Open access • Journal Article • DOI:10.1021/IE050778Z

A Decomposition Method for the Simultaneous Planning and Scheduling of Single-Stage Continuous Multiproduct Plants — Source link

Muge Erdirik Dogan, Ignacio E. Grossmann

Institutions: Carnegie Mellon University

Published on: 01 Jan 2006 - Industrial & Engineering Chemistry Research (American Chemical Society)

Topics: Job shop scheduling, Scheduling (production processes), Linear programming, Decomposition method (constraint satisfaction) and Time horizon

A Decomposition Method for the Simultaneous Planning and Scheduling of Single Stage Continuous Multiproduct Plants

Muge Erdirik Dogan and Ignacio E. Grossmann*
Department of Chemical Engineering
Carnegie Mellon University
Pittsburgh, PA 15213, U.S.A.

June 2005

***Author to whom correspondence should be addressed (grossmann@cmu.edu).**

Abstract

In this paper, we address the problem of simultaneously integrating planning and scheduling of continuous multiproduct plants consisting of a single processing unit. We present a multiperiod MILP optimization model that is based on a continuous time representation, which becomes computationally very expensive to solve as the length of the planning horizon increases. To circumvent this problem a rigorous bi-level decomposition algorithm is proposed to reduce the computational cost of the problem. The original simultaneous model is decomposed into an upper level planning problem and a lower level planning and scheduling problem. The upper level determines the potential products to be processed, their production levels and inventories. The lower level is solved in the reduced space of binary variables and determines production levels, product inventories, and the detailed sequence of products and their corresponding processing times. Integer cuts and logic cuts are proposed to reduce the feasible search space for the binary variables and to tighten the gap between the solutions of the two levels. Numerical examples for problems ranging from 4 to 24 weeks are presented to illustrate the performance of the algorithm and to compare it with a full space solution.

Introduction

Planning and scheduling of process systems are closely linked activities. Both planning and scheduling deal with the allocation of available resources over time to perform a collection of tasks required to manufacture one or several products (Bodington, 1995; Shah, 1998, Kallrath, 2002). The aim in planning is to determine high level decisions such as production levels and product inventories for given marketing forecasts and demands over a long time horizon (e.g. months to years). Scheduling, on the other hand, is defined over a short time horizon (e.g. days to weeks) and involves lower level decisions such as the sequence and detailed timing in which various products should be processed at each equipment in order to meet the production goals set by the planning problem.

Conceptually, the simplest alternative for solving planning and scheduling problems is to formulate a single simultaneous planning and scheduling model that spans the entire planning horizon of interest. However, the limitation of this approach is that when typical planning horizons are considered, the size of this detailed model becomes intractable due to the potential exponential increase in the computation. The traditional strategy for solving planning and scheduling problems is to follow a hierarchical approach in which the planning problem is solved first to define the production targets. The scheduling problem is solved next to meet these targets (Bodington, 1995; Shapiro, 2001). The problem of this approach, however, is that a solution determined at the planning level does not necessarily lead to feasible schedules. These infeasibilities may arise because the effects of changeovers are neglected at the planning level, thereby producing optimistic targets that cannot be met at the scheduling level. Therefore, there is a need to develop methods and approaches that can more effectively integrate planning and scheduling (Grossmann, 2005).

Most of the work that has been reported on integrating planning and scheduling has focused on batch processes and is based on two-level decomposition schemes (Graves, 1982). Bassett et al (1996) proposed a decomposition scheme for multipurpose batch plants, where an aggregate planning problem is solved in the upper level, and detailed scheduling problems are independently solved for each planning period in the lower level. Heuristic techniques that make use of shifting of operations were proposed to overcome the infeasibilities that arise in the scheduling problem. Bassett, Pekny, and Reklaitis (1996) introduced slack variables for capacity or inventory shortfalls to remove the infeasibilities. These authors also proposed a recursive backwards rolling horizon strategy where only one interval is solved in detail at every stage of the recursion. Changeover costs and times were not considered in the scheduling model. Subrahmanyam et al (1996) proposed a hierarchical decomposition algorithm for batch plants, where the planning problem is updated at each iteration by disaggregating the aggregate constraints for all infeasible scheduling subproblems within the planning problem. The drawback of this procedure is that it may require exploring all levels of decomposition. Birewar and Grossmann (1990) proposed a multiperiod LP formulation for simultaneous planning and scheduling of multiproduct batch plants with flowshop structure. In this formulation, batches belonging to the same products are aggregated and sequencing considerations for scheduling are accounted at the planning level by approximating the makespan with the cycle time. Production shortfalls are treated through penalties. Wilkinson et al (1996) used a constraint aggregation approach for obtaining approximate solutions to the large-scale production and distribution planning problems for multiple production sites that are represented with the State-Task Network (Kondili et al. 1993). In their work, an upper level aggregate model is solved to set production targets yielding a strict upper bound to the original problem, after which detailed scheduling is individually optimized for each site with fixed targets thus decreasing the computational effort. Zhu and Majozi (2001) proposed a two-level decomposition strategy for multipurpose batch plants. In the first level, the planning model is solved for the optimal allocation of raw materials to individual processes, and in the second level the raw material targets obtained at the planning model are incorporated into the scheduling models for individual processes and then solved independently. If the scheduling targets corresponding to the raw material inputs do not match the production targets predicted by the planning model, the latter is revised with more realistic targets predicted by the scheduling model.

The major goal of this paper is to propose a novel bilevel decomposition procedure that allows rigorous integration and optimization of planning and scheduling of continuous multiproduct plants consisting of a single processing unit. The proposed integration scheme ensures consistency and optimality within a specified tolerance while significantly reducing the computational effort. The paper is organized as follows. The MILP formulation is first presented. We then introduce the detailed and the aggregated models. The proposed decomposition algorithm is then described, which relies on the use of a novel set of cuts and is guaranteed to produce the same optimal solution as the full-space model. Finally, the effectiveness of the algorithm is demonstrated with several examples.

Problem Definition

Given are a number of products that are to be manufactured on a single continuously operating unit. Also, given is a planning horizon which is subdivided into weeks. At the end of each week, demands in the form of lower bounds are specified for each product. Constant production rates, production costs and selling prices are given for each product. Sequence dependent transition times that arise from the processing of two successive products are also given as well as the corresponding transition costs. The change of inventory levels with respect to time must be taken into account, and hence the corresponding inventory costs. The problem is then to determine the products to be produced in each week, the sequencing of these products, length of production times, amounts to be produced, and inventory levels for each product. The objective is to maximize the total profit in terms of sales revenues, operating costs, inventory costs and transition costs.

MILP Model

In this section basic ideas of the proposed mathematical model will be discussed. Demands, due dates and prices are assumed to be deterministic and a continuous time representation is adopted in this model. For convenience we assume that the due dates are specified at the end of each week of the specified time horizon.

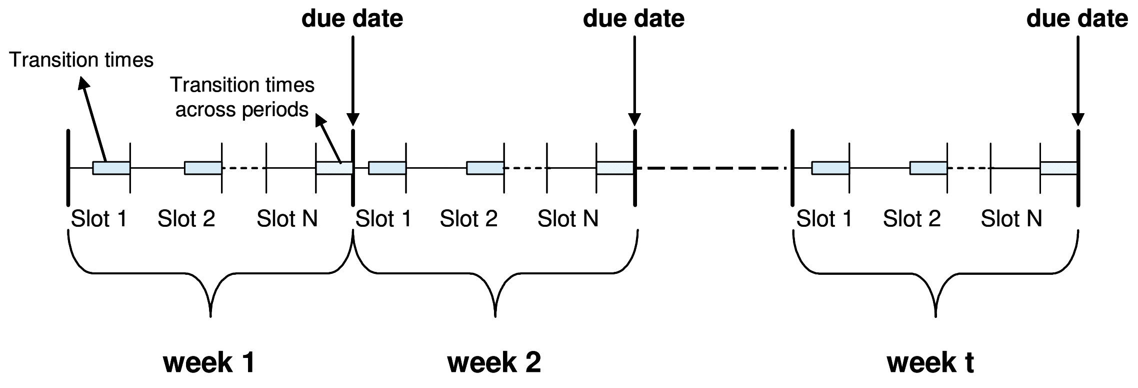

Figure 1. Time slots postulated for each week

N time slots are postulated in each week (Figure 1), where N is the total number of products. The length of each time slot, which consists of the assigned product's processing time and the corresponding transition time, is a continuous variable to be determined. The assignments of products to these time slots, is to be determined to define the sequence of the processing of the products. Binary variables $W_{i\ell t}$ are used to model the potential assignment of product, i, to slot ℓ in time period t. In order to model the transition times across adjacent weeks we enforce the constraint that each slot be utilized.

Both the transitions within each week and transitions across the weeks are activated depending on the assignments of products to slots and time periods.

The indices, parameters, and variables defined in the model are as follows:

1. Indices

Products	$i, k = 1, \ldots.., N$
Time Slots	$\ell, \ell\ell = 1, \ldots., N$
Time Periods	$t = 1, \ldots\ldots, Htot$

2. Variables

$W_{i\ell t}$	0-1 variable to denote if product i is assigned to slot ℓ of period t
YOP_{it}	0-1 variable to denote if product i is assigned to period t
$Z_{ik\ell t}$	to denote if product i is followed by product k in slot ℓ of period t
TRT_{ikt}	to denote if product i is followed by product k at the end of period t
NY_{it}	number of slots that product i is assigned in period t.
$\tilde{X}_{i\ell t}$	amount produced of i in slot ℓ of period t
X_{it}	amount produced of i in period t
$\tilde{\theta}_{i\ell t}$	production time of i in slot ℓ of period t
θ_{it}	production time of i in period t
$Ts_{\ell t}$	start time of slot ℓ in period t
$Te_{\ell t}$	end time of slot ℓ in period t
INV_{it}	inventory level of product i at the end of time period t
$INVO_{it}$	final inventory of product i at time t after the demands are satisfied
$AREA_{it}$	area below the inventory time graph for product i at period t
S_{it}	sales of i in period t
z^{p}	total profit over given time horizon

3. Parameters

r_i	Production rates
d_{it}	demand of product i in period t

τ_{ik}	transition time from product i to product k
$INVI_{io}$	initial inventory level of product i
c_{inv}	inventory cost
c_{it}^{oper}	operating cost for product i in period t
c_{ik}^{tran}	transition cost from product i to k
p_{it}	selling price of product i in period t
TR_i	minimum transition time for product i
U	maximum transition time
H_t	duration of the t^th time period
HT_t	time in terms of hours at the end of t^th time period
$Htot$	time at the end of the planning horizon

The MILP model (P) for the planning and scheduling problem is as follows:

a) Objective function

Maximize

$$z^P = \sum_i \sum_t p_{it} S_{it} - c_{inv} \sum_i \sum_t Area_{it} - \sum_i \sum_t c_{it}^{oper} * \tilde{X}_{ilt} - \sum_i \sum_k \sum_\ell \sum_t c_{ik}^{trans} * Z_{iklt}$$
$$- \sum_i \sum_k c_{ik}^{trans} * TRT_{ikt} \qquad (1)$$

The profit is given by the sum of sales revenues, the inventory costs, the operating costs, the transition costs within each week and transition costs across adjacent weeks. The cost coefficients in the objective function are defined so as to yield a profit in (\$). The selling price p_{it} and cost coefficient c_{it}^{oper} are in \$/kg whereas cost coefficient c_{ik}^{tran} is in \$ and the cost coefficient c_{inv} is in \$/kg.h.

b) Assignment and processing times

$$\sum_i W_{i\ell t} = 1 \quad \forall \ell, t \qquad (2)$$

$$0 \le \tilde{\theta}_{i\ell t} \le H_t * W_{i\ell t} \quad \forall i, \ell, t \qquad (3a)$$

$$\theta_{it} = \sum_\ell \tilde{\theta}_{i\ell t} \quad \forall i, t \qquad (3b)$$

$$\tilde{X}_{i\ell t} = r_i * \tilde{\theta}_{i\ell t} \quad \forall i, \ell, t \qquad (4a)$$

$$X_{it} = \sum_{\ell} \tilde{X}_{i\ell t} \quad \forall i,t \tag{4b}$$

Equation (2) represents the condition that exactly one product can be produced in each slot. Note that the same product can be produced in more than one slot in time period t. Also, the total number of time slots is equal to the number of total products.

According to constraint (3a), the time devoted to production of product i at slot ℓ of time period t is zero unless the product is assigned to slot ℓ of time t. The length of the week, H_t, is a valid upper bound on the processing time. Equation (3b) states that the processing time of product i at time period t, is the sum of processing times over the slots that are being utilized by product i at time t. Equations (4a) and (4b) represent the amounts produced, which are directly proportional to processing times and production rates. These production rates are constants that are product dependent.

c) Transitions

$$Z_{ik\ell t} \geq W_{i\ell t} + W_{k,\ell+1,t} - 1 \quad \forall i,k,\ell,t \tag{5}$$

Constraint (5) defines the transition variable $Z_{ik\ell t}$ which represents the transitions that occur within each week. $Z_{ik\ell t}$ is 1 if product i is followed by product k at slot ℓ of time period t, and becomes zero otherwise. Since transition costs are minimized in the objective function, the variables $Z_{ik\ell t}$ can be treated as continuous variables, $0 \leq Z_{ik\ell t} \leq 1$.

d) Timing relations

$$Te_{\ell t} = Ts_{\ell t} + \sum_{i} \tilde{\theta}_{i\ell t} + \sum_{i}\sum_{k} \tau_{ik} Z_{ik\ell t} \quad \forall \ell,t \tag{6}$$

$$TRT_{ikt} \geq W_{i\ell t} + W_{k\ell\ell t+1} - 1 \quad \forall i,k,t,\ell = N, \ell\ell = 1 \tag{7}$$

$$Te_{\ell t} + \sum_{i}\sum_{k} \tau_{ik} * TRT_{ikt} = Ts_{\ell\ell t+1} \quad \forall t, \ell = N, \ell\ell = 1 \tag{8}$$

$$Te_{\ell t} = Ts_{\ell+1 t} \quad \forall \ell \neq N, t \tag{9}$$

$$Te_{Nt} \leq HT_t \quad \forall t \tag{10}$$

According to equation (6), the end time of a slot is equal to the starting time plus summation of the processing times of the products that are being produced in that slot and the corresponding transition times. Note that according to equations (2) and (3a), exactly one processing time is non zero in the summation term of (6). Constraint (7) defines the transition variables for the transitions that occur across successive weeks. Transition variable TRT_{ikt} will become 1 if both $W_{i\ell t}$ and $W_{k\ell\ell t+1}$ are one. On the other hand, if at least one of them is zero, the constraint becomes redundant. Since transitions are cost terms in the model, they will naturally be set to zero, and therefore they can be treated as continuous variables. According to constraint (8), the start time of the first slot of time period t has to be equal to the end time of the last slot of the previous time period summed with the corresponding transition time between the time periods. Equation (9) ensures that the end time of one slot is equal to the start time of the preceding slot. The inequality in (10) ensures that the end time of the last slot in time period t can not exceed the duration of time period t.

e) Inventory

$$INV_{it} = INVI_{io} + \sum_{\ell} r_i * \tilde{\theta}_{i\ell t} \quad \forall i, t = 1 \tag{11a}$$

$$INV_{it} = INVO_{it-1} + \sum_{\ell} r_i * \tilde{\theta}_{i\ell t} \quad \forall i, t \neq 1 \tag{11b}$$

$$INVO_{it} = INV_{it} - S_{it} \quad \forall i, t \tag{12}$$

$$Area_{it} \geq INVO_{it-1} * H_t + r_i * \theta_{it} * H_t \quad \forall i, t \tag{13}$$

In equation (11a), the inventory level of product i in the first time period is defined as the sum of the initial inventory level for product i and the amount of product i produced in the first time period. In equation (11b), the inventory level of product i at the end of time t is defined as the sum of the final inventory level of product i at the end of time t-1, and the amount of product i produced during time period t. In equation (12), the final inventory level of product i is defined by subtracting the sales S_i. This final inventory level is the one after demands are satisfied at the end of each time period.

Since the inventory changes with time, the inventory cost will be proportional to the integral of the inventory function along time. The integral of the inventory function along time is equal to the area below the inventory function (Figure 2) as given by equation (14).

$$Area_{it} = INVO_{it-1} * H_t + r_i * \frac{\theta_{it} * \theta_{it}}{2} + (HT_t - Te_{\ell t}) * r_i * \theta_{it} \quad \forall i, \ell, t \tag{14}$$

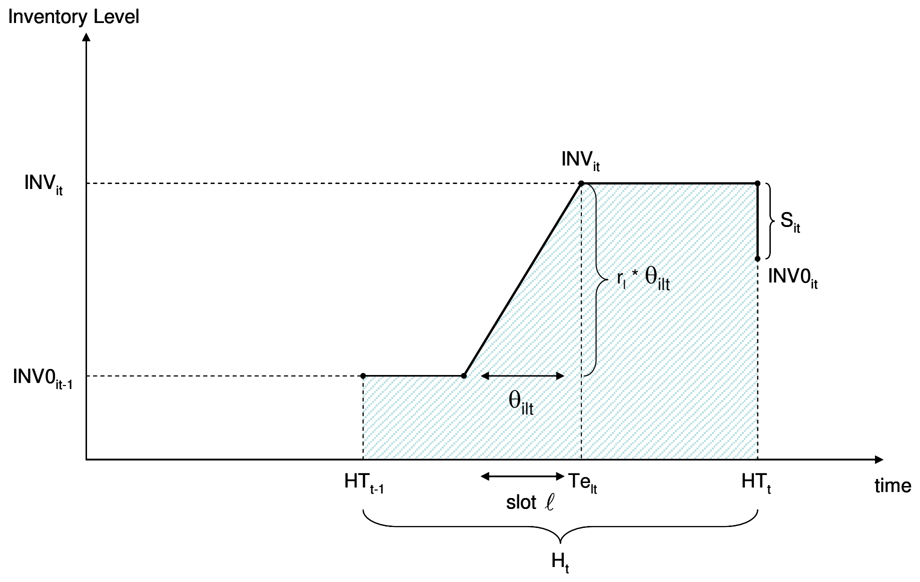

Figure 2. Inventory changes with time.

This equation, however, is nonlinear and nonconvex. To avoid the difficulty of handling such terms, we propose to use equation (13), which is a linear overestimation of the exact area that yields a valid upper bound on the inventory cost. The exact inventory cost and the corresponding overestimation are shown in Figure 3. It should be noted that for the time periods when no production occurs, equations (13) and (14) give the same inventory cost.

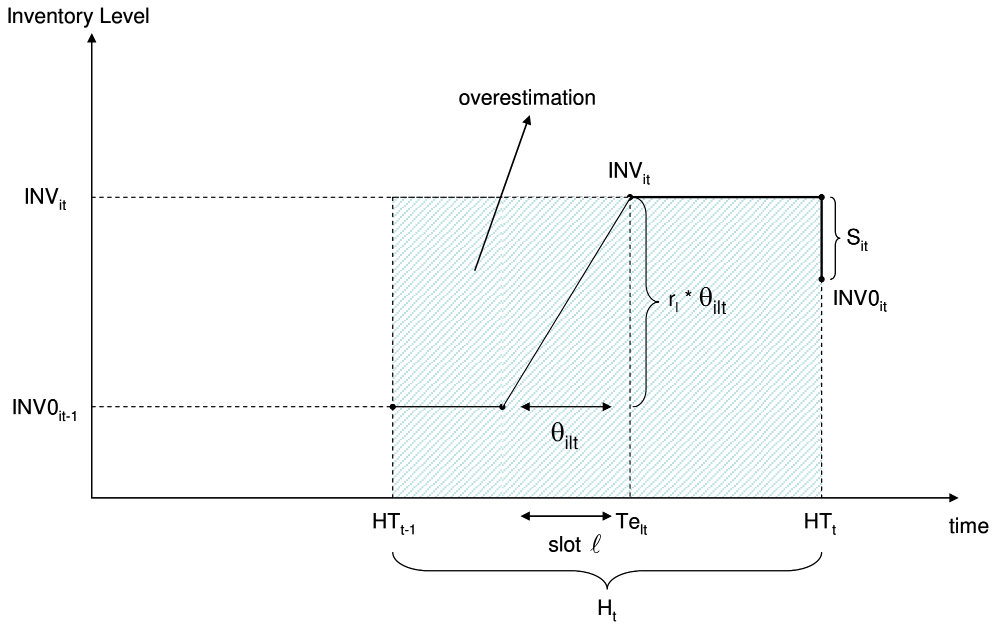

Figure 3. Overestimation of Inventory

f) Demand

$S_{it} \geq d_{it} \quad \forall i,t$ (15)

Constraint (15) states that the demand must be satisfied for all products in the plant and that the production may be exceeded. Note that demands to be satisfied are defined as lower bounds. Also, setting the values of the demands too high might lead to infeasible solutions.

It is important to note in the proposed MILP model that when the same product is assigned to more than one slot in time period t, the model will not assign that product to nonconsecutive slots. This is due to the fact that when nonconsecutive slots are used this will result in the same inventory cost but in higher transition costs. As an example in Figure 4, solution 1 is a feasible but non-optimal configuration. Therefore, we do not need to enforce the utilization of consecutive slots for the same product as this will be handled by optimality in the search.

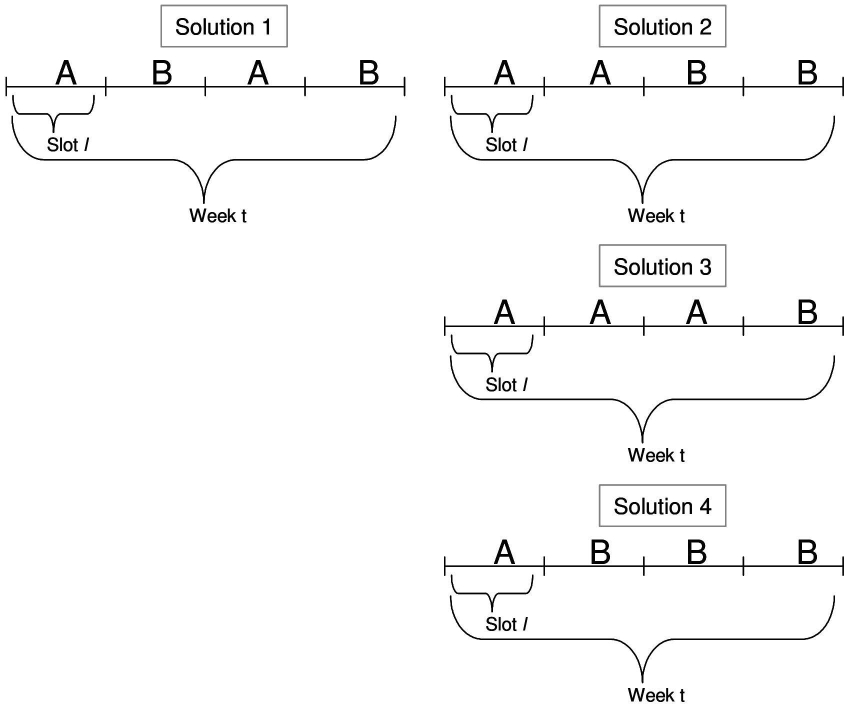

Figure 4. Symmetric Solutions.

However, using more than one slot for the same product can make the problem highly degenerate. Consider the case of solution 2 in Figure 4, where product A is produced in two consecutive slots. If the processing times of product A in the solution are p_1 and p_2, allocating the total processing time p_1+p_2 in any feasible way in week t will result in several alternative optima. As seen in Figure 4, solution 2, solution 3 and solution 4 are all equivalent. In order to prevent these degenerate solutions, we propose to add symmetry breaking constraints where we enforce the product that is assigned to the first slot to utilize all the slots except one slot for each of the other assigned products. For the case shown in Figure 4, the only feasible optimal solution would be solution 3.

The symmetry breaking constraints can be represented in logic form as follows:

$$\underset{\ell}{\vee} W_{i\ell t} \Rightarrow YOP_{it} \quad \forall i,t \tag{16}$$

$$\begin{bmatrix} YOP_{it} \\ NY_{it} \geq 1 \end{bmatrix} \vee \begin{bmatrix} \neg YOP_{it} \\ NY_{it} = 0 \end{bmatrix} \quad \forall i,t \tag{17}$$

$$\begin{bmatrix} W_{i1t} \\ NY_{it} = N - \left[\left(\sum_i YOP_{it}\right) - 1\right] \end{bmatrix} \vee \begin{bmatrix} \neg W_{i1t} \\ NY_{it} \geq 0 \end{bmatrix} \quad \forall i,t \tag{18}$$

where

$$NY_{it} = \sum_{\ell} W_{i\ell t} \quad \forall i,t \tag{19}$$

The implication in (16) states that if product i is assigned to at least one slot ℓ in time t then YOP_{it}, the assignment of i in week t, is true. The disjunction in (17) states that if YOP is true then there is at least one assignment of product i in time t; otherwise it is zero. Finally, the disjunction in (18) establishes that if product i is assigned to the first slot of time t then the number of assigned slots for product i, NY_{it} must be equal to the total number of slots N, minus the total number of product assignments in time t, minus one.

Transforming (16) to an inequality, using the convex hull transformation of (17) and the big-M reformulation of (18) yields (Raman and Grossmann, 1994),

$$YOP_{it} \geq W_{i\ell t} \quad \forall i,\ell,t \tag{20}$$

$$YOP_{it} \leq NY_{it} \leq N * YOP_{it} \quad \forall i,t \tag{21}$$

$$NY_{it} \geq N - \left[\left(\sum_i YOP_{it}\right) - 1\right] - M * (1 - W_{i1t}) \quad \forall i,t \tag{22}$$

$$NY_{it} \leq N - \left[\left(\sum_i YOP_{it} \right) - 1 \right] + M * (1 - W_{i1t}) \quad \forall i,t \qquad (23)$$

where M is a parameter that is chosen with a sufficiently large value to render (22) and (23) redundant when $W_{i1t} = 0$.

However, the proposed MILP model (1)-(13), (15) and (19)-(23), can become intractable when large number of products and long planning horizons are considered. As a result, simultaneous optimization techniques can fail to solve this model for large instances. Therefore, we propose a novel decomposition scheme for the integration of planning and scheduling that enables the rigorous solution at reasonable computational expense. For large scale problems we describe an approximation scheme. The proposed algorithm is described in the next section.

Solution Strategy/Decomposition Algorithm

In order to avoid the direct solution of the MILP model in the previous section, we propose a bi-level decomposition algorithm that exploits the hierarchical structure of planning and scheduling models. In particular, the original detailed planning and scheduling model is decomposed into an upper level planning and a lower level planning and scheduling problem. The upper level problem (PA) determines the products to be produced in each week, production levels and product inventories. (PA) is a relaxation of the original problem (P), and thus it yields an upper bound on the profit. In the lower level problem, the original problem (P) is solved by excluding products that were not selected by the upper level problem (PA). The lower level problem (P^r) corresponds to the sub problem of the MILP model (P) at iteration r and is in a reduced space since a subset of products is selected from the upper level (PA). A lower bound is obtained from the solution of (P^r) since its solution corresponds to a feasible solution of the original problem (P). The procedure iterates until the difference between the upper and the lower bounds is less than a specified tolerance. To expedite the search, integer cuts and logic cuts are added to the upper level. Integer cuts are used to exclude previous solutions, and logic cuts are used to exclude subsets and supersets of previously obtained configurations at the upper level problem, as well as potential solutions that violate capacity constraints. The decomposition algorithm is illustrated schematically in Figure 5.

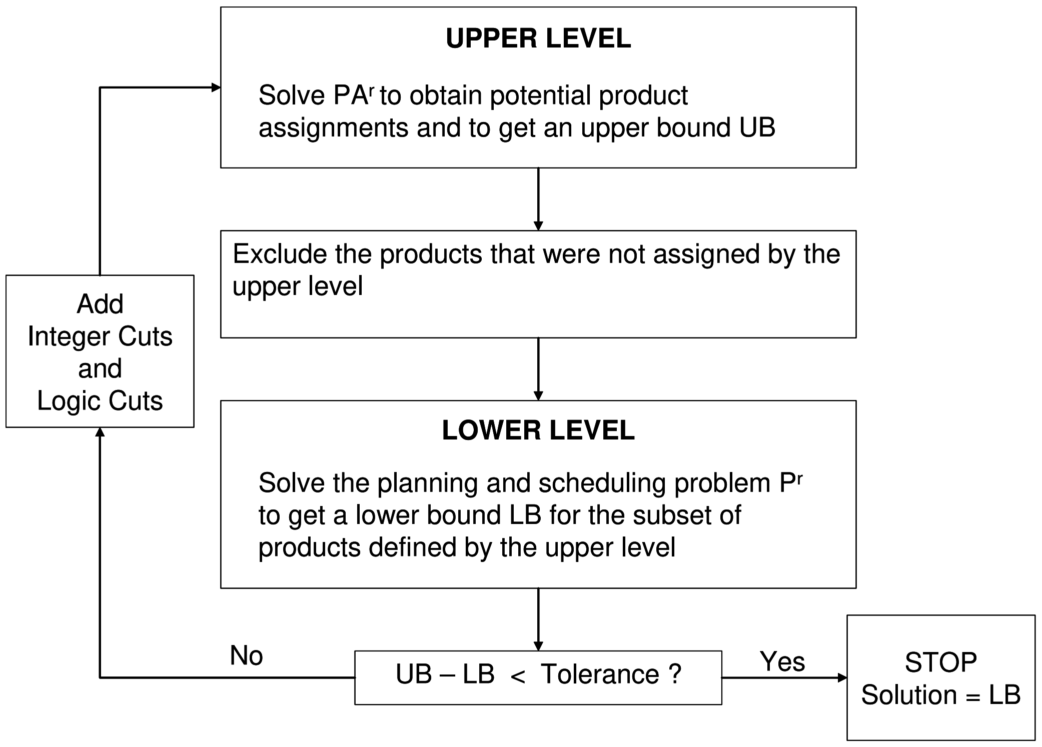
UPPER LEVEL
Solve PA^r to obtain potential product assignments and to get an upper bound UB
Exclude the products that were not assigned by the upper level
LOWER LEVEL
Solve the planning and scheduling problem P^r to get a lower bound LB for the subset of products defined by the upper level
UB – LB < Tolerance ?
No
Yes
STOP
Solution = LB
Add Integer Cuts and Logic Cuts

Figure 5. Flowchart for the proposed algorithm

Upper Level Problem

The aggregated MILP model (PA) is used to predict an upper bound. It is based on the idea of ignoring the detailed sequencing constraints and is largely concerned with the assignment of products at each week through the binary variables Y_{it}.

The indices, parameters, and variables defined in the model are as follows:

1. Indices

Products	$i, k = 1, \ldots\ldots, N$
Time Periods	$t = 1, \ldots\ldots, Htot$

2. Parameters

r_i	Production rates
d_{it}	demand of product i in period t
$INVI_{io}$	initial inventory level of product i
c_{inv}	inventory cost
c_{it}^{oper}	operating cost for product i in period t

p_{it}	selling price of product i in period t
TR_i	minimum transition time for product i
TRC_i	minimum transition cost for product i
H_t	duration of the tth time period

3. Variables

Y_{it}	0-1 variable to denote if product i is assigned to period t
U_t	maximum of the minimum transition times of products assigned to period t
$UTRC_t$	maximum of the minimum transition costs of products assigned to period t
X_{it}	amount produced of i in period t
θ_{it}	production time of i in period t
INV_{it}	inventory level of product i at the end of time period t
$INVO_{it}$	final inventory of product i at time t after the demands are satisfied
$AREA_{it}$	area below the inventory time graph for product i at period t
S_{it}	sales of i in period t

The proposed model representing the upper level is shown below:

$$Max\ z^{PA} = \sum_i \sum_t p_{it} S_{it} - c_{inv} \sum_i \sum_t Area_{it} - \sum_i \sum_t c_{it}^{oper} X_{it} - \sum_t \left((\sum_i TRC_i * Y_{it}) - UTRC_t \right) \quad (24)$$

$$X_{it} = r_i * \theta_{it} \quad \forall i,t \qquad (25)$$

$$\theta_{it} \le H_t * Y_{it} \quad \forall i,t \qquad (26)$$

$$U_t \ge TR_i * Y_{it} \quad \forall i,t \qquad (27a)$$

$$U_t \le \max_i \{TR_i\} \quad \forall t \qquad (27b)$$

$$\sum_i (\theta_{it} + TR_i * Y_{it}) - U_t \le H_t \quad \forall t \qquad (28)$$

$$UTRC_t \ge TRC_i * Y_{it} \quad \forall t \qquad (29a)$$

$$UTRC_t \le \max_i \{TRC_i\} \quad \forall i,t \qquad (29b)$$

$$INV_{it} = INVI_{io} + r_i * \theta_{it} \quad \forall i, t = 1 \tag{30}$$

$$INV_{it} = INVO_{it-1} + r_i * \theta_{it} \quad \forall i, t \neq 1 \tag{31}$$

$$INVO_{it} = INV_{it} - S_{it} \quad \forall i, t \tag{32}$$

$$Area_{it} \geq INVO_{it-1} * H_t + r_i * \theta_{it} * H_t \quad \forall i, t \tag{33}$$

$$S_{it} \geq d_{it} \quad \forall i, t \tag{34}$$

Since the detailed sequencing is ignored in the upper level, the exact changeover times and costs cannot be computed. However, ignoring changeover times and costs would yield very optimistic production targets and weak upper bounds. Therefore, we introduce binary variables Y_{it} to represent the potential production of product i in time period t so that we can account for lower bounds for the changeover times and costs within each week. The idea is to assign a minimum changeover time for each product assigned to period t. Since the changeover times across the periods are neglected, the total number of changeovers assigned will be set equal to the total number of products minus one. Therefore, the maximum of the minimum transition times is subtracted in constraint (28) and in the objective function (24). Specifically, in constraints (27a) and (27b), we define the maximum of the minimum changeover times of products assigned to period t, U_t. According to constraint (28), minimum changeover times TR_i are assigned for each product that is being produced in time t, and the maximum of the minimum changeover times, U_t is subtracted. As an example in Figure 6, products A, B and C are assigned to period t, and products A and C are assigned to period t+1 where $TR_B \geq TR_A \geq TR_C$. The maximum of the minimum transition times is TR_B in period t and TR_A in period t+1, therefore the changeover time assigned for product B is subtracted from period t, and the changeover time assigned for product A is subtracted from period t+1. The final assignments are shown in Figure 6c, which ensure a valid lower bound on the transitions throughout all the periods.

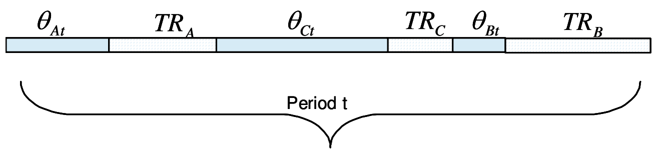

Figure 6a. Assignment of Products and changeover times to period t

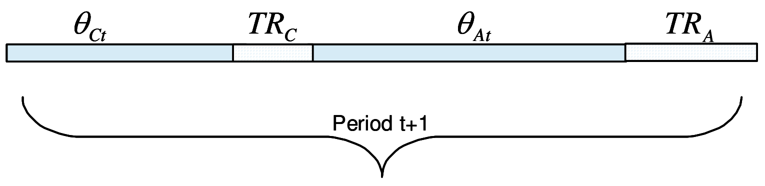

Figure 6b: Assignment of Products and changeover times to period t+1

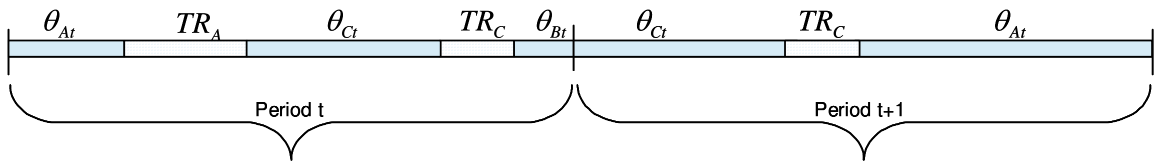

Figure 6c: Assignments with final changeover times

Constraints (29a) and (29b) are developed in analogy to constraints (27a) and (27b), and the term $\sum_t \left((\sum_i TRC_i * Y_{it}) - UTRC_t \right)$, which is an underestimation of the changeover costs, is subtracted from the objective function.

The upper level MILP given by (24)-(34) is formulated by neglecting the sequencing constraints, neglecting the transition times across the weeks and underestimating the transition times within each week. Thus, the upper level model, which is based on the MILP in (PA), is a relaxation of problem (P) and it yields an upper bound on the solution of this problem.

Property 1. *Problem (PA) yields an upper bound to the solution of Problem (P).*

We prove that (PA) is a relaxation of (P), thus yielding an upper bound.

i) Equations (2), (3a), (5), (7), (8), (9), (10), (19), (20), (21), (22), and (23) of the MILP Problem (P) are neglected.

ii) Equations (4a), (4b) and (3b) are aggregated to obtain (25) as shown below:

Substitute equation (4a) into (4b) to obtain: $x_{it} = \sum_{\ell} r_i * \overline{\theta}_{ilt}$. We can take out r_i since it is independent of the summation to obtain, $x_{it} = r_i * \sum_{\ell} \overline{\theta}_{ilt}$ and by substitution of equation (3b), this yields, $\theta_{it} = \sum_{\ell} \tilde{\theta}_{ilt}$. Equation (25) is obtained as: $x_{it} = r_i * \theta_{it}$.

iii) Equation (28) is obtained by relaxing (6) in Problem (P):

Sum equation (6) over ℓ to get: $\sum_{\ell} Te_{\ell t} = \sum_{\ell} Ts_{\ell t} + \sum_{\ell}\sum_{i} \tilde{\theta}_{i\ell t} + \sum_{\ell}\sum_{i}\sum_{k} \tau_{ik} Z_{ik\ell t}$

Note that $\sum_{\ell} Te_{\ell t} - \sum_{\ell} Ts_{\ell t} = H_t$. Substituting in this relation and (3b) to the above summation we get, $\sum_{i} \theta_{it} + \sum_{\ell}\sum_{i}\sum_{k} \tau_{ik} Z_{ik\ell t} = H_t$. We obtain a relaxation of this equation by underestimating the exact transition terms $\sum_{\ell}\sum_{i}\sum_{k} \tau_{ik} Z_{ik\ell t}$, by $TR_i \cdot y_{it} - U$. Hence this yields (28) which is a relaxation of (6).

iv) Equation (6) is relaxed to obtain equation (26):

We start from (6) which was rewritten to get $\sum_{i} \theta_{it} + \sum_{\ell}\sum_{i}\sum_{k} \tau_{ik} Z_{ik\ell t} = H_t$ in (iii). Since the transition term $\sum_{\ell}\sum_{i}\sum_{k} \tau_{ik} Z_{ik\ell t}$ is positive, $\sum_{i} \theta_{it} \le H_t$ is a valid relaxation of (6).

$\theta_{it} \le H_t$ is also a relaxation of $\sum_{i} \theta_{it} \le H_t$ since the minimum upper bound that $\theta_{it} \le H_t$ yields, is larger than the minimum upper bound obtained from $\sum_{i} \theta_{it} \le H_t$.

Note that the minimum upper bound that $\theta_{it} \le H_t$ yields, is obtained by summing $\theta_{it} \le H_t$ over i to get: $\sum_{i} \theta_{it} \le \sum_{i} H_t$ which is equivalent to $\sum_{i} \theta_{it} \le H_t * |N|$.

To summarize, $\sum_{i} \theta_{it} \le H_t$ is a relaxation of equation (6), and $\theta_{it} \le H_t$ is a relaxation of $\sum_{i} \theta_{it} \le H_t$. Therefore, $\theta_{it} \le H_t$ is a relaxation of (6).

v) Equations (30) and (31) are obtained by substituting equation (3b) to (11).

vi) Equations (32)-(33) and (34) are the same as equations (12)-(13) and (15) of Problem (P). □

Lower Level Problem

The lower level is represented by the detailed MILP planning and scheduling model (P), which is solved for only a subset of products predicted at the upper level at each time period. The main motivation for this procedure is that the number of binary variables, and hence the size of the lower level is reduced by excluding the products that were not selected by the upper level problem through the binary variable Y_{it}^{r} obtained at iteration r. Thus we impose the inequality,

$$YOP_{it} \leq Y_{it}^{r} \tag{35}$$

Constraint (35) is used at the lower level, and implies that if product i is not produced at time t at the upper level, (i.e. $Y_{it} = 0$), then that product will not be produced in time t at the lower level.

To exclude the previously obtained feasible configurations, and to ensure different solutions at each iteration, the following integer cut is added to problem (P) (Balas and Jeroslow, 1972),

$$\sum_{(i,t)\in M_r} YOP^{r}{}_{it} - \sum_{(i,t)\in N_r} YOP^{r}{}_{it} \leq |M_r| - 1 \tag{36}$$

where,

$$M_r = \{ i,t \mid YOP_{it}^{r} = 1 \, for\ configuration\ in\ \ iteration\ r \}$$
$$N_r = \{ i,t \mid YOP_{it}^{r} = 0 \, for\ configuration\ in\ \ iteration\ r \}$$

and YOP_{it}^{r} is the value of the binary variable YOP_{it} at iteration r.

Hence, the lower level problem (P^{r}) is defined by adding to problem (P) the inequalities in (35) and (36). Note that the model (P^{r}) yields a valid lower bound since its solution is a feasible solution of the original detailed planning and scheduling model. The upper level (PA) and the lower level (P^{r}) problems are solved iteratively until the bounds of each level converge within a specified tolerance.

Integer and Logic cuts

After each iteration, if the upper bound obtained from the upper level and the lower bound obtained from the lower level do not lie within a tolerance, we need to obtain a new solution from the upper level. Integer and logic cuts are used to generate new solutions in terms of the assignment variable Y_{it}.

The integer cut that excludes the previously obtained feasible solutions from the upper level model is as follows,

$$\sum_{(i,t)\in Z_1^r} Y^{r}{}_{it} - \sum_{(i,t)\in Z_0^r} Y^{r}{}_{it} \leq |Z^{r}{}_{1}| - 1 \tag{37}$$

where $Z_0{}^r = \{i,t \mid Y_{it}=0\}$ and $Z_1{}^r = \{i,t \mid Y_{it}=1\}$

Note that, $Z_0{}^r$ and $Z_1{}^r$ are obtained from the optimal solution at the upper level in terms of the assignment variable in iteration r.

In order to reduce the search space for the binary variables and the number of iterations in the decomposition procedure, logic cuts are used, which correspond to subset cuts, superset cuts and a special capacity cut that are used to eliminate suboptimal alternatives. The cuts presented below are motivated by the cuts proposed by Iyer and Grossmann (1998). The following property establishes the basis for the derivation of the subset cuts.

Property 2: A*ny subset of products in each week from the optimal solution of the upper level that was obtained at iteration r can be excluded from the upper level at any iteration* $s > r$.

Proof: During the solution at the lower level at iteration r, the optimal solution obtained at upper level, $Z_1{}^r$, and all possible subsets of $Z_1{}^r$ will be considered as alternatives due to the inequality in (35). Some of these subsets will correspond to infeasible solutions, while the others will correspond to feasible solutions. The best configuration amongst the feasible configurations obtained during the search, will be selected as the optimal solution of P^r for that specific iteration r. Since all subsets have already been considered as alternatives in the optimization of P^r, we do not need to further consider these subsets in later iterations $s > r$ at the upper level. Hence we can exclude the subsets from further consideration at the upper level problem PA^r, $s > r$. □

The cut for excluding the subsets imposes the condition that products i in time t that were not selected, $(i,t) \in Z_0{}^r = \{i,t \mid Y_{it}=0\}$, implies that selected products i in time t, $(i,t) \in Z_1{}^r = \{i,t \mid Y_{it}=1\}$ must remain. That is,

$$\left(\bigwedge_{(i,t)\in Z_o^r} \neg Y_{it} \right) \Rightarrow Y_{it} \quad \forall (i,t) \in Z_1{}^r \qquad (38)$$

which is logically equivalent to,

$$(\bigvee_{(i,t)\in Z_0^r} (Y_{it})) \vee Y_{it} \qquad (39)$$

Mathematically this proposition can be written as (Williams, 1985),

$$\sum_{(i,t)\in Z_o^r} Y_{it} + Y_{i't'} \geq 1 \quad \forall (i',t') \in Z_1^r \qquad (40)$$

In the following property, the derivation of the superset cuts is presented, which are used to exclude the supersets of previously obtained feasible configurations.

Property 3: *Given that* $Z_1{}^r = \{i,t \mid Y_{it} = 1\}$ *and* $Z_0{}^r = \{i,t \mid Y_{it} = 0\}$ *are the optimal solution of the upper level at iteration r, any superset of* $Z_1{}^r$ *at a later iteration* $s > r$*, will result in a solution of* the upper level *such that* $z_s^* \leq z_r^*$. *Hence, supersets of optimal configurations obtained at iteration r can be excluded from the solution of the upper level at iterations* $s > r$.

Proof: Let Z_1^r be the optimal solution of the upper level in iteration *r*. If any solution $Z_1^s \supset Z_1^r$ is not selected before iteration r, then this implies that the selection of any additional product $i \in Z_1^s$ and $i \notin Z_1^r$ does not result in an increase in the objective value of the upper level model. Therefore, $z_s^* \leq z_r^*$ and any superset of the optimal configuration, Z_1^r, can be excluded from further consideration at the upper level for all iterations $s > r$. □

The cut for excluding the supersets is logically written as,

$$\bigwedge_{(i,t)\in Z_1^r} Y_{it} \Rightarrow \bigwedge_{(i,t)\in Z_o^r} (\neg Y_{it}) \qquad (41)$$

The above proposition states that, if in any solution of an iteration $s > r$ all the Y_{it}'s in set Z_1^r are 1 (true), then all the other Y$_{it}$'s must be zero (false) to prevent a superset of $Z_1{}^r$. Following a similar reasoning as in (38) – (40), this can be written as follows,

$$\sum_{(i,t)\in Z_1^r} Y_{it} + Y_{i't'} \leq \left|Z_1^r\right| \quad \forall (i't') \in Z_o^r \qquad (42)$$

where $Z_0^r = \{i,t \mid Y_{it} = 0\}$ and $Z_1^r = \{i,t \mid Y_{it} = 1\}$

In the following property, the derivation of the capacity logic cut that helps to tighten the upper bound is given. The motivation for this cut is that it helps to tighten the upper bound, thereby reducing the difference between the upper and the lower bounds. This cut is different than the subset cuts and the superset cuts in the sense that it makes use of the information on the lower level solution, whereas the subset and superset cuts make use of information obtained from the upper level solution.

The cut for Property 4 is mathematically written as follows,

$$Htot \geq \sum_i \sum_t \theta_{it} + Trans^r * \left[\sum_{(i,t)\in M_r} Y_{it} + Y_{i't'} - |M_r| \right] \quad \forall (i't') \in N_r \tag{43}$$

where

$M_r = \{i,t \mid YOP_{it}^r = 1 \; \textit{for configuration in iteration } r\}$

$N_r = \{i,t \mid YOP_{it}^r = 0 \; \textit{for configuration in iteration } r\}$

and $Trans^r$ stands for the total transition time obtained at the optimal solution of (P^r) at iteration r.

Property 4: *If at the upper level, in iteration $s > r$, a solution is obtained, which is the superset of the optimal solution at the lower level at iteration r, then the total transition time of the upper level at iteration s, has to be at least equal to the total transition time obtained at the lower level at iteration r.*

Proof: When a superset of the optimal configuration of (P^r) is selected at the upper level, the multiplying factor for the right hand side becomes one. Therefore, the inequality $Htot \geq \sum_i \sum_t \theta_{it} + Trans^r$ is forced through equation (43) resulting in a decrease in the actual production times. This, however, implies a decrease in the production amounts and hence a decrease in the objective value of the upper level thereby tightening the difference between the upper and lower bounds. Furthermore, for any other choice of configuration M_s where $M_r \not\subset M_s$, the multiplication factor becomes less than zero thereby rendering the inequality (43) redundant. □

The implication of the above properties is that integer cuts and logic cuts developed at iteration r may be added to the upper level problem (PA^s) at each iteration $s > r$ .In this way, all supersets and subsets of Z_1^r and the solution Z_1^r are excluded from the solution of iteration $s > r$, thereby eliminating a large number of feasible configurations from the solution of the upper level and expediting the search. Note that these cuts are added cumulatively at each iteration resulting in an increase of the size of the problem.

The final decomposition algorithm is illustrated schematically in Figure 7.

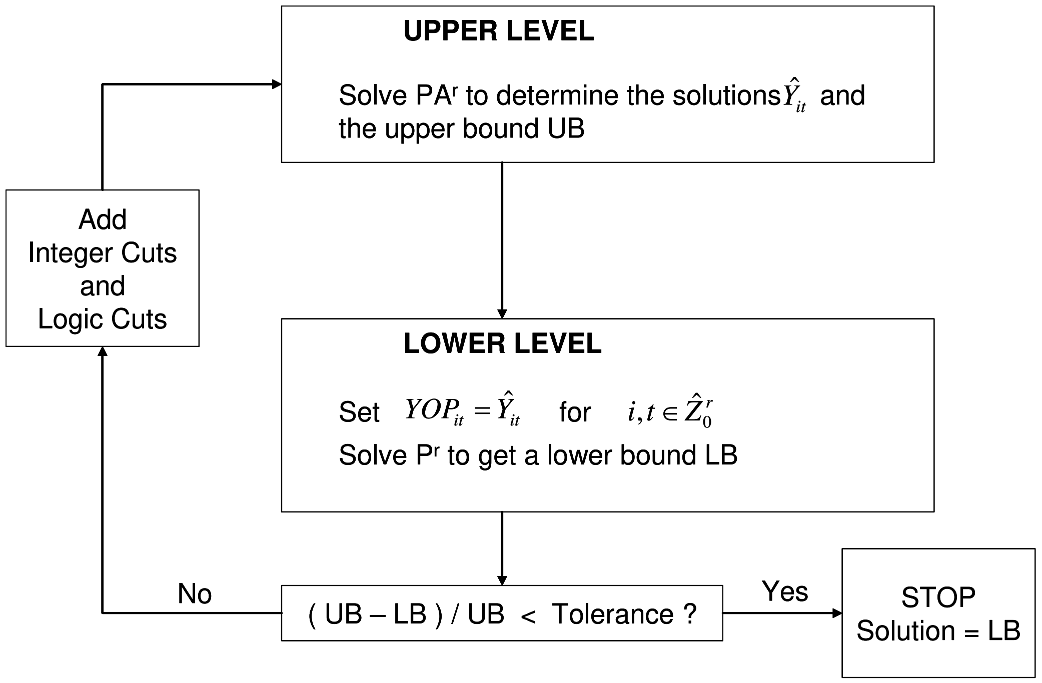
UPPER LEVEL
Solve PA^r to determine the solutions $\hat{Y}_{it}$ and the upper bound UB
LOWER LEVEL
Set $YOP_{it} = \hat{Y}_{it}$ for $i,t \in \hat{Z}_0^r$
Solve P^r to get a lower bound LB
(UB – LB) / UB < Tolerance ?
No
Yes
STOP
Solution = LB
Add Integer Cuts and Logic Cuts

Figure 7. Flowchart for the Final Decomposition Algorithm

Algorithmic Steps

The detailed steps of the proposed decomposition are as follows:

(1) Set iteration count $r=0$, upper bound $UB=\infty$, lower bound $LB=-\infty$, and optimality tolerance ε.

(2) Set $r=:r+1$.

(a) Solve the MILP aggregate model (PA^r), given by equations (24)-(34), to determine a configuration $\hat{Y}_{it}^r$ with upper bound UB.

Define

$$\hat{Z}_1^r = \{i,t \mid \hat{Y}_{it}^r = 1\}$$
$$\hat{Z}_0^r = \{i,t \mid \hat{Y}_{it}^r = 0\}$$

(3) Set

$$YOP_{it} = \hat{Y}_{it}^r \quad (i,t) \in \hat{Z}_0^r$$

Solve (P^r), given by equations (1)-(13), (15), (19)-(23), (35), to determine the products produced in each time period, YOP_{it}^r ,and a lower bound LB^r . Set $LB = \max_r \{LB^r\}$.

Define

$$M_r = \{i,t \mid YOP_{it}^r = 1\}$$
$$N_r = \{i,t \mid YOP_{it}^r = 0\}$$

(4) If $(UB - LB)/UB \leq (\varepsilon)$, stop. The solution corresponding to the LB is the optimal solution. Else, go to step 5.

(5) Add the following integer and logic cuts to (PA^r).

$$\sum_{(i,t)\in \hat{Z}_1^r} \hat{Y}_{it}^r - \sum_{(i,t)\in \hat{Z}_0^r} \hat{Y}_{it}^r \leq \left|\hat{Z}_1^r\right| - 1$$

$$\sum_{(i,t)\in \hat{Z}_0^r} \hat{Y}_{it}^r + \hat{Y}_{i't'} \geq 1 \quad \forall (i't') \in \hat{Z}_1^r$$

$$\sum_{(i,t)\in \hat{Z}_1^r} \hat{Y}_{it}^r + \hat{Y}_{i't'} \leq \hat{Z}_1^r \quad \forall (i't') \in \hat{Z}_0^r$$

$$Htot \geq \sum_i \sum_t \theta_{it} + Trans^r * \left[\sum_{(i,t)\in M_r} \hat{Y}_{it}^r + \hat{Y}_{i't'}^r - \left|M_r\right| \right] \quad \forall (i't') \in N_r$$

Go to step 2.

Remarks

1. The proposed decomposition algorithm produces the same global optimal solution as the full-space MILP model in a finite number of iterations since the number of possible assignments of products in each week is finite.
2. A major advantage of the proposed algorithm is that the size of the lower level model, which is the bottleneck for this multiperiod optimization problem, is reduced by considering a subset of assignment variables as obtained from the upper level, making it possible to fix a potentially large number of binary variables.
3. The integer and logic cuts in equations (36), (37), (40), (42) and (43) are added cumulatively at each iteration to the upper level models (PA), and may result in the increase in the size of the upper level models.
4. In order to reduce the number of iterations when there is slow convergence between the lower and upper bounds, a nonzero optimality tolerance ε can be specified in step 1 (e.g. 1% to 10%). Also, we should note that even though in larger problems it might be difficult to close the gap between the bounds, the proposed procedure will

be able to generate good feasible solutions whose global optimality can be guaranteed within the difference of the bounds.

5. For instances when either the upper level or the lower level models are expensive to solve, these can be solved approximately with Forward Rolling Horizon Algorithms (Dimitriadis et al, 1997) or with Temporal Lagrangean Decomposition (Guinard and Kim, 1987; Jackson and Grossmann, 2003).

Examples

To illustrate the application and computational effectiveness of the proposed algorithm, three examples are presented. The solutions obtained from the proposed method are compared to the full space solutions. The methods were implemented in GAMS (Brooke at al., 2002) and solved with the MILP solver CPLEX 9.0 (ILOG 2004) on an Intel 3.2 GHz machine. For simplicity, zero levels of initial inventories are assumed in all examples.

Example 1

This planning and scheduling problem consist of five different products, A-E, to be processed on the continuously operating single unit. The planning horizon consists of 4 weeks and due dates for orders are specified at the end of each week. The production rates are shown in Table 1, while the cost data are presented in Tables 2 and 3. This example was solved for two sets of lower bounds for the demands (high and low demand rates), which are shown in Tables 4a and 4b. The case of lower demands should be easier to solve because one product will be overproduced due to the continuous operation, which in turn has the effect of producing fewer products in each week.

Table 1. Production rates data for Example 1

Product	Production Rates(kg/hr)
A	800
B	900
C	1,000
D	1,000
E	1,200

Table 2. Cost Data for each time period for Example 1

	Operating Costs ($/kg)	Selling Price ($/kg)
A	0.19	0.25
B	0.32	0.40
C	0.55	0.65
D	0.49	0.55
E	0.38	0.45

Inventory Cost ($/kg.h)
0.0000306

Table 3. Transition times and transition costs for Example 1

	Product				
Product	**A**	**B**	**C**	**D**	**E**
			Transition times (hrs)		
A	0.00	2.00	1.50	1.00	0.75
B	1.00	0.00	2.00	0.75	0.50
C	1.00	1.25	0.00	1.50	2.00
D	0.50	1.00	2.00	0.00	1.75
E	0.70	1.75	2.00	1.50	0.00
			Transition costs ($)		
A	0	760	760	750	760
B	745	0	750	770	740
C	770	760	0	765	765
D	740	740	745	0	750
E	740	740	750	750	0

Table 4. Lower Bounds for Demands

a) High Demand Rates for Example 1a

	Time Period			
	1	**2**	**3**	**4**
		Demand (kg)		
A	10,000	20,000	30,000	10,000
B	25,000	20,000	15,000	25,000
C	30,000	40,000	50,000	30,000
D	30,000	20,000	13,000	30,000
E	30,000	20,000	12,000	30,000

b) Low Demand Rates for Example 1b

	Time Period			
	1	2	3	4
	Demand (kg)			
A	0	10,000	20,000	0
B	15,000	10,000	5,000	15,000
C	20,000	30,000	40,000	20,000
D	20,000	10,000	3,000	20,000
E	20,000	10,000	2,000	20,000

Table 5 shows the problem sizes and solution times for the proposed decomposition algorithm and the full space methods for Example 1a which corresponds to high demand rates. Table 6 shows the progress of the iterations for a 0% optimality tolerance. It should be noted that in Table 5, the sizes of the upper and lower level problems are from the last iteration in Table 6, while the corresponding times are the total times. Although the sizes of the lower level problem are similar to the original problem, it includes the constraints in (35), which effectively fixes many of the 0-1 variables.

The solution obtained using the proposed algorithm yields a profit of $43,120. The solution for the proposed approach was obtained in 15 major iterations in 207 CPU seconds. We should note that if the proposed method is solved with 1% tolerance, the number of iterations decreases to 4 and the CPU time decreases to 29 seconds. The full space method was solved with a 1% optimality tolerance and failed to terminate in 6000 CPUs yielding a feasible solution of $43,015 with an 8% gap between the bounds. Figure 8 shows the schedule that is predicted by the proposed method and Figure 9 shows the inventory levels.

Table 5. Results for Example 1a

Method	Number of binary variables	Number of continuous variables	Number of Equations	Time (CPUs)	Solution ($)
Full Space	120	987	906	6000*	43,015.9
Proposed algorithm				207.9	43,120.8
Problem UB	20	151	564	2.0	43,013.0
Problem LB	120	996	949	205.9	43,120.8

*Search not terminated, best feasible solution posted with 8% gap of bounds

Table 6. Progress of iterations for Example 1a

Iteration	Upper Bound (UB)	Lower Bound (LB)
1	43,659.5	42,351.7
2	43,623.5	42,229.6
3	43,543.7	42,994.0
4	43,540.5	43,120.776*
5	43,530.1	42,307.7
6	43,530.1	42,219.9
7	43,489.2	42,998.2
8	43,467.9	42,980.8
9	43,425.6	42,980.8
10	43,403.3	43,009.4
11	43,394.6	42,818.9
12	43,376.7	42,818.9
13	43,354.2	42,995.1
14	43,353.8	42,721.5
15	43,013.0	41,637.7

* Optimal solution

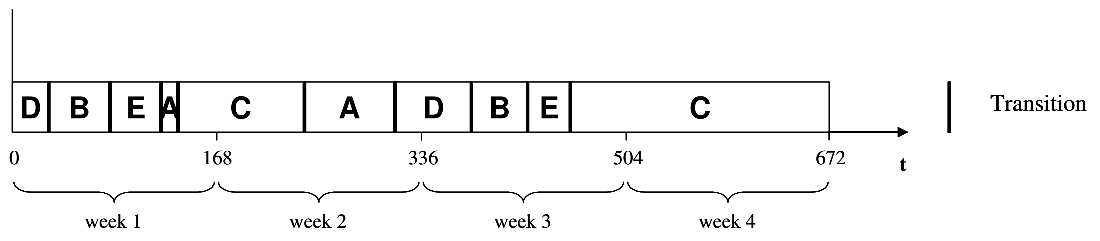

Figure 8. Gantt chart for Example 1a

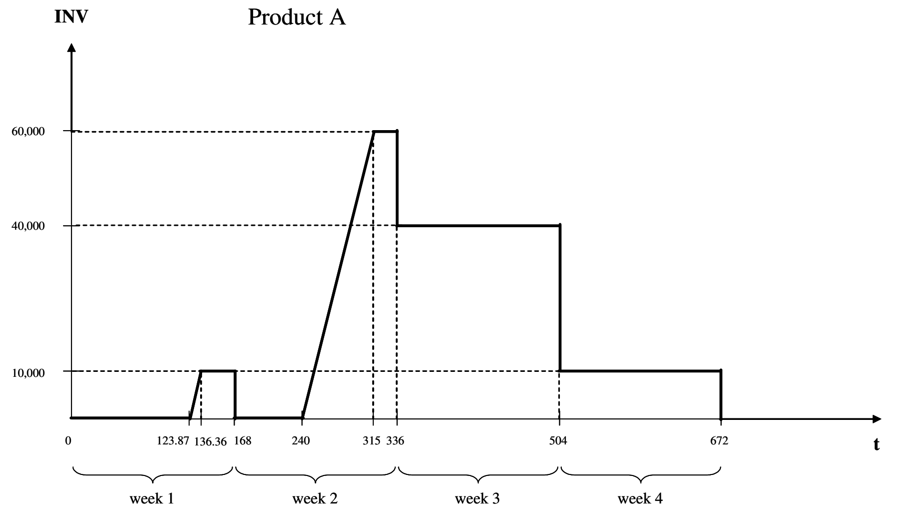

Figure 9a. Inventory Levels for Product A for Example 1a

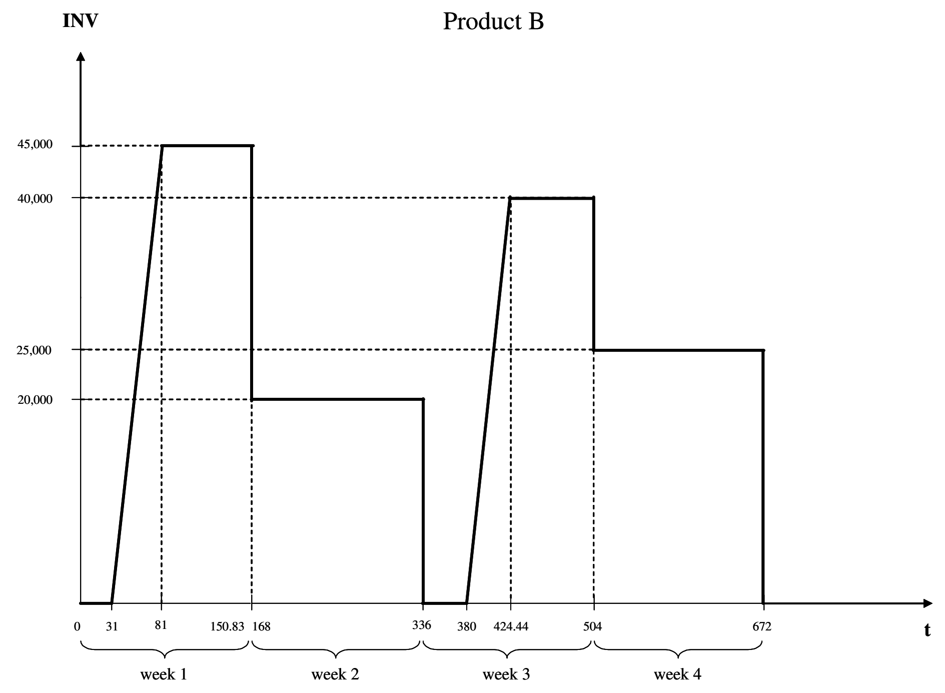

Figure 9b. Inventory Levels for Product B for Example 1a

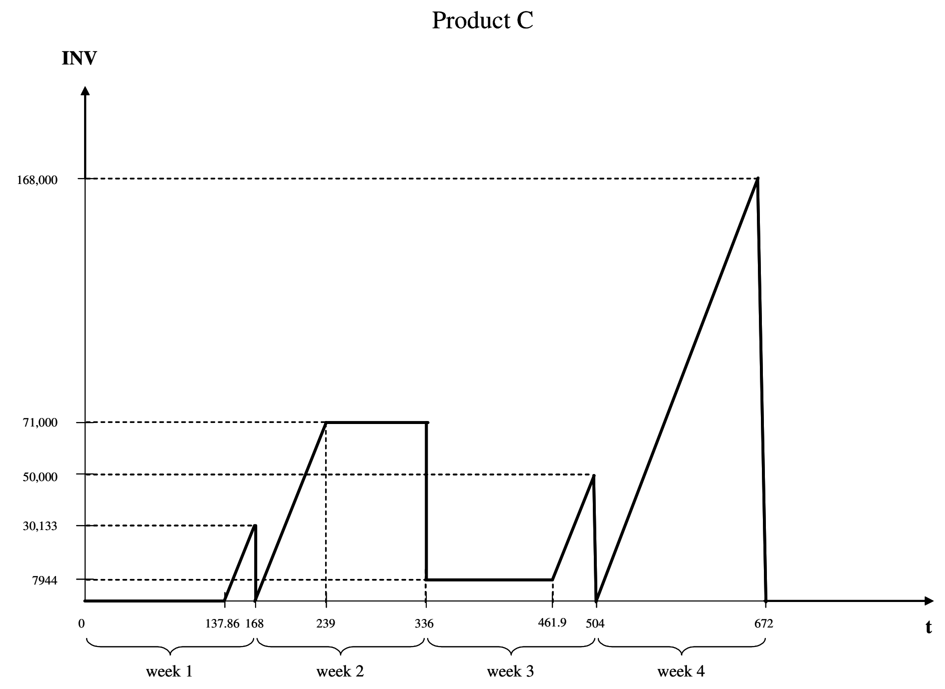

Figure 9c. Inventory Levels for Product C for Example 1a

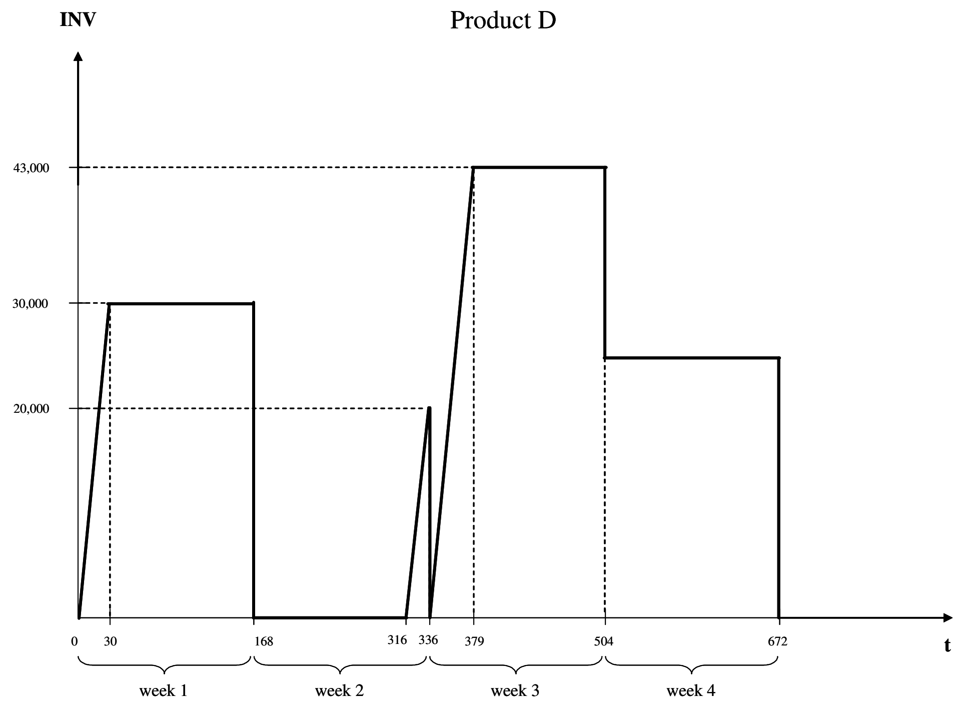

Figure 9d. Inventory Levels for Product D for Example 1a

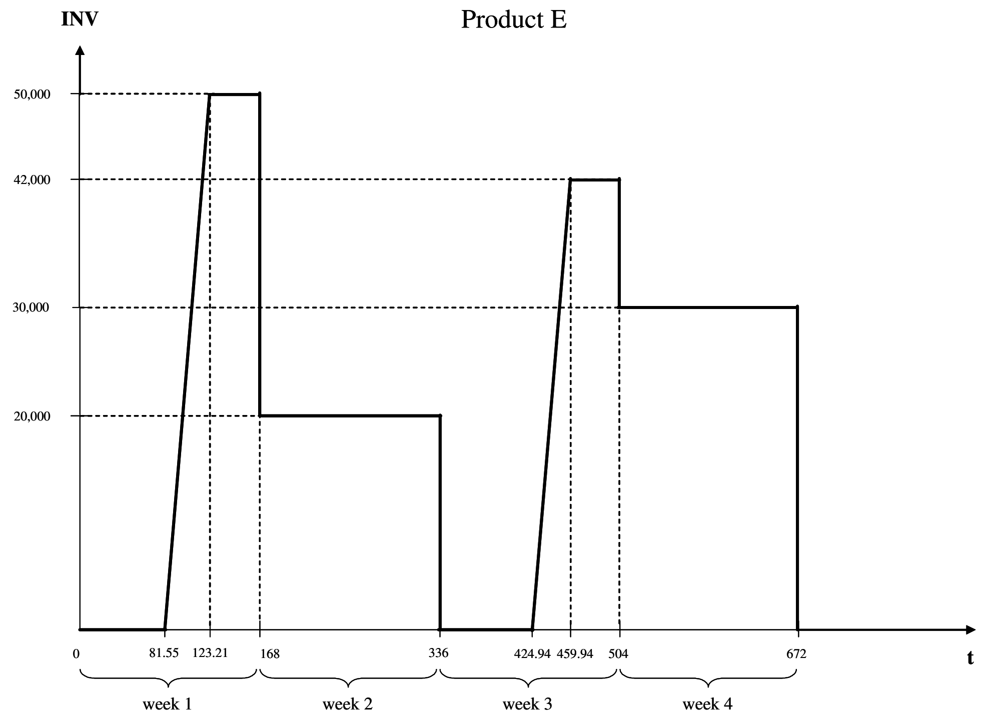

Figure 9e. Inventory Levels for Product E for Example 1a

Recall that in order to avoid nonlinearities in the objective function due to the inventory costs, an overestimation of the inventory costs was developed (eqtn 13). If we had calculated the exact inventory cost for Example 1a, it would have been $1,679 less than the overestimated inventory cost. While the overestimation of the inventory is rather high (about 40%), its impact in the total profit was much smaller (4% overestimation). As we will seen below, the the overestimation is smaller in the case of low demands.

Table 7 shows the problem sizes and solution times obtained for the proposed decomposition algorithm and the full space method for Example 1b, which has the low demands (Table 4b). The proposed algorithm yields a profit of $52,319 in 4.6 CPUs in 10 major iterations using a 0% tolerance. The full space method yields the same solution of $52,319 in 526 CPUs. The progress of the iterations with the proposed method is presented in Table 8. We should note that the reason the profit is higher in Example 1b is because more of Product C is produced than in Example 1a (see Figure 10) due to the smaller lower bounds for the demands. It should also be noted that in this case, if we had calculated the exact inventory costs, they would have been $840 less than the overestimated inventory costs. In this case the inventory costs are overestimated by 18%, while the total profit is overestimated by 1.6%.

The optimal schedule predicted by the proposed approach for Example 1b is shown in Figure 10a. As seen in Table 8, the difference between the solutions obtained at each iteration is very small. However, this is not true for every solution. To give an example, the profit for the heuristic schedule in Figure 10b that involves lower inventories but larger number of transitions is $48,754, which is significantly lower than the solutions in Table 7 (6.8%).

Table 7. Results for Example 1b

Method	Number of binary variables	Number of continuous variables	Number of Equations	Time (CPUs)	Solution ($)
Full Space	120	987	906	525.9	52319.9*
Proposed algorithm				4.6	52,319.9
Problem UB	20	151	455	0.5	51,988.1
Problem LB	120	996	944	4.1	52,319.9

*Optimal solution for the full space method with a 0% gap of bounds

Table 8. Progress of iterations for Example 1b

Iteration	Upper Bound (UB)	Lower Bound (LB)
1	52,712.3	51,556.6
2	52,693.3	51,509.1
3	52,640.3	51,423.7
4	52,635.2	51,508.2
5	52,611.1	51,422.1
6	52,609.5	51,283.0
7	52,573.5	52,315.303*
8	52,558.1	52,123.9
9	52,549.4	52,319.9
10	51,988.1	50,690.4

* Optimal solution

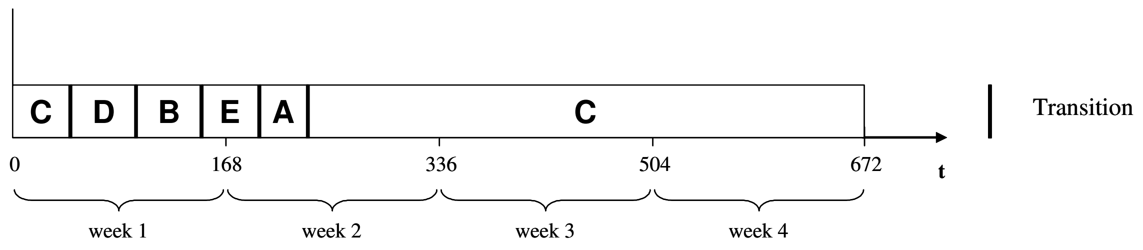

Figure 10a. Gantt chart for Example 1b

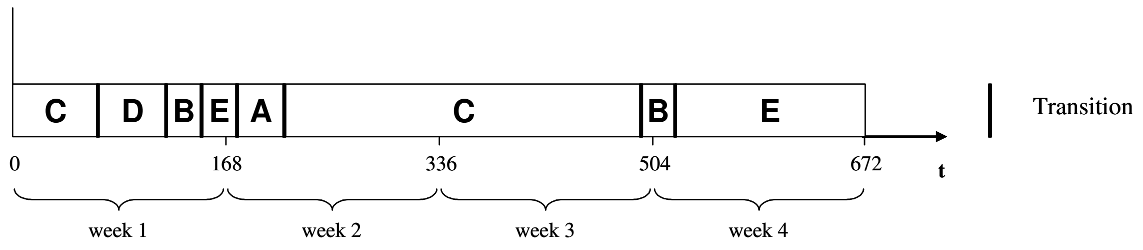

Figure 10b. Gantt chart for Heuristic Example

Example 2

This problem consists of the same five different products, A-E, to be processed over a 8 week horizon in which due dates are specified at the end of each week. The production rates data and the cost data are the same as for the first example. The lower bounds for the demand data for high rates are presented in Table 9a while the ones for low rates are presented in Table 9b.

Table 9. Lower Bounds for Demands for Example 2

a) High Demand Rates (Example 2a)

	Time Period							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
				Demand (kg)				
A	10,000	20,000	30,000	10,000	20,000	30,000	10,000	20,000
B	25,000	20,000	15,000	25,000	20,000	15,000	25,000	20,000
C	30,000	40,000	50,000	30,000	40,000	50,000	30,000	40,000
D	30,000	20,000	13,000	30,000	20,000	13,000	30,000	20,000
E	30,000	20,000	12,000	30,000	20,000	12,000	30,000	20,000

b) Low Demand Rates (Example 2b)

	Time Period							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
				Demand (kg)				
A	0	10,000	20,000	0	10,000	20,000	0	10,000
B	15,000	10,000	5,000	15,000	10,000	5,000	15,000	10,000
C	20,000	30,000	40,000	20,000	30,000	40,000	20,000	30,000
D	20,000	10,000	3,000	20,000	10,000	3,000	20,000	10,000
E	20,000	10,000	2,000	20,000	10,000	2,000	20,000	10,000

Table 10. Results for Example 2a

Method	Number of binary variables	Number of continuous variables	Number of Equations	Time (CPUs)	Solution ($)
Full Space	240	1,971	1,830	4000*	87,299.9
Proposed algorithm				1,041.8	88,100.1
Problem UB	40	299	703	105.4	90,650.8
Problem LB	240	1,984	1,889	936.5	88,100.1

* Search not terminated, best feasible solution posted with 13% gap of bounds

Table 10 shows the problem sizes and solution times for the proposed decomposition algorithm and the full space methods, which were solved with a 3% optimality tolerance. Table 11 shows the progress of the iterations with the proposed method for Example 2a (high demand rates), and Figure 11 shows the optimal schedule that is predicted by the proposed method for Example 2a.

The proposed algorithm yields a profit of $88,100 within 3% of the global optimum. The solution for the proposed approach was obtained in 7 major iterations in 1041 CPU seconds, whereas the full space method fails to terminate within the specified limit (4000 CPUs), only managing to produce a feasible solution with an objective value of $87,299.

Table 11. Progress of iterations for Example 2a

Iteration	Upper Bound (UB)	Lower Bound (LB)
1	90,694.90	87,360.31
2	90,687.02	86,469.02
3	90,676.69	86,469.02
4	90,671.93	86,312.44
5	90,669.19	86,450.69
6	90,665.3	87,528.22
7	90,650.80	88,100.074*

*Optimal solution

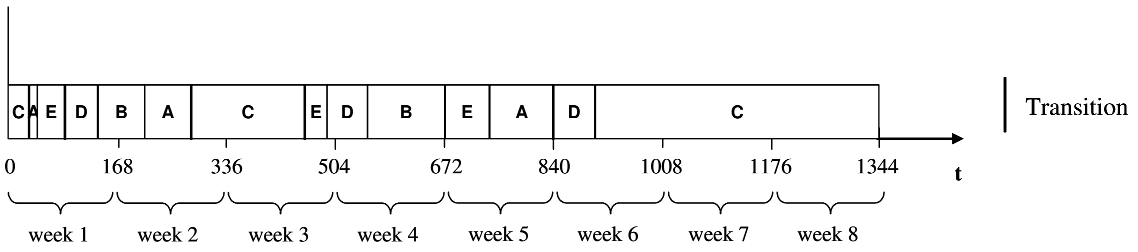

Figure 11. Gantt chart for Example 2a

Table 12. Results for Example 2b

Method	Number of binary variables	Number of continuous variables	Number of Equations	Time (CPUs)	Solution ($)
Full Space	240	1,971	1,830	3000*	104,671.5
Proposed algorithm				1.7	104,702.3
Problem UB	40	299	331	0.2	107,355.7
Problem LB	240	1,984	1,883	1.5	104,702.3

*Search not terminated, best feasible solution posted with 7% gap of bounds

Table 12 presents the results obtained for Example 2b, which corresponds to low demand rates. The proposed method yields a profit of $104,702 which lies within 3% of the global optimum in 2 CPUs in one major iteration. The full space method failed to terminate in 3000 CPUs and yielded a feasible solution of $104,671 with a 7% optimality gap. The proposed approach is at least two orders of magnitude faster than the full space method for the low demand rates.

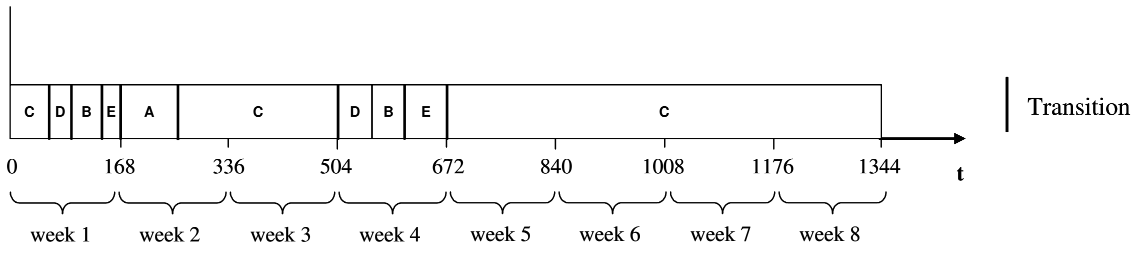

Figure 12. Gantt chart for Example 2b

It should be noted that decreasing the tolerance results in more accurate values of the objective function, but at the cost of increased CPU times. The effect of the optimality tolerance on the results of Example 2b is shown in Table 13.

Table 13. Effect of Tolerance on Results for Example 2b

Tolerance	Number of iterations	CPU(s)	Upper Bound ($)	Lower Bound ($)	Simultaneous CPU time	Simultaneous solution
1%	80	2591	106,234.70	104,957.12	3,000	104,671.46
2%	2	3.6	106,997.24	104,957.12	3,000	104,671.46
3%	1	1.7	107,355.70	104,702.30	3,000	104,671.46
5%	1	1.7	107,355.70	104,338.10	3,000	104,671.46

Example 3

In this example a planning horizon of 16 weeks is considered. The cost data and the production rates for products A-E are the same as for the first example. The lower bounds for high demand data are given in Table 14a and the lower bounds for the low demand data are presented in Table 14b.

The proposed algorithm yields a profit of $184,144 within 6% of the global optimum in 93 CPUs in three major iterations. We should note that, when the tolerance is decreased to 5%, a solution of $183,041 is obtained in 23,000 CPUs and 35 major iterations. The full space method produced a solution of $183,161 in 3,000 CPUs. The results for Example 3a are shown in Table 15, while the progress of iterations are shown in Table 16.

Table 14. Lower Bounds for Demands for Example 3

a) High Demand Rates (Example 3a)

Product	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	Demand (kg)															
A	10,000	20,000	30,000	10,000	20,000	30,000	10,000	20,000	30,000	10,000	20,000	30,000	10,000	20,000	30,000	10,000
B	25,000	20,000	15,000	25,000	20,000	15,000	25,000	20,000	15,000	25,000	20,000	15,000	25,000	20,000	15,000	25,000
C	30,000	40,000	50,000	30,000	40,000	50,000	30,000	40,000	50,000	30,000	40,000	50,000	30,000	40,000	50,000	30,000
D	30,000	20,000	13,000	30,000	20,000	13,000	30,000	20,000	13,000	30,000	20,000	13,000	30,000	20,000	13,000	30,000
E	30,000	20,000	12,000	30,000	20,000	12,000	30,000	20,000	12,000	30,000	20,000	12,000	30,000	20,000	12,000	30,000

b) Low Demand Rates (Example 3b)

	Time Period															
Product	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	Demand (kg)															
A	0	10,000	20,000	0	10,000	20,000	0	10,000	20,000	0	10,000	20,000	0	10,000	20,000	0
B	15,000	10,000	5,000	15,000	10,000	5,000	15,000	10,000	5,000	15,000	10,000	5,000	15,000	10,000	5,000	15,000
C	20,000	30,000	40,000	20,000	30,000	40,000	20,000	30,000	40,000	20,000	30,000	40,000	20,000	30,000	40,000	20,000
D	20,000	10,000	3,000	20,000	10,000	3,000	20,000	10,000	3,000	20,000	10,000	3,000	20,000	10,000	3,000	20,000
E	20,000	10,000	2,000	20,000	10,000	2,000	20,000	10,000	2,000	20,000	10,000	2,000	20,000	10,000	2,000	20,000

Table 15. Results for Example 3a

Method	Number of binary variables	Number of continuous variables	Number of Equations	Time (CPUs)	Solution ($)
Full Space	480	3,939	3,678	3000*	183,161.0
Proposed algorithm				92.6	184,144.7
Problem UB	80	595	918	1.6	194,220.5
Problem LB	480	3,960	3,781	90.9	184,144.7

* Search not terminated, best feasible solution posted with 17% gap of bounds

Table 16. Progress of iterations for Example 3a

Iteration	Upper Bound (UB)	Lower Bound (LB)
1	194,843.78	182,763.00
2	194,171.60	182,760.00
3	194,220.49	184,144.742*

*Optimal solution

Table 17. Results for Example 3b

Method	Number of binary variables	Number of continuous variables	Number of Equations	Time (CPUs)	Solution ($)
Full Space	480	3,678	3,939	1,000*	211,879.2
Proposed algorithm				151.0	210,879.5
Problem UB	80	595	4,488	137.5	217,803.8
Problem LB	480	3,960	3,807	13.5	210,879.5

* Search not terminated, best feasible solution posted with 10% gap of bounds

Table 18. Progress of iterations for Example 3b

Iteration	Upper Bound (UB)	Lower Bound (LB)
1	217,803.81	209,253.33
2	218,314.78	209,366.39
3	218,073.61	205,662.35
4	218,744.95	208,259.83
5	220,422.01	209,638.38
6	218,294.93	204,875.84
7	220,586.46	209,708.73
8	218,059.90	206,481.16
9	218,471.37	206,594.91
10	220,436.04	206,810.38
11	220,502.92	206,684.09
12	219,875.80	207,963.81
13	217,513.78	207,963.81
14	220,064.63	207,212.68
15	220,285.74	207,417.50
16	218,013.80	209,144.48
17	218,506.83	210,879.45*
18	219,690.71	209,908.00
19	217,821.57	208,224.97
20	219,834.70	209,120.11
21	219,516.03	208,594.87
22	217,578.13	206,246.97
23	218,367.29	209,674.75
24	218,131.54	204,806.03
25	218,198.43	203,974.34
26	219,082.18	206,392.90
27	218,666.82	208,599.21
28	218,539.90	205,120.69
29	217,308.54	208,334.30

* Optimal solution

The results for Example 3b are presented in Table 17. A profit of $210,879 within 3% of the global optimum is obtained with the proposed approach. The search is completed in 29 major iterations in 151 CPUs. Note that, the total number of constraints defining the upper level is more than the total number of equations in the lower level. This is due to the accumulation of logic and integer cuts in the upper level with each iteration. Hence, the upper level becomes the bottleneck for this example in contrast to the previous examples. In Table 18, the progress of iterations for Example 3b is presented. Note that in this case the simultaneous solution did not terminate the search within 1000 seconds, but it was able to find a solution with higher profit ($211,879.2) compared to the proposed approach ($210,879.5)

Example 4

In this example a planning horizon of 24 weeks is considered. The cost data and the production rates for products A-E are the same as for the first example. The lower bounds for the high demand case is presented in Table 19a, while the ones for low demand are presented in Table 19b.

Table 19. Lower Bounds for Demands for Example 4

a) High Demand Rates (Example 4a)

Product	1	2	3	4	5	6	7	8	9	10	11	12
								Demand (kg)				
A	10,000	20,000	30,000	10,000	20,000	30,000	10,000	20,000	30,000	10,000	20,000	30,000
B	25,000	20,000	15,000	25,000	20,000	15,000	25,000	20,000	15,000	25,000	20,000	15,000
C	30,000	40,000	50,000	30,000	40,000	50,000	30,000	40,000	50,000	30,000	40,000	50,000
D	30,000	20,000	13,000	30,000	20,000	13,000	30,000	20,000	13,000	30,000	20,000	13,000
E	30,000	20,000	12,000	30,000	20,000	12,000	30,000	20,000	12,000	30,000	20,000	12,000

Product	13	14	15	16	17	18	19	20	21	22	23	24
A	10,000	20,000	30,000	10,000	20,000	30,000	10,000	20,000	30,000	10,000	20,000	30,000
B	25,000	20,000	15,000	25,000	20,000	15,000	25,000	20,000	15,000	25,000	20,000	15,000
C	30,000	40,000	50,000	30,000	40,000	50,000	30,000	40,000	50,000	30,000	40,000	50,000
D	30,000	20,000	13,000	30,000	20,000	13,000	30,000	20,000	13,000	30,000	20,000	13,000
E	30,000	20,000	12,000	30,000	20,000	12,000	30,000	20,000	12,000	30,000	20,000	12,000

b) Low Demand Rates (Example 4b)

Product	1	2	3	4	5	6	7	8	9	10	11	12
								Demand (kg)				
A	0	10,000	20,000	0	10,000	20,000	0	10,000	20,000	0	10,000	20,000
B	15,000	10,000	5,000	15,000	10,000	5,000	15,000	10,000	5,000	15,000	10,000	5,000
C	20,000	30,000	40,000	20,000	30,000	40,000	20,000	30,000	40,000	20,000	30,000	40,000
D	20,000	10,000	3,000	20,000	10,000	3,000	20,000	10,000	3,000	20,000	10,000	3,000
E	20,000	10,000	2,000	20,000	10,000	2,000	20,000	10,000	2,000	20,000	10,000	2,000

Time Period

Product	13	14	15	16	17	18	19	20	21	22	23	24
						Demand (kg)						
A	0	10,000	20,000	0	10,000	20,000	0	10,000	20,000	0	10,000	20,000
B	15,000	10,000	5,000	15,000	10,000	5,000	15,000	10,000	5,000	15,000	10,000	5,000
C	20,000	30,000	40,000	20,000	30,000	40,000	20,000	30,000	40,000	20,000	30,000	40,000
D	20,000	10,000	3,000	20,000	10,000	3,000	20,000	10,000	3,000	20,000	10,000	3,000
E	20,000	10,000	2,000	20,000	10,000	2,000	20,000	10,000	2,000	20,000	10,000	2,000

Table 20. Results for Example 4a

Method	Number of binary variables	Number of continuous variables	Number of Equations	Time (CPUs)	Solution ($)
Full Space	720	5,907	5,526	4,000*	270,538
Proposed algorithm				3,190	272,474
Problem UB	120	891	1,181	90	286,728
Problem LB	720	5,936	5,676	3,098	272,474

* Search not terminated, best feasible solution posted with 18% gap of bounds

The results for Example 4a are presented in Table 20. A profit of $272,474 within 6% of the global optimum is obtained with the proposed approach. The search is completed in 2 major iterations in 3190 CPUs. The full space approach yielded a solution of $270,538 in 4000 CPUs. The progress of iterations for the proposed approach is shown in Table 21.

Table 21. Progress of iterations for Example 4a

Iteration	Upper Bound (UB)	Lower Bound (LB)
1	286,728	268,900
2	287,209	272474*

* Optimal solution

The results for Example 4b are presented in Table 22. An optimal solution of $313,285 is obtained within 4% of the global optimum solution. It took the proposed method 8 CPUs and 3 major iterations to solve the problem, whereas with the full space method obtained a solution of $315,207 in 3000 CPUs. The progress of iterations for the proposed method for Example 4b is shown in Table 23.

Table 22. Results for Example 4b

Method	Number of binary variables	Number of continuous variables	Number of Equations	Time (CPUs)	Solution ($)
Full Space	720	5,907	5,526	3000*	315,207
Proposed algorithm				8	313,285
Problem UB	120	891	1,396	6	325,576
Problem LB	720	5,936	5,677	2	313,285

* Search not terminated, best feasible solution posted with 11% gap

Table 23. Progress of iterations for Example 4b

Iteration	Upper Bound (UB)	Lower Bound (LB)
1	325,958	311,071
2	326,169	310,547
3	325,576	313,285*

* Optimal solution

Conclusions

An MILP model for the simultaneous planning and scheduling of continuous multiproduct plants with a single processing stage has been presented in this paper. The sequence dependent transition times, transition costs and inventory costs are readily accounted for. In order to avoid nonlinearities in the objective function that are due to the inventory costs, an overestimation was developed that can be expressed in linear form. Since the MILP model becomes very expensive to solve when a large number of products and long planning horizons are considered, a bi-level decomposition procedure has been proposed that allows rigorous integration and optimization of planning and scheduling. The main novelties in this method are the superset and the subset cuts, as well as capacity cuts that eliminate many solutions from the upper level aggregated model. The application of the algorithm was illustrated with eight examples for 5 products, ranging from 4 to 24 weeks and with high and low values for the lower bounds for the demands.

Our preliminary results show that the proposed method is significantly faster than the full-space method, although convergence with finite tolerance is required for reasonable computational times in the larger problems.

Also, the results show that, the performance of the proposed approach is more efficient when low demand rates are used. Furthermore, in this case, the overestimation of the inventory costs is found to be lower. This is due to the fact that, the model tends to produce each product in only one time period and satisfy the demands from the inventories when low demand rates are used.

Acknowledgement

The authors gratefully acknowledge financial support from the National Science Foundation under Grant ACI-0121497.

REFERENCES

1. Balas, E.; Jeroslow, R.; Canonical Cuts on the Unit Hypercube. *SIAM Journal of Applied Mathematics*, **1972**, 23, 61-79.
2. Bassett, M. H.; Dave, P.; Doyle III, F. J.; Kudva, G. K.; Pekny, J. F.; Reklaitis, G. V.; Subrahmanyam, S.; Miller, D. L., Zentner, M. G. Perspectives on Model Based Integration of Process Operations. *Computers Chem Engng*, **1996**, 20, 821-844.
3. Bassett, M. H.; Pekny, J. F.; Reklaitis, G. V. Decomposition Techniques for the Solution of Large-Scale Scheduling Problems. *AIChE Journal*, **1996**, 42, 3373-3384.
4. Birewar, D. B.; Grossmann I. E. Simultaneous Production Planning and Scheduling in Multiproduct Batch Plants. *Ind. Eng. Chem. Res.* **1990**, 29, 570-580.
5. Bodington, E. C. Planning, Scheduling and Control: Integration in the Process Industries. McGrawHill, 1995
6. Brooke, A.; D. Kendrick; A. Meeraus; R. Raman "GAMS A User's Guide" Release 2.25. The Scientific Press, Redwood City, CA 1992.
7. Dimitriadis, A.D.; Shah, N.; Pantelides, C. C. RTN-based Rolling Horizon Algorithms for MediumTerm Scheduling of Multipurpose Plants. *Computers Chem Engng,* **1997**, 21, 1061-1066.
8. Graves, S.C. Using Lagrangean Techniques to Solve Hierarchical Production Planning Problems. *Management Science*, **1982**, Vol.28, No.3, 260-275
9. Grossmann, I.E., “Enterprise-wide Optimization: A New Frontier in Process Systems Engineering,” *AIChE J.*, 2005, 51, 1846-1857.
10. Guignard, M.; Kim, S. Lagrangean Decomposition: A Model Yielding Stronger Lagrangean Bounds. *Mathematical Programming*, **1987**, 39, 215-228.

11. Iyer, R. R.; Grossmann I. E. A Bilevel Decomposition Algorithm for Long-Range Planning Of Process Networks. *Ind. Eng. Chem. Res.* **1998**, 37, 474-481.
12. Jackson, J. R.; Grossmann I. E. Temporal Decomposition Scheme for Nonlinear Multisite Production Planning and Distribution Models. *Ind. Eng. Chem. Res.* **2003**, 42, 3045-3055.
13. Kallrath, J. Planning and Scheduling in the Process Industry. *OR Spectrum*, **2002**, 24, 219-250
14. Kondilli E.; Pantelides, C. C.; Sargent, R.W.H. A General Algorithm for Short-Term Scheduling of Batch Operations-I. *Computers Chem Engng*, **1993**, 17, 211-227.
15. Raman, R; Grossmann, I. E. Relation between MILP Modeling and Logical Inference for Chemical Process Synthesis. *Computers Chem Engng*, **1993**, 17, 909-915.
16. Shah, N. Single and Multisite Planning and Scheduling: Current Status and Future Challenges. *AICHE Symposium and Monograph Series* **1998**, 94, 75-90.
17. Shapiro, J. F. Modeling the Supply Chain. **2001**, Duxbury, Thompson Learning.
18. Subrahmanyam, S.;Pekny, J. F.; Reklaitis, G. V. Decomposition Approaches to Batch Plant Design and Planning. *Ind. Eng. Chem. Res.* **1996**, 35, 1866-1876.
19. Subrahmanyam, S.; Bassett, M. H.; Pekny, J. F.; Reklaitis, G. V. Issues in Solving Large Scale Planning, Design and Scheduling Problems in Batch Chemical Plants. *Computers Chem Engng*, **1996**, 19, 577-582.
20. Wilkinson, S. J.; Shah, N.; Pantelides, C.C. Aggregate Modeling of Multipurpose Plant Operation. *Computers Chem Engng*, **1996**, 19,583,588.
21. Williams, H.P. Model Building in Mathematical Programming. **1985**, New York:Wiley-Interscience
22. Zhu, X. X.; Majozi, T. Novel Continuous Time MILP Formulation for Multipurpose Batch Plants. 2. Integrated Planning and Scheduling. *Ind. Eng. Chem. Res.* **2001**, 40, 5621-5634.