Annals of Operations Research 139, 131–162, 2005

Mixed Integer Linear Programming in Process Scheduling: Modeling, Algorithms, and Applications

CHRISTODOULOS A. FLOUDAS * floudas@titan.princeton.edu
XIAOXIA LIN
Department of Chemical Engineering, Princeton University, Princeton, NJ 08544-5263, USA

Abstract. This paper reviews the advances of mixed-integer linear programming (MILP) based approaches for the scheduling of chemical processing systems. We focus on the short-term scheduling of general network represented processes. First, the various mathematical models that have been proposed in the literature are classified mainly based on the time representation. Discrete-time and continuous-time models are presented along with their strengths and limitations. Several classes of approaches for improving the computational efficiency in the solution of MILP problems are discussed. Furthermore, a summary of computational experiences and applications is provided. The paper concludes with perspectives on future research directions for MILP based process scheduling technologies.

Keywords: chemical process scheduling, mixed-integer linear programming (MILP), discrete-time model, continuous-time model, branch and bound

Process scheduling has attracted an increasing amount of attention from both the academia and the industry in the past decade. The reason for this is twofold. First, it reflects the pressure faced by the chemical processing and manufacturing related industries to improve productivity and reduce costs. Second, it is driven by the substantial advances of related modeling and solution techniques, as well as the rapidly growing computational power. The problem of interest is to determine the most efficient way to produce a set of products in a time horizon given a set of limited resources and processing recipes. The activities to be scheduled usually take place in multiproduct and multipurpose plants, in which a wide variety of different products can be manufactured via the same recipe or different recipes by sharing limited resources, such as equipment, material, time, and utilities. These types of plants have long been employed to manufacture chemicals in small quantities which have high added-values or frequently changing process parameters or demands. Their inherent operational flexibility provides the platform for great benefits that can be obtained through good production schedules.

Mathematical programming, especially Mixed Integer Linear Programming (MILP), because of its rigorousness, flexibility and extensive modeling capability, has become one of the most widely explored methods for process scheduling problems. Applications of MILP based scheduling methods range from the simplest single-stage

*Corresponding author.

single-unit multiproduct processes to the most general multipurpose processes. These process scheduling problems are inherently combinatorial in nature because of the many discrete decisions involved, such as equipment assignment and task allocation over time. They belong to the set of NP-complete problems (Garey and Johnson, 1979), which means that the solution time scales exponentially as the problem size increases in the worst case. This has important implications for the solution of scheduling problems.

There has been a number of reviews related to process scheduling in the chemical engineering and the operations research literature. Reklaitis (1992) reviewed the scheduling and planning of batch process operations, focusing on the basic components of chemical process scheduling problems and the available solution methods. Rippin (1993) summarized the development of batch process systems engineering with particular reference to the areas of design, planning, scheduling and uncertainty. Grossmann et al. (1996) provided an overview of mixed-integer optimization techniques for the design and scheduling of batch chemical processes. They concentrated on basic solution methods and recent developments for mixed-integer linear and nonlinear programming problems and also discussed issues in modeling and reformulation. Bassett et al. (1996a) reviewed existing strategies for implementing integrated applications based on mathematical programming models and examined four classes of integration including scheduling, control, planning and scheduling across single and multiple sites, and design under uncertainty. Applequist et al. (1997) discussed the formulation and solution of process scheduling and planning problems, as well as issues associated with the development and use of scheduling software. Shah (1998) examined first different techniques for optimizing production schedules at individual sites, with an emphasis on formal mathematical methods, and then focused on progress in the overall planning of production and distribution in multisite flexible manufacturing systems. Pekny and Reklaitis (1998) discussed the nature and characteristics of the scheduling/planning problems and pointed out the key implications for the solution methodology for these problems. They reviewed the available scheduling technologies, including randomized search, rule-based methods, constraint guided search, simulation-based strategies, as well as mathematical programming formulation based approaches using conventional and engineered solution algorithms. Pinto and Grossmann (1998) presented an overview of assignment and sequencing models used in process scheduling with mathematical programming techniques. They identified two major categories of scheduling models—one for single-unit assignment and the other for multiple-unit assignment—and discussed the critical issues of time representation and network structure.

Given the computational complexity of combinatorial problems arising from process scheduling, it is of crucial importance to develop effective mathematical formulations to model the manufacturing processes and to explore efficient solution approaches for such problems. The objective of this paper is to present an overview of advances in MILP based approaches for process scheduling problems. We focus on the short-term scheduling of general network represented processes. The rest of this paper is organized as follows. First, the mathematical models that have been proposed in the literature are classified and presented along with their strengths and limitations. Then, a variety of approaches developed to overcome the computational difficulty in the solution of large MILP problems are discussed. Subsequently, a summary of computational experiences and applications follows. Finally, the paper will conclude with views on future research directions for MILP based process scheduling technologies.

1. MILP mathematical formulations

1.1. Characteristics and classification of process scheduling problems

In the context of chemical processing systems, the scheduling problem generally consists of the following components: (i) production recipes, which specify the sequences of tasks to be performed for manufacturing given products; (ii) available processing/storage equipment; (iii) intermediate storage policy; (iv) production requirements; (v) specifications of resources, such as utilities and manpower; and (vi) a time horizon of interest. The goal is to determine a schedule which includes the details of (i) the sequence of tasks to be performed in each piece of equipment; (ii) the timing of each task; and (iii) the amount of material to be processed (i.e., batch-size) by each task. The performance of a schedule is measured with one or more criteria, for example, the overall profit, the operating costs, and the makespan.

1.1.1. Process representation

Production recipes in chemical processes can be very complex. Furthermore, different products can have very low recipe similarities. For scheduling, network based representations have been developed to represent these production recipes in an ambiguity-free way. Kondili, Pantelides, and Sargent (1993) proposed a general framework of State-Task Network (STN). The STN representation of a chemical process is a directed graph with two types of distinctive nodes: the *state* nodes denoted by a circle, representing raw materials, intermediate materials or final products, and the *task* nodes denoted by a rectangle box, representing a physical or chemical operation, such as reaction, heating and separation. The fraction of a state consumed or produced by a task, if not equal to one, is given beside the arch linking the corresponding state and task nodes. An example of the STN representation is presented in figure 1. This process exhibits a number of features characteristic of many chemical processes, including multiple inputs and outputs (Task 4 has two inputs, S3 and S5, and two outputs, S2 and S4), shared raw material among different tasks (both Task2 and Task3 use S2), production of the same product by different tasks (S2 is produced by Task1 and Task4), and recycle (Task4 produces S2 which is used by Task2, an earlier task in the processing sequence).

Pantelides (1993) extended the STN to the Resource-Task Network (RTN) framework, which characterizes the entirely uniform description of available resources, such as materials, processing equipment, storage, and utilities. Each task transforms a set of resources to another set of resources. Figure 2 gives an example of the RTN framework. In addition to materials, other resources and their interactions with the tasks are also included in the network. For example, Resource1 $\sim$ Resource4 are materials that the

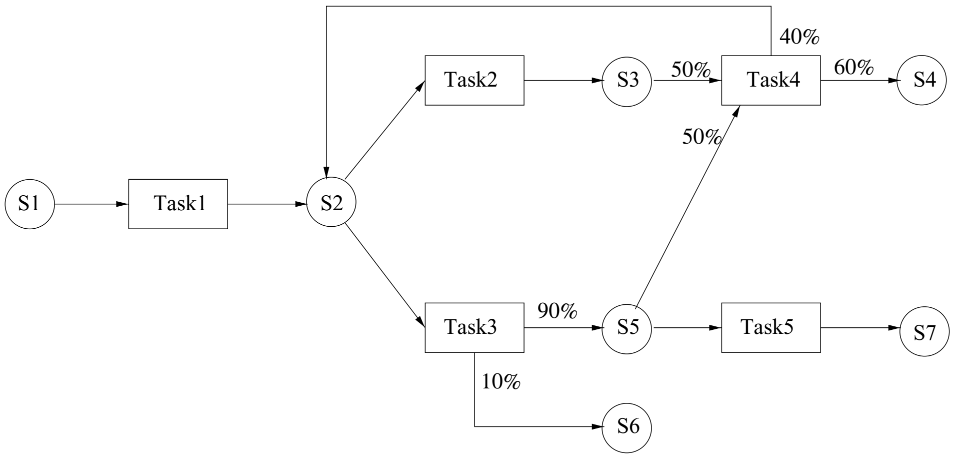


Figure 1. Example of State-Task Network (STN) process representation.

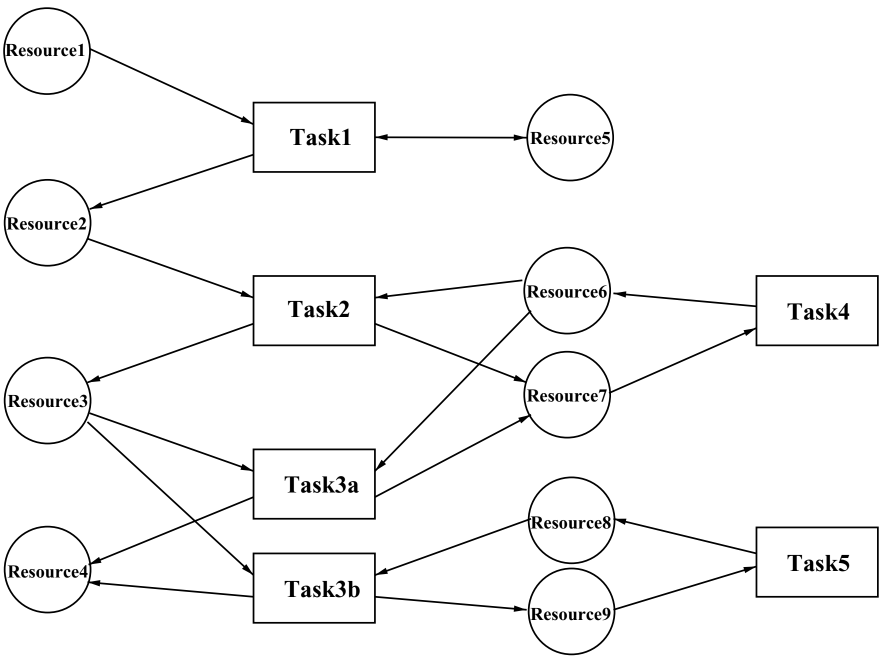


Figure 2. Example of Resource-Task Network (RTN) process representation.

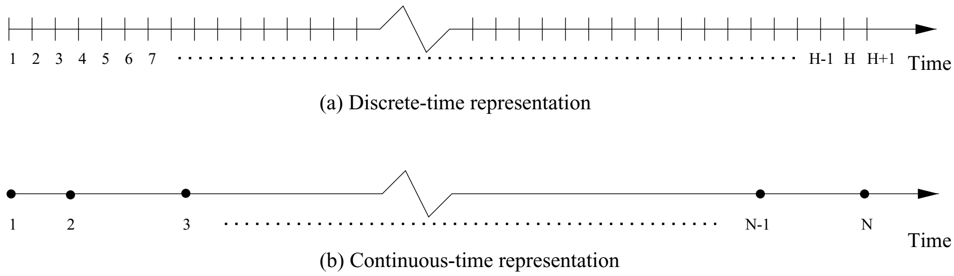
tasks consume and produce. Resource5 $\sim$ Resource9 represent equipment and they are considered to be consumed at the start of a task and produced at the end. Certain properties of a piece of equipment may be changed by a task (e.g., cleanliness) and it may require another task (e.g., cleaning) to restore it before being used again. In this case, the equipment is treated as two different resources before and after the task. For instance, in figure 2, Task2 uses Resource6, a clean unit, and "produces" Resource7, a soiled unit; Resource7 is restored back to Resource6 by Task4, a cleaning operation. Note that tasks taking place in different units are now regarded as different tasks, for example, Task3a and Task3b consume the same raw material, Resource3, and produce the same product Resource4, but they use two different units, Resource6 and Resource8, respectively.

It should be pointed out that there exists a special class of processes, called sequential processes, which involves relatively simple production recipes and are frequently employed in chemical and other industries. This type of processes exhibit linear structure in the production recipe without material merging/splitting or recycle. To make a product, the corresponding raw material undergoes one or multiple stages. Each stage consists of a simple operation with a single input from the previous stage and a single output going to the next stage, though there can be one or parallel units at each stage. Furthermore, different products follow the same or very similar processing sequences. Due to this special structure of the processing recipe, orders/batches/jobs are used to represent production and mass balances are often not taken into account explicitly. A significant amount of work has been dedicated to the development of MILP based approaches for this class of scheduling problems exploring the special structure of sequential processes, which can be classified into two main groups as methods based on time slots (for example, Pinto and Grossmann, 1995; Pinto et al., 1998; Karimi and McDonald, 1997; Lamba and Karimi, 2002a; Bok and Park, 1998; Moon and Hrymak, 1999) and methods based on direct definition of sequences and/or timings of orders/batches (for example, Ku and Karimi, 1998; Cerdá, Henning, and Grossmann, 1997; Méndez, Henning, and Cerdá, 2000b, 2001; Moon, Park, and Lee, 1996; Hui, Gupta, and Meulen, 2000; Hui and Gupta, 2001; Orçun, Altinel, and Hortaçsu, 2001; Lee et al., 2002). Readers interested in this class of scheduling problems are directed to a recent review by Floudas and Lin (2004). In this paper, we focus on the scheduling of general network-represented processes.

In addition to the complexity of processing recipes, scheduling problems of chemical processes are further complicated by a number of other considerations, including intermediate storage, changeovers, batch and continuous operation modes, demand patterns, resource restrictions, and a variety of objectives (for more details of these characteristics, see Pinto and Grossmann, 1998; Floudas and Lin, 2004). The complexity of the process scheduling problems necessitates the development of effective schemes for organizing the large amount of information required to describe most scheduling applications. For instance, Zentner et al. (1998) proposed a high level language as a compact and context independent means of expressing a wide variety of process scheduling problems.

1.1.2. Time representation

To formulate a mathematical model for any process scheduling problem, the first major issue that arises is how to represent the time. Based on two different ways for time representation, we classify all existing formulations into two main categories: discrete-time models and continuous-time models. The scheduling formulations in the first category follow the approach of time discretization. The time horizon of interest is divided into a number of time intervals with uniform durations and decisions to be made are associated



(a) Discrete-time representation

(b) Continuous-time representation

Figure 3. Discrete and continuous representations of time.

with these time intervals, as illustrated in part (a) of figure 3. Therefore, activities that affect the schedule or changes of the state of the manufacturing system, such as the start of a task or change of inventory, can only take place at a specific instance of each time interval, for example, at the beginning of each time interval. This is essentially an approximation of time. In many cases, a short length of time has to be used for the duration of the time intervals in order to achieve a reasonable degree of accuracy. For example, when all the processing times are fixed, their greatest common factor (GCF) can be selected as the duration of the time intervals, otherwise only suboptimal solutions can be obtained. When very small time interval length needs to be used and/or the time horizon under consideration is long, the number of the discretized time intervals can be very large. This leads to very large combinatorial problems, which are computationally very expensive to solve or even intractable. These two main drawbacks, namely the approximation nature and the computational difficulty in the solution of the resulting large combinatorial models, substantially restrict the application of the discrete-time approach, especially to real-world problems.

To address the aforementioned drawbacks of the discrete-time approach, research efforts have emerged to develop more effective continuous-time formulations for process scheduling. In contrast to the discrete-time approach, the continuous-time models introduce the key concept of event or variable time interval. The exact definition of event varies from one formulation to another, but essentially, they all correspond to a time instance in the continuous domain of the horizon. By associating the events to continuous variables which can take potentially any value in the time horizon, the activities or changes of system, for example, the start and the end of a task, are allowed to take place at any time in the horizon, which renders the capability of modeling the time most accurately. More importantly, by eliminating a major fraction of the inactive event-time interval assignments that is characteristic in the solution of a discrete-time model, a usually much smaller number of events or variable time intervals need to be introduced to model the production process compared to the number of fixed time intervals required by the discrete-time approach. Consequently, the continuous-time approach leads to mathematical programming problems of much smaller size, which, in many cases, requires less computational effort for their solution. On the other hand, however, because of the variable nature of the timings of the events, it becomes more challenging to model the scheduling process and the resulting mathematical models may exhibit more complicated structures compared to their discrete-time counterparts. The basic idea of the continuous-time approach is illustrated in part (b) of figure 3.

In the following sections, the various existing discrete-time and continuous-time models will be discussed. An extensive discussion of discrete-time versus continuous-time models for process scheduling can also be found in Floudas and Lin (2004).

1.2. Discrete-time models

Discrete-time scheduling formulations make use of the concept of discretization. The time horizon of interest is divided into a number of time intervals of uniform durations. The start/end of a task and other important events are associated with the boundaries of these time intervals. With such a common reference time grid for all operations competing for shared resources, such as equipment items, the various relationships in a scheduling problem can be formulated as constraints of relatively simple forms.

The earliest research contributions that employed this approach for scheduling problems were reported in the operations research literature. Bowman (1959) proposed the first mathematical formulation for the general jobshop scheduling problem. He introduced variables that were to take the values of zero or one to represent whether or not a product occupied a machine during a time period and formulated a (integer) linear programming problem. Manne (1960) presented another discrete-time model in which integer-value variables were defined to represent the time period at which a task was to begin and the resulted discrete linear programming problem involved fewer variables. Both work pointed out the computational challenges associated to the solution of such mixed integer programming models. It should be noted that these models focused on sequencing problems and didn't deal with the timing issue explicitly. Notable subsequent developments include the work by Pritsker, Watters, and Wolfe (1969) for resource-limited multiproject and jobshop scheduling.

More recently, the same approach was introduced to the chemical engineering community for the general process scheduling problem. Kondili, Pantelides, and Sargent (1988, 1993) proposed the first discrete-time formulation for the scheduling of general chemical processes based on the STN representation. In their formulation, the key decision variables, W_{ijt}, are introduced to account for the assignment of units to tasks. This set of binary variables determine whether or not a task (i) starts in unit (j) at the beginning of time interval (t) and the following allocation constraint is formulated to express the restriction that for any unit, at most one task can start at the beginning of each time interval.

$$\sum_{i \in I_j} W_{ijt} \leq 1, \quad \forall j \in J, t \in T, \tag{1}$$

where I_j is the set of tasks that can be performed in unit (j). Furthermore, if a task starts in a unit at a time interval, no other task can start in the same unit until this task is finished. This requirement is enforced as follows:

$$\sum_{i' \in I_j} \sum_{t'=t}^{t+\alpha_{ij}-1} W_{i'jt'} - 1 \le M(1 - W_{ijt}), \quad \forall j \in J, i \in I_j, t \in T, \tag{2}$$

where α_{ij} is the fixed processing time of task (i) in unit (j) and M is a sufficiently large positive number.

In addition to the binary variables, the following two sets of continuous variables are also important. B_{ijt} represents the amount of material which starts undergoing task (i) in unit (j) at time interval (t) (i.e., batch-size), and S_{st} determines the amount of material state (s) during time interval (t). The batch-size of a task is related to the corresponding assignment variable through the capacity constraints as follows:

$$W_{ijt} V_{ij}^{\text{Min}} \le B_{ijt} \le W_{ijt} V_{ij}^{\text{Max}}, \quad \forall i \in I, j \in J_i, t \in T, \tag{3}$$

where V_{ij}^{Min} and V_{ij}^{Max} are the minimum and maximum capacity of unit (j) for task (i), respectively. The mass balance can be expressed effectively by establishing the relationships between the inventory levels at two consecutive time intervals through the following constraint:

$$S_{st} = S_{s,t-1} + \sum_{i \in I_s^p} \rho_{is}^p \sum_{j \in J_i} B_{i,j,t-\alpha_{is}} - \sum_{i \in I_s^c} \rho_{is}^c \sum_{j \in J_i} B_{i,j,t} + R_{st} - D_{st}, \quad \forall s \in S, t \in T, \tag{4}$$

where I_s^p and I_s^c are the set of tasks that produce and consume state (s), respectively; ρ_{is}^p and ρ_{is}^c are the fractions of state (s) produced and consumed by task (i), respectively; J_i is the set of units suitable for task (i); α_{is} is the processing time for state (s) by task (i). R_{st} is the amount of state (s) received from external sources at time interval (t) and variable D_{st} represents the amount of state (s) delivered at time interval (t). The restriction on storage of a material state is then represented simply as upper bounds on the variables S_{st}:

$$0 \le S_{st} \le C_s, \quad \forall s \in S, t \in T, \tag{5}$$

where C_s is the storage capacity limit for state (s).

Other considerations, such as change-overs and limited utility resources, can also be incorporated readily in the discrete-time model (Kondili, Pantelides, and Sargent, 1988, 1993). There has been subsequent progresses following this discretization approach and discrete-time formulation and solution techniques have been developed for a variety of problems. Examples of related work include those presented by Pantelides (1993), Dedopoulos and Shah (1995), Pekny and Zentner (1993), Zentner et al. (1994), Bassett, Pekny, and Reklaitis (1996b) and Elkamel et al. (1997).

The main advantage of the discrete-time representation is that it provides a reference grid of time for all operations competing for shared resources, which renders the possibility of formulating the various constraints in the scheduling problem in a relatively simple manner and usually leads to well-structured mathematical programming problems. However, the discrete-time approach has two main limitations: the discrete approximation of time and the large size of resulting MILP problems. Due to the continuous nature of time and the concept of discretization, the discrete-time formulations are by definition only approximations of the actual problem. Furthermore, one of the key issues in this approach is the selection of the time interval duration, which always presents a tradeoff between the solution quality and the computational requirement. If a coarse discretization scheme is used, the problem size may be tractable, but there is an inevitable loss of model accuracy and it results by definition in suboptimal solutions. Furthermore, for operations with variable processing times such as continuous processes, which consume feeds and produce products continuously and in general can potentially run for a time period of any duration, the discrete-time approach only provides an approximate description of the actual process and the resulting schedule may deviate substantially from the true optimal solution. On the other hand, if a small time interval is used to achieve desirable degree of accuracy, the discrete-time approach inevitably leads to very large combinatorial problems that are difficult or even impossible to solve, especially for medium or large practical applications.

1.3. Continuous-time models

To address the aforementioned drawbacks of the discrete-time models, research efforts have been spent in the past decade on the development of more effective and efficient continuous-time approaches for process scheduling problems. For general network-represented processes, we classify the continuous-time models that have been presented in the literature into two groups. The first group of models define a set of events (or variable time slots, time points) that are used for all tasks and all units. We denote such formulations as "*global event based models*." The second group of models introduce event points on a unit basis, allowing tasks in different units associated with the same event point to take place at different times. We denote such models as "*unit-specific event based models*."

1.3.1. Global event based models

The time representation scheme employed in the global event based continuous-time scheduling models relies on the introduction of events or variable time slots that are universal for all units and the association of continuous variables to these events or time slots to determine their timings. The earliest work following this direction were presented by Zhang (1995), Zhang and Sargent (1996, 1998), Mockus and Reklaitis (1997a, 1999b), Mockus et al. (1997) and Schilling and Pantelides (1996). Recent developments include those presented by Castro, Barbosa-Póvoa, and Matos (2001), Majozi and Zhu (2001), Lee, Park, and Lee (2001), Burkard, Fortuna, and Hurkens (2002) and Wang and Guignard (2002).

Zhang (1995) and Zhang and Sargent (1996, 1998) developed the first continuous-time models for the scheduling of general network-represented processes. Their formulations employ either the STN or the RTN representation and can handle mixed production facilities involving both batch and continuous processes. One of the key variables in their formulations concerns the timings of events, T_k. This set of continuous variables are required to be monotonically increasing.

$$0 = T_1 < T_2 < \cdots < T_K \leq H, \tag{6}$$

where H is the time Horizon.

Then, based on the STN framework, two sets of binary variables can be defined to associate the tasks to the events. W_{ijk} determines whether or not task (i) starts at T_k in unit (j) and $X_{ijkk'}$ is activated if task (i) starts at T_k in unit (j) and completes at $T_{k'}$. With these variables, the allocation constraints can be written as follows to ensure that if a task starts in a unit at one event time, it finishes at exactly one later event time and that at each event time a unit can be occupied by at most one task.

$$W_{ijk} = \sum_{k' \geq k} X_{ijkk'}, \quad \forall i \in I, j \in J_i, k \in K, \tag{7}$$

$$\sum_{i \in I_j} \sum_{k \leq k' \leq k''} X_{ijkk''} \leq 1, \quad \forall j \in J, k' \in K. \tag{8}$$

Capacity constraints and mass balances are expressed in similar forms as those in the discrete-time models.

$$W_{ijk} V_{ij}^{\text{Min}} \leq B_{ijk} \leq W_{ijk} V_{ij}^{\text{Max}}, \quad \forall i \in I, j \in J_i, k \in K, \tag{9}$$

$$S_{sk} = S_{s,k-1} + \sum_{i \in I_s^p} \rho_{is}^p \sum_{j \in J_i} \sum_{k' \leq k} X_{ijk'k} B_{ijk'} - \sum_{i \in I_s^c} \rho_{is}^c \sum_{j \in J_i} W_{ijk} B_{ijk}, \quad \forall s \in S, k \in K. \tag{10}$$

The duration of a task, represented by variable t_{ijk}, is determined by the following timing constraint:

$$t_{ijk} = \sum_{k' > k} X_{ijkk'}(T_{k'} - T_k), \quad \forall i \in I, j \in J_i, k \in K. \tag{11}$$

Note that Constraints (10) and (11) involve bilinear products of binary and continuous variables. Exact linearization techniques (Glover, 1975; Floudas, 1995) can be applied to transform them into linear forms at the cost of introducing additional variables and constraints.

Mockus and Reklaitis (1999a) proposed an alternative definition of the task-event assignment variables. In their continuous-time formulations, which is called Non-Uniform Discrete-Time Model (NUDTM), two sets of binary variables, W_{ijk}^S and

W_{ijk}^F, are introduced to determine whether or not task (i) starts and finishes in unit (j) at the time of event (k), respectively. Due to this different definition, the allocation constraints are formulated in the following different form:

$$N_{jk} = N_{j,k-1} - \sum_{i \in I_j} W_{ijk}^S + \sum_{i \in I_j} W_{ijk}^F, \quad \forall j \in J, k \in K, \tag{12}$$

$$0 \le N_{jk} \le 1, \quad \forall j \in J, k \in K, \tag{13}$$

where N_{jk} represents the number of available unit (j) at time T_k and $N_{j0} = 1$.

Instead of the absolute times of the events, an alternative way to represent their timings is to define the duration between two events (i.e., the duration of time slots), ΔT_k, as the main timing variables, as suggested by Schilling and Pantelides (1996). The sum of the durations of all slots should be equal to the whole time horizon:

$$\sum_{k=1}^{K-1} \Delta T = H. \tag{14}$$

Schilling and Pantelides (1996) introduced binary variables $y_{ikk'}$ to determine whether or not task (i) starts at time (k) and is active over slot $(k') \ge (k)$. Based on the RTN framework, they formulated general resource balance and capacity constraints similar to the following:

$$R_{rk} = R_{r,k-1} + \sum_{i \in I_r} \left[\mu_{ri}^c N_{ik} + \nu_{ri}^c \xi_{ik} + \sum_{k'=1}^{k-1} \left(\mu_{ri}^p N_{ik'} + \nu_{ri}^p \xi_{ik'} \right) (y_{ik',k-1} - y_{ik',k}) \right], \quad \forall r \in R, k \in K, \tag{15}$$

$$R_r^{\min} \le R_{rk} \le R_r^{\max}, \quad \forall r \in R, k \in K, \tag{16}$$

where variable R_{rk} represents the amount of excess resource (r) during slot (k), N_{ik} is the number of instances of task (i) starting at the beginning of slot (k), and ξ_{ik} is the extent of all instances of task (i) starting at the beginning of slot (k) (e.g., the amount of material processed by the task). The duration of a task is now expressed as:

$$t_{ik} = \sum_{k' \ge k} y_{ikk'} \Delta T_{k'}, \quad \forall i \in I, k \in K. \tag{17}$$

This approach of using the slot durations rather than the absolute times of the events has the advantage that it may lead to tighter linearizations because of the smaller ranges of the slot durations (Schilling and Pantelides, 1996).

The formulations described above all lead to large scale MINLP problems. Many of the nonlinear terms can be linearized, but it usually requires the introduction of a large number of additional variables and constraints and thus increases the size of the resulting model. For certain classes of problems, for example, in the case of batch processes with simple forms of the objective function, all the nonlinearities can be eliminated and large MILP models are generated.

In addition to the alternative ways to represent variable timings and to define allocation and sequencing variables, the various global event based continuous-time models may also differ in the exact definition of events, the process representation framework, and the characteristics of the specific chemical process under investigation. For example, Castro, Barbosa-Póvoa, and Matos (2001) proposed an RTN based MILP formulation for the short-term scheduling of batch processes. Majozi and Zhu (2001) presented an MILP model for the short-term scheduling of batch processes based on a new process representation called State Sequence Network. They used time points to denote the use or production of states and introduced binary variables $y(s, p)$ associated with the usage of state (s) at time point (p). Their formulation leads to small MILP problems, but relies on the definition of effective states which are related to tasks and units. Lee, Park, and Lee (2001) reported an STN based MILP formulation for batch and continuous processes, which introduced three sets of binary variables to account for the start, process, and end events of each task. Burkard, Fortuna, and Hurkens (2002) developed an STN based MILP formulation for the makespan minimization problem for batch processes and discussed the choice of the objective function and additional constraints. Wang and Guignard (2002) presented an STN based MILP formulation for batch process scheduling problems, which proposed the definition of events associated with inventory changes to reduce the total number of events required to model a schedule. Maravelias and Grossmann (2003) proposed an STN based MILP model for the short-term scheduling of multipurpose batch processes which features the elimination of variables for task starting times, a set of tightening inequalities to improve the LP relaxation, and accounts for constraints on resources other than equipment.

The global event based continuous-time models can incorporate a wide variety of considerations in process scheduling, such as intermediate storage, change-over, batch and continuous operational modes, due dates, renewable resources, and various objective functions.

All the global event based continuous-time models discussed above use an *a priori* number of events/time slots/time points. As pointed out by Zhang and Sargent (1996), an important issue is the estimation and adjustment of this number. An underestimation may lead to suboptimal solutions or even infeasible problems, while an overestimation results in unnecessarily large problems, which increase even more the difficulty of the solution. Despite its significance, relatively little attention has been paid on this issue in the literature. Two exceptions are the work presented by Schilling (1997) and recently by Castro, Barbosa-Póvoa, and Matos (2001). They proposed an iterative procedure in which the model begins with a small number of events and then the number is gradually increased until no improvement can be achieved. However, as reported by Castro, Barbosa-Póvoa, and Matos (2001), in some cases, the solution may improve only after the addition of more than one event, which creates difficulty for the establishment of a stopping criterion that can guarantee the optimality of the solution. In a more recent work, Burkard, Fortuna, and Hurkens (2002) derived lower and upper bounds of the total number of batches

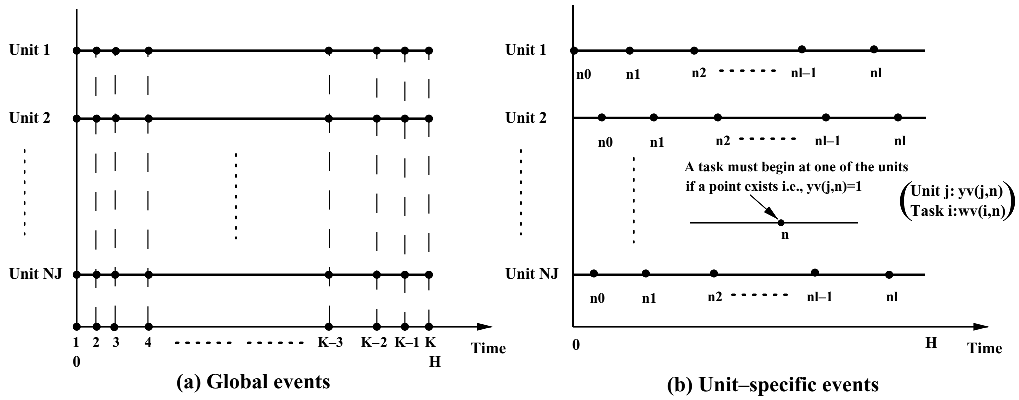


Figure 4. Global events and unit-specific events.

from simple aggregating mixed-integer models based on the minimum and maximum batch-sizes.

1.3.2. Unit-specific event based models

Ierapetritou and Floudas (1998a, 1998b, 2001), Ierapetritou, Hené, and Floudas (1999), and Lin and Floudas (2001) proposed a new approach to formulate continuous-time models for the short-term scheduling of general batch, semicontinuous, and continuous chemical processes. These formulations employ an 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 utilization of the unit. The main difference between this definition and those used in the global event based models described in the previous section is illustrated in figure 4. Each event point can be located at different positions along the time axis for different units, allowing different tasks to start at different time instances in different units for the same event point. 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 (compared to that in a global-event based model which considers the starting and the finishing of a task as two events), for the same scheduling problem, the number of event points required in this formulation is smaller than the number of events in the global event based models described in the previous section. Consequently, the number of binary variables is reduced substantially in this type of continuous-time formulations.

Based on the new event point concept, two sets of binary variables are defined as the key assignment variables. $wv(i, n)$ determines whether or not task (i) starts at event point (n); and $yv(j, n)$ determine whether or not unit (j) starts being utilized at event point (n). They are used in the following constraint to represent the restriction on allocation that in each unit and at each event point at most one of the tasks that can be performed in this unit should take place.

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

It should be emphasized that the number of event points is the same for all the units, however, the timing of each event point can be different in different units. Also note that if a task can be performed in multiple units, it is split into multiple tasks with each one performed in a different unit. This will increase the number of $wv(i, n)$ binary variables and in the worst case where every task can take place in every unit, the total number of tasks after splitting is equal to the number of the original tasks times the number of units.

Based on the STN process representation, the main continuous variables in this formulation include: $B(i, j, n)$, the batch-size of task (i) in unit (j) at event point (n); $ST(s, n)$ and $D(s, n)$, the amounts of state (s) stored and delivered at event point (n) respectively. Using these variables, the unit capacity, material balance and storage constraints can be formulated similarly to those in the discrete-time models and global event based continuous-time models.

$$V_{ij}^{\min} wv(i, n) \leq B(i, j, n) \leq V_{ij}^{\max} wv(i, n), \quad \forall i \in I, j \in J_i, n \in N, \tag{19}$$

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

$$0 \leq ST(s, n) \leq C_s, \quad \forall s \in S, n \in N. \tag{21}$$

Note that the above constraint (20) is written for batch tasks, while similar constraints can be written for continuous tasks which takes into account the different nature of the continuous operation mode (see Ierapetritou and Floudas, 1998b). It should be pointed out that the amount of state (s) at an event point (n), represented by $st(s, n)$, generally does not correspond to one well-defined time instance or time period due to the fact that the state can be consumed or produced by different tasks that take place in different units with different time axis. When there is a storage limit for the state, an upper bound can be imposed on the $ST(s, n)$ variable as an approximate way to model the storage restrictions, as represented by Constraint (21). A rigorous way is to introduce storage tasks and storage units with certain capacity ranges (see Ierapetritou and Floudas, 1998b; Lin and Floudas, 2001).

To determine the task timings, we resort to another two sets of continuous variables. $T^s(i, j, n)$ and $T^f(i, j, n)$ represent the start and end times of task (i) in unit (j) at event point (n). The duration of a task is represented as follows:

$$T^f(i, j, n) = T^s(i, j, n) + \alpha_{ij} wv(i, n) + \beta_{ij} B(i, j, n), \quad \forall i \in I, j \in J_i, n \in N. \tag{22}$$

This linear expression of variable processing time as a function of the batch-size is able to model a wide variety of processes. For example, in the case of fixed processing times, α_{ij} corresponds to the processing time of a task and β_{ij} is zero. While for tasks operating in the continuous mode, α_{ij} is zero and β_{ij} is the inverse of the processing rate.

In addition to the above duration constraints, the various relationships among the timings of tasks are formulated through the following three sets of sequence constraints, which forms a very important component characteristic of this formulation.

$$T^s(i, j, n+1) \geq T^f(i, j, n), \quad \forall i \in I, j \in J_i, n \in N, n \neq n_{\text{last}}; \tag{23}$$

$$T^s(i, j, n+1) \geq T^f(i', j, n) + \tau_{ji'i} wv(i', n) - H(1 - wv(i', n)),$$
$$\forall j \in J, i \in I_j, i' \in I_j, \quad i \neq i', n \in N, n \neq n_{\text{last}}; \tag{24}$$

$$T^s(i, j, n+1) \geq T^f(i', j', n) - H(1 - wv(i', n)),$$
$$\forall j, j' \in J, i \in I_j, i' \in I_{j'}, i \neq i', n \in N, n \neq n_{\text{last}}. \tag{25}$$

Constraints (23) are written for the same task in the same unit at two consecutive event points. They state that task (i) starting in unit (j) at event point $(n+1)$ should start after the end of the same task performed in the same unit which has already started at the previous event point (n). Constraints (24) establish the relationships between the timings of different tasks in the same unit at two consecutive event points. If $wv(i', n) = 1$ which means that task (i') takes place in unit (j) at event point (n), then the last term of constraint (24) becomes zero forcing task (i) in unit (j) at event point $(n+1)$ to start after the ending time of task (i') in unit (j) at event point (n) plus the required clean-up time; otherwise the right hand side of constraint (24) becomes negative and the constraint is trivially satisfied. It should be pointed out that this constraint actually imposes a lower bound not only on the starting time of task (i) at event point $(n+1)$ but also on the starting times of task (i) at the subsequent event points $(n+2)$, $(n+3)$, etc. because of the monotonically increasing relationships among the timings of the same task in the same unit at consecutive event points established by Constraint (23). In other words, if two tasks take place in the same unit consecutively, but at two event points with an idle event point in between, the requirement on their timings is also enforced. The last set of constraints (25) are written for different tasks (i, i') that are performed in different units (j, j') but take place consecutively according to the production recipe. If task (i') takes place in unit (j') at event point (n) (i.e., $wv(i', n) = 1$), then we have $T^s(i, j, n+1) \geq T^f(i', j', n)$ and hence task (i) in unit (j) has to start after the end of task (i') in unit (j'). Similar to Constraint (24), this constraint also establishes the relationships between tasks that are assigned to non-consecutive event points.

Note that the sequencing and timing relationships are modeled very efficiently with the above duration and sequencing constraints. No additional variables, such as $X_{ijkk'}$ in the global event based models discussed in the previous section, are required. This gives rise to the further significant reduction of the number of binary variables and the overall size of the resulting MILP models.

By linking demands to event points based on the relative timing of the due dates and the processing recipe, intermediate due dates can also be incorporated through the following constraints:

$$D(s, n) + SL(s, n) = da_{sn}, \quad \forall s \in S, n \in N, \tag{26}$$

$$T^s(i, j, n) \leq dd_{sn}, \quad \forall s \in S, i \in I_s^p, j \in J_i, \tag{27}$$

where da_{sn} and dd_{sn} are the amount and due date of the demand for state (s) at event point (n). Note that the tasks that produce the involved state are considered in Constraint (27) and it is guaranteed that the product delivered at the event point is produced before the due date of the demand. Slack variables $SL(s, n)$ are introduced to provide more flexibility in handling partial fulfillment of demands.

Furthermore, Janak, Lin, and Floudas (2004) have recently presented an enhanced formulation which extends the work of Floudas and coworkers (Ierapetritou and Floudas, 1998a, 1998b, 2001; Ierapetritou, Hené, and Floudas, 1999; Lin and Floudas, 2001), 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), fixed intermediate storage (FIS), no intermediate storage (NIS), and zero-wait (ZW) conditions. In their 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 which indicates whether or not a task starts at each event point and another which indicates whether or not a task ends at each event point. A continuous variable 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 amounts of states will be consistent and specified limits on both can be enforced. For instance, constraints are written to define the amount of a utility used to undertake a task, to keep track of the amount of utility available at each event point, and also to relate the duration and timing of a utility to the timing of the processing tasks which utilize that utility. In addition, constraints are included which govern the batch-size of a storage task, relate the storage task to processing tasks through material balances, and also relate the duration and timing of the storage task to the timing of the processing tasks which produce or consume the state being stored through the storage task. Furthermore, the allocation constraints are expanded in order to relate the above mentioned binary and continuous assignment 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 are added to relate the batch sizes for tasks which start and finish at the same or consecutive event points so that tasks which extend over more than one event point have consistent batch sizes at each event point. Moreover, the duration constraints for a processing task are expanded 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, a set of order satifaction constraints are included to allow orders for products to be due at intermediate due dates. These constraints force an order for a state to be met by a certain due date and in a certain amount. Finally, a set of tightening constraints is introduced to help improve the relaxed LP solution.

Maravelias and Grossmann (2003) have also recently introduced an approach for the short-term scheduling of multipurpose batch plants including resource constraints and mixed storage policies, however, their formulation uses a global event based model. When compared to the unit-specific event based model presented by Janak, Lin, and Floudas

(2004), their formulation always employs at least one more event point to determine the same objective function value and as a result, involves more binary and continuous variables. This is because global event based formulations require an event point for the start and end of each task whereas unit-specific event based formulations only require an event point for the start of a task. This means there will be an extra event point to account for the ending of the last task in a global event based model. Computational results presented in Janak, Lin, and Floudas (2004) demonstrate that unit-specific event based models are better suited for short-term scheduling problems where the minimization of the makespan is the objective and also for the case when a larger number of event points are considered in the problem, regardless of the objective function used.

It is important to note that the unit-specific event based formulation outlined above also requires the determination of an *a priori* number of event points. The general procedure is to start with a small number and iteratively increase it until no improvement of the objective function can be achieved (Ierapetritou and Floudas, 1998a). Under certain circumstances, special structures of the problem can be exploited to develop more efficient ways of determining the optimal number of event points. The possibility that it requires the addition of more than one event point to improve the solution for this formulation is much smaller than that for the global event based models, due to the more efficient utilization of event points and the smaller number of event points required to model a process.

The unit-specific event based continuous-time formulations described above lead to MILP models of smaller size mainly in terms of the number of binary variables, compared to the discrete-time models and the other continuous-time models. This can be shown by comparing the different approaches applied to a small example. This example is taken from literature (Zhang, 1995; Ierapetritou and Floudas, 1998a, 2001; Castro, Barbosa-Póvoa, and Matos, 2001) and involves variable processing times. The process considered has been extensively studied in the literature and the corresponding STN is shown in figure 5. As shown in Table 1, the discrete-time approach is an approximation of the actual process and by definition leads to suboptimal solutions that can deviate from the optimal solution substantially. Furthermore, the size of the resulting model and the required solution time explode exponentially as the number of time intervals increases to improve the degree of accuracy. When the number of time intervals is increased from 8 (corresponding to a discretization interval of 1 hr) to 32 (corresponding to a discretization interval of 0.25 hr), the discrete-time formulation attains better approximation and better solution, which improves from 620.2 to 1195.3. However, this is still suboptimal compared to the best solution of 1498.2 obtained through a unit-specific event based continuous-time formulation. Furthermore, the size of the resulting model grows significantly, reflected by the increase of the number of binary variables (from 38 to 591) and the required solution time (from less than a sec to over 100,000 sec on a HP-C160 work station without solving to optimality). In contrast, continuous-time approaches lead to more accurate models of smaller sizes. Two global event based formulations are shown in Table 1, representing the development of this type of continuous-time models. The model proposed by Zhang (1995) led to an MILP model with 147 binary variables when

Table 1
Comparison between discrete-time models and global event based, unit-specific event based continuous-time models.

Model	Discrete-time models			Continuous-time models: Global event based: Zhang (1995)	Continuous-time models: Global event based: Castro, Barbosa-Póvoa, and Matos (2001)	Continuous-time models: Unit-specific event based: Ierapetritou and Floudas (1998a, 2001)
Events/time intervals	8	16	32	7	5	5
Binary var.	**38**	**171**	**591**	**147**	**80**	**40**
Continuous var.	743	2386	8590	497	226	260
Constraints	1567	5135	18415	741	297	374
Obj. (profit)	**620.2**	**940.5**	**1195.3**	**1497.7**	**1480.06**	**1498.2**
Nodes	15	5123	~500,000	9575	60	51
CPU time	0.29s[a]	58s[a]	~100,000s[a]	1027.5s[b]	0.32s[c]	0.28s[a]

[a]HP-C160
[b]Sun Sparc 10/41
[c]Pentium III 450-MHz.

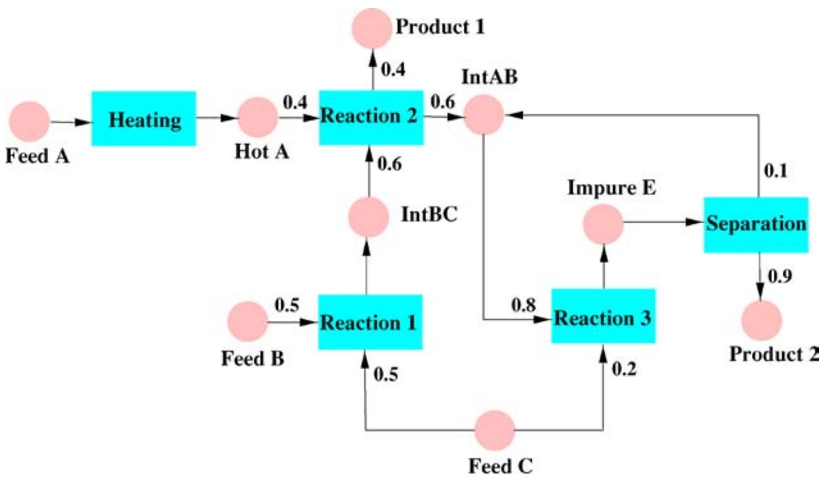


Figure 5. State-Task Network of the process involved in the example.

using 7 events, which was solved in reasonable time (1027.5 CPU sec on a Sun Sparc 10/41 work station) and achieved a much better objective value of 1497.7, compared to the discrete-time model. A more recent model proposed by Castro, Barbosa-Póvoa, and Matos (2001) required 80 binary variables with the use of 5 events and obtained an objective value of 1480.06 in 0.32 CPU sec on a Pentium III 450-MHz machine. The unit-specific event based approach proposed by Ierapetritou and Floudas (1998a, 2001) used 5 event points and further reduced the size of the resulting model to 40 binary variables. The model led to an objective value of 1498.2 and was solved in 0.28 CPU sec on a HP-C160 work station. This example may be exaggerating to some degree, however, the main differences among the various approaches are illustrated clearly.

2. Solution of MILP models

One of the most important issues in the application of mixed-integer programming techniques to the process scheduling problems lies in the computational efficiency for the solution of the resulting MILP problems, since realistic problems often lead to large scale models. To accelerate the solution process, several classes of approaches have been proposed to exploit special structures of specific problems.

2.1. Reformulation

After constraints have been formulated, in some cases, they can be re-written in alternative forms which are better from a computational point of view. The aim is to generate models with tighter integrality gaps, reduced number of binary variables, or special structures facilitating the solution. For example, Sahinidis and Grossmann (1991b) proposed the following disaggregated form of the allocation constraint (2) in the discrete-time model described in the previous section.

$$W_{ijt} + W_{i'jt'} \leq 1, \quad \forall j \in J, t \in T, i, i' \in I_j, t' = t-1, \ldots, t - \alpha_{ij} + 1. \tag{28}$$

Note that the large positive number M in constraint (2) is eliminated and the reformulated constraints lead to smaller integrality gaps at the cost of a much larger number of constraints. Shah, Pantelides, and Sargent (1993) suggested another alternative of the allocation constraint in an aggregated form:

$$\sum_{i \in I_j} \sum_{t'=t}^{t-\alpha_{ij}+1} W_{ijt'} \leq 1, \quad \forall j \in J, t \in T. \tag{29}$$

This reformulation not only reduces the integrality gap, but also requires only a small number of constraints.

As another example, Orçun, Altinel, and Hortaçsu (1999) applied Reformulation Linearization Technique (RLT) (Sherali and Adams, 1994) to an MILP scheduling model to reduce the integrality gap. The approach consists of a reformulation phase and a linearization phase. In the reformulation phase, selected constraints and binary variables are multiplied and the resulting new constraints are added to the original problem. Then the nonlinear model is linearized during the linearization phase.

2.2. Addition of cut constraints

It is known that the introduction of additional constraints into an MILP problem may cut off infeasible solutions at an early stage of the branch and bound searching process and thus reduce the solution time. For process scheduling problems, such effective cut constraints can be generated by exploiting special structures of the scheduling problem or existing insights on the physical problem. For example, Dedopoulos and Shah (1995) proposed a number of additional constraints which establish explicit relationships among the various binary variables in a discrete-time MILP model for the production and maintenance scheduling of multipurpose plants. Elkamel and Al-Enezi (1998) presented three sets of inequality constraints for a discrete-time model based on timing restrictions existing in batch processes with fixed processing times and suggested strategies to incorporate them in a selective manner. Yee and Shah (1998) developed cut constraints to force sequence dependent or sequence independent changeover tasks to take place in the LP relaxation solution, which leads to LP relaxations closer to the original MILP problem. Lin et al. (2002) introduced additional constraints to impose lower bounds on the total number of batches based on related product demands and unit capacities. Furthermore, additional timing constraints were used to reduce the solution space and to improve the quality of feasible integer solutions. Burkard, Fortuna, and Hurkens (2002) presented a number of additional constraints, including similar ones on the lower bounds of the number of batches and the total batch-size according to demands for each final product and intermediate material.

2.3. Use of heuristics

Another strategy to expedite the solution process relies on the use of heuristics, which may hold true in some cases but cannot guarantee optimality, to simplify the problem. Pinto and Grossmann (1995) proposed the preordering of orders based on their due dates and processing times and introduced logical relationships to impose the relative sequence of the orders. The preordering constraints increased the problem size and may lead to suboptimal solution, but they reduced the search space and thus the solution time, especially for problems involving a large number of orders. Cerdá, Henning, and Grossmann (1997) presented a number of heuristic rules to prune the set of feasible predecessors for each order in a unit. The set of predecessors for an order is reduced by selecting only those that are likely to lead to a good schedule. Although optimality could not be guaranteed, the use of heuristics reduced the solution time and accelerated the finding of good intermediate integer solutions. Blömer and Günther (2000) suggested a heuristic two-stage solution procedure for a discrete-time model with the objective of makespan minimization. An initial solution was obtained with a reduced number of feasible start-up time periods and then it was improved by left-shifting. The required computational effort was reduced at the cost of obtaining only suboptimal solutions.

2.4. Decomposition

The most widely employed strategy to overcome the computational difficulty for the solution of large MILP models is based on the idea of decomposition. The decomposition approach divides a large and complex problem, which may be computationally expensive or even intractable when formulated and solved directly as a single MILP model, to smaller subproblems, which can be solved much more efficiently. There have been a wide variety of decomposition approaches proposed in the literature. In addition to decomposition techniques developed for general forms of MILP problems, various approaches that exploit the characteristics of specific process scheduling problems have also been proposed. In most cases, the decomposition approaches only lead to suboptimal solutions, however, they substantially reduce the problem complexity and the solution time, which renders the possibility of apply MILP based techniques to large real-world problems.

Pinto and Grossmann (1995) proposed a decomposition scheme for large scheduling problems arising from multistage batch plants with the objective of minimizing earliness. First, an MILP model that minimizes the total in process time was solved to determine the assignments of orders to units. Then, the model is solved to minimize earliness with the assignment binary variables fixed. Wilkinson, Shah, and Pantelides (1995) presented an alternative method, aggregate modeling, for tackling the computational challenge of difficult scheduling problems in multipurpose plants. An aggregate model was generated from a detailed discrete-time formulation and was essentially a relaxation of the original model, but it could be solved in considerably less computational time and gave a tight upper bound on the solution to the original problem. Bassett, Pekny, and Reklaitis (1996b) discussed a number of time-based decomposition approaches for the discrete-time MILP formulation. The first approach used a hierarchy consisting of a planning level and a scheduling level, which are solved iteratively followed by various techniques to remove infeasibilities. A second approach, called a reverse rolling window approach, utilized a hybrid planning/scheduling formulation. In this method, only a small section of the horizon is determined in full details at each step and a sequence of such problems with reduced combinatorial complexity are solved in the reverse order of the time windows. Elkamel et al. (1997) developed another decomposition algorithm consisting of two basic components. First, in spatial decomposition, the units in the plant are grouped into different subsets capable of performing different sets of tasks and the product orders are then decomposed according to the unit groups. Secondly, in temporal decomposition, the product orders are arranged into groups based on their due dates and then scheduled sequentially along the time axis. Harjunkoski and Grossmann (2001) presented a decomposition strategy for the scheduling of a steel plant production. First, the customer orders are partitioned into families with similar properties and each product family is then further disaggregated into groups. Next, each group is scheduled independently. Finally, an LP/MILP model is used to properly aggregate and improve the overall schedule. Lin et al. (2002) considered the medium-range scheduling problem of a multiproduct batch plant which involves a relatively long time horizon. Using a rolling horizon approach, the full scale problem in the whole scheduling period is decomposed into a series of smaller short-term scheduling subproblems in successive sub-horizons, which are connected through material and unit availabilities. A two-level MILP formulation is proposed to determine the current short sub-horizon and the products to be included, which takes into account demand distribution and unit utilization and imposes limits on the complexity of the resulting short-term scheduling problem. Then a continuous-time MILP based short-term scheduling model incorporating intermediate due dates is applied to determine the detailed schedule in the current sub-horizon. The decomposition model and the short-term scheduling model are employed iteratively until the schedules for the whole period under consideration are generated.

2.5. Intervention of the branch and bound solution procedure

The last main strategy to improve the computation efficiency is through the intervention of the branch and bound search process. For instance, Shah, Pantelides, and Sargent (1993) developed ways to reduce the size of the relaxed LP and perform post analysis of the LP solution at each node of the branch and bound tree. Dedopoulos and Shah (1995) proposed techniques to fix variables to values implied during the branch and bound procedure. Schilling and Pantelides (1996) designed a special branch and bound algorithm which branches on both the binary variables and the continuous τ_k variables which represent the lengths of time intervals in a global event based continuous-time formulation. Yi et al. (2000) presented special branching priorities based on manual scheduling practice. Higher priorities are given to the binary variables related to major products, which have higher demands or the scheduler pays special attention to, downstream tasks, and earlier times.

3. Computational experience and applications

The significant advances in the modeling and solution of scheduling formulations have led to their application to large and complex problems. Table 2 summarizes some of the largest MILP models that have been reported in the literature.

The MILP based scheduling approaches have been employed to a wide variety of real-world problems. Some examples of notable applications are presented below. Shah, Pantelides, and Sargent (1993) described a case study on the scheduling of a hydrolubes plant with adapted industrial data using the STN based discrete-time approach. Wilkinson et al. (1996) addressed a large scale production and distribution scheduling problem in which three multipurpose production facilities in different countries supply a large portfolio of fast moving consumer goods to the European market. They proposed a detailed formulation that considered all three plants simultaneously. Due to the very large size of the resulting problem, they generated an approximate formulation by aggregating constraints, whose solution gave a tight upper bound on the production capacity and facilitated the decomposition of the original problem into small sub-problems, each involving a single plant.

Table 2
Examples of largest MILP scheduling models reported in the literature.

Work	Process feature	Time representation	Bin. var., cont. var., constraints	Solution approach	CPU time
Shah, Pantelides, and Sargent (1993)	STN, 12 states/ 6 tasks/6 units	Discrete	2316,3855,3376	Modified branch and bound	48 min[a]
Pinto and Grossmann (1995)	sequential, 50 orders/ 5 stages/25 units	Continuous slot-based	1050,47917,48843	Decomposition	4.2 hr[b]
Schilling and Pantelides (1996b)	RTN, continuous 15 products/ 11 units	Continuous	1042,2746,4981	Modified branch and bound	57 min[c]
Zhang and Sargent (1998)	RTN, continuous 15 products/ 11 units	Continuous	1318,3237,4801	CPLEX	18 min[d]
Ierapetritou and Floudas (1998b)	STN, batch & continuous 28 products/13 units	Continuous	2375,29384,51000	GAMS/CPLEX MINOPT/CPLEX	2.7 hr[e]

[a]SUN SparcStation IPX
[b]HP 9000-730
[c]6 parallel processors
[d]Sun-Sparc10/41
[e]HP-C160.

Zhang (1995), Schilling and Pantelides (1996b), and Ierapetritou and Floudas (1998b) considered the scheduling of an industrial fast-moving consumer goods manufacturing plant involving batch and continuous processes. Continuous-time formulations were proposed using either global events (Zhang, 1995; Schilling and Pantelides, 1996b) or unit-specific events (Ierapetritou and Floudas, 1998b). Ierapetritou, Hené, and Floudas (1999) extended the continuous-time formulation in Ierapetritou and Floudas (1998a, 1998b) to deal with intermediate due dates and addressed a variety of problems, including the short-term scheduling of a single-stage multiproduct facility with multiple semicontinuous processors. Méndez and Cerdá (2000a) addressed the short-term scheduling of a two-stage multiproduct batch plant which delivered intermediate products to nearby end-product facilities and proposed a continuous-time MILP model. Georgiadis, Papageorgiou, and Macchietto (2000) considered the short-term cleaning scheduling in a special class of heat exchanger networks involving decaying equipment performance due to milk fouling. Discrete-time approaches were employed to formulate an MINLP model incorporating general fouling profiles, which is then linearized and solved as an MILP problem. Yi et al. (2000) studied the production scheduling of a polybutene process featuring the requirement of product quality check in intermediate storage tanks and developed a discrete-time MILP model.

Glismann and Gruhn (2001) developed an approach to integrate the short-term scheduling of multiproduct blending facilities and nonlinear recipe optimization. An

RTN based discrete-time MILP model was formulated for the scheduling problem. Harjunkoski and Grossmann (2001) presented a decomposition algorithm for the short-term scheduling of large scheduling problems in the steel making industry. Lamba and Karimi (2002b) developed a two-step decomposition algorithm for the short-term scheduling of a single-stage multiproduct facility with multiple semicontinuous production lines. The algorithm is based on item combinations and is applied to an industrial problem from a detergent plant. Castro, Matos, and Barbosa-Póvoa (2002) addressed the scheduling of a batch digester cooking system of an industrial acid sulphite pulp mill constrained by steam availability. A discrete time RTN based model featuring the most relevant steam-sharing alternatives was developed and the required process data were obtained with a dynamic model of the heating system. Lin et al. (2002) presented a systematic framework for the medium-range production scheduling of a large industrial polymer multiproduct plant, which uses a rolling horizon decomposition approach coupled with the unit-specific event based continuous-time scheduling formulation.

4. Conclusions and perspectives

In the previous sections, we have presented an overview of the developments of mixed-integer linear programming (MILP) based approaches for the scheduling of chemical processing systems in the past decade. It is apparent that significant advances have been achieved on the following important frontiers: (i) development of mathematical formulations for the effective modeling of a wide variety of chemical processes; (ii) development of algorithms for the efficient solution of difficult MILP models; and (iii) increasing application of the formal MILP optimization framework to real scheduling problems in process and related industries. However, further research work is needed to address classes of important large-scale industrial applications. More specifically, future research efforts should be directed to address the following important issues.

4.1. Reduction of integrality gaps

Further efforts are required to reduce and even close the integrality gap for medium and large scale scheduling applications. This can be achieved through the development of better mathematical models with tighter relaxations and/or more effective algorithms that exploit characteristics of the formulation or actual problem.

4.2. Medium-term scheduling

Most work presented in the literature so far has been dedicated to short-term scheduling or the scheduling of cyclic operations. Much less attention has been paid to medium-term scheduling which involves a relatively long time horizon and leads to very large scale problems. Dimitriadis, Shah, and Pantelides (1997) and Lin et al. (2002) reported two of the very few efforts dedicated to medium term scheduling. More work remains to be done to develop effective modeling and solution frameworks for this class of problems.

4.3. Multisite production and distribution scheduling

This is another promising candidate subject for future research. Integrated modeling of production scheduling at manufacturing sites and distribution scheduling as well as solving the resulted model of large size will be two major challenges.

4.4. Reactive scheduling

Due to the ubiquitous presence of unpredictable disturbances in the chemical processing environment, for example, uncertainty in processing times, prices, changes in product demands, and equipment failure/breakdown, it is of paramount importance to be able to adjust the current schedule upon realization of uncertain parameters or occurrence of unexpected events, which is called reactive scheduling. Most of the existing reactive scheduling approaches uses heuristics, such as time shifting, to update the schedule. New rigorous MILP based approaches are needed to address reactive scheduling.

4.5. Novel and more complex applications

There remain a variety of manufacturing systems in the chemical processing or related industries for which scheduling is still carried out based on heuristic or even manual approaches. It is expected that the introduction of MILP based scheduling technologies will bring great benefits, while the modeling and solution for the scheduling of these processes, especially those involving complex structures, will require additional efforts. For example, the scheduling of manufacturing operations in the semiconductor industry should be found promising, which features the presence of multiple reentrant flows and an extremely large number of processing tasks.

Notation

α_{ij} parameter, fixed processing time of task (i) in unit (j)

α_{is} parameter, processing time for state (s) by task (i)

β_{ij} parameter, linear coefficient of the variable term of the processing time of task (i) in unit (j)

ΔT_k continuous variable, the duration between event (k) and $(k+1)$ or time slot (k)

μ^c_{ri}, μ^p_{ri}, parameters, the amount of resource (r) consumed and produced by each instance of task (i), respectively

ν_{ri}^{c}, ν_{ri}^{p}, parameters, the amount of resource (r) consumed and produced per unit of the extent of task (i), respectively

ξ_{ik} continuous variable, the extent of all instances of task (i) starting at the beginning of slot (k)

ρ_{is}^{p}, ρ_{is}^{c} parameters, fractions of state (s) produced and consumed by task (i), respectively

$\tau_{ji'i}$ parameter, duration of the cleaning operation required for unit (j) to switch from task (i') to (i)

τ_k continuous variable, durations of time slot (k)

B_{ijk} continuous variable, amount of material which starts undergoing task (i) in unit (j) at event (k)

B_{ijt} continuous variable, amount of material which starts undergoing task (i) in unit (j) at time interval (t)

$B(i, j, n)$ continuous variable, amount of material which starts undergoing task (i) in unit (j) at event point (n)

C_s parameter, storage capacity limit for state (s)

D_{st} continuous variable, amount of state (s) delivered at time interval (t)

da_{sn} parameter, amount of the demand for state (s) at event point (n)

dd_{sn} parameter, due date of the demand for state (s) at event point (n)

$D(s, n)$ continuous variable, amounts of state (s) delivered at event point (n)

H parameter, time Horizon

i, i' indices, tasks

I set of all tasks

I_j set of tasks that can be performed in unit (j)

I_r set of tasks that consume or produce resource (r)

I_s^p, I_s^c sets of tasks that produce and consume state (s), respectively

j, j' index, units

J set of all units

J_i set of units suitable for task or order (i)

k, k', k'' indices, events or time slots

K set of events or time slots

M parameter, a sufficiently large positive number

m_i parameter, the lower bound on the total number of batches of task (i)

n index, event point

N set of event points

N_{ik} integer variable, the number of instances of task (i) starting at the beginning of slot (k)

N_{jk} (binary) variable, the number of available unit (j) at time T_k

n_{last} index, the last event point

p index, time point

r index, resource

R set of resources

$R_r^{\min}$, $R_r^{\max}$ parameters, the lower and upper bounds on the excess amount of resource (r)

R_{rk} continuous variable, the amount of excess resource (r) during slot (k)

R_{st} parameter, amount of state (s) received from external sources at time interval (t)

s index, material state

S set of material states

S_{sk} continuous variable, amount of material state (s) stored between event (k) and $(k+1)$ or during slot (k)

S_{st} continuous variable, amount of material state (s) stored during time interval (t)

$SL(s, n)$ continuous variable, difference between the amount of the demand for state (s) and the amount of state (s) delivered at event point (n)

$ST(s, n)$ continuous variable, amount of state (s) stored at event point (n)

t, t' indices, time intervals

T set of time intervals

t_{ijk} continuous variable, duration of task (i) which starts at T_k in unit (j)

t_{ik} continuous variable, duration of task (i) which starts at the beginning of slot (k)

T_k continuous variable, timing of event (k)

$T^s(i, j, n)$, $T^f(i, j, n)$ continuous variables, starting and ending times of task (i) in unit (j) at event point (n), respectively

$X_{ijkk'}$ binary variable, whether or not task (i) starts at T_k in unit (j) and completes at $T_{k'}$

$y_{ikk'}$ binary variable, whether or not task (i) starts at time (k) and is active over slot $(k') \geq (k)$

$y(s, p)$ binary variable, whether or not state (s) is used at time point (p)

$yv(j, n)$ binary variable, whether or not unit (j) starts being utilized at event point (n)

$V_{ij}^{\text{Min}}, V_{ij}^{\text{Max}}$ parameters, minimum and maximum capacity of unit (j) for task (i), respectively

W_{ijt} binary variable, whether or not task (i) starts in unit (j) at the beginning of time interval (t)

W_{ijk} binary variable, whether or not task (i) starts at T_k in unit (j)

W_{ijk}^{S}, W_{ijk}^{F} binary variables, whether or not task (i) starts and finishes in unit (j) at event (k), respectively

$wv(i, n)$ binary variable, whether or not task (i) starts at event point (n)

Acknowledgments

The authors gratefully acknowledge support from the National Science Foundation, and Atofina Chemicals, Inc.

References

Applequist, G., O. Samikoglu, J. Pekny, and G. Reklaitis. (1997). "Issues in the Use, Design and Evolution of Process Scheduling and Planning Systems." *ISA Transactions* 36, 81–121.

Bassett, M.H., P. Dave, F.J.D. III, G.K. Kudva, J.F. Pekny, G.V. Reklaitis, S. Subrahmanyam, D.L. Miller, and M.G. Zentner. (1996a). "Perspectives on Model Based Integration of Process Operations." *Computers & Chemical Engineering* 20, 821–844.

Bassett, M.H., J.F. Pekny, and G.V. Reklaitis. (1996b). "Decomposition Techniques for the Solution of Large-Scale Scheduling Problems." *AIChE Journal* 42, 3373–3387.

Blömer, F. and H.-O. Günther. (2000). "LP-Based Heuristics for Scheduling Chemical Batch Processes." *International Journal of Production Research* 38, 1029–1051.

Bok, J. and S. Park. (1998). "Continuous-Time Modeling for Short-Term Scheduling of Multipurpose Pipeless Plants." *Industrial & Engineering Chemistry Research* 37, 3652–3659.

Bowman, E.H. (1959). "The Schedule-Sequencing Problem." *Operations Research* 7, 621–624.

Burkard, R.E., T. Fortuna, and C.A.J. Hurkens. (2002). "Makespan Minimization for Chemical Batch Processes Using Non-Uniform Time Grids." *Computers & Chemical Engineering* 26, 1321–1332.

Castro, P., A.P.F.D. Barbosa-Póvoa, and H. Matos. (2001). "An Improved RTN Continuous-Time Formulation for the Short-Term Scheduling of Multipurpose Batch Plants." *Industrial & Engineering Chemistry Research* 40, 2059–2068.

Castro, P., H. Matos, and A.P.F.D. Barbosa-Póvoa. (2002). "Dynamic Modeling and Scheduling of an Industrial Batch System." *Computers & Chemical Engineering* 26, 671–686.

Cerdá, J., G.P. Henning, and I.E. Grossmann. (1997). "A Mixed-Integer Linear Programming Model for Short-Term Scheduling of Single-Stage Multiproduct Batch Plants with Parallel Lines." *Industrial & Engineering Chemistry Research* 36, 1695–1707.

Dedopoulos, I.T. and N. Shah. (1995). "Optimal Short-Term Scheduling of Maintenance and Production for Multipurpose Plants." *Industrial & Engineering Chemistry Research* 34, 192–201.

Dimitriadis, A.D., N. Shah, and C.C. Pantelides. (1997). "RTN-Based Rolling Horizon Algorithms for Medium Term Scheduling of Multipurpose Plants." *Computers & Chemical Engineering* 21, S1061–S1066.

Elkamel, A. and G. Al-Enezi. (1998). "Structured Valid Inequalities and Separation in Optimal Scheduling of the Resource-Constrained Batch Chemical Plant." *Mathematical Engineering in Industry* 6, 291–318.

Elkamel, A., M. Zentner, J.F. Pekny, and G.V. Reklaitis. (1997). "A Decomposition Heuristic for Scheduling the General Batch Chemical Plant." *Engineering Optimization* 28, 299–330.

Floudas, C.A. (1995). *Nonlinear and Mixed-Integer Optimization*. Oxford University Press.

Floudas, C.A. and X. Lin. (2004). "Continuous-Time versus Discrete-Time Approaches for Scheduling of Chemical Processes: A Review." *Computers & Chemical Engineering* 28, 2109–2129.

Garey, M.R. and D.R. Johnson. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: W. H. Freeman.

Georgiadis, M.C., L.G. Papageorgiou, and S. Macchietto. (2000). "Optimal Cleaning Policies in Heat Exchanger Networks under Rapid Fouling." *Industrial & Engineering Chemistry Research* 39, 441–454.

Glismann, K. and G. Gruhn. (2001). "Short-Term Scheduling and Recipe Optimization of Blending Processes." *Computers & Chemical Engineering* 25, 627–634.

Glover, F. (1975). "Improved Linear Integer Programming Formulations of Nonlinear Integer Problems." *Management Science* 22, 455–460.

Grossmann, I.E., I. Quesada, R. Raman, and V.T. Voudouris. (1996). "Mixed-Integer Optimization Techniques for the Design and Scheduling of Batch Processes." In G. V. Reklaitis, A. K. Sunol, D. W. T. Rippin and O. Hortacsu (ed.), *Batch Processing Systems Engineering*, Berlin: Springer, pp. 451–494.

Harjunkoski, I. and I.E. Grossmann. (2001). "A Decomposition Approach for the Scheduling of a Steel Plant Production." *Computers & Chemical Engineering* 25, 1647–1660.

Hui, C. and A. Gupta. (2001). "A Bi-index Continuous-Time Mixed-Integer Linear Programming Model for Single-Stage Batch Scheduling with Parallel Units." *Industrial & Engineering Chemistry Research* 40, 5960–5967.

Hui, C., A. Gupta, and H.A.J. van der Meulen. (2000). "A Novel MILP Formulation for Short-Term Scheduling of Multi-Stage Multi-Product Batch Plants with Sequence-Dependent Constraints." *Computers & Chemical Engineering* 24, 2705–2717.

Ierapetritou, M.G. and C.A. Floudas. (1998a). "Effective Continuous-Time Formulation for Short-Term Scheduling: 1. Multipurpose Batch Processes." *Industrial & Engineering Chemistry Research* 37, 4341–4359.

Ierapetritou, M.G. and C.A. Floudas. (1998b). "Effective Continuous-Time Formulation for Short-Term Scheduling: 2. Continuous and Semi-Continuous Processes." *Industrial & Engineering Chemistry Research* 37, 4360–4374.

Ierapetritou, M.G. and C.A. Floudas. (2001). Comments on "An Improved RTN Continuous-Time Formulation for the Short-term Scheduling of Multipurpose Batch Plants." *Industrial & Engineering Chemistry Research* 40, 5040–5041.

Ierapetritou, M.G., T.S. Hené, and C.A. Floudas. (1999). "Effective Continuous-Time Formulation for Short-Term Scheduling: 3. Multiple Intermediate Due Dates." *Industrial & Engineering Chemistry Research* 38, 3446–3461.

Janak, S.L., X. Lin, and C.A. Floudas. (2004). "Enhanced Continuous-Time Unit-Specific Event-Based Formulation for Short-Term Scheduling of Multipurpose Batch Processes: Resource Constraints and Mixed Storage Policies." *Industrial & Engineering Chemistry Research* 43, 2516–2533.

Karimi, I.A. and C.M. McDonald. (1997). "Planning and Scheduling of Parallel Semi-Continuous Processes. 2. Short-Term Scheduling." *Industrial & Engineering Chemistry Research* 36, 2701–2714.

Kondili, E., C.C. Pantelides, and R.W.H. Sargent. (1988). "A General Algorithm for Scheduling Batch Operations." In *Proceedings of the Third International Symposium on Process Systems Engineering*, Sydney, Australia, pp. 62–75.

Kondili, E., C.C. Pantelides, and R.W.H. Sargent. (1993). "A General Algorithm for Short-Term Scheduling of Batch Operations - I. MILP Formulation." *Computers & Chemical Engineering* 17, 211–227.

Ku, H. and I.A. Karimi. (1988). "Scheduling in Serial Multiproduct Batch Processes with Finite Interstage Storage: A Mixed Integer Linear Programming Formulation." *Industrial & Engineering Chemistry Research* 27, 1840–1848.

Lamba, N. and I.A. Karimi. (2002a). "Scheduling Parallel Production Lines with Resource Constraints. 1. Model Formulation." *Industrial & Engineering Chemistry Research* 41, 779–789.

Lamba, N. and I.A. Karimi. (2002b). "Scheduling Parallel Production Lines with Resource Constraints. 2. Decomposition Algorithm." *Industrial & Engineering Chemistry Research* 41, 790–800.

Lee, K., S. Heo, H. Lee, and I. Lee. (2002). "Scheduling of Single-Stage and Continuous Processes on Parallel Lines with Intermediate Due Dates." *Industrial & Engineering Chemistry Research* 41, 58–66.

Lee, K., H.I. Park, and I. Lee. (2001). "A Novel Nonuniform Discrete Time Formulation for Short-Term Scheduling of Batch and Continuous Processes." *Industrial & Engineering Chemistry Research* 40, 4902–4911.

Lin, X. and C.A. Floudas. (2001). "Design, Synthesis and Scheduling of Multipurpose Batch Plants via an Effective Continuous-Time Formulation." *Computers & Chemical Engineering* 25, 665–674.

Lin, X., C.A. Floudas, S. Modi, and N.M. Juhasz. (2002). "Continuous-Time Optimization Approach for Medium-Range Production Scheduling of a Multiproduct Batch Plant." *Industrial & Engineering Chemistry Research* 41, 3884–3906.

Majozi, T. and X.X. Zhu. (2001). "A Novel Continuous-Time MILP Formulation for Multipurpose Batch Plants. 1. Short-Term Scheduling." *Industrial & Engineering Chemistry Research* 40, 5935–5949.

Manne, A.S. (1960). "On the Job-Shop Scheduling Problem." *Operations Research* 8, 219–223.

Maravelias, C.T. and I.E. Grossmann. (2003). "A New General Continuous-Time State Task Network Formulation for Short-Term Scheduling of Multipurpose Batch Plants." *Industrial & Engineering Chemistry Research* 42, 3056–3074.

Méndez, C.A. and J. Cerdá. (2000a). "Optimal Scheduling of a Resource-Constrained Multiproduct Batch Plant Supplying Intermediates to Nearby End-Product Facilities." *Computers & Chemical Engineering* 24, 369–376.

Méndez, C.A., G.P. Henning, and J. Cerdá. (2000b). "Optimal Scheduling of Batch Plants Satisfying Multiple Product Orders with Different Due-Dates." *Computers & Chemical Engineering* 24, 2223–2245.

Méndez, C.A., G.P. Henning, and J. Cerdá. (2001). "An MILP Continuous-Time Approach to Short-Term Scheduling of Resource-Constrained Multistage Flowshop Batch Facilities." *Computers & Chemical Engineering* 25, 701–711.

Mockus, J., W. Eddy, A. Mockus, L. Mockus, and R.V. Reklaitis. (1997). *Bayesian Heuristic Approach to Discrete and Global Optimization: Algorithms, Visualization, Software, and Applications*. Kluwer Academic Publishers.

Mockus, L. and G.V. Reklaitis. (1997). "Mathematical Programming Formulation for Scheduling of Batch Operations Based on Nonuniform Time Discretization." *Computers & Chemical Engineering* 21, 1147–1156.

Mockus, L. and G.V. Reklaitis. (1999a). "Continuous Time Representation Approach to Batch and Continuous Process Scheduling. 1. MINLP Formulation." *Industrial & Engineering Chemistry Research* 38, 197–203.

Mockus, L. and G.V. Reklaitis. (1999b). "Continuous Time Representation Approach to Batch and Continuous Process Scheduling. 2. Computational Issues." *Industrial & Engineering Chemistry Research* 38, 204–210.

Moon, S. and A.N. Hrymak. (1999). "Mixed-Integer Linear Programming Model for Short-Term Scheduling of a Special Class of Multipurpose Batch Plants." *Industrial & Engineering Chemistry Research* 38, 2144–2150.

Moon, S., S. Park, and W.K. Lee. (1996). "New MILP Models for Scheduling of Multiproduct Batch Plants under Zero-Wait Policy." *Industrial & Engineering Chemistry Research* 35, 3458–3469.

Orçun, S., I.K. Altinel, and Ö Hortaçsu. (1999). "Reducing the Integrality Gap with a Modified Re-Formulation Linearization Approach." *Computers & Chemical Engineering Supplement*, pp. S539–S542.

Orçun, S., I.K., Altinel, and Ö Hortaçsu. (2001). "General Continuous Time Models for Production Planning and Scheduling of Batch Processing Plants: Mixed Integer Linear Program Formulations and Computational Issues." *Computers & Chemical Engineering* 25, 371–389.

Pantelides, C.C. (1993). "Unified Frameworks for Optimal Process Planning and Scheduling." In D.W.T. Rippin, J.C. Hale, and J. Davis (eds.), *Proceedings of the Second International Conference on Foundations of Computer-Aided Process Operations*, Crested Butte, Colorado, pp. 253–274.

Pekny, J.F. and G.V. Reklaitis. (1998). "Towards the Convergence of Theory and Practice: A Technology Guide for Scheduling/Planning Methodology." In J.F. Pekny and G.E. Blau (eds.), *Proceedings of the Third International Conference on Foundations of Computer-Aided Process Operations*, Snowbird, Utah, pp. 91–111.

Pekny, J.F. and M.G. Zentner. (1993). "Learning to Solve Process Scheduling Problems: The Role of Rigorous Knowledge Acquisition Frameworks." In D.W.T. Rippin, J.C. Hale, and J. Davis (eds.), *Proceedings of the Second International Conference on Foundations of Computer-Aided Process Operations*, Crested Butte, Colorado, pp. 275–309.

Pinto, J.M. and I.E. Grossmann. (1995). "A Continuous Time Mixed Integer Linear Programming Model for Short Term Scheduling of Multistage Batch Plants." *Industrial & Engineering Chemistry Research* 34, 3037–3051.

Pinto, J.M. and I.E. Grossmann. (1996). "An Alternate MILP Model for Short-Term Scheduling of Batch Plants with Preordering Constraints." *Industrial & Engineering Chemistry Research* 35, 338–342.

Pinto, J.M. and I.E. Grossmann. (1998). "Assignment and Sequencing Models for the Scheduling of Process Systems." *Annals of Operations Research* 81, 433–466.

Pinto, J.M., A. Türkay, B. Bolio, and I.E. Grossmann. (1998). "STBS: A Continuous-Time MILP Optimization for Short-Term Scheduling of Batch Plants." *Computers & Chemical Engineering* 22, 1297–1308.

Pritsker, A.A.B., L.J. Watters, and P.M. Wolfe. (1969). "Multiproject Scheduling with Limited Resources: A Zero-One Programming Approach." *Management Science* 16, 93–108.

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

Rippin, D.W.T. (1993). "Batch Process Systems Engineering: A Retrospective and Prospective Review." *Computers & Chemical Engineering* 17, S1–S13.

Sahinidis, N.V. and I.E. Grossmann. (1991b). "Reformulation of Multiperiod MILP Models for Planning and Scheduling of Chemical Processes." *Computers & Chemical Engineering* 15, 255–272.

Schilling, G. (1997). "Optimal Scheduling of Multipurpose Plants." PhD thesis, University of London.

Schilling, G. and C.C. Pantelides. (1996). "A Simple Continuous-Time Process Scheduling Formulation and a Novel Solution Algorithm." *Computers & Chemical Engineering* 20, S1221–S1226.

Schilling, G. and C.C. Pantelides. (1996b). "A Hybrid Branch and Bound Algorithm for Continuous-Time Process Scheduling Formulations." AIChE 1996 Annual Meeting, Chicago, IL, November 1996, Paper 171d.

Shah, N. (1998). "Single- And Multisite Planning and Scheduling: Current Status and Future Challenges." In J.F. Pekny and G.E. Blau (eds.), *Proceedings of the Third International Conference on Foundations of Computer-Aided Process Operations*, Snowbird, Utah, pp. 75–90.

Shah, N., C.C. Pantelides, and R.W.H. Sargent. (1993). "A General Algorithm for Short-Term Scheduling of Batch Operations-II. Computational Issues." *Computers & Chemical Engineering* 17, 229–244.

Sherali, H.D. and W.P. Adams. (1994). "A Hierarchy of Relaxations and Convex-Hull Characterizations for Mixed-Integer Zero-One Programming Problems." *Discrete Applied Mathematics* 52, 83–106.

Wang, S. and M. Guignard. (2002). "Redefining Event Variables for Efficient Modeling of Continuous-Time Batch Processing." *Annals of Operations Research* 116, 113–126.

Wilkinson, S.J., A. Cortier, N. Shah, and C.C. Pantelides. (1996). "Integrated Production and Distribution Scheduling on a European-Wide Basis." *Computers & Chemical Engineering* 20, S1275–S1280.

Wilkinson, S.J., N. Shah, and C.C. Pantelides. (1995). "Aggregate Modelling of Multipurpose Plant Operation." *Computers & Chemical Engineering* 19, S583–S588.

Yee, K.L. and N. Shah. (1998). "Improving the Efficiency of Discrete Time Scheduling Formulation." *Computers & Chemical Engineering* 22, S403–S410.

Yi, G., K. Suh, B. Lee, and E.S. Lee. (2000). "Optimal Operation of Quality Controlled Product Storage." *Computers & Chemical Engineering* 24, 475–480.

Zentner, M.G., A. Elkamel, J.F. Pekny, and G.V. Reklaitis. (1998). "A Language for Describing Process Scheduling Problems." *Computers & Chemical Engineering* 22, 125–145.

Zentner, M.G., J.F. Pekny, G.V. Reklaitis, and J.N.D. Gupta. (1994). "Practical Considerations in Using Model-Based Optimization for the Scheduling and Planning of Batch/Semicontinuous Processes." *Journal of Process Control* 4, 259–280.

Zhang, X. (1995). "Algorithms for Optimal Scheduling Using Nonlinear Models." PhD thesis, University of London.

Zhang, X. and R.W.H. Sargent. (1996). "The Optimal Operation of Mixed Production Facilities—A General Formulation and Some Solution Approaches for the Solution." *Computers & Chemical Engineering* 20, 897–904.

Zhang, X. and R.W.H. Sargent. (1998). "The Optimal Operation of Mixed Production Facilities—Extensions and Improvements." *Computers & Chemical Engineering* 22, 1287–1295.