Mixed-Logic Dynamic Optimization Applied to Batch Distillation Process Design

Jan Oldenburg and Wolfgang Marquardt
Lehrstuhl für Prozesstechnik, RWTH Aachen University, D-52064 Aachen, Germany

Dieter Heinz and Daniel B. Leineweber
Bayer Technology Services, Bayer AG, D-51368 Leverkusen, Germany

An approach for the optimal configuration and sequencing of complex batch-distillation processes is presented. The proposed method is based on formulating a mixed-logic dynamic optimization problem in which a performance index is minimized subject to a disjunctive dynamic model and constraints. The disjunctive model comprises a given set of equipment modules, such as stacks of trays of different size or feed and product vessels, available for the distillation process under consideration along with logical relationships provided by the design context. The optimization based on this model may yield unconventional design and operational strategies, and thus improve the economic efficiency to a significant extent. The logic-based solution technique proposed in this work offers some favorable properties when compared to established mixed-integer dynamic optimization problem solution techniques reported in the literature. The design of a multistage batch-distillation process serves as an example to illustrate the proposed modeling and solution method.

Introduction

Batch distillation is frequently encountered in the specialty chemicals and pharmaceutical industries due to its high flexibility. Classic operating policies for batch distillation columns are characterized by either a constant reflux strategy with variable product composition or a variable reflux strategy where the key component in the product is held at constant composition. However, the most profitable batch operation is obtained when the distillation column is operated under optimal reflux. Optimal reflux strategies are usually determined by minimization of the operating costs (or economic profit maximization) subject to dynamic process models.

Besides the flexibility in terms of operational strategies, the transient nature of batch distillation allows for a number of different column configurations, such as regular, inverse, middle, or multivessel designs. Moreover, multifraction and multistage operation involving cuts and recycles are further alternatives. These emerging designs, combined with different possible operating modes, provide a wide flexibility and economic potential, but result in demanding optimization problems due to the combined discrete and continuous nature of the decision variables (Kim and Diwekar, 2001).

Methods that simultaneously address the structural design and the determination of operational strategies of transient reaction and separation processes in chemical engineering have been proposed by several authors, including Barrera and Evans (1989), Salomone and Iribarren (1992), Mujtaba and Macchietto (1996), and Bhatia and Biegler (1996). Allgor and Barton (1997) and Sharif et al. (1998) formulate the design problem as a *mixed-integer dynamic optimization* (MIDO) problem that incorporates both process dynamics and discrete decision variables in a *superstructure* model.

In this work, we propose a *mixed-logic dynamic optimization* (MLDO) approach for batch distillation design. MLDO is a method based on generalized disjunctive programming (Raman and Grossmann, 1994) that simultaneously addresses the optimal design of the structure and the operational strategies of a batch distillation process. It will be demonstrated that this optimization problem formulation is a rather natural and intuitive choice to represent the design problem. Using the MLDO approach, a batch distillation process is represented by a set of available equipment modules. Each module is de-

Correspondence concerning this article should be addressed to W. Marquardt.

scribed by a set of differential-algebraic equations that are formulated through disjunctions in order to mathematically express all possible design configurations and sequencing options together with the associated logical relationships. Obviously, this leads to a special type of dynamic optimization problem that cannot be solved with standard solution techniques. We therefore propose a tailored logic-based solution algorithm in this article.

A typical application would be the design of a multistage batch distillation process, for which a fixed number of columns, several stacks of trays, and basic instrumentation are available from the company's warehouse or an existing plant. This view of the design problem restricts the potential design alternatives to those that are economically most attractive. From a mathematical point of view, the modular design strategy helps reduce the number of disjunctions, and thus limits the combinatorial complexity. But still, the total number of design alternatives involved in the disjunctive process model will generally be too high to solve the MLDO problem without an algorithm that avoids full enumeration of the discrete decision space.

In the following section we present how the design problem considered in this article can be formulated in a rather natural way as a MLDO problem. Subsequently, we briefly review how the MLDO problem can be solved using a number of alternative solution approaches. These approaches require a transformation of the MLDO into an equivalent MIDO problem. A logic-based solution algorithm is then proposed that directly exploits the MLDO representation. Basic properties of the proposed algorithm are identified by means of an illustrative example problem.

Problem Formulation

A batch process, being inherently transient, comprises various time-continuous degrees of freedom, such as the reflux ratio or heat duty, in case of a batch distillation column. These *control* variables can be used to minimize a performance index, commonly termed objective function, which is formulated for the batch process considered. The objective function covers the operating and investment costs of the plant. The optimal trajectories of the control variables are, in general, time-variant and, in fact, good suboptimal profiles are rather hard to determine by trial and error using simulation experiments. Additionally, optimal trajectories of batch processes are often discontinuous. For these reasons, dynamic optimization techniques have been widely used for optimal trajectory design.

If, however, the batch process structure or parts thereof are not predetermined, additional degrees of freedom can be used to further improve the process performance. These additional degrees of freedom are of a discrete nature, since they represent the existence or nonexistence of a batch process unit or a part thereof, such as the tray of a batch distillation column (Sharif et al., 1998). Discrete decisions could be incorporated into the mathematical model by introducing binary variables $\boldsymbol{y} \in \{0,1\}^{n_y}$ to form a *superstructure* model. The binary variables can be interpreted as special time-invariant parameters that are restricted to adopt discrete values of either 0 or 1.

Batch process units can be connected to each other in two different ways. First, they can be interconnected through streams in the case of parallel operation in time. An optional stream between two units of the batch process, such as a distillation column and an intermediate product tank, can easily be incorporated into the mathematical superstructure by an algebraic equation.

Whereas stream connections are also known from continuously operated plants, batch process operation additionally often comprises a sequence of distinct operational tasks. A typical example for such a *multistage* process is a batch distillation column, which is fed with the product of a batch reactor (Charalambides et al., 1995). Note that this type of interconnection is defined for distinct time instants only. Therefore, the dynamic model is subdivided into n_s stages that represent the batch-process tasks. The (optional) connection between the process stages is expressed mathematically in terms of stage transition conditions.

The development of mathematical superstructure models is by far no trivial task, and the model formulation will have a major impact on the properties of the numerical solution method employed (Williams, 1999). For these reasons, considerable effort has been spent in defining ways to build superstructure models that are as generic as possible and to develop tailored numerical solution algorithms. Raman and Grossmann (1994) proposed to employ logic-based superstructure models involving disjunctions expressed by Boolean variables $\boldsymbol{Y} \in \{\text{True, False}\}^{n_Y}$. Each Boolean variable value Y_i, $i = 1, \ldots, n_Y$, is related to a disjunctive term comprising conditional equations and constraints that represent a unit of the process under consideration.

A disjunctive model representation reveals a close relation to the batch process design task since it captures the qualitative (logical) and the quantitative (equations) part of the design problem in a direct way (Grossmann and Hooker, 2000). In fact, disjunctive models were shown to have a number of favorable properties in the context of process synthesis of continuously operated plants (Raman and Grossmann, 1994). Türkay and Grossmann (1996) have developed tailored solution algorithms that exploit the structure of the disjunctive models.

Inspired by these results, we transferred the basic ideas of disjunctive programming to the dynamic case and, hence, state the following mixed-logic dynamic optimization problem that involves n_s batch process stages with the stage counter $k \in K = \{1, \ldots, n_s\}$

$$\min_{z_k(t), u_k(t), p, t_k, Y} \Phi := \sum_{k=1}^{n_s} \Phi_k\big(z_k(t_k), \boldsymbol{p}, t_k\big) + \sum_{i=1}^{n_Y} b_i \quad (1)$$

$$\text{s.t.} \quad \boldsymbol{f}_k\big(\dot{\boldsymbol{z}}_k^d(t), \boldsymbol{z}_k(t), \boldsymbol{u}_k(t), \boldsymbol{p}, t\big) = \boldsymbol{0}, \quad t \in [t_{k-1}, t_k], \quad k \in K \quad (2)$$

$$\boldsymbol{l}\big(\dot{\boldsymbol{z}}_1^d(t_0), \boldsymbol{z}_1(t_0), \boldsymbol{p}\big) = \boldsymbol{0} \quad (3)$$

$$\boldsymbol{g}_k\big(\boldsymbol{z}_k(t), \boldsymbol{u}_k(t), \boldsymbol{p}, t\big) \le \boldsymbol{0}, \quad t \in [t_{k-1}, t_k], \quad k \in K \quad (4)$$

$$\boldsymbol{g}_k^e\big(\boldsymbol{z}_k(t_k), \boldsymbol{u}_k(t_k), \boldsymbol{p}, t_k\big) \le \boldsymbol{0}, \quad k \in K \quad (5)$$

$$\boldsymbol{z}_{k+1}^d(t_k) - \boldsymbol{m}_k\big(\boldsymbol{z}_k(t_k), \boldsymbol{p}\big) = \boldsymbol{0}, \quad k \in K_m \quad (6)$$

$$\begin{bmatrix} Y_i \\ \\ \boldsymbol{q}_{k,i}\left(\dot{z}_k^d(t), z_k(t), \boldsymbol{u}_k(t), \boldsymbol{p}, t\right) = \boldsymbol{0} \\ t \in [t_{k-1}, t_k] \quad k \in K \\ \boldsymbol{s}_i\left(\dot{z}_1^d(t_0), z_1(t_0), \boldsymbol{p}\right) = \boldsymbol{0} \\ \boldsymbol{r}_{k,i}\left(z_k(t), \boldsymbol{u}_k(t), \boldsymbol{p}, t\right) \leq \boldsymbol{0} \\ t \in [t_{k-1}, t_k], \quad k \in K \\ \boldsymbol{r}_{k,i}^e\left(z_k(t_k), \boldsymbol{u}_k(t_k), \boldsymbol{p}, t_k\right) \leq \boldsymbol{0}, \quad k \in K \\ \boldsymbol{s}_{t,i} := z_{k+1}^d(t_k) - \boldsymbol{v}_{k,i}\left(z_k(t_k), \boldsymbol{p}\right) = \boldsymbol{0}, \quad k \in K_m \\ b_i = \gamma_i \end{bmatrix}$$

$$\vee \begin{bmatrix} \neg Y_i \\ \boldsymbol{B}_{k,i}\left[\boldsymbol{u}_k^T(t), \boldsymbol{p}^T, z_k^d(t_{k-1})^T, z_k(t)^T\right]^T = \boldsymbol{0} \\ t \in [t_{k-1}, t_k], \quad k \in K \\ b_i = 0 \end{bmatrix}$$

$$i = 1, \ldots, n_Y \tag{7}$$

$$\Omega(\boldsymbol{Y}) = \text{True} \tag{8}$$

where $z_k \in \mathbb{R}^{n_{z_k}}$ denotes the vector of differential and algebraic state variables for stage k. Time-invariant parameters and Boolean variables are represented by $\boldsymbol{p} \in \mathbb{R}^{n_p}$ and $\boldsymbol{Y} \in \{\text{True, False}\}^{n_Y}$, respectively, while control variables are denoted with $\boldsymbol{u}_k \in \mathbb{R}^{n_{u_k}}$. $\boldsymbol{f}_k \in \mathbb{R}^{n_{f_k}}$ represents the set of differential-algebraic equations (DAEs) with a differential index of at most 1 (cf. Eq. 2). We assume that the equation system can be transformed into semi-explicit form by simple algebraic manipulations. The initial conditions for the first stage are given in Eq. 3 by $\boldsymbol{l}$. Stage transition conditions (Eq. 6) are used to map the differential state variable values z_k^d across the stage boundaries. The indices of these conditions are collected in the set $K_m = \{1, \ldots, n_s - 1\}$. Inequalities 4 and 5, comprising $\boldsymbol{g}_k$ and $\boldsymbol{g}_k^e$, are used to enforce path and end point constraints. The objective function $\Phi \in \mathbb{R}$ in Eq. 1 covers the operating costs $\Phi_k \in \mathbb{R}$ induced by each batch stage k as well as the investment costs b_i. Without loss of generality, Φ_k is formulated as a Mayer-type objective function and evaluated at the final time, t_k, of each batch stage. Although not stated explicitly, control variables $\boldsymbol{u}_k(t)$ might be included in Φ_k through auxiliary state variables and additional model equations (Büskens, 1998).

Whereas $\boldsymbol{f}_k$, $\boldsymbol{l}$, $\boldsymbol{g}_k$, $\boldsymbol{g}_k^e$, and Φ hold globally, there are further equalities $\boldsymbol{q}_{k,i}$, inequalities $\boldsymbol{r}_{k,i}$, $\boldsymbol{r}_{k,i}^e$, and stage transition conditions included in the disjunctions (Eq. 7) that are only enforced if the corresponding Boolean variable Y_i is True. Otherwise, a subset of the system variables, time-invariant parameters, initial values of differential states, as well as fixed investment costs, b_i, for nonexisting units are set to zero. Thus, $\boldsymbol{B}_{k,i}$ is a square diagonal matrix with constant 1- or 0-valued matrix elements. Note that, although end point constraints $\boldsymbol{g}_k^e$ and $\boldsymbol{r}_{k,i}^e$ are formally stated as inequalities, these constraints may also be explicitly formulated as equalities.

Commonly, the Boolean variables $\boldsymbol{Y}$ themselves are partly related through propositional logical expressions (Eqs. 8) as proposed, for example, by Raman and Grossmann (1994). A typical logic expression, termed implication, would be

$$Y_{\text{regular}} \Rightarrow \neg Y_{\text{inverse}} \tag{9}$$

to express that if a regular mode of operation is chosen ($Y_{\text{regular}} = \text{True}$), the batch column cannot be operated inversely ($Y_{\text{inverse}} = \text{False}$). Logical expressions, such as the implication in Eq. 9, are usually transformed into a representation $\Omega(\boldsymbol{Y}) = \text{True}$, which involves logical operations only. Such a representation of Eq. 9 could be stated as (Raman and Grossmann, 1992)

$$\Omega(\boldsymbol{Y}) = \neg Y_{\text{regular}} \vee \neg Y_{\text{inverse}} = \text{True} \tag{10}$$

One generally aims at logical expressions of the form in Eq. 10 due to the fact that they can be easily transformed into an algebraic representation, as will be shown in the subsequent section. A large portion of the combinatorial complexity of the MLDO problem is governed by these logic relationships, a fact that will become obvious in conjunction with the illustrative example problems presented later. Besides the two-term disjunctions shown in Eq. 7, a generalized representation of disjunctions involving multiple terms will be used to mathematically express alternatives to sequence and configure a batch distillation process. This will be shown in conjunction with an illustrative example problem.

Note that, with a fixed choice of Boolean variables $\boldsymbol{Y}$, given control variables $\boldsymbol{u}_k$, parameters $\boldsymbol{p}$, and initial values $z_k^d(t_{k-1})$, the combined set of DAEs, $\boldsymbol{f}_k$, $\boldsymbol{q}_{k,i}$, the corresponding initial conditions $\boldsymbol{l}$, $\boldsymbol{s}_i$, and stage transition conditions (cf. Eqs. 2, 3, 6, 7) are assumed to uniquely determine the state-variable vector $z_k(t)$.

The optimization problem (Eqs. 1–8) can be solved using a number of different approaches. A review of approaches reported in the literature is provided in the next section in order to be able to classify the proposed method in the section titled "Logic-based solution approach". These approaches are based on a reformulation of the MLDO problem into a MIDO problem.

Solution Approaches via MIDO

A MIDO problem formulation, which is equivalent to Eqs. 1–8, can always be obtained by a reformulation.

Reformulation as a MIDO problem

A mixed-integer dynamic optimization problem is obtained by replacing the Boolean variables with binary variables $\boldsymbol{y} \in \{0,1\}^{n_y}$ and by representing the disjunctions (Eq. 7) either using big-M constraints (for example, Williams, 1999) or a convex-hull formulation (Balas, 1985; Türkay and Grossmann, 1998). The use of big-M constraints is illustrated by the reformulation of the inequality constraints $r_{k,i}(\boldsymbol{u}_k(t), t) \leq 0$ in Eq. 7, which is here assumed to be independent of process states and parameters for the sake of simplicity. With $M_{\mathfrak{U}}^{r_{k,i}}$ and $\boldsymbol{u}_{k,\mathfrak{U}}$, $\boldsymbol{u}_{k,\mathfrak{L}}$ being constants with a sufficiently large absolute value, we obtain

$$r_{k,i}\left(\boldsymbol{u}_k(t), t\right) \leq M_{\mathfrak{U}}^{r_{k,i}}(1 - y_i) \tag{11}$$

$$\boldsymbol{u}_{k,\mathfrak{L}}\, y_i \leq \boldsymbol{u}_k(t) \leq \boldsymbol{u}_{k,\mathfrak{U}}\, y_i \tag{12}$$

By substitution of $y_i = 1$ into Eq. 11 it can be seen that the constraint $r_{k,i} \le 0$ is enforced. Otherwise, if $y_i = 0$, $r_{k,i}$ is unconstrained for an appropriate choice of $M_{\boldsymbol{u}}^{r_{k,i}}$ and a subset of the variables $\boldsymbol{u}_k$ is set to zero (Eq. 12).

Both reformulation techniques are well-known in the context of mixed-integer nonlinear programming (MINLP) for the synthesis of process flow sheets. The basic concepts can also be applied to problems involving dynamic process models like the MLDO problem stated earlier. Thus, by employing big-M constraints for each disjunction in Eq. 7, we can state the following multistage mixed-integer dynamic optimization problem that corresponds to Eqs. 1–8

$$\min_{z_k(t),\boldsymbol{u}_k(t),\boldsymbol{p},t_k,y} \Phi := \sum_{k=1}^{n_s} \Phi_k(z_k(t_k),\boldsymbol{p},t_k) + \sum_{i=1}^{n_y} b_i \quad (13)$$

$$\text{s.t.} \quad \text{Eqs.} \quad 2\text{–}6 \quad (14)$$

$$\boldsymbol{M}_{\mathfrak{L}}^{q_{k,i}}(1-y_i) \le \boldsymbol{q}_{k,i}(\dot{z}_k^d(t),z_k(t),\boldsymbol{u}_k(t),\boldsymbol{p},t) \le \boldsymbol{M}_{\mathfrak{U}}^{q_{k,i}}(1-y_i)$$

$$t \in [t_{k-1},t_k], \quad k \in K$$

$$\boldsymbol{M}_{\mathfrak{L}}^{s_i}(1-y_i) \le \boldsymbol{s}_i(\dot{z}_1^d(t_0),z_1(t_0),\boldsymbol{p}) \le \boldsymbol{M}_{\mathfrak{U}}^{s_i}(1-y_i)$$

$$\boldsymbol{r}_{k,i}(z_k(t),\boldsymbol{u}_k(t),\boldsymbol{p},t) \le \boldsymbol{M}_{\mathfrak{U}}^{r_{k,i}}(1-y_i)$$

$$t \in [t_{k-1},t_k], \quad k \in K$$

$$\boldsymbol{r}_{k,i}^e(z_k(t_k),\boldsymbol{u}_k(t_k),\boldsymbol{p},t_k) \le \boldsymbol{M}_{\mathfrak{U}}^{r_{k,i}^e}(1-y_i)$$

$$k \in K$$

$$\boldsymbol{M}_{\mathfrak{L}}^{s_{t,i}}(1-y_i) \le z_{k+1}^d(t_k) - \boldsymbol{v}_{k,i}(z_k(t_k),\boldsymbol{p}) \le \boldsymbol{M}_{\mathfrak{U}}^{s_{t,i}}(1-y_i)$$

$$k \in K_m \quad (15)$$

$$\left[z_{k,\mathfrak{L}}^T, \boldsymbol{u}_{k,\mathfrak{L}}^T, \boldsymbol{p}_{\mathfrak{L}}^T, z_{k,\mathfrak{L}}^{d^T}\right]^T y_i$$

$$\le \boldsymbol{B}_{k,i}\left[z_k(t)^T, \boldsymbol{u}_k^T(t), \boldsymbol{p}^T, z_k^{d^T}(t_{k-1})\right]^T$$

$$\le \left[z_{k,u}^T, \boldsymbol{u}_{k,u}^T, \boldsymbol{p}_u^T, z_{k,u}^{d^T}\right]^T y_i \quad (16)$$

$$t \in [t_{k-1}, t_k],\ k \in K,$$

$$\boldsymbol{b_i} = \gamma_i \boldsymbol{y_i},\ \boldsymbol{i} = 1, \ldots, \boldsymbol{n_y},$$

$$Ay \le d \quad (17)$$

In this formulation the Boolean variables $\boldsymbol{Y}$ are replaced by the binary variables $\boldsymbol{y}(n_Y = n_y)$. The binary variables are used together with big-M constants to mathematically express the logic relationship between Boolean variables and constraints contained in the disjunctions (Eq. 7).

As shown, for example, by Türkay and Grossmann (1996), propositional logic constraints (Eqs. 8) formulated for the mixed-logic optimization problem can be expressed in terms of linear constraints (Eq. 17), including binary variables in a MIDO problem formulation. Again considering the example presented in conjunction with the MLDO problem formulation, where an inverse mode of operation is excluded from the set of alternatives in case the column is operated regularly (cf. Eq. 9), we would here state the linear inequality

$$1 - y_{\text{regular}} + 1 - y_{\text{inverse}} \ge 1 \quad (18)$$

or equivalently

$$y_{\text{regular}} + y_{\text{inverse}} \le 1 \quad (19)$$

in order to obtain an algebraic representation instead of a logical expression. The inequality in Eq. 18 can be deduced directly from Eq. 10.

Overview of solution methods for MIDO problems

Direct solution methods for dynamic optimization problems without discrete variables are capable of solving a broad class of problems, including applications governed by large-scale DAE systems (Ishikawa et al., 1997; Abel et al., 2000). They convert the time-continuous dynamic optimization problem into a finite-dimensional *nonlinear programming problem* (NLP) by discretization. The corresponding solution approaches can be divided into simultaneous and sequential methods. The former discretizes both state and control variables, leading to a large-scale NLP problem (Cuthrell and Biegler, 1987; Leineweber et al., 2003). Commonly, this method is also termed *full discretization* approach. Within the sequential or *control vector parameterization* method (Kraft, 1985) only control variables are discretized, while state variable trajectories are determined by solving the dynamic model constraints as an initial value problem.

By using one of these direct approaches, the mixed-integer dynamic optimization problem (Eqs. 13–17) can be converted into an algebraic problem that then forms a *mixed-integer nonlinear program* (MINLP) instead of an NLP. Fortunately, a number of solution methods are available for solving MINLP problems (Grossmann, 2002; Floudas, 1995). In fact, these algorithms were shown to be suitable for solving problems of considerable size. However, all of them suffer from the fact that they inherently rely on the convexity of the MINLP, which does not hold true for almost all practical applications.

MINLP algorithms reported in the literature can be divided into two classes. They are based either on enumeration or on decomposition. The former class comprises complete enumeration of the whole discrete decision space (which is only tractable for problems with a small number of discrete variables) and branch-and-bound-type methods (Nemhauser and Wolsey, 1999; Floudas, 1995). Branch-and-bound-type methods relax the problem in Eqs. 13–17 continuously, that is, the binary variables are allowed to take real values between 0 and 1. The relaxed problem is then solved in order to provide a lower bound to the solution of the optimization problem. Subsequently, different strategies can be applied to fix subsets of the binary variables in the nodes of a search tree. Parts of the search tree can be cut off, if the current solution is infeasible or greater than a known upper bound to the solution of the optimization problem, or if the continuously relaxed binary variables take discrete values 0 or 1. The (global) optimal solution is found when all binary variables take discrete values {0,1} and the relaxed dynamic optimization problems are solved to (global) optimality in all the nodes. A branch-and-bound solution strategy for MIDO problems was proposed recently by Buss et al. (2000).

On the other hand, decomposition methods are based on decomposing the optimization problem into two subproblems, which are solved in an iterative manner. *Outer Approximation* (OA) (Duran and Grossmann, 1986; Kocis and Grossmann, 1987; Viswanathan and Grossmann, 1990; Fletcher and Leyffer, 1994) and *Generalized Benders Decomposition* (GBD) (Geoffrion, 1972) are two well-known decomposition algorithms, which have been used, for example, by Schweiger and Floudas (1997) and Bansal et al. (2002) to solve MIDO problems of considerable size.

In this work we focus on the OA method, where the primal subproblem, a dynamic optimization problem with fixed binary variable values, is solved to yield an upper bound, $\mathcal{Z}_{\mathfrak{u}}^{L}$, for the optimization problem (Eqs. 13–17). Subsequently, a linear outer approximation of the overall mixed-integer problem is utilized within the master subproblem [a *mixed-integer linear programming* (MILP) problem] to provide a lower bound to problem in Eqs. 13–17 and a new set of binary variables for the next primal subproblem. After a finite number of iterations, the nondecreasing lower bound and the minimum of all upper bounds $Z_{\mathfrak{u}}^{L} := \min(\mathcal{Z}_{\mathfrak{u}}^{\ell}),\ \ \ell = 0, \ldots, L,$ approach each other up to a specified tolerance ϵ at the optimum value of the optimization problem (Eqs. 13–17), given the problem under consideration is convex. In practical applications, in which the problem generally is assumed to be nonconvex, termination within an ϵ-tolerance cannot be expected, and a heuristic stopping criterion has to be employed instead, such as to stop when there is no decrease in two successive primal solutions $\mathcal{Z}_{\mathfrak{u}}^{L}$. To tackle this problem, an extension of the OA algorithm, termed *Augmented Penalty*, has been proposed by Viswanathan and Grossmann (1990). Moreover, convexity tests can be applied to "validate" linearizations accumulated in the master problems [for more information, refer to Grossmann (2002)]. These extensions have been shown to work reasonably well for a number of nonconvex problems. However, in either case, there is no guarantee that the global optimum is located within a finite number of iterations. In order to circumvent this restriction, global solution algorithms for nonconvex problems of a relatively small size have been developed (Adjiman et al., 2000). Large-scale problems will not be tractable by these algorithms in the near future due to their high computational burden.

Allgor and Barton (1997) proposed an alternative approach where the MIDO problem is solved by iterating between two design subproblems, recipe design and equipment allocation. The recipe design problem constitutes a dynamic optimization problem that assumes a fixed process structure, and, thus, fixed discrete (binary) decision variables. However, in contrast to the approaches based on MINLP, Allgor and Barton (1997) generate the equipment allocation subproblems on the basis of *screening models* that are developed using "domain specific information gathered from physical laws and engineering insight." These screening models lead to MILP problems, which yield rigorous lower bounds to the MIDO problem. The development of these models is, however, a rather complex task.

Logic-Based Solution Approach

In this section we propose an alternative approach to solve the mixed-logic dynamic optimization problem. This solution method belongs to the class of decomposition strategies, and the basic concepts mentioned in the previous section can be carried over. In contrast to a MIDO solution technique, the logic-based method is directly applied to the MLDO problem (Eqs. 1–8) rather than to its big-M transformation (Eqs. 13–17).

In particular, the logic-based OA algorithm proposed by Türkay and Grossmann (1996) is extended to the dynamic case. The logic-based OA algorithm comprises, like its original variant, two subproblems termed primal and master problem, respectively. The same authors also suggested a logic-based variant of GBD. Moreover, a logic-based branch-and-bound algorithm can be found elsewhere (Lee and Grossmann, 2000).

The logic-based solution presented in this contribution is based on a control vector parameterization of the time continuous optimization problem 1–8. As a consequence, a special treatment of differential equations $\boldsymbol{q}_{k,i} = \boldsymbol{0}$ (and the corresponding negation $z_k(t) = \boldsymbol{0}$) contained in the disjunctions 7 is required, which is non-trivial as already indicated by Avraam et al. (1998). For the logic-based solution algorithm proposed in this article, it is assumed that no differential equations are present in the disjunctions, as is the case for the example problem presented later. Though not supported by an illustrative example, we will outline a modification of the solution approach with which a treatment of problem 1–8 will be possible in its general form.

The primal problem

Whereas the primal problem of a classic OA algorithm is obtained on the basis of a mixed-integer problem formulation [for more details, see Schweiger and Floudas (1997)] like Eqs. 13–17, we here directly employ the disjunctive problem formulation (Eqs. 1–8). With a fixed set of Boolean variable values determined by a master problem or provided by the user as initialization we solve an ordinary dynamic optimization problem using the sequential solution approach.

In the sequential approach, a control profile $u_l^k(t)$ is approximated on subintervals $l = 1, \ldots, n_{u_k}$ in a model stage $k \in K = \{1, \ldots, n_s\}$ by a piecewise polynomial expansion of the form

$$u_l^k(t) \approx \bar{u}_l^k(\boldsymbol{c}_l^k, t) = \sum_{j \in \Lambda_l^k} c_{l,j}^k \phi_{l,j}^k(t) \tag{20}$$

where Λ_l^k denotes the index set of the chosen parameterization functions $\phi_{l,j}^k(t)$ and the vector $\boldsymbol{c}_l^k$ contains the corresponding parameter vector. Each stage time interval $[t_{k-1}, t_k]$ is divided into subintervals defined on each model stage k with grid points $t_{k,j},\ j \in \Lambda_l^k$.

In this work, we only consider piecewise constant functions $\phi_{l,j}^k(t) := 1,\ \forall t_{k,j} \le t \le t_{k,j+1}$, otherwise $\phi_{l,j}^k(t) := 0$. The grid points for each $u_l^k, l = 1, \ldots, n_{u_k}$, are contained in the mesh $\Delta_{\Lambda_l^k} := \{t_{k,j} | j \in \Lambda_l^k\}$. For the evaluation of the path constraints $\boldsymbol{g}_k$, $\boldsymbol{r}_{k,i}$, we furthermore define a unified mesh for all control variables on stage k according to $\Delta_{\Lambda^k} := \bigcup_{l=1}^{n_{u_k}} \Delta_{\Lambda_l^k}$.

The state variables are calculated by numerical integration of the initial value problems (Eqs. 2, 3, 6) defined on n_s model stages. Hence, the dynamic optimization problem can be transformed into the following NLP for fixed Λ_l^k with the search variable vector $\boldsymbol{\theta}_k := [\boldsymbol{c}_{l=1,\ldots,n_{u_k}}^{k^T}, \boldsymbol{p}^T, t_k, \boldsymbol{z}_k^d(t_{k-1})^T]^T$, where $z_k^d(t_{k-1})$ denotes the vector of free initial values of differential state variables in a model stage k

$$\mathcal{Z}_{\mathfrak{U}}^L := \min_{\boldsymbol{\theta}_k} \sum_{k=1}^{n_s} \Phi_k\big(z_k(\boldsymbol{\theta}_k, t_k), \boldsymbol{p}, t_k\big) + \sum_{i=1}^{n_Y} b_i \tag{21}$$

$$\text{s.t.} \quad \boldsymbol{l}\big(\dot{z}_1^d(\boldsymbol{\theta}_1), z_1(\boldsymbol{\theta}_1), \boldsymbol{p}\big) = \boldsymbol{0}$$

$$\boldsymbol{g}_k\big(z_k(\boldsymbol{\theta}_k, t_{k,j}), \boldsymbol{\theta}_k, t_{k,j}\big) \le \boldsymbol{0}, \quad \forall t_{k,j} \in \Delta_{\Lambda^k}, \quad k \in K$$

$$\boldsymbol{g}_k^e\big(z_k(\boldsymbol{\theta}_k, t_k), \boldsymbol{\theta}_k, t_k\big) \le \boldsymbol{0}, \quad k \in K \tag{22}$$

$$\left.\begin{matrix} \boldsymbol{r}_{k,i}\big(z_k(\boldsymbol{\theta}_k, t_{k,j}), \boldsymbol{\theta}_k, t_{k,j}\big) \le \boldsymbol{0} \\ \forall t_{k,j} \in \Delta_{\Lambda^k}, \quad k \in K \\ \boldsymbol{r}_{k,i}^e\big(z_k(\boldsymbol{\theta}_k, t_k), \boldsymbol{\theta}_k, t_k\big) \le \boldsymbol{0} \\ k \in K \\ \boldsymbol{s}_i\big(\dot{z}_1^d(\boldsymbol{\theta}_1), z_1(\boldsymbol{\theta}_1), \boldsymbol{p}\big) = \boldsymbol{0} \\ \boldsymbol{s}_{t,i} := z_{k+1}^d(t_k) - \boldsymbol{v}_{k,i}\big(z_k(\boldsymbol{\theta}_k, t_k),\, \boldsymbol{p}\big) = \boldsymbol{0} \\ k \in K_m \\ b_i = \gamma_i, \quad i = 1, \ldots, n_Y \end{matrix}\right\} \text{ for } Y_i^L = \text{True} \tag{23}$$

$$\left.\begin{matrix} \overline{\boldsymbol{B}}_{k,i} \boldsymbol{\theta}_k = \boldsymbol{0} \\ k \in K \\ b_i = 0, \quad i = 1, \ldots, n_Y \end{matrix}\right\} \text{ for } Y_i^L = \text{False} \tag{24}$$

Note that the suffix 1 used in Eqs. 22 and 23 stands for $k = 1$. In the preceding problem formulation, the dynamic model constraints $\boldsymbol{f}_k = \boldsymbol{0}$ (cf. Eq. 2) and the stage transition conditions (Eq. 6) are not stated explicitly. These equations are solved numerically together as an aggregated initial value problem. The solution of this initial-value problem becomes a part of each NLP iteration through the state variable vector values $z_k = \mathfrak{F}(\boldsymbol{\theta}_k, t)$ and the state variable sensitivities $\boldsymbol{s}_k$ with respect to the discretized decision variables.

The matrix $\boldsymbol{B}_{k,i}$ of the time-continuous problem (Eqs. 1–8) is transformed into a new matrix $\overline{\boldsymbol{B}}_{k,i}$ within the preceding problem formulation to account for the discretization of the time domain. For convenience, the nonlinear constraints comprised in Eqs. 22 and 23 are collected in two different vectors according to

$$\boldsymbol{h}_{k,gl.} := \left[\boldsymbol{l}^T, \boldsymbol{g}_k^T|_{t_{k,j}}, \boldsymbol{g}_k^{e^T}\right]^T \tag{25}$$

$$\boldsymbol{h}_{k,dj.} := \left[\boldsymbol{r}_{k,i}^T|_{t_{k,j}}, \boldsymbol{r}_{k,i}^{e^T}, \boldsymbol{s}_i^T, \boldsymbol{s}_{t,i}^T\right]^T \tag{26}$$

$$t_{k,j} \in \Delta_{\Lambda^k}, \quad k \in K, \quad i = 1, \ldots, n_Y$$

which are enforced at grid points $t_{k,j} \in \Delta_{\Lambda^k}$ within a stage time interval $[t_{k-1}, t_k]$ or at the end point of a model stage t_k. Using this notation, the set of nonlinear primal subproblem constraints, which holds independently of the values of the discrete decision variable (cf. Eq. 22), can be covered by $\boldsymbol{h}_{k,gl.} \le \boldsymbol{0}$ and $\boldsymbol{h}_{k,gl.} = \boldsymbol{0}$. In a similar way, all nonlinear disjunctive constraints (cf. Eq. 23) can be expressed by $\boldsymbol{h}_{k,dj.} \le \boldsymbol{0}$ and $\boldsymbol{h}_{k,dj.} = \boldsymbol{0}$.

The derivatives of the objective function and the constraints with respect to the continuous degrees of freedom $\boldsymbol{\theta}_k$

$$\frac{d\boldsymbol{\Phi}}{d\boldsymbol{\theta}_k} = \left(\frac{\partial \Phi}{\partial z_k}\right)^T \frac{\partial z_k}{\partial \boldsymbol{\theta}_k} + \frac{\partial \boldsymbol{\Phi}}{\partial \boldsymbol{\theta}_k}, \quad \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{\theta}_k} = \left(\frac{\partial \boldsymbol{h}_k}{\partial z_k}\right)^T \frac{\partial z_k}{\partial \boldsymbol{\theta}_k} + \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{\theta}_k}, \tag{27}$$

required by the NLP solver are determined on the basis of state variable sensitivities

$$\boldsymbol{s}_k := \frac{\partial z_k}{\partial \boldsymbol{\theta}_k} \tag{28}$$

which are calculated by the numerical integration of the sensitivity DAE system.

Finally, the solution $\mathcal{Z}_{\mathfrak{U}}^L$ to the problem in Eqs. 21–24 contributes to the nonincreasing upper bound

$$Z_{\mathfrak{U}}^L := \min\big(\mathcal{Z}_{\mathfrak{U}}^\ell\big), \quad \ell = 0, \ldots, L \tag{29}$$

of the MLDO problem. Note that $\mathcal{Z}_{\mathfrak{U}}^L$ determined by a local NLP solver will be a valid upper bound even in the nonconvex case, since $\mathcal{Z}_{ub,\text{local}}^L \ge \mathcal{Z}_{ub,\text{global}}^L$ holds.

In order to improve the performance of the sequential solution method applied to the optimization problem just stated, we directly exploit the disjunctive structure of the primal problem. This is achieved by considering only those disjunctions in which the corresponding Boolean variable Y_i^L is True (cf. Eq. 23) while variables that are set to zero in nonexisting units (cf. Eq. 24) are removed from the problem. A symbolical elimination of these NLP search variables is always possible due to the structure of the linear equations $\overline{\boldsymbol{B}}_{k,i}\boldsymbol{\theta}_k = \boldsymbol{0}$ (cf. Eq. 24). As a result, the logic-based primal problem avoids the solution of a dynamic optimization problem for the entire superstructure. Thus, the dimensionality of the primal problem is reduced because a subset of the constraints and variables does not have to be considered explicitly.

The most significant reduction in numerical effort is obtained by removing parameters $\boldsymbol{\theta}_k$ set to zero in Eq. 24, since this helps reduce the size of the sensitivity DAE system to be solved in each NLP iteration. In fact, the elimination of one single parameter reduces the sensitivity DAE system by the number of state variables, n_{z_k}. To understand the significance of this point, we have to consider that solving primal subproblems involving detailed dynamic process models dominates the computing time of a MIDO/MLDO algorithm. Assuming a limited combinatorial complexity, that is, a few hundred design alternatives, the computing time required to solve the master problems can be neglected when compared to the primal problems. And finally, the largest portion of computational demand for solving the primal problem itself is required to determine the state variable sensitivities with respect to the (discretized) decision variables in each iteration of the NLP. Consequently, the size of the dynamic superstructure model and the number of discretized decision variables is a critical quantity. Currently, problems with several thousands of DAEs and in the order of several hundred control vector parameters can be solved.

The master problem

The master problem of the logic-based OA algorithm is based on accumulated linearizations of the discretized mixed-logic dynamic optimization problem at its solution in each iterate, L. This linear approximation yields an underestimation of the objective function and an overestimation of the feasible region, and thus a lower bound $Z_{\mathfrak{L}}^L$ to the solution of the optimization problem. Hence, the following disjunctive linear program is derived on the basis of the nonlinear problem (Eqs. 21–24)

$$Z_{\mathfrak{L}}^L := \min_{\alpha, \boldsymbol{Y}, \boldsymbol{\theta}_k, \boldsymbol{\sigma}_k^\ell} \alpha + \sum_{\ell=0}^{L} \sum_{k=1}^{n_s} \boldsymbol{w}_k^{\ell,T} \boldsymbol{\sigma}_k^\ell \tag{30}$$

$$\text{s.t.} \quad \Phi^\ell + \nabla \Phi^{\ell^T} \left[\boldsymbol{\theta}_k - \boldsymbol{\theta}_k^\ell \right] \le \alpha, \quad k \in K \qquad \ell = 0, \ldots, L \tag{31}$$

$$\boldsymbol{T}_{k,gl.}^\ell \left\{ \boldsymbol{h}_{k,gl.}^\ell + \nabla \boldsymbol{h}_{k,gl.}^{\ell^T} \left[\boldsymbol{\theta}_k - \boldsymbol{\theta}_k^\ell \right] \right\} \le \boldsymbol{\sigma}_k^\ell \quad k \in K \qquad \ell = 0, \ldots, L \tag{32}$$

$$\begin{bmatrix} Y_i \\ \boldsymbol{T}_{k,dj.}^\ell \left\{ \boldsymbol{h}_{k,dj.}^\ell + \nabla \boldsymbol{h}_{k,dj.}^{\ell^T} \left[\boldsymbol{\theta}_k - \boldsymbol{\theta}_k^\ell \right] \right\} \le \boldsymbol{\sigma}_k^\ell, \\ k \in K, \quad \ell \in K_{i,L}, \quad b_i = \gamma_i, \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ \overline{\boldsymbol{B}}_{i,k} \boldsymbol{\theta}_k = \boldsymbol{0}, \\ k \in K \\ b_i = 0 \end{bmatrix} \qquad i = 1, \ldots, n_Y \tag{33}$$

$$\Omega(\boldsymbol{Y}) = \text{True} \tag{34}$$

Note that $Z_{\mathfrak{L}}^L$ represents a rigorous lower bound only if the objective function Φ and the constraints $\boldsymbol{h}_k$ of the primal problem (Eqs. 21–24) are convex functions with respect to the degrees of freedom $\boldsymbol{\theta}_k$. The linearization of a nonconvex function does not yield an *outer* approximation, and thus parts of the feasible region may be cut off. In order to avoid infeasible master problems caused by the nonconvexity of the mixed-integer problem, *Augmented Penalty* OA (Viswanathan and Grossmann, 1990) is employed. This approach relaxes the linear inequality constraints contained in the master problem by introducing positive slack variables $\boldsymbol{\sigma}_k$. The slack variables are included in the objective function through a penalty term with weights $\boldsymbol{w}_k$ chosen to be sufficiently large [for more details, see Viswanathan and Grossmann (1990)].

Linearizations of the objective function (Eq. 21) and the global constraints $\boldsymbol{h}_{k,gl.}$ are accumulated in each major iteration L. Constraints $\boldsymbol{h}_{k,dj.}$ contained in disjunctions are only included in the master problem in case the corresponding Boolean variable Y_i^ℓ is True. In formal terms, this is expressed by the set $K_{i,L} = \{ \ell | Y_i^\ell = \text{True}, \ \ell = 0, \ldots, L \}$. Note that this property constitutes a major difference to the standard OA method (Duran and Grossmann, 1986) where linearizations of *all* constraints are included in the master problem. Equality constraints are again relaxed on the basis of the primal Lagrange multipliers by means of two square diagonal matrices $\boldsymbol{T}_{k,gl.}$ and $\boldsymbol{T}_{k,dj.}$.

The fact that nonlinear constraints contained in the disjunctions are linearized and included in the master problem only if the corresponding Boolean variable Y_i is True is a particularly favorable property of the logic-based solution method. In practical terms, this means that only process units or parts thereof that have been selected in the previous primal problem are considered at physically meaningful operation conditions, whereas linearizations of temporarily inactive parts of the batch process are simply discarded. More precisely, linearizations are avoided in the presence of zero flows or in case product or feed vessels are empty. As a result, one might say that the logic-based master problem collects filtered process information. Kocis and Grossmann (1989) present a case study in conjunction with process flow sheet synthesis that illustrate the points stated here.

Instead of solving the linear disjunctive programs (Eqs. 30–34) directly [see, for example, Beaumont (1990)], we convert it into a mixed-integer linear programming problem that can be solved using a standard branch-and-bound MILP solution technique (Nemhauser and Wolsey, 1999). This idea was originally proposed by Türkay and Grossmann (1996), who transformed the disjunctions into inequalities involving binary variables using the convex hull reformulation technique. We here employ a big-M representation of the disjunctions according to Yeomans and Grossmann (2000) and state an MILP as a master problem.

Treatment of differential equations contained in disjunctions

As mentioned in the previous section, nonlinear (differential) equations $\boldsymbol{q}_{k,i} = \boldsymbol{0}$ cannot directly be treated with the proposed solution algorithm due to their implicit treatment in the logic-based primal problem. The basic problem is the fact that an explicit representation is required for the master problem in order to relate a discrete design decision, that is, the Boolean variable values, to the continuous variables contained in $\boldsymbol{q}_{k,i}$. One possible way to circumvent this problem is to apply the logic-based decomposition method based on full discretization to solve the MLDO problem 1–8. In this case, the set of continuous decision variables $\boldsymbol{\theta}_k$ is extended by the discretized state variable vector $z_k(t)$, which is approximated by the following piecewise polynomial expansion on each model stage $k \in K$

$$z_l^k(t) \approx \bar{z}_l^k(\hat{z}_l^k, t) = \sum_{j \in \Psi_l^k} \hat{z}_{l,j}^k \psi_{l,j}^k(t), \quad l = 1, \ldots, n_{z_k} \qquad \forall k \in K \tag{35}$$

Lagrange polynomials are a typical choice for parameterization functions $\psi_{l,j}^k(t)$. Here, Ψ_l^k denotes the index set for the functions $\psi_{l,j}^k(t)$. Accordingly, the vector of the continuous search variables is defined as $\overline{\boldsymbol{\theta}}_k := [\hat{z}_{l=1,\ldots,n_{z_k}}^{k,T}, \boldsymbol{\theta}_k^T]^T$. By applying collocation on finite elements (see, for example, Cuthrell and Biegler (1987)) the differential equations $\boldsymbol{f}_k$ and $\boldsymbol{q}_{k,i}$ become equality constraints of the optimization problem, which are enforced at the collocation points contained in a mesh $\mathfrak{M}_k$. Thus, a direct treatment of the conditional equations $\boldsymbol{q}_{k,i}$ is possible.

Alternatively, it is possible to solve the dynamic disjunctive program 1–8 by solving the primal problem as stated in 21–24 and by only formulating the logic-based master problem using a full discretization approach. We here assume for simplicity that the DAE integration of the logic-based primal problem is performed using an implicit Runge-Kutta (IRK) method. With this assumption, it is possible to construct a logic-based primal problem based on collocation on finite elements which is equivalent to the problem 21–24 for a given order and fixed mesh subinterval lengths of the IRK integration method. As a consequence, there is a unique solution vector $\bar{\boldsymbol{\theta}}_k^* := [\hat{z}_{l=1,\ldots,n_{z_k}}^{*,k,T}, \boldsymbol{\theta}_k^{*,T}]^T$ satisfying the KKT conditions of the logic-based primal problem as formulated in Eqs. 21–24, as well as of a formulation based on collocation on finite elements. Hence, the master problem 30–34 can be replaced by the following linear disjunctive program with $\bar{\boldsymbol{\theta}}_k$ as search variable vector

$$Z_{\mathfrak{L}}^L := \min_{\alpha, \boldsymbol{Y}, \bar{\boldsymbol{\theta}}_k, \boldsymbol{\sigma}_k^\ell} \alpha + \sum_{\ell=0}^{L} \sum_{k=1}^{n_s} \boldsymbol{w}_k^{\ell,T} \boldsymbol{\sigma}_k^\ell \tag{36}$$

$$\text{s.t.} \quad \Phi^\ell + \nabla \Phi^{\ell^T} \left[\bar{\boldsymbol{\theta}}_k - \bar{\boldsymbol{\theta}}_k^\ell \right] \le \alpha, \quad k \in K$$

$$\ell = 0, \ldots, L \tag{37}$$

$$\boldsymbol{T}_{k,f}^\ell \left\{ \nabla \boldsymbol{f}_k^T |_{t_{k,j} \in \mathfrak{M}_k} \left[\bar{\boldsymbol{\theta}}_k - \bar{\boldsymbol{\theta}}_k^\ell \right] \right\} \le \boldsymbol{\sigma}_k^\ell \tag{38}$$

$$\boldsymbol{T}_{k,gl.}^\ell \left\{ \boldsymbol{h}_{k,gl.}^\ell + \nabla \boldsymbol{h}_{k,gl.}^{\ell^T} \left[\bar{\boldsymbol{\theta}}_k - \bar{\boldsymbol{\theta}}_k^\ell \right] \right\} \le \boldsymbol{\sigma}_k^\ell$$

$$k \in K, \quad \ell = 0, \ldots, L \tag{39}$$

$$\begin{bmatrix} Y_i \\ \boldsymbol{T}_{k,q}^\ell \left\{ \nabla \boldsymbol{q}_{k,i|t_{k,j} \in \mathfrak{M}_k}^T \left[\bar{\boldsymbol{\theta}}_k - \bar{\boldsymbol{\theta}}_k^\ell \right] \right\} \le \boldsymbol{\sigma}_k^\ell, \\ \boldsymbol{T}_{k,dj.}^\ell \left\{ \boldsymbol{h}_{k,dj.}^\ell + \nabla \boldsymbol{h}_{k,dj.}^{\ell^T} \left[\bar{\boldsymbol{\theta}}_k - \bar{\boldsymbol{\theta}}_k^\ell \right] \right\} \le \boldsymbol{\sigma}_k^\ell, \\ k \in K, \quad \ell \in K_{i,L}, \quad b_i = \gamma_i, \end{bmatrix} \vee \begin{bmatrix} \neg Y_i \\ \bar{\bar{\boldsymbol{B}}}_{k,i} \bar{\boldsymbol{\theta}}_k = \boldsymbol{0}, \\ k \in K \\ b_i = 0 \end{bmatrix} \tag{40}$$

$$i = 1, \ldots, n_Y$$

$$\Omega(\boldsymbol{Y}) = \text{True} \tag{41}$$

Unfortunately, no dual information in terms of Lagrange multipliers is available to decide about the relaxation of the linearized equations into inequalities due to the implicit treatment in the primal problem. In order to resolve this problem, an idea presented by Kravanja and Grossmann (1996) can be adopted. The solution of the following *linear program* (LP) with fixed Boolean variables and linearized constraints $\boldsymbol{f}_k^L$, $\boldsymbol{h}_{k,gl}^L$, $\boldsymbol{h}_{k,gl}^L$ and $\boldsymbol{q}_{k,i}^L$ problem yields the dual information required to decide about the relaxation of all linearized equations

$$Z_{LP}^L := \min_{\alpha, \bar{\boldsymbol{\theta}}_k} \alpha \tag{42}$$

$$\text{s.t.} \quad \Phi^L + \nabla \Phi^{L^T} \left[\bar{\boldsymbol{\theta}}_k - \bar{\boldsymbol{\theta}}_k^L \right] \le \alpha, \quad k \in K$$

$$\nabla \boldsymbol{f}_k^{L^T} |_{t_{k,j} \in \mathfrak{M}_k} \left[\bar{\boldsymbol{\theta}}_k - \bar{\boldsymbol{\theta}}_k^L \right] = \boldsymbol{0}, \quad k \in K$$

$$\boldsymbol{h}_{k,gl.}^L + \nabla \boldsymbol{h}_{k,gl.}^{L^T} \left[\bar{\boldsymbol{\theta}}_k - \bar{\boldsymbol{\theta}}_k^L \right] \left\{ \begin{matrix} = \\ \le \end{matrix} \right\} \boldsymbol{0}, \quad k \in K$$

$$\left. \begin{matrix} \nabla \boldsymbol{q}_{k,i}^{L^T} |_{t_{k,j} \in \mathfrak{M}_k} \left[\bar{\boldsymbol{\theta}}_k - \bar{\boldsymbol{\theta}}_k^L \right] = \boldsymbol{0}, \quad k \in K, \\ \boldsymbol{h}_{k,dj.}^L + \nabla \boldsymbol{h}_{k,dj.}^{L^T} \left[\bar{\boldsymbol{\theta}}_k - \bar{\boldsymbol{\theta}}_k^L \right] \left\{ \begin{matrix} = \\ \le \end{matrix} \right\} \boldsymbol{0}, \quad k \in K, \\ b_i = \gamma_i, \ i = 1, \ldots, n_Y \end{matrix} \right\} \text{ for } Y_i^L = \text{True}$$

$$\left. \begin{matrix} \bar{\bar{\boldsymbol{B}}}_{k,i} \bar{\boldsymbol{\theta}}_k = \boldsymbol{0}, \quad k \in K, \\ b_i = \gamma_i, \ i = 1, \ldots, n_Y, \end{matrix} \right\} \text{ for } Y_i^L = \text{False}$$

Hence, the relaxation matrices $\boldsymbol{T}_f^L$, $\boldsymbol{T}_{k,gl}^L$, $\boldsymbol{T}_{k,dj}^L$, and $\boldsymbol{T}_{k,q}^L$ can be determined on the basis of the Lagrange multipliers of the LP problem 42.

Initialization

For the initialization of the proposed logic-based algorithm, we need to provide linearizations of all nonlinear constraints contained in the disjunctions to build the first master problem. These linearizations are generated by solving a number of primal problems with fixed sets of Boolean variables. The minimum number of primal problems required and their corresponding Boolean variable values can be determined by solving a *set covering* problem (Türkay and Grossmann, 1996). The set covering problem (Nemhauser and Wolsey, 1999; Williams, 1999) is a special class of MILP problem used for a number of different applications.

For a problem with n_R design alternatives expressed by n_Y Boolean variables, we construct a matrix $\boldsymbol{A} \in \mathbb{R}^{n_R \times n_Y}$ that holds the value 1 if the Boolean variable Y_i is contained in alternative r. Otherwise, the element $\boldsymbol{A}_{r,i}$ is assigned a value of 0. Furthermore, we introduce a vector $\boldsymbol{w} \in \mathbb{R}^{n_R}$ with elements $w_r = 1$, if alternative r is selected, and $w_r = 0$ otherwise. This vector is determined in the following set-covering problem such that a minimum number of entries adopt the value 1

$$\min_{\boldsymbol{w}} \sum_{r=1}^{n_R} w_r \tag{43}$$

$$\text{s.t.} \quad \sum_{r=1}^{n_R} \boldsymbol{A}_{r,i} w_r \ge 1, \quad i = 1, \ldots, n_Y$$

The solution of this problem yields that each disjunction where Y_i is True is included at least once. However, some of them will typically appear more than once. For small problems, the additional numerical effort required for the solution of these primal subproblems will generally be high relative to the overall solution time. The extra demand in terms of computational time will, however, decrease for larger problems for which the number of subproblems to be solved to cover the initialization set is usually much smaller when compared to the complete set of design alternatives.

Software implementation

The proposed solution method is implemented in a software tool named DyOS (DyOS, 2002), that interfaces a number of different optimization and numerical integration routines to modeling platforms that are compliant to the so-called ESO interface definition, such as gPROMS (gPROMS, 2002). DyOS uses an adaptive control vector parameterization approach (Schlegel et al., 2001) and a highly efficient sensitivity solver (Schlegel et al., 2003).

The disjunctive model employed for the case study is implemented in gPROMS. DyOS is used to solve the batch design example. SNOPT (Gill et al., 1998) and CPLEX (ILOG, 2002) are employed as NLP and MILP solvers, respectively. Note that the modification of the solution algorithm required to be able to solve problems with disjunctive differential equations has not yet been integrated in the current version of the implementation.

Illustrative Example Problem

Sørensen and Skogestad (1996) have demonstrated that batch distillation systems can be successfully operated in an inverse mode for several separation tasks. Within this column configuration, the charge pot is located at the top of the column and the product is withdrawn at the bottom. In some cases, it is possible to predict in advance whether regular or inverse operation will lead to minimal energy consumption. However, in a general batch process development problem, it will be difficult to decide *a priori* how a batch distillation should be operated best, especially in cases where the batch column is part of a batch plant or when nonideal multicomponent mixtures are treated. In fact, the discrete decision regarding whether an inverse or a regular batch column is preferable can be covered in a dynamic process model involving disjunctions. Hence, in order to illustrate the design method proposed in this work, we take a closer look at this particular design problem. In particular, an ideal, quaternary separation problem is discussed to illustrate the disjunctive modeling technique and to show how the corresponding optimization problem can be formulated. A similar but simpler design problem has been presented by Oldenburg et al. (2002). The example is solved with the proposed MLDO formulation as well as with a MIDO solution technique.

The disjunctive process model

The example presented here is based on a patent specification (Patent, 1997), where a batch distillation problem involving a ternary mixture is discussed. We have modified this design problem by adding a fourth component. The task is to separate a quaternary mixture consisting of 100 kg *n*-pentane, 100 kg *n*-hexane, 600 kg *n*-heptane, and 200 kg *n*-octane into pure components with a predefined purity of at least $\varphi_i = 0.99\ \text{kg}_i/\text{kg}_{\text{total}}$, $i = 1, \ldots, 4$, with minimum energy demand in a sequence of three batch distillation tasks. The purities χ_i in terms of mole fractions, as employed for this design problem, are easily calculated as $\chi_i = (\varphi_i M_{\text{total}})/M_i$ using the molar masses M_i and M_{total} of each component and of the mixture.

The multistage batch distillation process is operated in one single batch column that can be operated either regularly or inversely in each of the three distillation tasks. For this work, this column is assumed to consist of a fixed number of $N = 10$ theoretical stages plus top and bottom trays with a constant pressure drop of 1 mbar per tray and a constant reboiler heat duty Q_B of 50 kW.

Since each batch distillation stage can be operated either regularly or inversely, and either the residue or the distillate of the first and second stages can be fed to the second and third stages, respectively, we eventually have a total number of 40 structural alternatives that can be used together with the time-varying reflux ratios of the batch stages to minimize the time (here, the batch time is directly proportional to energy demand) required for the complete separation. In this case study, we reduce the complexity of the problem from 40 to 32 design alternatives by considering the withdrawal of only pure components in each sequence of the multistage process. Two of these process alternatives are shown in Figure 1. The first alternative is characterized by three regularly operated process stages, where *n*-pentane, *n*-hexane, and *n*-heptane (components 1–3) are withdrawn as distillate from the top, and *n*-octane (component 4) is the bottom residue of the last stage. The second alternative is operated using both regular and inverse operation. Here, *n*-pentane is withdrawn at the top of the first stage, employing a regular mode of operation. In the second stage, which is also operated regularly, pure *n*-octane is obtained as the bottom residue. Finally, an inverse operation is used to separate *n*-hexane and *n*-heptane in the third stage.

The dynamic process model consists of three parts, the first of which comprises all model equations that hold independently of any discrete decision. The equations belonging to this part of the model are stated in Appendix A.

The second part of the superstructure model comprises the model equations that hold subject to discrete decisions expressed in terms of two-term disjunctions as shown in Eq. 7 or multiple-term disjunctions. The first disjunction, a two-term one, represents the discrete decision regarding whether the column is operated regularly or inversely

$$\begin{bmatrix} Y_1 \\ \\ D^{S_1}(t) \le D^{S_1}_{\mathfrak{U}}, \quad t \in [t_0, t_1], \\ B^{S_1}(t) = 0, \quad t \in [t_0, t_1], \\ H^{S_1}_B(t_0) = 10.325 \text{ kmol}, \\ H^{S_1}_C(t_0) = 0.01 \text{ kmol}, \end{bmatrix} \vee \begin{bmatrix} \neg Y_1 \\ \\ D^{S_1}(t) = 0, \quad t \in [t_0, t_1], \\ B^{S_1}(t) \le B^{S_1}_{\mathfrak{U}}, \quad t \in [t_0, t_1], \\ H^{S_1}_B(t_0) = 0.01 \text{ kmol}, \\ H^{S_1}_C(t_0) = 10.325 \text{ kmol}, \end{bmatrix} \tag{44}$$

Thus, for regular column operation ($Y_1 = \text{True}$), the still pot is charged with the fresh feed, whereas the initial condenser holdup, $H^{S_1}_C(t_0)$, of the first batch process stage S_1 is set to a predefined value of 0.01 kmol. Furthermore, the (positive) control variable distillate stream $D^{S_1}(t)$ is upper bounded by $D^{S_1}_{\mathfrak{U}}$ and the bottom stream $B^{S_1}(t)$ is set to 0. If the first batch stage is operated inversely ($Y_1 = \text{False}$), the right part of the disjunction is obtained analogously.

The decision regarding which component is to be removed as product from the first separation task can be modeled us-

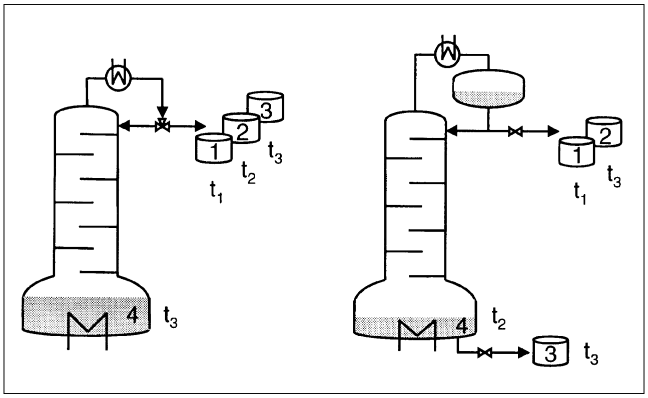

Figure 1. Three-stage batch-distillation process with regular operation (left), regular and inverse operation (right).

ing the following set of multiple-term disjunctions

$$\begin{bmatrix} Y_2 \\ \\ x_{B,4}^{S_1}(t_1) \ge \chi_4, \\ F^{S_2} = Dist^{S_1}(t_1), \\ x_{B,i}^{S_2}(t_1) = x_{C,i,acc.}^{S_1}(t_1), \end{bmatrix} \vee \begin{bmatrix} Y_3 \\ \\ x_{C,1,acc.}^{S_1}(t_1) \ge \chi_1, \\ F^{S_2} = H_B^{S_1}(t_1), \\ x_{B,i}^{S_2}(t_1) = x_{B,i}^{S_1}(t_1), \end{bmatrix}$$

$$\vee \begin{bmatrix} Y_4 \\ \\ x_{C,1}^{S_1}(t_1) \ge \chi_1, \\ F^{S_2} = Bot^{S_1}(t_1), \\ x_{B,i}^{S_2}(t_1) = x_{B,i,acc.}^{S_1}(t_1), \end{bmatrix} \vee \begin{bmatrix} Y_5 \\ \\ x_{B,4,acc.}^{S_1}(t_1) \ge \chi_4, \\ F^{S_2} = H_C^{S_1}(t_1), \\ x_{B,i}^{S_2}(t_1) = x_{C,i}^{S_1}(t_1), \end{bmatrix}$$

$$i = 1, \ldots, 4 \qquad (45)$$

One purity specification χ_i, $i = 1, \ldots, 4$, and a set of stage transition conditions are formulated for each of the disjunctive terms (Eq. 45) to decide which component is to be withdrawn either at the top (indicated by the suffix C) or at the bottom (suffix B). Furthermore, the product may be collected in an accumulator (suffix *acc.*) or withdrawn as residue at the top or the bottom of the column. The transfer of the intermediate product (holdup F^{S_2}, concentrations $x_{B,i}^{S_2}(t_1)$, $i = 1, \ldots, 4$) to the second batch stage S_2 is expressed in terms of stage transition conditions enforced at t_1, the final time of stage 1. These constraints are directly related to the purity specifications enforced within the disjunctions through the design context. The disjunctions for the second and third column sequence are stated in a similar way. A summary of all disjunctions used within the process model is provided in Appendix A.

The disjunctions stated in Eqs. 44 and 45 are related to each other by propositional logic constraints, the third part of the superstructure model

$$\begin{gathered} Y_1 \Rightarrow Y_2 \vee Y_3 \\ \neg Y_1 \Rightarrow Y_4 \vee Y_5 \\ Y_2 \underline{\vee} Y_3 \underline{\vee} Y_4 \underline{\vee} Y_5 \end{gathered} \qquad (46)$$

In this way, certain combinations of Boolean variable values are excluded from the set of feasible solutions in the optimization problem (Eqs. 47). These conditions are stated for each batch stage S_k, $k = 1, 2, 3$, to relate the mode of operation to potential purity specifications and stage transition conditions that map intermediate products of one stage to the other. According to Eqs. 46, at least one of the Boolean variables, Y_2 or Y_3, is True, if the first process stage is operated regularly, while Y_4 or Y_5 are True for an inverse operation. The last equation in Eqs. 46 is used to express that exactly one Boolean variable of $Y_2 - Y_5$ is True. Similar logical expressions are stated for the disjunctions of the batch stages 2 and 3. They are presented in Appendix A in conjunction with the disjunctive constraints.

The complete disjunctive process model (cf. Eqs. A1–A17), which is summarized in Appendix A, comprises decision variables of a different nature. The (positive) control variables distillate and bottom streams in each stage $D^{S_k}(t)$, $B^{S_k}(t)$, $t \in [t_{k-1}, t_k]$, $k = 1, 2, 3$, represent time-variant degrees of freedom. The initial values of the liquid holdups in the condenser $H_C^{S_k}(t_{k-1})$ and the still pot $H_B^{S_k}(t_{k-1})$ for each of the batch stages, the amounts of intermediate product fed to the second and third stages F^{S_k}, $k = 2, 3$, and the mole fractions $x_{B,i}^{S_k}(t_{k-1})$, $k = 2, 3$, $i = 1, \ldots, 4$, of the second and third stages, are time-invariant parameters and free initial values of differential state variables. Besides these continuous degrees of freedom, the problem comprises 21 discrete variables in terms of the Boolean variables $Y_1, Y_2, \ldots, Y_{21}$, which are, however, interrelated by logical expressions (cf. Eqs. A13–A17). In this way, the combinatorial complexity is reduced from 2^{21} to $2^5 = 32$ potential combinations of Boolean variables. Finally, the mixed-logic dynamic optimization problem reads

$$\min_{\substack{D^{S_k}(t), B^{S_k}(t) \\ H_C^{S_k}(t_{k-1}),\, H_B^{S_k}(t_{k-1}),\, t_k,\, k=1,2,3 \\ F^{S_k},\, x_{B,i}^{S_k}(t_{k-1}),\, k=2,3,\, i=1,\ldots,4 \\ Y_j,\, j=1,\ldots,n_Y}} \sum_{k=1}^{3} t_k$$

$$\text{s.t.} \quad \text{Eqs. A1–A17} \qquad (47)$$

The multistage batch process model as well as the disjunctive constraints (Eqs. A1–A15) have been implemented in a gPROMS model file, which comprises 330 DAES per page without counting linear stream equations. Note that this example with a low combinatorial complexity of 32 design alternatives has been selected to evaluate and illustrate the modeling and solution method and not to show a large-scale application.

Logic-based solution

In order to initialize the MLDO algorithm, we have to determine a set of primal subproblems, such that at least one linearization is generated for each nonlinear constraint contained in any of the disjunctions. The smallest set possible is determined by solving the set covering problem in Eq. 43 stated in the previous section. This set comprises eight design alternatives, as shown in Table 1. Note that different sets of design alternatives can be used as the initialization, since the

Table 1. Initialization: Design Alternatives

	Operat. Strat. S_1	Interm. Product of S_1 Fed to S_2	Operat. Strat. S_2	Interm. Product of S_2 Fed to S_3	Operat. Strat. S_3
1	Regular	Top accumulation	Regular	Top accumulation	Regular
2	Regular	Bottom residue	Regular	Top accumulation	Regular
3	Inverse	Bottom accumulation	Inverse	Top residue	Inverse
4	Inverse	Top residue	Inverse	Top residue	Inverse
5	Regular	Top accumulation	Regular	Bottom residue	Regular
6	Regular	Bottom residue	Regular	Bottom residue	Regular
7	Inverse	Bottom accumulation	Inverse	Bottom accumulation	Inverse
8	Inverse	Top residue	Inverse	Bottom accumulation	Inverse

solution to the set covering problem is not unique. This is, without doubt, a large number when compared to the complete set of 32 alternatives. In fact, if linearizations of these design alternatives are calculated by solving the corresponding primal subproblems to optimality, as in this case study, the additional effort might be considered to be too high. However, the linearizations do not necessarily have to be determined this way. By avoiding solving the full set of subproblems to optimality, and instead by starting with an initial process structure and suboptimizing the remaining subsystems (Kocis and Grossmann, 1989), it would be possible to save a large amount of numerical effort, depending on the case considered. Furthermore, an alternative approach, which eliminates the need for solving a set covering problem and the corresponding NLP subproblems for the problem initialization, would be to linearize around the complete superstructure model for an initial process configuration. This method reduces the computational demand for problems where the set covering problem solution indicates that a very large number of initializing subproblems are to be solved.

However, it is important to note that the ratio between the number of design alternatives contained in the initializing set and the total number of alternatives will usually decrease for more complex problems. As a result, the proposed method will be most efficient for solving problems involving a large number of discrete alternatives, that is, a number that exceeds 32 by a great deal.

For the primal problems, the control variables in each process stage k of the batch separation task are approximated by piecewise constant trial functions on eight equidistant time elements according to Eq. 20. In each primal subproblem iteration, the fixed values of the Boolean variables determine which disjunction is included and which degree of freedom can be eliminated from the problem. As a result, we have only to consider three out of the total number of six control variables in each iteration, since each batch sequence is either operated regularly or inversely, a fact that is expressed mathematically by the two-term disjunctions Y_1, Y_6, and Y_{15}. Furthermore, depending on the process structure selected in the current iteration, only a subset of the purity constraints and stage transition conditions have to be considered in each primal problem. The master problem is formulated according to Eqs. 30–34.

Table 2. Results Obtained with MLDO Algorithm

Major Iteration	0	1	2	3	4	5
Primal problem	Initialization					
Process structure	**1–8**	**1**	**2**	**17**	**18***	**5**
Batch time (h)	—	13.39	14.30	14.37	13.18	14.35
Master problem						
Process structure	**1**	**2**	**17**	**18**	**5**	
Batch time (h)	17.82	18.78	19.53	18.63	19.58	
No. active slacks	14	15	16	16	21	

*Optimal solution.

The results of the batch distillation design problem are summarized in Table 2. Note that the values of the final batch time given in Table 2 coincide with the objective function values for the primal problems only. The monotonically increasing master problem objective function values comprise a penalty term in addition to the final batch time. The optimal solution, design alternative 18 (see Figure 1, right), is found after five major iterations of the MLDO algorithm. The termination criterion for the nonconvex problem is taken from Viswanathan and Grossmann (1990), who stop the algorithm as soon as the primal solutions do not decrease any more. We additionally enforce a lower limit on the number of major iterations $L_{\min} = 3$. Note that the primal problems of major iterations 1, 2, and 5 have already been solved within the initialization procedure. Hence, a direct reuse of the corresponding solution and linearization information required for the master problems is possible.

For a better understanding of the behavior of the proposed algorithm in this nonconvex situation and of the results, we have determined the optimal batch time for each design alternative by a full enumeration of the MLDO problem. Interestingly, this optimization yields a division of the design alternatives into two groups according to the achieved process performance. Sixteen alternatives require a total batch time of 13 to 15 h, whereas the remaining configurations are unfavorable and require significant additional effort, measured in terms of batch time or energy demand, leading to total batch times of 20 to 24 h. A deeper analysis reveals that in each of the unfavorable cases, exactly one batch process period takes a disproportionately long batch time. These long periods occur when *n*-octane is accumulated at high purity in one of the three stages using an inverse operational strategy. This finding is in agreement with the results of Sørensen and Skogestad (1996), who have shown for an ideal binary mixture that a regular mode of operation is preferable if the feed is rich in light component and the product purity is high. On the other hand, the column should be operated inversely if the opposite is true, that is, the feed is rich in the heavy component. This design heuristic holds in our case, since *n*-octane is always present in relatively small amounts in the feed of all batch stages.

From an algorithmic point of view, it is possible to draw two important conclusions on the basis of the information we gathered from the enumeration of the MLDO problem. First, design alternative 18 determined by the MLDO algorithm is indeed the best process configuration attainable. Hence, the algorithm has found the global optimum (assuming the primal problems have been solved globally) of the nonconvex problem in this case. Second, in each of the major iterations, the algorithm has proposed exclusively favorable design strategies, a fact that is rather important from a practical point of view. Without doubt, further case studies will be required to verify these results.

In order to show that different results are indeed obtained with a MIDO formulation and a standard OA solution algorithm, we have solved the design problem again using this method.

Transformation into a MIDO problem and solution

In order to be able to apply a MIDO solution algorithm to solve the problem in Eqs. A1−A17, we have to transform the mixed-logic dynamic optimization problem into a mixed-integer problem. We represent each disjunction contained in Eqs. A7−A13 using big-M constraints and replace the Boolean variables Y_i by binary variables y_i. Due to space limitations, we only state the transformation of two disjunctions, that is, Eq. 46 and the first term of Eq. 45

$$\begin{aligned} &D^{S_1}(t) \le D^{S_1}_{\mathfrak{u}} y_1, \quad t \in [t_0, t_1] \\ &B^{S_1}(t) \le B^{S_1}_{\mathfrak{u}}(1-y_1), \quad t \in [t_0, t_1] \\ &H^{S_1}_B(t_0) = 0.01 + 10.315 y_1 \\ &H^{S_1}_C(t_0) = 0.01 + 10.315(1-y_1) \end{aligned} \tag{48}$$

$$\begin{aligned} &x^{S_1}_{B,4}(t_1) \ge \chi_4 y_2 \\ &M^F_{\mathfrak{L}}(1-y_2) \le F^{S_2} - Dist^{S_1}_C(t_1) \le M^F_{\mathfrak{u}}(1-y_2) \\ &M^x_{\mathfrak{L}}(1-y_2) \le x^{S_2}_{B,i}(t_1) - x^{S_1}_{C,i,acc.}(t_1) \le M^x_{\mathfrak{u}}(1-y_2) \\ &\qquad i = 1,\ldots,4 \end{aligned} \tag{49}$$

The remaining big-M constraints can be deduced in a straightforward manner. The complete mixed-integer model representation is found in Appendix B.

As a result, we can state the following mixed-integer dynamic optimization problem

$$\min_{\substack{D^{S_k}(t),\, B^{S_k}(t),\\ H^{S_k}_C(t_{k-1}),\, H^{S_k}_B(t_{k-1}),\, t_k,\, k=1,2,3\\ F^{S_k},\, x^{S_k}_{B,i}(t_{k-1}),\, k=2,3,\, i=1,\ldots,4\\ y_j,\, j=1,\ldots,n_y}} \sum_{k=1}^{3} t_k$$

$$\text{s.t.} \quad \text{Eqs. A1−A7, B1−B10} \tag{50}$$

Once the mixed-logic dynamic optimization problem has been converted into a mixed-integer dynamic optimization problem, we are ready to apply a MIDO solution technique, which in this case is based on the OA algorithm. Since the problem under consideration is nonconvex, we have to use the Augmented Penalty extension (Viswanathan and Grossmann, 1990) to avoid infeasible master subproblems. The first primal problem is initialized with the classic operational strategy of three batch column stages, that is, a regular mode of operation with product withdrawal at the top and the feed charged to the bottom of each stage (design alternative 6, see Table 1).

The control variables of each batch process stage are again approximated by piecewise constant trial functions on eight equidistant time elements. The optimal solution is found after four major iterations. The termination criterion is identical to the one used for the solution of the MLDO problem. In order to ensure that the master subproblems contain sufficient information about the process superstructure in terms of accumulated linearizations, a lower bound on the number of major iterations, $L_{\min} = 3$, has been imposed, as in the logic-based solution.

Table 3. Results Obtained with MIDO Algorithm

Major Iteration	0	1	2	3	4
Primal problem					
Process structure	**6**	**22**	**2**	**18***	**9**
Batch time (h)	14.80	22.96	14.30	13.18	14.60
Master problem					
Process structure	**22**	**2**	**18**	**9**	
Batch time (h)	11.96	10.47	16.98	9.16	
No. active slacks	—	13	35	55	

*Optimal solution.

As the result in Table 3 shows, the best configuration (design alternative 18) is also found by the MIDO algorithm. It even requires a smaller number of major iterations as the MLDO solution algorithm. Hence, relatively few iterations were necessary to figure out the best solution out of 32 potential candidates. However, a qualitative difference in the solutions obtained with the two different solution approaches is nevertheless easy to identify. Whereas the number of active slack variables (slacks that are strictly greater than 0) grows quite rapidly throughout the major iterations of the MIDO solution, this number stays reasonably small when the MLDO algorithm is applied. This is explained by the fact that linearizations of nonconvex constraints of nonexisting parts of the batch process superstructure model are not accumulated in the master problems of the MLDO algorithm.

To start on the basis of a set covering problem can be seen as an advantage of the MLDO algorithm, since this initialization scheme provides a better representation of the overall design problem than an arbitrary start configuration, for which some of the linearizations are generated at physically meaningless operating conditions. This point needs to be analyzed on the basis of further test scenarios. Furthermore, rigorous comparisons of the total CPU time spent for both of the algorithms have not yet been carried out. The example problem was solved on a 1.2-GHz personal computer. The computing time for both algorithms was almost entirely spent solving the primal subproblems. Essentially, this means that the performance of both algorithms strongly depends on the complexity of the primal subproblems and, thus, on the efficiency of the underlying dynamic optimization solution methods.

Conclusions and Future Perspectives

In this contribution, the configuration and sequencing of batch distillation processes is addressed by formulating a mixed-logic dynamic optimization problem, for which a tailored solution method is proposed. Here, all potential design candidates are comprised in the mathematical problem formulation in terms of a disjunctive dynamic process model, which is shown to be a rather natural way of representing batch distillation design problems. Moreover, the disjunctive problem formulation provides a number of favorable properties that can be exploited by a logic-based solution technique based on OA in order to efficiently and reliably solve design problems of the class considered.

The solution of MLDO/MIDO problems involving dynamic process models with several thousands of DAEs is known to be a difficult task. On the one hand, the problem size and complexity lead to a very high computational effort. On the other hand, due to the nonconvexity of the problem, no guarantee can be given that the solution obtained is globally optimal or at least close to the global optimum. The logic-based solution method proposed in this article allows us to efficiently solve primal subproblems, that is, large-scale dynamic optimization problems, with a reduced set of free parameters as well as constraints. The restriction that the solutions obtained can only be guaranteed to be locally optimal cannot be circumvented by applying the logic-based modeling and solution strategy. However, the fact that linearizations are only generated at operating conditions where the corresponding part of the process model has been active in a primal problem, can help improve the quality of the master problems. Furthermore, case studies will be required to analyze this point in greater detail.

A further step within this research project will be made by verifying the results obtained for a batch distillation design problem with a fully detailed dynamic process model without simplifying assumptions. We will then investigate a more complex batch separation problem involving an azeotropic mixture in order to analyze the properties of the method in conjunction with nonideal problems, where the use of simplified process models or design heuristics is insufficient. In this context, the treatment of disjunctive differential equations using the proposed modification of the solution algorithm has to be analyzed. Moreover, the MLDO algorithm needs to be developed further. Here, we intend to improve the initialization of the primal subproblems, which was found to be a very challenging problem when solving the example problem. The fact that each new design alternative proposed by a master problem will usually change the complete process dynamics constitutes the difficulty of this task.

Acknowledgments

Financial support from Bayer Technology Services, Bayer AG, Leverkusen, and from Deutsche Forschungsgemeinschaft under the Gottfried Wilhelm Leibniz program is gratefully acknowledged. Additionally, we thank Dominique Bonvin for fruitful discussions, and Robert Grosch for his work in the early stage of the project.

Literature Cited

Abel, O., A. Helbig, and W. Marquardt, "Productivity Optimization of an Industrial Semi-Batch Polymerization Reactor Under Safety Constraints," *J. Process Control*, **10**, 351 (2000).

Adjiman, C. S., I. P. Androulakis, and C. A. Floudas, "Global Optimization of Mixed-Integer Nonlinear Problems," *AIChE J.*, **46**(9), 1769 (2000).

Allgor, R. J., and P. I. Barton, "Mixed-Integer Dynamic Optimization," *Comput. Chem. Eng.*, **21**, S451 (1997).

Avraam, M. P., N. Shah, and C. C. Pantelides, "Modelling and Optimisation of General Hybrid Systems in the Continuous Time Domain," *Comput. Chem. Eng.*, **22**, S221 (1998).

Balas, E., "Disjunctive Programming and a Hierarchy of Relaxations of Discrete Optimization Problems," *SIAM J. Algebraic Discrete Methods*, **6**, 466 (1985).

Bansal, V., J. D. Perkins, and E. N. Piskopoulos, "A Case Study in Simultaneous Design and Control Using Rigorous, Mixed-Integer Dynamic Optimization Models," *Ind. Eng. Chem. Res.*, **41**(4), 760 (2002).

Barrera, M. D., and L. B. Evans, "Optimal Design and Operation of Batch Processes," *Chem. Eng. Commun.*, **82**, 45 (1989).

Beaumont, N., "An Algorithm for Disjunctive Programs," *Eur. J. Optim. Res.*, **48**, 362 (1990).

Bhatia, T. K., and L. T. Biegler, "Dynamic Optimization in the Design and Scheduling of Multiproduct Batch Plants," *Ind. Eng. Chem. Res.*, **35**, 2234 (1996).

Büskens, C., *Optimierungsmethoden und Sensitivitätsanalyse für optimale Steuerprozesse mit Steuer- und Zustandsbeschränkungen*, PhD Thesis, Westfälische Wilhelms-Universität Münster (1998).

Buss, M., O. von Stryk, R. Bulirsch, and G. Schmidt, "Towards Hybrid Optimal Control," *at-Automatisierungstechnik*, **48**(9), 448 (2000).

Charalambides, M. S., N. Shah, and C. C. Pantelides, "Synthesis of Reaction/Distillation Processes Using Detailed Dynamic Models," *Comput. Chem. Eng.*, **S19**, 167 (1995).

Cuthrell, J. E., and L. T. Biegler, "On the Optimization of Differential-Algebraic Process Systems," *AIChE J.*, **33**(8), 1257 (1987).

Duran, M., and I. E. Grossmann, "A Mixed-Integer Nonlinear Programming Algorithm for Process Systems Synthesis," *AIChE J.*, **32**(4), 592 (1986).

DyOS, *DyOS User Manual, Release 2.1*, Lehrstuhl für Prozesstechnik, RWTH Aachen, Aachen, Germany (2002).

Fieg, G., W. Gutermuth, W. Kothe, H.-H. Mayer, S. Nagel, H. Wendeler, and G. Wozny, "A Standard Interface for Use of Thermodynamics in Process Simulation," *Comput. Chem. Eng.*, **19**, S317 (1995).

Fletcher, R., and S. Leyffer, "Solving Mixed Integer Nonlinear Programs by Outer Approximation," *Math. Program.*, **66**, 327 (1994).

Floudas, C. A., *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*, Oxford Univ. Press, New York (1995).

Geoffrion, A. M., "Generalized Benders Decomposition," *J. Optimiz. Theory Appl.*, **10**(4), 237 (1972).

Gill, P. E., W. Murray, and M. A. Saunders, "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," Tech. Rep. Stanford Univ., Stanford, CA (1998).

gPROMS, *gPROMS User Guide (Release 2.1.1)*, Process Systems Enterprise Ltd., London, UK (2002).

Grossmann, I. E., "Review of Nonlinear Mixed-Integer and Disjunctive Programming Techniques," *Optim. Eng.*, **3**, 227 (2002).

Grossmann, I. E., and J. Hooker, "Logic Based Approaches for Mixed Integer Programming Models and Their Application in Process Synthesis," *Foundations of Computer-Aided Process Design 323*, M. Malone, J. Trainham, and B. Carnhan, eds., AIChE Symp. Ser., CACHE Publications, New York, p. 70 (2000).

ILOG, *CPLEX 7.5 User's Manual*, Ilog, Inc., France (2002).

Ishikawa, T., Y. Natori, L. Liberis, and C. C. Pantelides, "Modelling and Optimisation in an Industrial Batch Process for the Production of Dioctyl Phthalate," *Comput. Chem. Eng.*, **21**, 1239 (1997).

Kim, K.-J., and U. M. Diwekar, "New Era in Batch Distillation: Computer Aided Analysis, Optimal Design and Control," *Rev. Chem. Eng.*, **17**(2), 111 (2001).

Kocis, G. R., and I. E. Grossmann, "Relaxation Strategy for the Structural Optimization of Process Flow Sheets," *Ind. Eng. Chem. Res.*, **26**, 1869 (1987).

Kocis, G. R., and I. E. Grossmann, "A Modelling and Decomposition Strategy for the MINLP Optimization of Process Flowsheets," *Comput. Chem. Eng.*, **13**(7), 797 (1989).

Kraft, D., "On Converting Optimal Control Problems into Nonlinear Programming Problems," *Comput. Math. Prog.*, **15**, 261 (1985).

Kravanja, Z., and I. E. Grossmann, "A Computational Approach for the Modeling/Decomposition Strategy in the MINLP Optimization of Process Flowsheets with Implicit Models," *Ind. Eng. Chem. Res.*, **35**(6), 2065 (1996).

Lee, S., and I. E. Grossmann, "New Algorithms for Nonlinear Generalized Disjunctive Programming," *Comput. Chem. Eng.*, **24**, 2125 (2000).

Leineweber, D. B., I. Bauer, H. G. Bock, and J. P. Schlöder, "An Efficient Multiple Shooting Based Reduced SQP Strategy for Large-Scale Dynamic Process Optimization—Part I: Theoretical Aspects," *Comput. Chem. Eng.*, **27**, 157 (2003).

Mujtaba, I. M., and S. Macchietto, "Simultaneous Optimization of Design and Operation of Multicomponent Batch Distillation Col-

umn—Single and Multiple Separation Duties," *J. Process Control*, **6**(1), 27 (1996).
Nemhauser, G. L., and L. A. Wolsey, *Integer and Combinatorial Optimization*, Wiley, New York (1999).
Oldenburg, J., W. Marquardt, D. Heinz, and D. B. Leineweber, "Mixed-Logic Dynamic Optimization Applied to Configuration and Sequencing of Batch Distillation Processes," *European Symposium on Computer Aided Process Engineering—12*, J. Grievink and J. van Schijndel, eds., Elsevier, Amsterdam, p. 283 (2002).
Patent, *Verfahren zur Durchführung von destillativen Trennungen in diskontinuierlicher Fahrweise*, Holder of Patent; BASF AG, Patent No. EP0638345B1, Eur. Patent Office (1997).
Raman, R., and I. E. Grossmann, "Integration of Logic and Heuristic Knowledge in MINLP Optimization for Process Synthesis," *Comput. Chem. Eng.*, **16**(3), 155 (1992).
Raman, R., and I. E. Grossmann, "Modelling and Computational Techniques for Logic Based Integer Programming," *Comput. Chem. Eng.*, **18**(7), 563 (1994).
Salomone, H. E., and O. A. Iribarren, "Posynomial Modeling of Batch Plants: A Procedure to Include Process Decision Variables," *Comput. Chem. Eng.*, **16**(3), 173 (1992).
Schlegel, M., T. Binder, A. Cruse, J. Oldenburg, and W. Marquardt, "Component-Based Implementation of a Dynamic Optimization Algorithm Using Adaptive Parameterization," *European Symposium on Computer Aided Process Engineering—11*, R. Gani and S. Jørgensen, eds., Elsevier, Amsterdam, p. 1071 (2001).
Schlegel, M., W. Marquardt, R. Ehrig, and U. Nowak, "Sensitivity Analysis of Linearly-Implicit Differential-Algebraic Systems by One-Step Extrapolation," *Appl. Numer. Math.*, in press (2003).
Schweiger, C. A., and C. A. Floudas, "Interaction of Design and Control: Optimization with Dynamic Models," *Optimal Control: Theory, Algorithms, and Applications*, W. Hager and P. Paradolos, eds., Kluwer, Dordrecht, The Netherlands, p. 388 (1997).
Sharif, M., N. Shah, and C. C. Pantelides, "On the Design of Multicomponent Batch Distillation Columns," *Comput. Chem. Eng.*, **S22**(Suppl.), 69 (1998).
Sørensen, E., and S. Skogestad, "Comparison of Regular and Inverted Batch Distillation," *Comput. Chem. Eng.*, **22**(51), 4949 (1996).
Türkay, M., and I. E. Grossmann, "Logic-Based MINLP Algorithms for the Optimal Synthesis of Process Networks," *Comput. Chem. Eng.*, **20**(8), 959 (1996).
Türkay, M., and I. E. Grossmann, "Tight Mixed-Integer Optimization Models for the Solution of Linear and Nonlinear Systems of Disjunctive Equations," *Comput. Chem. Eng.*, **22**, 1229 (1998).
Viswanathan, J., and I. E. Grossmann, "A Combined Penalty Function and Outer-Approximation Method for MINLP Optimization," *Comput. Chem. Eng.*, **14**(7), 769 (1990).
Williams, H. P., *Model Building in Mathematical Programming*, Wiley, Chichester, England (1999).
Yeomans, H., and I. E. Grossmann, "Disjunctive Programming Models for the Optimal Design Distillation Columns and Separation Sequences," *Ind. Eng. Chem. Res.*, **39**, 1637 (2000).

Appendix A: Basic Batch Distillation Process Model

In this section, we present all the model equations of the process model, which hold independently of any discrete decision. These model equations represent the equipment modules available for the separation: a batch distillation column employed throughout three subsequent batch stages, including condenser, still pot, reflux, and reboil valves as well as product accumulators. Each of these three models represents one batch process stage, indicated by the superscripts S_1, S_2, and S_3. Since the basic batch column model is identical for all stages, we here omit stating the stage superscripts explicitly. The process model (Eqs. A1–A4) is based on common simplifying assumptions, that is, negligible vapor holdup throughout the column, constant liquid holdups H_j of 0.001 kmol on the trays $j=1,\ldots,N=10$, and constant molar overflow. A simplified enthalpy balance is formulated for the reboiler to calculate the column vapor rate V_B over time. We here assume that the time derivative of the liquid phase enthalpy and the difference between the liquid phase enthalpies between tray N and the still pot are small. Thus, Eq. A3 is obtained by substitution of the molar balance and by neglecting the terms $L_N(h_N^{liq}-h_B^{liq})$ and $(dh_B^{liq}/dt)H_B$. All physical properties were calculated using the physical properties package IKCAPE (Fieg et al., 1995), which is connected to the gPROMS model via a so-called foreign object interface.

Condenser, reflux drum and valve

$$\begin{aligned}
\frac{dH_C}{dt} &= V_1 - L_0 - D\\
\frac{dH_C x_{C,i}}{dt} &= V_1 y_{C,i} - L_0 x_{C,i} - D x_{C,i}\\
R &= \frac{L_0}{V_1}\\
y_{C,i}\, p &= x_{C,i}\, p^s_{C,i}\\
p^s_{C,i} &= f_{Ant,i}(T_C), \quad i=1,\ldots,4\\
\sum_{i=1}^{4} y_{C,i} &= 1
\end{aligned} \tag{A1}$$

Trays

$$\begin{aligned}
H_j\frac{dx_{j,i}}{dt} &= L_{j-1}x_{j-1,i} + V_{j+1}y_{j+1,i} - L_j x_{j,i} - V_j y_{j,i}\\
V_{j+1} &= V_j\\
0 &= L_{j-1} + V_{j+1} - L_j - V_j\\
y_{j,i}\, p &= x_{j,i}\, p^s_{j,i}\\
p^s_{j,i} &= f_{Ant,i}(T_j), \quad j=1,\ldots,N, \quad i=1,\ldots,4\\
\sum_{i=1}^{4} y_{j,i} &= 1, \quad j=1,\ldots,N
\end{aligned} \tag{A2}$$

Reboiler, bottom tray, and reflux valve

$$\begin{aligned}
\frac{dH_B}{dt} &= L_N - V_B - B\\
\frac{dH_B x_{B,i}}{dt} &= L_N x_{N,i} - V_B y_{B,i} - B x_{B,i}\\
R_B &= \frac{V_B}{L_N}\\
0 &= -V_B\left(h_B^{vap} - h_B^{liq}\right) + Q_B\\
y_{B,i}\, p &= x_{B,i}\, p^s_{B,i}\\
p^s_{B,i} &= f_{Ant,i}(T_B), \quad i=1,\ldots,4\\
\sum_{i=1}^{4} y_{B,i} &= 1
\end{aligned} \tag{A3}$$

Accumulator for bottom and top product flows

$$\frac{dDist}{dt} = D,\ Dist(t_{k-1}) = 0$$

$$\frac{dx_{C,i,acc}}{dt} = \frac{x_{C,i}D}{Dist}, \quad x_{C,i,acc.}(k-1) = 0$$

$$\frac{dBot}{dt} = B,\ Bot(t_{k-1}) = 0$$

$$\frac{dx_{B,i,acc}}{dt} = \frac{x_{B,i}B}{Bot}, \quad x_{B,i,acc.}(t_{k-1}) = 0, \quad i = 1,\dots,4$$

$$k = 1, 2, 3 \quad \text{(A4)}$$

Top holdup equation

$$V_1 - L_0 - D = -B \quad \text{(A5)}$$

Equation A5, which is only valid under the assumption that the tray and reflux drum holdups are time-invariant, is used to replace the missing holdup relation in Eqs. A1. For a regular column operation, where $B = 0$, we have a time-invariant holdup in the top by substitution of Eq. A5 into the top mass balance. If the column is operated inversely ($D = 0$), we have $dH_C/dt = -B$ and $dH_B/dt = 0$. The latter equation is obtained by substitution of Eq. A5 into the bottom mass balance (cf. Eqs. A3). The relation between B, D, and the mode operation is expressed through the disjunctions stated in Eqs. A8, A10, and A12.

The initial conditions of the first batch stage, S_1, are specified as follows

$$x_{B,1}^{S_1}(t_0) = 0.13, \quad x_{B,2}^{S_1}(t_0) = 0.11, \quad x_{B,3}^{S_1}(t_0) = 0.58$$

$$x_{B,4}^{S_1}(t_0) = 0.18 \quad \text{(A6)}$$

We further assume that all trays, condenser, and still pot of all three batch stages are fed with a mixture of equivalent composition. Hence, the corresponding mole fractions follow from

$$x_{C,i}^{S_k}(t_{k-1}) = x_{B,i}^{S_k}(t_{k-1})$$

$$x_{j,i}^{S_k}(t_{k-1}) = x_{B,i}^{S_k}(t_{k-1}), \quad j = 1,\dots,N, \quad i = 1,\dots,4$$

$$k = 1,2,3 \quad \text{(A7)}$$

Disjunctive constraints of the process model

This section lists all model equations that hold subject to discrete decisions expressed in terms of two-term disjunctions, as shown in Eq. 7, or multiple-term disjunctions.

For the sake of completeness, we start again with the disjunctions related to the Boolean variables $Y_1 - Y_5$

$$\begin{bmatrix} Y_1 \\ \\ D^{S_1}(t) \le D_{\mathfrak{U}}^{S_1}, \quad t \in [t_0,t_1], \\ B^{S_1}(t) = 0, \quad t \in [t_0,t_1], \\ H_B^{S_1}(t_0) = 10.325 \text{ kmol}, \\ H_C^{S_1}(t_0) = 0.01 \text{ kmol}, \end{bmatrix} \vee \begin{bmatrix} \neg Y_1 \\ \\ D^{S_1}(t) = 0, \quad t \in [t_0,t_1], \\ B^{S_1}(t) \le B_{\mathfrak{U}}^{S_1}, \quad t \in [t_0,t_1], \\ H_B^{S_1}(t_0) = 0.01 \text{ kmol}, \\ H_C^{S_1}(t_0) = 10.325 \text{ kmol}, \end{bmatrix} \quad \text{(A8)}$$

$$\begin{bmatrix} Y_2 \\ \\ x_{B,4}^{S_1}(t_1) \ge \chi_4, \\ F^{S_2} = Dist^{S_1}(t_1), \\ x_{B,i}^{S_2}(t_1) = x_{C,i,acc.}^{S_1}(t_1), \end{bmatrix} \vee \begin{bmatrix} Y_3 \\ \\ x_{C,1,acc.}^{S_1}(t_1) \ge \chi_1, \\ F^{S_2} = H_B^{S_1}(t_1), \\ x_{B,i}^{S_2}(t_1) = x_{B,i}^{S_1}(t_1), \end{bmatrix} \vee \begin{bmatrix} Y_4 \\ \\ x_{C,1}^{S_1}(t_1) \ge \chi_1, \\ F^{S_2} = Bot^{S_1}(t_1), \\ x_{B,i}^{S_2}(t_1) = x_{B,i,acc.}^{S_1}(t_1), \end{bmatrix} \vee \begin{bmatrix} Y_5 \\ \\ x_{B,4,acc.}^{S_1}(t_1) \ge \chi_4, \\ F^{S_2} = H_C^{S_1}(t_1), \\ x_{B,i}^{S_2}(t_1) = x_{C,i}^{S_1}(t_1), \end{bmatrix}$$

$$i = 1,\dots,4 \quad \text{(A9)}$$

A further (two-term) disjunction is used to decide about the mode of operation in batch stage 2

$$\begin{bmatrix} Y_6 \\ \\ D^{S_2}(t) \le D_{\mathfrak{U}}^{S_2}, \quad t \in [t_1,t_2], \\ B^{S_2}(t) = 0, \quad t \in [t_1,t_2], \\ H_B^{S_2}(t_1) = F^{S_2}, \\ H_C^{S_2}(t_1) = 0.01 \text{ kmol}, \end{bmatrix} \vee \begin{bmatrix} \neg Y_6 \\ \\ D^{S_2}(t) = 0, \quad t \in [t_1,t_2], \\ B^{S_2}(t) \le B_{\mathfrak{U}}^{S_2}, \quad t \in [t_1,t_2], \\ H_B^{S_2}(t_1) = 0.01 \text{ kmol}, \\ H_C^{S_2}(t_1) = F^{S_2}. \end{bmatrix} \quad \text{(A10)}$$

Purity specifications and stage transition conditions between batch stages 2 and 3 are contained in the multiple-term disjunctions stated below

$$\begin{bmatrix} Y_7 \\ \\ x_{B,4}^{S_2}(t_2) \ge \chi_4, \\ F^{S_3} = Dist^{S_2}(t_2), \\ x_{B,i}^{S_3}(t_2) = x_{C,i,acc.}^{S_2}(t_2), \end{bmatrix} \vee \begin{bmatrix} Y_8 \\ \\ x_{C,2,acc.}^{S_2}(t_2) \ge \chi_2, \\ F^{S_3} = H_B^{S_2}(t_2), \\ x_{B,i}^{S_3}(t_2) = x_{B,i}^{S_2}(t_2), \end{bmatrix}$$

$$\vee \begin{bmatrix} Y_9 \\ \\ x_{C,2}^{S_2}(t_2) \ge \chi_2, \\ F^{S_3} = Bot^{S_2}(t_2), \\ x_{B,i}^{S_3}(t_2) = x_{B,i,acc.}^{S_2}(t_2), \end{bmatrix} \vee \begin{bmatrix} Y_{10} \\ \\ x_{B,4,acc.}^{S_2}(t_2) \ge \chi_4, \\ F^{S_3} = H_C^{S_2}(t_2), \\ x_{B,i}^{S_3}(t_2) = x_{C,i}^{S_2}(t_2), \end{bmatrix}$$

$$\vee \begin{bmatrix} Y_{11} \\ \\ x_{B,3}^{S_2}(t_2) \ge \chi_3, \\ F^{S_3} = Dist^{S_2}(t_2), \\ x_{B,i}^{S_3}(t_2) = x_{C,i,acc.}^{S_2}(t_2), \end{bmatrix} \vee \begin{bmatrix} Y_{12} \\ \\ x_{C,1,acc.}^{S_2}(t_2) \ge \chi_1, \\ F^{S_3} = H_B^{S_2}(t_2), \\ x_{B,i}^{S_3}(t_2) = x_{B,i}^{S_2}(t_2), \end{bmatrix}$$

$$\vee \begin{bmatrix} Y_{13} \\ \\ x_{C,1}^{S_2}(t_2) \ge \chi_1, \\ F^{S_3} = Bot^{S_2}(t_2), \\ x_{B,i}^{S_3}(t_2) = x_{B,i,acc.}^{S_2}(t_2), \end{bmatrix} \vee \begin{bmatrix} Y_{14} \\ \\ x_{B,3,acc.}^{S_2}(t_2) \ge \chi_3, \\ F^{S_3} = H_C^{S_2}(t_2), \\ x_{B,i}^{S_3}(t_2) = x_{C,i}^{S_2}(t_2), \end{bmatrix}$$

$$i = 1, \ldots, 4 \quad \text{(A11)}$$

The mode of operation in the third batch stage is modeled by

$$\begin{bmatrix} Y_{15} \\ \\ D^{S_3}(t) \le D_{\mathfrak{u}}^{S_3}, \quad t \in [t_2, t_3], \\ B^{S_3}(t) = 0, \quad t \in [t_2, t_3], \\ H_B^{S_3}(t_2) = F^{S_3}, \\ H_C^{S_3}(t_2) = 0.01 \text{ kmol}, \end{bmatrix}$$

$$\vee \begin{bmatrix} \neg Y_{15} \\ \\ D^{S_3}(t) = 0, \quad t \in [t_2, t_3], \\ B^{S_3}(t) \le B_{\mathfrak{u}}^{S_3}, \quad t \in [t_2, t_3], \\ H_B^{S_3}(t_2) = 0.01 \text{ kmol}, \\ H_C^{S_3}(t_2) = F^{S_3}. \end{bmatrix} \quad \text{(A12)}$$

A further set of multiple-term disjunctions is used to model the withdrawal of the final products

$$\begin{bmatrix} Y_{16} \\ x_{C,2,acc.}^{S_3}(t_3) \ge \chi_2, \\ x_{B,3}^{S_3}(t_3) \ge \chi_3, \end{bmatrix} \vee \begin{bmatrix} Y_{17} \\ x_{C,2}^{S_3}(t_3) \ge \chi_2, \\ x_{B,3,acc.}^{S_3}(t_3) \ge \chi_3, \end{bmatrix}$$

$$\vee \begin{bmatrix} Y_{18} \\ x_{C,3,acc.}^{S_3}(t_3) \ge \chi_3, \\ x_{B,4}^{S_3}(t_3) \ge \chi_4, \end{bmatrix} \vee \begin{bmatrix} Y_{19} \\ x_{C,3}^{S_3}(t_3) \ge \chi_3, \\ x_{B,2,acc.}^{S_3}(t_3) \ge \chi_4, \end{bmatrix}$$

$$\vee \begin{bmatrix} Y_{20} \\ x_{C,1,acc.}^{S_3}(t_3) \ge \chi_1, \\ x_{B,2}^{S_3}(t_3) \ge \chi_2, \end{bmatrix} \vee \begin{bmatrix} Y_{21} \\ x_{C,1}^{S_3}(t_3) \ge \chi_1, \\ x_{B,2,acc.}^{S_3}(t_3) \ge \chi_2. \end{bmatrix} \quad \text{(A13)}$$

The disjunctions stated in this section are related to each other by propositional logic constraints, as we will show in the following section.

Propositional logic constraints of the process model

The first part of the propositional logic constraints have already been introduced. They relate the mode of operation to potential purity specifications of the first batch stage

$$\begin{aligned} &Y_1 \Rightarrow Y_2 \vee Y_3 \\ &\neg Y_1 \Rightarrow Y_4 \vee Y_5 \\ &Y_2 \underline{\vee} Y_3 \underline{\vee} Y_4 \underline{\vee} Y_5 \end{aligned} \quad \text{(A14)}$$

Similar relationships can be deduced for the two subsequent batch stages

$$\begin{aligned} &Y_6 \Rightarrow Y_7 \vee Y_8 \vee Y_{11} \vee Y_{12} \\ &\neg Y_6 \Rightarrow Y_9 \vee Y_{10} \vee Y_{13} \vee Y_{14} \\ &Y_7 \underline{\vee} Y_8 \underline{\vee} Y_9 \underline{\vee} Y_{10} \underline{\vee} Y_{11} \underline{\vee} Y_{12} \underline{\vee} Y_{13} \underline{\vee} Y_{14} \\ &Y_{15} \Rightarrow Y_{16} \vee Y_{18} \vee Y_{20} \\ &\neg Y_{15} \Rightarrow Y_{17} \vee Y_{19} \vee Y_{21} \\ &Y_{16} \underline{\vee} Y_{17} \underline{\vee} Y_{18} \underline{\vee} Y_{19} \underline{\vee} Y_{20} \underline{\vee} Y_{21} \end{aligned} \quad \text{(A15)}$$

Furthermore, purity specifications enforced in one of the batch stages will exclude certain candidates of subsequent stages. Particularly, specifications of the first stage affect the decision in which disjunction of stage 2 and 3 a Boolean variable might become True. The same applies for the second stage that affects stage 3

$$\begin{aligned} &Y_2 \Rightarrow \neg Y_7 \wedge \neg Y_8 \wedge \neg Y_9 \wedge \neg Y_{10} \wedge \neg Y_{18} \wedge \neg Y_{19} \\ &Y_3 \Rightarrow \neg Y_{11} \wedge \neg Y_{12} \wedge \neg Y_{13} \wedge \neg Y_{14} \wedge \neg Y_{20} \wedge \neg Y_{21} \\ &Y_4 \Rightarrow \neg Y_{11} \wedge \neg Y_{12} \wedge \neg Y_{13} \wedge \neg Y_{14} \wedge \neg Y_{20} \wedge \neg Y_{21} \\ &Y_5 \Rightarrow \neg Y_7 \wedge \neg Y_8 \wedge \neg Y_9 \wedge \neg Y_{10} \wedge \neg Y_{18} \wedge \neg Y_{19} \end{aligned} \quad \text{(A16)}$$

$$Y_7 \Rightarrow \neg Y_{18} \wedge \neg Y_{19}$$

$$Y_8 \Rightarrow \neg Y_{16} \wedge \neg Y_{17} \wedge \neg Y_{20} \wedge \neg Y_{21}$$

$$Y_9 \Rightarrow \neg Y_{16} \wedge \neg Y_{17} \wedge \neg Y_{20} \wedge \neg Y_{21}$$

$$Y_{10} \Rightarrow \neg Y_{18} \wedge \neg Y_{19}$$

$$Y_{11} \Rightarrow \neg Y_{16} \wedge \neg Y_{17} \wedge \neg Y_{18} \wedge \neg Y_{19}$$

$$Y_{12} \Rightarrow \neg Y_{20} \wedge \neg Y_{21}$$

$$Y_{13} \Rightarrow \neg Y_{20} \wedge \neg Y_{21}$$

$$Y_{14} \Rightarrow \neg Y_{16} \wedge \neg Y_{17} \wedge \neg Y_{18} \wedge \neg Y_{19} \quad \text{(A17)}$$

Appendix B: MIDO Model Representation

This Appendix provides a reformulation of the disjunctive process model (cf. Eqs. A1–A17) stated in Appendix A. We provide the big-M transformation of the disjunctive and propositional logic constraints (cf. Eqs. A7–A17). Note that the first part of the process model, which is independent of discrete decisions, remains unchanged for a MIDO model representation

$$D^{S_1}(t) \le D^{S_1}_{\mathfrak{U}} y_1, \quad t \in [t_0, t_1]$$

$$B^{S_1}(t) \le B^{S_1}_{\mathfrak{U}}(1-y_1), \quad t \in [t_0, t_1]$$

$$H^{S_1}_B(t_0) = 0.01 + 10.315 y_1$$

$$H^{S_1}_C(t_0) = 0.01 + 10.315(1-y_1) \quad \text{(B1)}$$

$$x^{S_1}_{B,4}(t_1) \ge \chi_4 y_2$$
$$M^F_{\mathfrak{L}}(1-y_2) \le F^{S_2} - Dist^{S_1}(t_1) \le M^F_{\mathfrak{U}}(1-y_2)$$
$$M^x_{\mathfrak{L}}(1-y_2) \le x^{S_2}_{B,i}(t_1) - x^{S_1}_{C,i,acc.}(t_1) \le M^x_{\mathfrak{U}}(1-y_2)$$

$$x^{S_1}_{C,1,acc.}(t_1) \ge \chi_1 y_3$$
$$M^F_{\mathfrak{L}}(1-y_3) \le F^{S_2} - H^{S_1}_B(t_1) \le M^F_{\mathfrak{U}}(1-y_3)$$
$$M^x_{\mathfrak{L}}(1-y_3) \le x^{S_2}_{B,i}(t_1) - x^{S_1}_{B,i}(t_1) \le M^x_{\mathfrak{U}}(1-y_3)$$

$$x^{S_1}_{C,1}(t_1) \ge \chi_1 y_4$$
$$M^F_{\mathfrak{L}}(1-y_4) \le F^{S_2} - Bot^{S_1}(t_1) \le M^F_{\mathfrak{U}}(1-y_4)$$
$$M^x_{\mathfrak{L}}(1-y_4) \le x^{S_2}_{B,i}(t_1) - x^{S_1}_{B,i,acc.}(t_1) \le M^x_{\mathfrak{U}}(1-y_4)$$

$$x^{S_1}_{B,4,acc.}(t_1) \ge \chi_4 y_5$$
$$M^F_{\mathfrak{L}}(1-y_5) \le F^{S_2} - H^{S_1}_C(t_1) \le M^F_{\mathfrak{U}}(1-y_5)$$
$$M^x_{\mathfrak{L}}(1-y_5) \le x^{S_2}_{B,i}(t_1) - x^{S_1}_{C,i}(t_1) \le M^x_{\mathfrak{U}}(1-y_5)$$

$$i = 1, \ldots, 4 \quad \text{(B2)}$$

$$D^{S_2}(t) \le D^{S_2}_{\mathfrak{U}} y_6, \quad t \in [t_1, t_2]$$

$$B^{S_2}(t) \le B^{S_2}_{\mathfrak{U}}(1-y_6), \quad t \in [t_1, t_2]$$

$$M^F_{\mathfrak{L}}(1-y_6) \le H^{S_2}_B(t_1) - F^{S_2} \le M^F_{\mathfrak{U}}(1-y_6) \quad \text{(B3)}$$

$$M^F_{\mathfrak{L}} y_6 \le H^{S_2}_C(t_1) - F^{S_2} \le M^F_{\mathfrak{U}} y_6$$

$$0.01 = H^{S_2}_B(t_1) + H^{S_2}_C(t_1) - F^{S_2}$$

$$x^{S_2}_{B,4}(t_2) \ge \chi_4 y_7$$
$$M^F_{\mathfrak{L}}(1-y_7) \le F^{S_3} - Dist^{S_2}(t_2) \le M^F_{\mathfrak{U}}(1-y_7)$$
$$M^x_{\mathfrak{L}}(1-y_7) \le x^{S_3}_{B,i}(t_2) - x^{S_2}_{C,i,acc.}(t_2) \le M^x_{\mathfrak{U}}(1-y_7)$$

$$x^{S_2}_{C,2,acc.}(t_2) \ge \chi_2 y_8$$
$$M^F_{\mathfrak{L}}(1-y_8) \le F^{S_3} - H^{S_2}_B(t_2) \le M^F_{\mathfrak{U}}(1-y_8)$$
$$M^x_{\mathfrak{L}}(1-y_8) \le x^{S_3}_{B,i}(t_2) - x^{S_2}_{B,i}(t_2) \le M^x_{\mathfrak{U}}(1-y_8)$$

$$x^{S_2}_{C,2}(t_2) \ge \chi_2 y_9$$
$$M^F_{\mathfrak{L}}(1-y_9) \le F^{S_3} - Bot^{S_2}(t_2) \le M^F_{\mathfrak{U}}(1-y_9)$$
$$M^x_{\mathfrak{L}}(1-y_9) \le x^{S_3}_{B,i}(t_2) - x^{S_2}_{B,i,acc.}(t_2) \le M^x_{\mathfrak{U}}(1-y_9)$$

$$x^{S_2}_{B,4acc.}(t_2) \ge \chi_4 y_{10}$$
$$M^F_{\mathfrak{L}}(1-y_{10}) \le F^{S_3} - H^{S_2}_C(t_2) \le M^F_{\mathfrak{U}}(1-y_{10})$$
$$M^x_{\mathfrak{L}}(1-y_{10}) \le x^{S_3}_{B,i}(t_2) - x^{S_2}_{c,i}(t_2) \le M^x_{\mathfrak{U}}(1-y_{10})$$

$$x^{S_2}_{B,3}(t_2) \ge \chi_3 y_{11}$$
$$M^F_{\mathfrak{L}}(1-y_{11}) \le F^{S_3} - Dist^{S_2}(t_2) \le M^F_{\mathfrak{U}}(1-y_{11}) \quad \text{(B4)}$$
$$M^x_{\mathfrak{L}}(1-y_{11}) \le x^{S_3}_{B,i}(t_2) - x^{S_2}_{C,i,acc.}(t_2) \le M^x_{\mathfrak{U}}(1-y_{11})$$

$$x^{S_2}_{C,1,acc.}(t_2) \ge \chi_1 y_{12}$$
$$M^F_{\mathfrak{L}}(1-y_{12}) \le F^{S_3} - H^{S_2}_B(t_2) \le M^F_{\mathfrak{U}}(1-y_{12})$$
$$M^x_{\mathfrak{L}}(1-y_{12}) \le x^{S_3}_{B,i}(t_2) - x^{S_2}_{B,i}(t_2) \le M^x_{\mathfrak{U}}(1-y_{12})$$

$$x^{S_2}_{C,1}(t_2) \ge \chi_1, y_{13}$$
$$M^F_{\mathfrak{L}}(1-y_{13}) \le F^{S_3} - Bot^{S_2}(t_2) \le M^F_{\mathfrak{U}}(1-y_{13})$$
$$M^x_{\mathfrak{L}}(1-y_{13}) \le x^{S_3}_{B,i}(t_2) - x^{S_2}_{B,i,acc.}(t_2) \le M^x_{\mathfrak{U}}(1-y_{13})$$

$$x^{S_2}_{B,3,acc.}(t_2) \ge \chi_3 y_{14}$$
$$M^F_{\mathfrak{L}}(1-y_{14}) \le F^{S_3} - H^{S_2}_C(t_2) \le M^F_{\mathfrak{U}}(1-y_{14})$$
$$M^x_{\mathfrak{L}}(1-y_{14}) \le x^{S_3}_{B,i}(t_2) - x^{S_2}_{C,i}(t_2) \le M^x_{\mathfrak{U}}(1-y_{14})$$

$$i = 1, \ldots, 4$$

$$D^{S_3}(t) \le D^{S_3}_{\mathfrak{U}} y_{15}, \quad t \in [t_2, t_3]$$

$$B^{S_3}(t) \le B^{S_3}_{\mathfrak{U}}(1-y_{15}), \quad t \in [t_2, t_3]$$

$$M^F_{\mathfrak{L}}(1-y_{15}) \le H^{S_3}_B(t_2) - F^{S_3} \le M^F_{\mathfrak{U}}(1-y_{15}) \quad \text{(B5)}$$

$$M^F_{\mathfrak{L}} y_{15} \le H^{S_3}_C(t_2) - F^{S_3} \le M^F_{\mathfrak{U}} y_{15}$$

$$0.01 = H^{S_3}_B(t_2) + H^{S_3}_C(t_2) - F^{S_3}$$

$$x^{S_3}_{C,2,acc.}(t_3) \ge \chi_2 y_{16}$$

$$x^{S_3}_{B,3}(t_3) \ge \chi_3 y_{16}$$

$$x^{S_3}_{C,2}(t_3) \ge \chi_2 y_{17}$$

$$
\begin{aligned}
x^{S_3}_{B,3,acc.}(t_3) &\geq \chi_3 y_{17} \\
x^{S_3}_{C,3,acc.}(t_3) &\geq \chi_3 y_{18} \\
x^{S_3}_{B,4}(t_3) &\geq \chi_4 y_{18} \\
x^{S_3}_{C,3}(t_3) &\geq \chi_3 y_{19} \\
x^{S_3}_{B,4,acc.}(t_3) &\geq \chi_4 y_{19} \\
x^{S_3}_{C,1,acc.}(t_3) &\geq \chi_1 y_{20} \\
x^{S_3}_{B,2}(t_3) &\geq \chi_2 y_{20} \\
x^{S_3}_{C,1}(t_3) &\geq \chi_1 y_{21} \\
x^{S_3}_{B,2,acc.}(t_3) &\geq \chi_2 y_{21}
\end{aligned}
\qquad \text{(B6)}
$$

The propositional logic expressions given in Eqs. A13–A17 are transformed into linear integer constraints according to

$$
\begin{aligned}
-y_1 + y_2 + y_3 &\geq 0 \\
y_1 + y_4 + y_5 &\geq 1 \\
y_1 + y_2 + y_3 + y_4 &= 1
\end{aligned}
\qquad \text{(B7)}
$$

$$
\begin{aligned}
-y_6 + y_7 + y_8 + y_{11} + y_{12} &\geq 0 \\
y_6 + y_9 + y_{10} + y_{13} + y_{14} &\geq 1 \\
y_7 + y_8 + y_9 + y_{10} + y_{11} + y_{12} + y_{13} + y_{14} &= 1 \\
-y_{15} + y_{16} + y_{18} + y_{20} &\geq 0 \\
y_{15} + y_{17} + y_{19} + y_{21} &\geq 1 \\
y_{16} + y_{17} + y_{18} + y_{19} + y_{20} + y_{21} &= 1
\end{aligned}
\qquad \text{(B8)}
$$

$$
\begin{array}{lll}
y_2 + y_{11} \leq 1, & y_2 + y_{12} \leq 1, & y_2 + y_{13} \leq 1 \\
y_2 + y_{14} \leq 1, & y_2 + y_{18} \leq 1, & y_2 + y_{19} \leq 1 \\
y_3 + y_7 \leq 1, & y_3 + y_8 \leq 1, & y_3 + y_9 \leq 1 \\
y_3 + y_{10} \leq 1, & y_3 + y_{20} \leq 1, & y_3 + y_{21} \leq 1 \\
y_4 + y_7 \leq 1, & y_4 + y_8 \leq 1, & y_4 + y_9 \leq 1 \\
y_4 + y_{10} \leq 1, & y_4 + y_{20} \leq 1, & y_4 + y_{21} \leq 1 \\
y_5 + y_{11} \leq 1, & y_5 + y_{12} \leq 1, & y_5 + y_{13} \leq 1 \\
y_4 + y_{14} \leq 1, & y_5 + y_{18} \leq 1, & y_5 + y_{19} \leq 1
\end{array}
\qquad \text{(B9)}
$$

$$
\begin{array}{llll}
 & & y_7 + y_{18} \leq 1, & y_7 + y_{19} \leq 1 \\
y_8 + y_{16} \leq 1, & y_8 + y_{17} \leq 1, & y_8 + y_{20} \leq 1, & y_8 + y_{21} \leq 1 \\
y_9 + y_{16} \leq 1, & y_9 + y_{17} \leq 1, & y_9 + y_{20} \leq 1, & y_9 + y_{21} \leq 1 \\
 & & y_{10} + y_{18} \leq 1, & y_{10} + y_{19} \leq 1 \\
y_{11} + y_{16} \leq 1, & y_{11} + y_{17} \leq 1, & y_{11} + y_{18} \leq 1, & y_{11} + y_{19} \leq 1 \\
 & & y_{12} + y_{20} \leq 1, & y_{12} + y_{21} \leq 1 \\
 & & y_{13} + y_{20} \leq 1, & y_{13} + y_{21} \leq 1 \\
y_{14} + y_{16} \leq 1, & y_{14} + y_{17} \leq 1, & y_{14} + y_{18} \leq 1, & y_{14} + y_{19} \leq 1
\end{array}
\qquad \text{(B10)}
$$

The big-M constants are specified according to

$$
\begin{aligned}
& B^{S_i}_{\mathfrak{u}} = 12\,\frac{\text{kmol}}{\text{h}}, \quad D^{S_i}_{\mathfrak{u}} = 12\,\frac{\text{kmol}}{\text{h}}, \quad i = 1, 2, 3 \\
& M^{x}_{\mathfrak{L}} = -1, \quad M^{x}_{\mathfrak{u}} = 1, \quad M^{F}_{\mathfrak{L}} = -11\ \text{kmol} \\
& M^{F}_{\mathfrak{u}} = 11\ \text{kmol}
\end{aligned}
$$

Manuscript received Nov. 11, 2002, revision received Apr. 2, 2003, and final revision received June 30, 2003.