Inventory Pinch Based, Multiscale Models for Integrated Planning and Scheduling-Part I: Gasoline Blend Planning

Pedro A. Castillo Castillo and Vladimir Mahalec
Dept. of Chemical Engineering, McMaster University, Hamilton, ON, Canada L8S 4L8

DOI 10.1002/aic.14423
Published online March 7, 2014 in Wiley Online Library (wileyonlinelibrary.com)

A two-level algorithm to compute blend plans that have much smaller number of different recipes, much shorter execution times, and the same cost as the corresponding multiperiod mixed-integer nonlinear programming is introduced. These plans become a starting point for computation of approximate schedules, which minimize total number of switches in blenders and swing tanks. The algorithm uses inventory pinch points to delineate time periods where optimal blend recipes are likely constant. At the first level, nonlinear blend models are optimized via nonlinear programming. The second level uses fixed recipes (from the first level) in a multiperiod mixed-integer linear programming to determine optimal production plan followed by an approximate schedule. Approximate schedules computed by the multiperiod inventory pinch algorithm in most of the case studies are slightly better than those computed by global optimizers (ANTIGONE, GloMIQO) while requiring significantly shorter execution times. Such schedules provide constraints for subsequent detailed scheduling in Part II. © 2014 American Institute of Chemical Engineers *AIChE J*, 60: 2158–2178, 2014

Keywords: gasoline blend planning, inventory pinch, recipe optimization, minimum number of recipes, multiscale planning model

Introduction

A supply chain is a system of all the activities required to produce and distribute a product starting from raw materials to the final product delivery to the end customer. Supply chain optimization involves making decisions at different temporal, spatial, and process scales (see Figure 1). Strategic planning determines decisions such as supplier selection, plant location, production system selection, distribution structure, and sales programs. Medium-term planning defines, in a weekly or monthly basis, the production targets of each plant site, the necessary stock levels of the inventories at various locations, and how much of each product is going to be transported from each plant to each warehouse or storage depot. Short-term planning is similar to medium-term planning but it is carried out for a shorter time horizon with smaller time scale (e.g., daily basis). Short-term production planning is usually called production scheduling and it deals with the assignment and sequencing of tasks for specific production units. Integration of these different decision levels is one of the main issues in the petroleum supply chain management due to the large physical supply network and the complex operations in an oil refinery.[1]

The refinery is the main element of the petroleum supply chain. The final section of a crude-oil refinery consists of the blending units to produce the final liquid products and shipping operations. Minimization of the gasoline blending costs has a major impact on profitability of crude-oil refineries.[2,3] A sample gasoline blending system is shown in Figure 2. Accurate optimization of gasoline blends requires nonlinear models. Nonlinearities may appear in the blending model due to (1) nonlinear blending properties, (2) inclusion of the qualities of future product inventory in the multiperiod model, and (3) other attributes of the pooling problem. Kelly[4] pointed out that formulation of planning models with nonlinearities is becoming more spread in practice due to: (1) recent complex government regulations, (2) raw materials being more expensive and of poorer quality, (3) new and more sophisticated production processes, and (4) higher energy, chemical, and utility costs.

One approach to gasoline blending is to compute blend recipes and detailed blend schedules simultaneously via continuous time model.[5,6] Due to complexity of the mixed integer formulation, such approaches have been limited to using linear blending models. Even with linear models, this approach often leads to very large computational times. Another approach is to decompose the problem into planning and scheduling. Gasoline blend planning determines the ratios of blend components to be used to produce specific products (i.e., the blend recipes) and the volume to blend of each product along the planning horizon in order to meet product quality specifications, product demand requirements, and minimum blend cost. This has the advantage that nonlinear models can be used at the planning, as integer variables appear only with the minimum blend threshold constraint. As blend plans are feasible at the time period boundaries, the constraints they provide for detailed blend scheduling may be intraperiod infeasible. Hence, integration of planning and scheduling is required to ensure feasibility.

Additional Supporting Information may be found in the online version of this article.

Correspondence concerning this article should be addressed to V. Mahalec at mahalec@mcmaster.ca.

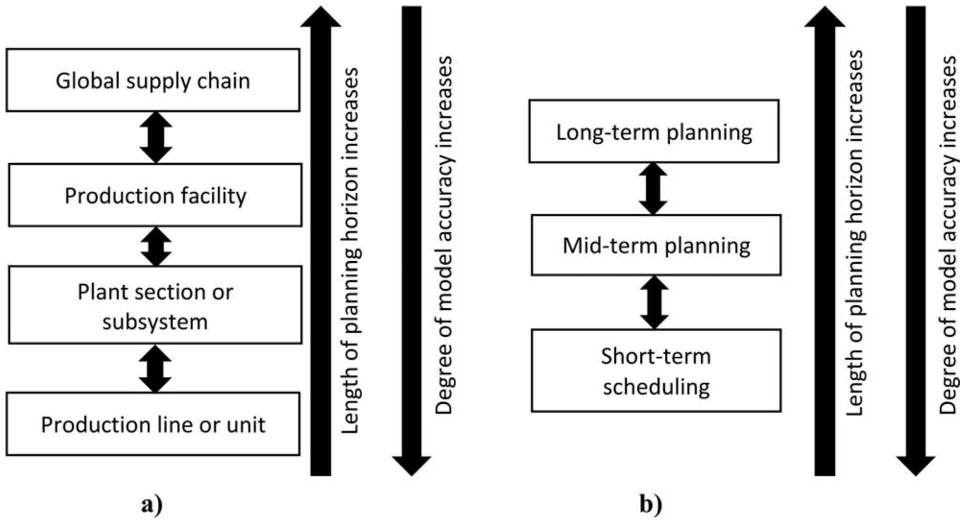

Global supply chain
Production facility
Plant section or subsystem
Production line or unit
Length of planning horizon increases
Degree of model accuracy increases
a)
Long-term planning
Mid-term planning
Short-term scheduling
Length of planning horizon increases
Degree of model accuracy increases
b)

Figure 1. General scheme of hierarchical production planning with respect to (a) spatial scale and (b) time scale.

One problem that practitioners face when using nonlinear, multiperiod blend planning models is the blend recipe variations from one time period to another along the planning horizon, at the solution of the optimization model. In general, blend recipes vary for every blend instance and every time period, and they differ as well from one solver to another (even though the value of the economic objective function of the solutions is the same). Current practice is to avoid such variations by penalizing deviations of blend recipes in every period from some preferred blend recipe (e.g., Mendez et al.[7]). The disadvantage of this technique is that the final blend recipe and the cost will depend on the value given to the penalty coefficients. Another option is to use a set of preferred blend recipes (e.g., Jia and Ierapetritou[2]); however, such blends may not be the lowest cost blends.

This work presents an inventory pinch based, two-level algorithm to solve two problems:

1. Compute a blend plan based on a nonlinear blend model, such that the plan has a minimum number of optimal blend recipes along the planning horizon, and
2. Integrate nonlinear blending with approximate scheduling.

At the first level, we use the inventory pinch concept to determine minimal number of periods, each one of them having a single blend recipe for each grade of gasoline. The inventory pinch points delineate the initial time periods in a nonlinear programming (NLP) model used to compute these optimal blend recipes. At the second level, we use these optimal blend recipes from the first level to formulate a fixed-recipe multiperiod mixed-integer linear programming (MILP) to compute:

1. Blend plan: how much of each grade to produce and when to produce it, and when to change swing tanks from service to another. This model provides feasible inventory profiles at the second level period boundaries.
2. Approximate schedule: determine the production sequence that minimizes blender switches and switches of the swing tanks from one service to another along the planning horizon. Period boundaries at the second level are typically 1/2 day or 1 day long; that duration is set based on the minimum time a swing tank is expected to be in a given service. We call this an approximate schedule because it is computed by an aggregated model, which does not consider minimum changeover times and other logistic rules that may be in place. This is similar to the definition of approximate scheduling model used by Maravelias and Sung.[8]

The second level MILP model contains slack variables to obtain a numerical solution even if it is not physically feasible. If the second level MILP slack variables have non-zero values, we subdivide the corresponding period at the first level NLP model, recompute the blend recipes and then recompute the second level. Hence, we increase number of recipes only when such a change is required to ensure optimality and feasibility of the solution.

For blend planning problems, we compare our solutions to the corresponding full-space mixed-integer nonlinear programming (MINLP) model. The results show that the blend costs computed by the full-space MINLP and our algorithm are the same. Our solutions have substantially smaller number of distinct blend recipes and also require much shorter

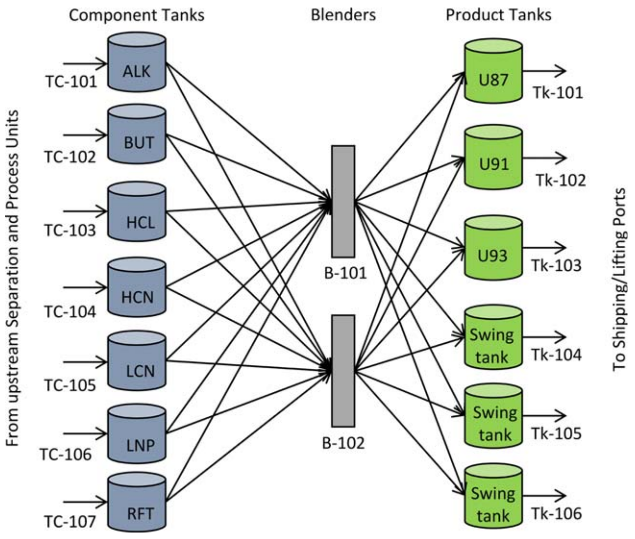


Figure 2. Sample gasoline blending system.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

execution times. Approximate schedules computed by our algorithm define constraints for detailed scheduling. Part II of this article integrates planning and detailed scheduling by introducing a third decomposition level, which computes detailed schedule.

Problem Statement

Blend planning

In this work, we address the gasoline blend planning problem stated as follows:

Given.

1. A planning horizon [0, H].
2. A set of components, their properties, initial inventories, costs, and flow rates along the planning horizon (i.e., supply profile).
3. A set of products (i.e., gasoline grades) with prescribed minimum and maximum quality specifications, their initial inventories, and corresponding initial quality.
4. A set of delivery orders for each product along the planning horizon (i.e., demand profile).
5. The maximum blending capacity of each blender.
6. A set of storage tanks and their minimum and maximum hold ups.
7. Nonlinear blending model.

The Objective is to Determine.

1. The products that the blenders should produce in each time period and how much.
2. The best blend recipes for each product.
3. The inventory profiles of all the component and product tanks.
4. Swing tanks allocation in each time period.

Minimizing.

1. The blend cost associated with the amount of blend components used.

Subject to.

1. Extending the use of the same blend recipe as long as possible.
2. If a blender is to produce a product, it must blend at least a minimum amount.

Assuming.

1. Flow rate profile of each component from the upstream process is piecewise constant.
2. Component quality profile is piecewise constant.
3. Perfect mixing occurs in the blender.
4. Changeover times between product runs on the blender are product dependent but sequence independent.
5. Each order involves only one product and is completed during the respective time delivery window (otherwise, the problem is considered infeasible).
6. Swing tanks are allocated to a particular service throughout duration on a given period at the second level of the algorithm.

Approximate blend scheduling

Blend planning problem is extended to approximate blend scheduling by:

Minimizing.

1. The blend cost associated with the amount of blend components used plus the cost of switching blender operations plus the cost of switching swing tanks to different service.

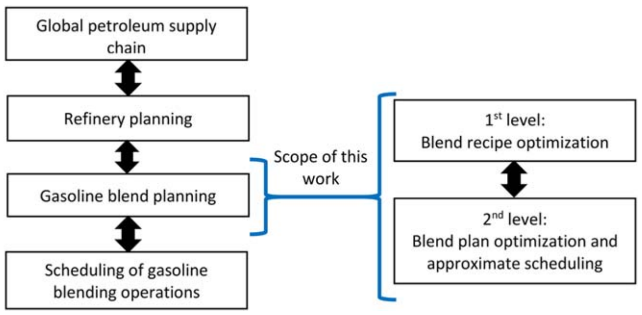

Global petroleum supply chain
Refinery planning
Gasoline blend planning
Scheduling of gasoline blending operations
Scope of this work
1st level: Blend recipe optimization
2nd level: Blend plan optimization and approximate scheduling

Figure 3. Proposed decomposition of the gasoline blend planning problem.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Solution Approach

Although production planning optimization of the global supply chain of an enterprise can be carried out by a single MILP and MINLP model, a hierarchical framework is preferred because: (1) it solves more tractable and smaller-size models and (2) it does not require large amount of forecast data, thus reducing the magnitude of the forecast errors in the planning process.[9] In a hierarchical planning system, decisions at the upper levels are made first and they impose constraints on the decisions at the lower levels. Lower levels' solutions provide feedback to evaluate the upper levels' decisions. Each hierarchical level has its own characteristics, including length of the planning horizon, level of detail of the required information and forecasts (see Figure 1), and scope of the planning activity.[9]

In this work, we solve the gasoline blend planning problem using the hierarchical framework shown in Figure 3. The first level computes the optimal blend recipes and the second level computes how much of a given product and when it should be blended using the recipes from the first level. We assume that the flow rates of the refinery intermediate products arriving to the blending system (i.e., blend components) are given by the solution of the refinery planning level; therefore, the inventory cost need not to be included in the gasoline blend planning model as the refinery will carry the inventories as either blend components or as finished gasoline grades.

Aggregate planning

To reduce the disadvantages of a detailed formulation (e.g., large execution times, intractable models, large effect of the forecast errors and so forth), models at the upper levels are usually aggregated formulations of the lower levels.[9] The adjective "aggregated" indicates that (1) some resources or tasks have been grouped into corresponding families and/or (2) the availability of the resources in one time interval is the cumulative amount available during that interval. Several aggregation/disaggregation techniques for production planning have been discussed in the literature. Nam and Logendran[10] presented a survey of methodologies to formulate and solve aggregate production planning problems.

Bitran and Hax[9] proposed a hierarchical planning model of a multiproduct batch plant. Feedback and interactions among the hierarchical levels are carried out by incorporating conditions not included initially in the models.

Axsater and Jonsson[11] presented a planning model for a manufacturing company where items and machines are grouped according to the ratio between capacity requirements and available capacity, or according to the flows of items. Disaggregation is carried out by simulation using two different heuristic procedures.

Verderame and Floudas[12] proposed a multiperiod MILP planning model of a multisite production and distribution network. The model computes the daily production profile for each facility and the shipment profiles from the facilities to the distribution centers. Key assumptions in their model are: (1) sequencing constraints are not included (i.e., the solution of the planning model will be used as a basis to compute the detailed schedule), (2) more than one task can be executed by the same unit in the same facility and the same time period, and (3) unused time of the production units is transferred to future time periods. Therefore, the time periods, even as small as one day, can be considered as aggregated periods. Assumption (1) reduces the number of discrete variables, and Assumptions (2) and (3) eliminate unnecessary downtime of the processing units due to finite discretization of the time horizon. Comparison with the Kallrath planning model for multisite production (Timpe and Kallrath[13]) shows that Verderame and Floudas[12] model yields a tighter upper bound on the true production capacity.

Thakral and Mahalec[14] developed an algorithm that optimizes blend recipes using a multiperiod MINLP planning model, minimizes blender switches through the use of a genetic algorithm, and detects infeasibilities via agent-based simulation. The time periods of the MINLP model are considered as aggregated as the same resources (e.g., blenders) are shared for different tasks (e.g., blends of different products) in the same time period; therefore, intraperiod infeasibilities may appear. If an infeasibility is encountered, the corresponding period is subdivided and the blend recipes are recomputed. The iterative algorithm stops when no infeasibilities appear.

Pinch analysis was used in aggregate production planning by Singhvi and Shenoy.[15] They presented the demand and production composite (cumulative) curves, as well as a grand composite curve that showed the inventory levels as a function of time. They define the pinch as the point where the production composite touches the demand composite. They considered only one product and their main objective was to determine the production targets along the planning horizon from the pinch analysis, and then use these targets to solve a MILP model that minimizes the material, inventory, and labor costs, subject to material balance equations and production capacity constraints. The inventory levels are penalized in the objective function but no inventory capacity constraints are included. This methodology aims to provide at least a near-optimal solution. Singhvi et al.[16] extended the pinch analysis to the scenario with multiple products and presented an algorithm to determine the production sequence based on some heuristic rules and assuming that the demand must be met only at the end of the horizon. Ludwig et al.[17] applied the pinch analysis of Singhvi and Shenoy[15] to a process with seasonal supply. The composite supply curve is constructed similarly to the composite demand curve and the composite production curve must not cross either curve to avoid supply shortages or product stock-outs, respectively.

Foo et al.[18] implemented the graphical pinch methodology from Singhvi and Shenoy[15] into an algebraic technique which can be easily programmed into a spreadsheet. They include minimum and maximum product inventory constraints and the capability of scheduling a plant shut down. However, their algorithm considers only one product. They pointed out that further improvements of this technique were required to handle more complex supply chain planning problems; they expressed a belief that pinch analysis may provide a deeper analysis of the physical problem. They viewed the pinch point as an indicator of the time within a planning horizon when more accurate forecast data were needed because the pinch point represented the scheduling bottleneck.

Castillo et al.[19] defined the inventory pinch point for the gasoline blend planning problem in a similar manner to Singhvi and Shenoy,[15] but they utilized it to define the aggregated time intervals where one single blend recipe per product was likely to be used without incurring an infeasible operation. In addition, Castillo et al.[19] included in their model: nonlinear product quality constraints, blending equations, availability of raw materials, time delivery windows, and capacity loss due to product switching in the blenders.

Inventory pinch concept

In this work, we use an aggregation/disaggregation approach based on inventory pinch points in order to (1) reduce execution times when compared with full-space MINLP model and (2) reduce the number of different blend recipes. A brief review of the inventory pinch concept is presented next; a more detailed explanation is found in Castillo et al.[19]. To explain the inventory pinch concept, we define the cumulative total demand (CTD) curve and the cumulative average total production (CATP) curve. The CTD curve is constructed by adding the cumulative demands of all products over time. The CATP curve consists of straight line segments, its starting point at time zero is the initial total product inventory available to deliver (i.e., the sum of all initial product inventories minus the minimum hold ups of the tanks), denoted as $V(0)$, and its final point at the end of the planning horizon corresponds to that of the CTD curve plus the final aggregated target inventories. The slope of each segment of the CATP curve must be equal or greater than zero and less than the maximum aggregated blending capacity (i.e., all blenders are aggregated as one single blending unit). The slope of the CATP curve can only change if the CATP is touching the CTD curve. CATP curve has the minimum number of segments required to remain above the CTD curve within the planning horizon. The inventory pinch points are defined as the points in time where the CATP curve changes its slope. Figure 4 shows examples of this concept and the grand composite curve which is computed as the difference between the cumulative total production (CTP) curve and the CTD curve. The inventory pinch points delimit the smallest number of time intervals were a constant blend recipe is likely to lead to a feasible blend plan; that is, no inventory infeasibilities appear when disaggregating the first level decisions at the second level. An inventory infeasibility is defined as the missing volume to fulfill certain demand order or the overflow (or runout) that occurs in a storage tank.

Outline of the algorithm

At the first level, the boundaries of the time periods are delimited by the inventory pinch points, or by the times when the quality values of the blend components change, and the times when the unit cost of blend components or products vary. At the second level, the boundaries of the

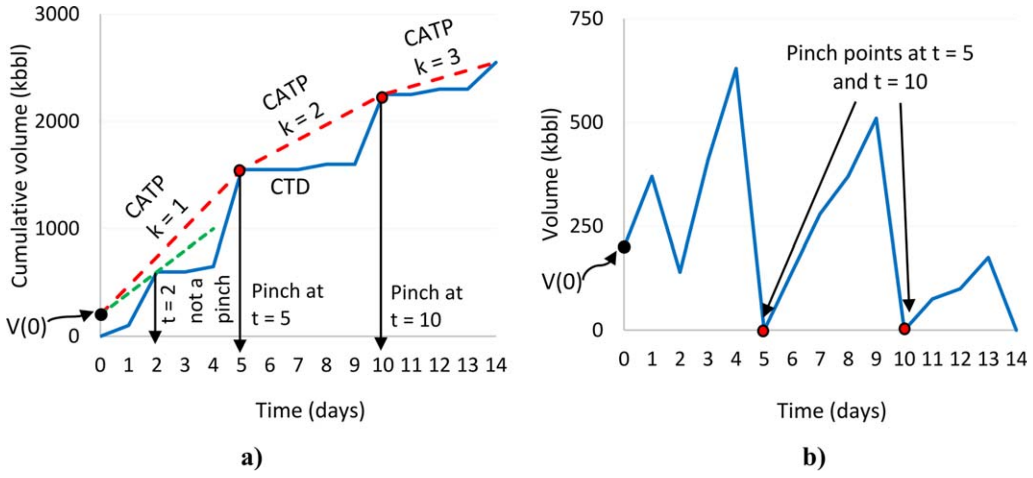


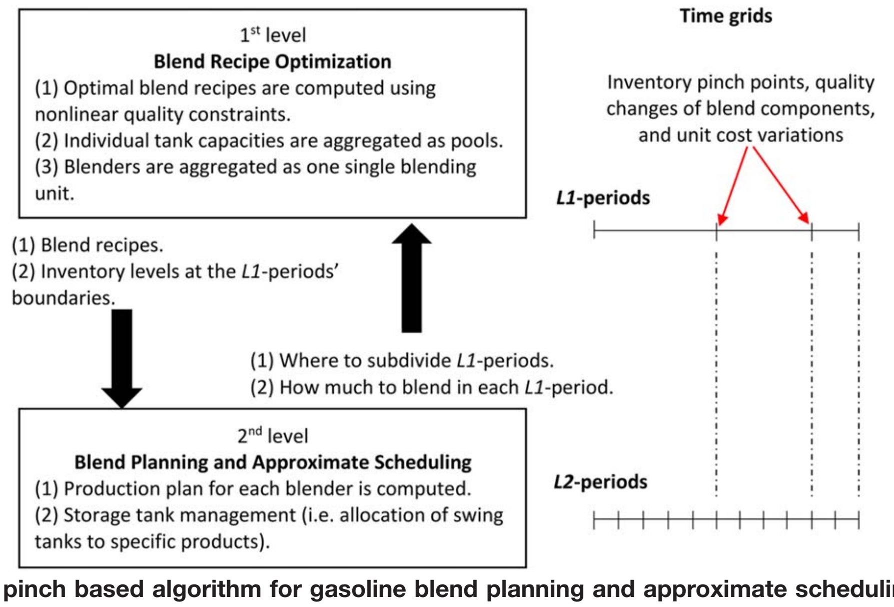
Figure 4. Inventory pinch point examples (a) on the cumulative curves and (b) on the composite curve.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

time periods are defined by the planner based on operational conditions, for example, the minimum time a storage tank will be holding a specific product or the minimum time that a blending unit will require to produce the minimum threshold amount. In addition, boundaries of the first level also become period boundaries at the second level. Notice that the length of the time periods at either level does not need to be uniform. For sake of exposition, the time periods of the first level are denoted as L1-periods and those of the second level as L2-periods. One L1-period contains one or more L2-periods; therefore, the product demand, blend component supply, and blend capacity in a L1-period are the aggregated values of the corresponding L2-periods (see Figure 5).

The inventory pinch points delineate the time periods of constant blend recipes that do not produce inventory infeasibilities due to peaks in demand because these peaks are already considered in the computation of blend recipes. However, other inventory infeasibilities can appear due to the logistic constraints such as minimum blend runs, specific production sequences, or due to highly variable supply profiles of blend components. As these infeasibilities cannot be detected in advance, they will be accounted for in an iterative manner that refines the blend recipes if inventory infeasibilities are encountered. Hence, the inventory pinch concept allows us to start with the smallest number of time periods at the first level and increase that number only if required to eliminate inventory infeasibilities. It also provides the minimum production quantities to meet the demand of each product by solving a simple material balance for each time period delimited by pinch points: (aggregated production) = (aggregated demand) + (final inventory) − (initial inventory).

Although we consider only one storage tank per blend component, we assume that there are tanks that may store different gasoline grades at different times in the planning horizon, that is, swing tanks. At the first level, individual product tanks are aggregated into a single inventory capacity,



1st level
Blend Recipe Optimization
(1) Optimal blend recipes are computed using nonlinear quality constraints.
(2) Individual tank capacities are aggregated as pools.
(3) Blenders are aggregated as one single blending unit.
(1) Blend recipes.
(2) Inventory levels at the L1-periods' boundaries.
(1) Where to subdivide L1-periods.
(2) How much to blend in each L1-period.
2nd level
Blend Planning and Approximate Scheduling
(1) Production plan for each blender is computed.
(2) Storage tank management (i.e. allocation of swing tanks to specific products).
Time grids
Inventory pinch points, quality changes of blend components, and unit cost variations
L1-periods
L2-periods

Figure 5. Inventory pinch based algorithm for gasoline blend planning and approximate scheduling.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

that is, a product pool; then at the second level, these pools are disaggregated into individual storage tanks.

Mathematical Models

The models presented here are developed based on the gasoline blend planning problem; however, they can be used for similar processes with few modifications. The following sets are used:

A =	$\{(\alpha)$ ∣ set of different supply flow rates of blend components$\}$
Bl =	$\{(\text{bl})$ ∣ set of blenders$\}$
E =	$\{(e)$ ∣ set of quality properties$\}$
I =	$\{(i)$∣ set of blend components$\}$
J =	$\{(j)$ ∣ set of product storage tanks$\}$
K =	$\{(k)$ ∣ set of time periods at the first level or L1-periods$\}$
M =	$\{(m)$ ∣ set of time periods at the second level or L2-periods$\}$
O =	$\{(o)$ ∣ set of orders$\}$
P =	$\{(p)$ ∣ set of products$\}$
BJ =	$\{(\text{bl}, j)$ ∣ blender bl can feed tank $j\}$
BP =	$\{(\text{bl}, p)$ ∣ blender bl can process product $p\}$
JP =	$\{(j, p)$ ∣ tank j can hold product $p\}$
KA =	$\{(k, \alpha)$ ∣ blend component supply profile a occurs within L1-period $k\}$
MA =	$\{(m, \alpha)$ ∣ blend component supply profile a occurs within L2-period $m\}$
MK =	$\{(m, k)$ ∣ L2-periods contained in each L1-period$\}$
MKF =	$\{(m, k)$ ∣ last L2-period of each L1-period$\}$

First level—Blend recipe optimization

The objective of the first level is to determine the volume fractions (i.e., blend recipes) to mix different blend components available at the refinery in such a way that the final products meet the demand and quality specifications and the blend cost is the minimum possible. The values of the product demand, blend component supply, and blend capacity are aggregated values for each of the L1-periods. At this level, product storage tanks are aggregated into product pools. The objective function defined by Eq. 1 minimizes the blend cost and the inventory infeasibilities. The penalty coefficients for the product slacks variables are much greater than the penalty coefficients for the component slacks variables (i.e., $\text{Penalty}_{\text{pool,L1}} >> \text{Penalty}_{\text{bc,L1}}$). The blend cost is given by Eq. 2. Slack variables will be zero at the optimal solution (i.e., penalty coefficients will not affect the final blend cost); if slack variables have non-zero values, then the problem is infeasible as there are not enough blend components to blend products as required by the quality specifications or demand requirements.

Since the inventories will be at the minimum level allowed at the end of each L1-period (because the CATP curve touches the CTD curve), the inventory cost need not to be included in the objective function of the first level

$$\min Z_{\text{L1}} = \text{Blend Cost}_{\text{L1}} + \sum_{k=1}^{K} \left\{ \begin{array}{l} \sum_{i\in I} \text{Penalty}_{\text{bc,L1}} \cdot \left(S^{+}_{\text{bc,L1}}(i,k)+S^{-}_{\text{bc,L1}}(i,k)\right) \\ + \sum_{p\in P} \text{Penalty}_{\text{pool,L1}} \cdot \left(S^{+}_{\text{pool,L1}}(p,k)+S^{-}_{\text{pool,L1}}(p,k)\right) \end{array} \right\} \tag{1}$$

$$\text{Blend Cost}_{\text{L1}} = \sum_{k=1}^{K} \left(\sum_{i\in I, p\in P} \text{Cost}_{\text{bc}}(i) \cdot V_{\text{comp,L1}}(i,p,k) \right) \tag{2}$$

The volume balance equations and inventory constraints for component and product tanks for every period k are given by Eqs. 3–6

$$\begin{gathered} \sum_{a\in \text{KA}} \left[F_{\text{bc}}(i,a) \cdot t_{\text{bc,L1}}(a,k)\right] \\ +V_{\text{bc,L1}}(i,k-1) - V_{\text{bc,L1}}(i,k) - \sum_{p\in P} \\ V_{\text{comp,L1}}(i,p,k) + S^{+}_{\text{bc,L1}}(i,k) - S^{-}_{\text{bc,L1}}(i,k) = 0 \\ \forall i,\ k \geq 1 \end{gathered} \tag{3}$$

$$\begin{gathered} V_{\text{blend,L1}}(p,k) + V_{\text{pool,L1}}(p,k-1) - V_{\text{pool,L1}}(p,k) \\ -\text{Demand}_{\text{pool,L1}}(p,k) + S^{+}_{\text{pool,L1}}(p,k) - \\ S^{-}_{\text{pool,L1}}(p,k) = 0 \\ \forall p,\ k \geq 1 \end{gathered} \tag{4}$$

$$V^{\min}_{\text{bc}}(i) \leq V_{\text{bc,L1}}(i,k) \leq V^{\max}_{\text{bc}}(i) \quad \forall i,k \tag{5}$$

$$V^{\min}_{\text{pool}}(p) \leq V_{\text{pool,L1}}(p,k) \leq V^{\max}_{\text{pool}}(p) \quad \forall p,k \tag{6}$$

In Eq. 3, α stands for a specific supply profile of blend components. Parameter $t_{\text{bc,L1}}(\alpha,k)$ defines the aggregate duration of the time intervals when supply profile α occurs during period k. Notice that the sum of the durations of these time intervals must be equal to the length of period k

$$\sum_{a\in \text{KA}} t_{\text{bc,L1}}(a,k) = t_{\text{L1}}(k)$$

Therefore, the first term in Eq. 3 represents the aggregated availability of blend component i in period k. Equations 7 and 8 specify the initial inventories of the component tanks and product pools, respectively

$$V_{\text{bc,L1}}(i,k=0) = V^{\text{start}}_{\text{bc}}(i) \quad \forall i \tag{7}$$

$$V_{\text{pool,L1}}(p,k=0) = V^{\text{start}}_{\text{pool}}(p) \quad \forall p \tag{8}$$

Equations 9 and 10 are used to determine the blend recipes. Equation 11 sets the minimum and maximum compositions of each product

$$V_{\text{comp,L1}}(i,p,k) = r(i,p,k) \cdot V_{\text{blend,L1}}(p,k) \quad \forall i,p,k \geq 1 \tag{9}$$

$$\sum_{i\in I} r(i,p,k) = 1 \quad \forall p,k \geq 1 \tag{10}$$

$$r^{\min}(i,p) \leq r(i,p,k) \leq r^{\max}(i,p) \quad \forall i,p,k \geq 1 \tag{11}$$

Equations 12a and 12b are the linear quality constraints, that is, the quality property e is assumed to blend linearly in a volumetric basis

$$\begin{gathered} \sum_{i\in I} V_{\text{comp,L1}}(i,p,k) \cdot Q_{\text{bc}}(i,e,k) \\ \leq Q^{\max}_{\text{pr}}(p,e) \cdot V_{\text{blend,L1}}(p,k) \quad \forall e,p,k \geq 1 \end{gathered} \tag{12a}$$

$$\begin{gathered} \sum_{i\in I} V_{\text{comp,L1}}(i,p,k) \cdot Q_{\text{bc}}(i,e,k) \\ \geq Q^{\min}_{\text{pr}}(p,e) \cdot V_{\text{blend,L1}}(p,k) \quad \forall e,p,k \geq 1 \end{gathered} \tag{12b}$$

Equations 13 and 14 are the nonlinear quality constraints; they are expressed as a function of the blend recipe and the qualities of blend components

$$Q_{\text{pr}}(p,e,k) = f(r(i,p,k), Q_{\text{bc}}(i,e,k)) \quad \forall e,p,k \geq 1 \tag{13}$$

$$Q^{\min}_{\text{pr}}(p,e) \leq Q_{\text{pr}}(p,e,k) \leq Q^{\max}_{\text{pr}}(p,e) \quad \forall e,p,k \geq 1 \tag{14}$$

The only nonlinear quality constraint in our case studies is Eq. 15, which computes the Reid vapor pressure (RVP) (Sing et al.[3])

$$Q_{\mathrm{pr}}(p,e,k)=\left[\sum_{i\in I} r(i,p,k)\cdot Q_{\mathrm{bc}}(i,e,k)^{1.25}\right]^{0.8} \quad \forall e=\mathrm{RVP},\ p,\ k\ \geq\ 1 \tag{15}$$

Equations 1–15 complete the NLP model for the first level. In the first iteration of the algorithm, the volumes to be blended in each L1-period are the minimum amount required to fulfill the demand. This provides a lower bound of the global blend cost.

It is important to note that discrete variables are not required at the first level to observe minimum blend size thresholds and maximum blenders' capacities. The volumes to be blended are adjusted if needed before solving the first level model (see Algorithm Steps).

*Second level—Blend plan optimization and approximate blend schedule optimiza*tion

The blend plan defines (1) how much to blend of each product and in which blender in each L2-period; (2) allocation of swing tanks to specific products in each L2-period; and (3) the inventory profiles of all storage tanks along the planning horizon.

The second level uses the optimal blend recipes from the first level: the blend recipes of each L1-period are fixed in the corresponding L2-periods. The aggregated decisions of the first level are now disaggregated at the second level. This disaggregation step uses a MILP model to set constraints such as the minimum blend size threshold, and others that require discrete variables. The second level is solved first as a blend planning problem. This enables rapid computation of an optimal blend plan which is feasible with respect to the component and product availabilities and inventories. Once an optimal blend plan is known, scheduling of blenders and swing tankage is carried out (approximate scheduling).

Blend Planning—Objective Function at the Second Level. The objective function of the second level, Eq. 16, minimizes the blend cost and the inventory slack variables. Inventory slack variables will be zero at the solution of the second level if a feasible operation can be obtained using the blend recipes computed at the first level. If this is not the case, the inventory slacks will show which specific products and in which L2-periods cannot be produced in the amounts required to meet the demand. To ensure this, two things are required:

1. Penalty coefficients for the products' inventory slack variables are smaller in comparison with the penalty for the components' inventory slacks (i.e., $\mathrm{Penalty}_{\mathrm{bc,L2}} >> \mathrm{Penalty}_{\mathrm{pr,L2}}(m)\ \forall\ m$).
2. The penalty coefficients for the product inventory slacks decrease with time (i.e., $\mathrm{Penalty}_{\mathrm{pr,L2}}(m) > \mathrm{Penalty}_{\mathrm{pr,L2}}(m+1)\ \forall\ m$) to move the inventory infeasibilities as late as possible in the planning horizon, thus maximizing the use of a given blend recipe. The penalty coefficients must decrease as fast as possible and after each L1-period boundary a significant change must take place.

If the solution of the MILP has inventory infeasibilities, it signifies that component supply or blender constraints are such that the recipes computed at the first level are not feasible within a L1-period. The algorithm will subdivide such L1-periods and reoptimize the blend recipes

$$\min Z_{\mathrm{L2}}^{\mathrm{feas}}=\mathrm{Blend\,Cost}_{\mathrm{L2}}+\sum_{m=1}^{M}\left\{\begin{array}{c}\sum_{i\in I}\mathrm{Penalty}_{\mathrm{bc,L2}}(m)\cdot\left(S^{+}_{\mathrm{bc,L2}}(i,m)+S^{-}_{\mathrm{bc,L2}}(i,m)\right)\\ +\sum_{p\in P}\mathrm{Penalty}_{\mathrm{pool,L2}}(m)\cdot\left(S^{+}_{\mathrm{pool,L2}}(p,m)+S^{-}_{\mathrm{pool,L2}}(p,m)\right)\\ +\sum_{j\in J}\mathrm{Penalty}_{\mathrm{pr,L2}}(m)\cdot\left(S^{+}_{\mathrm{pr,L2}}(j,m)+S^{-}_{\mathrm{pr,L2}}(j,m)\right)\end{array}\right\} \tag{16}$$

Equation 17 computes the blend cost

$$\mathrm{Blend\,Cost}_{\mathrm{L2}}=\sum_{m=1}^{M}\left(\sum_{(\mathrm{bl},p)\in\mathrm{BP}}\sum_{i\in I}\mathrm{Cost}_{\mathrm{bc}}(i)\cdot V_{\mathrm{comp,L2}}(i,p,\mathrm{bl},m)\right) \tag{17}$$

Approximate Scheduling—Objective Function at the Second Level. After the blend recipes of the first level have been proven to generate a feasible blend plan, then, the approximate scheduling is solved. The scheduling objective function at the second level (Eq. 18) minimizes the number of possible product transitions in the blenders ($\mathrm{xe}_{\mathrm{L2}}(\mathrm{bl},g)$) and in the product storage tanks ($\mathrm{ue}_{\mathrm{L2}}(j,m)$), as well as the number of blend instances ($x_{\mathrm{L2}}(p,\mathrm{bl},m)$). Binary variable $x_{\mathrm{L2}}(p,\mathrm{bl},m)$ defines a blend instance; that is, it determines if product p is going to be produced in blender bl during period m if its value is 1. Blend instances are penalized because a solution at the second level can suggest to blend the same product in the same blender for several adjacent L2-periods, thus not incurring in a penalty for product changeover in the blender; however, long blend runs are inefficient if the blender is not working close to full capacity. Therefore, it is better to have the minimum blend instances, and then reduce the expected number of product changeovers (i.e., $\mathrm{PenaltyBR}_{\mathrm{L2}}(\mathrm{bl}) >> \mathrm{PenaltyBS}_{\mathrm{L2}}$, for all bl). Because the inventory slacks are zero at the blend planning solution, we know that the blend recipes and inventory targets from the first level solution can yield a feasible blend plan; then, if those are fixed at the second level, the blend cost need not to be included in Eq. 18

$$\min Z_{\mathrm{L2}}^{\mathrm{opt}}=\sum_{m=1}^{M}\left\{\begin{array}{l}\sum_{(\mathrm{bl},p)\in\mathrm{BP}}\mathrm{PenaltyBR}_{\mathrm{L2}}(\mathrm{bl})\cdot x_{\mathrm{L2}}(p,\mathrm{bl},m)\\ +\sum_{j\in J}\mathrm{PenaltyTS}_{\mathrm{L2}}(j)\cdot\mathrm{ue}_{\mathrm{L2}}(j,m)\end{array}\right\}+\sum_{g=1}^{G}\left(\sum_{\mathrm{bl}\in\mathrm{Bl}}\mathrm{PenaltyBS}_{\mathrm{L2}}\cdot\mathrm{xe}_{\mathrm{L2}}(\mathrm{bl},g)\right) \tag{18}$$

Thus, the total cost at the second level is given by $Z_{\mathrm{L2}}=Z_{\mathrm{L2}}^{\mathrm{feas}}+Z_{\mathrm{L2}}^{\mathrm{opt}}$. Given the assumption that the flow rates of blend components are given by the solution of the refinery planning level, the inventory cost need not to be included in the gasoline blend planning level as the refinery will carry the total inventories as either blend components or as finished gasoline grades.

Constraints at the Second Level. The volume balance equations for blend component tanks for every L2-period are given by Eq. 19

$$\sum_{a\in \mathrm{MA}} F_{\mathrm{bc}}(i,a)\cdot t_{\mathrm{bc,L2}}(a,m)+V_{\mathrm{bc,L2}}(i,m-1) -V_{\mathrm{bc,L2}}(i,m)-\sum_{(\mathrm{bl},p)\in \mathrm{BP}} V_{\mathrm{comp,L2}}(i,p,\mathrm{bl},m) +S^{+}_{\mathrm{bc,L2}}(i,m)-S^{-}_{\mathrm{bc,L2}}(i,m)=0 \quad \forall i, m\geq 1 \tag{19}$$

Once again, α stands for a specific supply profile of blend components. Parameter $t_{\mathrm{bc,L2}}(\alpha,m)$ defines the aggregate duration of the time intervals when supply profile α occurs during period m. Notice that the sum of the durations of these time intervals within a L2-period, must be equal to the length of such L2-period

$$\sum_{a\in \mathrm{MA}} t_{\mathrm{bc,L2}}(a,m)=t_{\mathrm{L2}}(m)$$

Similarly to Eq. 3, the first term in Eq. 18 represents the aggregated availability of blend component i in period m. In our case studies, there is only one interval α in each L2-period m.

Equation 20 fixes the blend recipe in its corresponding L2-periods. Note that $r(i,p,k)$ is a parameter and not a variable at the second level

$$V_{\mathrm{comp,L2}}(i,p,\mathrm{bl},m)=r(i,p,k)\cdot V_{\mathrm{blend,L2}}(p,\mathrm{bl},m) \quad \forall i,\ (p,\ \mathrm{bl})\in \mathrm{BP},\ (m,\ k)\ \in \mathrm{MK} \tag{20}$$

Equation 21 establishes that a blender may only blend a certain number of products during a L2-period. $x_{\mathrm{L2}}(p,\mathrm{bl},m)$ is a binary variable with value of 1 if product p is blended in bl during period m, and 0 otherwise

$$\sum_{p\in \mathrm{BP}} x_{\mathrm{L2}}(p,\mathrm{bl},m)\leq \mathrm{np}(\mathrm{bl})\forall \mathrm{bl}, m\geq 1 \tag{21}$$

Maximum blender capacity is enforced by Eq. 22. Because every product transition in the blender involves an idle time (due to sensor recalibration, equipment start-up, or other reasons), part of the blend capacity is lost. We assume that the blend capacity loss is equal to the product of the maximum blending capacity and the minimum idle time required by a blender to process product p

$$\textstyle\sum_{p\in \mathrm{BP}} V_{\mathrm{blend,L2}}(p,\mathrm{bl},m)+F^{\max}_{\mathrm{blend}}(\mathrm{bl})\cdot \sum_{p\in \mathrm{BP}}\left(\mathrm{it}^{\min}_{\mathrm{blend}}(p,\mathrm{bl})\cdot x_{\mathrm{L2}}(p,\mathrm{bl},m)\right) \leq F^{\max}_{\mathrm{blend}}(\mathrm{bl})\cdot t_{\mathrm{L2}}(m) \quad \forall \mathrm{bl}, m\geq 1 \tag{22}$$

A blend run of a particular product must be less than or equal to the volume that the blender can process at maximum blending capacity in period m (Eq. 23), and greater than a threshold amount prespecified by the planner (Eq. 24). A blend run of any product must be greater than the volume that a blender can process at minimum blending rate in period m (Eq. 25)

$$V_{\mathrm{blend,L2}}(p,\mathrm{bl},m)\leq F^{\max}_{\mathrm{blend}}(\mathrm{bl})\cdot t_{\mathrm{L2}}(m)\cdot x_{\mathrm{L2}}(p,\mathrm{bl},m) \quad \forall (p,\mathrm{bl})\in \mathrm{BP},\ m\geq 1 \tag{23}$$

$$V_{\mathrm{blend,L2}}(p,\mathrm{bl},m)\geq \mathrm{VMIN}_{\mathrm{blend}}(p,\mathrm{bl})\cdot x_{\mathrm{L2}}(p,\mathrm{bl},m) \quad \forall (p,\mathrm{bl})\in \mathrm{BP},\ m\geq 1 \tag{24}$$

$$V_{\mathrm{blend,L2}}(p,\mathrm{bl},m)\geq F^{\min}_{\mathrm{blend}}(\mathrm{bl})\cdot t^{\min}_{\mathrm{blend}}(\mathrm{bl})\cdot x_{\mathrm{L2}}(p,\mathrm{bl},m) \quad \forall (p,\mathrm{bl})\in \mathrm{BP},\ m\geq 1 \tag{25}$$

Equations 22–25 are not enough to guarantee feasibility. The running times of each blend must be estimated and their sum plus the idle time must be equal to or less than the duration of the corresponding L2-period m. Equation 26 constraints the running time of a blend to be equal or greater than the minimum running time. Equations 27 and 28 set the lower and upper limits of a running time of a blend, respectively. Equation 29 enforces that the sum of the running times, plus product changeover times, is equal to or less than the L2-period duration

$$t_{\mathrm{blend,L2}}(p,\mathrm{bl},m)\geq t^{\min}_{\mathrm{blend}}(\mathrm{bl})\cdot x_{\mathrm{L2}}(p,\mathrm{bl},m) \quad \forall (p,\mathrm{bl})\in \mathrm{BP},\ m\geq 1 \tag{26}$$

$$t_{\mathrm{blend,L2}}(p,\mathrm{bl},m)\geq \frac{V_{\mathrm{blend,L2}}(p,\mathrm{bl},m)}{F^{\max}_{\mathrm{blend}}(\mathrm{bl})} \quad \forall (p,\mathrm{bl})\in \mathrm{BP},\ m\geq 1 \tag{27}$$

$$t_{\mathrm{blend,L2}}(p,\mathrm{bl},m)\leq \frac{V_{\mathrm{blend,L2}}(p,\mathrm{bl},m)}{F^{\min}_{\mathrm{blend}}(\mathrm{bl})} \quad \forall (p,\mathrm{bl})\in \mathrm{BP},\ m\geq 1 \tag{28}$$

$$\sum_{p\in \mathrm{BP}} t_{\mathrm{blend,L2}}(p,\mathrm{bl},m)+\sum_{p\in \mathrm{BP}} \mathrm{it}^{\min}_{\mathrm{blend}}(p,\mathrm{bl})\cdot x_{\mathrm{L2}}(p,\mathrm{bl},m) \leq t_{\mathrm{L2}}(m) \quad \forall \mathrm{bl}, m\geq 1 \tag{29}$$

Equations 30 and 31 are the volumetric balance equations on the product pools and the individual storage tanks, respectively

$$\textstyle\sum_{\mathrm{bl}\in \mathrm{BP}} V_{\mathrm{blend,L2}}(p,\mathrm{bl},m)+V_{\mathrm{pool,L2}}(p,m-1) -V_{\mathrm{pool,L2}}(p,m)-\mathrm{Deliver}_{\mathrm{pool,L2}}(p,m) +S^{+}_{\mathrm{pool,L2}}(p,m)-S^{-}_{\mathrm{pool,L2}}(p,m)=0 \quad \forall p, m\geq 1 \tag{30}$$

$$\textstyle\sum_{\mathrm{bl}\in \mathrm{BPJ}} V_{\mathrm{trans,L2}}(j,p,\mathrm{bl},m)+V_{\mathrm{pr,L2}}(j,p,m-1) -V_{\mathrm{pr,L2}}(j,p,m)-\mathrm{Deliver}_{\mathrm{pr,L2}}(j,p,m) +S^{+}_{\mathrm{pr,L2}}(j,m)-S^{-}_{\mathrm{pr,L2}}(j,m)=0 \quad \forall\ (j,p)\ \in \mathrm{JP},\ m\geq 1 \tag{31}$$

The volume blended of product p in blender bl can only be sent to one product tank j during L2-period m, and that tank j must contain already product p or be empty (i.e., mixing of different products is not allowed in storage tanks). These constraints are represented by Eqs. 32–35. Binary variable $v_{\mathrm{L2}}(j,p,\mathrm{bl},m)$ specifies if product p is transferred from blender bl to product tank j during period m. Binary variable $u_{\mathrm{L2}}(j,p,m)$ specifies if product p is being stored in product tank j during period m

$$V_{\mathrm{trans,L2}}(j,p,\mathrm{bl},m)\leq F^{\max}_{\mathrm{blend}}(\mathrm{bl})\cdot t_{\mathrm{L2}}(m)\cdot v_{\mathrm{L2}}(j,p,\mathrm{bl},m) \quad \forall\ (\mathrm{bl},p)\in\ \mathrm{BP},\ (j,\mathrm{bl})\in\ \mathrm{BJ},\ (j,p)\in\ \mathrm{JP}, m\geq 1 \tag{32}$$

$$\textstyle\sum_{j\in \mathrm{JP}} V_{\mathrm{trans,L2}}(j,p,\mathrm{bl},m)=V_{\mathrm{blend,L2}}(p,\mathrm{bl},m) \quad \forall\ (\mathrm{bl},p)\in\ \mathrm{BP},\ (j,\mathrm{bl})\in\ \mathrm{BJ}, m\geq 1 \tag{33}$$

$$\sum_{j\in \mathrm{JP}} v_{\mathrm{L2}}(j,p,\mathrm{bl},m) \le 1 \quad \forall\ (\mathrm{bl},p)\in \mathrm{BP},\ (j,\mathrm{bl})\in \mathrm{BJ}, m\ge 1 \tag{34}$$

$$v_{\mathrm{L2}}(j,p,\mathrm{bl},m) \le u_{\mathrm{L2}}(j,p,m) \quad \forall\ (\mathrm{bl},p)\in \mathrm{BP},\ (j,\mathrm{bl})\in \mathrm{BJ},\ (j,p)\in \mathrm{JP}, m\ge 1 \tag{35}$$

Equation 36 states that only one product can be stored in a product tank j during a period m. Equation 37 constraints the volume stored of product p in tank j to be less than the maximum capacity of such tank. Equation 38 relates the product pool inventory with the individual tank inventories

$$\sum_{p\in \mathrm{JP}} u_{\mathrm{L2}}(j,p,m)=1 \ \forall\ j,\ m \tag{36}$$

$$V_{\mathrm{pr,L2}}(j,p,m) \le V_{\mathrm{pr}}^{\max}(j)\cdot u_{\mathrm{L2}}(j,p,m) \quad \forall\ (j,p)\in \mathrm{JP}, m \tag{37}$$

$$V_{\mathrm{pool,L2}}(p,m)=\sum_{j\in \mathrm{JP}} V_{\mathrm{pr,L2}}(j,p,m) \quad \forall\ p,m \tag{38}$$

Binary variable ue$_{\mathrm{L2}}$(j,m) is introduced to determine if a product transition in tank j has taken place at the beginning of period m (Eqs. 39 and 40). Equation 42 forces a product transition to occur on tank j if it is empty at the end of the previous period m

$$\mathrm{ue}_{\mathrm{L2}}(j,m) \ge u_{\mathrm{L2}}(j,p,m)-u_{\mathrm{L2}}(j,p,m-1) \quad \forall\ (j,p)\in \mathrm{JP}, m\ge 1 \tag{39}$$

$$\mathrm{ue}_{\mathrm{L2}}(j,m) \ge u_{\mathrm{L2}}(j,p,m-1)-u_{\mathrm{L2}}(j,p,m) \quad \forall\ (j,p)\in \mathrm{JP}, m\ge 1 \tag{40}$$

$$V_{\mathrm{pr,L2}}(j,p,m-1) \le V_{\mathrm{pr}}^{\max}(j)\cdot (1-\mathrm{ue}_{\mathrm{L2}}(j,m)) \quad \forall\ (j,p)\in \mathrm{JP}, m\ge 1 \tag{41}$$

Equation 42 defines the products stored in the product tanks at the beginning of the scheduling horizon

$$u_{\mathrm{L2}}(j,p,m{=}0)=u^{\mathrm{start}}(j,p) \quad \forall\ (j,p)\in \mathrm{JP} \tag{42}$$

Inventory constraints on storage tanks are given by Eqs. 43–45. Maximum pool capacity is no longer enforced at this level as there are constraints on the individual product tanks (see Eq. 37)

$$V_{\mathrm{bc}}^{\min}(i) \le V_{\mathrm{bc,L2}}(i,m) \le V_{\mathrm{bc}}^{\max}(i) \quad \forall\ i,m \tag{43}$$

$$V_{\mathrm{pool,L2}}(p,m) \ge V_{\mathrm{pool}}^{\min}(p) \quad \forall\ p,m \tag{44}$$

$$V_{\mathrm{pr,L2}}(j,p,m) \ge V_{\mathrm{pr}}^{\min}(j)\cdot u_{\mathrm{L2}}(j,p,m) \quad \forall\ (j,p)\in \mathrm{JP}, m \tag{45}$$

Inventory levels at the boundaries of the L1-periods must be equal at the corresponding point in time at the second level. This is enforced by Eq. 46

$$V_{\mathrm{pool,L2}}(p,m)=V_{\mathrm{pool,L1}}(p,k) \quad \forall\ p,(m,k)\in \mathrm{MKF} \tag{46}$$

Equations 47 and 48 set the initial state of the component tanks and product tanks, respectively

$$V_{\mathrm{bc,L2}}(i,m{=}0)=V_{\mathrm{bc}}^{\mathrm{start}}(i) \quad \forall\ i \tag{47}$$

$$V_{\mathrm{pr,L2}}(j,p,m{=}0)=V_{\mathrm{pr}}^{\mathrm{start}}(j,p) \quad \forall\ (j,p)\in \mathrm{JP} \tag{48}$$

Equation 49 constraints the shipped/lifted amount of product p from tank j in period m to be equal or less than the volume that such tank can deliver at maximum delivery rate during period m. The amount of product p delivered by all product tanks is equal to that amount delivered from the corresponding product pool (Eq. 50). Equation 51 establishes that the demand must be met in each L2-period

$$\mathrm{Deliver}_{\mathrm{pr,L2}}(j,p,m) \le D_{\mathrm{pr}}^{\max}(j)\cdot t_{\mathrm{L2}}(m)\cdot u_{\mathrm{L2}}(j,p,m) \quad \forall\ (j,p)\in \mathrm{JP}, m\ge 1 \tag{49}$$

$$\mathrm{Deliver}_{\mathrm{pool,L2}}(p,m)=\sum_{j\in \mathrm{JP}} \mathrm{Deliver}_{\mathrm{pr,L2}}(j,p,m) \quad \forall\ p,m\ge 1 \tag{50}$$

$$\mathrm{Deliver}_{\mathrm{pool,L2}}(p,m)=\mathrm{Demand}_{\mathrm{pr,L2}}(p,m) \quad \forall\ p,m\ge 1 \tag{51}$$

Equation 50 is used when the demand requirement for each L2-period is known exactly, that is, the delivery windows of the orders do not span more than one L2-period. Equation 50 can be substituted by Eqs. 52–55, which are included in order to handle delivery windows spanning several L2-periods. Equation 52 establishes that the amount delivered in period m is equal to a fraction of the total demand of order o, denoted as of$_{\mathrm{L2}}$(o,m). uof$_{\mathrm{L2}}$(o,m) is a parameter that represents if order o can be delivered in period m if its value is 1 (i.e., delivery time window of order o is contained totally or partially by L2-period m), or not if its value is zero. Equation 53 ensures that only the contracted demand for order o is shipped/lifted. Equation 54 forces completion of order o within the scheduling horizon. Equation 55 constraints the fraction delivered of order o during period m to be less than the maximum possible delivery of such order

$$\mathrm{Deliver}_{\mathrm{pool,L2}}(p,m)=\sum_{o\in \mathrm{OP}} \mathrm{Demand}(o)\cdot \mathrm{of}_{\mathrm{L2}}(o,m) \quad \forall\ p,m\ge 1 \tag{52}$$

$$\mathrm{of}_{\mathrm{L2}}(o,m) \le \mathrm{uof}_{\mathrm{L2}}(o,m) \quad \forall\ o,m\ge 1 \tag{53}$$

$$\sum_{m=1}^{M} \mathrm{of}_{\mathrm{L2}}(o,m)=1 \quad \forall\ o \tag{54}$$

$$\mathrm{Demand}(o)\cdot \mathrm{of}_{\mathrm{L2}}(o,m) \le D_{\mathrm{order}}^{\max}(o)\cdot dt_{\mathrm{L2}}(o,m) \quad \forall\ o,m\ge 1 \tag{55}$$

Constraints Specific to the Approximate Scheduling at the Second Level. Equations 56–61 are included in order to compute the product transitions in the blenders. The production sequence of each blender is determined by binary variable y_{L2}(p,bl,g), which is computed based on the values from variable x_{L2}(p,bl,m). The g index refers to position in the sequence where product p is processed by blender bl; that is, g can be interpreted as well as a time slot to allocate the blend instances of period m to estimate the production sequence of blender bl. The number of these slots for one L2-period is equal to the number of different products. Equation 56 defines the products being processed in the blenders at the beginning of the scheduling horizon. Equation 57 ensures that only a product that is being blended in period m is allocated in a slot g corresponding to m. Equation 58 constraints that only one product can be allocated in slot g. Equation 59 allows a product being blended in period m to use more than one slot g corresponding to m. Equations 60 and 61 determine if a product transition has taken place. xe$_{\mathrm{L2}}$(bl,g) is a continuous variable that can only take 0–1 values due to Eqs. 60 and 61, and for being penalized in Eq. 18

$$y_{\mathrm{L2}}(p,\mathrm{bl},g{=}0)=x^{\mathrm{start}}(p,\mathrm{bl}) \quad \forall\ (\mathrm{bl},p)\in \mathrm{BP} \tag{56}$$

$$y_{\mathrm{L2}}(p,\mathrm{bl},g) \le x_{\mathrm{L2}}(p,\mathrm{bl},m) \quad \forall\ p,\mathrm{bl},(g,m)\in \mathrm{GM} \tag{57}$$

$$\textstyle\sum_{p\in P} y_{\mathrm{L2}}(p,\mathrm{bl},g) \le 1 \quad \forall\ \mathrm{bl}, g \ge 1 \tag{58}$$

$$\textstyle\sum_{g\in m} y_{\mathrm{L2}}(p,\mathrm{bl},g) \ge x_{\mathrm{L2}}(p,\mathrm{bl},m) \quad \forall\ p,\mathrm{bl},m \ge 1 \tag{59}$$

$$\mathrm{xe}_{\mathrm{L2}}(\mathrm{bl},g) \ge y_{\mathrm{L2}}(p,\mathrm{bl},g) - y_{\mathrm{L2}}(p,\mathrm{bl},g-1) \quad \forall\ p,\mathrm{bl},g \ge 1 \tag{60}$$

$$\mathrm{xe}_{\mathrm{L2}}(\mathrm{bl},g) \ge y_{\mathrm{L2}}(p,\mathrm{bl},g-1) - y_{\mathrm{L2}}(p,\mathrm{bl},g) \quad \forall\ p,\mathrm{bl},g \ge 1 \tag{61}$$

It is important to note that the production sequence given by $y_{\mathrm{L2}}(p,\mathrm{bl},g)$ provides a lower bound of the number of product transitions in the blenders, but the actual production sequence must still be computed. This is done in Part II of this article where detailed scheduling is integrated to the model.

Note that the second level MILP model does not deal with the blend recipe optimization and do not include the nonlinear terms corresponding to the quality constraints. Thus, all the nonlinearities intrinsic to the blending problem are solved at the first level. Appendix shows that the first and second level models are equivalent to full-space MINLP model.

Corresponding full-space MINLP model

Corresponding full-space MINLP model can be obtained from the second level model by dropping Eq. 20 and adding the following equations from the first level (such equations and its variables are written for all $m \ge 1$):

- The blend recipe computation: Eqs. 9–11; and
- The quality constraints: Eqs. 12a–15.

Although it is possible, we do not include constraints to minimize blend recipe variation to find the optimal solution of the original full-space MINLP model and compare it with the solution of our algorithm.

Special case: full-space MILP model

The second level model can be transformed into a full-space MILP planning model if:

a. Blend recipes are not required to be computed directly in the model. In this case, Eq. 20 is dropped from the second level model, and Eqs. 62 and 63 are added

$$\sum_{i\in I} V_{\mathrm{comp,L2}}(i,p,\mathrm{bl},m) = V_{\mathrm{blend,L2}}(p,\mathrm{bl},m) \quad \forall\ i,(p,\mathrm{bl}) \in \mathrm{BP}, m \ge 1 \tag{62}$$

$$r^{\min}(i,p)\cdot V_{\mathrm{blend,L2}}(p,\mathrm{bl},m) \le V_{\mathrm{comp,L2}}(i,p,\mathrm{bl},m) \le r^{\max}(i,p)\cdot V_{\mathrm{blend,L2}}(p,\mathrm{bl},m) \quad \forall\ i,(p,\mathrm{bl}) \in \mathrm{BP}, m \ge 1 \tag{63}$$

a. All the quality properties are assumed to blend linearly (Eqs. 12a and 12b are added to the second level model but rewritten for all L2-periods) or if the nonlinear equations can be rewritten as linear constraints (e.g., the RVP nonlinear constraint given by Eq. 15 can be substituted by Eq. 64, which is linear)

$$Q_{\mathrm{pr}}^{\min}(p,e)^{1.25}\cdot V_{\mathrm{blend,L2}}(p,m) \le \textstyle\sum_{i\in I} V_{\mathrm{comp,L2}}(i,p,m) \cdot Q_{\mathrm{bc}}(i,e,m)^{1.25} \le Q_{\mathrm{pr}}^{\max}(p,e)^{1.25}\cdot V_{\mathrm{blend,L1}}(p,m) \quad \forall\ e = \mathrm{RVP}, p, m \ge 1 \tag{64}$$

This linear transformation allows us to compare performance of our algorithm with a full-space MILP model. Not all nonlinear quality constraints can be rewritten exactly as linear constraints, and the aim of our algorithm is to handle problems with those nonlinearities.

As blend recipes are not computed directly by the model, it is only possible to minimize variations with respect to a given set of preferred blend recipes. These constraints are not included to find the optimal solution of the original full-space model.

Multiperiod Inventory Pinch Algorithm for Gasoline Blend Planning

The flowchart of the multiperiod inventory pinch (MPIP) algorithm is presented in Figure 6. Although this algorithm is based on the gasoline blend planning problem, it can be applicable to any system or process with similar features. The steps of the algorithm are the following:

Step 1: Construct the cumulative curves (CTD and CATP) and determine the pinch point(s) location.

Step 2: Set iteration counter iter = 1. Divide the planning horizon (at the first level) in the number of L1-periods indicated by the pinch points and the times when the quality values of the blend components change (i.e., $K^{(1)}$).

Step 3: Solve the first level Blend Recipe Optimization model.

- In the first iteration (iter = 1), the volumes to produce of each product in each L1-period k are the minimum amounts required to meet the aggregated demand in each L1-period

$$V_{\mathrm{blend,L1}}(p,k) = \mathrm{Demand}_{\mathrm{pool,L1}}(p,k) + V_{\mathrm{pool,L1}}^{\mathrm{target}}(p,k) - V_{\mathrm{pool,L1}}(p,k-1) \tag{65}$$

where $V_{\mathrm{pool,L1}}^{\mathrm{target}}(p,k)$ is the target inventory. Notice that if no target inventories are set, $V_{\mathrm{pool,L1}}^{\mathrm{target}}(p,k) = V_{\mathrm{pool,L1}}^{\min}(p)$.

- If $V_{\mathrm{blend,L1}}(p,k)$ violates the maximum blend capacity or the minimum blend size threshold constraints, volumes are adjusted by moving the least amount possible of volume to previous L1-periods (i.e., preblending).
- In subsequent iterations (iter > 1), the volumes to produce are defined according to the solution of the second level (see Step 6).
- If any inventory slack variable has a non-zero value at the solution, the problem is infeasible as the availability or quality of blend components is not sufficient to meet the product quality specifications (or to deliver the product within the delivery window). Stop.

Step 4: Solve the second level Blend Planning model.

Step 5: If the inventory slack variables from Step 4 are zero, an optimal blend plan has been found; go to Step 8. Otherwise, continue to Step 6.

Step 6: Subdivide the L1-period with the first infeasibility (see Figure 7).

- The volumes to be blended in each new L1-period are given by the solution of Step 5 plus the positive slacks minus the negative slacks

$$V_{\mathrm{blend,L1}}(p,k) = \sum_{m\in \mathrm{MK}} \left[\sum_{\mathrm{bl}\in \mathrm{BP}} V_{\mathrm{blend,L2}}(p,\mathrm{bl},m) + S_{\mathrm{pool,L2}}^{+}(p,m) - S_{\mathrm{pool,L2}}^{-}(p,m)\right] \tag{66}$$

- If $V_{\mathrm{blend,L1}}(p,k)$ violates the maximum capacity or minimum blend threshold constraints, volumes are adjusted by moving the least amount possible of volume to the previous L1-period or from the next L1-period, respectively.

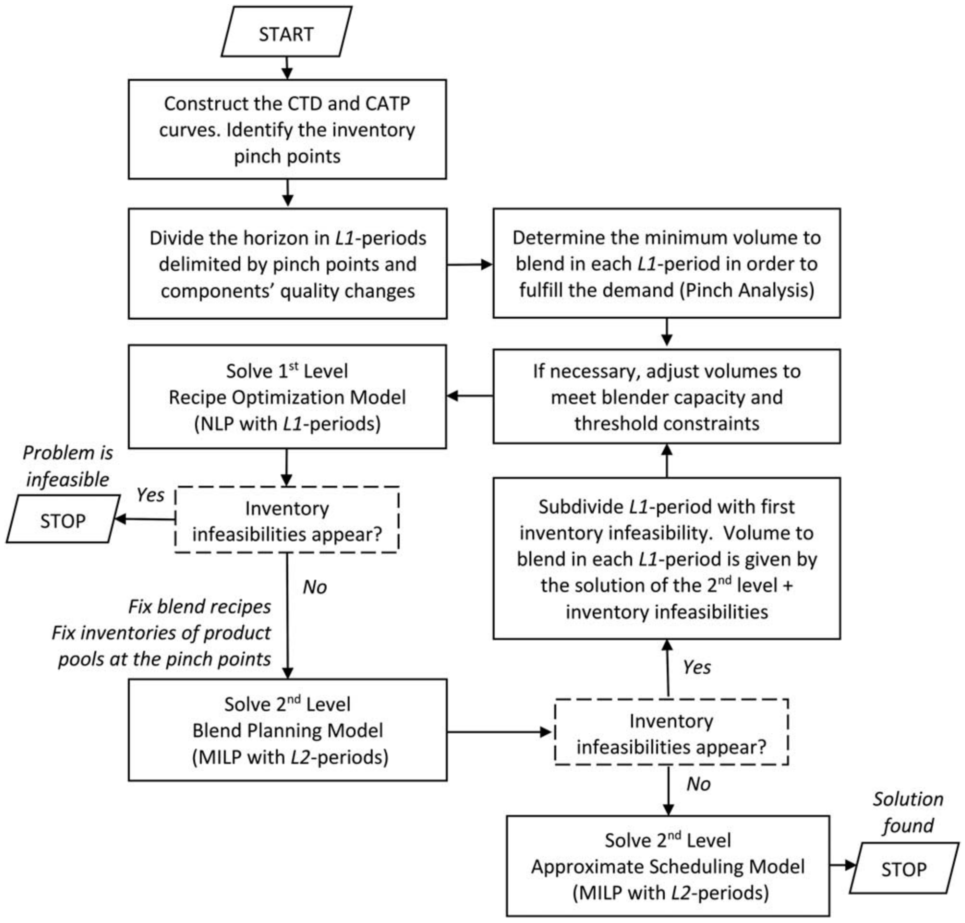

START
Construct the CTD and CATP curves. Identify the inventory pinch points
Divide the horizon in L1-periods delimited by pinch points and components' quality changes
Determine the minimum volume to blend in each L1-period in order to fulfill the demand (Pinch Analysis)
Solve 1st Level Recipe Optimization Model (NLP with L1-periods)
If necessary, adjust volumes to meet blender capacity and threshold constraints
Problem is infeasible
Yes
STOP
Inventory infeasibilities appear?
Subdivide L1-period with first inventory infeasibility. Volume to blend in each L1-period is given by the solution of the 2nd level + inventory infeasibilities
No
Fix blend recipes
Fix inventories of product pools at the pinch points
Yes
Solve 2nd Level Blend Planning Model (MILP with L2-periods)
Inventory infeasibilities appear?
No
Solution found
Solve 2nd Level Approximate Scheduling Model (MILP with L2-periods)
STOP

Figure 6. Flowchart for the MPIP planning algorithm.

Step 7: $K^{(\text{iter}+1)} = K^{(\text{iter})} + 1$. iter = iter + 1. Go back to Step 3.

Step 8: Solve the second level Approximate Scheduling model, to reduce the number of blend instances and product transitions in the blenders and storage tanks. Stop.

Case Studies

The gasoline blending system shown in Figure 2, in configurations with one, two, and three blenders, has been studied. In all cases, the system uses seven blend components (ALK, BUT, HCL, HCN, LCN, LNP, and RFT) and produces three products (grades of gasoline U87, U91, and U93). Each component has their particular storage tank. There are three dedicated product tanks, one per each gasoline grade, and there are three swing tanks which can store any product, but only one at a given time. A planning horizon of 14 1-day L2-periods is used in all case studies ($H = 14$ days, $t_{\text{L2}}(m) = 1$ day for all m). The minimum volume to blend in each blender in each L2-period is $\text{VMIN}_{\text{blend}}(p,\text{bl}) = 30 \times 10^3$ bbl, for all p and bl. The demand orders involve a single product and their delivery time windows are assumed to be 1 day. Eight blend properties are considered: aromatic content (% by volume, ARO), benzene content (% by volume, BEN), olefin content (% by volume, OLF), motor octane number (MON), research octane number (RON), RVP (psi), specific gravity (SPG), and sulfur content (% by volume, SUL). All blend properties are assumed to blend linearly except RVP. Supporting Information of this article contains data describing product specifications, properties, and availabilities of blend components, tankage capacities, and information required for full specification of the test cases. The cumulative curves and pinch points for each profile demand are shown in Figure 8. There is no maximum delivery rate for each order, that is, the maximum delivery rate of the tanks is the maximum limit. For the MILP volume allocation problem, the cost coefficients for component inventory slack variables are set to 1×10^9, and the cost coefficients for product inventory slack variables appear in Supporting Information of this article. Table 1 shows data describing the blenders. At the second level, for all blenders, the penalty for a blend instance is

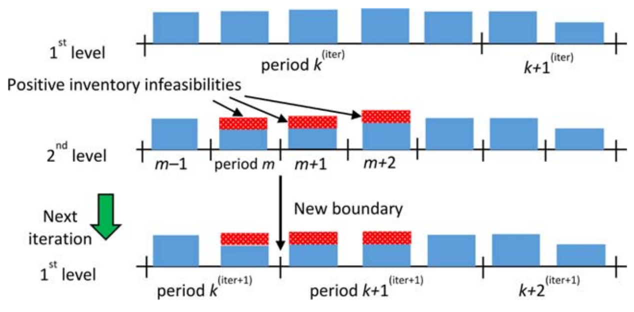


Figure 7. Subdivision of L1-periods according to the solution of the second level model.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

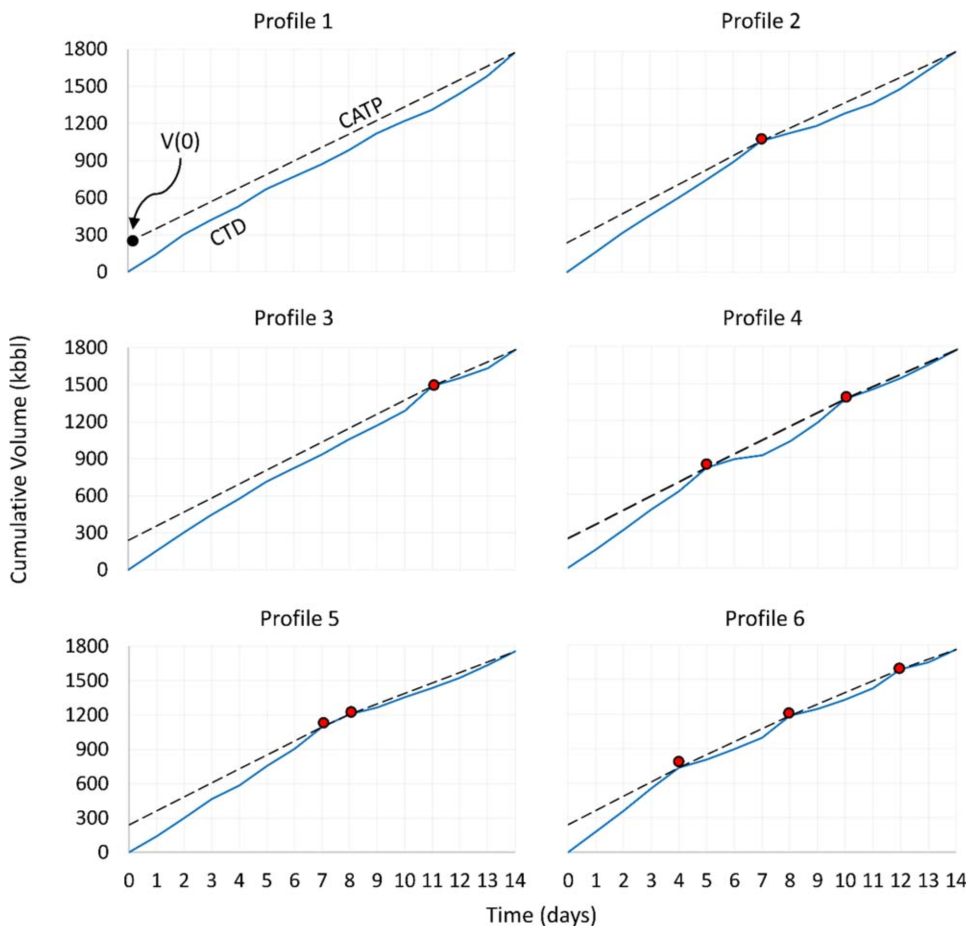


Figure 8. Cumulative curves and pinch points for the six demand profiles.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

20×10^3 \$, and for a product transition in a blender is 7×10^3 \$. Notice that a blend instance is more penalized than a product transition in a blender to use more efficiently the blenders (i.e., at higher production rates). All data are presented in English system of units as it prevails in the refining industry.

All case studies have been computed on a DELL PowerEdge T310 (Intel® Xeon® CPU, 2.40 GHz, and 12 Gb RAM) running Windows Server 2008 R2 OS. GAMS IDE 24.2.1 was used to solve each one of the case studies. The first level NLP model was solved using IPOPT, and the second level model was solved using CPLEX 12.6. To compare the results obtained with the inventory pinch algorithm, the corresponding full-space MINLP and MILP models were solved for both the blend planning and approximate scheduling problems. DICOPT (using IPOPT version 3.8), BARON 9.3.1, ANTIGONE[20] 1.1, and GloMIQO[21,22] 2.3 were the solvers used to solve the full-space MINLP models, and CPLEX 12.6 to solve the full-space MILP model. The mixed-integer quadratically constrained program (MIQCP)

Table 1. Blenders Data

Blender			A	B	C
Allowable products			U87, U91, U93	U91, U93	U87, U91
Minimum volume to blend in each L2-period ($\times 10^3$ bbl)			30	30	30
Maximum blending rate ($\times 10^3$ bbl/h)	Case study	1, 2, 3, 8, 9, 10	7.5	–	–
		4, 5, 6, 11, 12, 13	4.5	3	–
		7, 14	3	2.5	2
Minimum blending rate ($\times 10^3$ bbl/h)		1, 2, 3, 8, 9, 10	5	–	–
		4, 5, 6, 11, 12, 13	3	2	–
		7, 14	2	1.8	1.5
Minimum running time (h)	Product	U87	6	–	6
		U91	6	6	6
		U93	6	6	–
Minimum idle time (h)		U87	1	–	2
		U91	1	1	1
		U93	2	1	–

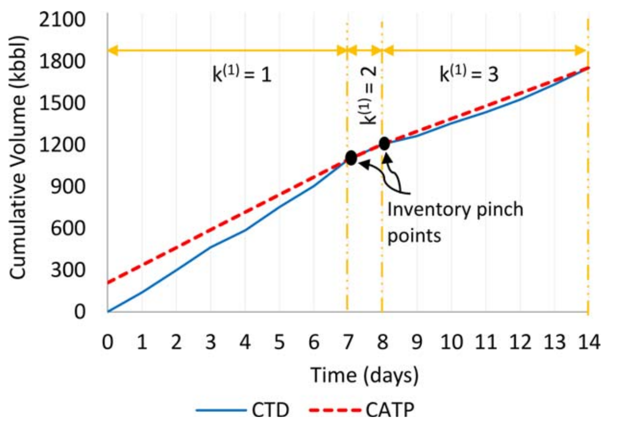


Figure 9. Case study 13—Cumulative curves, inventory pinch points, and L1-periods. Blend planning first iteration.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

required by GloMIQO is the full-space MINLP model described in the Mathematical Models section, but Eq. 64 replaces Eq. 15.

When solving the full-space MINLP model with DICOPT, starting points are important to find an optimal solution, or even a feasible one. In our case studies, the starting point of variable $V_{\text{blend,L2}}(p,\text{bl},m)$, for all p, bl, and m, was set above the minimum blend size threshold (i.e., $\text{VMIN}_{\text{blend}}(p,\text{bl})$).

As blend recipes computed by MINLP model are not exactly the same, in order to compute the number of different recipes in the MINLP solution, two recipes are considered to be different if the absolute change of composition percentage of any component is greater than 1%. As three grades of gasoline are being blended, the total number of different blend recipes for all products is divided by three.

Illustrative example with two blenders and irregular component supply

Case study 13 will be used to illustrate the steps of the MPIP planning algorithm. This case study has demand profile #5, two blenders (A and B), and the supply flow rate of components is irregular along the planning horizon.

Blend Planning: Iteration 1. The corresponding cumulative curves are shown in Figure 9, where it can be seen that there are two inventory pinch points, one at the end of Day 7, and the other at the end of Day 8. Hence, three L1-periods are required initially, $k^{(1)} = 1$, starts at the beginning of Day 1 until the first pinch point, $k^{(1)} = 2$ corresponds to Day 8, and $k^{(1)} = 3$ goes from the beginning of Day 9 to the end of the planning horizon. In the first iteration of the algorithm, the minimum production to meet the aggregated demand in each L1-period is given by Eq. 65. The blend recipes computed by the first level model are given in Table 2, and the blend cost is Blend $\text{Cost}_{\text{L1}} = 37{,}784.52 \times 10^3$ \$.

Using the blend recipes from the first level, the second level planning model is solved. Figure 10 shows the blend plan for each blender, but it contains inventory infeasibilities. The positive slack variables appear in L2-periods $m = 4$ (12.77×10^3 bbl) and $m = 5$ (14×10^3 bbl), both for product U91. The negative slack variables appears in period $m = 14$ for product U87 (10.42×10^3 bbl). From Figure 10 can be seen that blender A and B are working at full capacity during period $m = 5$; thus, the slack in period $m = 5$ must be preblended (see rule 6.a of the algorithm). L2-Periods before $m = 5$ have enough blending capacity

Table 2. Case Study 13—Optimal Blend Recipes (First Level)

	First Iteration				Second Iteration			
Blend Comp.	L1-Period	U87	U91	U93	L1-Period	U87	U91	U93
ALK	$\mathbf{k^{(1)} = 1}$	0.1747	0.245	0.1414	$\mathbf{k^{(2)} = 1}$	0.1636	0.2422	0.1874
BUT		0.0261	0.0368	0.0436		0.0262	0.0355	0.0435
HCL		0.0241	0.0374	0.0327		0.027	0.0309	0.029
HCN		0.0456	0.0615	0.0523		0.0366	0.04	0.0394
LCN		0.2765	0.1698	0.1168		0.2458	0.1978	0.1293
LNP		0.1708	0.0784	0.0368		0.1842	0.0854	0.031
RFT		0.2823	0.3711	0.5765		0.3165	0.3682	0.5404
ALK					$\mathbf{k^{(2)} = 2}$	0.1543	0.2486	0.1824
BUT						0.0263	0.0373	0.0442
HCL						0.0239	0.0465	0.0314
HCN						0.058	0.0793	0.0529
LCN						0.2791	0.1648	0.1174
LNP						0.169	0.0653	0.0276
RFT						0.2895	0.3583	0.5442
ALK	$\mathbf{k^{(1)} = 2}$	0.1537	0.2449	0.1788	$\mathbf{k^{(2)} = 3}$	0.1535	0.2434	0.1803
BUT		0.0252	0.036	0.0434		0.0252	0.036	0.0434
HCL		0.0286	0.038	0.0278		0.0289	0.0382	0.0277
HCN		0.0522	0.0606	0.0443		0.0533	0.0616	0.0443
LCN		0.3058	0.2003	0.1369		0.3056	0.1996	0.1372
LNP		0.1619	0.0702	0.0293		0.1613	0.0702	0.0289
RFT		0.2726	0.35	0.5395		0.2722	0.351	0.5381
ALK	$\mathbf{k^{(1)} = 3}$	0.0152	0.1363	0.09	$\mathbf{k^{(2)} = 4}$	0.0146	0.1347	0.0924
BUT		0.0197	0.0349	0.0416		0.0197	0.0351	0.0414
HCL		0	0	0		0	0	0
HCN		0.0011	0.0021	0.0017		0.0011	0.0021	0.0017
LCN		0.427	0.1617	0.1496		0.4255	0.1566	0.1564
LNP		0.2016	0.1554	0.0825		0.2022	0.1571	0.0801
RFT		0.3355	0.5096	0.6346		0.3371	0.5144	0.628

The significance of the bold characters is that they represent the L1-periods, where k represents the elements of the set K = (L1-periods). The superscript inside the parenthesis represent the iteration number.

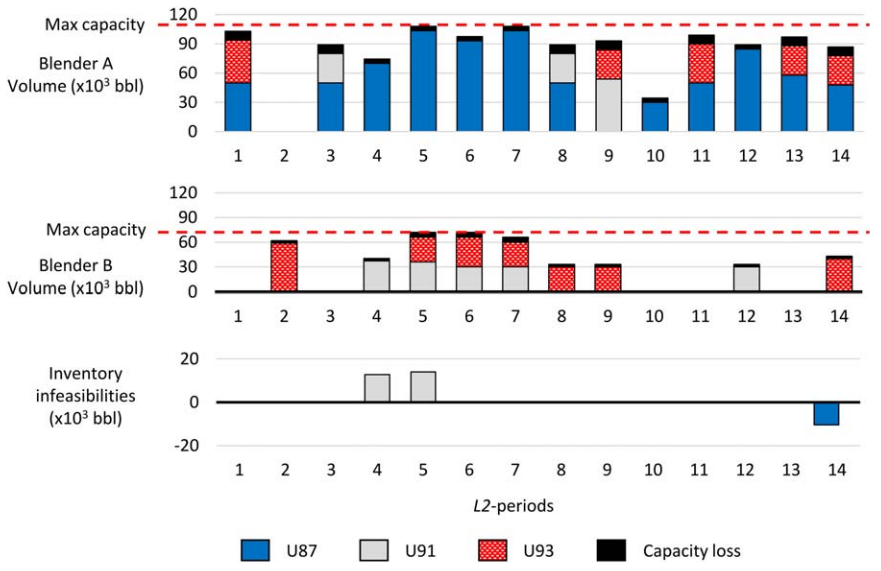


Figure 10. Case study 13—Blend plan with inventory infeasibilities. Blend planning first iteration.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

available to blend both infeasibilities. Therefore, the new boundary in the first level must be placed at the end of day 4.

The new period boundaries at the first level are shown in Figure 11; however, this time the CATP curve is replaced with the actual CTP curve that is constructed by aggregating the total production [i.e., $\Sigma_{p,\mathrm{bl}}V_{\mathrm{blend,L2}}(p,\mathrm{bl},m)$] over time. When the CTP curve is below the CTD, there are inventory infeasibilities. This is easier to see on the grand composite curve.

Blend Planning: Iteration 2. In this iteration, the production targets for each L1-period are computed using Eq. 66 and rule 6.a of the algorithm. The blend recipes computed by the first level model are given in Table 2, and the blend cost is Blend $\mathrm{Cost}_{\mathrm{L1}} = 37{,}784.52 \times 10^3$ \$, the same cost as in the first iteration.

Using the blend recipes from the second iteration, the second level planning model is solved. An optimal blend plan is found with a cost equal to Blend $\mathrm{Cost}_{\mathrm{L2}} = 37{,}784.52 \times 10^3$ \$, the same as that of the first level. Because there are no inventory infeasibilities this time, we proceed to solve the second level approximate scheduling model.

Approximate Scheduling. The approximate schedule with reduced number of blend instances and product transitions in the blenders is shown in Figure 12 and its total cost is $Z_{\mathrm{L2}} = 38{,}522.52 \times 10^3$ \$. Idle times between blend runs are not shown since the actual production schedule is not determined at this level. It is also determined that in this case no product changeovers are required in the swing tanks.

The component and product inventory profiles corresponding to the second level approximate scheduling (Figure 13) are at the allowed minimums at the inventory pinch points (i.e., at the end of the seventh and eighth day). Additionally, it is observed that the inventory levels of some blend components are at the minimum allowed at some points in the planning horizon. In this particular case study, this is observed at the inventory pinch points. In general, these "pinch points on the components' inventory profiles" will appear at least for one blend component at the end of the planning horizon and at least at one inventory pinch point. There are two reasons for the appearance of these pinch points on the components' inventory profiles: (a) cheaper components are used as much as possible, and (b) components with the necessary qualities to produce products under specification are scarce. If the "components' pinch" occurs because of reason "(a)," trying to blend more volume (i.e., increased production) before a components' pinch might result in a blend cost increase (i.e., more expensive materials are used since cheaper components are not available). If this

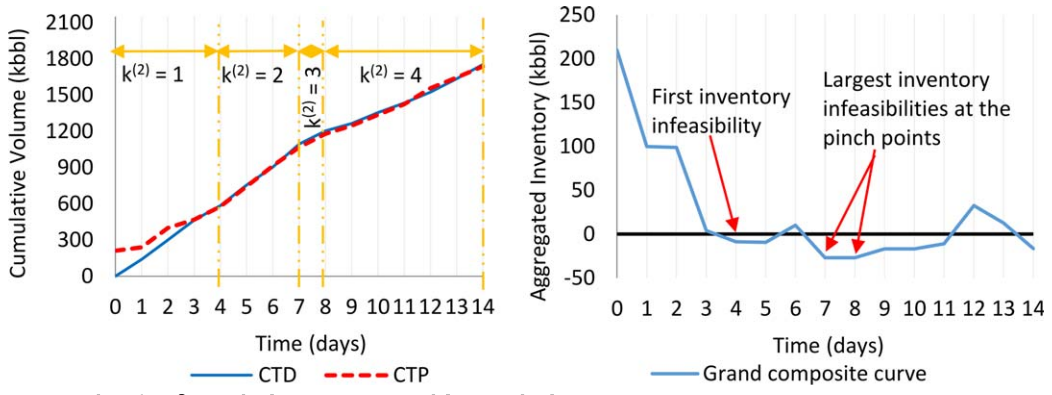


Figure 11. Case study 13—Cumulative curves and L1-periods.

Blend planning second iteration. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

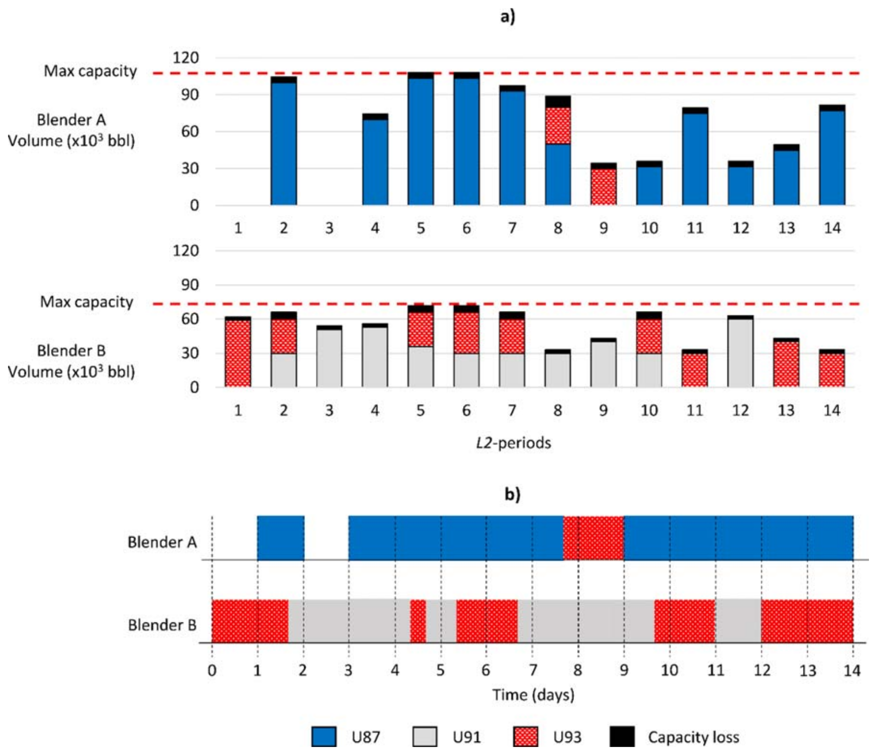


Figure 12. Case study 13—(a) Blend plan and (b) production sequence.

Approximate scheduling. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

type of pinch appears because of reason "(b)," then trying to increase production before the components' pinch is not possible (first level will present non-zero slack variables) since the blend cannot reach the quality specifications. In this illustrative example, the components' pinch occurs because of reason "(b)" there is not enough ALK and RFT available to blend more before Day 8. ALK and RFT have the highest RON and MON values from among the blend components; other components have RON and MON values smaller than the minimum spec, or if they have a greater RON and MON value, they have RVP and SPG value above the maximum spec. Therefore, no more than the production targets given in can be blended before Day 7 or Day 8 in Case study 13.

Results and Discussion

To evaluate the performance of the MPIP algorithm, we compared the solutions from our algorithm with those provided by the full-space models and the execution times required to obtain them.

Blend planning

All problems were solved to an optimality gap less than or equal to 0.001%. The MPIP algorithm and the full-space models found the same optimal objective function value (see Table 3). The solution times of the MPIP planning algorithm (using IPOPT for the NLP model and CPLEX for the MILP model) are much smaller than those required by DICOPT to solve the corresponding MINLP model (almost two orders of magnitude lower). We chose to compare with DICOPT solver because it provided the smallest execution times when compared with the other selected MINLP solvers. We know that the solutions found by DICOPT and by MPIP algorithm are globally optimal, as they are equal to the solution of full-space MILP (which in this special case, via transformations, is able to solve our test problems as linear MILP models). It is interesting to note that our algorithm leads to execution times which are about the same as the times required to solve the equivalent MILP model. In addition, our algorithm requires a small number of iterations (in most of our case studies only one iteration is required). Moreover, our approach leads to a smaller number of different blend

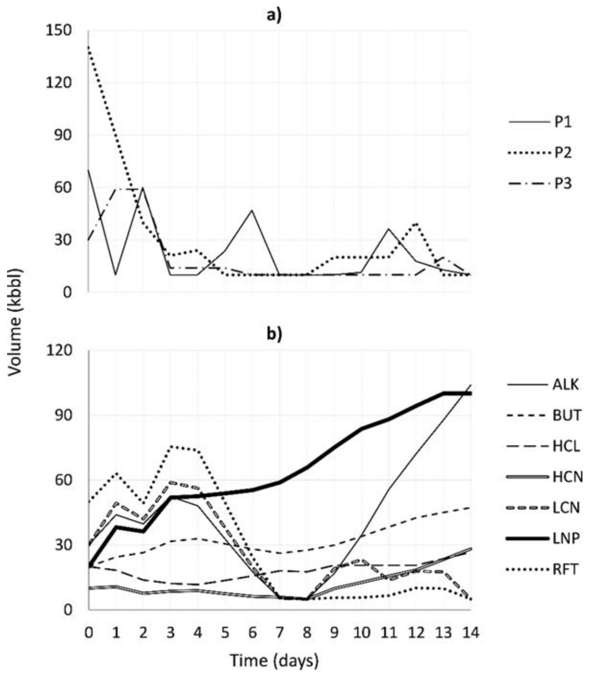


Figure 13. Case study 13—(a) Product and (b) component inventory profiles.

Approximate scheduling.

Table 3. Blend Planning—Comparison Between Solutions of Full-Space Models and the MPIP Planning Algorithm

Case Study ID	Demand Profile	# Pinch Points	# Blenders	Component Supply	Blend Cost[a] ($\times 10^3$ \$)	Full-Space MINLP (DICOPT) # Recipes[b]	Full-Space MINLP (DICOPT) Time[c] (s)	Full-Space MILP (CPLEX) # Recipes[b]	Full-Space MILP (CPLEX) Time[c] (s)	MPIP Algorithm (NLP: IPOPT, MILP: CPLEX) # Recipes[d]	MPIP Algorithm (NLP: IPOPT, MILP: CPLEX) Time[c] (s)	MPIP Algorithm (NLP: IPOPT, MILP: CPLEX) # Iterations
1	1	0	1	Constant	37,542.5	6	152.2	6	0.822	1	0.625	1
2	2	1	1	Constant	38,309.8	6	240.5	5	0.752	2	0.778	1
3	3	1	1	Constant	37,991.1	7	227.9	7	1.068	2	0.969	1
4	3	1	2	Constant	37,991.1	8	192.9	7	1.754	2	0.938	1
5	4	2	2	Constant	37,681.1	8	312.6	8	0.733	3	0.960	1
6	5	2	2	Constant	37,324.3	9	258.5	8	0.969	6	10.634	4
7	6	3	3	Constant	37,377.5	10	366.4	10	1.524	4	1.277	1
8	1	0	1	Irregular	37,943.4	6	119.7	6	0.829	1	0.505	1
9	2	1	1	Irregular	38,754.2	7	176.8	6	0.858	2	0.766	1
10	3	1	1	Irregular	38,405.2	6	165.2	7	0.547	2	0.768	1
11	3	1	2	Irregular	38,405.2	8	213.3	8	1.001	2	0.730	1
12	4	2	2	Irregular	38,073.4	9	501.3	8	0.768	3	0.750	1
13	5	2	2	Irregular	37,784.5	9	281.7	8	0.973	4	1.836	2
14	6	3	3	Irregular	37,796.4	11	842.8	10	1.208	4	0.906	1

[a]All approaches computed the same blend cost.
[b]Blend recipe is considered different if the absolute change of composition percentage of any component is greater than 1%.
[c]CPU time.
[d]Repeated blend recipes are exactly the same.

Table 4. Approximate Scheduling—Comparison between Solutions of Full-Space Models and the MPIP Planning Algorithm

					Global Optimum Full-Space MILP (CPLEX)				Full-Space MINLP ANTIGONE	Full-Space MIQCP GloMIQO	MPIP Algorithm (IPOPT + CPLEX)	
					Stop at 10^{-5} Gap or 10,800 s				All Runs Stopped at 10,800 s with Gap Greater than 0.2%		Stop at 10^{-5} gap or 10,800 s	
Case Study ID	Demand Profile ID	# Pinch Points	# Blenders	Supply Profile	Total Cost ($\times 10^3$ \$)	Gap (%)	Best Lower Bound ($\times 10^3$ \$)	CPU Time (s)	Total Cost ($\times 10^3$ \$)	Total Cost ($\times 10^3$ \$)	Total Cost ($\times 10^3$ \$)	CPU Time (s)
1	1	0	1	C	38,004	0.17	37,939	10,800	38,109	38,096	38,007	10,800
2	2	1	1	C	38,782	0.00	38,782	3,264	38,891	38,827	38,896	83
3	3	1	1	C	38,503	0.02	38,497	10,800	38,599	38,558	38,543	76
4	3	1	2	C	38,529	0.11	38,486	10,800	38,703	38,598	38,563	10,800
5	4	2	2	C	38,253	0.16	38,193	10,800	38,401	38,358	38,306	817
6	5	2	2	C	37,916	0.09	37,883	10,800	38,060	37,951	37,981	66
7	6	3	3	C	38,076	0.14	38,024	10,800	38,323	38,249	38,157	3,183
8	1	0	1	I	38,387	0.12	38,342	10,800	38,498	38,512	38,423	647
9	2	1	1	I	39,220	0.00	39,220	1,995	39,301	39,248	39,420	17
10	3	1	1	I	38,917	0.09	38,882	10,800	38,998	38,992	39,004	21
11	3	1	2	I	38,951	0.17	38,884	10,800	39,144	39,042	39,023	313
12	4	2	2	I	38,621	0.12	38,577	10,800	38,788	38,745	38,698	649
13	5	2	2	I	38,370	0.06	38,346	10,800	38,562	38,533	38,496	15
14	6	3	3	I	38,497	0.13	38,447	10,800	38,738	38,710	38,596	9,017
Average difference with respect to best lower bound ($\times 10^3$ \$)									186	137	115	–

Table 5. Approximate Scheduling—Results for MPIP Planning Algorithm

	MPIP Algorithm (IPOPT + CPLEX)			
Case Study ID	Total Cost (×103 $)	Blend Cost (×103 $)	Switching Cost (×103 $)	CPU Time (s)
1	38,014.5	37,542.5	472.00	2809.3
2	38,901.8	38,309.8	592.00	18.5
3	38,550.1	37,991.1	559.00	22.5
4	38,569.1	37,991.1	578.00	1871.8
5	38,312.6	37,680.6	632.00	72.4
6	37,988.3	37,324.3	664.00	26.2
7	38,156.5	37,377.5	779.00	118.7
8	38,422.9	37,943.4	479.50	140.2
9	39,427.2	38,754.2	673.00	8.7
10	39,004.2	38,405.2	599.00	14.3
11	39,030.2	38,405.2	625.00	119.6
12	38,705.4	38,073.4	632.00	83.4
13	38,522.5	37,784.5	738.00	7.5
14	38,603.4	37,796.4	807.00	1104.1
Average difference of the total cost with respect to best lower bound ($\times 10^3$ $)				122

Stopping criteria: 0.1% optimality gap or 10,800 s.

recipes per product compared to the solution of the corresponding full-space models (see Table 3). Significant reduction is seen in the cases with few number of pinch points and blenders; this is because the full-space models can have different recipes for the same product for different blenders in the same time period. In addition, repeated recipes from the solution of the full-space models may or may not be used in adjacent periods; thus, the number of recipe switching is equal or greater than the number of different recipes.

Approximate scheduling

The approximate schedules computed by the MPIP planning algorithm were compared with the full-space models. The termination criteria was 10,800 s (CPU time) or an optimality gap less than or equal to 0.001%. DICOPT was not used in this evaluation since it requires to solve the MILP subproblems to optimality to guarantee a local optimum; however, it was not possible to achieve that within the maximum allocated time of 3 h. Results from BARON are not shown because it was able to find feasible integer solutions only for some of our case studies and with optimality gaps greater than 3%.

Table 4 shows the results obtained by CPLEX, ANTIGONE, GloMIQO, and the MPIP planning algorithm. The results from full-space MILP were used to determine the global optimum and the best lower bound. In the majority of our case studies, MPIP algorithm computed better solutions than ANTIGONE and GloMIQO solvers and in much shorter execution times. All ANTIGONE and GloMIQO runs stopped at 10,800 s with optimality gaps larger than 0.2%. Only in two test cases (no. 1 and no. 4) MPIP algorithm stopped at 10,800 s time limit. In both of these cases, the results from MPIP are slightly better than the results computed by GloMIQO and ANTIGONE. In test case no. 9, MPIP algorithm converged after 17 s, but the solution is 0.4% higher than GloMIQO.

Although the MILP model at the second level achieves optimality gaps smaller than or equal to 0.001%, it is noted that the solutions of the MPIP planning algorithm are higher than the best integer solution given by the full-space MILP. The reason for this difference is that the blend recipes define the feasible region at the second level and they are only being redefined to account for inventory feasibility, but there is no feedback that would cause the computation of new recipes that will minimize the number of product transitions in the blenders and swing tanks. Nevertheless, the difference between the MPIP algorithm solution and the true optimum will only be significant when the penalty for such transitions is much greater than the unit costs of the blend components. For our case studies, the average relative difference between the MPIP solution and the best lower bound is less than 0.33%. As another test of MPIP algorithm performance, we solved all test cases with 0.1% optimality gap at the second level. The execution times are even shorter and the function values are on average still closer to the best lower bound than those computed by ANTIGONE and GloMIQO (compare data in Tables 4 and 5).

Table 6 shows the number of equations, continuous variables, discrete variables, and non-zero elements in the full-

Table 6. Model Size Comparison

Model	# Equations	# Continuous Variables	# Discrete Variables	# Non-Zeros	# Nonlinear Terms
Full-space MINLP model (one blender)	5928	2753	609	16,646	1176
Full-space MINLP model (two blenders)	7906	3655	837	23,455	1960
Full-space MILP model (one blender)	5550	2375	609	15,554	0
Full-space MILP model (two blenders)	7600	3306	837	23,052	0
Firstlevel NLP model (MPIP algorithm, two L1-periods)	326	162	0	955	42
First level NLP model (MPIP algorithm, three L1-periods)	473	237	0	1416	63
Second level MILP model (MPIP algorithm, one blender, two fixed recipes)	4541	2375	609	11,017	0
Second level MILP model (MPIP algorithm, two blenders, two fixed recipes)	6003	3306	837	15,575	0

space models and our decomposition approach for blend planning case studies. Nonlinear aspects of the blending model are solved at the first level, where number of equations and variables are significantly smaller than in the full-space MINLP model. The second level model MILP is approximately the same size as the full-space models. Separation of the nonlinear blend optimization from linear model-based production planning enables MPIP algorithm to optimize blend plans or approximate blend schedules much faster than the corresponding full-space MINLP models.

Conclusions

We have presented a new inventory pinch based, two-level decomposition approach which (1) incorporates nonlinear blending models into integrated planning and approximate scheduling of gasoline blends, (2) includes blender switching and swing tankage management, (3) computes optimal blend plans with significantly smaller number of distinct blend recipes than the fine grid multiperiod MINLP models, (4) computes approximate schedules with similar or lower cost than those computed by global MINLP solvers used in this study, and (5) achieves much shorter execution times.

Rapid computation of optimal blend plans is accomplished by decomposing the blend planning optimization model into two levels: the first level handles the constraints related to operating conditions (e.g., quality constraints and blend recipe computation) by solving a NLP model, and the second level determines the actual production plan and allocation of swing tankage using optimal blend recipes from the first level in a MILP model, subject to availability of blend components, inventory storage limits, and minimum blend threshold constraints. These blend plans are then used to compute approximate blend schedules, which minimize total number of blend switches and product changeovers in the swing tanks.

Case studies with one, two, and three blenders have been presented. Our case studies have been constructed in such a way that after some transformations they can also be solved as an MILP model. That has enabled us to compute rapidly the global optimum and confirm that the blend plans computed by MPIP are also globally optimal. In 10 out of 14 case studies, MPIP algorithm finds a better solution for the approximate scheduling problem than ANTIGONE or GloMIQO and typically in much shorter execution times.

Our results show that the solutions computed by the MPIP planning algorithm are exactly the same as the optimum solutions computed by the corresponding full-space MINLP and MILP models when the objective function of the second level contains only variables that are aggregated at the first level (e.g., volumes to blend); and close-to-optimum solutions when the objective function of the second level contains variables that are not aggregated at the first level (e.g., switching variables) with penalty coefficients similar to the unit costs of blend components.

Part II of this article deals with detailed scheduling based on the approximate blend schedule computed at the second level of the MPIP algorithm.

Acknowledgments

Support by Ontario Research Foundation and McMaster Advanced Control Consortium is gratefully acknowledged. The authors thank Dr. Xiang Li (Queens University) for suggesting that they transform the example problems into MILP models in order to verify global optimality. Moreover, the authors thank GAMS Corporation for granting them a license to use ANTIGONE and GloMIQO in their tests.

Notation

Subscripts

bc =	refers to a variable or parameter related to the blend component tanks
blend =	refers to a variable or parameter related to the blenders
comp =	refers to a variable or parameter related to the transfer of volume between component tanks and blenders
L1 =	refers to a variable or parameter of the first level
L2 =	refers to a variable or parameter of the second level
order =	refers to a variable or parameter related to the product orders
pool =	refers to a variable or parameter related to the product pools
pr =	refers to a variable or parameter related to the individual product tanks
trans =	refers to a variable or parameter related to the transfer of volume between blenders and product tanks or pools

Superscripts

max =	refers to a maximum value that a variable may have
min =	refers to a minimum value that a variable may have if different from zero
start =	refers to the initial value at the beginning of the planning horizon that a variable may have
target =	refers to a target value for a variable

Parameters

$\text{Cost}_{\text{bc}}(i)$ =	cost of blend component i
$D_{\text{pr}}^{\max}(j)$ =	maximum delivery rate of tank j
$\text{Demand}(o)$ =	demand of order o for the complete scheduling horizon
$\text{Demand}_{\text{pool,L1}}(p,k)$ =	aggregated demand of product p in L1-period k
$\text{Demand}_{\text{pr,L2}}(p,m)$ =	demand of product p in L2-period m
$\text{dt}_{\text{L2}}(o,m)$ =	time available to deliver order o during L2-period m
$F_{\text{bc}}(i,\alpha)$ =	supply flow rate of blend component i during time interval α
$F_{\text{blend}}^{\max}(\text{bl})$ =	maximum blending rate of blender bl
$F_{\text{blend}}^{\min}(\text{bl})$ =	minimum blending rate of blender bl
H =	length of the planning horizon
$\text{it}_{\text{blend}}^{\min}(p,\text{bl})$ =	minimum idle time required by blender bl before processing product p
$\text{np}(\text{bl})$ =	number of products that can be produced in a L2-*period* in blender bl
$\text{Penalty}_{\text{bc,L1}}$ =	penalty for the inventory slack variables of blend component tanks
$\text{Penalty}_{\text{pool,L1}}$ =	penalty for the inventory slack variables of product pools
$\text{Penalty}_{\text{bc,L2}}(m)$ =	penalty for the inventory slack variables of component i in L2-period m
$\text{Penalty}_{\text{pool,L2}}(m)$ =	penalty for the inventory slack variables of product pool p in L2-period m
$\text{Penalty}_{\text{pr,L2}}(m)$ =	penalty for the inventory slack variables of product tank j in L2-period m
$\text{PenaltyBR}_{\text{L2}}(\text{bl})$ =	penalty for a blend instance processed in blender bl during a L2-period m
$\text{PenaltyBS}_{\text{L2}}$ =	penalty for a product transition in a blender at the second level
$\text{PenaltyTS}(j)$ =	penalty for a product transition in a swing tank at the second level
$Q_{\text{bc}}(i,e,k)$ =	quality e of blend component i during L1-period k
$Q_{\text{pr}}^{\max}(p,e)$ =	maximum requirement of quality e in grade p
$Q_{\text{pr}}^{\min}(p,e)$ =	minimum requirement of quality e in grade p
$r^{\max}(i,p)$ =	maximum composition specification of product p regarding blend component i
$r^{\min}(i,p)$ =	minimum composition specification of product p regarding blend component i
$t_{\text{bc,L1}}(\alpha,k)$ =	duration of time interval where blend component supply flow rate α occurs in L1-period k
$t_{\text{bc,L2}}(\alpha,m)$ =	duration of time interval where blend component supply flow rate α occurs in L2-period m
$t_{\text{blend}}^{\min}(p,\text{bl})$ =	minimum running time required by blender bl when processing product p

$t_{L1}(k)$ =	duration of L1-period k
$t_{L2}(m)$ =	duration of L2-period m
$u^{start}(j,p)$ =	product p stored in tank j at the beginning of the planning horizon
$uof_{L2}(o,m)$ =	parameter which value is 1 if order o may be delivered during L2-period m (i.e., delivery window of order o spans L2-period m) and 0 otherwise
$V_{bc}^{max}(i)$ =	maximum holdup of tank with blend component i
$V_{bc}^{min}(i)$ =	minimum holdup of tank with blend component i
$V_{bc}^{start}(j,p)$ =	volume of blend component i stored at the beginning of the planning horizon
$V_{blend,L1}(p,k)$ =	volume of product p to blend in L1-period k
$V_{pool}^{max}(p)$ =	maximum holdup of product pool p
$V_{pool}^{min}(p)$ =	minimum holdup of product pool p
$V_{pool}^{start}(p)$ =	total volume of product p stored at the beginning of the planning horizon
$V_{pool,L1}^{target}(p,k)$ =	target inventory for product pool p in L1-period k
$V_{pr}^{max}(j)$ =	maximum holdup of tank j
$V_{pr}^{min}(j)$ =	minimum holdup of tank j
$V_{pr}^{start}(j)$ =	volume stored in tank j at the beginning of the planning horizon
$VMIN_{blend}(p,bl)$ =	minimum volume allowed to blend of product p in blender bl during each L2-period

Binary variables

$u_{L2}(j,p,m)$ =	binary variable that indicates if tank j is storing product p in L2-period m
$ue_{L2}(j,m)$ =	binary variable that indicates if there is a product transition in tank j at the beginning of L2-period m
$v_{L2}(j,p,bl,m)$ =	binary variable that indicates if tank j is receiving product p from blender bl in L2-period m
$y_{L2}(p,bl,g)$ =	binary variable that indicates if product p occupies slot g in order to estimate the production sequence of blender bl at the second level
$x_{L2}(p,bl,m)$ =	binary variable that indicates if product p is processed in blender bl in L2-period m

Continuous variables

Blend =	$Cost_{L1}$ = total blend cost at the first level
Blend =	$Cost_{L2}$ = total blend cost at the second level
$Deliver_{pr,L2}(j,m)$ =	volume of tank j shipped/lifted at the end of L2-period m
$Deliver_{pool,L2}(p,m)$ =	volume of product p shipped/lifted at the end of L2-period m
$of_{L2}(o,m)$ =	fraction of order o to be delivered during L2-period m
$Q_{pr,L1}(p,e,k)$ =	quality level of property e of product p in L1-period k
$r(i,p,k)$ =	volume of blend component i into product p in L1-period k (it becomes a parameter in the second level model)
$S_{bc,L1}^{+}(i,k)$ =	positive inventory slack variable of blend component i at L1-period k
$S_{bc,L1}^{-}(i,k)$ =	negative inventory slack variable of blend component i at L1-period k
$S_{bc,L2}^{+}(i,m)$ =	positive inventory slack variable of blend component i at L2-period m
$S_{bc,L2}^{-}(i,m)$ =	negative inventory slack variable of blend component i at L2-period m
$S_{pool,L1}^{+}(p,k)$ =	positive inventory slack variable of product pool p at L1-period k
$S_{pool,L1}^{-}(p,k)$ =	negative inventory slack variable of product pool p at L1-period k
$S_{pool,L2}^{+}(p,m)$ =	positive inventory slack variable of product pool p at L2-period m
$S_{pool,L2}^{-}(p,m)$ =	negative inventory slack variable of product pool p at L2-period m
$S_{pr,L1}^{+}(i,k)$ =	positive inventory slack variable of product tank j at L1-period k
$S_{pr,L1}^{-}(i,k)$ =	negative inventory slack variable of product tank j at L1-period k
$S_{pr,L2}^{+}(i,m)$ =	positive inventory slack variable of product tank j at L2-period m
$S_{pr,L2}^{-}(i,m)$ =	negative inventory slack variable of product tank j at L2-period m
$t_{blend,L2}(p,bl,m)$ =	estimated time to process product p in blender bl in L2-period m
$V_{bc,L1}(i,k)$ =	volume stored in component tank i at the end of L1-period k
$V_{bc,L2}(i,m)$ =	volume stored in component tank i at the end of L2-period m
$V_{blend,L2}(p,bl,m)$ =	volume of product p to process in blender bl in L2-period m
$V_{comp,L1}(i,p,k)$ =	volume of blend component i into product p in L1-period k
$V_{comp,L2}(i,p,bl,m)$ =	volume of blend component i into product p in blender bl in L2-period m
$V_{pool,L1}(p,k)$ =	volume stored in product pool p at the end of L1-period k
$V_{pool,L2}(p,m)$ =	volume stored in product pool p at the end of L2-period m
$V_{pr,L1}(j,k)$ =	volume stored in product tank j at the end of L1-period k
$V_{pr,L2}(j,m)$ =	volume stored in product tank j at the end of L2-period m
$V_{trans,L2}(j,p,bl,m)$ =	volume transferred of product p from blender bl to tank j in L2-period m
$xe_{L2}(p,bl,m)$ =	0–1 continuous variable that indicates if a state transition has occurred in blender bl at the beginning of L2-period m
Z_{L1} =	objective function value at the first level
Z_{L2} =	total cost at the second level
Z_{L2}^{feas} =	objective function value at the second level, blend planning
Z_{L2}^{opt} =	objective function value at the second level, approximate scheduling

Literature Cited

1. Shah NK, Li Z, Ierapetritou MG. Petroleum refining operations: key issues, advances, and opportunities. *Ind Eng Chem Res.* 2011;50:1161–1170.
2. Jia Z, Ierapetritou M. Mixed-integer linear programming model for gasoline blending and distribution scheduling. *Ind Eng Chem Res.* 2003;42:825–835.
3. Sing A, Forbes JF, Vermeer PJ, Woo SS. Model-based real-time optimization of automotive gasoline blending operations. *J Process Control.* 2000;10:43–58.
4. Kelly JD. Formulating production planning models. *Chem Eng Prog.* 2004;100:43–50.
5. Li J, Karimi IA, Srinivasan R. Recipe determination and scheduling of gasoline blending operations. *AIChE J.* 2010;56:441–465.
6. Li J, Karimi IA. Scheduling gasoline blending operations from recipe determination to shipping using unit slots. *Ind Eng Chem Res.* 2011;50:9156–9174.
7. Mendez CA, Grossman IE, Harjunkoski I, Kabore P. A simultaneous optimization approach for off-line blending and scheduling of oil refinery operations. *Comput Chem Eng.* 2006;30:614–634.
8. Maravelias C, Sung C. Integration of production planning and scheduling: overview, challenges and opportunities. *Comput Chem Eng.* 2009;33:1919–1930.
9. Bitran GR, Hax AC. On the design of hierarchical production planning systems. *Decis Sci.* 1977;8(1):28–55.
10. Nam S, Logendran R. Aggregate production planning—a survey of models and methodologies. *Eur J Oper Res.* 1992;61:255–272.
11. Axsater S, Jonsson H. Aggregation and disaggregation in production planning. *Eur J Oper Res.* 1984;17:338–350.
12. Verderame PM, Floudas CA. Operational planning framework for multisite production and distribution networks. *Comput Chem Eng.* 2009;33:1036–1050.
13. Timpe CH, Kallrath J. Optimal operational planning in large multisite production networks. *Eur J Oper Res.* 2000;126:422–435.
14. Thakral A, Mahalec V. Composite planning and scheduling algorithm addressing intra-period infeasibilities of gasoline blend planning models. *Can J Chem Eng.* 2013;91:1244–1255.
15. Singhvi A, Shenoy UV. Aggregate planning in supply chains by pinch analysis. *Chem Eng Res Des.* 2002;80:597–605.
16. Singhvi A, Madhavan KP, Shenoy UV. Pinch analysis for aggregate production planning in supply chains. *Comput Chem Eng.* 2004;28:993–999.
17. Ludwig J, Treitz M, Rentz O, Geldermann J. Production planning by pinch analysis for biomass use in dynamic and seasonal markets. *Int J Prod Res.* 2009;47(8):2079–2090.

18. Foo DCY, Ooi MBL, Tan RR, Tan JS. A heuristic-based algebraic targeting technique for aggregate planning in supply chains. *Comput Chem Eng.* 2008;32:2217–2232.
19. Castillo PAC, Kelly JD, Mahalec V. Inventory pinch algorithm for gasoline blend planning. *AIChE J.* 2013;59(10):3748–3766.
20. Misener R, Floudas CA. ANTIGONE: algorithms for continuous/integer global optimization of nonlinear equations. *J Glob Optim.* In press.
21. Misener R, Floudas CA. Global optimization of mixed-integer quadratically-constrained quadratic programs (MIQCP) through piecewise-linear and edge-concave relaxations. *Math Program.* 2012;136:155–182.
22. Misener R, Floudas CA. GloMIQO: global mixed-integer quadratic optimizer. *J Glob Optim.* 2013;57(1):3–50.
23. Li Z, Ierapetritou MG. Integrated production planning and scheduling using a decomposition framework. *Chem Eng Sci.* 2009;64:3585–3597.

Appendix A: Equivalence of the Full-Space MINLP/MILP and the MPIP Planning Models

In this appendix, we will show how the first and second level models described in the article are can be obtained from the full-space model. The derivation presented here uses reasoning similar to Li and Ierapetritou.[23] Let us consider the following full-space optimization model (A1), also known as a single level model

$$\begin{aligned} &\min_x cx \\ &\text{s.t.} \quad x \in X \\ &x \geq 0 \end{aligned} \tag{A1}$$

x represents all the variables (continuous and integer), and X represents the feasible space given by all the sets of constraints on x (linear and nonlinear). Therefore, (A1) may represent a full-space MINLP or MILP model. Let us consider that we have a planning horizon originally discretized in M time periods ($M=\{m\}$, i.e., the L2-periods). Suppose we can aggregate over time some of the variables in the model. This aggregation is carried out over K nonoverlapping time intervals that span the entire planning horizon ($K=\{k\}$, i.e., the L1-periods). Let us define x in terms of variables that can be aggregated ($x_2(m)$), variables that may have a fixed value during an aggregated time interval ($y_2(m)$), variables that cannot be aggregated ($z_2(m)$), the aggregated variables ($x_1(k)$), and variables representing the fixed property/condition during an aggregated interval ($y_1(k)$)

$$x \equiv \{x_1(k), y_1(k), x_2(m), y_2(m), z_2(m)\} \tag{A2}$$

By definition

$$x_1(k) = \sum_{m \in k} x_2(m) \qquad \forall k \tag{A3}$$

$$y_1(k) = y_2(m) \qquad \forall (m,k) \in \text{MK}(m,k) \tag{A4}$$

where MK(m,k) is the set that indicates which adjacent time periods m are aggregated as the interval k. It is noted that $x_1(k)$ can only represent continuous variables, whereas $y_1(k)$ can represent continuous and discrete variables. In terms of the gasoline blend planning problem, $x_1(k)$ represents the aggregated production volumes at the first level, and $y_1(k)$ represents the blend recipes and the quality values of the blends. Let us rewrite model (A1) as (A5), which is still the original full-space model

$$\begin{aligned} &\min_{x_2,y_2,z_2} \textstyle\sum_m [c_1x_2(m)+c_2y_2(m)+c_3z_2(m)] \\ &\text{s.t.} \quad x_2(m), y_2(m), z_2(m) \in X \quad \forall m \\ &x_2(m), y_2(m), z_2(m) \geq 0 \quad \forall m \end{aligned} \tag{A5}$$

Model (A6) is obtained by substituting the aggregated variables into Model (A5). The constraints can now be divided into two sets. One set will contain all the equations regarding the $x_1(k)$ and $y_1(k)$ variables, and the associated feasible space is denoted as X_1. The second set of constraints will consist of all the equations containing the $x_2(m)$, $y_2(m)$, and $z_2(m)$ variables, with X_2 as the associated feasible space

$$\begin{aligned} &\min_{x_1,y_1,z_2} \textstyle\sum_k [c_1x_1(k)+c_2i(k)y_1(k)] + \sum_m c_3z_2(m) \\ &\text{s.t.} \quad x_1(k), y_1(k) \in X_1 \quad \forall k \\ &\textstyle\sum_{m\in k} x_2(m) = x_1(k) \quad \forall k \\ &y_2(m) = y_1(k) \qquad \forall (m,k) \in MK(m,k) \\ &x_2(m), y_2(m), z_2(m) \in X_2 \quad \forall m \\ &x_1(k), y_1(k), x_2(m), y_2(m), z_2(m) \geq 0 \quad \forall k, m \end{aligned} \tag{A6}$$

Model (A6) can be written as a two-level problem given by (A7), which is analogous to our first level model plus the second level approximate scheduling model of the MPIP planning algorithm

$$\begin{aligned} &\min_{x_1,y_1} \textstyle\sum_k [c_1x_1(k)+c_2i(k)y_1(k)] + f_2 \\ &\text{s.t.} \quad x_1(k), y_1(k) \in X_1 \quad \forall k \\ &x_1(k), y_1(k) \geq 0 \quad \forall k \end{aligned}$$

where

$$\begin{aligned} &f_2 = \min \textstyle\sum_m c_3z_2(m) \\ &\text{s.t.} \quad \textstyle\sum_{m\in k} x_2(m) = x_1(k) \quad \forall k \\ &y_2(m) = y_1(k) \qquad \forall (m,k) \in \text{MK}(m,k) \\ &x_2(m), y_2(m), z_2(m) \in X_2 \quad \forall m \\ &x_2(m), y_2(m), z_2(m) \geq 0 \quad \forall m \end{aligned} \tag{A7}$$

It can be noted that X_1 is a relaxation of X because not all the original equations are considered in it. X_2 is a contraction of X because all the original equations are considered (except those corresponding to $y_2(m)$, but because $y_1(k) \in X_1$, then $y_2(m)$ is feasible) plus some new restrictions [i.e., (A3) and (A4)]. Therefore, the feasible space X_2 depends on the values of $x_1(k)$ and $y_1(k)$. It is possible that X_2 will include the optimum x^* from the original X when the number of aggregated intervals K is greater than some minimum number k^*; in other words, when $K \geq k^*$ the values of $x_1(k)$ and $y_1(k)$ are such that the feasible space X_2 will contain the optimum x^* from X. Finding this minimum number k^* is the objective of the MPIP planning algorithm.

If we assume that the objective function only contains aggregated variables, that is, $c_3=0$, Model (A7) can be expressed as Model (A8)

$$\begin{aligned} &\min_{x_1,y_1} \textstyle\sum_k [c_1x_1(k)+c_2i(k)y_1(k)] + f_2 \\ &\text{s.t.} \quad x_1(k), y_1(k) \in X_1 \quad \forall k \\ &x_1(k), y_1(k) \geq 0 \quad \forall k \end{aligned}$$

where

$$\begin{aligned} &f_2 = 0 \\ &\text{s.t.} \quad \textstyle\sum_{m\in k} x_2(m) = x_1(k) \quad \forall k \\ &y_2(m) = y_1(k) \qquad \forall (m,k) \in \text{MK}(m,k) \\ &x_2(m), y_2(m), z_2(m) \in X_2 \quad \forall m \\ &x_2(m), y_2(m), z_2(m) \geq 0 \quad \forall m \end{aligned} \tag{A8}$$

Therefore, it can be seen that the second level model is just a feasibility problem. Model (A8) is analogous to the first level plus the second level blend planning model of our MPIP planning algorithm. We can include slack variables ($s_1(k)$ and $s_2(m)$) at each level of the model to obtain a numerical feasible solution even when some constraints are violated. P represents the penalty coefficients. A physically feasible solution will have all the slack variables equal to zero. From Model (A8), it can be seen that the blend cost value is fixed according to $x_1(k)$ and $y_1(k)$; however, we observed that the inclusion of the blend cost term in our MPIP second level model formulation speeded up the solution of the feasibility problem.

We can define the lower and upper bounds (LB and UB, respectively) as follows

$$\mathrm{LB} = \arg\left\{\min_{x_1, y_1 \in X_1} \sum_k \left[c_1 x_1(k) + c_2 y_1(k) + P s_1(k)\right]\right\} \quad \text{(A9)}$$

$$\mathrm{UB} = \arg\left\{\min_{x_2, y_2 \in X_2} \sum_m \left[c_1 x_2(m) + c_2 y_2(m) + P s_2(m)\right]\right\} \quad \text{(A10)}$$

The MPIP algorithm finds the minimum number of aggregated time intervals, k^*, by subdividing the intervals at the first level at each iteration (iter) if $\mathrm{UB}^{(\mathrm{iter})} > \mathrm{LB}^{(\mathrm{iter})}$. For blend planning (Model A8), the MPIP algorithm computes optimal solutions since the pinch points define the minimum production target and the rules to eliminate infeasibilities preblend as low as possible. For approximate scheduling (Model A7), current solutions may only be considered near-optimal as long as the coefficients denoted by c_3 are similar or smaller in magnitude to coefficients c_1 and c_2. This is due to term f_2 not being estimated at the first level by our algorithm.

Manuscript received Oct. 24, 2013, and revision received Feb. 4, 2014.