Global Mixed-Integer Dynamic Optimization

Benoît Chachuat, Adam B. Singer, and Paul I. Barton
Process Systems Engineering Laboratory, Dept. of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139

DOI 10.1002/aic.10494
Published online June 3, 2005 in Wiley InterScience (www.interscience.wiley.com).

Recent advances in process synthesis, design, operations, and control have created an increasing demand for efficient numerical algorithms for optimizing a dynamic system coupled with discrete decisions; these problems are termed mixed-integer dynamic optimization (MIDO). In this communication, we develop a decomposition approach for a quite general class of MIDO problems that is capable of guaranteeing finding a global solution despite the nonconvexities inherent in the dynamic optimization subproblems. Two distinct algorithms are considered. On finite termination, the first algorithm guarantees finding a global solution of the MIDO within nonzero tolerance; the second algorithm finds rigorous bounds bracketing the global solution value, with a substantial reduction in computational expense relative to the first algorithm. A case study is presented in connection with the optimal design and operation of a batch process consisting of a series reaction followed by a separation with no intermediate storage. The developed algorithms demonstrate efficiency and applicability in solving this problem. Several heuristics are tested to enhance convergence of the algorithms; in particular, the use of bounds tightening techniques and the addition of cuts resulting from a screening model of the batch process are considered. © 2005 American Institute of Chemical Engineers
AIChE J, 51: 2235–2253, 2005
Keywords: mixed-integer dynamic optimization, nonconvex problems, global optimization, decomposition algorithms, outer-approximation, batch process design

1 Introduction

Many problems in chemical engineering can be formulated as mixed-integer programs (MIPs), that is, optimization problems that involve discrete-valued as well as continuous-valued variables. The discrete variables are typically used to model the selection of unit operations in a flowsheet, temporal sequencing decisions (binary variables), or numbers of equipment items, batches, functional groups, and so forth (integer variables). The continuous variables correspond to design decisions and operating conditions pertaining to a certain system structure (such as unit sizes, pressures, temperatures, etc.). Over the last two decades, interest in dynamic simulation and optimization of chemical processes has increased significantly, with particular emphasis placed on examining the performance of a process system operating under transient conditions. Chemical processes are typically modeled using ordinary differential equations (ODEs) or differential/algebraic equations (DAEs); these equations describe mass, energy, and momentum balances and thus ensure physical and thermodynamic consistency. Recent advances in process synthesis, design, operations, and control have created an increasing demand for efficient numerical algorithms capable of optimizing a dynamic system coupled with discrete decisions; such problems are termed *mixed-integer dynamic optimization* (MIDO). Areas of application for MIDO include batch process synthesis and development,[1-5] design of batch distillation columns,[6-8] solvent design in batch processes,[9] simultaneous design and control,[10-13] reduction of kinetic mechanisms,[14,15] and optimization of hybrid discrete/continuous systems.[16-18] In contrast to dynamic optimization problems (including global dynamic optimization methods[19,20]), for which direct solution methods are capable of

Correspondence concerning this article should be addressed to P. I. Barton at pib@mit.edu.

© 2005 American Institute of Chemical Engineers

solving a broad class of problems, only limited progress has been achieved in addressing MIDO problems. In particular, no general procedure has yet been proposed that guarantees convergence to a global solution of a MIDO problem.

The solution approaches for dynamic optimization problems without discrete variables participating can be divided into simultaneous and sequential methods. The former method, also termed total discretization, converts the problem into a finite dimensional nonlinear program (NLP) by discretization of the states and controls using, for example, orthogonal collocation on finite elements.[21] Within the sequential, or control vector parameterization, approach[22] only control variables are discretized; the state variables, which appear indirectly in the differential constraints, are determined by numerically solving an initial value problem. By using either of these approaches, a MIDO problem can be converted into a mixed-integer nonlinear program (MINLP). All the approaches for MIDO that have been proposed thus far in the literature are based on these principles. Avraam et al.[16] and Mohideen et al.[12] applied the total discretization approach and solved the resulting MINLP based on decomposition strategies, that is, the outer-approximation[23,24] (OA) and generalized Benders decomposition[25] (GBD) algorithms. However, this approach is marked by two major drawbacks. First, the number of variables in the MINLP explodes. This increase in problem size severely restricts the number of parameters and state variables for which the original problem may be practically solved. Second, the nonconvexities inherent in the total discretization method reduce the algorithm to an ad hoc improvement strategy when used in conjunction with a convex MINLP solver. On the other hand, Schweiger and Floudas,[13] Sharif et al.,[8] and Bansal et al.[10] decompose the MIDO algorithm into a sequence of primal problems (nonconvex dynamic optimizations), where the discrete variables are fixed, and relaxed master problems, which generate a new realization for the discrete variables. The computational effort is substantially reduced when using numerical integration to provide function evaluations and gradients for the dynamic optimization problems. However, neither the use of dual information nor the use of linearizations of the primal problem functions can guarantee providing valid support functions because of nonconvexity of the participating functions; therefore, such procedures are prone to excluding (possibly large) portions of the feasible region within which a global solution may occur. Allgor and Barton[2,26] acknowledged that the construction of "support" functions from primal or dual subproblems may yield potentially invalid lower bounding representations arising from the nonconvexities inherent in the dynamic optimization subproblems. Instead, screening models[3] derived from domain specific knowledge gathered from physical laws and engineering insight are used to construct a valid master problem. Furthermore, a decomposition approach is proposed by these authors that terminates with rigorous upper and lower bounds on the global solution value to the MIDO problem and a solution that is potentially suboptimal. This procedure has been applied to batch process development.[27,28]

In recent years, deterministic global optimization algorithms for MINLPs, where the participating functions are nonconvex, have begun to emerge. These methods rely either on a branch-and-bound procedure (branch-and-reduce,[29,30] SMIN-αBB and GMIN-αBB,[31] reformulation/spatial branch-and-bound[32]) or on a decomposition strategy (OA for nonconvex separable MINLPs[33] and nonconvex nonseparable MINLPs[34]). Furthermore, branch-and-bound and decomposition-based algorithms have been shown to be particular instances of a generalized branch-and-cut (GBC) framework used with different sets of heuristics.[35] By combining these global MINLP algorithms with the new convexity theory developed by Singer and Barton[20] that enables the construction of convex relaxations for general, nonconvex, Bolza-type functionals subject to an embedded linear time-varying (LTV) dynamic system, Lee and Barton[36] and Barton and Lee[17] recently solved MIDO problems with embedded LTV dynamic systems to global optimality. Applications of this procedure are given for linear hybrid discrete/continuous systems where the sequence of modes is optimized. The limitations, of course, are that such algorithms apply only to problems with linear dynamic systems embedded.

In this article, building on recent developments in both deterministic global optimization for MINLPs and relaxation techniques for optimization problems with embedded ODEs,[37,38] we develop a decomposition approach for a quite general class of MIDO problems that is capable of finding a global solution despite the nonconvexities inherent in the dynamic optimization subproblems, while still potentially avoiding total enumeration of the discrete alternatives. More specifically, we extend the OA algorithm proposed by Kesavan et al.[33] to mixed-integer problems with nonlinear ODEs embedded. The convexity theory and relaxation techniques developed in Singer and Barton[37,38] are used to construct valid convex relaxations for the functions with state variables participating. One reason for selecting the OA algorithm is the feasibility of developing an implementation using parallel computations; such an approach would significantly reduce the computational time and allow the solution of larger problems,[39] especially when solution of the primal problem dominates the running time, which is the case here. Ostensibly, any other branch-and-bound or decomposition-based global MINLP algorithm could be used to solve the problem, provided that rigorous lower bounding problems could be constructed from existing relaxation techniques. To the authors' knowledge, our contribution represents the first practical and truly rigorous deterministic global optimization algorithm in the area of mixed-integer dynamic optimization.

The remainder of the paper is organized in the following manner. In section 2, we formulate the MIDO problem and derive the subproblems used in the OA algorithm; special emphasis is placed on the construction of valid relaxations for functions with state variables participating. The proposed algorithm for solution of MIDO problems to guaranteed global optimality is presented in section 3, and several aspects related to its numerical implementation are discussed. The final section of the paper presents a relatively simple batch process development example that demonstrates the applicability and effectiveness of the proposed approach.

2 Theoretical Background

Let $P = [\mathbf{p}^L, \mathbf{p}^U] \subset \mathbb{R}^{n_p}$, $Y = \{0, 1\}^{n_y}$, and $X \subseteq \mathbb{R}^{n_x}$ be such that $\mathbf{x}(\mathbf{p}, \mathbf{y}, t) \in X$, $\forall (\mathbf{p}, \mathbf{y}, t) \in P \times Y \times [t_0, t_f]$. We consider the class of MIDO problems that conform to the following formulation:

$$
\begin{aligned}
\min_{\mathbf{p},\mathbf{y}} \mathcal{J} &= \phi_0[\mathbf{x}(\mathbf{p},\mathbf{y},t_f),\mathbf{p},\mathbf{y}] + \int_{t_0}^{t_f} \psi_0[\mathbf{x}(\mathbf{p},\mathbf{y},t),\mathbf{p},\mathbf{y},t]dt \\
\text{s.t.} \quad 0 &\geq \phi_k[\mathbf{x}(\mathbf{p},\mathbf{y},t_f),\mathbf{p},\mathbf{y}] + \int_{t_0}^{t_f} \psi_k[\mathbf{x}(\mathbf{p},\mathbf{y},t),\mathbf{p},\mathbf{y},t]dt \qquad k = 1,\ldots,n_c \\
\dot{\mathbf{x}}(\mathbf{p},\mathbf{y},t) &= \mathbf{f}[\mathbf{x}(\mathbf{p},\mathbf{y},t),\mathbf{p},\mathbf{y},t] \qquad \forall\, t \in [t_0,t_f] \\
\mathbf{x}(\mathbf{p},\mathbf{y},t_0) &= \mathbf{h}(\mathbf{p},\mathbf{y}) \\
\mathbf{p} &\in P \\
\mathbf{y} &\in Y
\end{aligned} \tag{P}
$$

where $\mathbf{p}$ denotes the continuous time-invariant parameters; $\mathbf{y}$ is a special set of time invariant parameters that can take only 0–1 values; t_0 and t_f denote the initial and final time, respectively; $\mathbf{x}$ represents the continuous variables describing the state of the process; $\phi_k : X \times P \times Y \mapsto \mathbb{R}$ and $\psi_k : X \times P \times Y \times [t_0, t_f] \mapsto \mathbb{R}$, $k = 0, \ldots, n_c$ are continuous, potentially nonconvex mappings; $f_i : X \times P \times Y^c \times [t_0, t_f] \mapsto \mathbb{R}$, $i = 1, \ldots, n_x$, and $\mathrm{h}_i : P \times Y^c \mapsto \mathbb{R}$, $i = 1, \ldots, n_x$ are continuous mappings; $Y^c = \mathrm{conv}(Y) = [0, 1]^{n_y}$ denotes the convex hull of Y.

Remark 2.1. The objective and constraint functions in (P) can include a finite number of point and integral terms internal to the time domain, but these are omitted here for the sake of clarity.

Remark 2.2. Problems containing function-valued decision variables, $\mathbf{u}(t)$, can be converted into the formulation given in (P) by applying standard control parameterization techniques, which consist of approximating the infinite dimensional control variables, $\mathbf{u}(t)$, with parametric functions, $\mathcal{U}(\mathbf{p}, t)$, such as piecewise constant or piecewise linear functions. A piecewise constant approximation could also be considered for problems containing binary functions, $\mathbf{y}(t)$, that is, functions whose time profile is restricted to take 0–1 values.

Remark 2.3. The final time t_f in problem (P) is fixed. However, MIDO problems with a varying end time can be addressed by transformation, such as by normalizing the time t as $\tau = (t - t_0)/\Delta t$, where $\Delta t = t_f - t_0$. Accordingly, the objective and constraint functions in (P), as well as the differential system, would be reformulated as

$$
\mathcal{J} = \phi_0[\mathbf{x}(\mathbf{p},\mathbf{y},1),\mathbf{p},\mathbf{y}] + \int_0^1 \Delta t \times \psi_0[\mathbf{x}(\mathbf{p},\mathbf{y},\tau),\mathbf{p},\mathbf{y},\tau\Delta t + t_0]d\tau
$$

$$
0 \geq \phi_k[\mathbf{x}(\mathbf{p},\mathbf{y},1),\mathbf{p},\mathbf{y}] + \int_0^1 \Delta t \times \psi_k[\mathbf{x}(\mathbf{p},\mathbf{y},\tau),\mathbf{p},\mathbf{y},\tau\Delta t + t_0]d\tau \qquad k = 1,\ldots,n_c
$$

$$
\dot{\mathbf{x}}(\mathbf{p},\mathbf{y},\tau) = \Delta t \times \mathbf{f}[\mathbf{x}(\mathbf{p},\mathbf{y},\tau),\mathbf{p},\mathbf{y},\tau\Delta t + t_0] \qquad \forall\, \tau \in [0,1]
$$

$$
\mathbf{x}(\mathbf{p},\mathbf{y},0) = \mathbf{h}(\mathbf{p},\mathbf{y})
$$

and Δt could then be considered as a decision variable in the optimization problem. Also, piecewise control functions have to be on a fixed partition of the time interval, but this can be addressed by applying similar time transformations.

Remark 2.4. In contrast to other recent, quite general MIDO formulations where the dynamic systems are described by a set of DAEs,[2,10] problem (P) considers only embedded ODE systems. This is because no technique has been proposed to date for constructing valid relaxations of a problem with DAEs embedded.

The methodology adopted to solve MIDO problems as specified in (P) to guarantee global optimality consists of extending the OA algorithms originally developed by Kesavan et al.[33] for nonconvex MINLPs. Two different algorithms are considered. On finite termination, the first algorithm finds a global solution of (P), whereas the second algorithm finds rigorous bounds bracketing the global solution value of (P) and a solution that is potentially suboptimal. Each of these algorithms is detailed in section 3. They are both based on construction of the following subproblems:

- *Primal problem*: a nonconvex dynamic optimization problem obtained by fixing the binary variables $\mathbf{y}$ in (P), any feasible solution of which yields a rigorous upper bound to the solution value of the MIDO problem (P).
- *Lower Bounding Convex MIDO problem*: a convex MIDO problem, the solution of which yields a valid lower bound to the global solution value of problem (P).

• *Relaxed Master problem*: a MILP, the solution of which represents a valid lower bound on that subset of Y not yet explored by the algorithm.

• *Primal Bounding problem*: a convex dynamic optimization problem, the solution of which provides a valid and tighter lower bound to the Primal problem for each fixed binary realization $\mathbf{y}$ than that provided by the Relaxed Master problem that generates $\mathbf{y}$.

A prerequisite for constructing any of these subproblems, excluding the Primal problem, is a convexity theory for dynamic optimization and the ability to build valid convex relaxations for the functions in problem (P). Techniques for deriving such relaxations are discussed in subsection 2.1. The aforementioned subproblems are then formulated and described thoroughly in subsections 2.2–2.5. We conclude this section by considering a simple mathematical example that illustrates construction of the relaxations and subproblems in subsection 2.6.

2.1 Constructing convex relaxations of the problem functions

To derive valid convex relaxations of the original MIDO problem, one has to convexify and relax the nonconvex functions defined in both the continuous and discrete variables.

In the last three decades, significant developments have been achieved in deriving convex relaxations for functions of special structure without state variables participating. For example, the convex envelope for bilinear and univariate concave functions has been derived[40]; the convex and concave envelopes for fractional, multilinear, and similar expressions have also been derived[41,42]; convex underestimators for separable[43] and factorable[40] functions can be constructed; and α-based convex underestimators can be obtained for twice continuously differentiable nonconvex functions.[44]

Recently, a convexity theory has been developed that enables the convex relaxations of elementary functions on Euclidian spaces (described above) to be harnessed in the construction of convex relaxations of general, nonconvex Bolza-type functionals subject to an embedded (LTV) dynamic system.[20] The fundamental theorem for generating convex underestimators for parameter-embedded integrals states that partial convexity of the integrand implies convexity of the integral. Furthermore, several constructive methods have been proposed in recent years to construct convex relaxations of functions with state variables participating. Papamichail and Adjiman[19] substitute a new real-valued decision variable and introduce a new equality constraint for each state at a fixed time. This additional constraint can then be relaxed by deriving α-based convex lower and concave upper bounding inequalities, where the α-values are determined rigorously from a combination of interval arithmetic techniques and the simultaneous solution of second-order sensitivity equations. Chachuat and Latifi[45] propose to convexify every term in the objective and constraint functionals with state variables participating by adding a suitable quadratic term. Two approaches, one based on the sensitivity system and one based on the adjoint equations, are described to compute these quadratic terms. Singer and Barton[37] develop a technique for constructing convex and concave relaxations for the solution of a system of non-quasi-monotone ODEs based on an OA method. Subsequently, they derive convex relaxations for both point and integral terms by using the former ODE relaxations in conjunction with McCormick's composition result and factorable representation.[40]

Note that any of these approaches can be readily adapted to relax the solutions of functions with state and binary variables participating, such as by relaxing the discrete variables as continuous variables $\mathbf{y} \in Y^c$. In the remainder of this paper, we focus exclusively on the relaxation technique proposed by Singer and Barton.[37] As compared to the other available relaxation methods for problems with embedded ODEs, this technique does not need to evaluate second-order derivatives and presents the interesting property that the relaxations for the solution of the differential system are affine in the parameter space at any fixed time. Accordingly, a substantial amount of computational time can be saved by exploiting this special structure in the MIDO algorithms; this aspect of our implementation is discussed in more detail in subsection 3.3.

Constructing convex relaxations for terms with state variables participating proceeds in three main steps:

(1) *State bounds*: Compute time varying enclosures $\mathcal{X}(t) = [\mathbf{x}^L(t), \mathbf{x}^U(t)]$, $t \in [t_0, t_f]$ for the solution of the embedded dynamic system on $P \times Y^c$ by applying any suitable state bounding technique. When a set $\bar{\mathcal{X}}(t)$ of natural bounds is known for the dynamic system, valid enclosures are obtained by solving any set of ODEs that satisfy the following inequalities in $\mathbf{x}^L$ and $\mathbf{x}^U$ (see Corollary 2.4 by Singer and Barton[37] for proof and discussion):

$$\left.\begin{aligned} \dot{x}_i^L(t) &\le \inf_{\tilde{\mathbf{x}},\tilde{\mathbf{p}},\tilde{\mathbf{y}}}\{f_i(\tilde{\mathbf{x}},\tilde{\mathbf{p}},\tilde{\mathbf{y}},t) : \tilde{\mathbf{x}} \in \mathcal{X}(t) \cap \bar{\mathcal{X}}(t), \tilde{x}_i = x_i^L(t), \tilde{\mathbf{p}} \in P, \tilde{\mathbf{y}} \in Y^c\} \\ \dot{x}_i^U(t) &\ge \sup_{\tilde{\mathbf{x}},\tilde{\mathbf{p}},\tilde{\mathbf{y}}}\{f_i(\tilde{\mathbf{x}},\tilde{\mathbf{p}},\tilde{\mathbf{y}},t) : \tilde{\mathbf{x}} \in \mathcal{X}(t) \cap \bar{\mathcal{X}}(t), \tilde{x}_i = x_i^U(t), \tilde{\mathbf{p}} \in P, \tilde{\mathbf{y}} \in Y^c\} \end{aligned}\right\} \quad \begin{aligned} &\forall\, t \in (t_0, t_f] \\ &i = 1, \ldots, n_x \end{aligned}$$

$$\left.\begin{aligned} x_i^L(t_0) &\le \inf_{\tilde{\mathbf{p}},\tilde{\mathbf{y}}}\{h_i(\tilde{\mathbf{p}},\tilde{\mathbf{y}}) : \tilde{\mathbf{p}} \in P, \tilde{\mathbf{y}} \in Y^c\} \\ x_i^U(t_0) &\ge \sup_{\tilde{\mathbf{p}},\tilde{\mathbf{y}}}\{h_i(\tilde{\mathbf{p}},\tilde{\mathbf{y}}) : \tilde{\mathbf{p}} \in P, \tilde{\mathbf{y}} \in Y^c\} \end{aligned}\right\} \quad i = 1, \ldots, n_x$$

(2) *State relaxations*: Construct convex underestimators and concave overestimators for the solution of the embedded differential system. Given time-varying enclosures $\mathcal{X}(t)$, $t \in [t_0, t_f]$, a convex underestimator $\mathbf{c}(\mathbf{p}, \mathbf{y}, \underline{t})$, and a concave overestimator $\mathbf{C}(\mathbf{p}, \mathbf{y}, \underline{t})$ for $\mathbf{x}(\mathbf{p}, \mathbf{y}, \underline{t})$ are obtained on $P \times Y^c$, at each fixed $\underline{t}$ in $[t_0, t_f]$, by solving the following set of ODEs (see Theorem 3.2 by Singer and Barton[37] for proof and discussion):

$$\left.\begin{aligned}\dot{c}_i(\mathbf{p},\mathbf{y},t) &= \inf_{\tilde{\mathbf{x}}}\{\mathscr{L}_{u_i}(\tilde{\mathbf{x}},\mathbf{p},\mathbf{y},t)|_{\mathbf{x}^*(t),\mathbf{p}^*,\mathbf{y}^*} : \tilde{\mathbf{x}}\in\mathscr{C}(\mathbf{p},\mathbf{y},t),\tilde{x}_i = c_i(\mathbf{p},\mathbf{y},t)\}\\ \dot{C}_i(\mathbf{p},\mathbf{y},t) &= \sup_{\tilde{\mathbf{x}}}\{\mathscr{L}_{o_i}(\tilde{\mathbf{x}},\mathbf{p},\mathbf{y},t)|_{\mathbf{x}^*(t),\mathbf{p}^*,\mathbf{y}^*} : \tilde{\mathbf{x}}\in\mathscr{C}(\mathbf{p},\mathbf{y},t),\tilde{x}_i = C_i(\mathbf{p},\mathbf{y},t)\}\end{aligned}\right\}\quad \begin{aligned}&\forall\, t\in(t_0,t_f]\\ &i=1,\ldots,n_x\end{aligned}$$

$$\mathbf{c}(t_0)\le\mathbf{h}(\mathbf{p},\mathbf{y})\le\mathbf{C}(t_0)\qquad\forall\,(\mathbf{p},\mathbf{y})\in P\times Y^c$$

for some reference trajectory $[\mathbf{x}^*(t),\,\mathbf{p}^*,\,\mathbf{y}^*]\in\mathscr{X}(t)\times P\times Y^c$; where $u_i(\mathbf{x},\mathbf{p},\mathbf{y},\underline{t})$ and $o_i(\mathbf{x},\mathbf{p},\mathbf{y},\underline{t})$ are a convex underestimator and a concave overestimator of f_i on $\mathscr{X}(\underline{t})\times P\times Y^c$ for each fixed $\underline{t}\in[t_0,t_f]$, respectively; $\mathscr{L}_f(\mathbf{x},\mathbf{p},\mathbf{y},\underline{t})\,|_{\mathbf{x}^*(\underline{t}),\mathbf{p}^*,\mathbf{y}^*}$ denotes the linearization of $f(\mathbf{x},\mathbf{p},\mathbf{y},\underline{t})$ at point $[\mathbf{x}^*(\underline{t}),\,\mathbf{p}^*,\,\mathbf{y}^*]$; and $\mathscr{C}(\mathbf{p},\mathbf{y},t)=\{\tilde{\mathbf{x}}\,|\,\mathbf{c}(\mathbf{p},\mathbf{y},t)\le\tilde{\mathbf{x}}\le\mathbf{C}(\mathbf{p},\mathbf{y},t)\}$. Note that the aforementioned differential system is a LTV ODE with a fixed sequence of events; the solution functions $\mathbf{c}(\cdot,\cdot,\underline{t})$ and $\mathbf{C}(\cdot,\cdot,\underline{t})$ are therefore affine in $\mathbf{p}$ and $\mathbf{y}$ at fixed $\underline{t}$.[46] Also note that, in practice, the differential equations for $\mathbf{c}$ and $\mathbf{C}$ are integrated simultaneously with the equations yielding $\mathbf{x}^L$ and $\mathbf{x}^U$.

(3) *Function relaxations*: Derive convex underestimators for the terms with state variables participating. This can be accomplished by applying McCormick's technique for relaxing factorable functions. Consider, for example, the function $\varphi[x_i(\mathbf{p},\mathbf{y},\underline{t})]$ ($i\in\{1,\ldots,n_x\}$, $\underline{t}\in[t_0,t_f]$ fixed), let e_φ be a convex underestimator of φ on $\mathscr{X}_i(\underline{t})$ (obtained by using any suitable technique for relaxing functions without state variables participating as discussed earlier), and denote z_{min} a point at which $e_\varphi(\cdot)$ attains its infimum on $\mathscr{X}_i(\underline{t})$. Then,

$$u_\varphi(\mathbf{p},\mathbf{y}) = e_\varphi[\mathrm{mid}\{c_i(\mathbf{p},\mathbf{y},\underline{t}),\,C_i(\mathbf{p},\mathbf{y},\underline{t}),\,z_{min}\}]$$

is a convex underestimator for $\varphi[x_i(\mathbf{p},\mathbf{y},\underline{t})]$ on $P\times Y^c$ (see McCormick[47] for proof). An analogous result holds for constructing concave overestimators. In these relaxations, the mid function selects the middle value of three scalars. Its use merits some comments because it may introduce nonsmoothness in the resulting convex underestimators. For example, this situation occurs when the bounds $\mathbf{x}^L$ or $\mathbf{x}^U$ for the state variables yield tighter relaxations than those provided by the functions $\mathbf{c}(\mathbf{p},\mathbf{y},\underline{t})$ or $\mathbf{C}(\mathbf{p},\mathbf{y},\underline{t})$, respectively, for some points $(\mathbf{p},\mathbf{y})\in P\times Y^c$. An alternative approach providing smooth relaxations consists of adding a new variable ω and new inequality constraints as

$$\begin{aligned}u_\varphi(\mathbf{p},\mathbf{y}) &= e_\varphi(\omega)\\ \text{s.t.}\quad \omega &\ge c_i(\mathbf{p},\mathbf{y},\underline{t})\\ \omega &\ge x_i^L(\underline{t})\\ \omega &\le C_i(\mathbf{p},\mathbf{y},\underline{t})\\ \omega &\le x_i^U(\underline{t})\end{aligned}$$

It is demonstrated in Tawarmalani and Sahinidis[30] (Theorem 2) that the underestimator in the mid function approach is implied in the relaxation derived through the application of the decomposition approach as long as the same function e_φ is used as the aforementioned underestimator for φ.

Concerning integral terms, valid convex/concave relaxations are derived by exploiting the monotonicity of the Lebesgue integral, as illustrated in Singer and Barton.[38] More precisely, integrating a pointwise in time convex underestimator (resp. concave overestimator) for an integrand provides a convex underestimator (resp. concave overestimator) for an integral.

2.2 Lower bounding convex MIDO problem

The development of rigorous decomposition algorithms to solve MIDO problems relies on the ability to construct valid support functions. As discussed in the introduction of this paper, such support functions cannot be obtained directly by linearizing the objective and constraint functionals in problem (P) because of the nonconvexity inherent in the dynamic optimization subproblems. Rather, a lower bounding convex MIDO problem [subsequently referred to as (LBP)] is considered.

$$\begin{aligned}\min_{\mathbf{p},\mathbf{y}}\ & u_{\mathscr{J}} = u_{\phi_0}(\mathbf{p},\mathbf{y}) + \int_{t_0}^{t_f} u_{\psi_0}(\mathbf{p},\mathbf{y},t)dt\\ \text{s.t.}\quad & 0\ge u_{\phi_k}(\mathbf{p},\mathbf{y}) + \int_{t_0}^{t_f} u_{\psi_k}(\mathbf{p},\mathbf{y},t)dt \qquad k=1,\ldots,n_c\\ & \mathbf{p}\in P\\ & \mathbf{y}\in Y\end{aligned}\tag{LBP}$$

where u_{ϕ_k} and u_{ψ_k}, $k=0,\ldots,n_c$, denote convex underestimators of φ_k and ψ_k, respectively, for $(\mathbf{p},\mathbf{y})\in P\times Y^c$ and each fixed $\underline{t}\in[t_0,t_f]$, as obtained by applying the relaxation procedure described in subsection 2.1, or any alternative valid relaxation procedure. Note that problem (LBP) contains the feasible set of the original MIDO problem (P) for each integer realization $\mathbf{y}\in Y$. Furthermore, note that the objective function $u_{\mathscr{J}}$ of (LBP) underestimates the objective function $\mathscr{J}$ of (P) for each $\mathbf{y}\in Y$. By construction, a solution of (LBP) therefore provides a valid lower bound to the global solution value of (P).

The following assumptions on (LBP) are necessary:

Assumption 1 The relaxed functions u_{ϕ_k} and u_{ψ_k}, $k=0,\ldots,n_c$, are once continuously differentiable in an open neighborhood of $(\mathbf{p}^j,\,\mathbf{y}^j)$, where $\mathbf{p}^j$ is a KKT (Karush–Kuhn–Tucker) point for the convex subproblem obtained by fixing the binary variables to $\mathbf{y}^j$ in (LBP).

Assumption 2 A constraint qualification holds at the solution of every subproblem obtained by fixing the binary variables $\mathbf{y}$ in (LBP).

These two assumptions guarantee that the KKT conditions are both necessary and sufficient to identify the global minimum of the Primal Bounding problems as formulated in subsection 2.4 below. They were first considered by Fletcher and Leyffer[24] in their seminal work on OA algorithms for convex MINLPs. We refer the interested reader to that paper for a discussion on these assumptions.

2.3 Primal problem

The Primal problem [P($\mathbf{y}^j$)] is a nonconvex dynamic optimization problem obtained by fixing the binary variables $\mathbf{y} = \mathbf{y}^j$ in (P) to yield

$$\begin{aligned}
\min_{\mathbf{p}} \ \mathcal{J} &= \phi_0[\mathbf{x}(\mathbf{p}, \mathbf{y}^j, t_f), \mathbf{p}, \mathbf{y}^j] + \int_{t_0}^{t_f} \psi_0[\mathbf{x}(\mathbf{p}, \mathbf{y}^j, t), \mathbf{p}, \mathbf{y}^j, t]dt \\
\text{s.t.} \quad 0 &\geq \phi_k[\mathbf{x}(\mathbf{p}, \mathbf{y}^j, t_f), \mathbf{p}, \mathbf{y}^j] + \int_{t_0}^{t_f} \psi_k[\mathbf{x}(\mathbf{p}, \mathbf{y}^j, t), \mathbf{p}, \mathbf{y}^j, t]dt \qquad k = 1, \ldots, n_c \\
\dot{\mathbf{x}}(\mathbf{p}, \mathbf{y}^j, t) &= \mathbf{f}[\mathbf{x}(\mathbf{p}, \mathbf{y}^j, t), \mathbf{p}, \mathbf{y}^j, t] \qquad \forall\, t \in [t_0, t_f] \\
\mathbf{x}(\mathbf{p}, \mathbf{y}^j, t_0) &= \mathbf{h}(\mathbf{p}, \mathbf{y}^j) \\
\mathbf{p} &\in P \qquad [\mathrm{P}(\mathbf{y}^j)]
\end{aligned}$$

Obviously, any feasible point for the primal problem provides a valid upper bound (UBD) on the global solution value of (P).

The two variants of the MIDO algorithm described later differ as to whether the primal problem is solved to global optimality or local optimality (see section 3). In the former case, the algorithm assumes that [P($\mathbf{y}^j$)] can be solved to ε-optimality in a finite number of steps using deterministic global optimization methods. Several such methods have been proposed in recent years for optimization problems with embedded ODEs. For example, this can be achieved by using gradient-based methods in a branch-and-bound framework, where the requisite convex relaxations are constructed by applying one of the techniques discussed previously in subsection 2.1. On the other hand, standard optimization methods, such as the sequential approach[22] or the simultaneous approach,[21] can be applied in the second algorithm to find a local solution of the primal problem.

2.4 Primal Bounding problem

Fixing the binary variables in the Lower Bounding Convex MIDO problem (LBP) yields the following dynamic optimization problem

$$\begin{aligned}
\min_{\mathbf{p}\in P} \ u_{\mathcal{J}}(\mathbf{y}^j) &= u_{\phi_0}(\mathbf{p}, \mathbf{y}^j) + \int_{t_0}^{t_f} u_{\psi_0}(\mathbf{p}, \mathbf{y}^j, t)dt \\
\text{s.t.} \quad 0 &\geq u_{\phi_k}(\mathbf{p}, \mathbf{y}^j) + \int_{t_0}^{t_f} u_{\psi_k}(\mathbf{p}, \mathbf{y}^j, t)dt \\
& \qquad k = 1, \ldots, n_c \quad [\mathrm{PB}(\mathbf{y}^j)]
\end{aligned}$$

By construction, [PB($\mathbf{y}^j$)] is a convex problem whose feasible set overestimates the feasible set of [P($\mathbf{y}^j$)] and whose objective function underestimates the objective function of [P($\mathbf{y}^j$)] for each fixed binary realization $\mathbf{y}^j$ in the set Y. Accordingly, the solution value of [PB($\mathbf{y}^j$)] is a valid lower bound on the solution value of [P($\mathbf{y}^j$)]. Furthermore, the solution value of [PB($\mathbf{y}^j$)] is also greater than or equal to the solution value of the Relaxed Master problem that generates $\mathbf{y}^j$ (see subsection 2.5). Therefore, [PB($\mathbf{y}^j$)] represents a Primal Bounding problem for (P).[33]

When the Primal Bounding problem is infeasible for a given binary assignment $\mathbf{y}^j$, a feasibility problem is solved. It consists of minimizing a given measure of the constraint violations, given in general by an ℓ^1 or ℓ^∞ sum of constraint violations. The reader is referred to the article by Fletcher and Leyffer[24] for a discussion of the feasibility problem formulation and properties. In the case of an ℓ^1 sum of constraint violations, the Primal Bounding Feasibility problem, denoted as [PBF($\mathbf{y}^j$)], can be formulated as

$$\begin{aligned}
\min_{\boldsymbol{\mu}\geq\mathbf{0}, \mathbf{p}\in P} \ & \sum_{k=1}^{n_c} \mu_k \\
\text{s.t.} \quad \mu_k &\geq u_{\phi_k}(\mathbf{p}, \mathbf{y}^j) + \int_{t_0}^{t_f} u_{\psi_k}(\mathbf{p}, \mathbf{y}^j, t)dt \\
& \qquad k = 1, \ldots, n_c \quad [\mathrm{PBF}(\mathbf{y}^j)]
\end{aligned}$$

2.5 Relaxed Master problem

The Lower Bounding Convex MIDO problem (LBP) is a MIDO problem, the functions of which are convex (by construction) and once continuously differentiable (by Assumption 1). Therefore, the OA technique of Duran and Grossmann,[23] later corrected by Fletcher and Leyffer,[24] can be applied to derive an equivalent MILP, termed the Master problem, the solution value of which is identical to that of (LBP). That is, the solution value of the Master problem provides a valid lower bound to the solution value of (P). The Master problem is

$$\begin{aligned}
&\min_{\mathbf{p},\mathbf{y},\eta} \ \eta \\
\text{s.t.}\quad & \left.\begin{aligned} \eta &\geq \mathcal{L}_{u_{\phi_0}(\cdot,\cdot)+\int_{t_0}^{t_f} u_{\psi_0}(\cdot,\cdot,t)dt}(\mathbf{p},\mathbf{y})\big|_{\mathbf{p}^i,\mathbf{y}^i} \\ 0 &\geq \mathcal{L}_{u_{\phi_k}(\cdot,\cdot)+\int_{t_0}^{t_f} u_{\psi_k}(\cdot,\cdot,t)dt}(\mathbf{p},\mathbf{y})\big|_{\mathbf{p}^i,\mathbf{y}^i} \qquad k=1,\ldots,n_c \end{aligned}\right\} \quad \forall\, i \in \mathcal{T} \\
& 0 \geq \mathcal{L}_{u_{\phi_k}(\cdot,\cdot)+\int_{t_0}^{t_f} u_{\psi_k}(\cdot,\cdot,t)dt}(\mathbf{p},\mathbf{y})\big|_{\mathbf{p}^i,\mathbf{y}^i} \qquad k=1,\ldots,n_c \qquad \forall\, i \in \mathcal{S} \\
& \mathbf{p} \in P \\
& \mathbf{y} \in Y
\end{aligned} \tag{M}$$

where the sets $\mathcal{T}$ and $\mathcal{S}$ are defined as

$$\mathcal{T} = \{i : \text{PB}(\mathbf{y}^i) \text{ is feasible with optimal solution } \mathbf{p}^i\}$$

$$\mathcal{S} = \{i : \text{PB}(\mathbf{y}^i) \text{ is infeasible and } \mathbf{p}^i \text{ solves PBF}(\mathbf{y}^i)\}$$

and

$$\mathcal{L}_f(\mathbf{p},\mathbf{y})\big|_{\tilde{\mathbf{p}},\tilde{\mathbf{y}}} = f(\tilde{\mathbf{p}},\tilde{\mathbf{y}}) + \nabla f(\tilde{\mathbf{p}},\tilde{\mathbf{y}})^T \begin{pmatrix} \mathbf{p}-\tilde{\mathbf{p}} \\ \mathbf{y}-\tilde{\mathbf{y}} \end{pmatrix}$$

Remark 2.5. Rigorously speaking, the equivalence between problems (LBP) and (M) holds under the assertion of a constraint qualification at those solution points of the Primal Bounding problems on which the OA is based. This condition is satisfied by Assumption 2.

The Master problem is impractical to solve, however, because it requires the solution of all the Primal Bounding problems. Rather, a relaxation of the Master problem [referred to as (RM^j)] is considered, where the feasible set is progressively restricted by adding constraints at each new binary assignment, $\mathbf{y}^j$, visited by the algorithm. More specifically, the constraints correspond to linearizations of the objective and active constraint functions of the Lower Bounding Convex problem (LBP) at ($\mathbf{p}^j$, $\mathbf{y}^j$), where $\mathbf{p}^j$ denotes an optimal solution of problem [PB($\mathbf{y}^j$)] or its related feasibility problem [PBF($\mathbf{y}^j$)] as the case may be.[33] In the case of an infeasible Primal Bounding problem, note that the linearizations derived from the feasibility problem exclude the current binary assignment[24]; otherwise, an integer cut[48] is added to the Relaxed Master problem that excludes the currently examined binary realization. Accordingly, problem (RM^j) provides a new binary assignment, $\mathbf{y}^{j+1} \in Y$ (or the Relaxed Master problem becomes infeasible). Problem (RM^j) is defined as follows

$$\begin{aligned}
&\min_{\mathbf{p},\mathbf{y},\eta} \ \eta \\
&\text{s.t.} \quad \eta < \text{UBD} \\
& \left.\begin{aligned} \eta &\geq \mathcal{L}_{u_{\phi_0}(\cdot,\cdot)+\int_{t_0}^{t_f} u_{\psi_0}(\cdot,\cdot,t)dt}(\mathbf{p},\mathbf{y})\big|_{\mathbf{p}^i,\mathbf{y}^i} \\ 0 &\geq \mathcal{L}_{u_{\phi_k}(\cdot,\cdot)+\int_{t_0}^{t_f} u_{\psi_k}(\cdot,\cdot,t)dt}(\mathbf{p},\mathbf{y})\big|_{\mathbf{p}^i,\mathbf{y}^i} \quad k=1,\ldots,n_c \\ |\mathcal{B}^i| &\geq \sum_{k\in\mathcal{B}^i} y_k - \sum_{k\in\mathcal{NB}^i} y_k + 1 \\ \mathcal{B}^i &= \{k : y_k^i = 1\} \qquad \mathcal{NB}^i = \{k : y_k^i = 0\} \end{aligned}\right\} \forall\, i \in \mathcal{T}^j \\
& 0 \geq \mathcal{L}_{u_{\phi_k}(\cdot,\cdot)+\int_{t_0}^{t_f} u_{\psi_k}(\cdot,\cdot,t)dt}(\mathbf{p},\mathbf{y})\big|_{\mathbf{p}^i,\mathbf{y}^i} \qquad k=1,\ldots,n_c \qquad \forall\, i \in \mathcal{S}^j \\
& \mathbf{p} \in P \\
& \mathbf{y} \in Y
\end{aligned} \tag{RM^j}$$

where the sets $\mathcal{T}^j$ and $\mathcal{S}^j$ are defined as

$$\mathcal{T}^j = \{i | i \leq j : \text{PB}(\mathbf{y}^i) \text{ is feasible with optimal solution } \mathbf{p}^i\}$$

$$\mathcal{S}^j = \{i | i \leq j : \text{PB}(\mathbf{y}^i) \text{ is infeasible and } \mathbf{p}^i \text{ solves PBF}(\mathbf{y}^i)\}$$

Remark 2.6. Arbitrarily large parts of the feasible domain could be excluded if the linearizations were directly generated from the original nonconvex MIDO problem (P), instead of the Lower Bounding problem (LBP). By doing so, the global solution of the problem could be potentially excluded, thus reducing the MIDO algorithm to an ad hoc improvement strategy.

2.6 Illustrative example

Consider the following MIDO problem:

$$\begin{aligned}
&\min_{p,\mathbf{y}} \ \mathcal{J} = x_1^2(p,\mathbf{y},1) - x_2^2(p,\mathbf{y},1) \\
\text{s.t.}\quad & \left.\begin{aligned} \dot{x}_1(p,\mathbf{y},t) &= [-x_1(p,\mathbf{y},t) + 2y_1]p \\ \dot{x}_2(p,\mathbf{y},t) &= [2x_1(p,\mathbf{y},t) - x_2(p,\mathbf{y},t) + y_2]p \end{aligned}\right\} \quad \forall\, t \in [0,1] \\
& x_1(p,\mathbf{y},0) = 0 \\
& x_2(p,\mathbf{y},0) = 1 \\
& p \in [0.5,\ 1.5] = P \\
& \mathbf{y} \in \{0,1\}^2 = Y
\end{aligned} \tag{EX1}$$

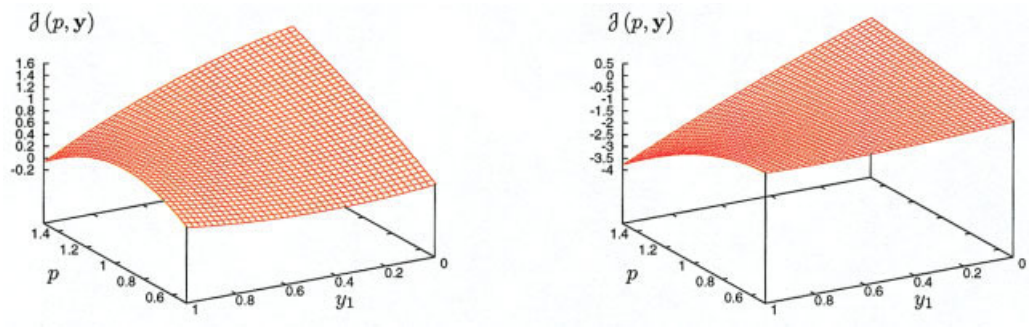

Figure 1. Objective function of problem (EX1) for $(p, y_1) \in P\times$ [0, 1].

Left plot: $y_2 = 0$; right plot: $y_2 = 1$.[Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

(EX1) is a nonconvex MIDO problem, the global solution value of which corresponds to $\mathcal{J}^* = -3.752$ and is attained at $p^* = 1.5$ for the binary assignment $\mathbf{y}^* = (1, 1)$. Figure 1 depicts $\mathcal{J}$ in the joint p–y_1 space, where variable y_2 is assigned the value 0 or 1.

A direct consequence of the nonconvexities in (EX1) is the presence of multiple local minima in several of the Primal problems. One such local minimum is $\mathcal{J} = 0.1835$ at $p = 0.5$, which occurs in the Primal problem obtained by fixing $y_1 = 1$ and $y_2 = 0$. Furthermore, the linearization of the objective function about this point is

$$\mathcal{L}_{\mathcal{J}}(p, y_1, y_2)\big|_{(\tilde{p},\tilde{y}_1,\tilde{y}_2)=(0.5,1,0)} = 0.1835 + 1.736(p - 0.5) + 0.5405(y_1 - 1) - 0.7612y_2$$

Note that an invalid cut would be generated if the linearization were derived from the nonconvex MIDO problem directly and not from the Lower Bounding Convex MIDO. In fact, as Figure 2 illustrates, most of the feasible domain of the problem would be excluded, including the global minimum of (EX1). Thus, an OA algorithm generating linearizations from the nonconvex MIDO problem itself would fail to locate the global solution and would terminate at an arbitrary suboptimal point.

As mentioned in subsection 2.1, the first and second steps for constructing a convex relaxation of the objective function consist of generating a convex underestimator **c** and a concave overestimator **C** of the solutions of the embedded differential system. The state variables and their relaxations at final time are depicted in Figure 3 in the joint p–y_1 space (by setting $y_2 = 0$). In this plot, the reference trajectory for linearizing the right-hand side of the differential system was chosen as $[\mathbf{x}^*(t), p^*, \mathbf{y}^*] = [\mathbf{x}^U(t), p^U, \mathbf{y}^U]$. Also note that the state bounds and relaxations were derived automatically, using an operator-overloading approach implemented in C++, because constructing the relaxed differential system yielding the state bounds and relaxations quickly becomes a tedious and error-prone task when performed manually.

A convex underestimator for the objective function is finally obtained by applying McCormick's composition technique from the bounds/relaxations of the state variables at the final time. Both terms in the objective function are relaxed separately, thus yielding the following overall convex underestimator $u_{\mathcal{J}}(p, \mathbf{y})$ for the objective function on $P\times [0, 1]^2$:

$$u_{\mathcal{J}}(p, \mathbf{y}) = [\mathrm{mid}\{c_1(p, \mathbf{y}, 1), C_1(p, \mathbf{y}, 1), z_1^{\min}\}]^2 + x_2^L(1) \times x_2^U(1) - [x_2^L(1) + x_2^U(1)] \times \mathrm{mid}\{c_2(p, \mathbf{y}, 1), C_2(p, \mathbf{y}, 1), z_2^{\max}\}$$

where

$$z_1^{\min} = \mathrm{mid}\{x_1^L(1), x_1^U(1), 0\}$$

$$z_2^{\max} = \begin{cases} x_2^L(1) & \text{if } |x_2^L(1)| > |x_2^U(1)| \\ x_2^U(1) & \text{otherwise} \end{cases}$$

The nonconvex objective function $\mathcal{J}$ of Problem (EX1) and its convex relaxation $u_{\mathcal{J}}$ in the joint continuous/discrete space are shown in Figure 4. Note that $u_{\mathcal{J}}$ is nonsmooth at some points

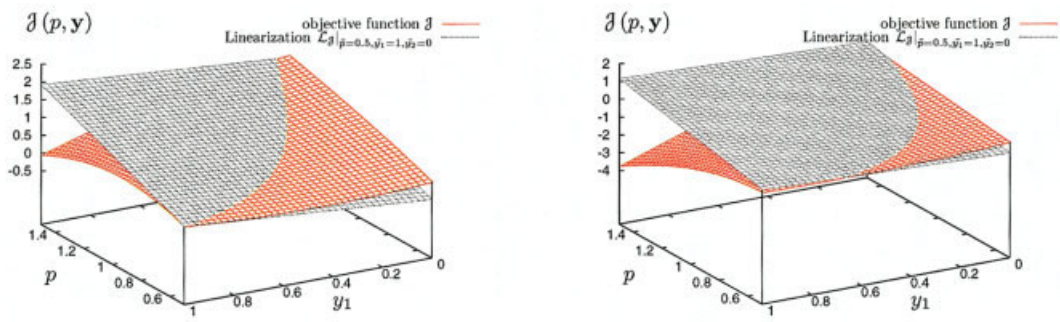

Figure 2. Objective function $\mathcal{J}$ of (EX1) and its linearization about $(\tilde{p}, \tilde{y}_1, \tilde{y}_2) = (0.5, 1, 0)$.

Left plot: $y_2 = 0$; right plot: $y_2 = 1$. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

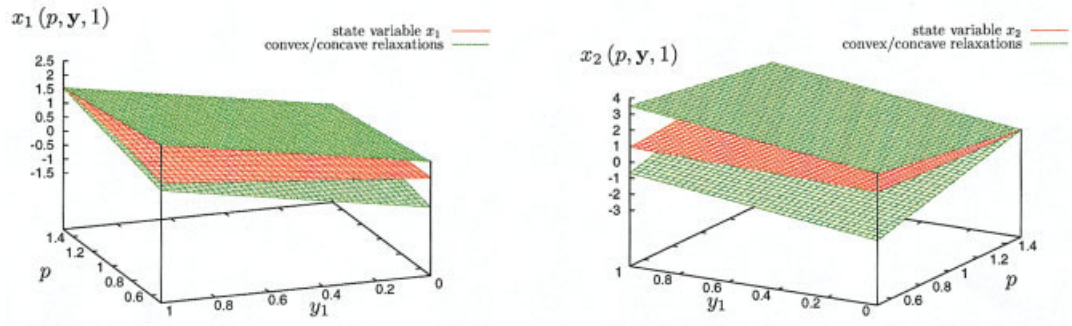

Figure 3. Convex and concave relaxations of the state variables at final time, for $(p, y_1) \in P \times [0, 1]$, $y_2 = 0$, with the outer-approximation point being chosen as $[\mathbf{x}^*(t), p^*, \mathbf{y}^*] = [\mathbf{x}^U(t), p^U, \mathbf{y}^U]$.

Left plot: x_1; right plot: x_2.

because of the use of the mid function and the conditional statement. An alternative, smooth Lower Bounding Convex MIDO can be formulated by introducing two new variables, ω_1 and ω_2, as well as constraints on ω_1 and ω_2 (see subsection 2.1). In this case, the following Lower Bounding Convex MIDO subproblem is obtained:

$$\min_{\mathbf{p},\mathbf{y},\omega_1,\omega_2} \omega_1^2 + x_2^L(1)x_2^U(1) - [x_2^L(1) + x_2^U(1)]\omega_2$$

$$\text{s.t.} \quad c_1(p, \mathbf{y}, 1) \le \omega_1 \le C_1(p, \mathbf{y}, 1)$$

$$x_1^L(1) \le \omega_1 \le x_1^U(1)$$

$$c_2(p, \mathbf{y}, 1) \le \omega_2 \le C_2(p, \mathbf{y}, 1)$$

$$x_2^L(1) \le \omega_2 \le x_2^U(1)$$

$$p \in P \qquad \mathbf{y} \in \{0, 1\}^2$$

The relaxed, convex objective function in the Primal Bounding problem is obtained by setting the relaxed binary variable y_2 to 0/1 in Figure 4. Furthermore, the cuts in the Relaxed Master problem correspond to the supporting planes at the minimum point of the Primal Bounding problem; these cuts are valid, for $\mathbf{u}_{\mathcal{J}}$ is convex on $P \times [0, 1]^2$

3 Algorithms

The algorithms developed for solving MIDO problems of the form (P) consist of solving an alternating sequence of Relaxed Master problems, Primal Bounding problems, and Primal problems. Two different versions are considered differing only on whether the nonconvex Primal problems are solved to guaranteed global optimality (algorithm 1) or local optimality (algorithm 2).

Algorithm 1: global solution of MIDO problems

Initialize

1. Set $j = 0$, $\ell = 1$, $\mathcal{F}^0 = \mathcal{T}^0 = \mathcal{U}^0 = \varnothing$.
2. UBD $= +\infty$, UBDPB $= +\infty$.
3. binary realization $\mathbf{y}^1$ is given.

REPEAT

IF ($j = 0$ or (RM^j) is feasible) THEN

REPEAT

1. Set $j = j + 1$.
2. Solve [PB($\mathbf{y}^j$)];
   IF [PB($\mathbf{y}^j$)] is infeasible, solve the feasibility problem [PBF($\mathbf{y}^j$)];
   Let a solution be $\mathbf{p}^j$.
3. Linearize the objective and active constraint functions of (LBP) about ($\mathbf{p}^j$, $\mathbf{y}^j$);

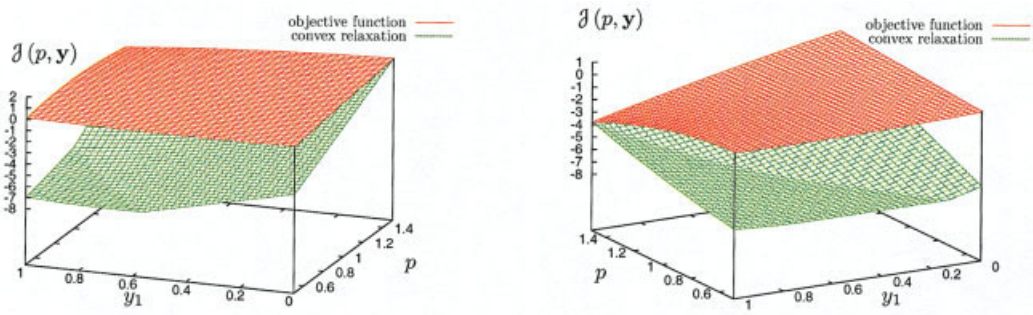

Figure 4. Convex relaxations of the objective function of problem (EX1) for $(p, y_1) \in P \times [0, 1]$ and the outer-approximation point being chosen as $[\mathbf{x}^*(t), p^*, \mathbf{y}^*] = [\mathbf{x}^U(t), p^U, \mathbf{y}^U]$.

Left plot: $y_2 = 0$; right plot: $y_2 = 1$. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

Set ($\mathcal{S}^j = \mathcal{S}^{j-1}$, $\mathcal{T}^j = \mathcal{T}^{j-1} \cup \{j\}$) or ($\mathcal{S}^j = \mathcal{S}^{j-1} \cup \{j\}$, $\mathcal{T}^j = \mathcal{T}^{j-1}$) as the case may be.

4. If [PB($\mathbf{y}^j$)] is feasible and $u_{\phi_0}(\mathbf{p}^j, \mathbf{y}^j) + \int_{t_0}^{t_f} u_{\psi_0}(\mathbf{p}^j, \mathbf{y}^j, t)dt <$ UBDPB, update $\mathbf{p}_B^* = \mathbf{p}^j$, $\mathbf{y}_B^* = \mathbf{y}^j$, $j_B^* = j$, UBDPB $= u_{\phi_0}(\mathbf{p}^j, \mathbf{y}^j) + \int_{t_0}^{t_f} u_{\psi_0}(\mathbf{p}^j, \mathbf{y}^j, t)dt$.
5. Solve the current relaxation (RM^j) of (P) (solution value η^j);
   Let a solution be $\mathbf{y}^{j+1}$.

UNTIL ($\eta^j \geq$ UBDPB or (RM^j) is infeasible)
END IF

IF (UBDPB $<$ UBD) THEN

1. Solve P($\mathbf{y}_B^*$) to global optimality;
   Set $\mathcal{U}^\ell = \mathcal{U}^{\ell-1} \cup \{j_B^*\}$;
   If P($\mathbf{y}_B^*$) is feasible, let a solution be $\mathbf{p}_P^j$, and if $\phi_0(\mathbf{p}_P^j, \mathbf{y}_B^*) + \int_{t_0}^{t_f} \psi_0(\mathbf{p}_P^j, \mathbf{y}_B^*, t)dt <$ UBD, update $\mathbf{p}_P^* = \mathbf{p}_P^j$, $\mathbf{y}_P^* = \mathbf{y}_B^*$, $j^* = j$, UBD $= \phi_0(\mathbf{p}_P^j, \mathbf{y}_B^*) + \int_{t_0}^{t_f} \psi_0(\mathbf{p}_P^j, \mathbf{y}_B^*, t)dt$.
2. If $\mathcal{T}^j \setminus \mathcal{U}^\ell \neq \varnothing$, update UBDPB $= \min\{u_{\phi_0}(\mathbf{p}^m, \mathbf{y}^m) + \int_{t_0}^{t_f} u_{\psi_0}(\mathbf{p}^m, \mathbf{y}^m, t)dt : m \in \mathcal{T}^j \setminus \mathcal{U}^\ell\}$, let ($\mathbf{p}^s$, $\mathbf{y}^s$) correspond to the Primal Bounding solution value UBDPB, update $\mathbf{p}_B^* = \mathbf{p}^s$, $\mathbf{y}_B^* = \mathbf{y}^s$, $\mathbf{j}_B^* = s$, set $\ell = \ell + 1$;
   Otherwise, set UBDPB $= +\infty$.

END IF

UNTIL (UBDPB $\geq$ UBD and ((RM^j) is infeasible or $\eta^j \geq$ UBD))

A global solution of problem (P) is given by the current UBD, $\mathbf{p}_P^*$, $\mathbf{y}_P^*$.

Remark 3.1. In the algorithm presented above, the global solution of the Primal problem [P($\mathbf{y}^j$)] is postponed until the lower bound exceeds the current best Primal Bounding problem [PB($\mathbf{y}^j$)] solution value (see, for example, Kesavan et al.[33] for a discussion of these aspects). Therefore, the computationally expensive task of solving the Primal problem to global optimality at every iteration is avoided. Furthermore, delayed solution of the Primal problem also provides an improved upper bound that can be used to solve efficiently other Primal problems. For example, if branch-and-bound is used to solve the nonconvex Primal problem to global optimality, the current upper bound on the overall problem can be used as an incumbent to fathom nodes thus reducing computational expense.

Remark 3.2. A variant of the outer approximation-algorithm was proposed by Gatzke and Barton[39] for solving nonconvex MINLPs on distributed memory parallel computing architectures. In this modified version, an alternating sequence of Primal Bounding and Relaxed Master problems is solved on the root node of the cluster, whereas a Primal problem is spawned on an available node whenever a new binary realization is generated with a feasible Primal Bounding solution less than the current upper bound. This parallel version of the outer-approximation algorithm can be readily adapted to solve MIDO problems, thus reducing the overall computational time and enabling solution of realistic engineering problems.

The following property holds for Algorithm 1:

Property 1 *If assumptions with respect to problems* (P) *and* (LBP) *hold,* $|Y| < \infty$, *and* ε*-optimality can be achieved in a finite number of steps for* [P($\mathbf{y}^j$)], *then Algorithm 1 either terminates in a finite number of steps with an* ε*-optimal solution of* (P) *or with an indication that problem* (P) *is infeasible.*

Proof. The proof directly follows from that presented in Kesavan et al.[33] □

3.2 Algorithm 2: rigorous bounds on the global solution value of MIDO problems

Solving Primal problems to global optimality typically represents the majority of the overall computational time. This requirement is weakened in Algorithm 2 because only a feasible point (if any) is provided for the Primal problem [P($\mathbf{y}^j$)]. Accordingly, the convergence properties of Algorithm 2 are weakened relative to Algorithm 1 because Algorithm 2 procedure produces valid upper and lower bounds only on the global solution value, plus a solution that may potentially be suboptimal. In this case, an upper bound is given by the current UBD, $\mathbf{p}_P^*$, $\mathbf{y}_P^*$, whereas a lower bound is obtained by

$$\text{LBD} = \min_{i \in \mathcal{T}^n} \text{PB}(\mathbf{y}^i)$$

where n is the iteration at which the last feasible Primal Bounding solution is obtained before the algorithm terminated. On finite termination of Algorithm 2, the distance between the global solution value of (P) and the solution value obtained, ($\mathbf{p}_P^*$, $\mathbf{y}_P^*$), is therefore guaranteed to be less than or equal to UBD$-$LBD.

Algorithm 2 is obtained by replacing the statement:

- "Solve P($\mathbf{y}_B^*$) to global optimality;"

in Algorithm 1 with the following statement:

- "Solve P($\mathbf{y}_B^*$) for any feasible point (global minimum unnecessary);"

Remark 3.3. A substantial reduction in the computational expense is generally achieved by application of Algorithm 2 in lieu of Algorithm 1. This is illustrated in section 4 for a practical example. In particular, we anticipate that this algorithm will be of great interest in solving realistic engineering problems such as large-scale chemical process design, or for other similar applications where obtaining a global solution of the problem is not essential. Furthermore, the likelihood of finding a global solution to the problem can be increased by solving the Primal problem to local optimality from a number of randomly selected starting points, by stopping the branch-and-bound algorithm after a fixed number of iterations, or terminating the branch-and-bound algorithm after some fixed amount of time.

3.3 Implementation issues

The proposed algorithms for solving MIDO problems rely heavily on the use of convex relaxations for the nonconvex functions ϕ_k and ψ_k. These relaxations are required for the solution of the Primal Bounding problems and are also used in the global solution of the Primal problems, such as whether a branch-and-bound procedure is used. A significant portion of the computational effort required to solve MIDO problems therefore lies in the numerical solution of the relaxed system of ODEs.

As described in subsection 2.1, computing the relaxations for the solutions of the embedded differential system consists of calculating the bounds $\mathbf{x}^L$ and $\mathbf{x}^U$ on the state variables as well as computing the functions $\mathbf{c}$ and $\mathbf{C}$ at a fixed time $\underline{t} \in [t_0, t_f]$.

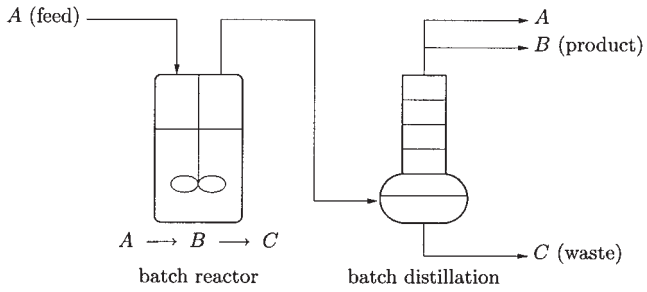

Figure 5. Task network of a two-stage batch process.

The state bounds are parameter independent by construction. Additionally, the affine structure of **c** and **C** allows us to write:

$$\mathbf{c}(\mathbf{p}, \mathbf{y}, \underline{t}) = \mathbf{M}_c(\underline{t})\begin{pmatrix}\mathbf{p} - \mathbf{p}^* \\ \mathbf{y} - \mathbf{y}^*\end{pmatrix} + \mathbf{n}_c(\underline{t}) \tag{1}$$

$$\mathbf{C}(\mathbf{p}, \mathbf{y}, \underline{t}) = \mathbf{M}_C(\underline{t})\begin{pmatrix}\mathbf{p} - \mathbf{p}^* \\ \mathbf{y} - \mathbf{y}^*\end{pmatrix} + \mathbf{n}_C(\underline{t}) \tag{2}$$

where $(\mathbf{p}^*, \mathbf{y}^*) \in P \times Y^c$ is a reference point; $\mathbf{M}_c$ and $\mathbf{M}_C$ denote the parametric sensitivities of **c** and **C** at $(\mathbf{p}^*, \mathbf{y}^*)$, respectively; $\mathbf{n}_c$ and $\mathbf{n}_C$ are the values of **c** and **C** at $(\mathbf{p}^*, \mathbf{y}^*)$, respectively. A direct consequence of the relationships 1 and 2 is that the values of **c** and **C** can be calculated by linear algebra instead of integrating the relaxed system of ODEs and its first-order sensitivities whenever $\mathbf{M}_c$, $\mathbf{M}_C$, $\mathbf{n}_c$, and $\mathbf{n}_C$ are already available (and provided the objective and constraints do not contain integral terms). Note in particular that if the bounds $[\mathbf{p}^L, \mathbf{p}^U]$ on the parameters remain constant with the binary variables fixed, then the derived quantities $\mathbf{M}_c$, $\mathbf{M}_C$, $\mathbf{n}_c$, and $\mathbf{n}_C$ remain constant as well. Therefore, one needs only to integrate the relaxed differential system once during the OA algorithm, whereas subsequent calculations of **c** and **C** can be performed without integration. Furthermore, a staggered corrector method can be used to compute the sensitivities efficiently.[49] Also note that similar considerations hold in a branch-and-bound algorithm for the global optimization of dynamic optimization problems as discussed in Singer and Barton.[38] Exploiting the affine structure of **c** and **C** thus leads to substantial reductions in the overall computational expense.

The efficiency of the OA algorithm can be further improved by applying bounds-tightening techniques to reduce the size of the parameter space. As the parameter bounds are tightened, so are the convex relaxations for the nonconvex functions, potentially enhancing the convergence of the OA iterative procedure. The time required to solve the Primal problem to global optimality may also be improved as the search space is reduced. In the OA algorithm presented above, a bounds-tightening procedure can be considered immediately before (or after) the Relaxed Master problem solution; it consists in solving $2 \times n_p$ additional MILP problems as follows:

$$\left.\begin{array}{l}\min/\max\limits_{\mathbf{p},y,\eta\ \ \mathbf{p},y,\eta} p_k \\ \text{s.t.} \quad \text{same constraints as in } (\text{RM}^j)\end{array}\right\} \quad k = 1, \ldots, n_p$$

Clearly, whether bounds tightening techniques should or should not be applied results from a trade-off between the computational expense associated with the additional MILPs and the overall reduction of computational time obtained from convergence enhancement. Also note that the relaxed system of ODEs (and its first-order sensitivities) must be reintegrated whenever the parameter bounds are modified, thus adding computational expense. Our experience in solving MIDO problems indicates that the bounds-tightening procedure is generally beneficial in Algorithm 1 because the majority of the CPU time is devoted to solving globally the Primal problems. On the contrary, bounds tightening tends to no longer be beneficial in Algorithm 2, given that the solution time for MILPs dominates. In this latter case, one may want to solve the bounds-tightening problems once every N iterations to reduce the cost associated with the additional MILPs. This is illustrated in the case study presented in the following section.

4 Case Study

The application of the MIDO algorithms is now demonstrated with an example based on a batch process. This case study was first considered in the work of Kesavan and Barton,[28] where rigorous bounds on the global solution value were obtained by applying the screening model approach developed by Allgor et al.[3]

The batch process under consideration is shown in Figure 5. It consists of a series reaction ($A \rightarrow B \rightarrow C$) followed by separation with no intermediate storage (NIS). A specified inventory of equipment, their rental cost, and their characteristics are reported in Table A1 (Appendix). The physical properties of the pure components A, B, and C are found in Table A2 (Appendix).

The objective of the problem is to select the optimal process design and operating conditions that minimize the overall manufacturing cost to produce a specified amount of product in a given time. The model equations are derived by assuming that the process operates at a cyclic steady state and by neglecting the start-up and shutdown of the process. Although the problem of determining optimal start-up and shutdown costs can be decoupled from that of the optimal cyclic steady-state cost, in general, these two problems are coupled in the determination of the overall cost of a manufacturing campaign. The inclusion of these aspects will be the topic of future work.

Reactors. The concentration profiles of the reactant (A) and products (B and C) are governed by the following set of ODEs:

$$\left.\begin{array}{l}\dot{C}_A = -k_1 C_A \\ \dot{C}_B = k_1 C_A - k_2 C_B \\ \dot{C}_C = k_2 C_B\end{array}\right\} \qquad \forall\, t \in [0, t_r^{\text{proc}}]$$

where $C_A(0) = C_{A_0}$, $C_B(0) = C_C(0) = 0$, and t_r^{proc} denotes the processing time for the reaction task. Note that, because the processing time for the operation is a decision variable, the time transformation outlined in Remark 2.3 must be applied before constructing convex relaxations. The temperature dependency of the kinetic rates k_1 and k_2 is expressed according to the Arrhenius rate expressions:

$$k_1 = k_{10}e^{-E_1/RT}$$

$$k_2 = k_{20}e^{-E_2/RT}$$

Furthermore, we consider the case where the activation energy (E_1) for the first reaction ($A \rightarrow B$) is lower than the activation energy (E_2) of the second reaction ($B \rightarrow C$); the values of the corresponding parameters are given in Table A3 (Appendix). From an extent of reaction analysis, the following invariant may be derived for the dynamic system

$$C_A(t) + C_B(t) + C_C(t) = C_{A_0} \qquad \forall\, t \in [0,\, t_r^{\text{proc}}]$$

and the concentrations in A, B, and C are nonnegative; the following set of natural bounds is thus used in deriving tight bounds for the state variables:

$$\left.\begin{array}{l} 0 \le C_A(t) \le C_{A_0} \\ 0 \le C_B(t) \le C_{A_0} \\ 0 \le C_C(t) \le C_{A_0} \end{array}\right\} \qquad \forall\, t \in [0,\, t_r^{\text{proc}}]$$

The following logical constraints enforce that at least one reactor is assigned for the reaction task and that the total volume processed per batch (V_b) is less than or equal to the volume of the assigned reactors:

$$\sum_{n=1}^{N_r^i} y_r^{i,n} \le 1 \qquad i = 1, \ldots, T_r$$

$$\sum_{i=1}^{T_r} \sum_{n=1}^{N_r^i} y_r^{i,n} \ge 1$$

$$\sum_{i=1}^{T_r} \sum_{n=1}^{N_r^i} n y_r^{i,n} V_r^i \ge V_b$$

where $T_r = 3$ and $N_r^i = 1$ ($i = 1, \ldots, T_r$); $y_r^{i,n}$ is a binary variable indicating whether n reactors of type i are used for the reaction task.

Distillation Columns. A perfect split of components is assumed for the batch distillation. This implies a sufficiently large number of trays and reflux ratio for the columns. Accordingly, the processing time t_d^{proc} for the distillation task can be calculated as

$$t_d^{\text{proc}} = V_b[C_A(t_r^{\text{proc}}) + C_B(t_r^{\text{proc}})] \sum_{i=1}^{T_d} \sum_{n=1}^{N_d^i} \frac{y_d^{i,n}(1 + R_{\min}^i)}{n v_d^i}$$

where $T_d = 6$ and $N_d^i = 6$ ($i = 1, \ldots, T_d$); $y_d^{i,n}$ is a binary variable indicating whether n columns of type i are used for the distillation task; $R_{\min}^i$ and v_d^i denote the minimum reflux ratio and the maximum vapor rate for distillation column of type i, respectively.

The following logical constraints enforce that distillation columns operating in parallel (if any exist) are of the same type and that the total volume processed per batch does not exceed the volume of the assigned columns:

$$\sum_{i=1}^{T_d} \sum_{n=1}^{N_d^i} y_d^{i,n} = 1$$

$$\sum_{i=1}^{T_d} \sum_{n=1}^{N_d^i} n y_d^{i,n} V_d^i \ge V_b$$

Two different cost minimization problems are subsequently investigated. A fixed production rate is imposed for product B in the first problem (subsection 4.2), whereas the second problem defines a fixed production amount of B to be manufactured over a given time horizon (subsection 4.3). Several aspects concerning the numerical implementation of the MIDO algorithms are detailed beforehand in the next subsection.

4.1 Numerical implementation

A number of general software components have been developed to implement Algorithms 1 and 2; they are detailed in the following list:

- The Primal Bounding problem relies on creation of convex relaxations for the nonconvex functions in Problem (P) as well as for the right-hand side of the embedded nonlinear differential equations (similar relaxations are also needed for the global solution of the Primal problem in Algorithm 1). Any nonlinear equality constraints are relaxed to a pair of convex inequalities. In this work, the relaxations are generated automatically based on an operator-overloading approach using C++ (only factorable nonconvex functions can be considered at the present time). The first-order derivatives and natural interval extensions are calculated similarly. Although the operator-overloading approach is known to be less efficient than code generation (the overhead increases with the number of operations), its use is convenient for algorithmic prototyping. To increase efficiency, a code generation technique is currently under development.
- A Feasibility problem is solved whenever a binary realization yields an infeasible Primal Bounding problem. It is constructed by defining positive slack variables that augment the problem to relax all constraints. The sum of the slack variables (ℓ^1 sum of constraint violation) is then minimized.
- Local dynamic optimizations are performed by using the code DYNO,[50] which implements a control parameterization approach. In Algorithm 1, the Primal problems are solved to global optimality by applying a branch-and-bound procedure that searches over the possible values for the continuous variables. This is achieved by using our in-house, C++ branch-and-bound library libBandB 3.2,[51] within a relative tolerance set to $\varepsilon = 10^{-3}$.
- The Relaxed Master problems are derived from linearizations of the Lower Bounding Convex MIDO problem. The Jacobian values for the participating functions are provided by DYNO, and the resulting MILPs are solved using CPLEX 9.0 callable libraries.[52]

Table 1. Results for the Minimization of the Overall Cost Subject to a Given Production Rate of *B*

Heuristics	Iterations	CPU Time (s)			
		Primal	Primal Bounding	Relaxed Master	Overall
MIDO algorithm version 1:					
Without domain reduction	57/32	4177	9.1	0.5	4187
With domain reduction	32/31	2828	13.4	1.1	2843
MIDO algorithm version 2:					
Without domain reduction	57/32	10.5	9.2	0.5	20.6
With domain reduction	32/31	8.5	13.1	1.2	23.0

The computational platform used to run the program was a 3.4-GHz Pentium 4 processor with 1 GB memory, running Linux 2.4.21. The compiler used was gcc 3.3.1.

4.2 Problem 1: minimum cost for a fixed production rate of B

In addition to the model equations and logical constraints presented previously, the objective function and production/time constraints for this problem are defined as follows:

Objective Function. The cost associated with each batch processed consists of the raw material cost, the waste disposal cost, and the equipment costs for the reactors and distillation columns:

$$\mathcal{J} = \underbrace{V_b C_{A_0} M_A CO_A^{\text{raw}}}_{\text{Raw material cost}} + \underbrace{V_b[C_A(t_r^{\text{proc}})M_A CO_A^{\text{disp}} + C_C(t_r^{\text{proc}})M_C CO_C^{\text{disp}}]}_{\text{Waste disposal cost}} + \underbrace{\left(\sum_{i=1}^{T_r}\sum_{n=1}^{N_r^i} n y_r^{i,n} CO_r^i\right)t^{\text{batch}}}_{\text{Equipment cost (reactor)}} + \underbrace{\left(\sum_{i=1}^{T_d}\sum_{n=1}^{N_d^i} n y_d^{i,n} CO_d^i\right)t^{\text{batch}}}_{\text{Equipment cost (column)}}$$

where t^{batch} denotes the time per batch; the cost of purchase of A (CO_A^{raw}), and the costs of disposal of A (CO_A^{disp}) and C (CO_C^{disp}) are specified in Table A4 (Appendix).

Fixed Production Constraint. Product B is to be manufactured at a fixed production rate of $r_B^{\text{out}} = 1$ kmol/h. The corresponding constraint in the optimization problem is given by

$$r_B^{\text{out}} t^{\text{batch}} = V_b C_B(t_r^{\text{proc}})$$

Time Constraints. These constraints enforce the time per batch t^{batch} to be greater than or equal to the total processing times for the react ion and distillation tasks:

$$t^{\text{batch}} \geq t_r^{\text{charge}} + t_r^{\text{proc}} + t_r^{\text{empty}}$$

$$t^{\text{batch}} \geq t_d^{\text{charge}} + t_d^{\text{reflux}} + t_d^{\text{proc}} + t_d^{\text{empty}}$$

where the times required to charge (t_r^{charge}) and empty (t_r^{empty}) a reactor, as well as the times required to charge (t_d^{charge}), empty (t_d^{empty}), and bring a column to total reflux (t_d^{reflux}) are given in Table A5 (Appendix).

The resulting problem is a nonconvex MIDO that contains one control variable ($T \in [298, 323$ K]), four design parameters ($t^{\text{batch}} \in [0, 10$ h], $t_r^{\text{proc}} \in [0, 10$ h], $t_d^{\text{proc}} \in [0, 10$ h], $V_b \in [0, 700$ L]), and 39 binary variables ($y_r^{i,n}$, $y_d^{i,n}$). By accounting for the existing SOS1 sets of binary variables, the total number of discrete alternatives in this problem is 252. Also note that the temperature profile is assumed to be constant, that is, it is simply considered as an additional decision variable (one can verify that optimizing the temperature as a time varying profile provides only a marginal reduction of the overall cost).

The global solution value corresponds to an overall cost of $\mathcal{J}^* = \$1.404 \times 10^5$ per batch. The optimal batch process configuration is as follows (see Table A1):

- reactor R1: 1 unit
- reactor R3: 1 unit
- distillation column C2: 1 unit

and the optimal values for the decision variables are $t^{\text{batch}} = 3.73$ h, $t_r^{\text{proc}} = 1.73$ h, $t_d^{\text{proc}} = 0.73$ h, $V_b = 480.9$ L, and $T = 314.2$ K.

The computational times for solving Problem 1 are reported in Table 1. The initial binary realization in these computations was chosen as: R1, 1 unit; C1, 1 unit. In agreement with the conclusions drawn in Kesavan et al.[33] for the solution of separable nonconvex MINLPs, the outer-approximation algorithm appears to be rather insensitive to the initial guess for the binary realization. The solution time for this problem using Algorithm 1 is around 70 min without using any bound reduction heuristics, whereas the application of the bounds tightening technique, as described in subsection 3.3, reduces the CPU time to about 47 min. The corresponding numbers of Relaxed Master problems and Primal problems solved by the algorithms are reported in the iteration column, respectively. In both cases, the outer-approximation algorithm avoids total enumeration and demonstrates its efficiency, given that only a small fraction of all possible process structure alternatives is visited. The iterations of MIDO Algorithm 1 for this problem are depicted in Figure 6: at each iteration j, the solution value of the Relaxed Master problem (RM^j) is displayed; the global solution value of the corresponding Primal problem [$\text{P}(\mathbf{y}^j)$], if feasible, is also represented (note that this value is not available from the algorithm when a better upper bound was found from the previous Primal problems; these values were computed independently for presentation purposes). Although the number of Primal problems solved is comparable in both cases (31 Primal solved with domain reduction, 32 otherwise), the bounds reduction improves the overall computational time by more than 30%. When the domain reduc-

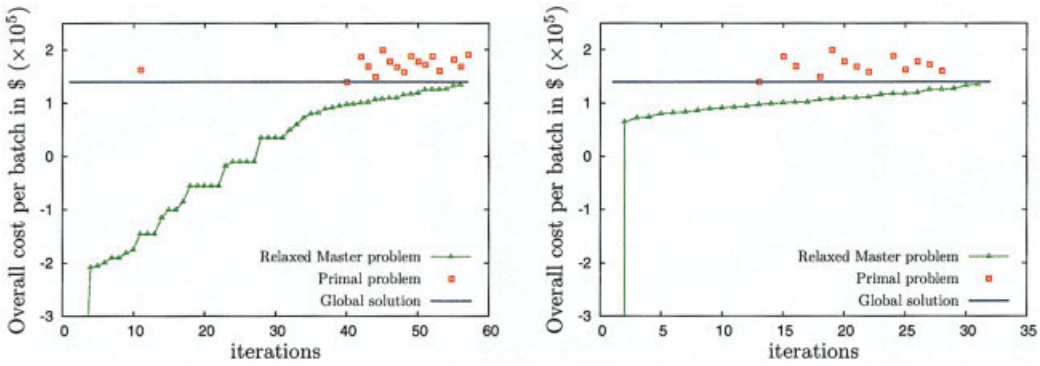

Figure 6. Minimization of the overall cost per batch subject to a given production rate of *B* (Algorithm 1).

Left plot: MIDO iterations without domain reduction heuristic; right plot: with domain reduction. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

tion heuristic is used, the convergence of the branch-and-bound algorithm is indeed enhanced because the search space in the Primal problem is progressively reduced. On the contrary, the CPU time required to solve the Primal Bounding problems increases when domain reduction is applied because the relaxed system of ODEs and its first-order sensitivities need to be recalculated whenever the parameter bounds are modified; the solution time for the Relaxed Master problems also increases because it accounts for the additional MILPs solved for updating the bounds.

The application of Algorithm 2 to solve this problem was also considered. It is worth noting that this simplified algorithm finds the global solution of Problem 1 despite the fact that no theoretical guarantee can be given. In addition, the CPU time required to solve the problem is dramatically reduced, given that Algorithm 2 terminates in about 20 s. Also note that the application of the bounds tightening heuristics is no longer beneficial because the modification of the parameter bounds requires additional integrations of the differential system to update the parametric sensitivities of **c** and **C** at time t_r^{proc} (see subsection 3.3).

4.3 Problem 2: minimum cost for a fixed production amount of B

Problem 2 imposes the production of a fixed amount $f_B^{\text{out}} = 30$ kmol of B over a given time horizon $t^{\text{horiz}} = 100$ h. In addition to selecting the optimal process structure and operating conditions, one must also determine the optimal number of batches N_b over the optimization horizon. The objective, the production constraints, and the time constraints are modified as follows:

$$\mathcal{J} = V_b[C_{A_0}M_A CO_A^{\text{raw}} + C_A(t_r^{\text{proc}})M_A CO_A^{\text{disp}} + C_C(t_r^{\text{proc}})M_C CO_C^{\text{disp}}]N_b + \left(\sum_{i=1}^{T_r}\sum_{n=1}^{N_r^i} ny_r^{i,n}CO_r^i + \sum_{i=1}^{T_d}\sum_{n=1}^{N_d^i} ny_d^{i,n}CO_d^i\right)t^{\text{camp}}$$

$$f_B^{\text{out}} = V_b C_B(t_r^{\text{proc}})N_b$$

$$t^{\text{camp}} \le t^{\text{horiz}}$$

$$t^{\text{camp}} \ge (t_r^{\text{charge}} + t_r^{\text{proc}} + t_r^{\text{empty}})N_b$$

$$t^{\text{camp}} \ge (t_d^{\text{charge}} + t_d^{\text{reflux}} + t_d^{\text{proc}} + t_d^{\text{empty}})N_b$$

where t^{camp} denotes the length of the manufacturing campaign. The model equations, as well as the logical constraints, remain unchanged.

The overall problem now contains one control variable ($T \in$ [298, 323 K], assumed constant), four design parameters ($t^{\text{camp}} \in$ [0, 100 h], $t_r^{\text{proc}} \in$ [0, 100 h], $t_d^{\text{proc}} \in$ [0, 100 h], $V_b \in$ [0, 700 L]), 39 binary variables ($y_r^{i,n}$, $y_d^{i,n}$), and one integer variable ($N_b \in \{1, \ldots, 33\}$). Note that this latter integer variable was reformulated as a SOS1 set of binary variables. Furthermore, the total number of discrete realizations in this example is 8316.

The global solution for Problem 2 corresponds to an overall cost of $\mathcal{J}^* = \$1.111 \times 10^6$. The optimal batch process configuration is as follows (see Table A1):

- reactor R3: 1 unit
- distillation column C1: 2 units
- $N_b = 9$ batches

and the optimal values for the decision variables are $t^{\text{camp}} = 44.37$ h, $t_r^{\text{proc}} = 2.93$ h, $t_d^{\text{proc}} = 1.93$ h, $V_b = 400$ L, and $T = 310.8$ K. Note that this design differs substantially from the optimal solution found in Problem 1. Also note that the global solution value found by the outer-approximation algorithm is better than the upper bound reported in Kesavan and Barton[28] for the same problem.

The computational times for solving Problem 2 are reported in Table 2. The application of Algorithm 1 provides the global solution to the problem in about 14.3 and 15.5 h, depending on whether domain reduction is applied; in both cases, total enumeration of the process structure alternatives is avoided, given that only 1082 binary realizations out of 8316 were visited in the former case and 1305 in the latter. As expected, the major computational expense derives from the solution of Primal problems to ε-optimality. Statistics indicate that the average CPU time for solving a Primal problem is about 50 s (with a maximum CPU time of 8 min), and the average number of nodes in the branch-and-bound tree is around 170 (with a maximum value of 1752). Also note that, for some binary realizations, the computational expense for proving Primal

Table 2. Results for the Minimization of the Overall Cost Subject to a Fixed Production Amount of *B*

Heuristics	Iterations	CPU Time (s)			
		Primal	Primal Bounding	Relaxed Master	Overall
MIDO algorithm version 1:					
Without domain reduction	1305/1266	53,631	272	1584	55,656
With domain reduction	1082/1062	50,416	235	703	51,465
MIDO algorithm version 2:					
Without domain reduction	1305/1266	334	269	1587	2357
With domain reduction	1082/1062	371	234	703	1417

infeasibility may take as long as the solution of a feasible Primal problem to finite ε-optimality.

Roughly, the more promising binary realizations are visited early by the outer approximation algorithm, and the upper bound UBD rapidly reaches the global minimum of the problem (after about 200 iterations); this can be seen from Figure 7 (left plot), which depicts the solution values of the Primal and Relaxed Master problems vs. the iteration count (counter j; see subsection 3.1). By using UBD as an incumbent in the branch-and-bound procedure, the computational expense for subsequent Primal problems is then progressively reduced as it becomes faster to detect whether a given binary realization will yield a worse upper bound or is infeasible. These considerations are illustrated in the right plot in Figure 7, which depicts the computational expense vs. the iteration count. Also note that the reduction of CPU time obtained under the application of domain reduction is not as large as it was in problem 1. This is because domain reduction is effective primarily during the last iterations, that is, when solving Primal problems becomes cheaper.

The application of Algorithm 2 was also investigated for this problem. One sees from Table 2 that the computational expense for solving the MIDO problem is again dramatically reduced in that case. Also note that the use of the bounds tightening heuristics (one bound reduction performed every 10 iterations) reduces the number of Relaxed Master problems from 1305 to 1082, and the number of Primal problems from 1266 to 1062. Correlatively, a substantial decrease of the computational time is observed as the bounds are tightened from 40 to <24 min. This rather unexpected decrease of CPU time is attributed to the reduced number of Relaxed Master Problems solved. Indeed, as the number of iterations grows, so does the number of constraints in the Relaxed Master MILP, and the associated computational effort for solving the Relaxed Master MILP becomes dominant. This result demonstrates the utility of using the domain reduction heuristics in Algorithm 2.

Convergence Enhancement from the Use of Screening Model Cuts. Allgor et al.[3] introduced screening models for batch processes consisting of reaction and distillation tasks, and illustrated their role in the development of batch processes. Because the solution of screening models yields rigorous lower bounds on the cost of the resulting design, they can be used as the foundation for a rigorous decomposition algorithm for the detailed design of a batch process using MIDO.[2] Although this procedure finds rigorous upper and lower bounds on the global solution value, it may, however, terminate with a solution that is potentially suboptimal.

In this subsection, we consider valid cuts derived from the screening model and add them to the relaxed MIDO problem formulated in subsection 2.2 to enhance the convergence of the MIDO algorithms. Let ξ_1 and ξ_2 denote the extents of reactions $A \rightarrow B$ and $B \rightarrow C$, respectively. Because the reactor is assumed to operate isothermally, the following valid upper and lower bounds are obtained on ξ_1 and ξ_2 (see Allgor and Barton[27] for discussion):

$$\dot{f}_A^{\text{in}}(1 - e^{-k_1^L t_r^{\text{proc}}}) \le \xi_1 \le \dot{f}_A^{\text{in}}(1 - e^{-k_1^U t_r^{\text{proc}}})$$

$$\xi_1\left(1 + \frac{k_1^L e^{-k_2^L t_r^{\text{proc}}}}{k_2^L - k_1^L} - \frac{k_2^L e^{-k_1^L t_r^{\text{proc}}}}{k_2^L - k_1^L}\right) \le \xi_2 \le \xi_1(1 - e^{-k_2^U t_r^{\text{proc}}})$$

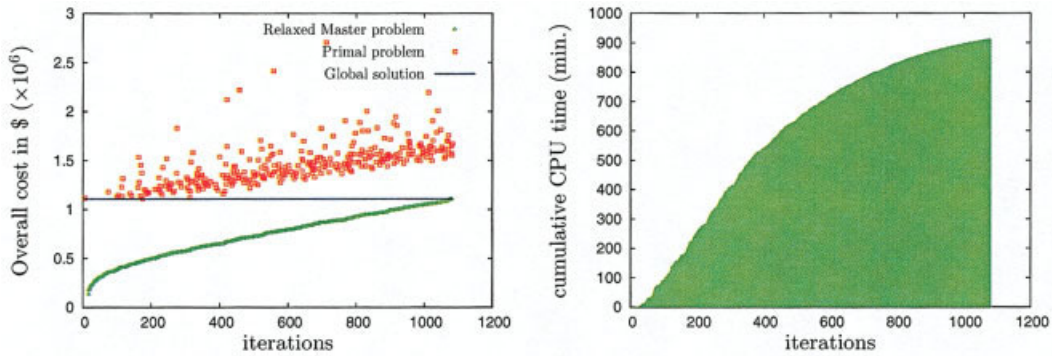

Figure 7. Minimization of the overall cost to produce a given amount of *B* (Algorithm 1).

Left plot: MIDO iterations; right plot: CPU time (with domain reduction). [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

Table 3. Results for the Minimization of the Overall Cost Subject to a Fixed Production Amount of *B*, with Screening Model Cuts Added

Heuristics	Iterations	CPU Time (s)			
		Primal	Primal Bounding	Relaxed Master	Overall
MIDO algorithm version 1: With domain reduction and screening model cuts	512/482	12,243	303	420	13,020
MIDO algorithm version 2: With domain reduction and screening model cuts	512/482	66	298	421	839

where

$$k_1^U = k_{10} e^{-E_1/RT^U}$$

$$k_2^U = k_{20} e^{-E_2/RT^U}$$

$$k_1^L = k_{10} e^{-E_1/RT^L}$$

$$k_2^L = k_{20} e^{-E_2/RT^L}$$

When expressed in terms of the reaction extents ξ_1 and ξ_2, the processing time for the distillation column and the fixed production constraint for B are

$$t_d^{\text{proc}} = (\dot{f}_A^{\text{in}} - \xi_2) \sum_{i=1}^{T_d} \sum_{n=1}^{N_d^i} \frac{y_d^{i,n}(1 + R_{\min}^i)}{n v_d^i}$$

$$f_B^{\text{out}} = (\xi_1 - \xi_2) N_b$$

Additionally, the equations yielding the raw and waste material costs are transformed as

$$\mathcal{J}^{\text{raw}} = \dot{f}_A^{\text{in}} M_A CO_A^{\text{raw}} N_b$$

$$\mathcal{J}^{\text{disp}} = V_b[(\dot{f}_A^{\text{in}} - \xi_1) M_A CO_A^{\text{disp}} + \xi_2 M_C CO_C^{\text{disp}}] N_b$$

Because the aforementioned equality and inequality constraints are nonconvex, they must be relaxed before adding them into the relaxed MIDO problem. The results obtained by considering these additional cuts are given in Table 3 for Algorithms 1 and 2. One sees that the overall number of iterations is reduced by half in both cases because the screening model cuts allow tighter relaxations, thus excluding a larger number of binary realizations. For Algorithm 1, these additional cuts also improve the convergence of the branch-and-bound algorithm used for solving the Primal problems to ε-optimality; fewer iterations are indeed required because the relaxations are tighter and infeasible Primal problems are detected earlier. Accordingly, Problem 2 can now be solved to guaranteed global optimality in less than 3.7 h. Furthermore, the global solution of Problem 2 is also found with Algorithm 2, while enabling a substantial reduction in computational time. Note, however, that the computational expense for solving Primal Bounding problems is increased when screening model cuts are added because the relaxations contain the additional continuous variables ξ_1 and ξ_2 as well as additional constraints (see Table 2 for comparison). Nevertheless, the corresponding overhead remains small in comparison to the overall improvement. These results therefore demonstrate that large benefits can be realized by considering screening models for batch process development purposes.

5 Conclusions

In this article, we presented an algorithm for solving MIDO problems to guaranteed global optimality that potentially avoids total enumeration of the discrete alternatives. The algorithm implements an outer-approximation method for nonconvex MIPs in combination with a relaxation technique for constructing convex underestimators of functions with state variables participating. It consists of solving an alternating sequence of Relaxed Master, Primal Bounding, and Primal problems. Two variants of the algorithm were considered differing in whether the Primal problems are solved to ε-optimality (Algorithm 1) or only for any feasible point (Algorithm 2). On finite termination, the former finds a global solution of the MIDO problem, whereas the latter yields rigorous bounds bracketing the global solution value and a solution that is potentially suboptimal. Several aspects concerning numerical implementation were also discussed to demonstrate techniques for improving the convergence rate of the algorithms.

After describing the mathematical foundations of the MIDO algorithm, a case study was presented with respect to the batch process development. The objective was to select the optimal process design and operation that minimize the overall manufacturing cost subject to either a fixed production rate constraint (problem 1) or a fixed amount of product in given time (problem 2). The algorithms demonstrate applicability in solving both problems. Interestingly, the global solution of Problems 1 and 2 was found by Algorithm 2, while greatly reducing the computational expense relative to Algorithm 1. More generally, we conjecture that the use of Algorithm 2 may be of great interest for solving those practical engineering problems for which obtaining a global solution is not absolutely essential. Finally, a link was established to the screening model approach developed in Allgor et al.[3] by introducing additional cuts to tighten the MIDO problem relaxations. We found that these cuts provided substantial reductions in terms of the number of iterations and the overall computational time.

Current research aims at reducing the overall computational expense, especially for Algorithm 1, to deal with large-scale problems in a reasonable amount of time. In particular, the focus is on the derivation of tighter convex relaxations for functions with state variables participating. Great advantage could also arise by considering the generalized branch-and-cut framework developed by Kesavan and Barton[35] to reduce the

number of binary realizations at which the Primal is solved. However, it should be noted that the running time of the algorithms is much more sensitive to the number of binary variables **y** and parameters **p** than to the number of state variables **x**. Therefore, global mixed-integer optimization with a complex dynamic model and a moderate number of decision variables **y**, **p** appears within reach.

Acknowledgments

B. Chachuat is grateful to the French Ministry of Foreign Affairs (Lavoisier postdoctoral fellowships) for financial support. This work is based on work supported by the National Science Foundation under Grant 0120441.

Notation

Acronyms

DAE = differential-algebraic equation
GBC = generalized branch-and-cut
GBD = generalized Benders decomposition
LBD = lower bound
LTV = linear time-varying (system)
MIDO = mixed-integer dynamic optimization
MILP = mixed-integer linear programming
MINLP = mixed-integer nonlinear programming
NLP = nonlinear programming
OA = outer-approximation
ODE = ordinary differential equation
UBD = upper bound for the primal problem
UBDPB = upper bound for the primal bounding problem

Batch process notation

C_{A_0} = initial concentration of A charged into the reactor
C_A, C_B, C_C = concentrations of A, B, and C during reaction task
CO_A^{raw} = cost of purchase of unit mass of A
CO_A^{disp}, CO_C^{disp} = cost of disposal of unit mass of A and C
CO_d^i = rental rate for distillation column of type i
CO_r^i = rental rate for reactor of type i
E_1, E_2 = activation energies for reactions $A \rightarrow B$ and $B \rightarrow C$, respectively
f_A^{in} = amount of A available for reaction
f_B^{out} = total moles of B to be produced
k_1, k_2 = kinetic rates for reactions $A \rightarrow B$ and $B \rightarrow C$, respectively
k_{01}, k_{02} = preexponential factors for reactions $A \rightarrow B$ and $B \rightarrow C$, respectively
M_A, M_B, M_C = molecular weight of A, B, and C
N_b = number of batches during the manufacturing campaign
N_d^i = number of column units of type i in the manufacturing facility
N_r^i = number of reactor units of type i in the manufacturing facility
r_B^{out} = production rate of B
R_{min}^i = reflux ratio for proper gas/liquid contacting in distillation column of type i
t^{batch} = total time per batch
t^{camp} = length of the manufacturing campaign
t^{horiz} = time horizon for manufacture
t_d^{proc} = processing time per batch for distillation column
t_d^{charge} = time required to charge one batch of material for distillation column
t_d^{reflux} = time required to bring column to total reflux
t_d^{empty} = time required to empty one batch of material for distillation column
t_r^{proc} = processing time for reaction task
t_r^{charge} = time required to charge one batch of material for reactor
t_r^{empty} = time required to empty one batch of material for reactor
T = temperature of operation for reaction task
T_d = types of distillation columns in the manufacturing facility
T_r = types of reactors in the manufacturing facility
v_d^i = maximum vapor rate for distillation column of type i
V_b = volume of material processed per batch
V_d^i = volume of column of type i
V_r^i = volume of reactor of type i
$y_r^{i,n}$ = Are n reactors of type i being used for the reaction task?
$y_d^{i,n}$ = Are n columns of type i being used for the distillation task?
ξ_1, ξ_2 = extents of reactions $A \rightarrow B$ and $B \rightarrow C$, respectively

Mathematical notation

$\mathcal{J}$ = objective function
$\mathcal{L}$ = linearization operator
n_c = number of constraints
n_p = number of continuous decision variables
n_x = number of state variables
n_y = number of discrete decision variables
o = concave overestimator
$\mathbf{p}$ = continuous decision variables
P = parameter set
t = time
t_0 = initial time
t_f = final time
u = convex underestimator
$\mathbf{x}$ = state variables
$\mathbf{y}$ = discrete decision variables
Y = discrete variable space
Y^c = convex hull of Y

Literature Cited

1. Allgor RJ, Barrera MD, Barton PI, Evans LB. Optimal batch process development. *Comput Chem Eng*. 1996;20:885-896.
2. Allgor RJ, Barton PI. Mixed-integer dynamic optimization. I—Problem formulation. *Comput Chem Eng*. 1999;23:567-584.
3. Allgor RJ, Evans LB, Barton PI. Screening models for batch process development, Part 1. Design targets for reaction/distillation networks. *Chem Eng Sci*. 1999;54:4145-4164.
4. Bhatia TK, Biegler LT. Dynamic optimization in the design and scheduling of multiproduct batch plants. *Ind Eng Chem Res*. 1996;35:2234-2246.
5. Charalambides MS. *Optimal Design of Integrated Batch Processes.* PhD Thesis. London: UK: University of London; 1996.
6. Low KH, Sorensen E. Simultaneous optimal design and operation of multipurpose batch distillation columns. *Chem Eng Process*. 2004;43:273-289.
7. Oldenburg J, Marquardt W, Heinz D, Leineweber DB. Mixed-logic dynamic optimization applied to batch distillation process design. *AIChE J*. 2003;49:2900-2917.
8. Sharif M, Shah N, Pantelides CC. On the design of multicomponent batch distillation columns. *Comput Chem Eng*. 1998;22:S69-S76.
9. Giovanoglou A, Barlatier J, Adjiman CS, Pistikopoulos EN, Cordiner JL. Optimal solvent design for batch separation based on economic performance. *AIChE J*. 2003;49:3095-3109.
10. Bansal V, Sakizlis V, Ross R, Perkins JD, Pistikopoulos EN. New algorithms for mixed-integer dynamic optimization. *Comput Chem Eng*. 2003;27:647-668.
11. Kookos IK, Perkins JD. An algorithm for simultaneous process design and control. *AIChE J*. 2001;40:4079-4088.
12. Mohideen MJ, Perkins JD, Pistikopoulos EN. Towards an efficient numerical procedure for mixed integer optimal control. *Comput Chem Eng*. 1997;21:S457-S462.
13. Schweiger CA, Floudas CA. Interaction of design and control: Optimization with dynamic models. In: Hager WW, Pardalos PM, eds. *Optimal Control: Theory, Algorithms, and Applications.* Dordrecht, The Netherlands: Kluwer Academic; 1997:388-435.
14. Androulakis IP. Kinetic mechanism reduction based on an integer programming approach. *AIChE J*. 2000;46:361-371.

15. Petzold L, Zhu W. Model reduction for chemical kinetics: An optimization approach. *AIChE J*. 1999;45:869-886.
16. Avraam MP, Shah N, Pantelides CC. A decomposition algorithm for the optimisation of hybrid dynamic processes. *Comput Chem Eng*. 1999;23:S451-S454.
17. Barton PI, Lee CK. Design of process operations using hybrid dynamic optimization. *Comput Chem Eng*. 2004;28:955-969.
18. Galan S, Barton PI. Dynamic optimization of hybrid systems. *Comput Chem Eng*. 1998;22:S183-S190.
19. Papamichail I, Adjiman CS. A rigorous global optimization algorithm for problems with ordinary differential equations. *J Global Optim*. 2002;24:1-33.
20. Singer AB, Barton PI. Global solution of linear dynamic embedded optimization problems. *J Optim Theor Appl*. 2004;121:149-182.
21. Cuthrell JE, Biegler LT. On the optimization of differential-algebraic process systems. *AIChE J*. 1987;33:1257-1270.
22. Teo K, Goh G, Wong K. *A Unified Computational Approach to Optimal Control Problems* (Pitman Monographs and Surveys in Pure and Applied Mathematics). New York, NY: Wiley; 1991.
23. Duran MA, Grossmann IE. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Math Program*. 1986;36: 307-339.
24. Fletcher R, Leyffer S. Solving mixed-integer nonlinear programs by outer approximation. *Math Program*. 1994;66:327-349.
25. Geoffrion AM. Generalized Benders decomposition. *J Optim Theor Appl*. 1972;10:237-262.
26. Allgor RJ, Barton PI. Mixed-integer dynamic optimization. *Comput Chem Eng*. 1997;21:S451-S456.
27. Allgor RJ, Barton PI. Screening models for batch process development, Part 2. Case studies. *Chem Eng Sci*. 1999;54:4165-4187.
28. Kesavan P, Barton PI. Numerical experience with mixed-integer dynamic optimization. Proc of NSF Design and Manufacturing Grantees Conf, Long Beach, CA, January 5–8; 1999.
29. Ryoo HS, Sahinidis NV. Global optimization of nonconvex NLPs and MINLPs with applications in process design. *Comput Chem Eng*. 1995;19:551-566.
30. Tawarmalani M, Sahinidis NV. Global optimization of mixed-integer nonlinear programs: A theoretical and practical study. *Math Program*. 2004;99:563-591.
31. Adjiman CS, Androulakis IP, Floudas CA. Global optimization of mixed-integer nonlinear problems. *AIChE J*. 2000;46:1769-1797.
32. Smith EMB, Pantelides CC. A symbolic reformulation/spatial branch-and-bound algorithm for the global optimization of nonconvex MINLPs. *Comput Chem Eng*. 1999;23:457-478.
33. Kesavan P, Allgor RJ, Gatzke EP, Barton PI. Outer approximation algorithms for separable nonconvex mixed-integer nonlinear programs. *Math Program*. 2004;100:517-535.
34. Kesavan P, Barton PI. Decomposition algorithms for nonconvex mixed-integer nonlinear programs. *AIChE Symp Ser*. 2000;96:458-461.
35. Kesavan P, Barton PI. Generalized branch-and-cut framework for mixed-integer nonlinear optimization problems. *Comput Chem Eng*. 2000;24:1361-1366.
36. Lee CK, Barton PI. Determining the optimal mode sequence. In: Engell S, Zaytoon J, Gueguen H, eds. Proc of the IFAC Conf on Analysis and Design of Hybrid Systems, St. Malo, France; 2003.
37. Singer AB, Barton PI. Bounding the solutions of parameter dependent nonlinear ordinary differential equations. *Journal*. 2004;00:000-000.
38. Singer AB, Barton PI. Global optimization with nonlinear ordinary differential equations. *Journal*. 2004;00:000-000.
39. Gatzke EP, Barton PI. Parallel mixed-integer nonlinear optimization using outer-approximation methods. Proc of AIChE Annual Meeting, Reno, NV, November 4–9; 2001.
40. McCormick GP. Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems. *Math Program*. 1976;10:147-175.
41. Tawarmalani M, Sahinidis NV. Convexification and global optimization in continuous and mixed-integer nonlinear programming: Theory, algorithms, software, and applications. *Nonconvex Optimization and Its Applications*. Boston, MA: Kluwer Academic; 2002.
42. Zamora JM, Grossmann IE. A branch and contract algorithm for problems with concave univariate, bilinear and linear fractional terms. *J Global Optim*. 1999;14:217-249.
43. Falk JE, Soland RM. An algorithm for separable nonconvex programming problems. *Manage Sci*. 1969;15:550-569.
44. Adjiman CS, Androulakis IP, Floudas CA, Neumaier A. A global optimization method, αBB, for general twice-differentiable constrained NLPs—I. Theoretical advances. *Comput Chem Eng*. 1998;22: 1137-1158.
45. Chachuat B, Latifi MA. A new approach in deterministic global optimization of problems with ordinary differential equations. In: Floudas CA, Pardalos PM, eds. *Frontiers in Global Optimization: Nonconvex Optimization and Its Applications*. Vol. 74. Dordrecht, The Netherlands: Kluwer Academic; 2003:83-108.
46. Lee CK, Singer AB, Barton PI. Global optimization of linear hybrid systems with explicit transitions. *Syst Control Lett*. 2004;51:363-375.
47. McCormick GP. *Nonlinear Programming: Theory, Algorithms and Applications.* New York, NY: Wiley; 1983.
48. Balas E, Jeroslow R. Canonical cuts on the unit hypercube. *SIAM J Appl Math*. 1972;23:61-79.
49. Feehery WF, Tolsma JE, Barton PI. Efficient sensitivity analysis of large-scale differential-algebraic systems. *Appl Numer Math*. 1997;25: 41-54.
50. Fikar M, Latifi MA. User's guide for Fortran dynamic optimisation code DYNO, Technical Report. Nancy, France: LSGC-CNRS and Bratislava, Slovak Republic: Slovak Technical University Bratislava; 2002.
51. Singer AB. LibBandB.a version 3.2 manual, Technical Report. Cambridge, MA: Massachusetts Institute of Technology; 2004.
52. ILOG. CPLEX Callable Library 9.0, Reference Manual; 2003.

Appendix: Data for the Batch Process

Table A1. Available Equipment for Reaction and Separation Tasks

Equipment Type	No. of Units Available	Rental Cost ($/h)	Volume (L)	Maximum Vapor Rate (kmol/h)	Minimum Reflux Ratio
Reactor R1	1	2500	100	n/a	n/a
Reactor R2	1	5000	200	n/a	n/a
Reactor R3	1	8000	400	n/a	n/a
Column C1	6	3000	500	2	1
Column C2	6	10,000	500	16	1.5
Column C3	6	15,000	100	12	0.5
Column C4	6	15,000	100	24	2
Column C5	6	15,000	200	32	1
Column C6	6	15,000	200	48	0.8

Table A2. Physical Properties of the Pure Components *A*, *B*, and *C*

Component	Molecular Weight (g/mol)	Molar Volume (cm^3/mol)
A	100	1
B	100	1
C	100	1

Table A3. Constants for Arrhenius Rate Expression

	Case $E_1 < E_2$	
Reaction	k_{i0} (h^{-1})	E_i (kcal mol^{-1})
$A \rightarrow B$	32.10×10^{11}	18
$B \rightarrow C$	27.66×10^{18}	30

Table A4. Cost of Purchase/Disposal of *A* and *C*

Parameter	Cost ($/kg)
CO_A^{raw}	100
CO_A^{disp}	10
CO_C^{disp}	1000

Table A5. Fixed Processing Times for the Reaction and Distillation Tasks

Parameter	Time (h)	Parameter	Time (h)
t_r^{charge}	1	t_d^{charge}	1
t_r^{empty}	1	t_d^{reflux}	1
		t_d^{empty}	1

Manuscript received May 11, 2004, and revision received Dec. 13, 2004.