Effective Continuous-Time Formulation for Short-Term Scheduling. 3. Multiple Intermediate Due Dates[1,2]

M. G. Ierapetritou,† T. S. Hené, and C. A. Floudas*

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

The problem of short-term scheduling often involves the satisfaction of variable product demands at specific due dates within the time horizon under consideration. Ierapetritou and Floudas (*Ind. Eng. Chem. Res.* **1998**, *37*, 4341−4359, 4360−4374) presented a new continuous time formulation to effectively address the problem of short-term scheduling in batch, continuous, and mixed production facilities where product demands are specified at the end of the time horizon. The primary objective of this paper is to extend the continuous-time formulation so as to deal with intermediate due dates. The mathematical model is developed and the operation mode of the plant (batch or semicontinuous) is further exploited to result in the most efficient solution strategy. Several examples are provided to illustrate the capabilities of the proposed continuous-time formulations, and it is demonstrated that a variety of problems presented in the literature can be addressed efficiently.

1. Introduction

A significant body of research work appeared in the literature to deal with the problem of short-term scheduling, especially in batch plants. In this work, we will refer to those papers that have intermediate due dates for both batch and semicontinuous plants. For batch plants, Sahinidis and Grossmann[3] proposed a reformulation of the multiperiod mixed integer linear programming (MILP) model based on the lot-sizing substructures to improve the solution efficiency. Their reformulation performed better than earlier MILP formulations since it provided a tighter LP relaxation.

Mockus and Reklaitis[4] provided a new approach to solve short-term scheduling problems for batch plants. The problem is formulated as a nonconvex mixed integer nonlinear programming (MINLP) problem, and a global optimization algorithm is used for its solution. Nonlinearities occurred in the material balances (products of binary and continuous variables.) Solutions were reported for the proposed branch and bound method as well as a heuristic approach. On the basis of the computational results reported, the algorithmic performance is problem-dependent.

Pinto and Grossmann[5] proposed a formulation based on the definition of the binary variables over all units, orders, time slots, and production stages. In their formulation, a production stage consists of a set of similar processes, and all batches have to be processed exactly once in all of the production stages. A number of time slots is defined for each unit based on the number of orders it is able to handle and in the case of large problems the number of parallel units. They reported computational results on large-scale problems involving 29 different orders in which resource constraints are not considered.

For semicontinuous plants, Karimi and McDonald[6] proposed two mathematical models for the short-term scheduling problem involving intermediate due dates that differ on the preassignment of slots to time periods. The proposed formulation can handle the problem of a single-stage multiproduct facility with parallel semicontinuous processors. The model complexity requires the preassignment of slot to time periods, and problem decomposition to address medium-to-large-size problems. The proposed formulation is very much case dependent and tailored to address the specific problems presented. The preassignment of slots and the prepostulation of the minimum number of slots are needed for the solution of the presented examples.

In this paper, a novel continuous-time formulation is proposed to address the requirement of multiple intermediate due dates. In section 2, a review of the basic concepts of the approach of Ierapetritou and Floudas[1,2] is provided. In section 3, two models are proposed to address the problem of short-term scheduling of batch plants with intermediate due dates. The first model facilitates the solution of the problem where no resource constraints are considered and the second model considers the overall short-term scheduling problem of batch plants including resource constraints. The rationale for considering two different approaches is based on the ability to exploit the nature of the problems by avoiding the introduction of unnecessary variable and constraints. Examples are presented to illustrate the applicability of the proposed approaches and their advantages with the already existing formulations. Section 5 then considers the problem of short-term scheduling with intermediate due dates in semicontinuous plants. The proposed approach satisfies all the requirements suggested in the literature, Karimi and McDonald.[6] Computational results on a recently published industrial case study are presented.

2. Basic Concepts of Continuous-Time Formulation

The proposed formulation for short-term scheduling of batch and semicontinuous plants with multiple due

† Current address: Department of Chemical and Biochemical Engineering, Rutgers University, 98 Brett Rd., Piscataway NJ 08854.

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

10.1021/ie990108r CCC: $18.00
Published on Web 08/07/1999

dates is based on the novel concepts of the framework presented in Ierapetritou and Floudas.[1,2] In this section a brief review of the basic ideas of the continuous-time formulation is proposed to facilitate the presentation of the proposed models for multiple intermediate due dates. More specifically, the proposed approach does the following:

(a) follows a *continuous time* representation in which the event points, where a task starts being processed or/and a unit starts operation, are unknown and constitute variables to the optimization problem.
(b) uses different binary variables for *task events*, $wv(i,n)$, that represent the start of task (i) at event point (n), and different binary variables for *unit events*, $yv(j,n)$, that correspond to the beginning of unit (j) utilization at event point (n).
(c) allows for variable processing times with respect to the amount of material processed by the specific task.

Note that the most important point is the consideration of two different sets of variables to represent the assignment of tasks and units to event points which results in decoupling of tasks from units. This results in a major reduction of binary variables, but the modeling of scheduling characteristics given the decoupling set of variables needs to be addressed. As is shown in Ierapetritou and Floudas,[1,2] this goal is achieved by a set of allocation constraints, material balances, capacity and demand constraints, duration constraints, and time sequence constraints. More specifically, the allocation constraints offer the desired connection between units and tasks, whereas the material balances together with the capacity and demand constraints guarantee the production/consumption of the correct amount of materials and demand satisfaction at the end of the time horizon. Finally, a set of timing constraints involving duration and sequence constraints enforce the correct duration and sequence on tasks throughout the time horizon under consideration. As demonstrated in Ierapetritou and Floudas,[1,2] the proposed formulation results in much smaller models in terms not only of the binary variables but also of the continuous variables, and constraints and the decrease of the integrality gap that enable the more efficient solution of the short-term scheduling problems and the consideration of large case studies. In the next section, a novel formulation is proposed for the incorporation of multiple due dates requirements in the short-term scheduling of batch plants based on the aforementioned basic concepts.

3. Short-Term Scheduling of Batch Plants

3.1. No Resource Constraints. The proposed mathematical formulation for the short-term scheduling of a batch plant with multiple intermediate due dates involves the following characteristics:

(a) *Preordering*: Orders are put in sequence a priori and linked to nodes as proposed by Pinto and Grossmann,[5]
(b) *Binary Variables*: The binary variables $wv(i,n)$ employed in Ierapetritou and Floudas[1,2] become parameters since each order (i) is linked to a particular event point (n), for which $wv(i,n)$ is set to 1. The $wv(i,n)$ are set to 0 for all other combinations. Consequently, $yv(j,n)$ are the only binary variables.
(c) *Elimination of Material Balance*: The material balances are eliminated from the proposed formulation,[1,2] since no resource constraints are considered.
(d) *Elimination of States*: The states are also eliminated from the formulation proposed by Ierapetritou and Floudas[1,2] since they are superfluous due to the absence of resource constraints. Tasks are instead directly related to their due dates.
(e) *Due Dates*: Due dates are set to the end of the time intervals, since tasks performed in interval (n) are ready at the end of this node.

3.1.1. Single-Stage Mathematical Model. Using the aforementioned characteristics that are particular to the case of no resource constraints, we introduce the following indices, sets, parameters, and variables:

Notation

Indices

i, i' = orders
j = units
n = event points
n_i, $n_{i'}$ = event points assigned to orders (i), (i'), respectively

Sets

I = orders
I_j = tasks which can be performed in unit (j)
J = units
J_i = units which are suitable for performing task (i)
N = event points within the time horizon
N_i = event points assigned to order (i)

Parameters

α_{ij} = constant term of processing time of task (i) at unit (j)
β_j = constant term of setup time in unit (j)
H = time horizon
$wv(i,n)$ = parameters that assign the beginning of task (i) at event point (n)
$due(i)$ = due date of order (i)
$wt(i)$ = relative importance of order (i) in the objective function

Variables

$yv(j,n)$ = binary variables that assign the utilization of unit (j) at event point (n)
$T^s(i,j,n)$ = time that task (i) starts in unit (j) at event point (n)
$T^f(i,j,n)$ = time that task (i) finishes in unit (j) while it starts at event point (n)

On the basis of this notation, the mathematical model for the short-term scheduling of a single-stage batch plant without considering resource constraints involves the following constraints.

1. Allocation Constraints.

$$\sum_{j \in J_i} yv(j,n) = 1, \qquad \forall\ i \in I, \quad n \in N_i \tag{1}$$

These constraints express that an order only has to be processed once, since only one unit can be assigned to it.

2. Duration Constraints.

$$T^f(i,j,n) = T^s(i,j,n) + yv(j,n)(\alpha_{ij} + \beta_j), \qquad \forall\ i \in I_j, \quad j \in J_i, \quad n \in N_i \tag{2}$$

These constraints ensure that the processing time of an order (i) in a particular machine (j) is the sum of the setup time characteristic in the unit, (β_j), and the processing time of this order in this particular unit, (α_{ij}).

Table 1. Data, Single Stage

		processing time (days)			
order	due date (day)	1	2	3	4
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
13	30	11.250			6.716
14	25	2.632			1.527
15	24	5.000			2.985
16	30	1.250			0.783
17	30	4.474			3.036
18	30		1.492		
19	13		3.130		2.687
20	19	2.424		1.074	1.600
21	30	7.317		3.614	
22	20			0.864	
23	12			3.624	
24	30			2.667	4.000
25	17	5.952		3.448	4.902
26	20	3.824			1.757
27	11	6.410			3.937
28	30	5.500			3.235
29	25				4.286
transition		0.180	0.175	0.000	0.237

3. Sequence Constraints: Different Tasks in the Same Unit.

$$T^{s}(i',j,n_{i'}) \geq T^{f}(i,j,n_i), \qquad \forall\ i \in I_j,\quad i' \in I_j,\quad j \in J_i,\ n_{i'} > n_i,\quad i \neq i' \tag{3}$$

These constraints express the requirement that only one order at a time can be processed in a unit; the beginning of a later event must follow after the end of all earlier events in the same unit.

4. Due Dates Constraints.

$$T^{f}(i,j,n_i) \leq \mathrm{due}(i), \qquad \forall\ i \in I,\quad \forall\ j \in J_i \tag{4}$$

where n_i corresponds to the assignment of node (n) to order (i). These constraints ensure that the order would be ready at its due date, due(i).

5. Objective.

$$\text{Maximize} \sum_i \mathrm{wt}(i) \sum_{j \in J_i, n \in N_i} [T^{s}(i,j,n) + \mathrm{yv}(j,n) \times (\alpha_{ij} + \beta_j)] \tag{5}$$

The objective function represents the maximization of the weighted starting times for orders in all stages and determines the schedule that satisfies the due dates in a most efficient way.

3.1.2. Illustration. In this section an example from Pinto and Grossmann[5] is considered to illustrate the potential of the proposed approach. From the data given in the problem (see Table 1), it can be noted that there is a very large number of orders that need to be scheduled on very few units. Units 1 and 4 can handle many orders, while units 2 and 3 can only process a few. Preordering can indeed be applied so that orders could be assigned to nodes, and the combination of the node and order could be assigned an upper time limit.

Table 2. Single-Stage Preordering

due date (day)	orders due
11	27
12	23
13	19
15	1
17	25
19	20
20	5 22 26
21	7 12
22	3
24	15
25	4 14 29
26	8
29	10
30	2 6 9 11 13 16 17 18 21 24 28

Table 3. Number Assignment

order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
number	4	28	12	15	9	23	11	17	27	18	25	10	19	16	13
order	16	17	18	19	20	21	22	23	24	25	26	27	28	29	
number	29	24	26	3	6	20	8	2	21	5	7	1	22	14	

The coupling of orders to time slots was performed according to the following rules, as suggested by Pinto and Grossmann:[5]

(1) increasing due dates
(2) decreasing processing times (if ties occur)

Preordering was performed simultaneously for all units. This is done because the assignment of nodes had to take place such that, in each unit, an order is coupled *to the same node number*. To avoid conflicts, the processing times of each order should have the same relative order in each machine. If conflicts do arise, the unit with the least possible orders should govern the node assignment decisions because there is a greater chance that this unit will handle the order.

First, preordering is performed based only on the due dates of the orders. This gives the results shown in Table 2. Then, the order in which each unit should handle orders with the same due date is determined. The processing times of orders in units are compared, but only if two orders are processed in the same unit. For example, orders 1, 6, and 17 can both be processed in both units 1 and 4. Order 16 takes 1.250 days in unit 1 and 0.783 days in unit 4. Order 17 takes 4.474 days in unit 1 and 3.036 days in unit 4. Therefore, in both units order 16 requires more time and should therefore be processed after order 17.

Note that, in the above preordering procedure, the following assumption is made. To keep the node assignment consistent, the sequencing of orders has to be the same in each unit; that is, the processing times must be of the same relative size in all units. For example, if order A takes longer to process than order B in unit 1, one can assume that the processing time in unit 2 is also going to be longer than the processing time of unit B in unit 2.

The orders are numbered 1−29, in ascending order, according to aforementioned sequencing. This number is then assigned to each order as its node number. The results are shown in Table 3.

On the basis of the above preordering, the formulation presented in section 3.1 is applied for the cases where 8 orders and all 29 orders are considered. The results of the proposed approach are presented in Tables 4 and 5 and compared with the results of the formulation proposed by Pinto and Grossmann.[5] Note that the proposed formulation results in smaller models in terms

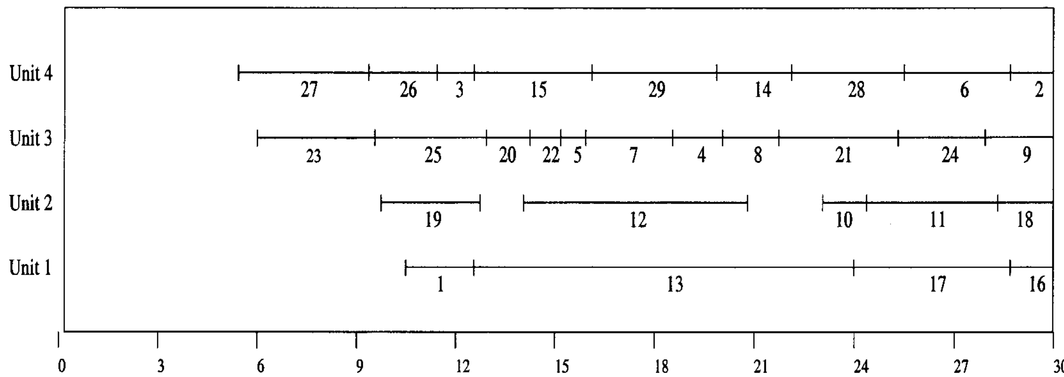

Figure 1. Gantt chart for a preordered single-stage problem.

Table 4. Single-Stage Results with Preordering—8 Orders

	Pinto and Grossmann	Proposed Approach
binary vars	51	17
continuous vars	120	52
constraints	267	70
CPU time	2.17[a]	0.040
nodes	47	3

[a] HP 9000-730 workstation.

Table 5. Single-Stage Results with Preordering—29 Orders

	Pinto and Grossmann	proposed approach
binary vars	441	57
continuous vars	875	172
constraints	1791	559
CPU time	1257.17[a]	0.280
nodes	204	5

[a] HP 9000-730 workstation.

of binary and continuous variables and constraints. In particular, for the large example involving 29 orders, the proposed formulation requires 57 and 172 binary and continuous variables, respectively, in comparison with 441 and 875 binary and continuous variables required by Pinto and Grossmann[5] formulation. In addition, the proposed formulation requires 559 constraints instead of 1791 used by Pinto and Grossmann.[5] Moreover, the resulting model is more easily solved, requiring the exploitation of only 5 nodes in the branch-and-bound tree and the need of 0.28 CPU s using GAMS/CPLEX on a HP C160 workstation.

Figure 1 depicts the Gantt chart of the optimal solution. The numbers below the horizontal lines indicate the order number being processed at that time.

3.2. Consideration of Resource Constraints. In this section, the overall short-term scheduling problem is considered, including resource constraints. In the sets, indices, and parameters used in section 3.1, the following additions are made.

Notation (cont.)

Indices

s = states

n_s = event points assigned to due dates of state (s)

Sets

S = states

I_s = tasks which produce/consume state (s)

N_s = event points assigned to state (s)

Parameters

due(s) = due date of state (s)

ρ^{p}_{si}, ρ^{c}_{si} = proportion of state (s) produced, consumed from task (i), respectively

$R(s,n)$ = amount of state (s) required at event point (n)

Variables

wv(i,n) = binary variables that assign the beginning of task (i) at event point (n)

$B(i,j,n)$ = amount of material undertaking task (i) in unit (j) at event point (n)

st(s,n) = amount of state (s) at event point (n)

st0(s,n) = initial amount of state (s)

Note that the explicit consideration of resource constraints requires that we treat the wv(i,n) as binary variables and not as parameters. Also, for the introduction of material balances, we need to introduce the states (s) and their amounts at event point (n), $R(s,n)$. Following the same notation as in section 3.1, the mathematical model takes the following form.

1. Allocation Constraints.

$$\sum_{i\in I_j} \mathrm{wv}(i,n) = \mathrm{yv}(j,n), \quad \forall\, j \in J,\, n \in N \tag{6}$$

These constraints ensure that only one task (i) can take place at unit (j) at each event point (n).

2. Capacity Constraints.

$$V^{\min}_{ij}\mathrm{wv}(i,n) \le B(i,j,n) \le V^{\max}_{ij}\mathrm{wv}(i,n), \quad \forall\, i \in I,\, j \in J_i,\, n \in N \tag{7}$$

These constraints express the minimum and maximum capability of unit (j) when processing task (i), enforced at every event point (n).

3. Material Balance.

$$d(s,n) = \sum_{i\in I_s}\rho^{p}_{si}\sum_{j\in J_i}B(i,j,n-1) + \sum_{i\in I_s}\rho^{c}_{si}\sum_{j\in J_i}B(i,j,n), \quad \forall\, s \in S,\, n \in N \tag{8}$$

According to these constraints, the amount of material of state (s) delivered at event point (n), $d(s,n)$, is equal to the amount produced between the event points (n − 1) and (n), denoted as $B(i,j,n-1)$, minus the amount consumed between event points (n) and (n + 1), denoted as $B(i,j,n)$.

4. Demand Constraints.

$$d(s,n) = R(s,n), \quad n \in N, s \in S$$

These constraints ensure the satisfaction of product demand at event point (n), $R(s,n)$. Note that although the above formulation considers the demand as a hard constraint, partial demand satisfaction can be easily incorporated as shown for the semicontinuous plants in section 4.

Linking Demands to Event Points. Demands have to be linked to particular event points, on the basis of the relative time at which the demand has to be filled, the number of stages required to get to the final product, and the number of other tasks that may take place in the same unit.

An important issue to note is that because of the nature of the batch operation mode, the amount of state (s) produced in event point (n) cannot be consumed until the beginning of event point ($n + 1$). Therefore, even the most basic problem (one task with a single demand) requires two event points. In the first event point, a task produces the product, and at the beginning of the second event point, it can meet the demand.

Also, since the production time depends on the number of stages involved, the number of event points should also be stage-dependent, which means that if a task occurs in the second or higher stage, then additional event points need to be considered (one for stage two, two for stage three, etc).

Due Dates Constraints. Following the aforementioned linking of demands to event points, the following set of time constraints ensure the satisfaction of product demand by the corresponding due date,

$$T^{s}(i,j,n_s) \leq \text{due}(s), \quad \forall\, s \in S,\, i \in I_s,\, j \in J_i \tag{9}$$

where $n_s \in N_s$ represent the subset of nodes linked to the due dates of state (s). Since the demand has to be met at the beginning of a particular node, the due date time limitations are incorporated as upper bounds on the starting times of the tasks producing state (s).

Duration Constraints.

$$T^{f}(i,j,n) = T^{s}(i,j,n) + \alpha_{ij} w(i,n) + \beta_{ij} B(i,j,n), \quad \forall\, i \in I_s,\, j \in J_i \cap J_s,\, n \in N \tag{10}$$

where α_{ij} and β_{ij} are the constant and variable term of the processing time of task (i) at unit (j).

Sequence Constraints. Since ordering had to be performed based on the due dates, intermediate event points may be left idle. Therefore, to ensure the correct sequencing, the sequence constraints take the following form, depending on whether we have the same or different units:

a. Same Unit.

$$T^{s}(i,j,n+1) \geq T^{f}(i,j,n), \quad \forall\, i \in I,\, j \in J_i,\, n \in N \tag{11}$$

b. Different Units.

$$T^{s}(i,j,n+1) \geq T^{f}(i',j',n), \quad \forall\, i \in I_j,\, i' \in I_{j'},\, j \in J,\, j' \in J,\, n \in N,\, i \neq i' \tag{12}$$

In this way, even unassigned nodes will have to follow the sequence constraint. If for example, node 2 is not assigned to a task, then $wv(i,n2) = 0$, $\forall\, i \in I$ and consequently from duration constraints its starting and end time will coincide:

$$T^{s}(i,j,n2) = T^{f}(i,j,n2), \quad \forall\, i \in I,\, j \in J_i$$

Moreover, constraints (11) and (12) for event points 2 and 3 take the forms

$$T^{s}(i,j,n2) \geq T^{f}(i,j,n1), \quad \forall\, i \in I,\, j \in J_i$$

$$T^{s}(i,j,n2) \geq T^{f}(i',j',n1), \quad \forall\, i, i' \in I,\, j \in J_i,\, j' \in J_{i'}$$

$$T^{s}(i,j,n3) \geq T^{f}(i,j,n2), \quad \forall\, i \in I,\, j \in J_i$$

$$T^{s}(i,j,n3) \geq T^{f}(i',j',n2), \quad \forall\, i, i' \in I,\, j \in J_i,\, j' \in J_{i'}$$

and consequently

$$T^{s}(i,j,n3) \geq T^{f}(i,j,n1), \quad \forall\, i \in I,\, j \in j_i$$

$$T^{s}(i,j,n3) \geq T^{f}(i',j',n1), \quad \forall\, i, i' \in I,\, j \in J_i,\, j' \in J_{i'}$$

Thus, event point 3 is constrained to start after the end of node 1 since node 2 is unassigned.

Sequence Constraints: Completion of Previous Tasks.

$$T^{s}(i,j,n+1) \geq \sum_{n' \in N, n' \leq n} \sum_{i' \in I_j} (T^{f}(i',j,n') - T^{s}(i',j,n')), \quad \forall\, i \in I,\, j \in J_i,\, n \in N,\, n \neq N \tag{13}$$

These constraints represent the requirement of a task (i) to start after the completion of all the tasks performed in past event points at the same unit (j).

Timing Constraints for Storage Tasks. Storage tasks are treated as additional tasks, which at any event point can consume or produce the corresponding state which means that the values of ρ^{c}_{si} and ρ^{p}_{si} are set to -1 and 1 for the consumption and production task, respectively. The following constraints are then added for the timing of the storage tasks.

$$T^{f}(i,j,n) = T^{s}(i,j,n) + \tau(i,j,n), \quad \forall\, i \in I_s,\, j \in J_s \cap J_i,\, n \in N \tag{14}$$

$$Dur(i,j) - H^{*}(1 - wv(i,n)) \leq \tau(i,j,n) \leq Dur(i,j), \quad \forall\, i \in I_s,\, j \in J_s \cap J_i,\, n \in N \tag{15}$$

$$\tau(i,j,n) \leq H^{*} wv(i,n), \quad \forall\, i \in I_s,\, j \in J_s \cap J_i,\, n \in N \tag{16}$$

where $\tau(i,j,n)$ is the variable duration of the storage task (i) in a unit (j), which is a product of the binary variable $wv(i,n)$, that indicates if the storage unit is used, and the continuous variable $Dur(i,j)$ which represents the storage time. The last two sets of constraints correspond to the Glover transformation constraints[7] used to linearize the product of a continuous and a binary variable, (*i.e.*, $wv(i,n)Dur(i,j)$).

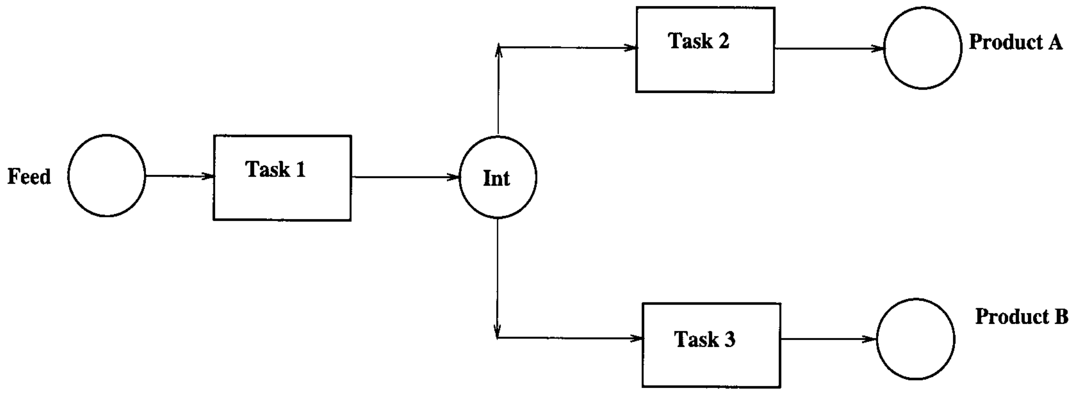

Figure 2. State task network for batch 1.

Table 6. Data, Batch 1

Units, Tasks			
units	size	units suitability	processing times
unit 1	1500	task 1	1
unit 2	1000	task 2	1
unit 3	1000	task 3	1

states		
states	capacity limits	prices
state 1 (feed)	unlimited	5
state 2 (intermediate	5000	
state 3 (product A)	unlimited	10
state 4 (product B)	unlimited	8

demands						
	time					
products	4	6	7	10	11	12
A	200		300	400	100	
B	50	150		200		100

cost data		
$\mathrm{costv}_{ijn} = 0.6$	$\mathrm{costc}_{ijn} = 200$	$\mathrm{costst}_{sn} = 0.18$

Objective Function. The objective function is the minimization of the operating cost:

Minimize

$$\sum_i \sum_j \sum_n (\mathrm{costv}_{ijn} b(i,j,n) + \sum_j \sum_n \mathrm{costc}_{jn} \mathrm{yv}(j,n)) + \sum_s \sum_n \mathrm{costst}_{sn} \mathrm{st}(s,n) + \sum_s \mathrm{cost}(s)\mathrm{st0}(s) - \sum_s \sum_n \mathrm{price}(s) d(s,n) \quad (17)$$

where costv_{ijn} is the batch-size-dependent cost of processing task (i) in unit (j) at event point (n), costc_{jn} is the setup cost (batch-size-independent) of unit (j) at event point (n), and costst_{sn} is the storage cost of state (s) at event point (n).

3.2.1. Computational Studies. The proposed formulation was applied to three examples, batch 1, 2, and 4, from Sahinidis and Grossmann[3] and Mockus and Reklaitis.[4] To clarify the proposed formulation, batch 1 is presented in detail while only the results for batch 2 and 4 are provided. All examples were solved using GAMS/CPLEX 4.0.8 on a HP C160 workstation. The increase of the computational complexity with respect to the time horizon depend on the specific problem since the proposed formulation is based on the continuous-time representation.

1. Example 1: Batch 1. In this example raw material, (s_1), is transformed into intermediate (s_2) in unit (j_1) where task (i_1) takes place. Intermediate (s_2) can then be transformed into product A (s_3) in unit j_2 by task (i_2), or product B (s_4) by task (i_3) in unit (j_3). The state task network representation for this example is shown in Figure 2, while the data for the problem is given in Table 6. For both products A and B, four intermediate due dates are provided where different product amounts are required.

The mathematical formulation for this example involves the following sets and constraints:

$$J = \{j_1, j_2, j_3\}$$

$$I = \{i_1, i_2, i_3, i_4\}$$

$$S = \{s_1, s_2, s_3, s_4\}$$

$$N = \{n_1, n_2, ..., n_7\}$$

Allocation Constraints.

$$\text{Unit 1:}\ \mathrm{wv}(i_1,n) = \mathrm{yv}(j_1,n), \quad \forall\, n \in N$$

$$\text{Unit 2:}\ \mathrm{wv}(i_2,n) = \mathrm{yv}(j_2,n), \quad \forall\, n \in N$$

$$\text{Unit 3:}\ \mathrm{wv}(i_3,n) = \mathrm{yv}(j_3,n), \quad \forall\, n \in N$$

Capacity Constraints. The maximum capacities of units 1, 2, and 3 are 1500, 1000, and 1000, respectively.

$$\text{Unit 1:}\ 0 \le B(i_1,j_1,n) \le 1500\mathrm{yv}(j_1,n), \quad \forall\, n \in N$$

$$\text{Unit 2:}\ 0 \le B(i_2,j_2,n) \le 1000\mathrm{yv}(j_2,n), \quad \forall\, n \in N$$

$$\text{Unit 3:}\ 0 \le B(i_3,j_3,n) \le 1000\mathrm{yv}(j_3,n), \quad \forall\, n \in N$$

Storage Constraints. The raw material and final products are not limited by storage capacities; only the intermediate (state 2) has a 5000-unit limit.

$$\mathrm{ST}(s_2,n) \le 5000, \quad \forall\, n \in N$$

Material Balances.

$$\text{State 1: } \mathrm{ST}(s_1,n) = \mathrm{ST}(s_1,n-1) - B(i_1,j_1,n-1), \quad \forall\, n \in N$$

$$\text{State 2: } \mathrm{ST}(s_2,n) = \mathrm{ST}(s_2,n-1) + B(i_1,j_1,n-1) - B(i_2,j_2,n), \quad \forall\, n \in N$$

$$\text{State 3: } \mathrm{ST}(s_3,n) = \mathrm{ST}(s_3,n-1) + B(i_2,j_2,n-1) - d(s_3,n), \quad \forall\, n \in N$$

$$\text{State 4: } \mathrm{ST}(s_4,n) = \mathrm{ST}(s_4,n-1) + B(i_3,j_3,n-1) - d(s_4,n), \quad \forall\, n \in N$$

Duration Constraints. For all processes, the fixed processing time is 1 time unit, while there is no variable component to the processing time ($\alpha = 1$ and $\beta = 0$).

$$\text{Unit 1: } T^{\mathrm{f}}(i_1,j_1,n) = T^{\mathrm{s}}(i_1,j_1,n) + \mathrm{yv}(j_1,n), \quad \forall\, n \in N$$

$$\text{Unit 2: } T^{\mathrm{f}}(i_2,j_2,n) = T^{\mathrm{s}}(i_2,j_2,n) + \mathrm{yv}(j_2,n), \quad \forall\, n \in N$$

$$\text{Unit 3: } T^{\mathrm{f}}(i_3,j_3,n) = T^{\mathrm{s}}(i_3,j_3,n) + \mathrm{yv}(j_3,n), \quad \forall\, n \in N$$

Sequence Constraints: Same Task in the Same Unit.

$$\text{Unit 1: } T^{\mathrm{s}}(i_1,j_1,n+1) \geq T^{\mathrm{f}}(i_1,j_1,n), \quad \forall\, n \in N$$

$$\text{Unit 2: } T^{\mathrm{s}}(i_2,j_2,n+1) \geq T^{\mathrm{f}}(i_2,j_2,n), \quad \forall\, n \in N$$

$$\text{Unit 3: } T^{\mathrm{s}}(i_3,j_3,n+1) \geq T^{\mathrm{f}}(i_3,j_3,n), \quad \forall\, n \in N$$

Sequence Constraints: Different Tasks in Different Units.

$$T^{\mathrm{s}}(i_2,j_2,n+1) \geq T^{\mathrm{f}}(i_1,j_1,n), \quad \forall\, n \in N$$

$$T^{\mathrm{s}}(i_3,j_3,n+1) \geq T^{\mathrm{f}}(i_1,j_1,n), \quad \forall\, n \in N$$

Demand Constraints. The demands of product 1 are linked to event points n_1, n_2, n_5, and n_6 in order of increasing due date. Similarly, the demands for product B were linked to event points n_2, n_3, n_5, and n_6. The demand parameters are then set up as

$$R(s_3,n_2) = 200$$
$$R(s_3,n_3) = 300$$
$$R(s_3,n_5) = 400$$
$$R(s_3,n_6) = 100$$

$$R(s_4,n_2) = 50$$
$$R(s_4,n_3) = 150$$
$$R(s_4,n_5) = 200$$
$$R(s_4,n_6) = 100$$

Consequently, the demand constraints for products A (s_3) and B (s_4) are

State 3:

$$d(s_3,n_2) = 200$$
$$d(s_3,n_3) = 300$$
$$d(s_3,n_5) = 400$$
$$d(s_3,n_6) = 100$$

State 4:

$$d(s_4,n_2) = 50$$
$$d(s_4,n_3) = 150$$
$$d(s_4,n_5) = 200$$
$$d(s_4,n_6) = 100$$

These demands are linked to all units that can perform the task that produces the state that is demanded. Since task (i_2) produces state (s_3) in unit (j_2) and task (i_3) produces state (s_4) in unit (j_3), the following bound constraints are incorporated:

$$T^{\mathrm{s}}_{\mathrm{up}}(i_2,j_2,n_2) = 4$$
$$T^{\mathrm{s}}_{\mathrm{up}}(i_2,j_2,n_3) = 7$$
$$T^{\mathrm{s}}_{\mathrm{up}}(i_2,j_2,n_5) = 10$$
$$T^{\mathrm{s}}_{\mathrm{up}}(i_2,j_2,n_6) = 11$$

$$T^{\mathrm{s}}_{\mathrm{up}}(i_3,j_3,n_2) = 4$$
$$T^{\mathrm{s}}_{\mathrm{up}}(i_3,j_3,n_3) = 6$$
$$T^{\mathrm{s}}_{\mathrm{up}}(i_3,j_3,n_5) = 10$$
$$T^{\mathrm{s}}_{\mathrm{up}}(i_3,j_3,n_6) = 12$$

Objective.

Minimize

$$0.6 \times \sum_{i\in I, j\in J_i, n\in N} b(i,j,n) + 200 \times \sum_{j\in J, n\in N} y(j,n) + 5 \times \sum_{n\in N} \mathrm{ST}(s_1,n) + 0.18 \times \sum_{s\in S n\in N} \mathrm{ST}(s,n) - 10 \times \sum_{n\in N} d(s_3,n) - 8 \times \sum_{n\in N} d(s_4,n)$$

The objective is the maximization of the profit formulated above as a minimization problem, where the first term corresponds to the operating cost, the second term to the cost of purchasing the raw materials, the third term to the inventory cost, and the last term to the revenue of product sales.

The proposed formulation involves 24 binary and 213 continuous variables and 148 constraints and requires 0.05 CPU s for its solution within 10^{-6} integrality tolerance using GAMS/CPLEX on a HP C160 worksta-

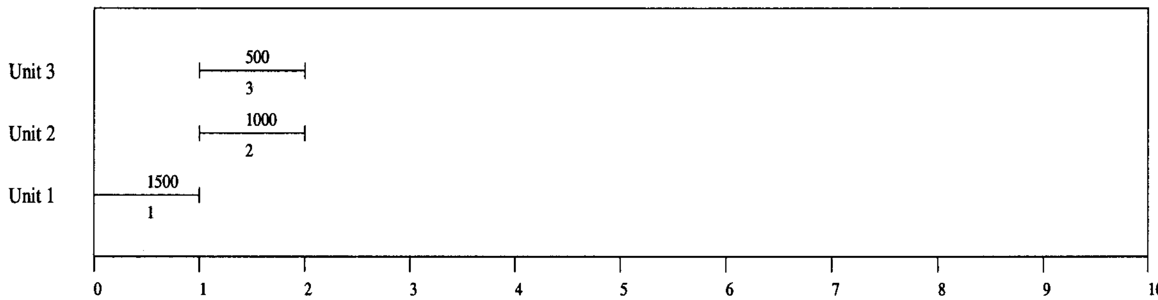

Figure 3. Gantt chart for batch 1.

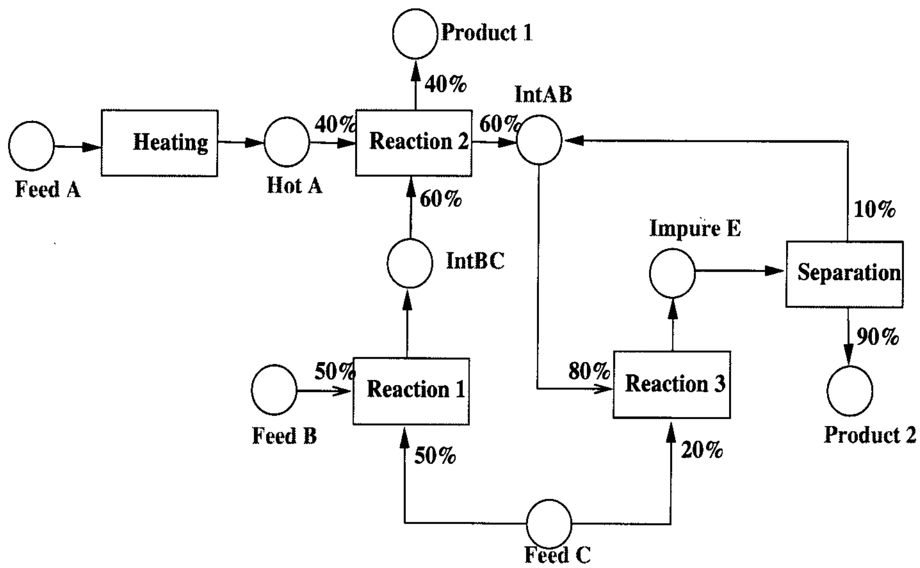

Figure 4. State task network for batch 2.

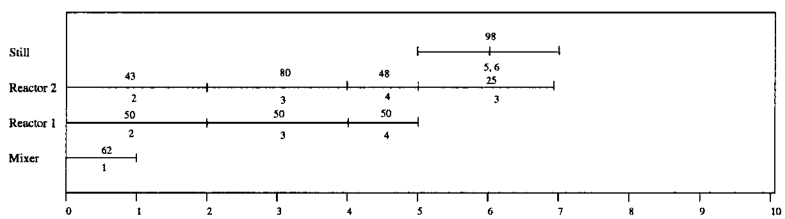

Figure 5. Gantt chart for batch 2.

Table 7. Results for Batch 1

	proposed formulation
constraints	148
variables	213
binary vars	24
integer optimum	3551
relaxation optimum	3651
integrality gap	0.025
no. nodes	0
CPU time (s)	0.05

tion. The integrality gap of the proposed formulation is 100 with 3651 being the objective of the LP relaxation problem. The resulting Gantt chart is shown in Figure 3 and the results are summarized in Table 7.

2. Example 2: Batch 2. The STN representation of the plant flowsheet is shown in Figure 4. The data for this example are presented in Table 8.

The resulting MILP formulation involves 441 constraints, 316 continuous variables, and 54 binary variables (Table 9). The solution of this problem with GAMS/CPLEX requires 0.15 CPU s in HP-C160. The optimal objective function corresponds to 5771 units. The corresponding Gantt chart is shown in Figure 5.

3. Example 3: Batch 4. The STN representation and data for batch 4 are shown in Figure 6 and Table 10, respectively. The proposed formulation gives rise to an MILP problem with 833 constraints, 798 continuous variables, and 64 binary variables. The solution of this problem with GAMS/CPLEX requires 0.27 CPU s in a HP-C160 workstation. The optimal objective function corresponds to 60297 units (Table 11). The corresponding Gantt chart is shown in Figure 7.

4. Short-Term Scheduling of Semicontinuous Plants

In this section, a new mathematical formulation is proposed for the short term scheduling of semi-continuous plants. This is based on the ideas presented in Ierapetritou and Floudas[2] and reviewed in section 2, which exploit further the nature of the semi-continuous processes involved in the plant.

4.1. Mathematical Model. The notation used in section 3 is also adopted here to describe the problem of short-term scheduling on semicontinuous plants considering multiple due dates. In addition, the following set of parameters and variables are introduced.

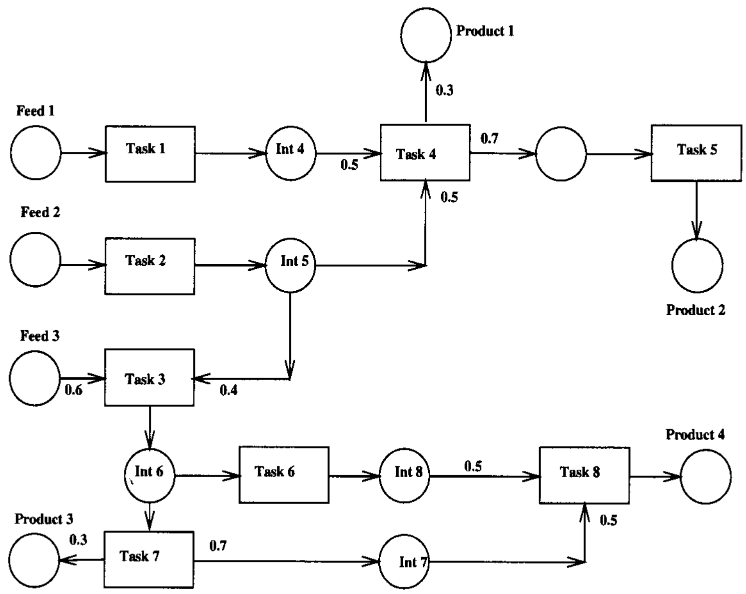

Figure 6. State task network for batch 4.

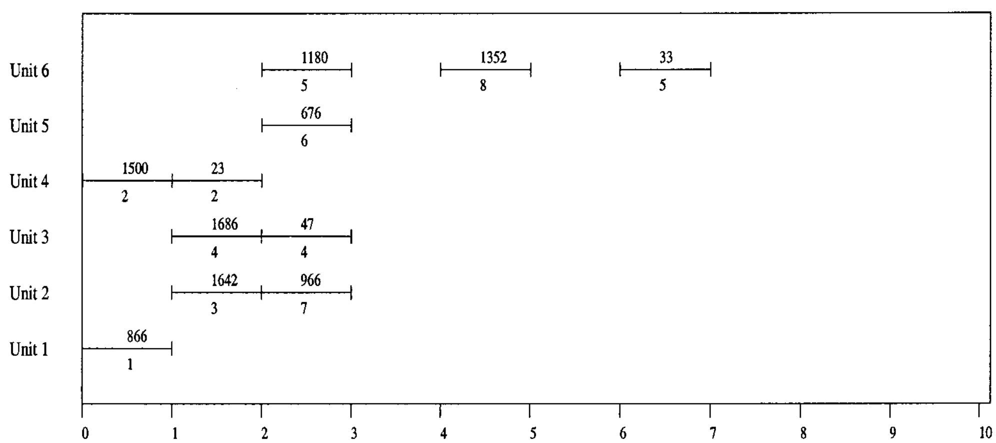

Figure 7. Gantt chart for batch 4.

Notation (cont.)

Sets

IM_j = set of maintenance tasks needed for unit (j)
IO_j = set of test tasks needed for unit (j)

Indices

N_{nd} = subset of event point set N that are connected with the given due dates
im = maintenance tasks
io = test tasks

Parameters

DD(nd) = due dates
Lm(im,j) = duration of maintenance task (im) at unit (j)
Lo(io,j) = duration of test task (io) at unit (j)

Variables

sh(s,n) = part of the demand of state (s) that cannot be met at event point (n)
wvm(im,n) = binary variables that assign the beginning of the maintenance task (im) at event point (n)
wvo(io,n) = binary variables that assign the beginning of the test task (io) at event point (n)
$T_m^s(\mathrm{im},j,n)$ = starting time of maintenance task (im) at event point (n) at unit (j)
$T_o^s(\mathrm{io},j,n)$ = starting time of test task (io) at event point (n) at unit (j)

On the basis of the aforementioned notation, the mathematical model takes the following form:

1. Allocation Constraints.

$$\sum_{i \in I_j} \mathrm{wv}(i,n) = \mathrm{yv}(j,n), \quad \forall\, j \in J,\, n \in N \qquad (18)$$

These constraints ensure that only one task (i) can be performed at each unit (j) at each event point (n).

2. Capacity Constraints.

$$R_{ij}^{\min}[T^{\mathrm{f}}(i,j,n) - T^{\mathrm{s}}(i,j,n)] \le B(i,j,n) \le R_{ij}^{\max}[T^{\mathrm{f}}(i,j,n) - T^{\mathrm{s}}(i,j,n)], \quad \forall\, i \in I,\, j \in J_i,\, n \in N \quad (19)$$

Since the processes involved operate in a continuous mode, the maximum and minimum amount of material being produced at each event point (n) at unit (j), denoted as $B(i,j,n)$, are expressed in terms of the maximum and minimum production rates for this unit when the particular task is performed, $R_{ij}^{\max}$ and $R_{ij}^{\min}$, and the duration of this task in this unit at the particular event point (n).

Table 8. Data for Batch 2

units, tasks			
units	size	units suitability	processing times
heater	100	heating	1
reactor 1	50	reactions 1,2,3	2,2,1
reactor 2	80	reactions 1,2,3	2,2,1
still	200	separation	1 (product 2) 2 (intermediate AB)

states		
states	capacity limits	prices
feeds A,B,C	unlimited	0
hot A	100	
intermediate AB	200	
intermediate BC	150	
intermediate E	200	
product 1	unlimited	60
product 2	unlimited	45

demands					
	time				
products	5	6	8	9	10
1	20	10	20	12	
2				32.5	32.5

cost data				
costc_{ijn} = 20	costv_{11n} = 0.1	costv_{23n} = 0.25	costv_{33n} = 0.25	costst$_{sn}$ = 0.18
	costv_{43n} = 0.15	costv_{22n} = 0.16	costv_{32n} = 0.35	
		costv_{42n} = 0.1		

Table 9. Results for Batch 2

	proposed formulation
constraints	441
variables	316
binary vars	54
integer optimum	5771
relaxation optimum	5861
integrality gap	0.015
no. nodes	10
CPU time (s)	0.15

3. Material Balances.

$$\mathrm{ST}(s,n) = \mathrm{ST}(s,n-1) - d(s,n) + \sum_{i\in I_s}\rho^{\mathrm{p}}_{si}\sum_{j\in J_i} B(i,j,n-1) \quad \forall\, s \in S,\, n \in N \tag{20}$$

$$\mathrm{ST}(s,n) = \mathrm{ST0}(s) \tag{21}$$

where ST0(s) is the initial amount of material (s) and $\rho^{\mathrm{p}}_{si} \geq 0$ represent the proportion of state (s) produced from task (i). To facilitate the incorporation of due dates as it will be discussed later in this section, the material balances are written so as the amount of material that has been produced at event point ($n-1$), $B(i,j,n-1)$, is used for demand satisfaction, $d(s,n)$, at event point (n).

4. Demand Constraints.

$$\mathrm{sh}(s,n) = \mathrm{sh}(s,n-1) - d(s,n) + R(s,n) \quad \forall\, s \in S,\ \forall\, n \in N \tag{22}$$

$$\mathrm{sh}(s,n_0) = R(s,n_0), \quad \forall\, s \in S \tag{23}$$

where $R(s,n)$ correspond to the amount of material required at event point (n). Note that demand constraints allow the partial satisfaction of required demand, since sh(s,n) represent the part of the demand that cannot be met. Also, shortage in material availability SH(s,n) at point (n) is transferred at the next event point ($n+1$).

Table 10. Data for Batch 4

units, tasks			
units	size	units suitability	processing times
unit 1	1000	task 1	1
unit 2	2500	tasks 3,7	1
unit 3	3500	task 4	1
unit 4	1500	task 2	1
unit 5	1000	task 6	1
unit 6	4000	task 5,8	1

states		
states	capacity limits	prices
feeds 1,2,3	unlimited	0
intermediate 4	1000	
intermediate 5	1000	
intermediate 6	1500	
intermediate 7	2000	
intermediate 8	0	
intermediate 9	3000	
products 1,2,3,4	unlimited	18,19,20,21

demands						
	time					
products	3	4	5	6	7	8
1	110	110	133.3	100	33.3	33.3
2		233.1	260		360	360
3			116.6		56.6	116.6
4				333.3	333.3	685.8

cost data		
costv_{ijn} = 0.55	costc_{ijn} = 20	costst$_{sn}$ = 0.1

Table 11. Results for Batch 4

	proposed formulation
constraints	833
variables	798
binary vars	64
integer optimum	60297
relaxation optimum	60356
integrality gap	0.001
no. nodes	34
CPU time (s)	0.27

5. Duration Constraints.

$$L\mathrm{wv}(i,n) \leq T^{\mathrm{f}}(i,j,n) - T^{\mathrm{s}}(i,j,n) \leq H\mathrm{wv}(i,n), \quad \forall\, i \in I,\, j \in J_i,\, n \in N \tag{24}$$

These constraints ensure that the duration of task (i) performed at unit (j) at event point (n), represented by the difference $T^{\mathrm{f}}(i,j,n) - T^{\mathrm{s}}(i,j,n)$ is within the considered time horizon (H). An artificial lower bound of (L) is introduced to ensure that the duration becomes zero whenever this task does not take place.

6. Sequence Constraints.

$$T^{\mathrm{s}}(i,j,n+1) \geq T^{\mathrm{f}}(i',j,n) \quad \forall\, j \in J,\, i \in I_j,\, i' \in I_j,\, n \in N \tag{25}$$

As mentioned in section 3.2.4, these constraints maintain the correct sequencing of the tasks, accounting for the fact that some nodes may remain idle.

7. Due Dates. Products are demanded in specific time periods during operation. This can be accommodated in the above formulation by connecting the event points to due dates in a similar way to that done in section

3.2. Considering N_{nd} due dates, DD(nd), and their demands, the following constraints should be incorporated in the model:

$$T^{s}(i,j,n) = \mathrm{DD(nd)}, \quad \forall\, i \in I, j \in J, n \in N_{nd} \quad (26)$$

where N_{nd} is the subset of event points N that are connected with the given due dates. Note that although the units operate in a semicontinuous mode, the starting times of the tasks are connected to due dates of products because of the way the material balances are written.

8. Transition Constraints. To model the transition from one task to another, we need to model properly the transition cost in the objective function and introduce the following set of constraints:

$$x(i,i',n) \le \mathrm{wv}(i,n), \quad \forall\, i \in I, i' \in I, |i| \ne |i'|, n \in N \quad (27)$$

$$x(i,i',n) \le \mathrm{wv}(i',n+1) + (1 - \sum_{j \in J_i} \mathrm{yv}(j,n+1)), \quad \forall\, i \in I, i' \in I, |i| \ne |i'|, n \in N \quad (28)$$

$$x(i,i',n) \le \mathrm{wv}(i',n+2) + (1 - \sum_{j \in J_i} \mathrm{yv}(j,n+2)), \quad \forall\, i \in I, i' \in I, |i| \ne |i'|, n \in N \quad (29)$$

$$x(i,i',n) \ge \mathrm{wv}(i,n) + \mathrm{wv}(i',n+1) - 1, \quad \forall\, i \in I, i' \in I, |i| \ne |i'|, n \in N \quad (30)$$

$$x(i,i',n) \ge \mathrm{wv}(i,n) + \mathrm{wv}(i',n+2) - 1 - \sum_{j \in J_i} \mathrm{yv}(j,n+1), \quad \forall\, i \in I, i' \in I, |i| \ne |i'|, n \in N \quad (31)$$

where $x(i,i',n)$ are the transition variables representing the transition from task (i) to task (i') at event point (n). Note that the above constraints can handle the possibility of a node to remain idle in the following way. Constraints in (27) ensure that if task (i) does not take place at event point (n), then $x(i,i',n) = 0, \quad \forall\, i' \in I$ since no transition exists from task (i) at event point (n). From constraints in (28), $x(i,i',n) = 0$ if task (i') does not take place at event point ($n + 1$). Note, however, that these constraints are relaxed in the case where no task is performed at event point ($n + 1$), in which case yv(j,n+1)= 0 and constraints in (28) take the form $x(i,i',n) \le 1$. Constraints in (29) are written in the same way, but for tasks that take place at event point ($n + 2$) so that if event point ($n + 1$) is idle, the transition will be defined on the basis of the task that takes place on event point ($n + 2$). Constraints in (30) ensure that if task (i) takes place at event point (n) and task (i') at event point ($n + 1$), then $x(i,i',n) \ge 1$, and consequently since from constraints in (27) and (28), $x(i,i',n) \le 1$, $x(i,i',n) = 1$. In the case where no task is performed at event point ($n + 1$), then constraints in (30) are relaxed and $x(i,i',n)$ are defined from constraints in (31) based on the task that take place at event point ($n + 2$).

9. Minimum Run Length. The production requirement of a minimum run length for every operation can be incorporated in the above formulation by enforcing the continuation of the same task until the end of the required run length. This is case-dependent and differs with respect to the relative duration of required due date intervals and minimum run lengths. For example, in the problems presented in section 4.2 where the due date intervals are 168 h and the minimum run length 240 h, only two event points suffice to maintain the same task to accommodate the minimum run length. This is enforced by the following constraints:

$$\mathrm{wv}(i,n) = \mathrm{wv}(i,n-1), \quad \forall\, i \in I, n \in N \quad (32)$$

$$(T^{f}(i,j,n) - T^{s}(i,j,n)) + (T^{f}(i,j,n-1) - T^{s}(i,j,n-1)) \ge L - H(2 - \mathrm{wv}(i,n) - \mathrm{wv}(i,n-1)), \quad \forall\, j \in J, i \in I_j, n \in N \quad (33)$$

where L is the minimum required run length. The first set of the above constraints ensures that the same task (i) is performed in two consecutive event points ($n - 1$) and (n), whereas the second set enforces the requirement for the overall duration of those two consecutive tasks, represented by the summation of duration of task (i) at event point ($n - 1$) and the duration of task (i) at event point (n) to be greater than the minimum run length L. Note that constraint (41) is relaxed if task (i) does not take place at event point ($n - 1$) and consequently, from constraint (40), at event point (n).

10. Test and Maintenance Tasks Constraints. Maintenance and test tasks that need to be performed during the time horizon of operation in unit (j) are considered as additional tasks in the formulation, im $\in$ IM_j and io $\in$ IO_j, respectively. These tasks take place at event point n, if wvm(im,n) = 1 and wvo(io,n) = 1 or they do not take place if wvm(im,n) = 0 and wvo(io,n) = 0. The required lengths of these tasks are ensured by the following set of constraints:

Duration Constraints.

$$T^{f}_{o}(\mathrm{io},j,n) - T^{s}_{o}(\mathrm{io},j,n) = \mathrm{Lo}(\mathrm{io},j)\mathrm{wvo}(\mathrm{io},n), \quad \forall\, j \in J, \mathrm{io} \in \mathrm{IO}_j, n \in N \quad (34)$$

$$T^{f}_{m}(\mathrm{im},j,n) - T^{s}_{m}(\mathrm{im},j,n) = \mathrm{Lm}(\mathrm{im},j)\mathrm{wvm}(\mathrm{im},n), \quad \forall\, j \in J, \mathrm{io} \in \mathrm{IM}_j, n \in N \quad (35)$$

These constraints express the requirement that the duration of test and maintenance tasks represented by the differences between the final and the starting times, ($T^{f}_{o}(\mathrm{io},j,n) - T^{s}_{o}(\mathrm{io},j,n)$) and ($T^{f}_{m}(\mathrm{im},j,n) - T^{s}_{m}(\mathrm{im},j,n)$), respectively, should be equal to the required duration denoted as Lo(io,j) and Lm(im,j), respectively.

Sequence Constraints.

$$T^{s}(i,j,n+1) \ge T^{f}_{o}(\mathrm{io},j,n), \quad \forall\, j \in J, i \in I_j, \mathrm{io} \in \mathrm{IO}_j, n \in N \quad (36)$$

These constraints express the requirement that the production tasks should follow the completion of both test and maintenance tasks.

Required Starting Time.

$$T^{s}_{o}(\mathrm{io},j,n) \ge T_{o\mathrm{start}}(\mathrm{io},j,n)\mathrm{wvo}(\mathrm{io},n), \quad \forall\, j \in J, i \in I_j, \mathrm{io} \in \mathrm{IO}_j, n \in N \quad (37)$$

$$T^{s}_{m}(\mathrm{im},j,n) \ge T_{m\mathrm{start}}(\mathrm{im},j,n)\mathrm{wvm}(\mathrm{im},n), \quad \forall\, j \in J, i \in I_j, \mathrm{io} \in \mathrm{IM}_j, n \in N \quad (38)$$

where $T_{o\mathrm{start}}(\mathrm{io},j,n)$ and $T_{m\mathrm{start}}(\mathrm{im},j,n)$ are the required starting times for the test and maintenance tasks.

Required End Time.

$$T_o^s(\text{io},j,n) \le (T_{o\text{end}}(\text{io},j,n) - \text{Lo}(\text{io},j)) + H(1 - \text{wvo}(\text{io},n)), \quad \forall\, j \in J,\, i \in I_j,\, \text{io} \in \text{IO}_j,\, n \in N \quad (39)$$

$$T_m^s(\text{im},j,n) \le (T_{m\text{end}}(\text{im},j,n) - \text{Lm}(\text{im},j)) + H(1 - \text{wvm}(\text{im},n)), \quad \forall\, j \in J,\, i \in I_j,\, \text{io} \in \text{IM}_j,\, n \in N \quad (40)$$

where $T_{o\text{end}}(\text{io},j,n)$ and $T_{m\text{end}}(\text{im},j,n)$, are the required end times for the test and maintenance tasks. Constraints (35) and (36) ensure that the test and maintenance tasks start after the required starting times, while constraints (37) and (38) ensure that they are completed within the prespecified time windows. Note also that the allocation constraints need to be modified to accommodate the additional test and maintenance tasks at each event point (n) in the following way:

$$\sum_{i\in I_j}\text{wv}(i,n) + \sum_{\text{io}\in\text{IO}_j}\text{wvo}(\text{io},n) + \sum_{\text{im}\in IM_j}\text{wvm}(\text{im},n) = \text{yv}(j,n), \quad \forall\, j \in J,\, n \in N \quad (41)$$

When these constraints are followed if a maintenance or a test task is performed in unit (j) at event point (n), no other production task can be performed.

11. Objective Function: Cost Minimization. *Cost of Not Meeting the Demand.*

$$\sum_{s,n} c\text{supply}(s)\text{sh}(s,n)$$

where $c$supply(s) is the penalty of not meeting the demand of product (s).

Cost of Not Meeting the Minimum Stock.

$$\sum_{s,n} c\text{stock}(s,n)\text{sth}(s,n)$$

where $c$stock(s,n) is the penalty of not meeting the required levels of safety stock for product (s) at event point (n); sth is defined from $s\text{th}(s,n) \ge$ safety stock $-$ st(s,n).

Cost of Transition.

$$\sum_{i\in I_j, i'\in I_j, j\in J, n\in N} c\text{trans}(i,i')x(i,i',n) + \sum_{\text{im}\in\text{IM}_j, i\in I_j, j\in J, n\in N} c\text{trmi}(\text{im},i)\text{xm}(\text{im},i,n) + \sum_{\text{io}\in\text{IO}_j, i\in I_j, j\in J, n\in N} c\text{troi}(\text{io},i)\text{xo}(\text{io},i,n)$$

where $c$trans(i,i') is the transition cost from task (i) to task (i'), $c$trmi(im,i) is the cost of restarting the operation after a maintenance task (im), and $c$troi(io,i) is the cost of restarting the operation after a test task (io).

Inventory Cost.

$$\sum_{s\in S, n\in N} c\text{hold}(s,n)\left[\text{st}(s,n-1) + \frac{1}{2}\sum_{i\in I_s}\rho^{\text{p}}_{si}\, B(i,j,n-1)\right]$$

where $c$hold(s,n) is the inventory cost for the unit of product (s). On the basis of the assumption that the amount of product (s), $\sum_{i\in I_s}\rho^{\text{p}}_{si}\, B(i,j,n-1)$, is produced uniformly between event points ($n-1$) and (n) (Karimi and McDonald[6]), the holding cost for this amount of product (s) is given by $^1/_2\sum_{i\in I_s}\rho^{\text{p}}_{si}\, B(i,j,n-1)$.

The overall mathematical model corresponds to a MILP problem where wv(i,n) and yv(j,n) are the binary variables.

4.2. Computational Studies. In this section the examples presented by Karimi and McDonald[6] are considered. The 5 examples correspond to the 5 subplants of the multiproduct plant involving 7 machines and producing 14 products. Orders are placed at the end of the first 4 weeks and then at the end of the second and third month. The minimum run length required is 10 days. Problem data involving product demands, initial inventory levels, safety stock targets, machine suitability, and production rate and cost data can be found in Karimi and McDonald.[6] Example 1 that corresponds to the subplant involving machines J1 and the production of products I3, I6, and I11 is presented in detail, and only the results for examples 2−5 are presented. Also, example 3 is used to demonstrate the capability of the formulation to accommodate the production of the same product in different units and the consideration of maintenance and test tasks.

4.2.1. Example 1. The subplant considered here involves the production of three products I3, I6, and I11 in unit J1. To accommodate the minimum run length of 240 h and the due dates that are specified at the end of the first 4 weeks (168 h) and at the end of the second and third month that correspond to 1392 and 2112 h, respectively, 15 event points are considered, including event point n_0. The due dates are connected to nodes n_2, n_6, n_8, n_{11}, and n_{14} as shown in Figure 8.

The sets considered in the mathematical model are

$$J = \{J1\},\ I_{j1} = \{I3, I6, I11\}, \quad S = \{S3, S6, S11\}, \quad N = \{n_0, n_1, ..., n_{14}\}$$

The following constraints are then utilized in the description of the problem.

1. Allocation Constraints.

$$\sum_{i\in I_j}\text{wv}(i,n) = \text{yv}(j,n), \quad \forall\, j \in J,\, n \in N$$

2. Capacity Constraints.

$$R^{\min}_{ij}[T^{\text{f}}(i,j,n) - T^{\text{s}}(i,j,n)] \le B(i,j,n) \le R^{\max}_{ij}[T^{\text{f}}(i,j,n) - T^{\text{s}}(i,j,n)], \quad \forall\, i \in I,\, j \in J,\, n \in N$$

where $R^{\min}_{i3,j1} = 61.6$, $R^{\min}_{i6,j1} = 75.2$, and $R^{\min}_{i11,j1} = 66.3$ are the minimum production rates for machine $J1$, depending on the product that is being produced. The maximum production rates are $R^{\max}_{i3,j1} = 123.3$, $R^{\max}_{i6,j1} = 150.3$, and $R^{\max}_{i11,j1} = 132.5$.

3. Material Balances.

$$\text{ST}(s,n) = \text{ST}(s,n-1) - d(s,n) + \sum_{i\in I_s}\rho^{\text{p}}_{si}\sum_{j\in J_i} B(i,j,n-1) \quad \forall\, s \in S,\, n \in N$$

$$\text{ST}(s,n_0) = \text{ST0}(s)$$

In this example there is an one-to-one correspondence between task and product. Consequently, only $\rho^{\text{p}}_{s6i6} = 1$, $\rho^{\text{p}}_{s3i3} = 1$, and $\rho^{\text{p}}_{s11i11} = 1$ and the rest of the parameters $\rho^{\text{p}}_{si} = 0$.

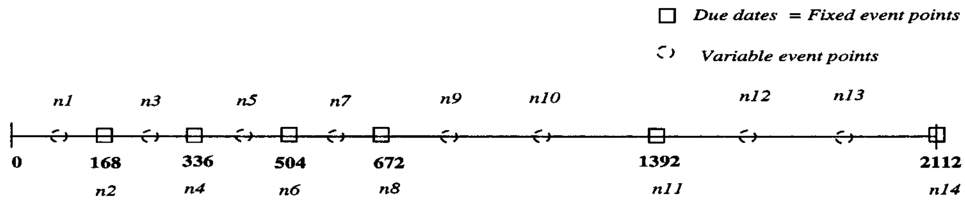

Figure 8. Location of the event points.

4. Demand Constraints.

$$\mathrm{sh}(s,n) = \mathrm{sh}(s,n-1) - d(s,n) + R(s,n), \quad \forall\, s \in S, \; \forall\, n \in N$$

$$\mathrm{sh}(s,n_0) = R(s,n_0), \quad \forall\, s \in S$$

Since products are only required at event points n_2, n_4, n_6, n_8, n_{11}, and n_{14}, then $R(s,n)$ at these points correspond to the amount of material s required at due dates.

5. Duration Constraints.

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

$$T^{\mathrm{f}}(i,j,n) - T^{\mathrm{s}}(i,j,n) \ge l\mathrm{wv}(i,n), \quad \forall\, i \in I, j \in J_i, n \in N$$

where $l = 1$, introduced to ensure a minimum task duration if this task takes place at event point n.

6. Sequence Constraints.

$$T^{\mathrm{s}}(i,j,n+1) \ge T^{\mathrm{f}}(i',j,n), \quad \forall\, j \in J, i \in I_j, i' \in I_j, n \in N$$

7. Minimum Run Length Constraints.

$$\mathrm{wv}(i,n_2) = \mathrm{wv}(i,n_1) \;\forall\, i \in I_j$$

$$\mathrm{wv}(i,n_4) = \mathrm{wv}(i,n_3) \;\forall\, i \in I_j$$

$$\mathrm{wv}(i,n_6) = \mathrm{wv}(i,n_5) \;\forall\, i \in I_j$$

$$\mathrm{wv}(i,n_8) = \mathrm{wv}(i,n_7) \;\forall\, i \in I_j$$

$$\mathrm{wv}(i,n_{11}) = \mathrm{wv}(i,n_{10}) \;\forall\, i \in I_j$$

$$(T^{\mathrm{f}}(i,j,n) - T^{\mathrm{s}}(i,j,n)) + (T^{\mathrm{f}}(i,j,n-1) - T^{\mathrm{s}}(i,j,n-1)) \ge L - H(2 - \mathrm{wv}(i,n) - \mathrm{wv}(i,n-1)), \quad \forall\, j \in J, i \in I_j, n = n_2, n_4, n_6, n_8, n_{11}, n_{14}$$

where $L = 240$ h is the required minimum production time length. Note that to allow the continuation of a task in two consecutive event points so as to ensure minimum run length, the above constraints are altered so as to become inactive if the task continues in more than two event points.

$$(T^{\mathrm{f}}(i,j,n) - T^{\mathrm{s}}(i,j,n)) + (T^{\mathrm{f}}(i,j,n-1) - T^{\mathrm{s}}(i,j,n-1)) \ge L - H(2 - \mathrm{wv}(i,n) - \mathrm{wv}(i,n-1)) - H((\mathrm{wv}(i,n-2) + \mathrm{wv}(i,n-3)), \quad \forall\, j \in J, i \in I_j, n = n_4, n_6, n_8, n_{11}, n_{14}$$

and the following constraints need to be incorporated:

$$\sum_{\mathrm{ns}=n-3}^{\mathrm{ns}=n} (T^{\mathrm{f}}(i,j,\mathrm{ns}) - T^{\mathrm{s}}(i,j,\mathrm{ns})) \ge L - H(4 - \sum_{\mathrm{ns}=n-3}^{\mathrm{ns}=n} \mathrm{wv}(i,\mathrm{ns})), \;\forall\, j \in J, i \in I_j, n = n_4, n_6, n_8$$

8. Objective: Cost Minimization.

$$\sum_{s,n} c\mathrm{supply}(s)\mathrm{sh}(s,n) + \sum_{s,n} c\mathrm{stock}(s,n)\mathrm{sth}(s,n) + \sum_{i \in I_j, i' \in I_j, j \in J, n \in N} c\mathrm{trans}(i,i')x(i,i',n) + \sum_{\mathrm{im} \in \mathrm{IM}_j, i \in I_j, j \in J, n \in N} c\mathrm{trmi}(\mathrm{im},i)\mathrm{xm}(\mathrm{im},i,n)) + \sum_{s,n} c\mathrm{hold}_{s \in S, n \in N}(\mathrm{st}(s,n-1) + (1/2)\rho^{\mathrm{p}}_{si}\, b(i,j,n-1))$$

4.2.2. Results. The above mathematical model is a MILP problem involving 44 binary variables, 428 continuous variables, and 750 constraints. The problem is solved using GAMS/CPLEX in a HP-C160 workstation requiring 0.72 CPU s and the exploitation of 80 nodes to find the optimal solution of cost \$125,602 within 10^{-6} of the best integer solution. The optimal schedule for this machine is shown in Figure 9. Note that the proposed formulation requires fewer binary and continuous variables, 44 and 428, compared to 70 and 467 required by the formulation proposed by Karimi and McDonald[6] as illustrated in Table 13. Their formulation also requires the exploitation of more nodes, 178, solved in 5.0 CPU s in a IBM RS-6000 workstation compared to 80 nodes needed for the solution of the proposed model that takes 0.72 CPU s in a HP-C160 workstation. For the solution of all the examples, priorities are assigned for binary variables $\mathrm{wv}(i,n)$ for branching in the branch-and-bound method based on the value of event point (n). In particular, higher priority is given for lower values of (n) since decisions taken at earlier times would affect the decisions at later event points. Also, it was found that the use of an artificial term $\sum_{i,n}$-term $\mathrm{wv}(i,n)$, does not affect the quality of the solution, but it results in great computational savings since it helps reduce the integrality gap.

The rest of the examples considered correspond to the subplants of the multiproduct plant presented by Karimi and McDonald[6] as shown in Table 12.

Examples 3 and 4 involve the production of the same products in different machines. Also, examples 3–5 require the execution of a maintenance or a test task during the time horizon of 3 months. The exact constraints for the incorporation of these features follow for example 3.

1. Same Tasks—Different Machines. To maintain the main features of the proposed formulation as mentioned in Ierapetritou and Floudas,[1] the consideration of the same tasks in different units is achieved by representing these tasks as different tasks, depending on the machine where they take place. For example 3, machine J3 performs tasks I1, I2, I7, and I14 to produce products S1, S2, S7, and S14, whereas machine J4 produces products S2, S7, and S14 by performing tasks I2a, I7a, and I14a. When the different tasks are modeled in this way, no modifications are needed in the mathematical model.

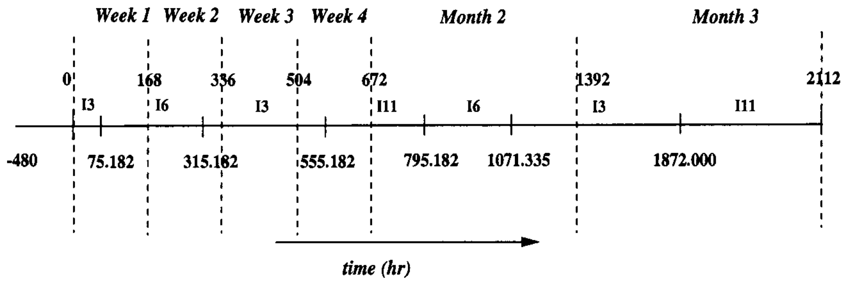

Figure 9. Schedule for example 1.

Table 12. Subplants

example	machines	products	test(t)/maintenance(m)
1	J1	I3, I6, I11	
2	J2	I4, I10	
3	J3	I1, I2, I7, I14	I15, I16 (t1,t2)
	J4	I2, I7, I14	
4	J5	I5, I12	I18(m)
	J6	I5, I12	
5	J7	I8, I9, I13	I17 (t3)

2. Test Tasks. To incorporate the test tasks that need to be performed in unit J3 during operation and in particular between [15,30] and [30,60] day, respectively, the following constraints need to be incorporated.

Allocation Constraints.

$$\sum_{i\in I_j} \mathrm{wv}(i,n) + \sum_{\mathrm{io}\in \mathrm{IO}_j} \mathrm{wvo}(\mathrm{io},n) = \mathrm{yv}(j,n), \quad \forall\, j \in J,\ n \in N$$

where $\mathrm{IO}_{J3} = \{I15, I16\}$ and $\mathrm{IO}_{J4} = \{\ \}$.

Duration Constraints.

$$T_o^{\mathrm{f}}(\mathrm{io},j,n) - T_o^{\mathrm{s}}(\mathrm{io},j,n) \geq \mathrm{Lo}(\mathrm{io},j)\mathrm{wvo}(\mathrm{io},n), \quad \forall\, j \in J,\ \mathrm{io} \in \mathrm{IO}_j,\ n \in N$$

Sequence Constraints.

$$T^{\mathrm{s}}(i,j,n+1) \geq T_o^{\mathrm{f}}(\mathrm{io},j,n) \quad \forall\, j \in J,\ i \in I_j,\ \mathrm{io} \in \mathrm{IO}_j,\ n \in N$$

Required Starting Time.

$$T_o^{\mathrm{s}}(\mathrm{io},j,n) \geq T_{\mathrm{ostart}}(\mathrm{io},j,n)\mathrm{wvo}(\mathrm{io},n) \quad \forall\, j \in J,\ i \in I_j,\ \mathrm{io} \in \mathrm{IO}_j,\ n, \in N$$

where $T_{\mathrm{ostart}}(I15,J3,n) = 360$ h and $T_{\mathrm{ostart}}(I16,J3,n) = 720$ h are the required starting times for test tasks I15 and I16.

Required End Time.

$$T_o^{\mathrm{s}}(\mathrm{io},j,n) \leq (T_{\mathrm{oend}}(\mathrm{io},j,n) - \mathrm{Lo}(\mathrm{io},j)) + H(1 - \mathrm{wvo}(\mathrm{io},n)) \quad \forall\, j \in J,\ i \in I_j,\ \mathrm{io} \in \mathrm{IO}_j,\ n \in N$$

where $T_{\mathrm{oend}}(I15,J3,n) = 720$ h and $T_{\mathrm{oend}}(I16,J3,n) = 1440$ h are the required end times for the test tasks.

The results for examples 1−5 are given in Table 13 and the optimal schedules obtained are illustrated in Figure 10. The results are compared with the results obtained by the second model of Karimi and McDonald,[6] which proved to outperform model 1. The CPU times they report for all the examples correspond to a IBM RS-6000 workstation. Note that in their formulation to improve model performance, additional constraints are added. For instance, for the first model, they proposed constraints that enforce the idle slots to stack up at the last slots in a schedule, constraints that eliminate the schedules where the demands are not satisfied at a given period but there is some free time available, and constraints that eliminate schedules where two consecutive slots are assigned to the same task. The presence of such constraints, however, restrict the feasible space of the scheduling problem, leading to the most expected solution for these particular examples but are not necessarily applicable to different scheduling.

Table 13. Results for Examples 1−5

	proposed approach	Karimi and McDonald
	Example 1	
binary vars	44	70
continuous vars	428	467
constraints	750	620
CPU time	0.72[a]	5.0[b]
nodes	80	178
cost	125602.6	125602.6
LP relaxation	3652.46	4292
	Example 2	
binary vars	30	48
continuous vars	287	314
constraints	534	449
CPU time	0.31[a]	2.0[b]
nodes	41	113
cost	16137.7	16137.7
LP relaxation	1287.05	3044
	Example 3	
binary vars	140	152
continuous vars	1277	1378
constraints	5927	1417
CPU time	35.6[a]	320[b]
nodes	802	2563
cost	350216.2	350257
LP relaxation	259368.5	302934
	Example 4	
binary vars	96	107
continuous vars	570	682
constraints	1751	893
CPU time	13.76[a]	22[b]
nodes	861	475
cost	794385.7	794385.7
LP relaxation	541037.3	791271
	Example 5	
binary vars	49	67
continuous vars	538	572
constraints	1042	627
CPU time	9.92[a]	15[b]
nodes	792	660
cost	42072.2	42072.2
LP relaxation	1776.87	14233

[a] HP-C160 workstation. [b] IBM RS-6000 workstation.

In particular, for example 2, the proposed formulation requires fewer binary and continuous variables, 30 and

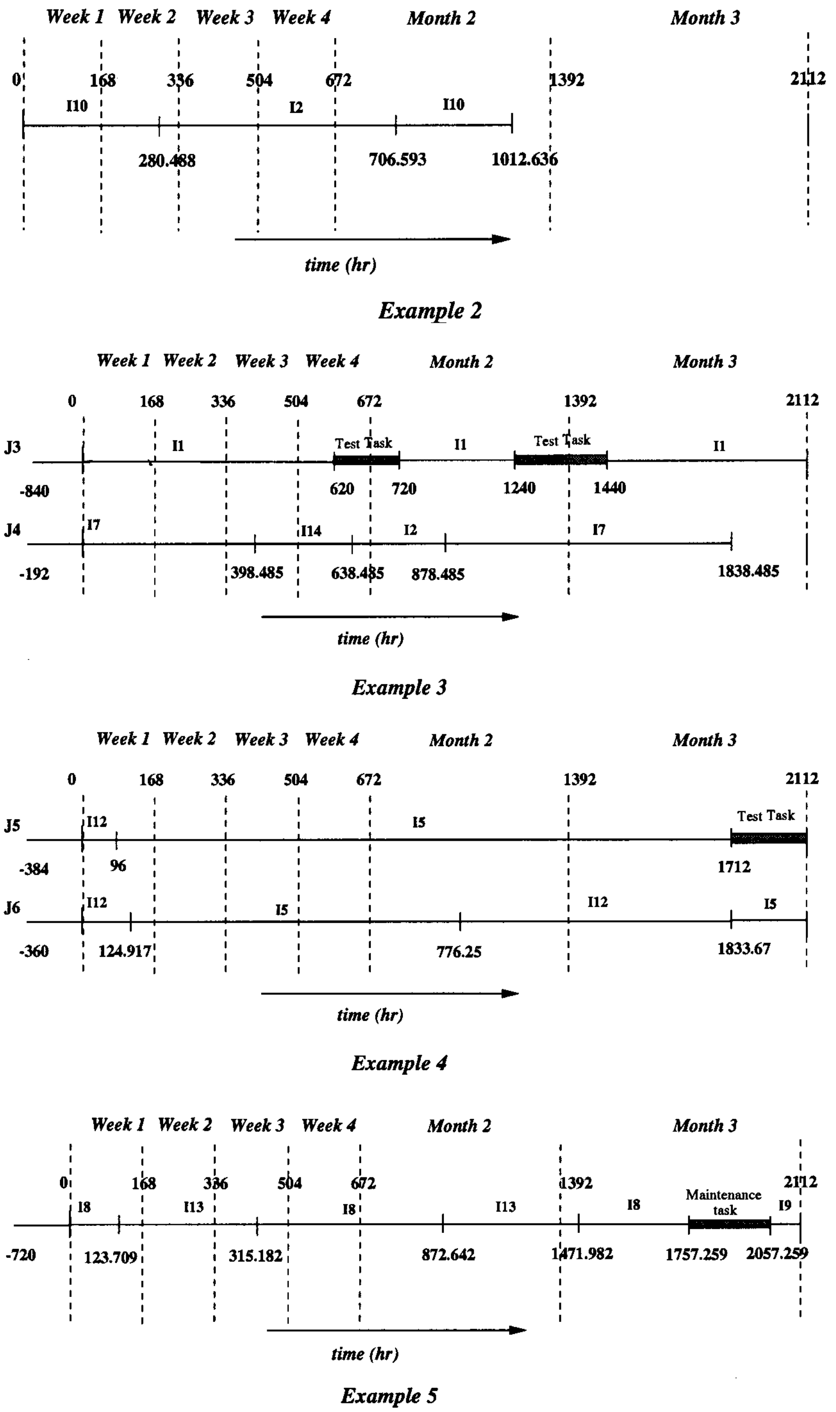

Figure 10. Schedule for examples 2−5.

287 compared to 48 and 314 used by the formulation proposed by Karimi and McDonald.[6] Note that although it exhibits a larger integrality gap, it requires fewer branch-and-bound nodes, 41 compared to 113 needed by Karimi and McDonald.[6] For example 3, a better solution is obtained which also corresponds to a smaller makespan for unit J4 as illustrated in Figure 10. For this example a more efficient solution is achieved on the basis of the fact that the two machines are not competing for different tasks since the rate range for unit J3 is much lower than the rate range for unit J4 and product I1 can only be produced in unit J3. Note also that since two test tasks are required for unit J3, its remaining operation time is not sufficient to meet the demand, even of product I1. The above features suggest a further decomposition of this example into unit J3 where only product I1 is produced and unit J4 that produces products I2, I7, and I14. For example 4, the proposed approach requires fewer binary and continuous variables, 96 and 570 compared to 107 and 682 required by Karimi and McDonald.[6] However, since both units have the same rates, the problem is more difficult to solve than that of example 3. Finally, for example 5, fewer binary and continuous variables are needed by the proposed approach than by the Karimi and McDonald formulation, namely, 49 and 538 compared to 67 and 572, respectively. However, in the last two examples, the proposed formulation explores more branch-and-bound nodes to find the optimal solution. It is important to point out that in these examples the

restriction of minimum length requirement of 10 days does not allow the full exploitation of continuous time-scheduling formulation advantages since intermediate time constraints in addition to the ones that correspond to due dates have to be introduced.

5. Conclusions

In this paper, a novel formulation for the short-term scheduling of batch and semicontinuous plants is proposed. The formulation is tailored to accommodate intermediate due dates where specific product demands have to be satisfied. The mathematical model was based on the previous work of Ierapetritou and Floudas[1,2] where a new continuous time formulation was presented to effectively address the problem of short-term scheduling in batch, continuous, and mixed production facilities where product demands are specified at the end of the time horizon. Further exploitation of the plant operation mode results in the most efficient solution of the problem. Several examples are provided to illustrate the capabilities of the proposed continuous-time formulations, and it was demonstrated that a variety of problems presented in the literature can be addressed efficiently.

Acknowledgment

The authors gratefully acknowledge financial support from the National Science Foundation and Mobil Technology Co.

Literature Cited

(1) 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.

(2) 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*, 4360−4374.

(3) Sahinidis, N. V.; Grossmann, I. E. Reformulation of Multiperiod Models for Planning and Scheduling of Chemical Processes. *Comput. Chem. Eng.* **1991**, *15*, 255−272.

(4) Mockus, L.; Reklaitis, G. V. A New Global Optimization Algorithm for Batch Process Scheduling. In *State of the Art in Global Optimization*; Floudas, C. A., Pardalos, P. M., Eds.; Kluwer Academic Publishers: Dordrecht, 1996; pp 521−538.

(5) 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*, 3037−3051.

(6) Karimi, I. A.; McDonald, C. M. Planning and Scheduling of Parallel Semicontinuous Processes. 2. Short-Term Scheduling. *Ind. Eng. Chem. Res.* **1997**, *36*, 2701−2714.

(7) Glover, F. Improved Linear Integer Programming Formulations of Nonlinear Integer Problems. *Man. Sci.* **1975**, *22*.

Received for review February 12, 1999
Revised manuscript received May 27, 1999
Accepted June 7, 1999

IE990108R