

Multitask and Multistage Production Planning and Scheduling for Process Industries

Francesco Gaglioppa, Lisa A. Miller, Saif Benjaafar

Graduate Program in Industrial and Systems Engineering, Department of Mechanical Engineering, University of Minnesota, Minneapolis, Minnesota 55455 {fgagliop@me.umn.edu, lmiller@me.umn.edu, saif@umn.edu}

We consider the planning and scheduling of production in a multitask/multistage batch manufacturing process typical of industries such as chemical manufacturing, food processing, and oil refining. We allow instances in which multiple sequences of tasks may be used to produce end products. We formulate the problem as a mixed-integer linear program and show that the linear programming relaxation has a large integrality gap and requires significant computational effort to solve to optimality for large instances. Using echelon inventory, we construct a new family of valid inequalities for this problem. The formulation with the additional constraints leads to a significantly tighter linear programming relaxation and to greatly reduced solution times for the mixed-integer linear program.

Subject classifications: production planning/scheduling; echelon inventory; integer programming.

Area of review: Optimization.

History: Received July 2004; revisions received May 2006, May 2007; accepted May 2007.

1. Introduction

We consider the planning and scheduling of production in a multitask/multistage batch manufacturing process typical of industries such as chemical manufacturing, food processing, and oil refining (see Figure 1). We first consider a system with a single processing unit. The processing unit is capable of carrying out several tasks, each consuming one or more inputs and producing one or more outputs. Inputs for each task might consist of raw resources (feeds) or semifinished products (intermediates). Similarly, outputs from each task may consist of intermediates or finished products. It is possible for the same intermediate or finished product to be produced via more than one task. Consequently, each intermediate or finished product can be the result of one or more sequences of tasks. Each task is associated with a variable batch size, a variable production cost, a fixed processing time, and a task-specific setup time and setup cost.

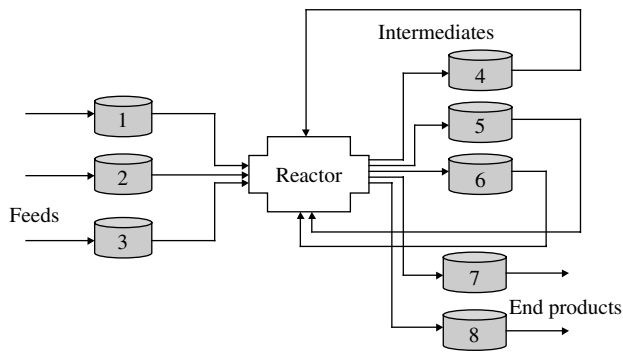
We consider an environment in which time is divided into discrete uniform periods. A period is chosen sufficiently small to allow the modeling of start and end times of each task (e.g., the length of a time period is a common divisor to all task processing times). In each period, there may be external demand for one or more finished products or intermediates. To meet demand while satisfying capacity constraints, the plant may choose to produce ahead of demand and hold inventory. In that case, a holding cost per unit of inventory per period is incurred. All costs, including production, inventory holding, and setup costs, could vary from period to period.

Our objective is to develop production schedules that specify production quantities and production start times

that minimize the sum of production, setup, and inventory holding costs while meeting demand on time and satisfying constraints on production capacity and processing unit availability. We adopt a representation scheme similar to the state-task-network formalism introduced by Kondili et al. (1993), where a system is described by a set of states (i.e., feeds, intermediates, and finished products) and a set of tasks that transform material from one state to another. We allow for the possibility of multiple tasks being carried out on the same unit and for those tasks to have overlapping sets of inputs and outputs. We also allow for the possibility of multistep processing, where a material can undergo a series of tasks on the same units. We refer to our problem as *the multitask/multistage production planning and scheduling problem (MPSP)*.

We formulate the MPSP as a mixed-integer linear program (MILP). We observe that the formulation leads to an NP-hard problem with a large integrality gap (gap between the optimal solution of the MILP and the optimal solution of the linear programming relaxation). We use the notion of *echelon inventory* to construct new valid inequalities (cutting planes) for the formulation. We show that the formulation with the additional constraints leads to a significantly tighter LP relaxation and to much-reduced solution times for the MILP. We compare the impact of *echelon inventory constraints* with that of *single-stage inventory constraints* that have been used in related settings such as capacitated lot-sizing problems (see Wolsey 1997). We show that echelon constraints can significantly outperform single-stage constraints. Based on an extensive numerical study, we highlight cases where echelon inventory constraints are particularly useful. We first treat the case of systems with a

Figure 1. An example of the multitask/multistage batch process.



single processing unit. Then, we extend the formulation and the additional valid inequalities to systems with multiple processing units.

The MPSP is related to the large body of literature on production planning and scheduling in process industries, as well as to the capacitated lot-sizing problem (CLSP) in discrete manufacturing. However, in contrast to the CLSP, there is not necessarily a one-to-one correspondence between tasks and input/output materials in the MPSP. This makes the scheduling problem considerably more difficult because the manufacturing of one material could affect the availability of several others. The complexity of the problem can be further compounded by the reentrant nature of the flows and the possibility of producing the same material via alternative routes. Because the problem is a generalization of the CLSP, we expect exact solutions to large problems to be difficult to find.

The rest of this paper is organized as follows. In §2, we provide a brief review of the related literature. In §3, we present the problem formulation of the MPSP and discuss modeling assumptions. In §4, we describe the notion of echelon inventory and use it to construct valid inequalities. In §5, we extend our results to systems with multiple processing units. In §6, we report on numerical results. In §7, we provide a summary and a brief discussion of various extensions.

2. Related Literature

There is an extensive literature on production planning and scheduling in process industries. Recent reviews can be found in Kallrath (2003), Pekny (2002), Shah (1998), and Applequist et al. (1997). In process industries, two modes of production can be distinguished: *continuous* and *batch*. Continuous production is adopted when there are few products with similar routings and relatively stable demand. Batch production is adopted when the number of products is large and demand for each product varies with time. Batch production in process manufacturing differs from batch production in discrete manufacturing in that each operation could require multiple inputs and could produce

multiple outputs. In contrast to discrete manufacturing, the quantities of both inputs and outputs are typically continuous. The output from a process might revisit the same process several times for further processing. Hence, there can be significant reentrant flows.

An important development in the modeling of planning and scheduling in process manufacturing has been the state-task network (STN) representation introduced by Kondili et al. (1993). The STN framework uses materials (states) and tasks as building blocks for the process description, with each task consuming and producing materials while using equipment. An enhancement to the STN representation is the resource-task network (RTN) proposed by Pantelides (1994), which unifies the treatment of both equipment and materials as resources that are consumed (produced) at the start (end) of a task.

Although the boundaries are overlapping, the existing literature can be classified as pertaining to either planning or scheduling. For planning, time is typically discretized into planning periods where only aggregate capacity is taken into account and the primary decisions are the quantities produced of each material in each period. Examples of recent papers include Papageorgiou and Pantelides (1996a) and van den Heever and Grossman (1999). Formulations with continuous-time representation can be found in Schilling and Pantelides (1996), Zhang and Sargent (1996), and Mockus and Reklaitis (1999). A review can be found in Maravelias and Grossman (2003a). Planning problems are typically formulated as linear programs and can be solved relatively efficiently using standard methods. For scheduling, time is either finely discretized or treated as a continuous parameter. In addition to production quantities for each material, decisions in a scheduling problem include the start and end time of individual tasks on specific production units. Scheduling problems are typically formulated as MILPs. In most cases, the formulation leads to an NP-hard problem. Recent examples include Maravelias and Grossman (2003b), Majozi and Zhu (2001), and Neumann et al. (2003). To cope with problem complexity, several papers propose decomposition approaches, where the original problem is decomposed into a series of subproblems with smaller time horizons (see, for example, Elkamel et al. 1997 and Lin et al. 2002). Others develop reformulations that are relatively easier to solve (see, for example, Sahinidis and Grossman 1991, Shah et al. 1993, and Ierapetritou and Floudas 1998).

Planning and scheduling in process industries can be seen as a generalization of the CLSP in discrete manufacturing. The CLSP has been widely studied. Review of the literature and recent advances can be found in Wolsey (2002), Miller and Wolsey (2003), and Atamtürk and Muñoz (2004). The CLSP, which is NP-hard, can be formulated as an MILP and solved via standard branch and bound for relatively small problems. Reformulations and the introduction of valid inequalities have been successful in reducing solution times in some cases for larger problem instances.

For example, Barany et al. (1984) proposed the so-called (I, S) inequalities. Several other authors have found valid inequalities for other variations of the CLSP; for example, see Magnanti and Vachani (1990), Constantino (1996), Belvaux and Wolsey (2000, 2001), and Miller et al. (2003).

The literature on the CLSP with multiple stages is more limited. The problem is computationally harder than the simple CLSP. Therefore, the solution of large problems invariably involves heuristic approaches; see Katok et al. (1998), Tempelmeier and Destroff (1996), Stadler (2003), and the references therein. The notion of echelon inventory first introduced by Clark and Scarf (1960) has been used to reformulate the CLSP with multiple stages and improve computational efficiency; see, for example, Afentakis and Gavish (1986), Pochet and Wolsey (1991), and Belvaux and Wolsey (2000).

3. Formulation

We first introduce a formulation for the MPSP with just a single processing unit. The MPSP can be described in terms of a set of tasks, N , a set of materials, R , and a set of periods, T , over which demand is known. The demand in period t for material r is denoted by d_t^r . We allow demand to occur for both finished and intermediate products. Each task consumes a set of inputs in fixed proportions, with $\rho^{i,r}$ being the proportion of input to task i due to material r . Each task produces a set of outputs also in fixed proportions, with $\sigma^{i,r}$ being the proportion of output from task i in the form of material r . We denote the set of tasks for which material r is an input by $I(r)$ and the set of tasks from which material r is an output by $O(r)$. Each task requires a fixed processing time of α_i periods.

The process incurs a variable production cost p_t^i per unit of production quantity undertaken by task i in period t and a fixed setup cost g_t^i if task i is initiated in period t . The system also incurs a holding cost h_t^r per unit of inventory of material r held in period t . There is a maximum capacity, C_t^i , for the production quantity of task i in period t .

There are four decision variables: (1) the production quantity, x_t^i , initiated by task i in period t ; (2) the status of the processing unit, y_t^i , where $y_t^i = 1$ if the unit is assigned to task i at time t and $y_t^i = 0$ otherwise; (3) the start of a task, z_t^i , where $z_t^i = 1$ if task i is initiated at time t ; and (4) the inventory level, s_t^r , of material r in period t . We assume that the initial inventory, s_0^r , of each material r is known.

The sequence of events within each period is as follows. At the beginning of a period t , a production run for a task i that was initiated at time $t - \alpha_i$ completes. This immediately increases the inventory levels of all corresponding outputs. The external demands, d_t^r , for all materials r in R are then fulfilled. This is followed by the initiation of any new production runs (note that a run of a task i of quantity x_t^i that is initiated at the beginning of period t will complete at the beginning of period $t + \alpha_i$). The level

of inventory on hand for all materials is then immediately updated to account for the fulfillment of both external demand and internal usage. The remaining inventory from each material incurs a holding cost for the entire current period.

The MPSP can now be formulated as follows:

$$\min \sum_{t \in T} \sum_{r \in R} h_t^r s_t^r + \sum_{t \in T} \sum_{i \in N} p_t^i x_t^i + \sum_{t \in T} \sum_{i \in N} g_t^i z_t^i \quad (1)$$

subject to

$$s_t^r = s_{t-1}^r + \sum_{i \in N} \sigma^{i,r} x_{t-\alpha_i}^i - \sum_{i \in N} \rho^{i,r} x_t^i - d_t^r \quad \forall r \in R, t \in T, \quad (2)$$

$$x_t^i \leq C_t^i z_t^i \quad \forall i \in N, t \in T, \quad (3)$$

$$\sum_{u=1}^{\alpha_i} z_{t-u+1}^i \leq y_t^i \quad \forall i \in N, t \in T, \quad (4)$$

$$z_t^i \geq y_t^i - y_{t-1}^i \quad \forall i \in N, t \in T, \quad (5)$$

$$\sum_{i \in N} y_t^i = 1 \quad \forall t \in T, \quad (6)$$

$$x_t^i \geq 0 \quad \forall i \in N, t \in T, \quad (7)$$

$$s_t^r \geq 0 \quad \forall r \in R, t \in T, \quad (8)$$

$$y_t^i, z_t^i \in \{0, 1\} \quad \forall i \in N, t \in T. \quad (9)$$

The objective function consists of minimizing the sum of inventory holding, production, and setup costs. Constraints (2) are flow conservation constraints. Constraints (3) are production capacity constraints. Constraints (4) require that at most one run of task i is initiated in any consecutive set of α_i periods, and that if task i is initiated in any of these periods, the processing unit must remain set up for task i for the next α_i periods. (This is stronger than the obvious constraint that the processing unit must be set up for task i to initiate it.) Constraints (5) ensure that production is initiated at time t whenever the process is set up for task i in period t and is not set up for task i in period $t - 1$. Moreover, the process remains in the same setup status if the production unit has to stay idle, ensuring that no unneeded setups are carried out. Constraints (6) guarantee that the processing unit is set up for exactly one task in each time period.

The above formulation makes several assumptions that are worth highlighting. We assume that proportions of input and output materials are fixed. In some environments (e.g., fuel blending), there is flexibility in how these proportions are chosen subject to constraints on quality of the outputs (see, for example, Karmarkar and Rajaram 2001). However, it is often the practice that once these proportions are chosen at the product design stage, they remain fixed. We assume that the various inputs are consumed at the beginning of each task and the various outputs become available when the task completes. In some settings, inputs are added

gradually over time (e.g., a cooking process). Similarly, outputs could be collected at various stages of the process, such as in distillation. Another assumption we make is that processing times are production quantity independent. Although this assumption holds for many processes, such as chemical reactions, it might not hold for others, such as blending. Additionally, only one task can be performed on the processing unit in each period, and once a task is initiated, it must continue running until completion. Finally, we assume that setup times and costs are sequence independent. This can be justified in many cases where setup costs are associated with the startup effort of initiating a new task or in instances where setup costs reflect poor usage of capacity or increased usage of labor. However, instances arise where it is important to capture sequence dependency (e.g., sequence-dependent cleaning operations requiring expensive solvents). We offer some discussion of this issue with possible extensions of the current model in §7.

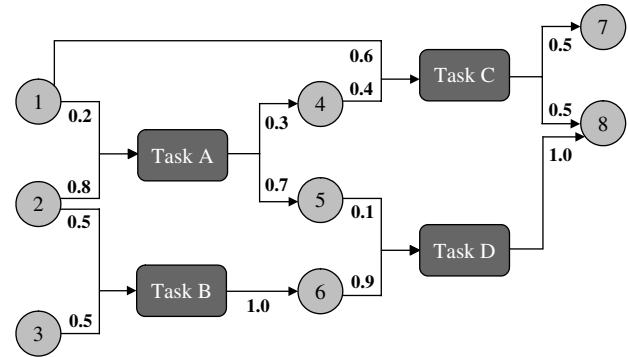
MPSP is an NP-hard problem because the NP-hard capacitated lot-sizing problem is a special case (Florian et al. 1980, Bitran and Yanasse 1982, Wolsey 2002). The uncapacitated joint replenishment problem, which is strongly NP-hard (Arkin et al. 1989), is also a special case of the MPSP. Experimentation with solving the MILP formulation of the MPSP using CPLEX 8.1 with its default settings shows that solution times grow quickly with problem-size, and the problem eventually becomes computationally prohibitive. Numerical results also show that the LP relaxation of the MILP formulation leads to poor lower bounds on the optimal solution.

We conclude this section by noting that the flow of material in the MPSP can be described by a directed graph (network) G , consisting of two sets of nodes, V_1 and V_2 , corresponding, respectively, to materials and tasks. Successor and predecessor nodes to a node in V_1 are always nodes in V_2 , and vice-versa, successor and predecessor nodes to a node in V_2 are always nodes in V_1 . Hence, the arcs in the graph always connect nodes from V_i to V_j , where $i \neq j$. An arc (r, i) from a node in $r \in V_1$ to a node in $i \in V_2$ is introduced if task i requires material r as an input. The label on arc (r, i) is $\rho^{i,r}$, the fraction of input to task i due to material r . Similarly, an arc (i, r) from a node in $i \in V_2$ to a node in $r \in V_1$ is included in the graph if task i produces material r . The label on arc (i, r) is $\sigma^{i,r}$, the fraction of output from task i in the form of material r . Figure 2 provides an example of such a network with eight materials (numbered 1–8) and four tasks (labelled A–D). We will refer to this graphical representation in future sections.

4. Valid Inequalities

In this section, we introduce two sets of valid inequalities for the MPSP. In §4.1, we introduce a family of inequalities based on local inventory levels and external demand of materials. In §4.2, we introduce the idea of echelon inventory for our setting and extend the valid inequalities to consider internal demand for materials.

Figure 2. Network representation of a process structure.



4.1. Single-Level Inequalities

We begin with the basic intuition that local inventory of material r increases at time t if and only if some task $i \in O(r)$ is initiated at time $t - \alpha_i$, where α_i is the processing time for task i . Likewise, local inventory of material r cannot increase between periods k and t if no production run for some $i \in O(r)$ is started between $k - \alpha_i$ and $t - \alpha_i$.

This gives us the first lemma, which states that if inventory of a material r does not increase in a time interval, then there must be enough on-hand inventory of that material at the beginning of the interval to satisfy all demand that occurs within the time interval. We first define a new variable: for any $t \geq k$, let $s_{k-1,t}^r$ represent the quantity of inventory of material r in period $k - 1$ that is used to satisfy demand in period t .

LEMMA 1. *The following set of inequalities is valid for the MPSP:*

$$s_{k-1,t}^r \geq d_t^r \left(1 - \sum_{i \in O(r)} \sum_{u=k-\alpha_i}^{t-\alpha_i} z_u^i \right) \quad \forall r \in R, k = 2, \dots, T, t = k, \dots, T. \quad (10)$$

PROOF. If any amount of material r is released from production within time interval $[k, t]$, then at least one of the z variables on the right-hand side of (10) is equal to one, which forces the right-hand side of the inequality to be nonpositive, and the inequality is trivially satisfied. Otherwise, the inequality reduces to $s_{k-1,t}^r \geq d_t^r$, which enforces that all demand for material r in period t is satisfied by inventory that was on-hand in period $k - 1$. Because no new inventory of material r is created between periods k and t , this must be true. \square

We can sum the above inequalities over a sequence of consecutive periods $t = k, \dots, l$ to derive valid inequalities in the original space of variables.

THEOREM 2. *The following set of inequalities is valid for the MPSP:*

$$s_{k-1}^r \geq \sum_{t=k}^l \left[d_t^r \left(1 - \sum_{i \in O(r)} \sum_{u=k-\alpha_i}^{t-\alpha_i} z_u^i \right) \right] \quad \forall r \in R, k = 2, \dots, T, l = k, \dots, T. \quad (11)$$

PROOF. For a given k and $l \geq k$, summing the inequalities defined in Lemma 1 for $t = k, \dots, l$ gives

$$\sum_{t=k}^l s_{k-1,t}^r \geq \sum_{t=k}^l \left[d_t^r \left(1 - \sum_{i \in O(r)} \sum_{u=k-\alpha_i}^{t-\alpha_i} z_u^i \right) \right].$$

The amount of on-hand inventory in period $k-1$ that is used to satisfy demand in periods k through l can be no more than the total on-hand inventory in period $k-1$, so $s_{k-1}^r \geq \sum_{t=k}^l s_{k-1,t}^r$. Therefore, (11) holds. \square

We call these inequalities *single-level inequalities* because they take into account the external demand for a material, without considering any requirement coming from more downstream levels. Note that the above set of constraints could present some redundancy if demand is zero in some periods. In fact, for a material r , it is sufficient to restrict the parameter l to periods in which demand for r occurs. For any l' with $d_{l'}^r = 0$, inequality (11) is equivalent to that generated for $l = l' - 1$.

COROLLARY 3. *The following sets of constraints:*

$$s_{k-1}^r \geq \sum_{t=k}^l \left[d_t^r \left(1 - \sum_{i \in O(r)} \sum_{u=k-\alpha_i}^{t-\alpha_i} z_u^i \right) \right] \\ \forall r \in R, k = 2, \dots, T, l = k, \dots, T \quad (12)$$

and

$$s_{k-1}^r \geq \sum_{t=k}^l \left[d_t^r \left(1 - \sum_{i \in O(r)} \sum_{u=k-\alpha_i}^{t-\alpha_i} z_u^i \right) \right] \\ \forall r \in R, k = 2, \dots, T, l \in L(r, k), \quad (13)$$

where $L(r, k) = \{t \mid d_t^r > 0 \text{ and } t \geq k\}$, are equivalent.

The additional number of constraints that are generated by single-level inequalities is $O(|R|T^2)$, where $|R|$ is the number of materials and T is the planning horizon. Although similar inequalities have been shown to perform well in the related setting of CLSP (see Belvaux and Wolsey 2000, 2001 for implementation and computational results), our experience with several instances of the MPSP shows that solution time can significantly increase when inequalities (13) are added. One possible explanation is that these inequalities consider only external demand for materials and therefore are trivially satisfied for intermediate materials for which external demand never occurs. Hence, any potential benefits from the tighter formulation are exceeded by the computational burden induced by the larger problem size. However, in the next section, we show that generating similar inequalities that consider internal demand created by tasks that require intermediate materials as input can significantly improve computational performance.

4.2. Echelon Inequalities

If we consider the system as a whole, material r can be observed at time t in three different forms: as r itself, located in inventory; as work in progress in an ongoing task i that required r as an input; and as embedded in materials produced by some task that required r as an input. Consider a task i that requires that a proportion $\rho^{i,r}$ of its input is material r , and that produces output, of which the proportion $\sigma^{i,g}$ is in the form material g . If at time t , task i is initiated in quantity x_t^i , then $\rho^{i,r}x_t^i$ of material r is consumed at time t and $\sigma^{i,g}x_t^i$ of material g is produced when task i completes. The local inventory s_t^r is decreased by $\rho^{i,r}x_t^i$, but this quantity does not actually leave the system. Instead, it is transformed into work in progress for α_i periods, at the end of which it is transformed into material g . We can say that for each unit of g that is released by the production run of task i , a fraction $\rho^{i,r}$ consists of r . A similar observation can be made for any material \bar{g} that is not produced directly from r . If \bar{g} is produced by a task that requires material g as an input, and g is partially composed of r , then a fraction of each unit of \bar{g} consists of r as well.

We refer to material g as a *successor* of r if it can be produced by a task with r or with another material consisting of r as one of its inputs. In this situation, we also refer to r as a *predecessor* of g . We denote the set of successors of r by $S(r)$. We also define $S'(r)$ as the set of the *immediate successors* of r . It contains all the materials that are produced from any task that consumes r —i.e., $S'(r) = \{g \in R \mid \exists i \in N: \rho^{i,r}\sigma^{i,g} > 0\}$. Using the network representation of the problem introduced in §3, $g \in S'(r)$ if there exists some $i \in N$ such that (r, i) and (i, g) are arcs in G . Similarly, $g \in S(r)$ if there is a directed path in G from r to g .

We now define the total amount of material r in the system at time t , as *echelon inventory*, \hat{s}_t^r . This definition is consistent with the one commonly used in the inventory theory literature; see, for example, Zipkin (2000).

Computing echelon inventory exactly in the MPSP setting is difficult for several reasons. First, if the process structure contains reentrant flows (directed cycles), then the amount of one material contained in a successor could depend on how many times the material has been recycled. For the purposes of this section, we will assume that process structures have no reentrant flows. We will be addressing the relaxation of this assumption in §4.4.

More significantly, there might exist several alternative tasks, or sets of tasks, that produce a particular material g , with each set of tasks possibly using different amounts of various intermediate materials. Exact computation of echelon inventory must therefore track which set of tasks were used to produce each unit of each material. Keeping track of production history within the MPSP requires introducing a large number of additional binary variables, making the problem potentially more difficult to solve when the process structure is complex.

However, we can make statements about the events that increase or decrease the echelon inventory of a material.

PROPERTY 4. *The only event that increases the echelon inventory of a material r is the completion of a task that produces r .*

PROPERTY 5. *The only events that decrease the echelon inventory for a material r are fulfillment of external demand for r and fulfillment of external demand for any of the successors of material r .*

To avoid the above difficulties associated with tracking echelon inventory exactly, we identify an upper bound on echelon inventory. This bound will be utilized in the valid inequalities derived later in this section.

Consider two materials, r and g , where g is a successor of r . If there are two or more tasks that produce g and have r as an input, then the amount of material g due to material r is at least the fraction of r required by the task that uses the least amount of r and at most the fraction of r required by the task that uses the highest amount of r . Similar reasoning applies for materials that are indirect successors of r .

We introduce the parameters $e^{r,g}$ and $f^{r,g}$, which we refer to as the *minimum coefficient of transformation* and the *maximum coefficient of transformation*, respectively, to measure the minimum and maximum amount of material r that may be used to produce one unit of material g . The characteristics of the network introduced in §3 allow us to compute these coefficients in a sequential manner. First, the nodes in the graph are given unique labels q from the set $\{1, \dots, |N| + |R|\}$. The labels are assigned so that for any pair of nodes $i, j \in V_1 \cup V_2$ for which a directed path exists in G from i to j , we have $q(i) < q(j)$. Under the assumption that there is no recycling of materials, there are no directed cycles in the graph, so such a labeling exists. Furthermore, note that $q(r) < q(g)$ for all successors g of r .

The following is an algorithm for computing the minimum coefficients of transformation:

Step 1. Set

$$e^{r,r} = 1 \quad \forall r \in R,$$

$$e^{r,g} = 0 \quad \forall r, g \neq r \in R.$$

Set $k = 1$.

Step 2. Select the node n with $q(n) = k$. If $n \in V_2$ (i.e., n corresponds to a task), go to Step 3. Otherwise, $\forall r \in R$, set

$$e^{r,n} = \min_{i \in O(n)} \sum_{g \in R} e^{r,g} \rho^{i,g}.$$

Step 3. If $k = |N| + |R|$, terminate. Else, set $k = k + 1$ and go to Step 2.

In Step 2, recall that $i \in O(n)$ means that task i produces material n , so $\rho^{i,g} > 0$ implies g is a predecessor of n . Therefore, $q(g) < q(n)$, and $e^{r,g}$ has already been computed. This algorithm computes the matrix $[e^{r,g}]$ in time $O(|R|^2|N|)$. A similar algorithm is used to compute the maximum coefficients of transformation $f^{r,g}$.

EXAMPLE. Consider the example shown in Figure 2. Materials 4 and 5 both include 20% of material 1, so $e^{1,4} = f^{1,4} = 0.2$ and $e^{1,5} = f^{1,5} = 0.2$. The coefficients of transformation $e^{1,6} = f^{1,6} = 0$ because material 1 is never used to produce 6. Consider material 8. There are two alternative tasks that can produce material 8, C and D . In fact, they both belong to $O(8)$, because $\sigma^{C,8} > 0$ and $\sigma^{D,8} > 0$. Applying the previous algorithm for C , we get

$$\sum_{g' \in R} e^{1,g'} \rho^{C,g'} = e^{1,1} \rho^{C,1} + e^{1,4} \rho^{C,4} = 1 \cdot 0.6 + 0.2 \cdot 0.4 = 0.68,$$

and for D ,

$$\sum_{g' \in R} e^{1,g'} \rho^{D,g'} = e^{1,5} \rho^{D,5} + e^{1,6} \rho^{D,6} = 0.2 \cdot 0.1 + 0 \cdot 0.9 = 0.02.$$

After considering all the possible tasks that have material 8 as an output, we can compute $e^{1,8}$ as

$$e^{1,8} = \min_{i \in O(8)} \sum_{g' \in R} e^{1,g'} \rho^{i,g'} = \min\{0.02, 0.68\} = 0.02.$$

Similarly,

$$f^{1,8} = \max_{i \in O(8)} \sum_{g' \in R} f^{1,g'} \rho^{i,g'} = \max\{0.02, 0.68\} = 0.68.$$

It is important to observe that, for the purpose of computing echelon inventory, we must consider the *mathematical* composition rather than the *physical* composition of materials. Consider the example in Figure 2. Because they are distinguished as separate materials, materials 4 and 5 likely contain different physical proportions of materials 1 and 2. However, it is impossible to produce 0.3 units of material 4 without also producing 0.7 units of material 5, and vice versa, and every unit of production of Task A requires 0.2 units of material 1 and 0.8 units of material 2. Therefore, mathematically, it is appropriate to assume that materials 4 and 5 have the same composition: 20% material 1 and 80% material 2.

Consider the alternative via an example. Suppose that, physically, material 4 is 100% composed of material 2 and, consequently, material 5 is 28.5% material 1 and 71.5% material 2 (these numbers are implied by conservation of flow). If these proportions were used to compute echelon inventory, then the information that material 1 must be used to create material 4 via Task A is lost, and any internal or external demand for material 4 has no implications for demand for material 1. In fact, even though material 4 does not physically contain material 1, material 1 must be

used to create material 4 and, therefore, any demand for material 4 induces echelon demand for material 1.

We can now compute an upper bound \hat{s}_t^r to \hat{s}_t^r of echelon inventory of material r in period t that is composed of three parts: the local inventory s_t^r of r , the fraction of the work in progress that is directly composed of r , and a proportion $f^{r,g}$ of the upper bounds of the echelon inventory of the immediate successors of r . The inventories of all successors of r are accounted for by the echelon inventory of the immediate successors of r . Formally,

$$\tilde{s}_t^r = s_t^r + \sum_{i \in N} \sum_{u=t-\alpha_i+1}^t \rho^{i,r} x_u^i + \sum_{g \in S^r(r)} f^{r,g} \tilde{s}_t^g. \quad (14)$$

If the process has a deterministic structure, it is possible to exactly compute the quantity of any material present in the system because there is only one sequence of operations that produces g starting from any material r , so the fraction of r embedded in one unit of g is unique. This leads to the following result.

REMARK 6. If the process structure is deterministic, then $\tilde{s}_t^r = \hat{s}_t^r$.

On the other hand, if the process contains multiple sequences of tasks that can be used to produce the same material, the network G would contain undirected cycles. We refer to this process structure as *choice structure*.

The definition (14) of \tilde{s}_t^r can be added to the formulation of the MPSP, and then the valid inequalities defined in the previous section can be extended to consider echelon inventory and internal demand. We are considering here the echelon inventory for material r at time $k-1$ and enforcing the following: if r is not released from production from time k to time l (thus not increasing the echelon inventory), then there must be enough echelon inventory on hand in period $k-1$ to cover the appropriate portions of demand for all successors of material r in periods k through l .

Additionally, we observe that for any time interval from k to l , part of the echelon inventory could be engaged as work in progress. If this material remains work in progress for the entire interval (i.e., the production began before period k and will not complete before period l), then this part of the echelon inventory is unavailable for fulfilling external demand. Formally, the portion of echelon inventory that will not be available to fulfill demand because it is work in progress in the time interval k to l is at least

$$\sum_{i \in N_{l-k}} \sum_{g \in R} \sum_{u=l-\alpha_i+1}^{k-1} e^{r,g} \rho^{i,g} x_u^i, \quad (15)$$

where N_{l-k} refers to the set of tasks where $\alpha_i > l-k$ for each task $i \in N_{l-k}$.

We can now extend the lemma and theorem of the previous section to consider echelon inventory.

LEMMA 7. For any $t \geq k$, let $\hat{s}_{k-1,t}^r$ represent the amount of echelon inventory of material r in period $k-1$ used to satisfy demand for r and all its successor products in period t . Then, the following set of inequalities is valid for MPSP:

$$\hat{s}_{k-1,t}^r \geq \sum_{g \in R} e^{r,g} d_t^g \left(1 - \sum_{i \in O(r)} \sum_{u=k-\alpha_i}^{t-\alpha_i} z_u^i \right) \quad \forall r \in R, k=2, \dots, T, t=k, \dots, T. \quad (16)$$

PROOF. If material r is not released from production between periods k and t (i.e., no task i that produces r was initiated within the time interval $k-\alpha_i$ to $t-\alpha_i$), then the echelon inventory level of r in the system will not increase during the interval from k to t , and therefore, the echelon inventory of material r at the end of period $k-1$ must be used to cover all demand for material r in period t , as well as at least the portion $e^{r,g}$ of demand for all successor products g of r . \square

If we consider a sequence of consecutive periods $t = k, \dots, l$, we can obtain a set of valid inequalities in the original space of variables.

THEOREM 8. The following set of inequalities is valid for the MPSP:

$$\begin{aligned} \tilde{s}_{k-1}^r \geq & \sum_{t=k}^l \left[\left(1 - \sum_{i \in O(r)} \sum_{u=k-\alpha_i}^{t-\alpha_i} z_u^i \right) \sum_{g \in R} e^{r,g} d_t^g \right] \\ & + \sum_{i \in N_{l-k}} \sum_{g \in R} \sum_{u=l-\alpha_i+1}^{k-1} e^{r,g} \rho^{i,g} x_u^i \quad \forall r \in R, k=2, \dots, T, l=k, \dots, T. \quad (17) \end{aligned}$$

PROOF. For a given k and $l \geq k$, we can sum the inequalities defined in Lemma 7 for $t = k, \dots, l$ to get

$$\sum_{t=k}^l \hat{s}_{k-1,t}^r \geq \sum_{t=k}^l \left[\sum_{g \in R} e^{r,g} d_t^g \left(1 - \sum_{i \in O(r)} \sum_{u=k-\alpha_i}^{t-\alpha_i} z_u^i \right) \right].$$

The upper bound on echelon inventory \tilde{s}_{k-1}^r of r in period $k-1$ must be at least the lower bound on the echelon inventory of r in period $k-1$ used to satisfy demand for r and its successor products in periods k, \dots, l plus the amount of r and its successor products that are work in progress for the entire interval k, \dots, l as described in (15), so (17) holds. \square

Define $L_{ech}(r, k) = \{t \mid \sum_{g \in R} e^{r,g} d_t^g > 0, t \geq k\}$ to be the set of periods in which there is positive external echelon demand for material r . With arguments similar to the ones we used in Corollary 3 for the single-level cuts, we can then state the following.

COROLLARY 9.

$$\begin{aligned} \bar{s}_{k-1}^r \geq & \sum_{t=k}^l \left[\left(1 - \sum_{i \in O(r)} \sum_{u=k-\alpha_i}^{t-\alpha_i} z_u^i \right) \sum_{g \in R} e^{r,g} d_t^g \right] \\ & + \sum_{i \in N} \sum_{g \in R} \sum_{u=l-\alpha_i+1}^{k-1} e^{r,g} \rho^{i,g} x_u^i \\ & \forall r \in R, k = 2, \dots, T, l = k, \dots, T \quad (18) \end{aligned}$$

and

$$\begin{aligned} \bar{s}_{k-1}^r \geq & \sum_{t=k}^l \left[\left(1 - \sum_{i \in O(r)} \sum_{u=k-\alpha_i}^{t-\alpha_i} z_u^i \right) \sum_{g \in R} e^{r,g} d_t^g \right] \\ & + \sum_{i \in N} \sum_{g \in R} \sum_{u=l-\alpha_i+1}^{k-1} e^{r,g} \rho^{i,g} x_u^i \\ & \forall r \in R, k = 2, \dots, T, l \in L_{ech}(r, k) \quad (19) \end{aligned}$$

are equivalent.

If demand occurs only for the final products and there is no demand for any other materials (either raw materials or intermediate products), then the single-level inequalities are simply a subset of the echelon inequalities. If demand for intermediate materials occurs, then the single-level inequalities would enforce constraints on the local inventories of these materials, and they are not generated as echelon cuts. Thus, they might tighten the formulation even further.

The coefficients of the z variables in the echelon constraints can be strengthened by recognizing that if the production of a task i that produces material g completes in period t , the production quantity of that task is limited by the machine capacity $C_{t-\alpha_i}^i$, and therefore, the echelon inventory of material r at time t increases by at most $C_{t-\alpha_i}^i \sigma^{i,r}$. This leads to the following set of valid constraints:

$$\begin{aligned} \bar{s}_{k-1}^r \geq & \sum_{t=k}^l \sum_{g \in R} e^{r,g} d_t^g - \sum_{t=k}^l \sum_{i \in O(r)} K_{i,t}^{r,l} z_t^i \\ & + \sum_{i \in N_{l-k}} \sum_{g \in R} \sum_{u=l-\alpha_i+1}^{k-1} e^{r,g} \rho^{i,g} x_u^i \\ & \forall r \in R, k = 2, \dots, T, l \in L_{ech}(r, k), \quad (20) \end{aligned}$$

where

$$K_{i,u}^{r,l} = \min \left\{ \sum_{\tau=u+\alpha_i}^l \sum_{g \in R} e^{r,g} d_\tau^g, C_u^i \sigma^{i,r} \right\}. \quad (21)$$

The above constraints have the additional advantage over those in (20) of not being trivially satisfied for some intervals $[t, k]$ when production of material r completes during that interval. We use the above constraints in our computational results reported in §6.

4.3. Preprocessing

By considering initial inventory levels, we can perform a simple preprocessing step to fix some startup variables to zero. If there is no inventory on hand for material r at the beginning of the time horizon, then any task that requires r as input cannot be initiated until sufficient time has passed for r to be produced. This “waiting period” will be at least $\alpha_{\min}^r = \min_{i \in O(r)} \alpha_i$. Therefore, we set $z_t^i = 0$ for all i such that $r \in I(i)$ and $s_0^r = 0$ and for all $t = 1, \dots, \alpha_{\min}^r$. Furthermore, if materials required as input to all tasks that produce material r are also not available in initial inventory, then the earliest starting time of a task requiring r can be similarly pushed back. Using the node-labeling technique from §4.2 and a simple depth-first search through the state-task network, the earliest start time for each task can be easily computed, and all startup variables z_t^i for periods prior to that time can be fixed to zero.

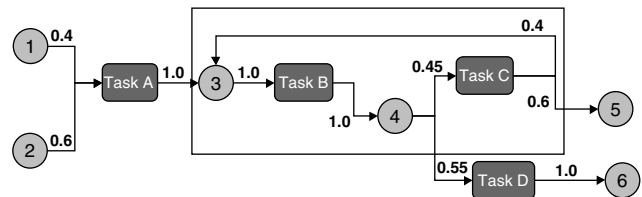
4.4. Process Structures with Reentry

Cases arise in practice where *material reentry* takes place. That is, a material that has been previously used as an input for a task is produced by the task itself or by a task using one or more of the material’s successors as input. Reentry tends to be present in the manufacturing of certain chemicals where there are processes that are able to purify a chemical from other previously used components. For example, catalysts can usually be salvaged after being used in catalysis reactions. In the state-task network representation, reentry is indicated by a directed cycle.

In general, the computation of echelon inventory, or even bounds on echelon inventory, are difficult because the amount of a predecessor material contained in one of its successors depends on how many times the predecessor material has been recycled. However, there are types of process structures with reentry for which an echelon approach could still be used, one of which is shown in Figure 3. In this case, it is still possible to identify successors and predecessors and compute fixed bounds on echelon inventory for some of the materials, both upstream and downstream of a loop.

In the example shown in Figure 3, we use the term *loop* to refer to the set of Tasks B and C, and materials 3 and 4. It is easy to see that material 3 is its own successor. The useful property exhibited by this process structure is that an intermediate product within the loop can be input to some

Figure 3. Reentry: Single-flow loop with multiple outlets.



task outside the loop (material 4 is input to Task D), and some tasks inside the loop produce materials that are never used as input within the loop (Task C produces material 5). Therefore, bounds on the amount of materials 1 and 2 contained in end products 5 and 6 can be computed, although the same is not true for materials 3 and 4, which are within the loop. Process structures where materials flow inside the loop via different intermediate materials are much more complex, and it is not clear how to extend the notion of echelon inventory in this case. We considered one problem with reentry (Multi-3) in our computational experiments, as described in §6.2.

5. Systems with Multiple Processing Units

In this section, we describe how the MPSP formulation and the additional valid inequalities can be extended to systems with multiple processing units. In systems with multiple processing units, different tasks might be carried out by different processing units, and/or the same task could be carried out by more than one processing unit.

5.1. Formulation

Let M denote the set of processing units and N_m denote the set of tasks that can be performed by processing unit $m \in M$. A task is defined in terms of inputs in specified proportions used to produce a set of outputs also in specified proportions. That is, each task is defined in terms of coefficients ($\rho^{i,r}$ and $\sigma^{i,r}$) independently of the processing units. However, the same task could have processing unit-dependent production time, α_i^m , production cost, $p_i^{i,m}$, startup cost, $g_i^{i,m}$, and production capacity, $C_i^{i,m}$.

For the MPSP with multiple processing units, which we refer to as the MPSP-m, there are four decision variables: (1) production quantity, $x_t^{i,m}$, initiated by task i in period t on processing unit m ; (2) the status of processing unit m in period t , $y_t^{i,m}$, where $y_t^{i,m} = 1$ if unit m is set up for task i in period t and zero otherwise; (3) a variable indicating the start of a task i at time t on processing unit m , $z_t^{i,m}$, where $z_t^{i,m} = 1$ if task i is initiated at time t on unit m and zero otherwise; and (4) the inventory level, s_t^r , of material r in period t .

The MPSP-m can now be formulated as follows:

$$\begin{aligned} \text{Min } & \sum_{t \in T} \sum_{r \in R} h_t^r s_t^r + \sum_{t \in T} \sum_{m \in M} \sum_{i \in N_m} p_t^{i,m} x_t^{i,m} \\ & + \sum_{t \in T} \sum_{m \in M} \sum_{i \in N_m} g_t^{i,m} z_t^{i,m} \end{aligned} \quad (22)$$

$$\text{st } s_t^r = s_{t-1}^r + \sum_{m \in M} \sum_{i \in N_m} \sigma^{i,r} x_t^{i,m} - \sum_{m \in M} \sum_{i \in N_m} \rho^{i,r} x_t^{i,m} - d_t^r \quad \forall r \in R, t \in T, \quad (23)$$

$$x_t^{i,m} \leq C_t^{i,m} z_t^{i,m} \quad \forall m \in M, i \in N_m, t \in T, \quad (24)$$

$$\sum_{u=1}^{\alpha_i^m} z_{t-u+1}^{i,m} \leq y_t^{i,m} \quad \forall m \in M, i \in N_m, t \in T, \quad (25)$$

$$z_t^{i,m} \geq y_t^{i,m} - y_{t-1}^{i,m} \quad \forall m \in M, i \in N_m, t \in T, \quad (26)$$

$$\sum_{i \in N_m} y_t^{i,m} = 1 \quad \forall m \in M, t \in T, \quad (27)$$

$$x_t^{i,m} \geq 0 \quad \forall m \in M, i \in N_m, t \in T, \quad (28)$$

$$s_t^r \geq 0 \quad \forall r \in R, t \in T, \quad (29)$$

$$y_t^{i,m}, z_t^{i,m} \in \{0, 1\} \quad \forall m \in M, i \in N_m, t \in T. \quad (30)$$

The objective function and constraints maintain the same interpretation they have in the case of a single processing unit (see §3). Note also that the flow of material can still be described in terms of the directed graph G described in §3 because tasks (and not processing units) transform materials from one state to another.

5.2. Valid Inequalities

The single-level (13) and echelon cuts (19) can be extended to the case with multiple processing units. The following set of single-level inequalities are valid for the MPSP-m:

$$\begin{aligned} s_{k-1}^r \geq \sum_{t=k}^l \left[d_t^r \left(1 - \sum_{m \in M} \sum_{i \in O(r) \cap N_m} \sum_{u=k-\alpha_i^m}^{t-\alpha_i^m} z_u^{i,m} \right) \right] \\ \forall r \in R, k = 2, \dots, T, l = k, \dots, T. \end{aligned} \quad (31)$$

Consistent with Corollary 3, we can restrict the set of cuts defined above as follows:

$$\begin{aligned} s_{k-1}^r \geq \sum_{t=k}^l \left[d_t^r \left(1 - \sum_{m \in M} \sum_{i \in O(r) \cap N_m} \sum_{u=k-\alpha_i^m}^{t-\alpha_i^m} z_u^{i,m} \right) \right] \\ \forall r \in R, k = 2, \dots, T, l \in L(r, k), \end{aligned} \quad (32)$$

where $L(r, k) = \{t \mid d_t^r > 0 \text{ and } t \geq k\}$.

An upper bound on echelon inventory can be obtained similarly to Equation (14) as follows:

$$\bar{s}_t^r = s_t^r + \sum_{m \in M} \sum_{i \in N_m} \sum_{u=t-\alpha_i^m+1}^t \rho^{i,r} x_u^{i,m} + \sum_{g \in S^r(r)} f^{r,g} \bar{s}_t^g, \quad (33)$$

with the portion of echelon inventory not available from periods k to l given by

$$\sum_{m \in M} \sum_{i \in N_m} \sum_{g \in R} \sum_{u=l-\alpha_i^m+1}^{k-1} e^{r,g} \rho^{i,g} x_u^{i,m}. \quad (34)$$

The echelon cuts (17) can be modified as follows:

$$\begin{aligned} \bar{s}_{k-1}^r \geq \sum_{t=k}^l \left[\left(1 - \sum_{m \in M} \sum_{i \in O(r) \cap N_m} \sum_{u=k-\alpha_i^m}^{t-\alpha_i^m} z_u^{i,m} \right) \sum_{g \in R} e^{r,g} d_t^g \right] \\ + \sum_{m \in M} \sum_{i \in N_m} \sum_{g \in R} \sum_{u=l-\alpha_i^m}^{k-1} e^{r,g} \rho^{i,g} x_u^{i,m} \\ \forall r \in R, k = 2, \dots, T, l = k, \dots, T. \end{aligned} \quad (35)$$

Again consistent with Corollary 9, the above constraints are equivalent to the following:

$$\begin{aligned} \bar{s}_{k-1}^r \geq & \sum_{t=k}^l \left[\left(1 - \sum_{m \in M} \sum_{i \in O(r) \cap N_m} \sum_{u=k-\alpha_i^m}^{t-\alpha_i^m} z_u^{i,m} \right) \sum_{g \in R} e^{r,g} d_t^g \right] \\ & + \sum_{m \in M} \sum_{i \in N_m} \sum_{g \in R} \sum_{u=l-\alpha_i^m+1}^{k-1} e^{r,g} \rho^{i,g} x_u^{i,m} \\ & \forall r \in R, k = 2, \dots, T, l \in L_{ech}(r, k), \end{aligned} \quad (36)$$

where $L_{ech}(r, k) = \{t \mid \sum_{g \in R} e^{r,g} d_t^g > 0, t \geq k\}$.

Following a reasoning similar to that done for the single-machine case, we can further strengthen Equation (36) by defining $K_{i,m,u}^{r,l}$ as the coefficient of $z_u^{i,m}$ in the echelon inequality that considers material r and an interval up to time l . Thus, for all $z_u^{i,m}$ with $i \in O(r)$, the portion of echelon demand covered by the z -variable is better bounded by the following:

$$K_{i,m,u}^{r,l} = \min \left\{ \sum_{\tau=u+\alpha_i}^l \sum_{g \in R} e^{r,g} d_\tau^g, C_u^{i,m} \sigma^{i,r} \right\}. \quad (37)$$

6. Computational Results

We first present computational results involving problem instances with a single processing unit. We then discuss results for problems with multiple processing units.

6.1. Problems with a Single Processing Unit

We tested the effectiveness of our cuts on a series of instances with varying sizes and characteristics. Results from 10 representative problems are discussed in this section. As shown in Table 1, the problems vary by number of materials (raw materials, intermediate products, and end products), number of tasks, number of stages (maximum number of sequential tasks needed to produce an end product), and process structure. We consider problems

Table 1. Single processing unit problem instances.

Problem	Materials	Tasks	Stages	Time horizon	Process structure
1	9	8	9	{50}	Series
2	13	12	13	{50}	Series
3	15	14	15	{50}	Series
4	13	7	4	{50}	Strictly convergent
5	13	7	4	{50}	Strictly divergent
6	9	4	4	{40, 50, 80, 120}	General deterministic network
7	13	8	6	{40, 50, 80, 120}	General deterministic network
8	13	8	6	{40, 50, 80, 120}	General deterministic network
9	13	9	6	{40, 50, 70, 100}	General choice network
10	13	9	6	{40, 50, 70, 100}	General choice network

with five different process structures: series, strictly convergent, strictly divergent, general deterministic network, and general choice network. A series structure refers to systems where each material has a unique immediate successor and a unique immediate predecessor and each material is produced by a single task. A strictly convergent structure (similar to an assembly structure) refers to systems where materials might have multiple immediate predecessors but a unique immediate successor. A strictly divergent structure (similar to a disassembly structure) refers to systems where each material has unique immediate predecessors but might have multiple immediate successors. For both strictly convergent and divergent structures, each material is produced by a single task. Systems with a general network structure might have tasks with multiple inputs and multiple outputs forming an arbitrary network. Depending on whether the network is choice or deterministic, the same material might or might not be produced by more than one task. Holding costs for various materials have been randomly generated such that downstream materials have higher holding costs than upstream materials. Production costs are also randomly generated. In this case, we always ensure that tasks with a larger number of input and/or output materials have higher production costs. Demands for each problem have been generated so that problems are feasible. However, capacity loading is varied from problem to problem. Tightly capacitated instances are Problem 7 (50-period and 80-period instance), Problem 8 (50-period and 80-period instance), and Problem 10 (all instances). The full data for each problem, along with MPS files, are available from the authors upon request.

We solved each instance with two methods. First, we solved the original formulation using the commercial solver CPLEX version 8.1, with all the default settings and with the standard cuts turned on. In the second method, we first applied the preprocessing technique to preassign some variables to zero. Then, we generated all echelon inequalities and added them as model cuts to CPLEX's *cutpool*. CPLEX, with its default settings and cuts and the new echelon cuts, was then used to obtain an optimal solution.

Table 2 shows the optimal objective value of the LP relaxation for both the original formulation (denoted LP) and the formulation with the echelon inequalities (LP_{cuts}), as well as the objective value of the best-known integer solution (IP). Values in column IP marked with an asterisk have not been proven to be optimal. Also shown is the ratio $[(LP_{cuts} - LP)/(IP - LP) \times 100\%]$. This ratio measures the percentage reduction in the gap in cost between the best-known IP solution and the LP relaxation solution. As can be seen, the gap is reduced by over 60% in all instances for which an optimal IP solution is obtained. For those instances where optimality was not proved, the value shown is a lower bound on the actual gap reduction.

Table 3 shows the CPU time and number of branch-and-bound nodes needed by CPLEX to prove optimality for both the original formulation and the formulation with

Table 2. Percentage reduction in the integrality gap.

Problem	Time horizon	LP	LP _{cuts}	IP	Percentage gap reduction (%)
1	50	13,549.2	22,894.6	24,521.0	85.18
2	50	23,793.3	60,412.7	67,942.0	82.95
3	50	22,554.3	48,061.9	54,762.0	79.20
4	50	12,607.0	33,689.1	35,919.7	90.43
5	50	73,024.8	139,372.8	156,548.4	79.44
6 (9/4/4)	40	9,525.3	16,748.7	17,931.2	85.93
	50	16,871.0	27,650.5	30,689.4	78.01
	80	37,453.2	51,284.3	55,088.5	78.43
	120	80,662.3	104,621.7	112,524.1*	≥75.20
7 (13/8/6)	40	26,111.5	50,605.0	56,666.9	80.16
	50	45,719.8	80,073.3	93,609.4	71.73
	80	99,894.5	157,144.5	181,577.5*	≥70.09
	120	95,068.9	165,646.2	237,361.8*	≥49.60
8 (13/8/6)	40	25,633.9	50,442.9	56,805.0	79.59
	50	44,714.9	78,967.5	92,607.6	71.52
	80	94,677.7	153,124.7	190,800.8*	≥60.80
	120	134,106.4	206,287.7	254,590.7*	≥59.91
9 (13/9/6)	40	14,694.7	40,750.1	56,961.7	62.04
	50	22,409.3	57,508.7	77,346.1	63.89
	70	22,926.3	61,494.2	117,661.4*	≥40.71
	100	30,008.0	92,640.6	148,483.7*	≥52.87
10 (13/9/6)	40	14,873.4	43,277.0	61,066.8	61.49
	50	22,409.3	57,508.7	77,346.1	63.89
	70	21,857.1	72,305.1	106,645.8*	≥59.50
	100	31,367.2	94,955.8	162,440.3*	≥48.89

*Optimality was not proved.

the echelon cuts. Also shown is the percent reduction in required CPU time $[(MILP - MILP_{cuts})/MILP \times 100\%]$, where MILP ($MILP_{cuts}$) denotes the amount of time taken by CPLEX to obtain an optimal solution using the original formulation (formulation with echelon cuts). An asterisk flags instances where optimality was not proved within the time limit we assigned (25,000 seconds) with either approach.

As we can see from Table 3, the addition of the echelon cuts reduces solution time for all instances except for those with the simple series structure and for the smallest instance with a general deterministic structure. The solution time is reduced by one order of magnitude for the more complex instances with either a larger number of stages, a longer planning horizon, or tighter capacity. We suspect the improved performance for tightly capacitated instances follows from two reasons. First, capacity plays a role in strengthening the echelon cuts, as described in Equation (20). Also, it is inherently more difficult to find feasible solutions to these instances, and the inequalities guide the solver to a feasible solution more quickly, thus keeping the size of the branch-and-bound tree from growing too large. The poor performance in the simple instances is likely due to the small size of the problems—both methods obtain an optimal solution in just few seconds. In this case, the additional cuts tend to hurt more than help;

Table 3. Reduction in computational time.

Problem	Time horizon	Original formulation		Echelon cuts added		Percentage of reduction in CPU time (%)
		CPU time (seconds)	Number of nodes	CPU time (seconds)	Number of nodes	
1	50	6.61	108	7.10	11	-7.41
2	50	9.10	162	15.21	49	-67.14
3	50	9.21	48	14.01	12	-52.12
4	50	561.94	21,308	115.17	163	79.50
5	50	2,777.05	14,578	564.89	1,143	79.66
6	40	1.72	86	3.10	25	-80.23
	50	19.64	1,453	27.43	566	-39.66
	80	254.48	9,923	314.02	4,041	-23.40
	120	25,000*	509,620	25,000*	77,097	—
7	40	115.36	2,489	39.14	417	66.07
	50	5,283.44	74,491	313.18	1,861	94.07
	80	25,000*	137,419	25,000*	38,234	—
	120	25,000*	108,371	25,000*	8,760	—
8	40	259.60	7,192	36.74	371	85.85
	50	2,531.17	40,068	189.54	649	92.51
	80	25,000*	112,569	25,000*	22,592	—
	120	25,000*	46,889	25,000*	8,717	—
9	40	1,668.74	33,853	176.3	1,054	89.44
	50	25,000*	240,760	1,231.21	2,780	≥95.07
	70	25,000*	80,975	25,000*	7,741	—
	100	25,000*	91,522	25,000*	7,806	—
10	40	3,310.68	53,385	387.27	2,685	88.30
	50	25,000*	219,224	1,061.34	2,780	≥95.75
	70	25,000*	78,300	25,000*	8,114	—
	100	25,000*	13,300	25,000*	3,545	—

*Optimality was not proved within 25,000 seconds.

Table 4. Best-known integer solutions.

Problem	Time horizon	Best-known integer solution	Original formulation		Echelon cuts added	
			Obj. val.	Gap (%)	Obj. val.	Gap (%)
6	120	112,524.0	113,507.3	5.52	112,524.0	2.41
7	80	181,577.5	186,554.1	17.83	181,577.5	6.00
	120	237,361.7	244,385.4	37.26	237,361.7	27.59
8	80	190,800.8	194,326.5	23.94	190,800.8	12.73
	120	254,590.6	276,026.0	29.31	254,590.6	15.16
9	70	118,951.2	127,080.4	51.68	118,951.2	36.94
	100	148,483.6	166,787.9	55.43	161,165.5	45.79
10	70	106,645.8	NA	...	106,645.8	20.54
	100	161,436.8	NA	...	161,436.8	37.31

this effect disappears as problem instances become larger. Also seen in Table 3, there are two instances (Problems 9 and 10 with 50 time periods) for which only the formulation with echelon cuts finds the optimal solution within 25,000 seconds.

For the instances where an optimal solution cannot be obtained using either formulation within 25,000 seconds, we performed further analysis. Table 4 shows the best integer solutions found using both the original formulation and the formulation with the echelon cuts within the assigned time limit of 25,000 seconds. The table also shows the ratios $GAP_{original} = [(SOL_{original} - SOL_{best})/SOL_{best} \times 100\%]$ and $GAP_{cuts} = [(SOL_{cuts} - SOL_{best})/SOL_{best} \times 100\%]$, where $SOL_{original}$ (SOL_{cuts}) refers to the cost of the best solution found within 25,000 seconds using the original formulation (formulation with echelon cuts) and SOL_{best} is the cost of the best-known integer solution. In each case, the formulation with the echelon cuts always finds a better integer feasible solution than the original formulation, and the final integrality gap is always tighter with the echelon-cut approach. More remarkably, when the echelon cuts are used, we find feasible solutions for two instances of Problem 10 when the default method could not find any within 25,000 seconds. Moreover, we observed that feasible solutions were always found in a shorter amount of time when using the formulation with echelon cuts.

In all the instances we tested, the formulation with echelon cuts led to a significantly fewer number of nodes being explored in the branch-and-bound tree generated by CPLEX. This is due to the better LP relaxations for each node subproblem and to the earlier identification of feasible integer solutions.

To test how solution times might be affected by various cost parameters, we performed a sensitivity analysis with respect to holding, setup, and production costs. The primary purpose of the analysis was to see if the ratios between holding, setup, and production costs have an effect on problem difficulty. In conducting the tests, we first generated five instances of Problem 7 from above with a time horizon of 50 periods, using the same probability distributions

for the cost parameters. For each cost parameter (holding, setup, production), we multiplied the costs by various scaling factors for each base instance and then solved the instance using both the default and echelon-cut approaches. For each solution approach, we then averaged the solution times of all five instances for each choice of cost parameters. The solution times when the echelon cuts were used were consistently low, always less than a few hundred seconds, while the solution times using only the initial formulation fluctuated significantly. For every instance, using the echelon cuts solved the instance faster than CPLEX solved the initial formulation.

In Figure 4, we show the impact of varying holding cost on CPU time for both approaches. We varied holding costs for each task by scaling the costs of a base case by a factor α ranging from 0.001 to 10,000. The results show that for the range of costs tested, the formulation with echelon cuts remains superior, with the difference in CPU time being in excess of 60% in all cases. The absolute difference is smallest when holding costs are very low. This makes sense because when holding costs are low, the tasks tend to run at full capacity, which forces the setup variables z_i^j to be naturally integer in the original formulation. Results for setup

Figure 4. Sensitivity analysis: Solution time vs. change in holding costs.

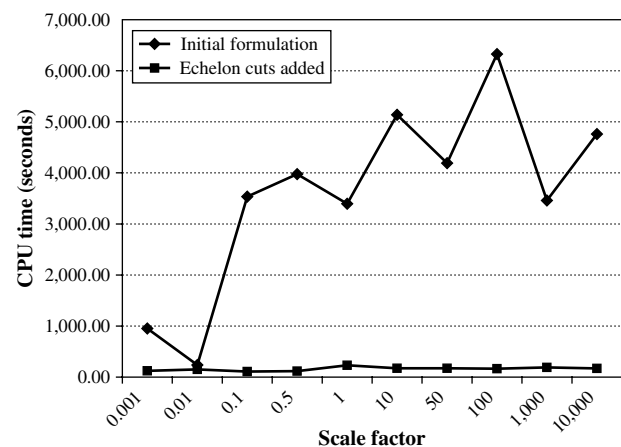
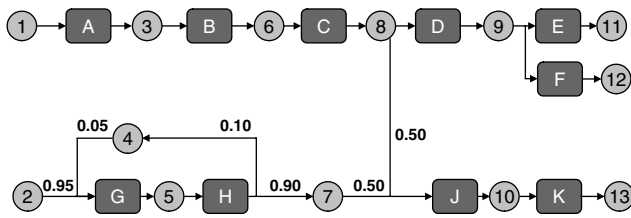


Figure 5. Problem Multi-3 process structure.



and production costs were similar. When setup costs are very high, tasks tend to run at full capacity, and the original formulation can be solved more quickly. The effects of changes to production costs are less clear, although solution times tended to decrease slightly as production costs became very large. This is likely because when production costs are high, the LP relaxation of the original formulation tends to yield a tight lower bound on the optimal cost because total production quantities are relatively constant across all feasible solutions and are represented by continuous variables. In summary, solution times for the original formulation were highly dependent on the cost parameters, while the solution method using echelon cuts performed consistently well regardless of the values of the cost coefficients.

6.2. Problems with Multiple Processing Units

We tested the effectiveness of the echelon cuts on three problem instances with multiple processing units. Process structures for Problems Multi-1 and Multi-2 are taken from Sahinidis and Grossman (1991) (Problem BATCH3) and Papageorgiou and Pantelides (1996b) (Problem Example 2), respectively, but different time horizons and demand profiles have been tested here. Both problems consider three parallel flow lines with one raw material, one or two intermediate materials, and one end product for each line.

Tasks at the same processing level share a processing unit, and each processing unit is dedicated to one level only. Demand and cost parameters were randomly generated for each instance. A second set of problems, denoted Multi-1a and Multi-2a, was generated by increasing the demand profile for each instance by 50%. Problem Multi-3 has a deterministic process structure with product recycling at upstream levels. There are 13 materials, 10 tasks, 6 processing units, and 6 levels. This problem is modified from

Table 5. Multiple processing unit problem instances.

Problem	Materials	Tasks	Stages	Processing units	Time horizon
Multi-1	9	4	3	4	{80}
Multi-2	12	8	4	3	{80}
Multi-3	13	10	6	6	{50}

Example 1 in Papageorgiou and Pantelides (1996b); see Figure 5. The characteristics of the three problems are summarized in Table 5.

Table 6 shows representative results from each class of problems that were solved with and without the echelon cuts. As with problems with a single processing unit, the echelon cuts can have a dramatic effect on computational performance. In some cases, optimality could be proved only within the specified time limit of 25,000 seconds when the echelon cuts were used.

6.3. Comparison to Other Formulations

We performed tests comparing our problem formulation to two alternative MILP formulations. The first, originally proposed by Kondili et al. (1993) (see also Shah et al. 1993), we refer to as the KPS formulation, and the second, originally proposed by Sahinidis and Grossman (1991), we refer to as the SG formulation. For brevity, the formulations are not reproduced here. Our objective is to (1) test whether or not the echelon cuts are still effective when applied to alternative formulations, and (2) compare the performance of the cuts applied to our formulation to their performance when applied to alternative formulations.

We implemented both the KPS and SG formulations using CPLEX and compared solution quality and solution times obtained with and without the echelon cuts for Problems 5–10. The results are consistent with those obtained using our formulation: when applied to alternate formulations, the echelon cuts continue to be useful. For brevity, the results are not included but are available from the authors upon request. Table 7 compares the computational time needed to solve these 20 instances using the MPSP, KPS, and SG formulations, each with the echelon cuts applied. In all cases, the MPSP formulation outperforms the others by either solving the instance to optimality more

Table 6. Reduction in computational time (multiple processing units).

Problem	Time horizon	Original formulation		Echelon cuts added		Percentage of reduction in CPU time (%)
		CPU time (seconds)	Number of nodes	CPU time (seconds)	Number of nodes	
Multi-1	80	25,000*	6,128,791	55.51	1,796	≥99.78
Multi-2	80	25,000*	76,003	645.68	1,285	≥97.42
Multi-1a	80	25,000*	6,307,237	7,068.47	793,506	≥71.73
Multi-2a	80	3,421.57	23,708	51.42	47	98.50
Multi-3	50	344.44	126,707	114.33	17,229	66.81

*Optimality was not proved within 25,000 seconds.

Table 7. Comparison of MPSP, KPS, and SG formulations with echelon cuts.

Problem	Time horizon	MPSP formulation			KPS formulation			SG formulation			MPSP vs. KPS formulation			MPSP vs. SG formulation			
		CPU time (seconds)	Final gap (%)		CPU time (seconds)	Final gap (%)		CPU time (seconds)	Final gap (%)		% Reduction in CPU time (%)	% Reduction in gap (%)		% Reduction in CPU time (%)	% Reduction in gap (%)		
6	40	3.10	—	1.31	—	1.05	—	—	—	—	—	—	—	—	—	—	
	50	27.43	—	38.78	—	35.54	—	—	—	—	—	—	—	—	—	—	
	80	314.02	—	782.27	—	399.40	—	—	—	—	—	—	—	—	—	—	
7	120	25,000*	2.41	25,000*	5.01	25,000*	3.83	—	—	—	—	—	—	—	—	37.15	
	40	39.14	—	88.86	—	26.62	—	—	—	—	—	—	—	—	—	—	
	50	313.18	—	2,064.07	—	390.79	—	—	—	—	—	—	—	—	—	—	
	80	25,000*	6.00	25,000*	**	25,000*	**	—	—	—	—	—	—	—	—	—	100
	120	25,000*	27.59	25,000*	31.16	25,000*	31.58	—	—	—	—	—	—	—	—	—	12.64
8	40	36.74	—	75.41	—	40.65	—	—	—	—	—	—	—	—	—	—	
	50	189.54	—	2,275.37	—	298.35	—	—	—	—	—	—	—	—	—	—	
	80	25,000*	12.73	25,000*	**	25,000*	13.81	—	—	—	—	—	—	—	—	—	7.78
	120	25,000*	15.16	25,000*	36.42	25,000*	17.15	—	—	—	—	—	—	—	—	—	11.59
9	40	176.30	—	451.21	—	60.93	—	—	—	—	—	—	—	—	—	—	
	50	1,231.21	—	12,567.35	—	2,534.13	—	—	—	—	—	—	—	—	—	—	
	70	25,000*	36.94	25,000*	39.74	25,000*	48.03	—	—	—	—	—	—	—	—	—	23.09
	100	25,000*	45.79	25,000*	50.12	25,000*	57.62	—	—	—	—	—	—	—	—	—	20.53
10	40	387.27	—	912.73	—	57.57	—	—	—	—	—	—	—	—	—	—	
	50	1,061.34	—	10,356.82	—	89.75	—	—	—	—	—	—	—	—	—	—	
	70	25,000*	20.54	25,000*	**	25,000*	31.24	—	—	—	—	—	—	—	—	—	34.25
	100	25,000*	37.31	25,000*	51.43	25,000*	27.45	—	—	—	—	—	—	—	—	—	100

* Optimality was not proved within 25,000 seconds.
 ** No feasible integer solution was found within 25,000 seconds.

quickly or finding a feasible solution with a smaller integrality gap within the time limit of 25,000 seconds.

7. Summary and Future Extensions

We considered a multitask/multistage production planning and scheduling problem (MPSP) found in process industries and formulated it as a mixed-integer program. We used the notion of echelon inventory to construct valid inequalities. Numerical experiments show that the echelon inequalities can significantly reduce the solution time needed to find optimal solutions or finds better feasible solutions within a fixed timeframe. This is particularly evident in problems with relatively complex process structures or long planning horizons. Therefore, this approach might be useful as a stand-alone tool in situations with complex process structures where good, but not necessarily optimal, solutions are desired, or as a subroutine within heuristics for solving large and/or complex problems.

There are several possible extensions worth exploring. We name three that we have started to examine: (a) treating problems with sequence-dependent setup costs, (b) allowing for the possibility of backorders, and (c) developing decomposition heuristics to solve large-scale problems. We offer brief comments on each.

Sequence-dependent setup costs can be included in our formulation without a considerable increase in complexity. For example, we might introduce the changeover variables $\chi_t^{ij,m}$, which take value one if task j is initiated at time t on processing unit m when the status for unit m is set to task i at time $t - 1$, and zero otherwise. In this case, a changeover cost is incurred to reflect the additional cost due to performing these tasks in sequence. The status variables $y_t^{i,m}$ allow us to make use of the reformulation for sequence-dependent changeover variables proposed by Karmarkar and Schrage (1985), which has the form of a network flow problem. The additional constraints form a totally unimodular matrix, and therefore the integrality of the changeover variables follows from the integrality of the status variables. Therefore, no new integer variables are added to the formulation. Furthermore, the form and validity of the echelon cuts is unaffected by the introduction of the changeover variables.

Allowing for backorders in the problem formulation is straightforward. However, backordering could invalidate the echelon cuts because they use the fact that demand must be satisfied on time. Two important exceptions are cases where backordering is allowed only within a time window or is limited to a fixed amount. In the first case, the same echelon cuts can be used by simply shifting the due date for demand in each period by the length of the time window. In the second case, echelon cuts can be used with respect to the difference between demand in each period and the maximum amount that can be backordered. In the more challenging case where unlimited backorders are allowed, it is possible to use a notion of echelon inventory to relate the

amount that is not backordered (a decision variable) with the production startup variables. Unfortunately, this gives rise to nonlinear constraints, which to be useful would have to be either linearized or bounded by linear constraints.

For problems involving a large number of periods, stages, or materials, reaching an optimal solution sufficiently quickly can be difficult even when the echelon cuts are used. For these large problems, some form of decomposition (e.g., solving a series of problems over shorter planning horizons) can become necessary. The echelon cuts, however, remain useful even when such decomposition is used. In particular, for each subproblem, echelon cuts can be used to speed solution time or improve solution quality.

Acknowledgments

The authors thank Tingliang Huang for performing some of the computational experiments presented in §6. They also thank two anonymous referees, whose valuable comments greatly improved the paper.

References

- Afentakis, P., B. Gavish. 1986. Optimal lot-sizing algorithms for complex product structures. *Oper. Res.* **34**(2) 237–249.
- Applequist, G., O. Samikoglu, J. Pekny, G. Reklaitis. 1997. Issues in the use, design and evolution of process scheduling and planning systems. *ISA Trans.* **36**(2) 81–121.
- Arkin, E., D. Joneja, R. Roundy. 1989. Computational complexity of uncapacitated multi-echelon production planning problems. *Oper. Res. Lett.* **8**(2) 61–66.
- Atamtürk, A., J. C. Muñoz. 2004. A study of the lot-sizing polytope. *Math. Programming* **99**(3, Ser. A) 443–465.
- Barany, I., T. J. V. Roy, L. A. Wolsey. 1984. Strong formulations for multi-item capacitated lot sizing. *Management Sci.* **30**(10) 1255–1261.
- Belvaux, G., L. A. Wolsey. 2000. *bc-prod*: A specialized branch-and-cut system for lot-sizing problems. *Management Sci.* **46** 724–738.
- Belvaux, G., L. A. Wolsey. 2001. Modelling practical lot-sizing problems as mixed-integer programs. *Management Sci.* **47**(7) 993–1007.
- Bitran, G., H. Yanasse. 1982. Computational complexity of the capacitated lot size problem. *Management Sci.* **28** 1174–1185.
- Clark, A., H. Scarf. 1960. Optimal policies for a multi-echelon inventory problem. *Management Sci.* **6** 475–490.
- Constantino, M. 1996. A cutting plane approach to capacitated lot-sizing with start-up cost. *Math. Programming* **75** 353–376.
- Elkamel, A., M. Zentner, J. Pekny, G. Reklaitis. 1997. A decomposition heuristic for scheduling the general batch chemical plant. *Engrg. Opt.* **28**(4) 299–330.
- Florian, M., J. K. Lenstra, A. H. G. Rinnooy Kan. 1980. Deterministic production planning: Algorithms and complexity. *Management Sci.* **26** 669–679.
- Ierapetritou, M., C. Floudas. 1998. Effective continuous-time formulation for short-term scheduling. 1. Multipurpose batch processes. *Indust. Engrg. Chemistry Res.* **37** 4341–4359.
- Kallrath, J. 2003. Planning and scheduling in the process industry. H. O. Gunther, P. van Beek, eds. *Advanced Planning and Scheduling Solutions in Process Industry*. Springer-Verlag, Berlin, 201–227.
- Karmarkar, U., K. Rajaram. 2001. Grade selection and blending to optimize cost and quality. *Oper. Res.* **49**(2) 271–280.
- Karmarkar, U., L. Schrage. 1985. The deterministic dynamic product cycling problem. *Oper. Res.* **33**(2) 326–345.

- Katok, E., H. S. Lewis, T. P. Harrison. 1998. Lot-sizing in general assembly systems with setup costs, setup times, and multiple constrained resources. *Management Sci.* **44**(6) 859–877.
- Kondili, E., C. C. Pantelides, R. W. H. Sargent. 1993. A general algorithm for short-term scheduling of batch operations—I. MILP formulation. *Comp. Chemical Engrg.* **17** 211–227.
- Lin, X., C. Floudas, S. Modi. 2002. Continuous-time optimization approach for medium-range production scheduling of a multiproduct batch plant. *Indust. Engrg. Chemistry Res.* **41** 3884–3906.
- Magnanti, T. L., R. Vachani. 1990. A strong cutting plane for production scheduling with changeover costs. *Oper. Res.* **38**(3) 456–473.
- Majozi, T., X. X. Zhu. 2001. A novel continuous-time MILP formulation for multipurpose batch plants. I. Short-term scheduling. *Indust. Engrg. Chemistry Res.* **40**(25) 5935–5949.
- Maravelias, C. T., I. Grossman. 2003a. New general continuous-time state-task network formulation for short-term scheduling of multipurpose batch plants. *Indust. Engrg. Chemistry Res.* **42** 3056–3074.
- Maravelias, C. T., I. Grossman. 2003b. Minimization of the makespan with a discrete-time state-task network formulation. *Indust. Engrg. Chemistry Res.* **42** 6252–6257.
- Miller, A. J., L. A. Wolsey. 2003. Tight MIP formulations for multi-item discrete lot sizing problems. *Oper. Res.* **51**(4) 557–605.
- Miller, A. J., G. L. Nemhauser, M. W. P. Savelsbergh. 2003. A multi-item production planning model with setup times: Algorithms, reformulations, and polyhedral characterizations for a special case. *Math. Programming* **95**(1) 71–90.
- Mockus, L., G. V. Reklaitis. 1999. Continuous time representation approach to batch and continuous process scheduling. I. MINLP formulation. *Indust. Engrg. Chemistry Res.* **38**(1) 197–203.
- Neumann, K., C. Schwindt, N. Trautmann. 2003. Advanced production scheduling for batch plants in process industries. H. O. Gunther, P. van Beek, eds. *Advanced Planning and Scheduling Solutions in Process Industry*. Springer-Verlag, Berlin, 43–71.
- Pantelides, C. C. 1994. Unified frameworks for the optimal process planning and scheduling. *Proc. Conf. Comput. Aided Process Oper. (FOCAPO)*, Cache Publications, New York, 253–274.
- Papageorgiou, L. G., C. C. Pantelides. 1996a. Optimal campaign planning/scheduling of multipurpose batch/semicontinuous plants. I. Mathematical formulation. *Indust. Engrg. Chemistry Res.* **35** 488–509.
- Papageorgiou, L. G., C. C. Pantelides. 1996b. Optimal campaign planning/scheduling of multipurpose batch/semicontinuous plants. 2. Computational issues. *Indust. Engrg. Chemistry Res.* **35** 510–529.
- Pekny, J. F. 2002. Algorithm architectures to support large-scale process systems engineering applications involving combinatorics, uncertainty, and risk management. *Comput. Chemical Engrg.* **26** 239–267.
- Pochet, Y., L. A. Wolsey. 1991. Multi-item lot-sizing problems using strong cutting planes. *Management Sci.* **37** 53–67.
- Sahinidis, N. V., I. E. Grossman. 1991. Reformulation of multiperiod MILP models for planning and scheduling of chemical processes. *Comp. Chemical Engrg.* **15**(4) 255–272.
- Schilling, G., C. C. Pantelides. 1996. A simple continuous-time process scheduling formulation and a novel solution algorithm. *Comput. Chemical Engrg.* **20**(4) S1221–S1226.
- Shah, N. 1998. Single- and multisite planning and scheduling: Current status and future challenges. *AIChE Sympos. Ser.* **94**(320) 75–90.
- Shah, N., C. C. Pantelides, R. W. H. Sargent. 1993. A general algorithm for short-term scheduling of batch operations—II. Computational issues. *Comput. Chemical Engrg.* **17** 229–244.
- Stadtler, H. 2003. Multilevel lot-sizing with setup times and multiple constrained resources: Internally rolling schedules with lot-sizing windows. *Oper. Res.* **51**(3) 487–502.
- Tempelmeier, H., M. Destroff. 1996. A Lagrangean-based heuristic for dynamic multilevel multiitem constrained lot-sizing in general assembly systems with setup times. *Management Sci.* **42**(5) 738–757.
- van den Heever, S. A., I. Grossman. 1999. Disjunctive multiperiod optimization methods for design and planning of chemical process systems. *Comput. Chemical Engrg.* **23**(3) 1075–1095.
- Wolsey, L. A. 1997. MIP modelling of changeovers in production planning and scheduling problems. *Eur. J. Oper. Res.* **99** 154–165.
- Wolsey, L. A. 2002. Solving multi-item lot-sizing problems with an MIP solver using classification and reformulation. *Management Sci.* **48**(12) 1587–1602.
- Zhang, X., R. W. H. Sargent. 1996. The optimal operation of mixed production facilities—A general formulation and some approaches for the solution. *Comput. Chemical Engrg.* **20**(6–7) 897–904.
- Zipkin, P. H. 2000. *Foundations of Inventory Management*. McGraw-Hill, New York.