tag{17}$$

where $\text{BU}(i,u,n)$ is the amount of utility u consumed by task i at event point n and $B_{\text{ut}}(u,n)$ is the amount of utility u available at event point n. Thus, constraints (17) express the mass balance on the utilities, requiring that the amount of utility at event point n is equal to the amount of utility at event point $n - 1$.

$$\sum_{i \in I_u} \text{BU}(i,u,n) + B_{\text{ut}}(u,n) = \text{av}_u, \quad \forall\, u \in U, n \in N, n = 1 \tag{18}$$

Constraints (18) express the requirement that the amount of utility u at the first event point, including the amount available, $B_{\text{ut}}(u,n)$, and the amount consumed, $\sum_{i \in I_u}\text{BU}(i,u,n)$, must be equal to the original amount of utility u available, av_u.

Material Balances.

$$\text{ST}(s,n) = \text{ST}(s,n-1) - D(s,n) + \sum_{i \in I_s^{\text{p}}} \rho_{is} \sum_{j \in J_i} B^{\text{f}}(i,j,n-1) - \sum_{i \in I_s^{\text{c}}} \rho_{is} \sum_{j \in J_i} B^{\text{s}}(i,j,n) + \sum_{i^{\text{st}} \in I_s^{\text{st}}} B_{\text{st}}(i^{\text{st}},n-1) - \sum_{i^{\text{st}} \in I_s^{\text{st}}} B_{\text{st}}(i^{\text{st}},n), \quad \forall\, s \in S, n \in N, n > 1 \tag{19}$$

According to these constraints, the amount of material of state s at event point n is equal to that at event point $n - 1$ increased by any amounts produced or stored at event point $n - 1$, decreased by any amounts consumed or stored at event point n, and decreased by the amount required by the market at event point n, $D(s,n)$.

$$\text{ST}(s,n) = \text{STO}(s) - \sum_{i \in I_s^{\text{c}}} \rho_{is} \sum_{j \in J_i} B^{\text{s}}(i,j,n) - \sum_{i^{\text{st}} \in I_s^{\text{st}}} B_{\text{st}}(i^{\text{st}},n), \quad \forall\, s \in S, n \in N, n = 1 \tag{20}$$

$$\text{STF}(s) = \text{ST}(s,n) - D(s,n) + \sum_{i \in I_s^{\text{p}}} \rho_{is} \sum_{j \in J_i} B^{\text{f}}(i,j,n) + \sum_{i^{\text{st}} \in I_s^{\text{st}}} B_{\text{st}}(i^{\text{st}},n), \quad \forall\, s \in S, n \in N, n = 1 \tag{21}$$

Constraints (20) and (21) represent the material balance on state s at the first and last event points, respectively. The amount of state s at the first event point is equal to the initial amount, $\text{STO}(s)$, decreased by any amounts

consumed or stored in the first event point. The total amount of state s at the end of the last event point, STF(s), is equal to the amount at the beginning of the last event point, ST(s,n), increased by any amounts produced or stored at the last event point and decreased by the amount required by the market at the last event point.

Duration Constraints: Processing Tasks.

$$T^{f}(i,j,n) \geq T^{s}(i,j,n), \quad \forall\, i \in I, j \in J_i, n \in N \tag{22}$$

These constraints represent the relationship between the starting and finishing times of task i in unit j at event point n. Because tasks can extend over multiple event points, the finishing time is not assigned from the starting time but must be greater than or equal to the starting time.

$$T^{f}(i,j,n) \leq T^{s}(i,j,n) + Hw(i,n), \quad \forall\, i \in I, j \in J_i, n \in N \tag{23}$$

Constraints (23) also represent the relationship between the starting and finishing times of task i in unit j at event point n. If task i does not take place at event point n, then along with constraint (22) the finishing time is set equal to the starting time. However, if task i does take place at event point n, then the constraint is relaxed.

$$T^{s}(i,j,n) \leq T^{f}(i,j,n-1) + H[1 - w(i,n-1) + \mathrm{wf}(i,n-1)], \quad \forall\, i \in I, j \in J_i, n \in N, n > 1 \tag{24}$$

Constraints (24) relate the starting time of task i in unit j at event point n to the finishing time of the same task in the same unit at the previous event point, $n - 1$. These constraints are relaxed unless task i is active and does not finish processing at event point $n - 1$. In this case, task i must extend to the following event point, n, so that this constraint, $T^{s}(i,j,n) \leq T^{f}(i,j,n-1)$, along with the sequencing constraint (29), $T^{s}(i,j,n) \geq T^{f}(i,j,n-1)$, results in the two times being equal.

$$T^{f}(i,j,n') - T^{s}(i,j,n) \geq \alpha_{ij}\mathrm{ws}(i,n) + \beta_{ij}B^{s}(i,j,n) - H[1 - \mathrm{ws}(i,n)] - H[1 - \mathrm{wf}(i,n')] - H\Big[\sum_{n \leq n'' < n'} \mathrm{wf}(i,n'')\Big], \quad \forall\, i \in I, j \in J_i, n \in N, n' \in N, n \leq n' \tag{25}$$

$$T^{f}(i,j,n') - T^{s}(i,j,n) \leq \alpha_{ij}\mathrm{ws}(i,n) + \beta_{ij}B^{s}(i,j,n) + H[1 - \mathrm{ws}(i,n)] + H[1 - \mathrm{wf}(i,n')] + H\Big[\sum_{n \leq n'' < n'} \mathrm{wf}(i,n'')\Big], \quad \forall\, i \notin I^{ps}, j \in J_i, n \in N, n' \in N, n \leq n' \tag{26}$$

Constraints (25) and (26) relate the starting time of task i in unit j at event point n with its finishing time at a later event point n'. If task i starts at event point n and finishes at event point n', then the two constraints force $T^{f}(i,j,n') = T^{s}(i,j,n) + \alpha_{ij} + \beta_{ij}B^{s}(i,j,n)$. However, if task i finishes at event point n'' where $n \leq n'' < n'$ or does not finish at event point n', then the constraints are relaxed. Note that constraint (26) is only valid for tasks i, which are not in the set I^{ps}, meaning that they cannot both process and store material. Thus, the processing time is not fixed for tasks that can both process and store material.

Duration Constraints: Storage Tasks.

$$T^{f}_{st}(i^{st},n) \geq T^{s}_{st}(i^{st},n), \quad \forall\, i^{st} \in I^{st}_{s}, n \in N \tag{27}$$

These constraints relate the starting and finishing times of a storage task (i^{st}) so that the finishing time must always be greater than or equal to the starting time.

Duration Constraints: Utility Tasks.

$$T^{f}_{ut}(u,n) \geq T^{s}_{ut}(u,n), \quad \forall\, u \in U, n \in N \tag{28}$$

These constraints relate the starting and finishing times of changes in the amount of utility u so that the finishing time must always be greater than or equal to the starting time.

Sequence Constraints: Same Task in the Same Unit.

$$T^{s}(i,j,n) \geq T^{f}(i,j,n-1), \quad \forall\, i \in I, j \in J_i, n \in N, n > 1 \tag{29}$$

These sequence constraints state that task i starting at event point n should start after the end of the same task performed at the same unit j that has finished at the previous event point, $n - 1$.

Sequence Constraints: Different Tasks in the Same Unit.

$$T^{s}(i,j,n) \geq T^{f}(i',j,n-1) - H[1 - w(i',n-1)], \quad \forall\, j \in J, i \in I_j, i' \in I_j, i \neq i', n \in N, n > 1 \tag{30}$$

The constraints (30) are written for tasks i and i' that are performed in the same unit j at event points n and $n - 1$, respectively. If both tasks take place in the same unit, they should be, at most, consecutive. Thus, if task i' occurs at event point $n - 1$, then the starting time of the following task must be greater than the finishing time of that task [i.e., $T^{s}(i,j,n) \geq T^{f}(i',j,n-1)$]. However, if task i' does not occur at event point $n - 1$, then the constraint is relaxed and the two times are not related.

Sequence Constraints: Different Tasks in Different Units.

$$T^{s}(i,j,n) \geq T^{f}(i',j',n-1) - H[1 - \mathrm{wf}(i',n-1)], \quad \forall\, s \in S, i \in I^{c}_{s}, i' \in I^{p}_{s}, j \in J_i, j' \in J_{i'}, j \neq j', n \in N, n > 1 \tag{31}$$

These constraints relate tasks i and i' that are performed in different units j and j' but take place consecutively according to the production recipe. Note that if task i' finishes in unit j' at event point $n - 1$, then we have $T^{s}(i,j,n) \geq T^{f}(i',j',n-1)$ and hence task i in unit j has to start after the end of task i' in unit j'. Otherwise, the constraint is relaxed and the two times are not related.

Sequence Constraints: No-Wait Condition (ZW Policy).

$$T^{s}(i,j,n) \leq T^{f}(i',j',n-1) + H[2 - \mathrm{wf}(i',n-1) - \mathrm{ws}(i,n)], \quad \forall\, s \in S^{z}, S^{f}, S^{n}, i \in I^{c}_{s}, i' \in I^{p}_{s}, j \in J_i, j' \in J_{i'}, j \neq j', n \in N, n > 1 \tag{32}$$

These constraints are written for different tasks i and i' that take place consecutively with the "zero-wait" (ZW) condition because of storage restrictions on the intermediate material. Combined with constraint (31), these constraints enforce that task i in unit j at event point n

starts immediately after the end of task i' in unit j' at event point $n-1$ if both tasks are activated.

Sequence Constraints: Storage Tasks.

$$T^{s}(i,j,n) \geq T^{f}_{st}(i^{st},n-1), \quad \forall\, s \in S,\ i \in I^{c}_{s},\ j \in J_i,\ i^{st} \in I^{st}_{s},\ n \in N,\ n > 1 \quad (33)$$

$$T^{s}(i,j,n) \leq T^{f}_{st}(i^{st},n-1) + H[1 - \mathrm{ws}(i,n)], \quad \forall\, s \in S^{f},\ i \in I^{c}_{s},\ j \in J_i,\ i^{st} \in I^{st}_{s},\ n \in N,\ n > 1 \quad (34)$$

These constraints relate the starting time of a processing task i at an event point n to the finishing time of a storage task (i^{st}) at the previous event point, $n-1$. Note that the constraint (34) is only written for states s with finite intermediate storage (FIS). Thus, if task i starts at event point n and consumes a state s that requires FIS, then we have $T^{s}(i,j,n) = T^{f}_{st}(i^{st},n-1)$ for all storage tasks i^{st} for state s. However, if task i does not start at event point n or storage task i^{st} for state s does require FIS, then the timing between the tasks is not enforced. In this way, the timing of the storage tasks for states with FIS and the consecutive processing task that consumes the FIS state are related.

$$T^{s}_{st}(i^{st},n) \geq T^{f}(i',j',n-1) - H[1 - \mathrm{wf}(i',n-1)], \quad \forall\, s \in S,\ i' \in I^{p}_{s},\ j' \in J_{i'},\ i^{st} \in I^{st}_{s},\ n \in N,\ n > 1 \quad (35)$$

$$T^{s}_{st}(i^{st},n) \leq T^{f}(i',j',n-1) + H[1 - \mathrm{wf}(i',n-1)], \quad \forall\, s \in S^{f},\ i' \in I^{p}_{s},\ j' \in J_{i'},\ i^{st} \in I^{st}_{s},\ n \in N,\ n > 1 \quad (36)$$

Constraints (35) and (36) relate the starting time of a storage task i^{st} at an event point n to the processing task i' at the previous event point, $n-1$. Similar to constraints (33) and (34), these constraints enforce the timing for a processing task that produces a FIS state and a storage task that stores the same FIS. Thus, if the processing task i' produces a FIS state s at event point $n-1$, meaning that task i' finishes at $n-1$, then we have $T^{s}_{st}(i^{st},n) = T^{f}(i',j',n-1)$. If task i' does not finish at $n-1$, the timing with the storage task at the next event point n is not enforced. In this way, the timing of a processing task that produces a FIS state and the consecutive storage task for the FIS state are related.

$$T^{s}_{st}(i^{st},n) = T^{f}_{st}(i^{st},n-1), \quad \forall\, i^{st} \in I^{st}_{s},\ n \in N,\ n > 1 \quad (37)$$

Constraints (37) relate the starting and finishing time of a storage task i^{st} at two consecutive event points. They ensure that, along with constraints (33)−(36), the timing for the storage of FIS states will be enforced so that storage limitations are not violated.

Sequence Constraints: Utility Related Tasks.

$$T^{f}(i,j,n-1) \geq T^{s}_{ut}(u,n) - H[1 - w(i,n-1) + \mathrm{wf}(i,n-1)], \quad \forall\, u \in U,\ i \in I_u,\ j \in J_i,\ n \in N,\ n > 1 \quad (38)$$

$$T^{f}(i,j,n-1) \leq T^{s}_{ut}(u,n) + H[1 - w(i,n-1)], \quad \forall\, u \in U,\ i \in I_u,\ j \in J_i,\ n \in N,\ n > 1 \quad (39)$$

These constraints relate the finishing time of a processing task i that utilizes utility u at an event point $n-1$ to the starting time of the usage of utility u at the next event point. If task i that uses utility u is activated and does not finish processing at event point $n-1$, then the starting time of the utility usage at the next event point, $T^{s}_{ut}(u,n)$, is forced to equal the finishing time of the processing task i at the current event point, $T^{f}(i,j,n-1)$. Thus, the timing of the usage of a utility is related to the processing times of the tasks that utilize that utility.

$$T^{s}_{ut}(u,n) \geq T^{s}(i,j,n) - H[1 - w(i,n)], \quad \forall\, u \in U,\ i \in I_u,\ j \in J_i,\ n \in N \quad (40)$$

$$T^{s}_{ut}(u,n) \leq T^{s}(i,j,n) + H[1 - w(i,n)], \quad \forall\, u \in U,\ i \in I_u,\ j \in J_i,\ n \in N \quad (41)$$

Constraints (40) and (41) relate the starting time of the usage of utility u at event point n to the processing task i that utilizes utility u at the current event point. If task i that uses utility u is activated at event point n, then the starting time of task i, $T^{s}(i,j,n)$, and the starting time of the usage of utility u at the current event point, $T^{s}_{ut}(u,n)$, are forced to be equal. However, if task i that uses utility u is not activated at event point n, then the constraints are relaxed.

$$T^{s}_{ut}(u,n) = T^{f}_{ut}(u,n-1), \quad \forall\, u \in U,\ n \in N,\ n > 1 \quad (42)$$

Constraints (42) relate the starting and finishing time of the usage of utility u at two consecutive event points. They ensure that, along with constraints (38)−(41), the timing for the changes in the utility level will be consistent and the amounts of utilities used can be monitored exactly and specified limits enforced.

Order Satisfaction Constraints. The order satisfaction constraints provided here are written for problems involving network-represented processes. Note that these constraints can easily be modified for the case of sequential process problems. This is done by relating orders to units in the same manner as orders are related to tasks below.

$$\sum_{i \in I_k} \sum_{n \in N} y(k,i,n) = 1, \quad \forall\, k \in K \quad (43)$$

These constraints ensure each order k is met exactly once. Thus, each order is processed by only one task i and is delivered at exactly one event point n.

$$\mathrm{wf}(i,n) = \sum_{k \in K_i} y(k,i,n), \quad \forall\, i \in I,\ n \in N \quad (44)$$

Constraints (44) relate the delivery of an order k through task i to the activation of task i at event point n.

$$D(s,n) = \sum_{k \in K_s} \sum_{i \in I_k} \mathrm{am}_k y(k,i,n), \quad \forall\, s \in S^{p},\ n \in N \quad (45)$$

Constraints (45) relate the amount of a state s delivered at event point n, $D(s,n)$, to the amount of that state due through order k. Thus, if state s has two orders associated with it, k_1 and k_2, of amounts am_{k_1} and am_{k_2}, respectively, and orders k_1 and k_2 are both delivered at event point n, then the delivery of state s at event point n is represented as $D(s,n) = \mathrm{am}_{k_1} + \mathrm{am}_{k_2}$ and the amount of the state delivered will be equal to the amount ordered.

$$T^{f}(i,j,n) \le \text{due}_k + H[2 - y(k,i,n) - \text{wf}(i,n)], \quad \forall\, s \in S,\ k \in K_s,\ i \in I_k,\ j \in J_i,\ n \in N \quad (46)$$

$$T^{f}(i,j,n) \ge \text{due}_k - H[2 - y(k,i,n) - \text{wf}(i,n)], \quad \forall\, s \in S,\ k \in K_s,\ i \in I_k,\ j \in J_i,\ n \in N \quad (47)$$

Constraints (46) and (47) relate the time that order k is due to the actual time that order k is delivered. Thus, if order k is delivered through task i at event point n, then $y(k,i,n) = 1$ and $\text{wf}(i, n) = 1$ and the constraints yield that $T^{f}(k,i,n) = \text{due}_k$ or that the finishing time of order k associated with the production of state s is equal to the actual time that order k is due. In this way, the delivery time of order k is set equal to the time that order k is due. Note that these constraints can be relaxed with slack variables if the delivery time of an order cannot be met.

Bound Constraints.

$$T^{f}(i,j,n) \le H, \quad \forall\, i \in I, j \in J_i, n \in N$$

$$T^{s}(i,j,n) \le H, \quad \forall\, i \in I, j \in J_i, n \in N$$

$$T^{f}_{ut}(u,n) \le H, \quad \forall\, u \in U, n \in N$$

$$T^{s}_{ut}(u,n) \le H, \quad \forall\, u \in U, n \in N$$

$$T^{s}_{ut}(u,n) = 0, \quad \forall\, u \in U, n \in N, n = 1$$

$$\text{STO}(s) = 0, \quad \forall\, s \notin S^{r}$$

$$\text{STO}(s) \le \text{ST}^{0}_{s}, \quad \forall\, s \in S^{r}$$

$$\text{ST}(s,n) = 0, \quad \forall\, s \in S^{z}, S^{f}, S^{n}, n \in N$$

$$\text{ST}(s,n) \le \text{ST}^{\max}_{s}, \quad \forall\, s \in S, n \in N$$

$$D(s,n) = 0, \quad \forall\, s \notin S^{p}, n \in N$$

$$0 \le w(i,n) \le 1, \quad \forall\, i \in I, n \in N \quad (48)$$

These constraints represent bounds on several of the continuous variables. The starting and finishing times of processing tasks and changes in the utility level must all be within the time horizon. The starting time for the changes in utilities at the first event point is set to zero. The initial amounts of all nonraw material states are set to zero, the intermediate amounts of all ZW, NIS, and FIS states are set to zero, and the amounts of all nonproduct deliveries are set to zero. Also, the continuous variable representing the activation of task i in unit j at event point n, $w(i,n)$, must fall between 0 and 1.

Objective Function. There are several different objective functions that can be employed with a general short-term scheduling problem. Three of the most common types are reviewed below.

Maximization of Sales

$$\text{Max} \sum_{s \in S^{p}} \text{price}_s \Big[\sum_{n \in N} D(s,n) + \text{STF}(s)\Big] \quad (49)$$

The objective function represents the maximization of the value of the final products.

Minimization of Makespan

$$\text{Min MS} \quad (50)$$

$$\text{s.t. MS} \ge t^{f}(i,j,n), \quad \forall\, i \in I, j \in J_i, n \in N \quad (51)$$

$$\text{STF}(s) = \text{dem}_s, \quad \forall\, s \in S^{p} \quad (52)$$

The objective function represents the minimization of the makespan, MS, of the process for a fixed demand for each state s, dem_s, contained in the set of final products, S^{p}.

Minimization of Order Earliness

$$\text{Min} \sum_{k \in K} \text{due}_k - \Big[\sum_{i \in I_k} \sum_{j \in J_i} \sum_{n \in N} t^{f}(i,j,n)\, y(k,i,n)\Big] \quad (53)$$

The objective function represents the minimization of the total earliness of all orders where the bilinear term, which is a product of a continuous and a binary variable, can be replaced with a continuous variable and supporting constraints using a Glover transformation.[37,38]

3.1. Remarks. 3.1.1. Number of Event Points. In this formulation, the number of event points is determined using the same approach as that proposed by Ierapetritou and Floudas.[32] First, the problem is solved with a small number of event points to obtain a solution. Then, the number of event points is increased by one and the problem resolved to obtain a better solution. This is repeated until an additional increase in the number of event points does not result in any improvement in the objective function.

3.1.2. Sequence-Dependent Setup Times. Sequence-dependent setup times can be easily incorporated within the proposed model. A parameter, $\tau_{ii'}$, is introduced to represent the sequence-dependent setup time when task i is followed by task i', where both tasks are suitable in unit j. Then, the sequencing constraint for different tasks in the same unit, constraint (30), must be modified as follows.

$$t^{s}(i',j,n) \ge t^{f}(i,j,n-1) + \tau_{ii'} - H[1 - \text{wf}(i,n-1) - \text{ws}(i',n)], \quad \forall\, j \in J,\ i \in I_j,\ i' \in I_j,\ i \ne i',\ n \in N,\ n > 1 \quad (54)$$

3.1.3. Shared Storage Tanks. The proposed formulation can also account for storage tanks shared by several states. The specification of which states s are linked to each storage task i^{st}, each of which is associated with a specific storage unit, must be modified to reflect the current storage situation. As a result, the set I^{st}_{s} is modified so that multiple states are linked to each storage task.

3.1.4. Tightening Constraints. Following the tightening constraints suggested by Maravelias and Grossmann,[31] similar constraints are introduced to tighten the relaxed solution of the proposed enhanced formulation. Specifically, constraints (55) tighten the formulation by enforcing the condition that the summation of the processing times of the tasks assigned to a specific unit j should be less than or equal to the time horizon.

$$\sum_{i \in I_j} \sum_{n \in N} \alpha_{ij}\, \text{ws}(i,n) + \beta_{ij} B^{s}(i,j,n) \le H, \quad \forall\, j \in J \quad (55)$$

Furthermore, this condition is also enforced for each unit j at each event point n as follows.

$$\sum_{i\in I_j}\sum_{n'\geq n}\alpha_{ij}\mathrm{ws}(i,n')+\beta_{ij}B^{s}(i,j,n')\leq H-\mathrm{tt}^{s}(j,n),\quad \forall\, j\in J,\ n\in N \quad (56)$$

$$\sum_{i\in I_j}\sum_{n'<n}\alpha_{ij}\mathrm{ws}(i,n')+\beta_{ij}B^{s}(i,j,n')\leq \mathrm{tt}^{f}(j,n-1)+H\Big[1-\sum_{i\in I_j}\mathrm{wf}(i,n-1)\Big],\quad \forall\, j\in J,\ n\in N,\ n>1 \quad (57)$$

where $\mathrm{tt}^{s}(j,n)$ and $\mathrm{tt}^{f}(j,n)$ are the starting and finishing times, respectively, of the task active in unit j at event point n. They are defined as follows.

$$\mathrm{tt}^{s}(j,n)\leq t^{s}(i,j,n)+H[1-\mathrm{ws}(i,n)],\quad \forall\, j\in J,\ i\in I_j,\ n\in N$$

$$\geq t^{s}(i,j,n)-H[1-\mathrm{ws}(i,n)],\quad \forall\, j\in J,\ i\in I_j,\ n\in N \quad (58)$$

$$\mathrm{tt}^{f}(j,n)\leq t^{f}(i,j,n)+H[1-\mathrm{wf}(i,n)],\quad \forall\, j\in J,\ i\in I_j,\ n\in N$$

$$\geq t^{f}(i,j,n)-H[1-\mathrm{wf}(i,n)],\quad \forall\, j\in J,\ i\in I_j,\ n\in N \quad (59)$$

Thus, constraints (56) enforce the condition that the summation of the processing times of all tasks starting in unit j at event points n or greater must be less than or equal to the amount of time remaining. Likewise, constraints (57) enforce the condition that the summation of the processing time of all tasks finishing in unit j before event point n must be less than or equal to the amount of time that has passed up to the beginning of event point n. Note that constraints (57) are only active if a task finishes at the previous event point, $n-1$; otherwise, $\mathrm{tt}^{f}(j,n)$ will not have an exact value, and the constraint is relaxed.

The addition of constraints (55)−(59) leads to relaxed solutions with smaller sums of processing times, or smaller durations. This then leads to fewer activated binary variables, $\mathrm{ws}(i,n)$ and $\mathrm{wf}(i,n)$. Moreover, the continuous variables including the batch sizes [$B^{s}(i,j,n)$, $B^{f}(i,j,n)$, and $B(i,j,n)$] and the amounts of states [$\mathrm{ST}(s,n)$ and $\mathrm{STF}(s)$] are all bounded by the binary variables. Finally, because these continuous variables appear in the objective function, the addition of these constraints results in tighter relaxations.

3.1.5. Sequential Processes. Single and multiple stage sequential processes are batch- or order-oriented and thus do not need to include tasks or states or any of the constraints involving states. The model described in the previous section can be applied to sequential processes with a few modifications. For instance, there are no defined tasks, states, batch sizes, or material amounts, and all material balances and capacity constraints are unnecessary. Thus, the basic constraints (1)−(5), (22)−(26), (29)−(32), and part of (48) all apply. The order satisfaction constraints (43)−(47) need to be modified as previously detailed. If storage constraints are to be considered, then constraints (27) and (33)−(37) need to be included, and if resource constraints are to be considered, then constraints (28) and (38)−(42) should also be included. Furthermore, all of the binary and continuous variables and their participating constraints should be modified similarly to the order satisfaction constraints to reflect a dependence on orders associated with units instead of tasks associated with units.

3.1.6. Summary of Important Enhancements of the Proposed Formulation. The proposed formulation, although based on the novel concept of unit-specific event points developed by Floudas and co-workers,[32,33,35,36] extends their formulation to take into account constraints on resources other than equipment items, such as utilities, and also considers various storage policies such as unlimited intermediate storage (UIS), FIS, no intermediate storage (NIS), and ZW conditions. In the proposed model, tasks are allowed to continue over several consecutive event points in order to accurately monitor the utilization of resources and the storage of states so that specified limits are enforced. As a result, two sets of binary variables are employed, one that indicates whether a task starts at each event point, $\mathrm{ws}(i,n)$, and another that indicates whether a task ends at each event point, $\mathrm{wf}(i,n)$. A continuous variable, $w(i,n)$, is also employed to indicate if a task is active at each event point. In addition, new tasks are defined for the storage of states and the utilization of resources. The sequence and timing of these new tasks and the processing tasks are then related so that the timing for changes in resource levels and the amounts of states will be consistent and specified limits on both can be enforced. For instance, constraints (16)−(18) define the amount of a utility used to undertake a task and keep track of the amount of utility available at each event point, while constraints (28) and (38)−(42) relate the duration and timing of a utility to the timing of the processing tasks that utilize that utility. Furthermore, constraints (7) and (19)−(21) govern the batch size of a storage task and relate the storage task to processing tasks through material balances, while constraints (27) and (33)−(37) relate the duration and timing of the storage task to the timing of the processing tasks that produce or consume the state being stored through the storage task. Note that, in addition to the above-mentioned constraints, there are several other sets of constraints that differ from those defined in the original models of Floudas and co-workers.[32,33,35,36] For instance, the allocation constraints are expanded to constraints (1)−(5) in order to relate the above-mentioned binary and continuous variables so that no tasks overlap, no tasks are assigned to the same event point in the same unit, and all tasks that start processing must finish processing. Also, constraints (8)−(15) are added to relate the batch sizes for tasks that start and finish at the same or consecutive event points so that tasks that extend over more than one event point have consistent batch sizes at each event point. Moreover, the duration constraints for a processing task have been expanded into constraints (22)−(26) to allow tasks to extend over several event points so that the finishing time is related to a starting time from a previous event point. Similarly, order satifaction constraints were added to allow orders to be due at intermediate due dates. Constraints (43)−(47) force an order for a state to be met by a certain due date and in a certain amount. Finally, the tightening constraints (55)−(59) were added, as discussed above, to improve the relaxed linear programming (LP) solution.

Thus, the proposed continuous-time formulation for the short-term scheduling of multipurpose batch plants with resource constraints, mixed storage policy, intermediate due dates, and sequence-dependent setup times consists of constraints (1)−(48) where constraint (30) is replaced with constraint (54), one of the objective

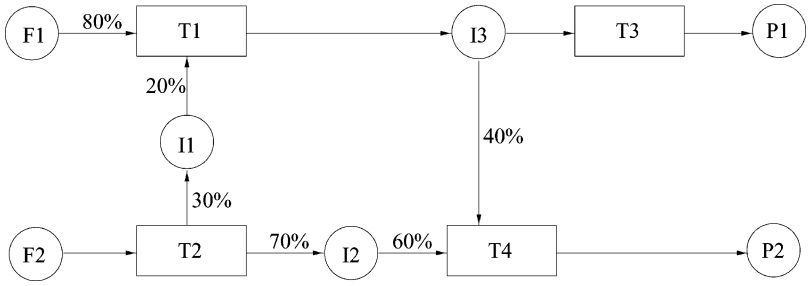

Figure 1. STN for example 1.

functions found in constraints (49), (50)−(52), and (53), and the tightenting constraints (55)−(59).

4. Computational Studies

In this section, four example problems are presented and the effectiveness of the proposed approach is illustrated. Both general network-represented and sequential processes are considered. Comparisons with previously published approaches are also provided. All examples are implemented with GAMS 2.50[39] and are solved using CPLEX 8.1 with a 3.00 GHz Linux workstation. The default GAMS/CPLEX options are used in all runs with the exception that the CPLEX option for feasibility is activated and a relative optimality tolerance equal to 0.01% was used as the termination criterion.

4.1. Example 1: Resource Constraints and Variable Batch Sizes and Processing Times. The first example comes from Maravelias and Grossmann[31] and involves the STN given in Figure 1. This example includes resource constraints and variable batch sizes, processing times, and utility requirements. The corresponding data for the example can be found in appendix B. There are two types of reactors available for the process (types I and II) with two reactors of type I (R1 and R2) and one reactor of type II (R3). There are four reactions that can take place. Reactions T1 and T2 require a type I reactor, whereas reactions T3 and T4 require a type II reactor. In addition, reactions T1 and T3 are endothermic and require heat, provided by steam (HS) available in limited amounts, and reactions T2 and T4 are exothermic and require cooling water (CW), also available in limited amounts. Each reactor allows variable batch sizes, where the minimum batch size is half of the capacity of the reactor. The utility requirements and processing times include a fixed term as well as a variable term that is proportional to the batch size. The processing times are set so that the minimum batch size is processed in 60% of the time needed for the maximum batch size. Also, there is unlimited storage for the raw materials and final products and finite storage for the intermediates.

To demonstrate the effect of resource availability on the short-term schedule of the process, we will consider two different cases. First, we assume that the availability of both HS and CW is 40 kg/min (case 1). Next, we assume that the availability of the utilities is 30 kg/min (case 2). We will also consider two different objective functions, the maximization of sales and the minimization of the makespan.

For both objective functions and both utility availability cases, the corresponding optimal production schedule and resource utilization levels are provided. Note that the optimal production schedules allow tasks to extend over more than one event point, so the Gantt charts provided show each event point at which a task is active. Thus, the same task extending over two adjacent event points appears as two separate tasks, each with the same batch size. Also, resource utilization levels are determined at each event point and thus do not necessarily reflect the actual utility consumption levels. This discrepancy occurs because of the definition of an event point as the beginning of the initialization of a task or as the beginning of the utilization of a unit. Thus, utility consumption levels are not calculated at the ends of tasks when the renewable utilities become available again. Although this method of utility record keeping is not exact, it does not allow for any infeasible or suboptimal solutions. In fact, the calculated utility consumption level is really an overestimation of the actual utility consumption level, and thus is obviously feasible. Furthermore, if another solution exists that is better than the current solution, the addition of one or more event points will yield the better solution; however, if a better solution does not exist, then the current overestimated solution corresponds to the optimal feasible solution.

4.1.1. Maximization of Sales. For case 1, using the objective function given in constraint (49) and a time horizon of 8 h, the optimal value of the sales is $6499.31. The production schedule and utility utilization levels are shown in Figure 2. For case 2, the optimal value of the sales is $5600.00 and the production schedule and resource utilization levels are shown in Figure 3.

For the maximization of sales with utility levels at 40 kg/min (i.e., case 1), note that the consumption level for both the HS and CW utilities goes above the 30 kg/min level, meaning that the solution for case 1 is not feasible for case 2. Thus, to keep the level of heating utility below 30 kg/min in case 2, the batch sizes of reaction task T1 in reactors R1 and R2 are smaller than those in case 1 and reaction task T3 in reactor R3 occurs after task T1 finishes, so that they are both not using the heating utility at the same time. Notice that this results in reaction task T3 occurring only once in case 2 instead of twice as in case 1, but it allows reaction task T4 to occur twice instead of only once as in case 1. Also, to keep the level of the CW utility below 30 kg/min for case 2, the batch sizes of reaction task T2 in reactor R1 and reaction task T4 in reactor R3 are both smaller than they were in case 1.

4.1.2. Minimization of the Makespan. When the minimization of the makespan, as given in constraints (50)−(52), is used as the objective function, the two cases given above are solved for a fixed demand of 100 kg of P1 and 80 kg of P2. For case 1, the minimum value of the makespan is 8.50 h and the production schedule and resource utilization levels are shown in Figure 4. For case 2, the minimum value of the makespan is 8.90 h and the production schedule and resource utilization levels are shown in Figure 5.

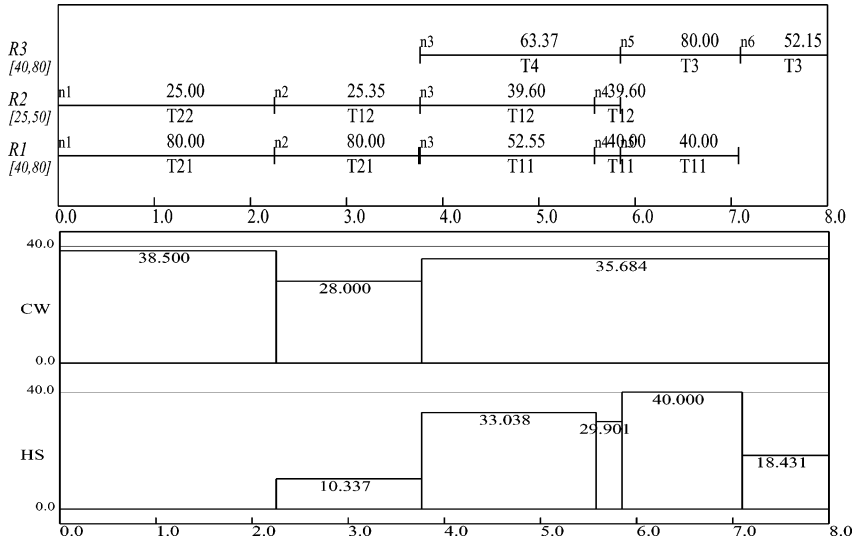

Figure 2. Schedule for example 1 for maximization of sales (case 1).

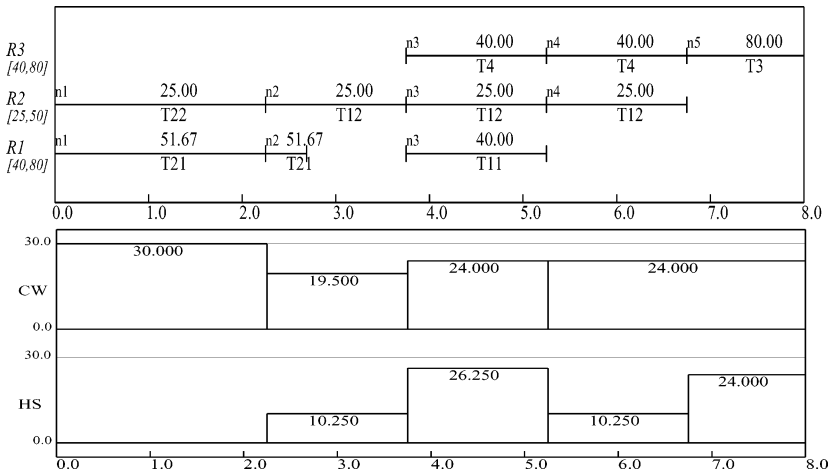

Figure 3. Schedule for example 1 for maximization of sales (case 2).

Table 1. Model and Solution Statistics for Example 1 (Proposed Formulation)

	max sales		min makespan	
	case 1	case 2	case 1	case 2
event points	6	5	6	6
binary variables	57	45	57	57
continuous variables	453	380	453	453
constraints	1528	1238	1568	1568
LP relaxation	10981.82	7200.00	7.00	7.00
objective	$6499.31	$5600.00	8.50 h	8.90 h
nodes	1246	143	42	141
CPU time (s)	5.35	0.51	0.58	0.94

For the minimization of the makespan, there is a fixed demand for the two products, so production in both cases 1 and 2 remains the same. However, in case 2, the order in which the final products are processed in reactor R3 is changed from that of case 1 to accommodate the lower level of utilities. Also, some of the reaction tasks are delayed or the batch sizes changed as a result of the final product processing sequence changes. Model and solution statistics for both objective functions and both cases are given in Table 1.

Note that the values shown in Table 1 for the number of binary variables used in the model solution reflect the number of truly unknown binary variables (e.g., task T1 in reactors R1 and R2 cannot take place at the first event point, and hence the associated binary variables are fixed to zero.)

To test the effectiveness of the proposed formulation, we performed a computational comparison for this example with the model from Maravelias and Grossmann.[31] Although this example was solved in their original paper, the objective function values and production schedules are inconsistent with the data reported in their appendix. Their objective function is reported to be the maximization of profit, while the value of the objective function is the maximization of sales. Also, the reported production schedules or resource utilization levels are inconsistent with the data given in their appendix for both cases with both objective functions, making each reported schedule inaccurate. Furthermore, the values of the objective function for the maximization of profit for both cases of the utility level and for the minimization of the makespan for the second case of the utility level are not optimal for the data reported in their appendix. The reported objectives for these problems are each suboptimal for the data provided (see the footnote in Table 2).

We re-solved both cases with both objective functions using the data presented in this paper and employing the model M* presented in Maravelias and Grossmann[31] consisting of constraints (1)−(36). The model and solution statistics for their model can be seen in Table 2. Note that extra constraints are added to model M* to account for the additional features specified in other

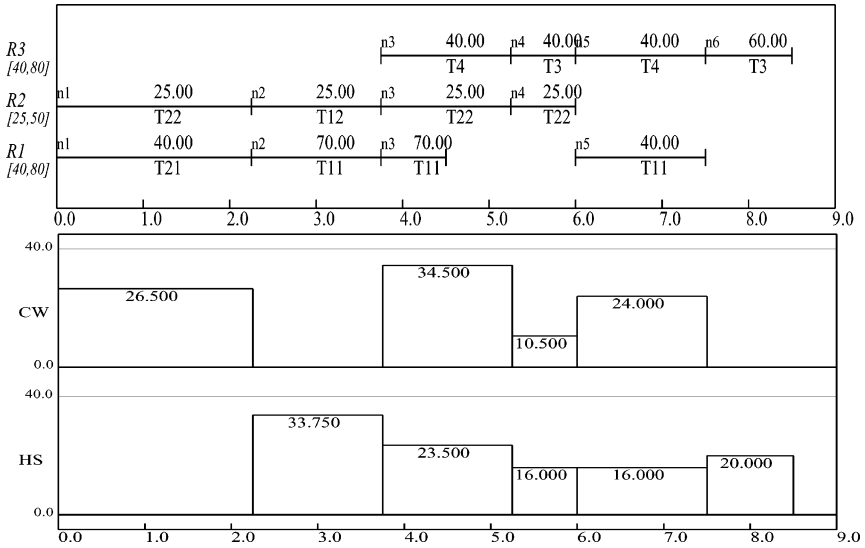

Figure 4. Schedule for example 1 for minimization of the makespan (case 1).

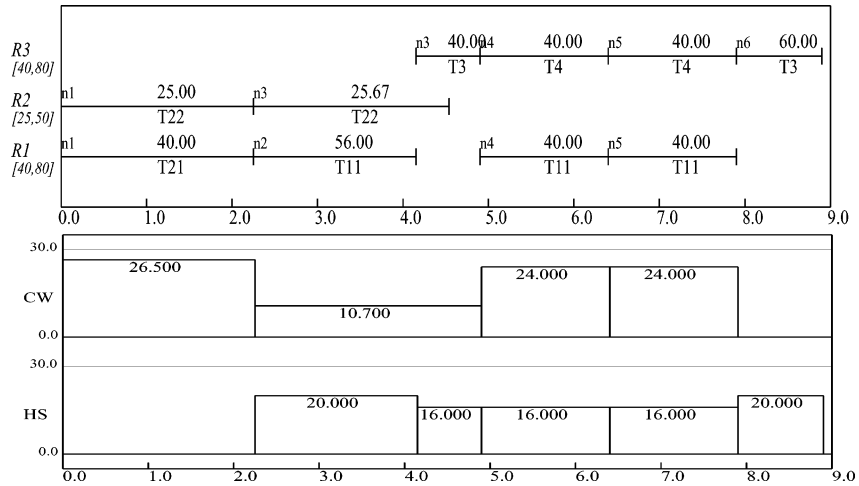

Figure 5. Schedule for example 1 for minimization of the makespan (case 2).

problems. Constraint (37) is added to model M* for the case of sequence-dependent setup times, constraints (38)−(40) are added to model M* for the case of shared storage tanks, and constraints (11′) and (41) are added to model M* when the objective function is the minimization of the makespan.

Compared to the proposed formulation, the model M* of Maravelias and Grossmann[31] always takes one more event point to determine the same objective function value. It also involves more binary and continuous variables but fewer constraints. However, note that the original paper reported using more constraints than we employed here (see the footnote in Table 2). It is possible that additional constraints were introduced that were not included in their manuscript. Although the difference in computational complexity is negligible for the two models when the maximization of sales is the objective function, it becomes much more apparent when we consider the minimization of the makespan as the objective. The model M* of Maravelias and Grossmann[31] takes over 10 times as many nodes to solve the first case and over 2 times as many nodes to solve the second case. This indicates that the consideration of unit-specific variable event points, which employ a smaller number of time points and thus fewer binary and continuous variables, may be better suited for short-term scheduling problems involving the minimization of the makespan.

4.2. Example 2: Resource Constraints, Mixed Storage Policies, and Variable Batch Sizes and Processing Times. The second example also comes from Maravelias and Grossmann[31] and involves the STN given in Figure 6. This example includes resource constraints, mixed storage policies and variable batch sizes, processing times, and utility requirements. The plant consists of 6 units involving 10 processing tasks and 14 states. UIS is available for raw materials F1 and F2, intermediates I1 and I2, and final products P1−P3 and WS. FIS is available for states S3 and S4, while NIS is available for states S2 and S6 and a ZW policy applies for states S1 and S5. There are three different renewable utilities: CW, low-pressure steam (LPS), and high-pressure steam (HPS). Tasks T2, T7, T9, and T10 require CW; tasks T1, T3, T5, and T8 require LPS; and tasks T4 and T6 require HPS. The maximum availabilities of CW, LPS, and HPS are 25, 40, and 20 kg/min, respectively. The corresponding data for the example can be found in appendix B. The objective function is the maximization of sales, and time horizons of 12 and 14 h are considered.

For a time horizon of 12 h, the optimal sales are $13 000 and eight event points are required. The production schedule and resource utilization levels can be seen in Figure 7. The problem involves 3318 constraints, 110 binary variables, and 1077 continuous

Table 2. Model and Solution Statistics for Example 1 (Maravelias and Grossmann[31] Formulation)[a]

	max sales		min makespan	
	case 1	case 2	case 1	case 2
event points	7	6	7	7
binary variables	84	72	84	84
continuous variables	661	567	661	661
constraints[b]	1145 (1335)	981 (1146)	1146 (1528)	1146 (1339)
LP relaxation	9081.31	7505.84	5.61	5.66
objective	$6499.31	$5600.00	8.50 h	8.90 h
nodes	1030	288	2318	1987
CPU time (s)	2.60	0.69	6.21	5.96

[a] For the maximization of profit with case 1 in example 2 of Maravelias and Grossmann,[31] the optimal objective function is reported to be $5904.00 instead of $6499.31, the timing of task R3 in unit RII in their Figure 12 is inconsistent for the given batch size of 40, and the utility levels for task R3 are too high. For the maximization of profit with case 2, the optimal objective function is reported to be $5227.80 instead of $5600.00, the timing of task R4 in unit RII in their Figure 13 is inconsistent for the given batch size of 47.7, and the type of utility and the utility level for tasks R3 and R4 are reversed while the utility level for task R3 is too high. For the minimization of the makespan with case 1, the utility level for task R3 is too high. For the minimization of the makespan with case 2, the optimal objective function is reported to be 9.025 instead of 8.90, the material balances in their Figure 15 are not satisfied for the first task R1 in unit RI1 and the second task R2 in unit RI2, thus the timings and utility levels for these tasks are inconsistent with the reported batch sizes and the utility level of task R3 is too large. [b] Numbers in parentheses represent values reported by Maravelias and Grossmann.[31]

variables. Its optimal solution was found in 222 nodes and 1.71 s. For a time horizon of 14 h, the optimal sales are $16 350 and eight event points are required. The production schedule and resource utilization levels can be seen in Figure 8. The problem involves 3354 constraints, 109 binary variables, and 1077 continuous variables. Its optimal solution was found in 2869 nodes and 15.65 s. Note that in both cases the limiting resource is CW, as can be seen from the resource utilization levels. In both schedules, tasks T2 and T7 occurring at the same time require the maximum amount of CW available.

Example 2 was also solved with the model M* of Maravelias and Grossmann[31] to compare the two formulations. Although this example was solved in their original paper, we have re-solved it here in order to compare the models using the same computational tools. The model and solution statistics using both models can be seen in Table 3. For the time horizon of 12 h using nine time points, the model involved 2396 constraints, 180 binary variables, and 1408 continuous variables. The same optimal solution of $13 000 was found in 23 235 nodes and 64.92 s. For the time horizon of 14 h using 10 time points, the model involved 2663 constraints, 200 binary variables, and 1564 constraints. The same optimal solution of $16 350 was found in 22 625 nodes and 112.66 s. Note that for both time horizons the model M* of Maravelias and Grossmann[31] takes at least one more time point and thus involves more binary and continuous variables. Also, the time horizon of 12 h took over 100 times more nodes to solve, while the time horizon of 14 h took over 10 times more nodes to solve. This indicates that when a larger number of time points are considered in a problem, the proposed model performs better computationally than the model of Maravelias and Grossmann,[31] even when the objective is the maximization of sales.

4.3. Example 3: Sequence-Dependent Setup Times, Shared Storage Tanks, and Variable Batch Sizes and Processing Times. The third example comes from Maravelias and Grossmann[31] and involves sequence-dependent setup times, shared storage tanks, and variable batch sizes and processing times. The STN is given in Figure 9. The process consists of two units undertaking six tasks involving eight states. There is one common storage tank, TU1, for states S11 and S12 and one common storage tank, TU2, for states S21 and S22. Tasks T11, T21, T13, and T23 are suitable in unit U1, while tasks T21 and T22 are suitable in unit U2. The corresponding data for the example can be found in appendix B. The objective is to maximize sales over a fixed time horizon of 12 h while meeting a minimum demand of 2 tons for each product.

Because of the lack of resource considerations, the simpler model outlined in Lin and Floudas[36] can be used to model this example. Note that the existence constraints and unit size constraints are not necessary. To correctly model the shared storage considerations, the set I_s^{st} was modified as described in section 3.1.3. Also, constraint (54) was included to account for sequence-dependent setup times. The optimal solution obtained was 8.96 tons using six event points. The process schedule is given in Figure 10. Note that if a state is not immediately transferred from one processing task to another, then it is stored in the appropriate storage unit using a storage task until it is processed. The problem involves 697 constraints, 36 binary, and 305 continuous variables. Its optimal solution was found in 287 nodes and 0.30 s. Note that if the problem is solved with the same number of event points with UIS for all states and without sequence-dependent setup times, it involves 299 equations, 24 binary, and 209 continuous variables and arrives at a solution of 9.51 after 176 nodes and 0.08 s. However, if we increase the event points to seven, the problem involves 351 equations, 30 binary variables, and 241 continuous variables and arrives at a solution of 10.11 after 942 nodes and 0.43 s. Thus, it can be concluded that the consideration of shared storage tanks and sequence-dependent setup times does not significantly increase the computational complexity of this problem. The number of event points

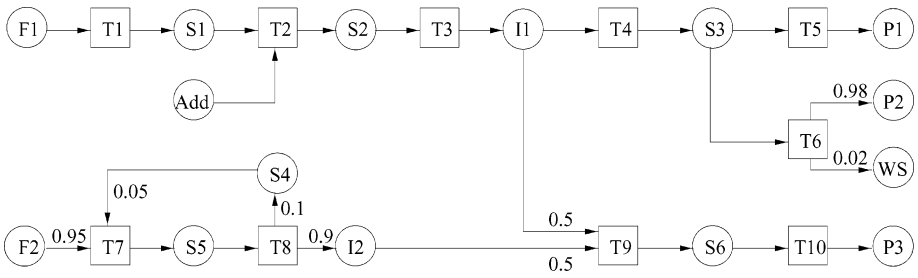

Figure 6. STN for example 2.

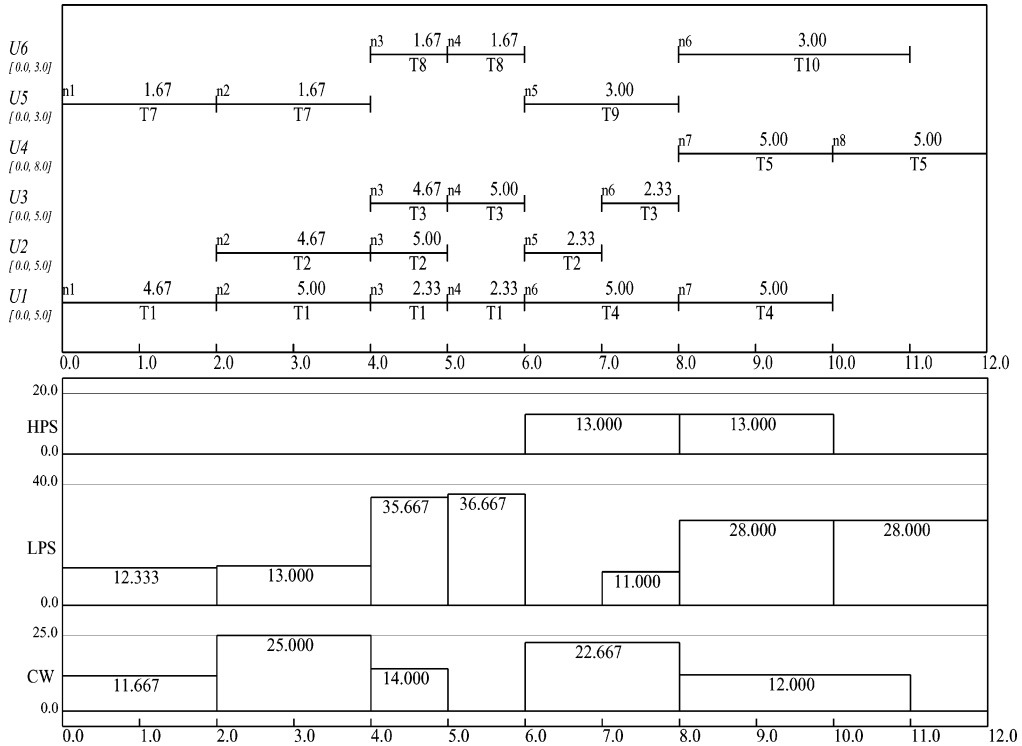

Figure 7. Schedule for example 2 with a time horizon of 12 h.

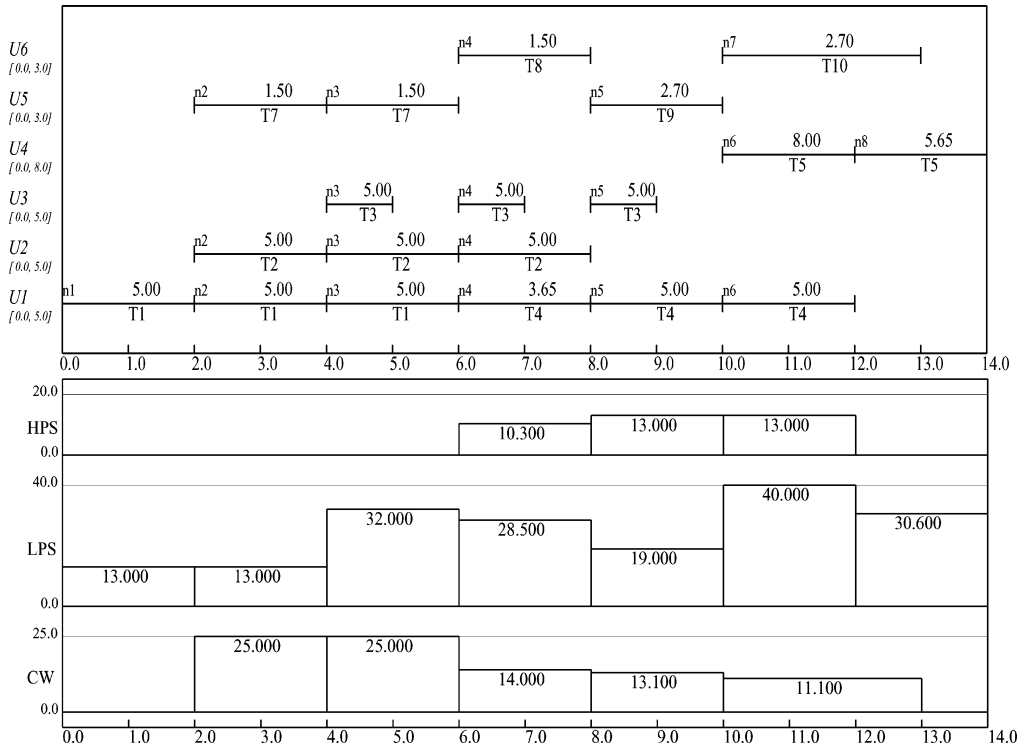

Figure 8. Schedule for example 2 with a time horizon of 14 h.

and thus the number of variables and constraints are seen to have a much more dramatic effect.

Example 3 was also solved with the model of Maravelias and Grossmann[31] to compare the two formulations when considering sequence-dependent setup times and shared storage tasks. This example was solved in their original paper, but the reported production schedule does not match the data given in their appendix and the objective function value is suboptimal for the reported data; thus, we have re-solved the problem here in order to compare the models using the same data. To achieve the same optimal objective function as the proposed formulation, we must utilize 12 time points with the model M* of Maravelias and Grossmann.[31] The problem involves 1888 constraints, 192 binary, and 975 continuous variables, and the optimal solution of 8.96 was found in 1 814 291 nodes and 11 638.24 s. Thus, the proposed formulation utilizes significantly fewer time points and thus fewer variables and, as a result, outperforms the model of Maravelias and Grossmann.[31] The model and solution statistics using both models can be seen in Table 4.

Table 3. Model and Solution Statistics for Example 2

	proposed formulation		Maravelias and Grossmann[31] formulation[a]	
	12 h	14 h	12 h	14 h
event points	8	8	9	10
binary variables	110	110	180	200
continuous variables	1077	1077	1408	1564
constraints	3318	3354	2396	2663
LP relaxation	19000	19000	18423.5	22186.7
objective	$13000	$16350	$13000	$16350
nodes[b]	222	2869	23235 (2107)	22625 (60070)
CPU time (s)	1.71	15.65	64.92	112.66

[a] The reported number of nodes and CPU seconds for both time horizons are different from those found by our application of model M* of Maravelias and Grossmann.[31] For a time horizon of 12 h, they reported 2107 nodes while our application of model M* took 23 235 nodes. For a time horizon of 14 h, they reported 60 070 nodes while our application of model M* took 22 625 nodes. [b] Numbers in parentheses represent values reported by Maravelias and Grossmann.[31]

4.4. Example 4: Sequential Process with Order-Dependent Processing Times. The fourth example is taken from Pinto and Grossmann[10] and involves a sequential process containing one stage with four parallel extruders of unequal capacity and with processing times depending on the order being processed. A total of 12 orders are due at specific times over a 30-day period. The corresponding processing rate and due date data for the example can be found in appendix B. The problem objective is to meet all orders while minimizing earliness, as seen in constraint (53).

The optimal processing schedule is given in Figure 11 with an objective function value of 1.026. The problem was modeled with the formulation of Ierapetritou and Floudas[32] using only the allocation, duration, and same task in the same unit and different tasks in the same unit sequencing constraints along with the order satisfaction constraints outlined in (43)−(47) and the objective given in constraint (53).

Suppose now that, because of limited manpower, there is a hard constraint on the number of extruders that can operate at the same time. We will consider the case where three extruders may operate simultaneously (case 1) and the case where only two extruders may operate simultaneously (case 2). For both cases, we employ the model outlined in section 3.1.5, again using the order satisfaction constraints and the objective function to minimize the earliness of the orders. For case 1 with three extruders, the optimal objective function value is 1.895 and the production schedule can be seen in Figure 12. For case 2, the optimal objective function value is 7.909 and the production schedule can be seen in Figure 13. Model and solution statistics for all three cases can be seen in Table 5.

It can be seen from Table 5 that consideration of resource constraints in the form of limited manpower increases the computational complexity of the problem. Resource considerations require a more complicated model involving more variables and constraints. For instance, resource considerations require an event point for every time the resource level changes or, in this case, for each order. However, a problem without resources only requires as many event points as the maximum number of sequential tasks. For this example, the simpler problem without resource considerations only requires four event points while the more complicated problem with resource constraints requires 12 event

Table 4. Model and Solution Statistics for Example 3

	Lin and Floudas[36] formulation	Maravelias and Grossmann[31] formulation[a]
event points	6	12
binary variables	36	192
continuous variables	305	975
constraints	697	1888
LP relaxation	12.00	13.19
objective	8.96	8.96
nodes	287	1 814 291
CPU time (s)	0.30	11 638.24

[a] For the maximization of production in example 4 of Maravelias and Grossmann,[31] the optimal objective function value is reported to be 5.019 instead of 8.096 and the reported schedules in their Figure 20 reflect data that is entirely different from that reported in their appendix. For instance, tasks T12 and T23 clearly vary in processing time between the two schedules even though they both do not have a variable term of processing time. Also, in Table C9 in their appendix, the fifth column heading should be "S21" instead of "S12" and the units of mass should be tons instead of kilograms. In Table C10 in their appendix, the first two entries in the last two columns should be unit U1 because there is no unit U3 and the minimum and maximum batch sizes for these entries should be 2.0 and 5.0, respectively, instead of 1.2 and 3.0.

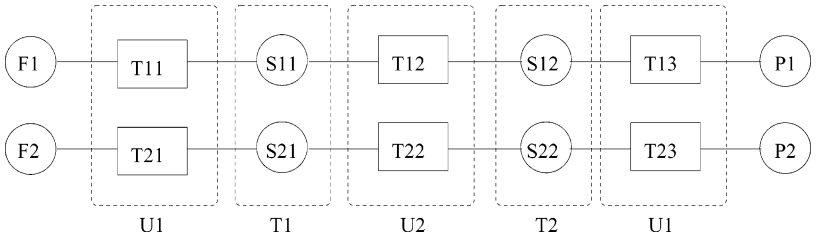

Figure 9. STN for example 3.

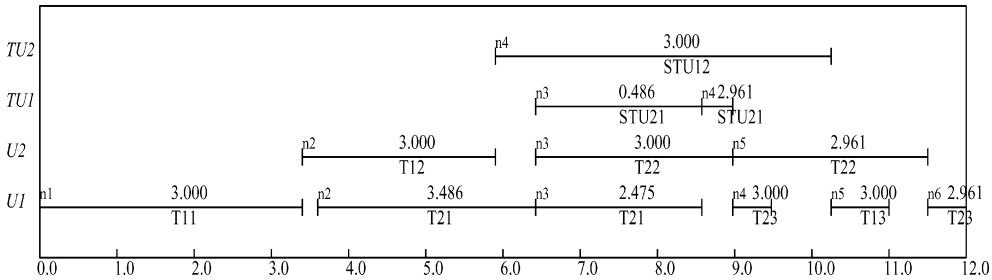

Figure 10. Schedule for example 3.

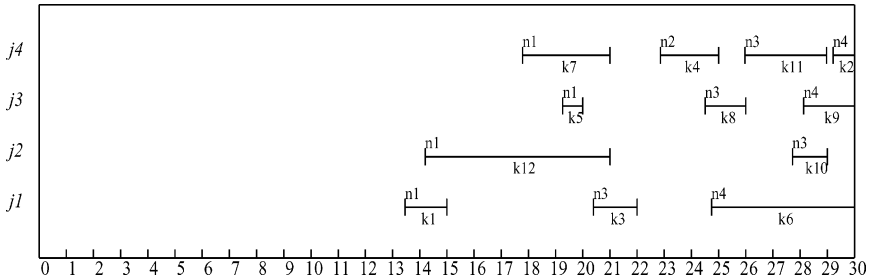

Figure 11. Schedule for example 4 without limited manpower.

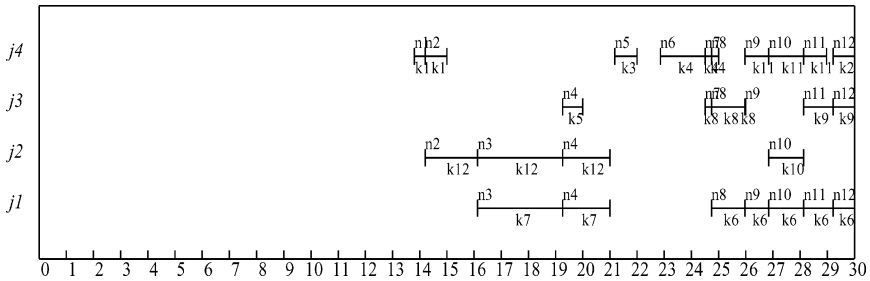

Figure 12. Schedule for example 4 with three extruders (case 1).

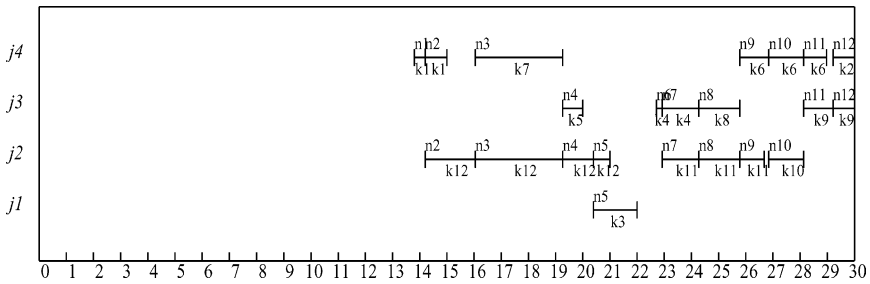

Figure 13. Schedule for example 4 with two extruders (case 2).

Table 5. Model and Solution Statistics for Example 4

	four extruders simultaneously	three extruders simultaneously	two extruders simultaneously
event points	4	12	12
binary variables	150	458	446
continuous variables	513	2137	2137
constraints	1389	10 382	10 381
LP relaxation	0	0	0
objective	1.026	1.895	7.909
nodes	7	1374	42 193
CPU time (s)	0.07	6.53	178.85

points, resulting in many more binary variables and thus a much more complex problem.

To test the effectiveness of the proposed formulation when used with sequential processes, we performed a computational comparison for this example with the model from Pinto and Grossmann.[10] Although this example was solved in their original paper, the objective function used was the maximization of starting times instead of the minimization of tardiness. So, we re-solved our model using the maximization of the starting times as the objective. Pinto and Grossmann[10] report optimal objectives of 269.10, 268.24, and 264.98 for the three cases of no resources, resources limited to three extruders, and resources limited to two extruders. Our optimal objective function values with the same objective were 269.10, 268.82, and 265.74, respectively. Thus, the proposed model found improved schedules with a better objective function value for the case where resources are limited to three extruders and the case where resources are limited to two extruders. This is not unexpected, however, because of the fact that the model used by Pinto and Grossmann[10] employs the concept of time slots and all slot-based formulations restrict the time representations and hence they can result, by definition, in suboptimal solutions. Note that model and solution statistics found in Table 5 for this problem using an objective function of minimization of order earliness are comparable to the model and solution statistics determined using an objective of maximization of starting times. We do not make a comparison with the model and solution statistics presented by Pinto and Grossmann[10] because the authors do not report integrality gaps or other optimality criterion used; thus, a direct comparison would not be meaningful.

5. Conclusions

In this paper, an enhanced continuous-time formulation is presented for the short-term scheduling of multipurpose batch plants with intermediate due dates. The proposed formulation incorporates several features including various storage policies (UIS, FIS, NIS, and ZW), resource constraints, variable batch sizes and processing times, batch mixing and splitting, and sequence-dependent changeover times. The key features of the proposed formulation include a continuous-time representation utilizing a necessary number of event points of unknown location corresponding to the activation of a task. Also, tasks are allowed to continue over several event points enabling resource quantities to be correctly determined at the beginning of each resource utilization. Four examples are presented to illustrate the effectiveness of the proposed formulation. The computational results are compared with those in the literature, and it is shown that the proposed formulation is significantly

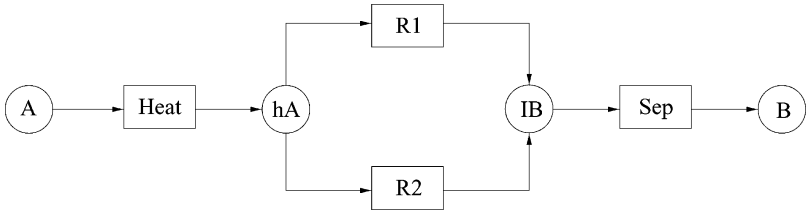

Figure 14. STN for the motivating example.

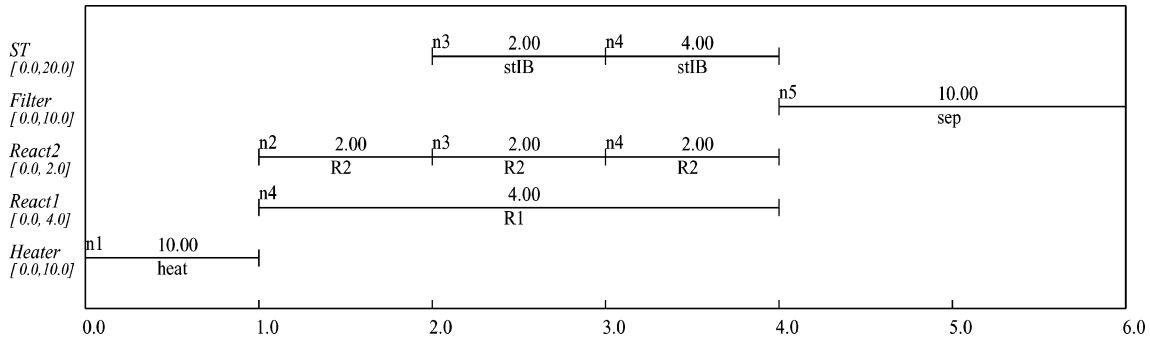

Figure 15. Production schedule for motivating example with the Lin and Floudas[36] model.

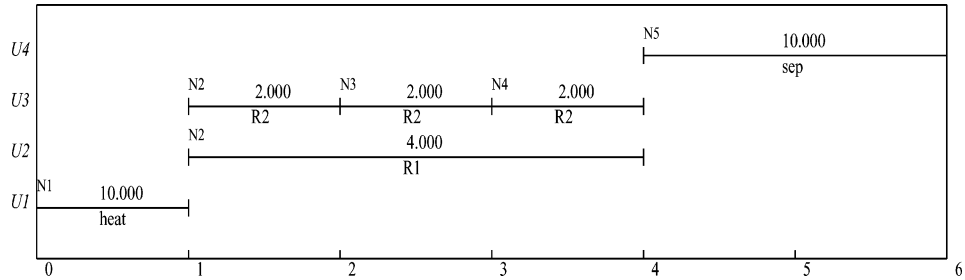

Figure 16. Production schedule for the motivating example with the Maravelias and Grossmann[31] model.

Table 6. Data for the Motivating Example

task	unit	duration (h)	cap^{max} (kg)
heat	heater	1	10
R1	reactor 1	3	4
R2	reactor 2	1	2
sep	filter	2	10

faster than other general resource-constrained models, especially for problems requiring many time points.

Acknowledgment

The authors gratefully acknowledge support from the National Science Foundation.

Appendix A: Advantages of Unit-Specific Event-Driven Models

In this appendix, we discuss the advantages of unit-specific event-driven models over models employing a continuous-time representation that is common for all units. We will consider the motivating example from the appendix of Maravelias and Grossmann[31] and compare the results from the two types of models. The STN for the motivating example can be seen in Figure 14 and involves four tasks processing four states. Each of the tasks is suitable in one of four different units. The maximum batch sizes for each unit and the fixed processing times for each task are given in Table 6. Unlimited storage is available for all states, and the time horizon of interest is 6 h. The demand for product B is 10 kg, and the due date is 6 h. We will choose the maximization of production as our objective function.

To model this example using an event-driven formulation, we will employ the model of Lin and Floudas[36] without the existence and unit size constraints. Note that this formulation utilizes storage tasks with dedicated storage units in order to store states between processing tasks. To model this example using a continuous-time formulation common to all units, we will employ the model of Maravelias and Grossmann[31] without the consideration of resource constraints.

Table 7. Model and Solution Statistics for the Motivating Example

	Lin and Floudas[36] formulation	Maravelias and Grossmann[31] formulation
event points	5	6
binary variables	25	48
continuous variables	129	296
constraints	217	584
LP relaxation	10.0	10.0
objective	10.0	10.0
nodes	2	34
CPU time (s)	0.01	0.04

The solution from the event-driven model is given in Figure 15, while the schedule from the continuous-time formulation common to all units can be seen in Figure 16. Note that the event-driven solution contains an extra storage unit that has a suitable storage task. This storage task allows state IB to be stored in the storage unit until task sep is ready to process it into the final product B. In this manner, we can limit the amount of intermediate stored, prevent it from being stored at all, or allow an unlimited amount to be stored in the storage tank. Also, note that the introduction of a storage task and a storage unit does not, in this case, cause the solution to require any more event points. The model and solution

statistics from both formulations can be found in Table 7.

It can be seen from the model and solution statistics in Table 7 that, for this example using the above two models, the continuous-time formulation common to all units requires more binary and continuous variables, more constraints, and many more nodes to solve. Most importantly, the solution requires an additional time point, which holds true for almost any problem modeled with the two different formulations. This is because the common-time representation requires a time point for the start and end of each task, whereas event-driven formulations only require a time point for the start of a task. This means there will be an extra time point to account for the ending of the last task in the common-time representation. Moreover, the number of time points is directly related to the computational complexity of a problem, as already discussed. Hence, the continuous-time formulation with timing common to all units will most likely always require more time points and take longer to solve. From this, we can deduce that unit-specific event-driven models, even when they take into account storage considerations, will require less time points and thus be less computationally demanding than a continuous-time formulation with timing common to all units.

Note that the above argument can also be extended to the case of resource constraints. The model presented in this work, including resource constraints and storage considerations, will usually require one less time point and be less computationally demanding than the continuous-time model presented by Maravelias and Grossmann.[31]

Appendix B: Data for Examples

See Tables 8−16.

Table 8. Data for Example 1

	F1	F2	I1	I2	I3	P1	P2
ST_s^{max} (kg)	1000	1000	200	100	500	1000	1000
ST_s^0 (kg)	400	400	0	0	0	0	0
$price_s$ ($/kg)	0	0	0	0	0	30	40

Table 9. Data for Example 1[a]

			T1		T2		T3		T4	
	cap^{min}	cap^{max}	α	β	α	β	α	β	α	β
R1	40	80	0.5	0.025	0.75	0.0375				
R2	25	50	0.5	0.04	0.75	0.06				
R3	40	80					0.25	0.0125	0.5	0.025

[a] cap^{min}/cap^{max} in kg, α in h, and β in h/kg.

Table 10. Data for Example 1[a]

	T1		T2		T3		T4	
	γ_{iHS}	δ_{iHS}	γ_{iCW}	δ_{iCW}	γ_{iHS}	δ_{iHS}	γ_{iCW}	δ_{iCW}
R1	6	0.25	4	0.3				
R2	4	0.25	3	0.3				
R3					8	0.2	4	0.5

[a] γ in kg/min and δ in kg/min per kg of batch.

Table 11. Data for Example 2

	F1	F2	S1	S2	S3	S4	S5	S6	INT1	INT2	P1	P2	P3
ST_s^{max} (kg)	∞	∞	0	0	15	40	0	0	∞	∞	∞	∞	∞
ST_s^0 (kg)	100	100	0	0	0	10	0	0	0	0	0	0	0
$price_s$ ($/kg)											1	1	1

Table 12. Data for Example 2[a]

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
unit	U1	U2	U3	U1	U4	U4	U5	U6	U5	U6
cap^{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] cap^{max} in kg, α in h, γ in kg/min, and δ in kg/min per kg of batch.

Table 13. Data for Example 3

	F1	F2	S11	S21	S12	S22	P1	P2
ST_s^{max} (tons)	100	100	5 in T1	5 in T1	3 in T2	3 in T2	100	100
ST_s^0 (tons)	100	100	0	0	0	0	0	0
$price_s$ (10^3 $/ton)							1	1

Table 14. Data for Example 3[a]

	T11	T21	T12	T22	T13	T23
unit	U1	U1	U2	U2	U1	U1
cap^{max}	5	5	3	3	5	5
cap^{min}	2	2	1.2	1.2	2	2
α	1	0.5	0.5	0.75	0.75	0.5
β	0.8	0.667	0.667	0.6		

[a] cap^{max}/cap^{min} in tons, α in h, and β in h/tons of batch.

Table 15. Setup Times for Example 3 (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

Table 16. Data for Example 4

		processing time (days)			
order	due date (days)	j1	j2	j3	j4
1	15	1.538			1.194
2	30	1.500			0.789
3	22	1.607			0.818
4	25			1.564	2.143
5	20			0.736	1.017
6	30	5.263			3.200
7	21	4.865		3.025	3.214
8	26			1.500	1.440
9	30			1.869	2.459
10	29		1.282		
11	30		3.750		3.000
12	21		6.796	7.000	5.600
transition		0.180	0.175	0.00	0.237

Nomenclature

Indices

i = processing tasks
i^{st} = storage tasks
j = units
k = orders
n = event points representing the beginning of a task
s = states
u = utilities

Sets

I = processing tasks
I_s^{st} = storage tasks for state s
I_j = tasks that can be performed in unit j
I_k = tasks that process order k
I^{ps} = tasks that are processing or storing
I_s^p = tasks that produce state s

I_s^c = tasks that consume state s
I_u = tasks that consume utility u
J = units
J_i = units that are suitable for performing task i
K = orders
K_i = orders that are processed by task i
K_s = orders that produce state s
N = event points within the time horizon
S = states
S^f = states with finite intermediate storage
S^n = states with no intermediate storage
S^p = states that are final products
S^r = states that are raw materials
S^z = states with zero-wait constraint
U = utilities

Parameters

α_{ij} = constant term of processing time of task i in unit j
β_{ij} = variable term of processing time of task i in unit j
δ_{iu} = variable term of consumption of utility u by task i
γ_{iu} = constant term of consumption of utility u by task i
ρ_{is} = proportion of state s produced, consumed by task i
$\tau_{ii'}$ = sequence-dependent setup time between tasks i and i'
am_k = amount of order k
av_u = maximum availability of utility u
$\mathrm{cap}_{ij}^{\max}$ = maximum capacity for task i in unit j
$\mathrm{cap}_{ij}^{\min}$ = minimum capacity for task i in unit j
cap_s^{st} = capacity of storage for state s
dem_s = demand of state s
due_k = due date of order k
H = time horizon
price_s = price of state s
ST_s^0 = initial available amount of state s
$\mathrm{ST}_s^{\max}$ = maximum amount of state s

Continuous Variables

$B(i,j,n)$ = amount of material undertaking task i in unit j at event point n
$B^s(i,j,n)$ = amount of material starting processing at event point n
$B^f(i,j,n)$ = amount of material finishing processing at event point n
$\mathrm{BU}(i,u,n)$ = amount of utility u consumed by task i at event point n
$B_{ut}(u,n)$ = remaining level of utility u at event point n
$B_{st}(i^{st},n)$ = amount of material stored by storage task i^{st} at event point n
$D(s,n)$ = amount of state s delivered at event point n
MS = makespan
$\mathrm{ST}(s,n)$ = amount of state s at event point n
$\mathrm{STF}(s)$ = final amount of state s at the end of the time horizon
$\mathrm{STO}(s)$ = initial amount of state s at the beginning of the time horizon
$T^s(i,j,n)$ = time at which task i starts in unit j at event point n
$T^f(i,j,n)$ = time at which task i finishes in unit j at event point n
$T_{st}^s(i^{st},n)$ = time at which storage task i^{st} starts at event point n
$T_{st}^f(i^{st},n)$ = time at which storage task i^{st} finishes at event point n
$T_{ut}^s(u,n)$ = starting time of a change in utility u at event point n
$T_{ut}^f(u,n)$ = finishing time of a change in utility u at event point n
$\mathrm{tt}^s(j,n)$ = starting time of the active task in unit j at event point n
$\mathrm{tt}^f(j,n)$ = finishing time of the active task in unit j at event point n
$w(i,n)$ = task i is activated at event point n

Binary Variables

$\mathrm{ws}(i,n)$ = beginning of task i at event point n
$\mathrm{wf}(i,n)$ = ending of task i at event point n
$y(k,i,n)$ = delivery of order k through task i at event point n

Literature Cited

(1) Reklaitis, G. V. Overview of Scheduling and Planning of Batch Process Operations. Presented at NATO Advanced Study Institute—Batch Process Systems Engineering, Antalya, Turkey, 1992.

(2) Pantelides, C. C. Unified Frameworks for Optimal Process Planning and Scheduling. In *Proceedings of the Second International Conference on Foundations of Computer-Aided Process Operations*, Crested Butte, CO, 1993; Rippin, D. W. T., Hale, J. C., Davis, J., Eds.; CACHE: Austin, TX, 1993; pp 253−274.

(3) Floudas, C. A.; Lin, X. Continuous-Time versus Discrete-Time Approaches for Scheduling of Chemical Processes: A Review. *Comput. Chem. Eng.* **2004**, in press.

(4) Floudas, C. A.; Lin, X. Mixed Integer Linear Programming in Process Scheduling: Modeling, Algorithms, and Applications. *Ann. Oper. Res.* **2004**, accepted for publication.

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

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

(7) Pinto, J. M.; Grossmann, I. E. Optimal Cyclic Scheduling of Multistage Continuous Multiproduct Plants. *Comput. Chem. Eng.* **1994**, *18*, 797.

(8) Pinto, J. M.; Grossmann, I. E. A Continuous Time Mixed Integer Linear Programming Model for Short-Term Scheduling of Multistage Batch Plants. *Ind. Eng. Chem. Res.* **1995**, *34*, 3037.

(9) Pinto, J. M.; Grossmann, I. E. An Alternate MILP Model for Short-Term Scheduling of Batch Plants with Preordering Constraints. *Ind. Eng. Chem. Res.* **1996**, *35*, 338.

(10) Pinto, J. M.; Grossmann, I. E. A Logic-Based Approach to Scheduling Problems with Resource Constraints. *Comput. Chem. Eng.* **1997**, *21*, 801.

(11) Karimi, I. A.; McDonald, C. M. Planning and Scheduling of Parallel Semi-Continuous Processes. 2. Short-Term Scheduling. *Ind. Eng. Chem. Res.* **1997**, *36*, 2701.

(12) Lamba, N.; Karimi, I. A. Scheduling Parallel Production Lines with Resource Constraints. 1. Model Formulation. *Ind. Eng. Chem. Res.* **2002**, *41*, 779.

(13) Lamba, N.; Karimi, I. A. Scheduling Parallel Production Lines with Resource Constraints. 2. Decomposition Algorithm. *Ind. Eng. Chem. Res.* **2002**, *41*, 790.

(14) Cerdá, J.; Henning, G. P.; Grossmann, I. E. A Mixed-Integer Linear Programming Model for Short-Term Scheduling of Single-Stage Multiproduct Batch Plants with Parallel Lines. *Ind. Eng. Chem. Res.* **1997**, *36*, 1695.

(15) Méndez, C. A.; Cerdá, J. Optimal Scheduling of a Resource-Constrained Multiproduct Batch Plant Supplying Intermediates to Nearby End-Product Facilities. *Comput. Chem. Eng.* **2000**, *24*, 369.

(16) Méndez, C. A.; Cerdá, J. An MILP Continuous-Time Framework for Short-Term Scheduling of Multipurpose Batch Processes under Different Operation Strategies. *Optim. Eng.* **2003**, *4*, 7.

(17) Méndez, C. A.; Henning, G. P.; Cerdá, J. An MILP Continuous-Time Approach to Short-Term Scheduling of Resource-Constrained Multistage Flowshop Batch Facilities. *Comput. Chem. Eng.* **2001**, *25*, 701.

(18) Méndez, C. A.; Henning, G. P.; Cerdá, J. Short-Term Scheduling of Multiproduct Batch Plants under Limited Resource Capacity. *Lat. Am. Appl. Res.* **2001**, *31*, 455.

(19) Lee, K.; Heo, S.; Lee, H.; Lee, I. Scheduling of Single-Stage and Continuous Processes on Parallel Lines with Intermediate Due Dates. *Ind. Eng. Chem. Res.* **2002**, *41*, 58.

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

(21) Zhang, X.; Sargent, R. W. H. The Optimal Operation of Mixed Production Facilities—Extensions and Improvements. *Comput. Chem. Eng.* **1998**, *22*, 1287.

(22) Mockus, L.; Reklaitis, G. V. Mathematical Programming Formulation for Scheduling of Batch Operations Based on Nonuniform Time Discretization. *Comput. Chem. Eng.* **1997**, *21*, 1147.

(23) 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.

(24) Mockus, L.; Reklaitis, G. V. Continuous Time Representation Approach to Batch and Continuous Process Scheduling. 2. Computational Issues. *Ind. Eng. Chem. Res.* **1999**, *38*, 204.

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

(26) Castro, P.; Barbosa-Póvoa, 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.

(27) Castro, P.; Barbosa-Póvoa, A. P. F. D.; Matos, H. Optimal Periodic Scheduling of Batch Plants using RTN-based Discrete and Continuous-Time Formulations: A Case Study Approach. *Ind. Eng. Chem. Res.* **2003**, *42*, 3346.

(28) Majozi, T.; Zhu, X. X. A Novel Continuous-Time MILP Formulation for Multipurpose Batch Plants. 1. Short-Term Scheduling. *Ind. Eng. Chem. Res.* **2001**, *40*, 5935.

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

(30) Wang, S.; Guignard, M. Redefining Event Variables for Efficient Modeling of Continuous-Time Batch Processing. *Ann. Oper. Res.* **2002**, *116*, 113.

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

(32) 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.

(33) Ierapetritou, M. G.; Floudas, C. A. Effective Continuous-Time Formulation for Short-Term Scheduling: 2. Continuous and Semi-continuous Processes. *Ind. Eng. Chem. Res.* **1998**, *37*, 4360.

(34) 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.

(35) Ierapetritou, M. G.; Hené, T. S.; Floudas, C. A. Effective Continuous-Time Formulation for Short-Term Scheduling: 3. Multiple Intermediate Due Dates. *Ind. Eng. Chem. Res.* **1999**, *38*, 3446.

(36) Lin, X.; Floudas, C. A. Design, Synthesis and Scheduling of Multipurpose Batch Plants via an Effective Continuous-Time Formulation. *Comput. Chem. Eng.* **2001**, *25*, 665.

(37) Glover, F. Improved Linear Integer Programming Formulations of Nonlinear Integer Problems. *Manage. Sci.* **1975**, *22*, 455.

(38) Floudas, C. A. *Nonlinear and Mixed-Integer Optimization*; Oxford University Press: Oxford, U.K., 1995.

(39) Brooke, A.; Kendrick, D.; Meeraus, A.; Raman, R. *GAMS: A User's Guide*; South San Francisco, CA, 2003.

Received for review October 1, 2003
Revised manuscript received February 13, 2004
Accepted March 3, 2004

IE0341597