Primary pharmaceutical manufacturing scheduling problem

SUBHASH C. SARIN*, HANIF D. SHERALI and LINGRUI LIAO
Grado Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, VA 24060, USA
E-mail: sarins@vt.edu

Received June 2011 and accepted October 2013

This article addresses an integrated lot-sizing and scheduling problem that arises in the primary manufacturing phase of a pharmaceutical supply chain. Multiple pharmaceutical ingredients and their intermediate products are to be scheduled on parallel and capacitated bays for production in batches. Sequence-dependent setup times and costs are incurred when cleaning a bay during changeovers between different product families. The problem also contains a high multiplicity asymmetric traveling salesman-type of substructure because of sequence-dependent setups and special restrictions. Mixed-integer programming formulations are proposed for this problem and several valid inequalities are developed to tighten the model. A column generation method along with a decomposition scheme and an advanced-start solution are designed to efficiently derive good solutions to this highly complex problem. A computational investigation is performed, based on instances that closely follow a real-life application, and it demonstrates the efficacy of the proposed solution approach.

Keywords: Lot-sizing, batch production, high multiplicity asymmetric traveling salesman problem, column generation

1. Introduction

A pharmaceutical supply chain contains two key manufacturing phases: primary manufacturing and secondary manufacturing. The primary phase produces Active Pharmaceutical Ingredients (APIs), which are the active substances constituting a drug. The secondary phase involves the mixing of APIs with inert materials to make final products. The competitiveness of a pharmaceutical supply chain heavily depends on the performance of its internal business processes (Shah, 2004), where the production control of primary manufacturing directly dictates its responsiveness to demands from the secondary phase. We designate this production control problem as the Primary Pharmaceutical Manufacturing Scheduling Problem (PPMSP).

1.1. Problem description

Production in the primary pharmaceutical manufacturing phase involves multiple products, flexible machines (bays) that work in parallel, and demands with different due dates. We address the problem of sequencing production of active pharmaceutical ingredients for this phase of manufacturing within processing bays in response to external demand (generated from the secondary phase). Each product involves a product-specific processing time and requires a sequence-dependent changeover time and Bill Of Materials (BOM) for its production. To facilitate production and optimization, we divide time into “big-bucket” periods that permit the production of (possibly) several products in a bay during a period. Our objective is to minimize the total cost incurred because of inventory holding and production setup times.

A typical primary manufacturing facility hosts multiple processing bays, each of which consists of a serial arrangement of four staged units: reactor, centrifuge, crystallizer, and dryer. These processing stages are equipped with containers of limited capacities and operate with discrete batches. Since there is no separate storage space between these containers, the production of a product within a bay has to be carried out without any interruption. Hence, we regard the entire bay as one processing device that works in a batch production mode. At any given time, a bay can be occupied only by a single batch for the production of a single product. The maximal size of a batch is limited by the capacity of the containers; however, maximal batch sizes of different products can differ for a given bay because of product-specific restrictions. Due to the inherent nature of the chemical/physical processes involved, the processing time of a batch is independent of its size, but it does vary across different products and bays.

In the primary manufacturing phase, the APIs are produced in response to demands from the secondary phase. In some cases, the production of an API (also called an end product, from a primary manufacturer’s perspective) requires multiple steps, each of which occupies a bay entirely while the corresponding batch is being processed during the
Primary pharmaceutical manufacturing scheduling problem

four processing stages. The preparation for the next batch at a bay cannot be started until the previous batch has been processed by all four stages. We call the outputs, which will be consumed later to produce other products, as intermediate products. For example, according to the BOM relations depicted in Fig. 1, the sets of end products and intermediate products are, respectively, \( \{p, q, r\} \) and \( \{a, b, c, d, e\} \). Note that we assume there is unlimited supply of input materials for the production of intermediate products that do not have any incoming link in the BOM relations (such as \( \{a, b, d\} \) in the example). On the other hand, the inputs \( a, b, \) and \( d \) required for the intermediate products \( c \) and \( e \) are explicitly specified and, therefore, the related conservation of flow constraints are explicitly represented in the developed model. We further partition products into mutually exclusive families based on the connectedness induced by their BOM relations. This definition of product family is motivated by particular characteristics of changeovers between products, which will be explained later in this section.

For the convenience of production planning, we divide the time horizon into multiple periods. In order to avoid the complexity of modeling work-in-process at period boundaries, we assume that a production batch has to start and finish within the same period. As in any multi-level lot-sizing problem, external demands also induce internal demands for intermediate products. Intermediate and end products that are not immediately consumed or shipped in the same period incur inventory holding costs, which are calculated as the product of end-of-period inventory levels and per unit holding costs.

Due to the variety of products and limited number of processing bays, it is impractical to dedicate one bay per product. Hence, multiple products need to time-share the same bay and, consequently, incur changeovers. In order to avoid cross-contamination between certain products, the changeovers require a thorough cleaning of the bay. This is usually the case when the two products that are produced in sequence are not connected via BOM relations and, hence, belong to different families. More specifically, the setup time and cost that are required to perform the changeover depend on both of the product families involved (i.e., they are sequence-dependent). On the other hand, the changeover required between batches of the same family is negligible and, therefore, ignored. Within any given period, we define a production campaign as an uninterrupted sequence of batches that belong to the same family. Hence, changeovers only occur at the campaign level, independent of batch-level details. In the remainder of this article, changeover and setup are used interchangeably to denote sequence-dependent setups between campaigns.

In addition, the PPMSP has the following particular features.

1. For each product type, there is a subset of bays that are compatible for its production. The sets of compatible bays may differ even for two products of the same family.
2. Changeovers are allowed to straddle multiple periods. The end setup status of the previous period is carried over to the beginning of the next period (also known as setup carryover). We further allow setup splitting, where a changeover that connects two periods (called an inter-period setup) overlaps time from both periods. As stated earlier, we do not allow a batch to straddle two periods, which also avoids the complexity of dealing with unfinished batches in the computation of inventory levels. However, we do recognize the possibility of this occurring in practice and we therefore allow adjustments of schedules that permit this in a post-optimization step.
3. There are non-adjacency restrictions prohibiting the sequential processing of certain pairs of product families within any bay in accordance with Food and Drug Administration (FDA) rules.
4. The length of a campaign (in terms of its number of batches) may have an upper bound due to managerial considerations.

Due to the last two features above, it may become necessary to have multiple campaigns of the same family (interspersed with campaigns of other families) within a bay during the same period. For example, if the production process involves four families where three families are not allowed to be sequenced next to each other, then two campaigns of the fourth family must be scheduled between them in order to construct a valid sequence. To further clarify the characteristics of the PPMSP, we depict a sample schedule in Fig. 2 based on the BOM relations in Fig. 1. Note that in Bay 1, the ending setup status of Period 2 is retained through Period 3 in the absence of production, and it necessitates a changeover before Family 2 can be produced in Period 4. The changeover in Bay 1 during period 2 and the one in Bay 2 during Period 4 are intra-period setups, as they connect campaigns of different products produced in the same period; the other three changeovers are inter-period setups.

The objective of the PPMSP is to minimize the total cost, which consists of inventory holding cost and production setup cost, where the latter is directly proportional to the setup time. However, our problem formulation and proposed solution approach do not depend on this
particular assumption and can be likewise applied to more general cases. There are several levels of decisions to be made in each bay during each period, namely, the number of campaigns to schedule for each family with a compatible product, the number of batches to produce for each such compatible product, and the corresponding production amount (which may or may not be the maximum batch size). In addition, inventory levels are determined based on demands, production amounts, and BOM relations within each family, whereas the sequence of campaigns and changeovers are determined for each bay across all periods. We develop mathematical programming formulations and appropriate solution techniques to solve the PPSMP efficiently.

1.2. Literature review

The PPMSP is a multi-product, multi-level, big-bucket lot-sizing problem with multi-purpose machines that work in discrete batch production modes, and sequence-dependent setups. Fandel and Stammer-Hegene (2006) have addressed a similar problem. However, they did not consider discrete batches, nor did they provide any solution approach. Due to the up-front cost of changeovers, the production cost in the PPMSP is a concave function of the production amount. This is recognized as a general feature of lot-sizing problems (Florian and Klein, 1971, 1972). Although the single-item, uncapacitated, lot-sizing problem is well solved in the literature (Wagner and Whitin, 1958; Federgruen and Tzur, 1991; Wagelmans et al., 1992), the capacitated version is a much more difficult problem, except for some special cases (Florian and Klein, 1971, 1972; Fleischmann, 1990). In view of the inherent complexity (see Bitran and Yanasse (1982)), various solution approaches have been developed for this problem, including exact methods based on valid inequalities (Pochet, 1988; Leung et al., 1989; Miller et al., 2000; Loparic et al., 2003; Atamturk and Munoz, 2004), as well as heuristic methods (Hindi, 1995). The presence of multiple items and multiple machines further complicates this lot-sizing problem. Depending on the granularity of time periods and how many different products are allowed within the same period, multi-item lot-sizing models can be divided into small-bucket and big-bucket models, where the latter split the planning horizon into larger time periods and allow production to switch from one product to another within the same period. In order to solve big-bucket models, problem reformulations (Eppen and Martin, 1987; Armentano et al., 1999), Lagrangean relaxation (Graves, 1982; Diaby et al., 1992) and Dantzig–Wolfe decomposition (De graeve and Jans, 2007) have been investigated, among other methods. In multi-level lot-sizing problems, BOM relations exist among products. Several heuristic methods have been developed to derive near-optimal solutions for problems of this type (Kimms, 1997; Stadtler, 2003).

The concepts of a product family and joint setup have also been considered in the context of coordinated capacitated lot-sizing problems (Robinson and Lawrence, 2004) and joint replenishment problems (Federgruen and Tzur, 1994), where a family-level setup is a prerequisite for product-level setup and production. Although the PPMSP shares some aspects of this feature, setups in the PPMSP are sequence dependent and more intricate to model. More specifically, if we regard product families as cities and campaigns as visits to a city, the campaign sequencing substructure in the PPMSP resembles the High Multiplicity Asymmetric Traveling Salesman Problem (HMATSP) (Grigoriev and van der Klundert, 2006). Subtour Elimination Constraints (SECs) are crucial for an effective formulation of the ATSP, and several alternative SECs have been developed over the years (Dantzig et al., 1954; Miller et al., 1960; Wong, 1980; Sarin et al., 2005; Sherali et al., 2006). These SECs have been suitably modified for the HMATSP (Sarin et al., 2011), which requires the salesman to visit a city multiple times.

A lot-sizing problem with sequence-dependent setups is, in general, difficult to solve. Specialized model formulations have been developed based on a further subdivision of periods (Fleischmann, 1994; Meyr, 2002; Dogan and Grossmann, 2006). A heuristic method (Gupta and Magnusson, 2005) and column generation methods (Kang et al., 1999; Haase and Kimms, 2000) have also been proposed for such problems. In these studies, however, a standing assumption is that each product type can have at most one batch per period on the same machine. This is clearly not the case for the PPMSP, where multiple batches and even multiple campaigns pertaining to the same product can co-exist within a period.

Optimization of pharmaceutical supply chains has attracted considerable attention due to a recent restructuring of the pharmaceutical industry induced by global competition (Shah, 2004). Production planning and scheduling in primary manufacturing facilities is an important part of this overall optimization problem. Past studies on production planning and scheduling in process industries have investigated several related aspects, such as batch scheduling (Kondili et al., 1993), cyclic campaign planning/scheduling (Papageorgiou and Pantelides, 1996), integration of plant design and campaign scheduling (Fumero et al., 2000; Loparic et al., 1999), Lagrangean relaxation (Graves, 1982; Diaby et al., 1992) and Dantzig–Wolfe decomposition (De graeve and Jans, 2007) have been investigated, among other methods. In multi-level lot-sizing problems, BOM relations exist among products. Several heuristic methods have been developed to derive near-optimal solutions for problems of this type (Kimms, 1997; Stadtler, 2003).
et al., 2011), and consideration of uncertainties in demand and processing time (Verderame and Floudas, 2010). Our approach differs from these efforts in several ways: we address campaign changeovers on a continuous time scale via HMASTP; we omit the details of stage-level resource allocations within each bay, where we regard the bays as batching machines; we consider sequencing at the campaign level, and we do not determine a detailed schedule for each batch individually.

The remainder of this article is organized as follows. In Section 2, we present a Mixed-Integer Programming (MIP) model for the PPMSP with two alternative SEC formulations. Valid inequalities are derived in Section 3 to help tighten the linear programming (LP) relaxation of the proposed model and hence improve its solvability. A column generation–based approach is developed in Section 4. Results of a numerical investigation to test the effectiveness of the proposed solution procedure are presented in Section 5. Finally, a summary and some concluding remarks are provided in Section 6.

2. Problem formulation

2.1. Basic formulation

We begin by formulating our principal MIP model for the PPMSP. For the convenience of describing the campaign sequence in each bay, we use a dummy family 0 to denote the start and end of the sequence in each period. These period-wise campaign sequences are further connected by inter-period changeovers, as necessary, to form a full sequence for each bay. In each period and each bay, the campaign sequencing subproblem has a traveling salesman problem structure, where a city denotes a product family to be produced, a directed edge between two cities denotes a changeover from the tail family to the head family of the edge, and a tour connecting a multi-set of cities (i.e., with possible repetitions) that starts from and ends at the dummy city 0 represents a valid campaign sequence. Since multiple campaigns of the same product family can exist in a period, a tour can visit the same city multiple times and, hence, this constitutes the HMASTP. In addition to incorporating the flow balance constraint for the tour itself, the HMASTP substructure is formulated with multi-commodity flow-based SECs (Wong, 1980 and Sarin et al., 2011). The idea is to send, from the starting city 0, unit amounts of specific commodities to reach each of the cities, where a commodity flow is forbidden on any edge that is not part of the tour. The connectedness of the tour is then implied by the existence of flow paths from the dummy city 0 to all the other involved cities.

Consider the following notation (note that the commodity flows are denoted by different variables than those representing the tour itself):

Sets:

- \( F \): Set of real product families.
- \( \hat{F} \): Set of general product families; \( \hat{F} \equiv F \cup \{0\} \), where the dummy family 0 corresponds to the starting and ending city in an HMASTP tour.
- \( A_j \): Set of families that are forbidden from immediately following family \( f \), \( \forall f \in F \).
- \( N_f \): Set of products in family \( f \), \( \forall f \in F \). Note that for \( f, g \in F \) s.t. \( f \neq g \), we have \( N_f \cap N_g = \emptyset \).
- \( N \): Set of all products; \( N \equiv \bigcup_{f \in F} N_f \).
- \( M \): Set of bays.
- \( N_k^f \): Set of products that can be produced in bay \( k \), \( \forall k \in M \).
- \( M^f \): Set of bays that can produce product \( i \), \( \forall i \in N; M^f \equiv \{ k \in M : i \in N_k^f \} \).
- \( F_k \): Set of real product families of which at least one product can be produced in bay \( k \), \( \forall k \in M; F_k \equiv \{ f \in F : N_f \cap N_k^f \neq \emptyset \} \).
- \( \hat{F}_k \): Set of general product families whose product(s) can be produced in bay \( k \), \( \forall k \in M; \hat{F}_k \equiv \hat{F} \cup \{0\} \).
- \( T \): Set of periods in the current planning horizon; \( T = \{1, \ldots, |T|\} \).
- \( \hat{T} \): Set of time periods including period 0, which precedes the first period of \( T \); \( \hat{T} = \{0\} \cup T \).

Note that the set \( F_k \) is induced because of the compatibility of certain products with bay \( k \), and this compatibility does not necessarily apply to other products of the same family (i.e., \( i \in N_f \) and \( f \in F_k \Rightarrow i \in N_k^f \)).

Parameters:

- \( \sigma \): Setup cost per unit setup time.
- \( f_0^k \): Initial setup status of bay \( k \) (given product family that is ready for processing at the end of period 0), \( \forall k \in M \).
- \( I_0^i \): Initial inventory of product \( i \), \( \forall i \in N \).
- \( h_i^t \): Inventory holding cost per period for a unit amount of product \( i \), \( \forall i \in N \).
- \( d_i^t \): External demand for product \( i \) at the end of period \( t \), \( \forall i \in T, i \in N \).
- \( q_k^i \): Maximum batch size for product \( i \) when produced in bay \( k \), \( \forall i \in N, k \in M^f \).
- \( p_k^i \): Batch processing time for product \( i \) when produced in bay \( k \), \( \forall i \in N, k \in M^f \).
- \( s_{f,g}^k \): Setup time required for a changeover from a campaign of family \( f \) to a campaign of family \( g \) in bay \( k \), \( \forall k \in M, f, g \in F_k \). Note that \( s_{f,g}^k = 0 \), \( \forall k \in M, f \in F_k \).
- \( c_k^t \): Time capacity of bay \( k \) during period \( t \), \( \forall k \in M, t \in T \).
- \( p_k^t \): Maximum number of batches per campaign for family \( f \) in bay \( k \) during period \( t \), \( \forall k \in M, t \in T, f \in F_k \); this is dictated by managerial considerations.
- \( m_k^f \): Maximum number of batches of family \( f \) that can be scheduled in bay \( k \) during period \( t \) without exceeding the time capacity, \( \forall k \in M, t \in T, f \in F_k \); \( m_k^f = \lceil c_k^t / \min_{i \in N_f \cap N_k^f}[p_k^i] \rceil \).
\[ b_{i,j} \]: Amount of product \( i \) required to produce one unit of product \( j \) in any bay (i.e., BOM ratio), \( \forall f \in F, i, j \in N_f, i \neq j \).

**Decision variables:**

\[ I_t^i := \text{Inventory level of product } i \text{ at the end of period } t, \forall t \in T, i \in N. \]

\[ G_t^i := \text{Amount of product } i \text{ produced during period } t, \forall t \in T, i \in N. \]

\[ P_{t}^{k,i} := \text{Amount of product } i \text{ produced in bay } k \text{ during period } t, \forall t \in T, i \in N, k \in M. \]

\[ W_{t}^{k,f} := \text{Number of batches of product } i \text{ produced in bay } k \text{ during period } t, \forall t \in T, i \in N, k \in M. \]

\[ R_{t}^{f} := \text{Number of campaigns of family } f \text{ scheduled in bay } k \text{ during period } t, \forall k \in M, t \in T, f \in F^k. \]

\[ Z_{t}^{f} := \text{Binary variable indicating whether any product of family } f \text{ is produced in bay } k \text{ during period } t, \forall k \in M, t \in T, f \in F^k. \]

\[ X_{t}^{k,f,g} := \text{Number of changeovers in bay } k \text{ from a campaign of family } f \text{ to a campaign of family } g \text{ in the same period } t, \forall k \in M, t \in T, f, g \in F^k; f \neq g. \] This is also the number of times that edge \((f, g)\) is traversed on the tour of the HMA\(TSP\) for period \(t\) and bay \(k\). Note that self-loops \((f = g)\) are not allowed for such intra-period setups in order to enforce campaign length restrictions.

\[ Y_{t}^{k,0,g} := \text{Indicator of an inter-period setup that switches from a campaign of family } f \text{ at the end of period } t_1 \text{ to a campaign of family } g \text{ at the beginning of period } t_2, \forall k \in M, t \in T, g \in F^k, t_1, t_2 \in T; t_1 < t_2 \text{ (possibly, } f = g). \]

\[ \beta_{t}^{k,f} := \text{Time allocated at the beginning of period } t \text{ in bay } k \text{ for an inter-period setup, } \forall k \in M, t \in T. \]

\[ \alpha_{t}^{k,f} := \text{Time allocated at the end of period } t \text{ in bay } k \text{ for an inter-period setup, } \forall k \in M, t \in T. \] Note that no changeover is needed at the end of the planning horizon. Hence, \( \alpha_{t}^{k,T} \equiv 0, \forall k \in M. \)

\[ \lambda_{t}^{k,u,f} := \text{Flow of commodity } u \text{ on edge } (f, g) \text{ that is sent to reach } u \text{ in bay } k \text{ during period } t, \forall k \in M, t \in T, u, g \in F^k, f \in F^k; f \neq g. \] For any family \( u \) that is produced in bay \( k \) during period \( t, \forall u \in F^k \), a unit amount of commodity \( u \) is sent from the dummy family 0 to reach family \( u \), following the path indicated by the \( \lambda \)-variables.

**Model PPMSP:**

Minimize

\[ \sum_{t \in T} \sum_{i \in N} h_i I_t^i + \sigma \sum_{k \in M} \sum_{f \in P^k} s_{f,g}^{k} \]

\[ \times \left( \sum_{i \in T} X_{t}^{k,f,g} + \sum_{n_1, t \in T} Y_{t}^{k,n,t} \right), \]

subject to:

\[ I_{t}^{i-1} + G_{t}^{i} = I_{t}^{i} + d_{i}^{t} + \sum_{j \in N_f} b_{i,j} G_{t}^{j}, \quad \forall t \in T, f \in F, i \in N_f. \]

\[ G_{t}^{i} = \sum_{k \in M} P_{t}^{k,i}, \quad \forall t \in T, i \in N. \]

\[ P_{t}^{k,i} \leq d_{k}^{t} W_{t}^{k,f}, \quad \forall t \in T, f \in F, i \in N_f, k \in M. \]

\[ \sum_{i \in N_f} P_{t}^{k,i} + \sum_{j \in N_f} s_{f,g}^{k} X_{t}^{k,f,g} + \alpha_{t}^{k,f} + \beta_{t}^{k,f} \leq \hat{c}_{t}^{k,f}, \quad \forall t \in T, k \in M. \]

\[ \sum_{k \in M} \sum_{j \in N_f} Y_{t}^{k,0,j} \leq 1, \quad \forall k \in M. \]

\[ Y_{t}^{k,0,g} = 0, \quad \forall k \in M, t \in T, f, g \in F^k; f \neq g. \]

\[ X_{0,t}^{f} = \sum_{f \in F^k} \sum_{n_1 \in T} Y_{t}^{k,n,t}, \quad \forall k \in M, t \in T, g \in F^k. \]

\[ X_{t}^{k,f,g} \geq \sum_{g \in F^k} \sum_{n_1 \in T} Y_{t}^{k,n,t}, \quad \forall k \in M, t \in T, f \in F^k. \]

\[ \sum_{f \in F^k} X_{t}^{k,f,g} = R_{t}^{k}, \quad \forall k \in M, t \in T, f \in F^k. \]

\[ \sum_{f \in F^k} X_{t}^{k,f,g} = R_{t}^{k}, \quad \forall k \in M, t \in T, f \in F^k. \]

\[ s_{f,g}^{k} Y_{t}^{k,0,1} = \beta_{t}^{k,f}, \quad \forall k \in M. \]

\[ \sum_{f \in F^k} s_{f,g}^{k} X_{t}^{k,f,g} = \alpha_{t}^{k,f} + \beta_{t}^{k,f}, \quad \forall k \in M, t \in T; t > 1. \]

\[ \sum_{i \in N_f \cap N_k} W_{t}^{i} \leq R_{t}^{k}, \quad \forall k \in M, t \in T, f \in F^k. \]

\[ \sum_{i \in N_f \cap N_k} W_{t}^{i} \geq R_{t}^{k}, \quad \forall k \in M, t \in T, f \in F^k. \]

\[ \sum_{i \in N_f \cap N_k} W_{t}^{i} \leq m_{f}^{k} Z_{t}^{f}, \quad \forall k \in M, t \in T, f \in F^k. \]

\[ R_{t}^{k} \geq Z_{t}^{f}, \quad \forall k \in M, t \in T, f \in F^k. \]

\[ X_{t}^{f,g} = 0, \quad \forall k \in M, t \in T, f \in F^k, g \in F^k \cap A_f, \]

\[ Y_{t}^{k,0,g} = 0, \quad \forall k \in M, t \in T, u, g \in F^k, \]

\[ \sum_{t \in T} X_{t}^{k,f,g} = Z_{t}^{u}, \quad \forall k \in M, t \in T, u \in F^k. \]

\[ \sum_{t \in T} X_{t}^{k,f,g} = Z_{t}^{u}, \quad \forall k \in M, t \in T, u \in F^k. \]
The objective function is composed of inventory holding costs plus setup costs pertaining to both intra-period and inter-period setups. The setup cost incurred depends on the time, chemicals, and special processes required for cleaning the containers of a bay. For simplicity in presentation, we have assumed that the setup cost is directly proportional to the setup time; however, our approach does not depend on this assumption and alternative computations of setup costs could be likewise accommodated. The cost of production is assumed to be time-invariant and bay-invariant and hence is omitted from the objective function. Constraint (1) enforces the inventory balance for each product, and is called the Inventory Balance Constraint (IBC). Note that the IBC does not require any positive inventory at the end of the planning horizon. If end inventory is required, it can be accommodated by specifying additional demand for the last period. Also, Constraint (1) enforces BOM requirements and conservation of flow among products. Constraint (2) (the Production Allocation Constraint) allocates the production of product \( i \) to compatible bays, whereas Constraint (3) (the Production Batch Constraint) defines batches for each product in each bay and ensures that there is a sufficient number of batches to produce the allocated amount. By substituting for the \( P^f_{i,j} \)-variables, these two constraints can be combined as follows:

\[
G_i^f \leq \sum_{k \in M} q_k^f W^k_i, \quad \forall t \in T, \ i \in N.
\]  

(23)

In this case, the exact amount of production in each bay will be determined from the value of \( G_i^f \) after solution is obtained (e.g., we can assume \( P^f_i \) to be proportional to \( q_k^f W^k_i \)). Other than when we need to retain the \( P^f_i \)-variables for the convenience of deriving valid inequalities (as discussed later in Section 3), we always apply the above substitution to reduce the size of the formulation. Constraint (4) is the Time Capacity Constraint (TCC). Constraints (5) to (10) pertain to the setup sequences of campaigns, and are referred to as the Sequencing Constraints. Constraints (5) and (6) dictate that in period 0 the first inter-period setup in bay \( k \), if it exists at all, always starts from family \( f_k^G \). For each period, the production in a bay is always initiated by an inter-period setup, which starts from an earlier period and ends in the current period. The product family that receives this incoming inter-period setup is also associated with an intra-period changeover from the dummy family 0, as dictated by Constraint (7). The last product family produced during a period is associated with an intra-period changeover to the dummy family 0. This either means the end of production in this bay, or the start of an inter-period setup pointing to a later period, as dictated by Constraint (8). For each product family, Constraints (9) and (10) ensure that the number of campaigns for a product family is equal to the number of changeovers that lead to and depart from this family, respectively. Constraints (11) and (12) allocate time for inter-period setups. They are referred to as the Setup Splitting Constraints. Note that for an inter-period setup that straddles more than two periods to connect two campaigns, there must be at least one idle period between campaigns. In that case, we assume that any changeover can be accommodated entirely during an idle period and, hence, omit explicit time allocation for this situation. As a result, Constraints (11) and (12) only pertain to inter-period setups between consecutive periods. Constraints (13) to (16) relate the numbers of campaigns to the numbers of production batches. They are referred to as the Campaign Production Constraints. Constraints (13) and (14), respectively, dictate the existence of at most \( f^k_i \) batches and at least one batch within each campaign, in the aggregate. The actual assignments of batches to the campaigns are not specified by the formulation, but they can be readily ascertained after a solution is obtained. Constraint (15) ensures that the production indicator is turned on if a batch of the corresponding family is produced, and Constraint (16) ensures that it is turned off if no campaign is scheduled. Note that whenever \( m^k_i \leq f^k_i \), Constraint (13) can be omitted, as it is implied by Constraints (15) and (16). Constraints (17) and (18) constitute the Setup Prohibition Constraints, which disallow changeovers between certain families according to FDA rules. Constraints (19) to (22) pertain to the sequence of campaigns within a period. They are referred to as the Graph Connectivity Constraints (GCCs). Since multiple production campaigns of a family may exist in the same bay during a period, the sequencing of campaigns in bay \( k \) during period \( t \) is an HMATSP. Note that for any \( k \in M, t \in T \) the set of active product families, \( V^k_i \equiv \{0\} \cup \{g \in F^k : Z^g_{k,t} = 1\} \), is regarded as the set of cities in the traveling salesman problem, and the tour starts from and ends at the dummy family 0. Constraints (19) to (22) are adapted from the polynomial-length HMATSP formulations in Wong (1980), Sherali et al. (2006), and Sarin et al. (2011). Constraints (20) and (21), respectively, initiate a unit flow of commodity \( u \) from the dummy family and end it at family \( u \), if and only if family \( u \) is in production (i.e., active). Due to Constraint (19), the commodity flow is permitted on an edge only if the
corresponding intra-period setup exists. Constraint (22) enforces the flow balance of commodity $u$ at each city (family).

Next, we present an alternative formulation for the GCCs.

### 2.2. An alternative formulation for the GCCs

In lieu of using the multi-commodity flow Constraints (19) to (22) to ensure the connectivity of the setup graph, we can apply the rank-order concept of Miller–Tucker–Zemlin (MTZ) SECs (Miller et al., 1960). Unlike the classic ATSP, the HMATSP allows sub-tours within a feasible sequence. Therefore, the standard MTZ formulation of SECs cannot be applied directly. To establish a consistent rank-order for the HMATSP, we rely on the edges of first-visits, which correspond to intra-period setups, the existence of such a flow to family $u$ indicates its connectedness with family 0 through intra-period setups. The connectivity of the setup graph is induced by the fact that all the families in $V_k$ are connected with family 0. The remaining Constraints (23) enforce logical restrictions on the decision variables. Note that Constraints (9) and (10) imply that

$$\sum_{g \in F_k} X_{g,f}^{k,t} - \sum_{g \in F_k} X_{g,f}^{0,t} = 0, \quad \forall k \in M, t \in T.$$  

Furthermore, substituting the $X$-variables in the above equation from Constraints (7) and (8), we have

$$\sum_{f \in F_k} \sum_{t_1 < t_2} X_{f,g}^{k,t_1,t_2} \geq \sum_{f \in F_k} \sum_{t_1 < t_2} X_{f,g}^{k,t_1,t_2}, \quad \forall k \in M, t \in T.$$  

This dictates that the number of incoming inter-period setups to bay $k$ in period $t$ is no less than the number of outgoing inter-period setups. To illustrate this idea, we depict in Fig. 3 both the inter-period and the intra-period setups of the schedule given in Fig. 2.

Next, we present an alternative formulation for the GCCs.

**Proposition 1.** Along with the sequencing Constraints (5) to (10), the following set of valid inequalities induces a feasible setup sequence:

$$\sum_{f \in F_k \setminus \{g\}} Y_{f,g}^{k,t} = Z_{g}^{k,t}, \quad \forall k \in M, t \in T, g \in F_k, \quad (25)$$

$$Y_{f,g}^{k,t} \leq X_{f,g}^{k,t}, \quad \forall k \in M, t \in T, f \in F_k, g \in F_k, f \neq g, \quad (26)$$

$$U_{f,g}^{k,t} - U_{g}^{k,t} + |F_k| \cdot Y_{f,g}^{k,t} \leq |F_k| - 1, \quad \forall k \in M, t \in T, f \neq g \in F_k. \quad (27)$$

**Proof.** Define the following:

- Set of active families: $V_{k,t} \equiv \{0\} \cup \{g \in F_k : Z_{g}^{k,t} = 1\}$.  
- Set of active edges: $E_{k,t} \equiv \{(f \rightarrow g) : f, g \in F_k, f \neq g, X_{f,g}^{k,t} > 0\}$.  
- Edge set of first-visits: $E_{k,t}^{1} \equiv \{(f \rightarrow g) : f \in F_k, g \in F_k, f \neq g, Y_{f,g}^{k,t} = 1\}$, $\forall k \in M, t \in T$.  

For any feasible solution, the edge set $E_{k,t}^{1}$ induces a directed graph over $V_{k,t}$ by Equations (5) to (10). A corresponding campaign sequence can be determined by using the procedure ConvertToSequence of Grigoriev and van de Klundert (2006). Hence, we can determine the values of the $Y_{f,g}^{k,t}$-variables according to the sequence obtained. If family $g$ is in production, its first campaign in the sequence must be preceded by the campaign of another real product family or by the dummy family 0. Hence, Constraint (25) is satisfied. Constraint (26) is also valid as $Y_{f,g}^{k,t} = 1$ indicates a changeover setup from family $f$ to family $g$, which is accounted for in the setup variable $X_{f,g}^{k,t}$. Note that the edge set of first visits, $E_{k,t}^{1}$, induces a directed tree graph over the set of active families as any active family can trace back to the dummy family by following the edges in $E_{k,t}^{1}$ in the reverse direction. Consequently, a full sequence of active families can be reconstructed by a preorder traversal (Bazaraa et al., 2010) of this tree graph. Constraint (27) is satisfied by using the rank-order of this sequence for $U$-values.

Conversely, if a solution satisfies the constraints of Proposition 1, the edge set $E_{k,t}^{1}$ induces a connected graph over $V_{k,t}$. To see this, suppose that, on the contrary, the graph induced by $E_{k,t}^{1}$ contains a proper subgraph $\Sigma$ that is disconnected from the rest of the families in $V_{k,t}$. Let $V(\Sigma)$ be the set of families (vertices) in the subgraph $\Sigma$. For each family $g \in V(\Sigma)$, there exists a family $f$ such that $Y_{f,g}^{k,t} = 1$, due to Constraint (25). Hence, there exist $|V(\Sigma)|$
edges that belong to the set \( \hat{E}_{k,l} \) and that are incident to families in \( V(\Sigma) \). Therefore, \( \Sigma \) contains a cycle. This violates Constraint (27), since a sequential rank-order cannot be established around a cycle.

The above MTZ-type SECs yield a more compact model representation, albeit weaker than that obtained using the multi-commodity flow-based SECs (Wong, 1980), as observed for the ATSP in Oncan et al. (2009). In Section 5, we compare the computational performance of these SECs within the context of the PPMSP.

3. Valid inequalities

In this section, we derive various classes of valid inequalities to further tighten the model representation.

3.1. Inequalities based on the TCC

**Proposition 2.** If there exists a product \( i^* \in N_f \cap N_k \) such that \( p_i^k > c_{k,t} / m_{i,t}^k \), then Inequality (15) can be strengthened by also incorporating the following valid inequality:

\[
\sum_{t \in T} p_i^k W_{i,t}^k \leq c_{k,t} Z_{f,t}^k, \quad \forall k \in M, t \in T, f \in F. \tag{28}
\]

**Proof.** See online Appendix I. ■

Note that neither of the two Inequalities (15) and (28) dominates the other. Hence, they can be simultaneously added to the model. Furthermore, we can strengthen Inequality (28) itself by deriving a smaller right-hand side coefficient from the following subset-sum problem, \( \forall k \in M, t \in T, f \in F^k \):

\[
\hat{c}_{f,t} = \max \left\{ \sum_{i \in N_f \cap N_k} p_i^k W_{i,t}^k \mid W_{i,t}^k \in \mathbb{Z}^+, \right\}.
\]

Consequently, Inequality (28) can be modified as

\[
\sum_{t \in T} p_i^k W_{i,t}^k \leq \hat{c}_{f,t} Z_{f,t}^k, \quad \forall k \in M, t \in T, f \in F^k. \tag{29}
\]

By considering the production of each product separately, we can further add the following Chvatal-type of valid inequality (Nemhauser and Wolsey, 1999), where \( \lfloor \cdot \rfloor \) denotes the round-down operator:

\[
W_{i,t}^k \leq \left\lfloor \frac{c_{k,t}}{p_i^k} \right\rfloor Z_{f,t}^k, \quad \forall t \in T, f \in F, i \in N_f, k \in M^f. \tag{30}
\]

3.2. Inequalities based on upper bounds on production amounts

In the basic model, an increment in the production level \( p_{i,t}^k \) tends to increase the number of batches \( W_{i,t}^k \), which, in turn, requires the production indicator \( Z_{f,t}^k \) to be positive. We consider the following valid inequality, which directly connects the production level with the production indicator:

\[
p_{i,t}^k \leq \theta_{i,t}^k Z_{f,t}^k, \quad \forall f \in F, i \in N_f, t \in T. \tag{31}
\]

where the parameter \( \theta_{i,t}^k \) is an upper bound on the amount of product \( i \) that can be produced in bay \( k \) during period \( t \). By Constraints (30) and (31), we can let \( \theta_{i,t}^k = q_i^k \left\lfloor c_{k,t} / p_i^k \right\rfloor \). However, the value of \( \theta_{i,t}^k \) can be further reduced as described next.

We define \( \hat{G}_i^t \) as the minimum amount of product \( i \) that needs to be produced in order to satisfy external and internal demands in period \( t \), assuming that there is no inventory of any product at the beginning of period \( t, \forall t \in T, i \in N \). On the other hand, when an initial inventory of materials is present, producing a product \( i \) can potentially reduce inventory holding costs of input and output materials and their corresponding BOMs. We define \( \tilde{G}_i^t \) as an upper bound on such a production that contributes toward reducing total inventory cost, irrespective of whether there is demand for product \( i \). Online Appendix I provides details on computing \( \hat{G}_i^t \) and \( \tilde{G}_i^t \). This leads to the following result:

**Proposition 3.** An upper bound on the amount of product \( i \) that can be produced in an optimal solution in bay \( k \) during period \( t \) is given by

\[
\theta_{i,t}^k = \min \left\{ q_i^k \left\lfloor \frac{c_{k,t}}{p_i^k} \right\rfloor, \left( \sum_{t \in T} \hat{G}_i^t + \tilde{G}_i^t \right) \right\}.
\]

**Proof.** See online Appendix I. ■

We introduce another inequality similar to Equation (31) to make a direct connection between the production-level \( G_{f,t}^i \) and the production indicator \( Z_{f,t}^i \) as follows:

\[
G_{f,t}^i \leq \sum_{k \in M^f} \theta_{i,t}^k Z_{f,t}^k, \quad \forall f \in F, i \in N_f. \tag{32}
\]

Note that Inequality (32) can be deduced from Inequalities (2) and (31), and is therefore redundant when Equation (31) is present. However, applying Inequality (32) instead of Inequality (31) can be beneficial as it allows the omission of the P-variables from the model formulation.

3.3. Inequalities based on the lot-sizing subproblem

**Proposition 4.** If the binary variable \( z_{i,t}^l \) indicates whether product \( i \) is produced in period \( t \), and the variable \( \eta_{i,j} \)
denotes the amount of product \( i \) needed (directly or indirectly, according to BOM relations) to produce a unit of product \( j \), \( \forall j \in F \) and \( i, j \in N_f \), then

\[
\sum_{j \in N_f} \eta_{i,j} I_j^{-1} \geq \sum_{j \in N_f} \eta_{i,j} d_j^t, \quad \forall t \in T, \ f \in F, \ i \in N_f. \tag{33}
\]

**Proof.** This follows from the fact that if there is no production in the current period, then the demand has to be satisfied by the inventory carried over from the previous period.

Note that the value of \( \eta_{i,j} \) is positive if and only if product \( i \) is an upstream product with respect to product \( j \). For brevity, the algorithm for calculating the \( \eta_{i,j} \)-values is relegated to online Appendix II.

Note that the following constraints establish the definitional roles of \( \zeta^f_i \):

\[
W_{i}^{k,t} \leq \left[ \frac{c_{k,t}}{p_{k}^{t}} \right] \zeta_t^i, \quad \forall t \in T, \ i \in N, \ k \in M, \tag{34}
\]

\[
\sum_{k \in M} W_{i}^{k,t} \geq \zeta_t^i, \quad \forall t \in T, \ i \in N. \tag{35}
\]

The smaller the value of \( \zeta^f_i \), the stronger Inequality (33) is. Hence, a lower bounding Constraint (34) on \( \zeta^f_i \) will not help to strengthen Inequality (33). On the other hand, Inequality (30) also implies an upper bound value of \( \left[ \frac{c_{k,t}}{p_{k}^{t}} \right] \) for \( W_{i}^{k,t} \). Hence, we can omit Inequality (34) if Inequality (30) is present. Moreover, we can add the following valid inequality, which dictates that if a product \( i \) is produced in period \( t \), then \( \hat{d}_i^t \) is produced in at least one of the compatible bays of \( M \):

\[
\sum_{k \in M} Z_{f}^{k,t} \geq \zeta_t^i, \quad \forall t \in T, \ f \in F, \ i \in N_f. \tag{36}
\]

Atamtürk and Muñoz (2004) have proposed bottleneck cover inequalities that are based on the same idea as that of Proposition 4. Their results have been extended by Glover and Sherali (2005) to consider the production status of multiple periods in general. For the PPMSP, we can consider the production indicators of the next two periods and derive the following valid inequality:

\[
\sum_{j \in N_f} \eta_{i,j} I_j^{-1} \geq (1 - \zeta^f_i) \hat{d}_i^t + (1 - \zeta^{f+1}_i) \times \min \left\{ \hat{d}_i^{t+1}, \max \left\{ \hat{d}_i^t + \hat{d}_i^{t+1} - \omega_i^t, 0 \right\} \right\},
\]

\[
\forall t \in T, \ i \in N; \ t < |T|, \tag{37}
\]

where \( \hat{d}_i^t \) is the total converted demand that is defined as \( \hat{d}_i^t \equiv \sum_{j \in N_f} \eta_{i,j} d_j^t \), and where \( \omega_i^t \) is an upper bound on the amount of product \( i \) produced in period \( t \). Specifically, we take \( \omega_i^t \equiv \sum_{k \in M} \theta_i^{k,t} \), where, as defined for Inequality (31), \( \theta_i^{k,t} \) is an upper bound on the amount of product \( i \) that is produced in bay \( k \) during period \( t \).

Note that Inequality (33) relates the aggregated production indicator \( \zeta^f_i \) with the inventory level. We can avoid the introduction of the \( \zeta \)-variables by considering the following disaggregated constraint, which is established by Proposition 5 below:

\[
\sum_{j \in N_f} \eta_{i,j} I_j^{-1} \geq \sum_{k \in M} \left( 1 - Z_{f}^{k,t} \right) \epsilon_{i}^{k,t}, \quad \forall t \in T, \ f \in F, \ i \in N_f, \ s.t. \sum_{k \in M} \epsilon_{i}^{k,t} \leq \hat{d}_i^t, \tag{38}
\]

where \( \epsilon_{i}^{k,t} \) is defined as the shortage in capacity if bay \( k \) is not used to produce product \( i \) during period \( t \); i.e., \( \epsilon_{i}^{k,t} = \max \{ 0, \hat{d}_i^t - (\sum_{k \in M \setminus \{k\}} \theta_i^{k,t}) \} \).

**Proposition 5.** Inequality (38) is valid if \( \sum_{k \in M} \epsilon_{i}^{k,t} \leq \hat{d}_i^t, \quad \forall t \in T, \ f \in F, \ i \in N_f. \)

**Proof.** See online Appendix I.

### 4. Column generation approach

We now design a column generation (Desaulniers et al., 2005) approach for the PPMSP in order to exploit its inherent special substructures. Various problem decomposition schemes can be used to separate out these substructures into pricing subproblems for their effective solution. Columns represented by solutions to the subproblems can then be inserted into the master program in order to improve the overall objective. The PPMSP can be decomposed in many different ways: by bays, periods, and families. We attempted several such schemes (see Liao (2011) for details) and, based on the results obtained, we shall focus here on the best performing scheme, denoted HMATSP2, wherein the subproblem determines the campaign sequence in each bay and each period, without considering the inter-period setup variables \( \gamma_{f}^{k,i,t} \). This induces a decomposition scheme by bay and by period, with each subproblem resembling an HMATSP.

Our overall approach is as follows: we relax the master program by omitting its integrality restrictions and solve it as an LP using column generation. Once the column generation process is complete for the LP relaxation, we fix the set of columns and re-impose the integrality restrictions in the master program. With this fixed set of columns, the master program is further solved using a branch-and-bound method to obtain a feasible solution to the PPMSP. Note that this approach is a heuristic since we do not apply branch-and-price to deal with the integrality restrictions.

To present the details of this approach, consider the following additional notation:

- **Sets:**
  - \( \Lambda^{k,t} \): Index set of the columns corresponding to bay \( k \) and period \( t \), \( \forall k \in M, \ t \in T \).
Primary pharmaceutical manufacturing scheduling problem

Parameters:

\( \mu \): Value of the variable contained in \( \mu \), which is obtained from the pricing subproblem and that defines an appropriate coefficient within the \( i \)th column of bay \( k \) and period \( t \), \( k \in M, t \in T, t \in \Lambda^{k,t} \) (note that the variable in \( \mu \) itself carries the indices \( k \) and \( t \), thus indicating the corresponding subproblem that produces its value).

Decision variables:

\( Q_{k,t}^i \): Binary variable indicating the inclusion/exclusion of the \( i \)th column of bay \( k \) and period \( t \), \( k \in M, t \in T, t \in \Lambda^{k,t} \) relaxed to take continuous values.

\( \nu_{1,t} \): Dual variable associated with the convexity constraint (39), \( k \in M, t \in T \).

\( \mu_{k,t} \): Dual variable associated with the TCC (40), \( \mu_{k,t} \leq 0, k \in M, t \in T \).

\( \psi_{k,t}^{n} \): Dual variable associated with Constraint (41), \( k \in M, t_1 \in T, f \in F^k \).

\( \xi_{f,t}^i \): Dual variable associated with Constraint (42), \( \xi_{f,t}^i \geq 0, k \in M, t_1 \in T, f \in F^k \).

\( \nu_{f,t}^i \): Dual variable associated with Constraint (43), \( \nu_{f,t}^i \leq 0, k \in M, t \in T, f \in F^k \).

\( \phi_{f,t}^i \): Dual variable associated with Constraint (44), \( \phi_{f,t}^i \leq 0, k \in M, t \in T, f \in F^k \).

\( \psi_{f,t}^i \): Dual variable associated with Constraint (45), \( \psi_{f,t}^i \leq 0, k \in M, t \in T, f \in F^k \).

\( \sigma_{f,t}^i \): Dual variable associated with Constraint (46), \( \sigma_{f,t}^i \leq 0, k \in M, t \in T, f \in F^k \).

\( \xi_{f,t}^i \): Dual variable associated with Constraint (47), \( \xi_{f,t}^i \leq 0, \forall t \in T, f \in F, i \in N_f \).

\( \sigma_{f,t}^i \): Dual variable associated with Constraint (48), \( \sigma_{f,t}^i \geq 0, \forall t \in T, f \in F, i \in N_f \).

The HMATSP2 decomposition scheme is implemented as follows:

**HMATSP2-Master Program:**

\[
\begin{align*}
\text{Minimize} & \quad \sum_{i \in T} \sum_{t \in N} h_i^t + \sum_{k \in M} \sum_{f, g \in \mathcal{F}^k} \sum_{n_i \in T} s_{f,g}^k \gamma_{f,g}^{n_k,n_i,t} \\
& \quad + \sum_{k \in M} \sum_{t \in T} \sum_{i \in \mathcal{A}^{k,t}} \sum_{f, g \in \mathcal{F}^k, f \neq g} Q_{k,t}^i \sum_{f, g \in \mathcal{F}^k} (X_{f,g}^{k,t})_i,
\end{align*}
\]

subject to Constraints (1)–(3), (5), (6), (11), (12), (33), (35) and

\[
\begin{align*}
\sum_{i \in \mathcal{A}^{k,t}} Q_{k,t}^i &= 1, \quad \forall t \in T, k \in M, \tag{39} \\
\sum_{i \in \mathcal{A}^{k,t}} p_{k} W_{k}^i + \sum_{i \in \mathcal{A}^{k,t}} Q_{k,t}^i \sum_{f, g \in \mathcal{F}^k, f \neq g} s_{f,g}^k (X_{f,g}^{k,t})_i + \alpha_{k,t}^i.
\end{align*}
\]

subject to Constraints (9), (10), (16), (19)–(22) and

\[
\begin{align*}
\sum_{f, g \in \mathcal{F}^k} X_{0,g}^{k,t} &\leq 1, \tag{49} \\
\sum_{f \in \mathcal{F}^k} X_{f,0}^{k,t} &\leq 1, \tag{50} \\
R_{j}^{k,t} &\leq m_{j}^{k,t} Z_{j}^{k,t}, \quad \forall f \in \mathcal{F}^k. \tag{51}
\end{align*}
\]
that is induced by Constraints (14) and (15), and dictates versions of Constraints (7) and (8), respectively. Constraint Constraints (49) and (50) in the subproblem are aggregated fine the coefficients of the column to be generated. Note that restrictions governing the variables that will ultimately de-

the objective value of the subproblem (i.e., the least reduced cost) is computed using the values of dual variables of the master program. The objec-
tive function of the pricing subproblem represents the re-
duced cost of any potential column to be generated, which is is computed during the column generation phase. In this context, the Complementary Column Generation (CCG) technique has been shown to yield feasible solutions for certain types of problems (Ghoniem and Sherali, 2009). However, CCG was unable to generate an adequate set of complementary columns due to the inherent complexity of PPSMP. On the other hand, our computational results in the next section show that the R-B heuristic is quite effective in obtaining an initial feasible solution with an associated set of columns as well as an improved final solution.

5. Computational results

We conducted a computational study to evaluate the efficacy of the proposed modeling and algorithmic constructs. The test instances used for this purpose are based on real-life data obtained from Boehringer Ingelheim, a primary pharmaceutical manufacturing plant located near Richmond, Virginia. The distributional characteristics of various problem parameters were obtained from historical data and were then applied to generate random problem instances. For the sake of brevity, we refer the reader to Liao (2011) for more details on the generation of problem instances.

Table 1 summarizes the dimensions of two different problem sets that were generated. Problem set A contains 10 moderately sized instances, whereas problem set B contains 10 larger instances that reflect the problem size encountered in a real-life facility, both from the viewpoint of facility size and number of jobs involved.

All computational runs were performed on a computer equipped with a 2.67 GHz dual core CPU, 2 GB of RAM, and OPL IDE 6.3 (which uses CPLEX 12.1 as the under-

cutting solver). We always apply the default CPLEX cuts to the model that incorporates (subsets of) our valid

| Parameters | | | | |
|---|---|---|---|---|
| Dataset A | 5 | 15 | 5 | 12 |
| Dataset B | 7 | 52 | 21 | 12 |
inequalities when solving an MIP. Furthermore, we report on the LP-based lower bound values at the root node both before and after the CPLEX default cuts are added (respectively denoted as Bound I and Bound II), in order to assess the effectiveness of these general-purpose cuts. We also evaluate the effectiveness of the proposed valid inequalities. For each of these bounds, we define the corresponding optimality gap as

\[ \text{gap} = \frac{100 \times \text{optimal objective value} - \text{lower bound value}}{\text{optimal objective value}} \% \]

### 5.1. Comparison of formulations

We compare the effectiveness of two different formulations without the valid inequalities introduced in Section 3. Formulation F1 relies on the concept of multiple commodity flows to enforce graph connectivity and includes Constraints (1) and (4) to (24). Formulation F2 extends the rank-order concept of MTZ SECs and includes Constraints (1), (4) to (18), (24), and (25) to (27). The related computational results obtained using dataset A are summarized in Table 2. Seven of the 30 problem instances in Dataset A were found to be infeasible and are therefore excluded from the presented results. In addition, among the 23 feasible instances, Problem 23 required a much longer CPU time than the others and, therefore, we present the results for this problem separately in Table 3 so that it does not dominate the computed average values. For all of the tables in this section, we highlight in each column the most favorable result in bold. Note that Formulation F1 produced better bounds and required less CPU time on average than Formulation F2. The application of default CPLEX cuts is very helpful for both formulations, as shown by the differences between Bound I and Bound II.

### 5.2. Effectiveness of valid inequalities

Next, we evaluated the effectiveness of the valid inequalities proposed in Section 3 by adding them individually to Formulation F1. Table 4 specifies how these cuts are configured. The results obtained using these valid inequalities for Dataset A are summarized in Table 5. Observe that adding any of these inequalities individually results in tightened LP bounds, although the average CPU time does not always decrease as a consequence. For the hardest instance (Problem 23), the CPU time was reduced almost universally except for the case of Cut G, which resulted in termination due to memory limitations (see the seventh column of Table 5).

We further investigated the simultaneous addition of multiple valid inequalities to the model formulation. Some selected cut combinations were chosen for exploration (see Table 6) based on the following observations: (i) among Cuts C1, C2, C3, and C4, which relate to the TCCs, C3 resulted in the shortest CPU time for the hardest instance, whereas C4 yielded the shortest average CPU time for the remaining instances; (ii) between Cuts P and G, which depend on upper bounds for production amounts, Cut P resulted in a relatively shorter CPU time for the hardest instance, whereas Cut G yielded a shorter average CPU time for the remaining instances; (iii) among Cuts X1, X2, and Z, which are based on the lot-sizing substructure, Cut Z resulted in the shortest CPU time for the hardest instance, whereas X1 yielded the shortest average CPU time for the remaining instances.

---

Table 2. Average performances of F1 and F2 for 22 instances of Dataset A

| Formulation | No. of non-zero coefficients | Optimality gap for bound I (%) | Optimality gap for bound II (%) | No. of nodes | CPU time (sec) |
|-------------|------------------------------|-------------------------------|-------------------------------|-------------|---------------|
| F1          | 9643                         | 74.57                         | 25.18                         | 42,529      | 71.7          |
| F2          | 8335                         | 75.17                         | 25.72                         | 81,124      | 131.6         |

Table 3. Performance of F1 and F2 for Problem 23 of Dataset A

| Formulation | No. of non-zero coefficients | Optimality gap for bound I (%) | Optimality gap for bound II (%) | No. of nodes | CPU time (sec) |
|-------------|------------------------------|-------------------------------|-------------------------------|-------------|---------------|
| F1          | 8788                         | 78.50                         | 60.50                         | 41,230,428  | 56,414.4      |
| F2          | 7900                         | 80.35                         | 60.11                         | 39,444,900* | 48,466.9*     |

*Solver terminated prematurely due to memory limitations.

Table 4. Cut configurations

| Name of cut | Inequalities added | Inequalities replaced |
|-------------|--------------------|----------------------|
| Cut C1      | (28)               | (15)                 |
| Cut C2      | (29)               | (15)                 |
| Cut C3      | (29)               | —                    |
| Cut C4      | (29), (30)         | —                    |
| Cut P       | (2), (3), (31)     | (24)                 |
| Cut G       | (32)               | —                    |
| Cut X1      | (33), (35), (36)   | —                    |
| Cut X2      | (37)               | —                    |
| Cut Z       | (38)               | —                    |
time for the remaining 22 instances; (iv) Cuts P and X1 resulted in the shortest CPU times for the hardest instance.

We implemented the selected cut combinations listed in Table 6 within Formulation F1 to solve the problem instances of Dataset A. The corresponding results are summarized in Table 7. Note that Combination 1 required the least average CPU time over all the 23 feasible instances and provided a significant improvement over simply using CPLEX’s defaults cuts.

We further applied this set of cuts to tighten Formulation F2. The results obtained are summarized in Tables 8 and 9. Note that compared with the results given in Tables 2 and 3, the addition of Combination 1 cuts tightened the LP relaxation and significantly reduced the number of nodes for both problem formulations. It is interesting to note that for the hardest instance (Problem 23), the CPU time required by Formulation F1 was reduced by more than 70%, from 56414.4 seconds to 15323.7 seconds (see the F1-related rows of Table 3 and Table 9, respectively). Although Formulation F2 with Combination 1 required the least average CPU time for the 22 relatively easier problem instances of Dataset A (see Table 8), it turns out that Formulation F1 with Cut Combination 1 yielded the shortest overall average CPU time, 730.1 seconds (see Table 7), versus 812.2 seconds required by Formulation F2 with Cut Combination 1 (note that this is obtained by combining the results of Table 8 and Table 9 as follows: $(22 \times 59.2 + 17377.4)/23 = 812.2$). Note that we have focused on experimenting with Cut Combination 1 based on its performance with Formulation F1, although it is possible that Formulation F2 might benefit more with some other combination of cuts. However, because of the generally superior performance of Formulation F1 over F2, we relegate this detailed investigation to future research.

5.3. Larger problem instances

Next, we applied the above model configurations to Dataset B, which consists of more difficult problem instances. To make the computations more manageable, we only solved these instances to a pre-specified optimality gap of 10%. For each of the 10 instances in Dataset B, we first applied Formulation 1 with Cut Combination 1 to solve it and recorded the corresponding CPU time as $t^*$ and then applied a time limit of $\max\{60, 2t^*\}$ seconds while solving the same instance with other model configurations to obtain comparative results. Since the optimal objective values were not always available, we regarded the best objective value obtained as the true optimal value for computing optimality gaps. These results are summarized in Table 10 and Fig. 4 (Problem 3 is infeasible and is thus excluded from this comparison).

Note that the addition of Cut Combination 1 to Formulation F1 leads to a significant reduction in CPU time. This observation is further illustrated in Fig. 4, where the

![Fig. 4. CPU times for problem instances in Dataset B.](image-url)
Table 7. Average performance of various cut combinations on Formulation F1 for 23 instances of Dataset A

| Cut combination | No. of non-zero coefficients | Optimality gap for bound I (%) | Optimality gap for bound II (%) | No. of nodes | CPU time (sec) | Max CPU time (sec) |
|-----------------|------------------------------|-------------------------------|-------------------------------|-------------|--------------|------------------|
| None            | 9606                         | 74.74                         | 26.72                         | 1833 307    | 2521.4       | 56 414.4         |
| 1               | 11 701                       | 38.19                         | 22.72                         | 585 169     | 730.1        | 15 323.7         |
| 2               | 10 812                       | 37.55                         | 21.49                         | 1264 895*   | 1466.4*      | 32 622.7*        |
| 3               | 12 188                       | 38.19                         | 22.56                         | 629 895     | 750.1        | 15 256.9         |
| 4               | 11 900                       | 38.30                         | 22.37                         | 1026 839    | 1448.3       | 31 527.5         |

*Solver terminated prematurely on Problem 23 due to memory limitations.

Table 8. Average performance of F1 and F2 with Cut Combination 1 for 22 instances of Dataset A

| Configuration          | No. of Non-zero coefficients | Optimality gap for bound I (%) | Optimality gap for bound II (%) | No. of nodes | CPU time (sec) |
|------------------------|------------------------------|-------------------------------|-------------------------------|-------------|--------------|
| Formulation F1 + Cut Comb. 1 | 11 738                       | 36.95                         | 21.27                         | 19 271      | 66.8         |
| Formulation F2 + Cut Comb. 1 | 10 430                       | 40.35                         | 21.34                         | 23 132      | 59.2         |

Table 9. Performance of F1 and F2 with Cut Combination 1 for Problem 23 of Dataset A

| Configuration          | No. of non-zero coefficients | Optimality gap for bound I (%) | Optimality gap for bound II (%) | No. of nodes | CPU time (sec) |
|------------------------|------------------------------|-------------------------------|-------------------------------|-------------|--------------|
| Formulation F1 + Cut Comb. 1 | 10 884                       | 65.42                         | 54.56                         | 13 034 931   | 15 323.7     |
| Formulation F2 + Cut Comb. 1 | 9996                        | 67.64                         | 53.45                         | 13 501 309   | 17 377.4     |

Table 10. Average performance of F1 and F2 for nine problem instances of Dataset B with and without Cut Combination 1

| Configuration          | No. of non-zero coefficients | Optimality gap for bound I (%) | Optimality gap for bound II (%) | No. of nodes | CPU time (sec) |
|------------------------|------------------------------|-------------------------------|-------------------------------|-------------|--------------|
| Formulation F1         | 48 329                       | 75.44                         | 36.22                         | 316 904     | 9342.9       |
| Formulation F1 + Cut Comb. 1 | 58 946                       | 47.36                         | 31.64                         | 160 177     | 6699.2       |
| Formulation F2 + Cut Comb. 1 | 52 140                       | 47.59                         | 31.65                         | 250 705     | 7627.4       |

arrows indicate that the solver terminated prematurely before an optimality gap of 10% was achieved, either due to the time limit (for Problems 1, 4, and 6) or due to memory limitations (for Problem 7). Although Formulation F2 with Cut Combination 1 is competitive for the relatively easier problem instances of Dataset B, its performance is dominated by Formulation F1 with Cut Combination 1 on average. We surmise that Cut Combination 1 seems to be effective because it exploits the batching aspect of the processing of a product.

5.4. Column generation method

To solve the PPMSP using the column generation method, we first tested the generation of incumbent solutions using our R-B heuristic described in Section 4. In Step 1 of this procedure, we used a target optimality gap of 10% and a time limit of 60 seconds to solve the relaxed problem. In Step 2, the restricted problem was solved to optimality to obtain an incumbent solution for the PPMSP. The results for the instances of Dataset A are summarized in Table 11.

Gap is defined as follows:

\[
100 \times \frac{z_h - z^*}{z^*} \%
\]

where \(z_h\) is the objective value of the solution obtained by the heuristic method, and \(z^*\) is the optimal objective value. For comparison, we also include in Table 11 the CPU time required to solve the problem by using Formulation F1 with Cut Combination 1 (direct solution). We next used the solution obtained from the R-B heuristic to construct an initial set of columns for the column generation method, in which Formulation F1 with Cut Combination 1 was decomposed according to the HMATSP2 scheme. The results of this step are also presented in Table 11. Note that the column generation step further reduced the average heuristic gap from 2.28 to 1.33% using a relatively short amount of CPU time. We also experimented with the Box-Step dual stabilization technique (Tran et al., 2006) in our column generation method. However, the corresponding results were not very promising, and so we refer the reader to Liao (2011) for further details.
Next, we applied the column generation method to the larger instances of Dataset B. Since the optimal objective value was not available for these instances, we used the best lower bound obtained in Section 5.3 (which turned out to be stronger than the bound values obtained by using our column generation method for all the problem instances of Dataset B) as the value of $z^*$ in Equation (53) to compute the relative optimality gap. As for Dataset A, we used the R-B heuristic to generate an initial set of columns for the column generation method. In addition, we experimented with incorporating Cut Combination 1 within the model formulations used for implementing the R-B heuristic. To limit the CPU time required to solve the MIP in Step 2 of the R-B heuristic, we imposed a time limit of 1800 seconds along with a 1% optimality gap tolerance.

The results for these two different versions of the R-B heuristic (with and without cuts) are summarized in Table 12, in comparison with those obtained by a direct solution of Formulation 1 with Cut Combination 1 using CPLEX. Note that the addition of Cut Combination 1 to the R-B heuristic reduced both the required CPU time and the optimality gap on average. Hence, continuing with Dataset B, we implemented this version to generate an initial set of columns for our proposed column generation method HMATSP2.

At the end of the column generation phase, a time limit of 3600 seconds was applied for the solution of the (restricted) master program as an MIP. The results obtained are also presented in Table 12. Note that, on average, the column generation method further reduced the optimality gap from 11.66 to 10.09%. The maximum gap was also significantly reduced from 22.54 to 12.96%. Since the R-B heuristic was used to generate initial columns for the column generation method, the summation of CPU times for these two steps gives the total CPU time required, which achieved an average value of $496.0 + 2313.6 = 2809.6$ seconds.

In order to check whether the best solutions are found early or late in the direct solution method, we further examined the progress of the objective value of the incumbent solutions in this direct solution method. As such, we collected the information on the CPU time required by the direct solution approach to reach the same solution values as the column generation method. This average CPU time was at least 6262 seconds (we used a target optimality gap of 10%, hence this is a conservative estimate). This is more than two times as long as the total time required by the R-B heuristic and the column generation method combined.

**Remark on implementation in practice:** Our computational study on the use of the proposed column generation-based method for the PPMSP has revealed its efficacy in solving real-life size problems. One of the key issues faced by a production control manager of a primary pharmaceutical manufacturing facility is to seek a right balance between the costs incurred in holding inventory and performing setups. In the proposed model, this is effectively achieved through the use of appropriate values of $h_i, \forall i \in N$, and $\sigma$ in the objective function. As the setup times involved in a primary pharmaceutical facility are rather long in duration, a general tendency for the production manager is to produce products in large quantities (typically, more than what is required) whenever a setup is scheduled at a bay. However, an application of our methodology to a real-life instance (involving 12 families, 15 bays, and 45 products) for which the data on actual batch sizes and setups used were available revealed a savings of 66% in total cost over that of the schedule actually implemented at the facility, where a large part of this saving was realized by a reduction in inventory cost. This was due to smaller batch sizes

---

**Table 11. Results for the direct solution method and the R-B heuristic for 23 problem instances of Dataset A**

| Method                              | Maximum heuristic gap (%) | Average heuristic gap (%) | Maximum CPU time (sec) | Average CPU time (sec) |
|-------------------------------------|---------------------------|---------------------------|------------------------|------------------------|
| Direct solution (F1 + Cut Comb. 1) | N/A                       | N/A                       | 15 323.7               | 730.1                  |
| R-B heuristic                       | 8.46                      | 2.28                      | 73.5                   | 19.9                   |
| Column generation with Cut Comb. 1  | 5.38                      | 1.33                      | 178.4                  | 38.8                   |

**Table 12. Results for the direct solution method, the R-B heuristic, and the column generation-based method for nine problem instances of Dataset B**

| Method                              | Maximum gap (%) | Average gap (%) | Maximum CPU time (sec) | Average CPU time (sec) |
|-------------------------------------|-----------------|-----------------|------------------------|------------------------|
| Direct solution                     | 11.11           | 9.74            | 51 159.8               | 6699.2                 |
| R-B heuristic                       | 18.11           | 14.32           | 1833.4                 | 589.1                  |
| R-B heuristic with Cut Comb. 1      | 22.54           | 11.66           | 1840.9                 | 496.0                  |
| Column generation with Cut Comb. 1  | 12.96           | 10.09           | 4322.1                 | 2313.6                 |
recommended by our approach (while still meeting demand constraints and slightly increasing the number of setups). This further highlights the importance of using a methodology of the type proposed herein for making judicious decisions in such complex environments.

6. Concluding remarks

The primary pharmaceutical manufacturing scheduling problem is an integrated lot-sizing and sequencing problem. We have developed a comprehensive mathematical programming model for this problem that includes realistic features of batch production, sequence-dependent setup time/cost, and setup splitting. Noting the resemblance of the campaign sequencing subproblem to the HMATSP, we have used HMATSP-inspired ideas to enforce the connectivity of the sequencing graph. Several additional classes of valid inequalities were derived that exploit different embedded problem substructures and help tighten the problem formulation. To solve this complex problem, we developed a column generation–based technique with an advanced-start set of initial columns. Computational results using real-life size problem instances demonstrated the effectiveness of our modeling and column generation–based solution approaches, which yielded solutions comparable to those obtained by a direct application of CPLEX to our cut-enhanced model, along with a more than 50% reduction in CPU time on average.

Acknowledgments

The authors thank the editors (in particular, the dedicated attention of Dr. Candace Yano) and three anonymous referees for their detailed and constructive comments that have greatly helped improve the clarity and substance of the presentation in this article.

Funding

This research has been supported by the National Science Foundation under grant CMMI-0856270.

Supplemental Material

Supplemental data for this article can be accessed on the publisher’s website.

References

Armentano, V.A., França, P.M. and de Toledo, F.M.B. (1999) A network flow model for the capacitated lot size problem. *Omega, 27*(2), 275–284.
Atamtürk, A. and Muñoz, J.C. (2004) A study of the lot-sizing polytope. *Mathematical Programming, 99*(3), 443–465.
Bazaraa, M.S., Jarvis, J.J. and Sherali, H.D. (2010) *Linear Programming and Network Flows*, fourth edition, John Wiley & Sons, New York, NY.

Bitran, G.R. and Yanasse, H.H. (1982) Computational complexity of the capacitated lot size problem. *Management Science, 28*(10), 1174–1186.
Dantzig, G., Fulkerson, R. and Johnson, S. (1954) Solution of a large-scale traveling-salesman problem. *Operations Research, 2*(4), 393–410.
Degraeve, Z. and Jans, R. (2007) A new Dantzig-Wolfe reformulation and branch-and-price algorithm for the capacitated lot-sizing problem with setup times. *Operations Research, 55*(5), 909–920.
Desaulniers, G., Desrosiers, J. and Solomon, M.M. (2005) *Column Generation*, Springer, New York, NY.
Diaby, M., Bahl, H.C., Karwan, M.H. and Zions, S. (1992) A Lagrangean relaxation approach for very-large-scale capacitated lot-sizing. *Management Science, 38*(9), 1329–1340.
Dogan, M.E. and Grossmann, I.E. (2006) A decomposition method for the simultaneous planning and scheduling of single-stage continuous multiproduct plants. *Industrial & Engineering Chemistry Research, 45*(1), 299–315.
Eppen, G.D. and Martin, R.K. (1987) Solving multi-item capacitated lot-sizing problems using variable redefinition. *Operations Research, 35*(6), 832–848.
Fandel, G. and Stammen-Hegene, C. (2006) Simultaneous lot sizing and scheduling for multi-product multi-level production. *International Journal of Production Economics, 104*(2), 308–316.
Federgruen, A. and Tzur, M. (1991) A simple forward algorithm to solve general dynamic lot sizing models with n periods in O(n log n) or O(n) time. *Management Science, 37*(8), 909–925.
Federgruen, A. and Tzur, M. (1994) The joint replenishment problem with time-varying costs and demands: efficient, asymptotic ε-optimal solutions. *Operations Research, 42*(6), 1067–1086.
Fleischmann, B. (1990) The discrete lot sizing and scheduling problem. *European Journal of Operational Research, 44*(3), 337–348.
Fleischmann, B. (1994) The discrete lot-sizing and scheduling problem with sequence-dependent setup costs. *European Journal of Operational Research, 75*(2), 395–404.
Florian, M. and Klein, M. (1971) Deterministic production planning with concave costs and capacity constraints. *Management Science, 18*(1), 12–20.
Florian, M. and Klein, M. (1972) Erratum: deterministic production planning with concave costs and capacity constraints. *Management Science, 18*(11), 721.
Fumero, Y., Corsano, G. and Montagna, J.M. (2011) Detailed design of multiproduct batch plants considering production scheduling. *Industrial & Engineering Chemistry Research, 50*(10), 6146–6160.
Ghoniem, A. and Sherali, H.D. (2009) Complementary column generation and bounding approaches for set partitioning formulations. *Optimization Letters, 3*(1), 123–136.
Glover, F. and Sherali, H.D. (2005) Some classes of valid inequalities and convex hull characterizations for dynamic fixed-charge problems under nested constraints. *Annals of Operations Research, 140*(1), 215–233.
Graves, S.C. (1982) Using Lagrangean techniques to solve hierarchical production planning problems. *Management Science, 28*(3), 260–275.
Grigoriev, A. and van de Klundert, J. (2006) On the high multiplicity traveling salesman problem. *Discrete Optimization, 3*(1), 50–62.
Gupta, D. and Magnusson, T. (2005) The capacitated lot-sizing and scheduling problem with sequence-dependent setup costs and setup times. *Computers & Operations Research, 32*(4), 727–747.
Haase, K. and Kimms, A. (2000) Lot sizing and scheduling with sequence-dependent setup costs and times and efficient rescheduling opportunities. *International Journal of Production Economics, 66*(2), 159–169.
Hindi, K.S. (1995) Solving the single-item, capacitated dynamic lot-sizing problem with startup and reservation costs by tabu search. *Computers & Industrial Engineering, 28*(4), 701–707.
Kimms, A. (1997) Demand shuffle—a method for multilevel proportional lot sizing and scheduling. *Naval Research Logistics, 44*(4), 319–340.

Kondili, E., Pantelides, C.C. and Sargent, R.W.H. (1993) A general algorithm for short-term scheduling of batch operations—I. MILP formulation. *Computers & Chemical Engineering, 17*, 211–227.

Leung, J.M.Y., Magnanti, T.L. and Vachani, R. (1989) Facets and algorithms for capacitated lot sizing. *Mathematical Programming, 45*(2), 331–359.

Liao, L. (2011) Novel approaches for some stochastic and deterministic scheduling problems, doctoral dissertation, Virginia Polytechnic Institute and State University, Blacksburg, VA.

Loparic, M., Marchand, H. and Wolsey, L.A. (2003) Dynamic knapsack sets and capacitated lot-sizing. *Mathematical Programming, 95*(1), 53–69.

Meyr, H. (2002) Simultaneous lot-sizing and scheduling on parallel machines. *European Journal of Operational Research, 139*(2), 277–292.

Miller, A.J., Nemhauser, G.L. and Savelsbergh, M.W.P. (2000) On the capacitated lot-sizing and continuous 0-1 knapsack polyhedra. *European Journal of Operational Research, 125*(2), 298–315.

Miller, C.E., Tucker, A.W. and Zemlin, R.A. (1960) Integer programming formulation of traveling salesman problems. *Journal of the ACM, 7*(4), 326–329.

Nemhauser, G.L. and Wolsey, L.A. (1999) *Integer and Combinatorial Optimization*, second edition, Wiley, New York, NY.

Öncan, T., Altinel, I.K. and Laporte, G. (2009) A comparative analysis of several asymmetric traveling salesman problem formulations. *Computers & Operations Research, 36*(3), 637–654.

Papageorgiou, L.G. and Pantelides, C.C. (1996) Optimal campaign planning/scheduling of multipurpose batch/semicontinuous plants. I. Mathematical formulation. *Industrial & Engineering Chemistry Research, 35*, 488–509.

Pochet, Y. (1988) Valid inequalities and separation for capacitated economic lot sizing. *Operations Research Letters, 7*(3), 109–115.

Robinson, E.P., Jr., and Lawrence, F.B. (2004) Coordinated capacitated lot-sizing problem with dynamic demand: a Lagrangean heuristic. *Decision Sciences, 35*(1), 25–53.

Sarin, S.C., Sherali, H.D. and Bhoobatta, A. (2005) New tighter polynomial length formulations for the asymmetric traveling salesman problem with and without precedence constraints. *Operations Research Letters, 33*(1), 62–70.

Sarin, S.C., Sherali, H.D. and Yao, L. (2011) New formulation for the high multiplicity asymmetric traveling salesman problem with application to the Chesapeake case. *Optimization Letters, 45*, 259–272.

Shah, N. (2004) Pharmaceutical supply chains: key issues and strategies for optimisation. *Computers & Chemical Engineering, 28*(6–7), 929–941.

Sherali, H.D., Sarin, S.C. and Tsai, P.-F. (2006) A class of lifted path and flow-based formulations for the asymmetric traveling salesman problem with and without precedence constraints. *Discrete Optimization, 3*(1), 20–32.

Stadtler, H. (2003) Multilevel lot sizing with setup times and multiple constrained resources: internally rolling schedules with lot-sizing windows. *Operations Research, 51*(3), 487–502.

Tran, V.H., Reinelt, G. and Bock, H.G. (2006) BoxStep methods for crew pairing problems. *Optimization and Engineering, 7*(1), 33–46.

Verderame, P.M. and Floudas, C.A. (2010) Integration of operational planning and medium-term scheduling for large-scale industrial batch plants under demand and processing time uncertainty. *Industrial Engineering and Chemistry Research, 49*(10), 4948–4965.

Wagelmans, A., van Hoesel, S. and Kolen, A. (1992) Economic lot sizing: an O(n log n) algorithm that runs in linear time in the Wagner-Whitin case. *Operations Research, 40*, 145–156.

Wagner, H.M. and Whitin, T.M. (1958) Dynamic version of the economic lot size model. *Management Science, 5*(1), 89–96.

Wong, R.T. (1980) Integer programming formulations of the traveling salesman problem, in *Proceedings of the IEEE International Conference of Circuits and Computers*, IEEE Press, Piscataway, NJ, pp. 149–152.

**Biographies**

Subhash C. Sarin is the Paul T. Norton Endowed Professor in the Grado Department of Industrial and Systems Engineering at Virginia Tech. His areas of specialty are production scheduling, applied mathematical programming, and the design and analysis of manufacturing systems. He has published several papers in these areas and has also co-authored two books. He is recipient of several prestigious awards at the university, state, and national levels. He has served on the editorial boards of many journals. He is a Fellow of the Institute of Industrial Engineers and a Full member of the Institute for Operations Research and Management Sciences.

Hani D. Sherali is a University Distinguished Professor Emeritus in the Industrial and Systems Engineering Department at Virginia Polytechnic Institute and State University. His areas of research interest are in analyzing problems and designing algorithms for specially structured linear, nonlinear, and integer programs arising in various applications, global optimization methods for non-convex programming problems, location and transportation theory and applications, and economic and energy mathematical modeling and analysis. He has published over 317 refereed articles in various operations research journals and has (co-)authored eight books in this area. He is a Fellow of INFORMS and IIE and an elected member of the National Academy of Engineering.

Lingrui Liao completed his doctorate degree from the Grado Department of Industrial and Systems Engineering at Virginia Tech. He is a graduate of Tsinghua University, Beijing, China. He is a co-author of a book, and his papers have appeared in *Naval Research Logistics* and *Journal of Scheduling*. Currently, he is working for Google.