Logic-Based Outer-Approximation Algorithm for Solving Discrete-Continuous Dynamic Optimization Problems

Ruben Ruiz-Femenia,Antonio Flores-Tlacuahuac,[*] Ignacio E. Grossmann

Department of Chemical Engineering
Carnegie-Mellon University
5000 Forbes Avenue, Pittsburgh, PA, 15213

April 11, 2013

Abstract

In this work we present an extension of the Logic Outer-Approximation algorithm for dealing with disjunctive discrete-continuous optimal control problems whose dynamic behavior is modeled in terms of differential-algebraic equations. Although the proposed algorithm can be applied to a wide variety of discrete-continouos optimal control problems we are mainly interested in problems where disjunctions are also present. Disjunctions are included to take into account only certain parts of the underlying model which become relevant under some processing conditions. By doing so the numerical robustness of the optimization algorithm improves since those parts of the model that are not active are discarded leading to a reduced size problem and avoiding potential model singularities. We test the proposed algorithm using 3 examples of different complex dynamic behavior. In all the case studies the number of iterations and the computational load required to get the optimal solutions is modest and the solutions are relatively easy to find

[*]Author to whom correspondence should be addressed. On leave from Universidad Iberoamericana, México. E-mail: antonio.flores@ibero.mx

1 Introduction

Traditionally the modeling, simulation, optimization and control of processing systems involves only continuous variables [1], [2]. However, new applications, especially for industrially relevant systems, require the use of discrete variables. It must also be pointed out that the implementation of control systems in real environments is always performed through digital or discrete control systems [3]. However, the discrete or hybrid modeling of chemical processing systems has only been considered recently [4].

The optimization of discrete and continuous systems, herein referred as hybrid systems, has been mainly directed towards optimizing the dynamics of start-up and shut-down operations [5], grade transitions [6], [7], batch systems [8], scheduling and control systems [9], [10], simultaneous design and control [11], [12], [13], [14]. The formal optimization formulation of these problems gives rise to Mixed-Integer Dynamic Optimization (MIDO) problems whose solution can be tackled by well known numerical approaches [15]. Sometimes the dynamic value of some decision variables (i.e. the control actions) may involve logic decisions that can be modeled as disjunctions [16], [17] giving rise to Logic MIDO (LMIDO) problems. In this case LMIDO problems are normally transformed into MIDO problems using either a big-M or convex-hull relaxations to approximate the behavior of the original logic disjunctions [18]. Once the problem is in a MIDO framework full discretization techniques [19], [15] (also called the direct transcription approach [20]) can be deployed to transform it into a normally large scale and sparse Mixed-Integer Nonlinear Programming (MINLP) problem whose solution can be sought by known techniques such as outer-approximations [21], Benders decomposition [22] or branch and bound techniques [23]. There are some other approaches suggested for solving MIDO problems based on partial discretization of the underling system coupled with the use of robust and efficient numerical integration solvers [24]. Another interesting approach to solve LMIDO problems is applying full discretization techniques based on the use of complementarity constraints [5], [25], [26] . In this approach LMIDO problems are discretized into Nonlinear Programming (NLP) problems avoiding the use of discrete decision variables and the need of solving complex MINLP problems. However, the use of complementarity constraints introduces non-convexities to the underlying problem increasing the complexity of the NLP problem. However, this is a promising optimization approach that deserves more research efforts.

As it is clear from the above discussion, most of the hybrid MINLP problems that have been addressed deal with the switching behavior of processing systems. Under this situation, the discrete variable is normally related to some control action such as opening/closing control valves, changing the scheduling production of a given system composed of several products, switching on/off heating/cooling systems, etc. Within this framework hybrid MINLP problems are commonly related to hybrid control problems [27], [28]. In most of the cited examples the structure of the underlying dynamic mathematical model remains the same during the operating horizon. However, there are instances where, depending upon the operating region, some parts of the dynamic model become irrelevant or redundant because they are not required. The idea is that at any time only those parts of the dynamic model that need to be applied are taken into account removing those parts of the model that do not apply. By doing so, we end up with a more robust dynamic model since parts of the model that are not required are not converged. Moreover, the optimization task of the reduced dynamic model can be improved by applying generalized disjunctive programming solution methods that exploit the logic in these models [16], [29]. It has been a well-known fact that the MINLP optimization of processing systems have had difficulties when those parts of a model that dot not apply also have to be converged [16]. This situation has been partially solved with the development of disjunctive optimization formulations aimed at providing robust optimization formulations [30], [31].

The aim of the present work is to provide an explicit optimization formulation for dealing with processing systems embedded with the dynamic behavior of hybrid systems using a disjunctive optimization formulation to include only those parts of a given dynamic model that are relevant. For this reason we will call to the resulting MIDO problem a Differential-Algebraic Generalized Disjunctive Programming (DAGDP). In particular, our objective is to demonstrate that better and more robust optimization formulations can be obtained when considering only those parts of the model that actually apply. Instead of transforming the DAGDP into a MINLP problem by applying the direct transcription approach, we use a logic outer-approximation algorithm [16] to show the computational advantages of using logic-based methods when addressing the solution of DAGDP problems. Three case studies of different complexity are solved to demonstrate the proposed extension of the algebraic logic outer-approximation algorithm to deal with disjunctive hybrid dynamic systems.

2 Problem statement

Commonly hybrid control problems involve both integer and continuous variables embedded in a dynamic system. We are interested in computing the optimal values of the control actions where binary variables are used to switch among control actions or to model an event where certain actions ought to be taken. In these kind of hybrid control problems the structure of the underlying mathematical model often remains the same (i.e. the mathematical model does not change during all the control problem). However, in the modeling and optimization of dynamic systems there are cases in which only a subset of the equations comprising the full model actually apply. Therefore, it would be desirable to take into account only that part of the model that does not vanish leading to a more robust solution method because potential non-singularities in the optimization model are removed. In this work we are interested in addressing this kind of optimization problems.

The optimization problem to be addressed in the present work can be stated as follows:

"*Given a dynamic system the goal is to compute the control actions that drive the system from an initial to a final state by optimizing a given objective function and switching among parts of the model that apply during such a transition*".

For addressing the dynamic optimization of hybrid systems we extend the algebraic generalized disjunctive programming formulation [16] to include differential and algebraic systems. The resulting differential-algebraic generalized disjunctive programming (DAGDP) reads as follows,

$$\text{Min } Z = \psi(z(t), x(t), y(t), u(t), p, t) \qquad \textbf{(DAGDP)}$$

s.t.

$$\frac{dz(t)}{dt} = \zeta(z(t), x(t), y(t), u(t), p, t)$$

$$\xi(z(t), x(t), y(t), u(t), p, t) = 0$$

$$\gamma(z(t), x(t), y(t), u(t), p, t) \leq 0$$

$$\bigvee_{r \in D_k} \begin{bmatrix} Y_{kr}(t) \\ \frac{dx(t)}{dt} = \phi_{kr}(z(t), x(t), y(t), u(t), p, t) \\ \varphi_{kr}(z(t), x(t), y(t), u(t), p, t) = 0 \end{bmatrix} \quad k \in K$$

$$\Omega(Y(t)) = \text{True}$$

$$z^o = z(0)$$

$$x^o = x(0)$$

$$z^L \leq z(t) \leq z^U$$

$$x^L \leq x(t) \leq x^U$$

$$y^L \leq y(t) \leq y^U$$

$$u^L \leq u(t) \leq u^U$$

$$p^L \leq p \leq p^U$$

$$z \in \mathbf{R}^m, x \in \mathbf{R}^n, Y_{kr} \in \{\text{True}, \text{False}\} \quad k \in K, r \in D_k$$

In order to represent a general hybrid logic optimization formulation we have divided the general dynamic algebraic model in two sets of dynamic algebraic submodels. The dynamic algebraic sub model represented by the maps $\zeta(x(t), z(t), y(t), u(t), p, t)$ and $\xi(x(t), z(t), y(t), u(t), p, t)$ stands for the set of differential-algebraic equations that are always enforced. On the other hand, the dynamic algebraic sub model represented by the maps $\phi_{ki}(x(t), z(t), y(t), u(t), p, t)$ and $\varphi_{ki}(x(t), z(t), y(t), u(t), p, t)$ stands for the set of differential-algebraic equations whose inclusion or removal depends on the value of certain logic decisions. Moreover, the algebraic constraints that must be met, irrespective of the logic decisions, are denoted by $\gamma(x(t), y(t), u(t), p, t)$. In addition, $z(t)$ and $x(t)$ are time depend vector of states, $y(t)$ is the set of algebraic variables, $u(t)$ is the control action, p is a vector system of decision variables and t is the time. In general, the DAGDP problem may involve nonlinearities in the objective function $\psi(\cdot)$ and related constraints $\zeta(\cdot), \xi(\cdot), \gamma(\cdot), \phi(\cdot)$ and $\varphi(\cdot)$. The logic decisions in the space of continuous variables is given by a set of disjunctions $k \in K$ which are linked by an XOR operator ($\underline{\vee}$). Each one of the disjunctions features $r \in D_k$ terms. Moreover, each one of the disjunctions features a boolean variable $Y_{kr}(t)$ and a set of constraints representing the differential-algebraic model whose inclusion depends upon meeting some logic decisions. The set of constraints are enforced when the disjunctions become active ($Y_{kr}(t) = True$) and removed for non-active disjunctions ($Y_{kr}(t) = False$). $\Omega(Y(t)) = True$ stands for the logic relationships among the boolean variables (i.e. discrete or logic conditions) stated in the form of propositional logic.

One approach to deal with the solution of the DAGDP formulation is to use the transcription approach [15] to fully transform the set of differential equations into a set of algebraic equations. In this way the DAGDP optimization problem is transformed into a generalized disjunctive programming GDP optimization formulation using the transcription approach [15]. This approach for handling optimal control problems is well established and features some useful properties such as handling unstable open-loop operating points. After carrying out such a transformation the following purely algebraic generalized disjunctive programming formulation is obtained:

$$\text{Min } Z = \psi(z_{ij}, x_{ij}, y_{ij}, u_{ij}, p) \qquad \textbf{(GDP)}$$

s.t.

$$\mathfrak{F}(z_{ij}, x_{ij}, y_{ij}, u_{ij}, p) \leq 0$$

$$\mathfrak{H}(z_{ij}, x_{ij}, y_{ij}, u_{ij}, p) \leq 0$$

$$g(z_{ij}, x_{ij}, y_{ij}, u_{ij}, p) \leq 0$$

$$\underset{r \in D_k}{\underline{\vee}} \begin{bmatrix} Y_{kr}^{ij} \\ f_{kr}(z_{ij}, x_{ij}, y_{ij}, u_{ij}, p) = 0 \\ \phi_{kr}(z_{ij}, x_{ij}, y_{ij}, u_{ij}, p) = 0 \end{bmatrix} \quad k \in K$$

$$\Omega(Y) = \text{True}$$

$$z_{ij} \in \mathbf{R}^m, x_{ij} \in \mathbf{R}^n, Y_{kr} \in \{\text{True}, \text{False}\}, \quad i \in N_e, j \in N_c, k \in K, r \in D_k$$

where N_e is the number of finite elements and N_c the number of internal collocation points used for the proper discretization of the differential-algebraic model. In the GDP formulation the set of constraints given by $g(\cdot) \leq 0$ are equivalent, after the application of the transcription method, to the $\gamma(\cdot) \leq 0$ constraints of the DAGDP formulation (i.e $\gamma(\cdot)$ is mapped into $g(\cdot)$). Similarly, $\zeta(\cdot)$ is mapped into $\mathfrak{F}(\cdot)$, $\xi(\cdot)$ is mapped into $\mathfrak{H}(\cdot)$, $\phi(\cdot)$ is mapped into $f(\cdot)$. Those mapping functions just represent the equivalence between the DAGDP and GDP formulations.

Figure 1 displays the discretization approach using orthogonal collocation on finite elements. Inside each one the S processing stages, the process dynamics behavior is represented by a series of finite elements whose right number depends upon the smoothness of such dynamic response. The size of a given finite element i represents certain length of the independent variable which in our case corresponds to the processing time. Of course, depending upon the application some other options of independent variables can be chosen. Having selected the number of finite elements (N_e), the number of internal collocation points (N_c) must also be specified. In our experience 3 or 4 internal collocation points suffices to approximate even complex dynamic behavior. Although larger values of N_c can in principle be chosen, this may lead to stability worries related to the orthogonal polynomials used for computing the location (roots) of the collocation points.

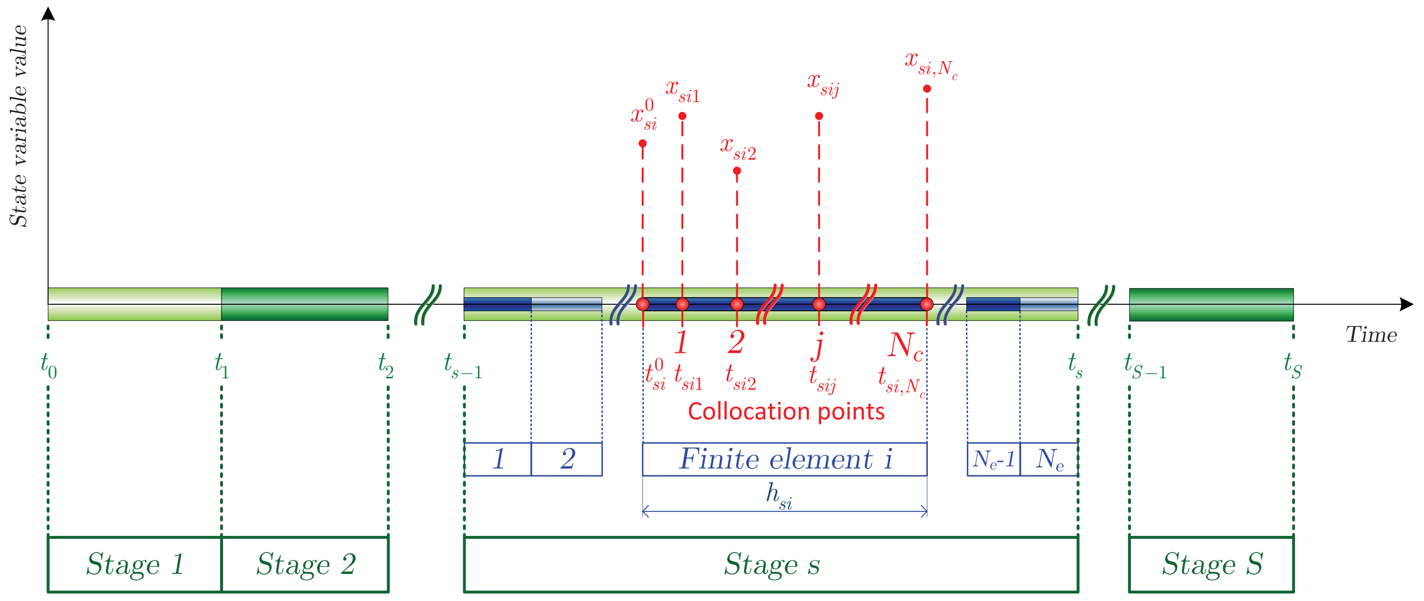


Figure 1: Discretization approach using the transcription approach. Inside each stage s, the time domain is divided into a series of finite elements $i = 1, ..., N_e$. Similarly, inside each finite element, the process behavior is approximated at certain collocation points $j = 1, ..., N_c$.

One of the approaches to handle the solution of the GDP formulation consists in transforming such a formulation into a mixed-integer nonlinear programming problem (MINLP) using either the big-M or the convex-hull formulations [32]. In contrast we will deal directly with problem GDP which has the important feature that only those parts of a given mathematical model that apply need to be enforced when using logic basic outer approximation [16]. The parts of the model that are not active are removed leading to a robust, smaller and better numerically conditioned optimization problem. In contrast, most of the hybrid control problems featuring logic decisions are dealt with by applying either a Big-M or convex-hull formulation to transform the underlying problem into what has been called a logic mixed-integer dynamic optimization (LMIDO) problem whose solution can be sought by either sequential or simultaneous solution methods [15]. However, even when some parts of the LMIDO problem could vanish, the full model is solved. There are at least two difficulties related to the solution of the full model when some parts of such a model are not applicable: a) it increases the solution time, and b) there is the potential presence of non-singularities because the solution method tries to converge redundant equations. Therefore, it would be desirable to remove those parts of a given mathematical model that do not apply. Moreover, the computation of the optimal solution can be greatly improved by applying an optimization algorithm that explicitly exploits the underlying logic structure of the optimization formulation. In conclusion, we claim that the solution of hybrid optimal control problems of a kind of systems can be improved by taking into account only those parts of the model that are relevant and by using a logic-based algorithm that is able to exploit the logic structure of the problem.

3 Logic Outer-Approximation algorithm for dynamic systems

The algebraic logic based outer approximation algorithm (LOA) [16] is an extension of the outer approximation algorithm (OA) [21] to solve problems that are posed as GDP models. It consists in splitting the solution of the original optimization problem into two problems: a master problem represented by a linear approximation of the GDP problem which provides a lower bound of the objective function, and a NLP subproblem whose solution provides an upper bound of the objective function. In this part we extend the algebraic LOA formulation to include hybrid dynamic models in terms of differential-algebraic systems.

- **NLP subproblem**

  Splitting the boolean variables into two sets: $Y_{k\hat{r}}^{ij} = True$ and $Y_{kr}^{ij} = False$, where $\hat{r} \neq r$, the NLP subproblem reads as follows,

$$\text{Min } \ Z = f(z_{ij}, x_{ij}, y_{ij}, u_{ij}, p) \qquad \textbf{(SNLP)}$$

s.t.

$$\mathfrak{F}(z_{ij}, x_{ij}, y_{ij}, u_{ij}, p) \leq 0$$

$$\mathfrak{H}(z_{ij}, x_{ij}, y_{ij}, u_{ij}, p) \leq 0$$

$$g(z_{ij}, x_{ij}, y_{ij}, u_{ij}, p) \leq 0$$

$$f_{kr}(z_{ij}, x_{ij}, y_{ij}, u_{ij}, p) = 0 \ \ for \ \ Y_{kr}^{ij} = True$$

$$\phi_{kr}(z_{ij}, x_{ij}, y_{ij}, u_{ij}, p) = 0 \ \ for \ \ Y_{kr}^{ij} = True$$

$$z^L \leq z_{ij} \leq z^U$$

$$x^L \leq x_{ij} \leq x^U$$

$$z_{ij} \in \mathbf{R}^m, x_{ij} \in \mathbf{R}^n, \quad i \in N_e, j \in N_c, k \in K, r \in D_k$$

It is important to stress that in the above formulation only those constraints whose terms are true within the disjunctions ($Y_{kr}^{ij} = True$) are enforced. Hence, the set of constraints associated to non-active disjunctions are removed. This is the reason why the LOA method leads to a substantial reduced size problem in comparison to using the traditional OA method in MINLP problems. Hence, we take advantage of this feature to improve the optimal solution of hybrid optimal control problems.

- **Master subproblem** To formulate the linear master problem of the GDP formulation we consider a set of linearizations $l = 1, ..., L$ for subsets of the disjunctions $L_{kr} = \{l | Y^l_{kr} = True\}$ giving rise to the following disjunctive OA master problem:

$$\text{Min } \ Z = \alpha \qquad (\textbf{MLGDP})$$

s.t.

$$\alpha \geq f(\omega^l_{ij}) + \nabla f(\omega^l_{ij})^T(\omega_{ij} - \omega^l_{ij}) \leq 0, \quad l = 1, ..., L$$
$$\mathfrak{F}(\omega^l_{ij}) + \nabla \mathfrak{F}(\omega^l_{ij})^T(\omega_{ij} - \omega^l_{ij}) \leq 0$$
$$\mathfrak{H}(\omega^l_{ij}) + \nabla \mathfrak{H}(\omega^l_{ij})^T(\omega_{ij} - \omega^l_{ij}) \leq 0$$
$$g(\omega^l_{ij}) + \nabla g(\omega^l_{ij})^T(\omega_{ij} - \omega^l_{ij}) \leq 0, \quad l = 1, ..., L$$

$$\bigvee_{r \in D_k} \begin{bmatrix} Y^{ij}_{kr} \\ f_{kr}(\omega^l_{ij}) + \nabla f_{kr}(\omega^l_{ij})(\omega_{ij} - \omega^l_{ij}) \leq 0, \ l \in L_{kr} \end{bmatrix} \quad k \in K$$

$$\Omega(Y) = \text{True}$$
$$\omega^L \leq \omega \leq \omega^U$$
$$\alpha \in \mathbf{R}^1, \omega \in \mathbf{R}^{n+m}, Y^{ij}_{kr} \in \{\text{True}, \text{False}\} \quad i \in N_e, j \in N_c, k \in K, r \in D_k$$

For simplicity in the notation of the above formulation the vector ω stands for the two sets of dynamic states: $z(t)$ and $x(t)$ (i.e. $\omega = [z \ x]$). When dealing with algebraic systems Turkay and Grossmann [16] have proposed a way to get the set of values around which the linearizations are carried out. However, for addressing the dynamic optimization of hybrid systems the linearization process is done in a different way. If the dynamic system is initially at steady-state conditions then the linearization can be done around the given steady-state. On the other hand, if the system is not at steady-state conditions or does not have a steady-state solution, the linearization can be done at a given reference dynamic trajectory. In this way the value of the control action $u(t)$ is slightly perturbed from an initial value, then the system response $\omega_{ij}(t)$ is recorded at some proper discrete points and the linearizations are carried out around the $\omega_{ij}(t)$ trajectory.

Overall, the solution process consists of the following steps:

1) Formulate the discrete-continuous dynamic optimization problem as a differential-algebraic generalized disjunctive program (DAGDP).

2) Use the transcription approach to approximate the dynamic behavior in DAGDP. In this step we use orthogonal collocation on finite elements for dealing with the approximation of differential algebraic equations which then give rise to the formulation in GDP.

3) Solve the underlying GDP by using a logic-based outer-approximation solution algorithm which consists of the following steps:

  a) For the first iteration perform an initial linearization of the differential-algebraic system as previously discussed. For subsequent iterations perform the linearization around the predicted values of the manipulated variables $u(t)$ obtained at the previous iteration.

  b) Solve the master problem (MLGDP) to obtain a new set of discrete variables values. The common approach here is to formulate the (MLGDP) as a MILP using either the big-M or convex hull reformulation for the disjunctions [32].

  c) Solve the NLP subproblem (SNLP) for fixed values of the binary variables by eliminating redundant terms and using a discretized differential-algebraic model.

  d) Repeat from step a) until there is no improvement in the solution of the NLP subproblems.

4 Examples

Three stages dynamic model switching

In this section we consider the dynamic optimization of a system featuring two model structures denoted by M_1 and M_2. We will also assume that a production facility is available which consists of 3 manufacturing stages [33]. As seen in Figure 2 either M_1 or M_2 modes can be enforced in each time period. The time horizon of each stage is 1 time unit. The structure of the two dynamic models reads as follows,

Model 1

$$\frac{dx}{dt} = -xe^{x-1} + u \tag{1}$$

Model 2

$$\frac{dx}{dt} = \frac{0.5x^3 + u}{20} \tag{2}$$

in the above model x stands for the state, u is the control action and t is the time. The aim of this problem is to compute which model structure (M_1 or M_2) to apply and also to compute the optimal control actions in each stage to solve the following hybrid optimal control problem:

$$\underset{u}{\text{minimize}}\ V(x,u) = -\int_{t_0}^{3} x^2(t) \tag{3}$$

s.t.

$$\begin{bmatrix} Y_1(t) \\ \frac{dx}{dt} = -xe^{x-1} + u \\ \frac{dx}{dt} = \frac{0.5x^3+u}{20} \\ 0 \le t \le 1 \end{bmatrix} \veebar \begin{bmatrix} Y_2(t) \\ \frac{dx}{dt} = -xe^{x-1} + u \\ \frac{dx}{dt} = \frac{0.5x^3+u}{20} \\ 1 \le t \le 2 \end{bmatrix} \veebar \begin{bmatrix} Y_3(t) \\ \frac{dx}{dt} = -xe^{x-1} + u \\ \frac{dx}{dt} = \frac{0.5x^3+u}{20} \\ 2 \le t \le 3 \end{bmatrix} \tag{4}$$

subject to the control constraint $u \in [-4, 4]$.

As previously stated the Logic-OA approximation algorithm involves the solution of a sequence of optimization problems where each sequence is composed of a MILP Master and NLP problems. The solution sequence stops when the objective functions of both problems are smaller or equal to a target value. We will now describe the structure of both optimization problems.

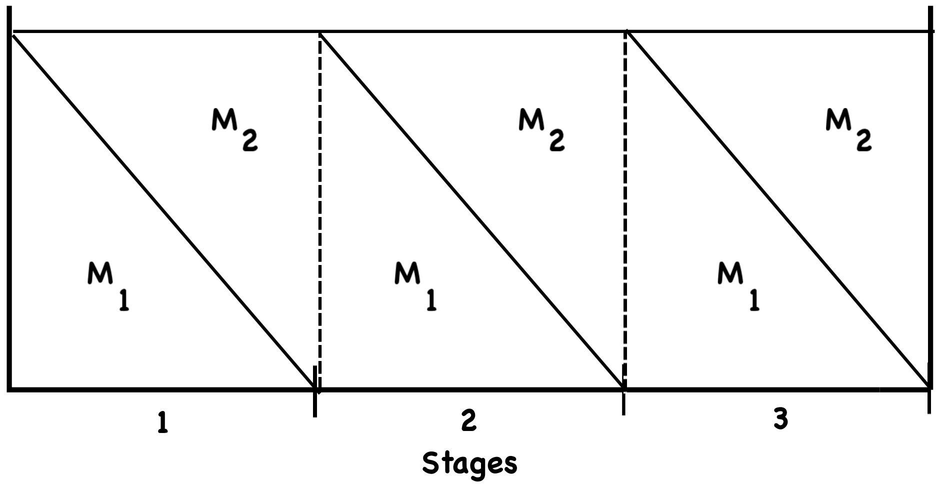


Figure 2: Production system comprised of 3 manufacturing stages. In each stage either the M_1 or M_2 model is enforced. The time horizon of each production stage is 1 time unit.

- **NLP problem**

$$\min_{\mathbf{x},\mathbf{u}} \quad V(\mathbf{x},\mathbf{u}) = \sum_{k}^{N_s} \alpha_k + \alpha_s \sum_i \sum_j S_{ij}^1 + \alpha_s \sum_i \sum_j S_{ij}^2 + \alpha_s \sum_i \sum_j S_{ij}^3 \tag{5a}$$

s.t.

$$\alpha_k = -\sum_i \sum_j h_i \Omega_{j3} (x_{ij}^k)^2 \tag{5b}$$

$$x_{ij}^k = x_{oi}^k + \theta^k h_i \sum_{k'} \Omega_{k'j} \dot{x}_{ik'} \tag{5c}$$

$$x_{oi}^k - \left\{ x_{o,i-1}^k + \theta^k h_{i-1} \sum_j \Omega_{j,3} \dot{x}_{i-1,j} \right\} \leq S_{i1}^k \tag{5d}$$

$$\dot{x}_{ij}^1 + x_{ij}^1 e^{x_{ij}^1 - 1} - u_{ij}^1 \leq S_{ij}^1 \tag{5e}$$

$$\dot{x}_{ij}^2 - 0.025 (x_{ij}^2)^3 - 0.05 u_{ij}^2 \leq S_{ij}^2 \tag{5f}$$

$$\dot{x}_{ij}^3 + x_{ij}^3 e^{x_{ij}^3 - 1} - u_{ij}^3 \leq S_{ij}^3 \tag{5g}$$

$$x_{o,1}^1 = x_{init}^1 \tag{5h}$$

$$x_{o,t'+1}^2 = x_{o,t',N_c}^1 \tag{5i}$$

$$x_{o,t''+1}^3 = x_{o,t'',N_c}^2 \tag{5j}$$

In this formulation equation 5a stands for the definition of the nonlinear objective function. The first term represents the quadrature procedure for evaluating the integral term in equation 3 (which is actually calculated in equation 5b), whereas the remaining three terms penalize the value of the slack variables S_{ij}^k by using a scalar weighting function α_s. Equations 5c-5g represent the discretization of the underlying dynamic system using the transcription approach. The subscripts i and j stand for the number of finite elements and number of internal collocation points, respectively, whereas k is the number of manufacturing stage. Also in this formulation, h is the size of the finite elements, θ^k is the transition time at stage k and Ω is the collocation matrix, $\hat{x}_{ij}^k$ stands for the value of the states trajectory, u_{ij} are the control actions and $\dot{x}_{ij}^k$ is the first derivative of the state variable. Finally, equations 5h-5j represent the state value when switching between the different production periods, where N_c stands for the number of internal collocation points. In this set of equations $t^{'}$ and $t^{''}$ stand for the finite element at which transitions between the manufacturing stages take place and the superscript *init* denotes an initial value.

- **MILP Master problem (Big-M)**

$$\min_{\mathbf{x},\mathbf{u}} \quad V(\mathbf{x},\mathbf{u}) = \sum_{k}^{N_s} \alpha_k + \alpha_s \sum_{i}^{N_e}\sum_{j}^{N_c} S_{ij}^1 + \alpha_s \sum_{i}^{N_e}\sum_{j}^{N_c} S_{ij}^2 + \alpha_s \sum_{i}^{N_e}\sum_{j}^{N_c} S_{ij}^3 \tag{6a}$$

s.t.

$$\alpha_k \geq \sum_{i}^{N_e}\sum_{j}^{N_c} y_{i,j}^k h_i \Omega_{j3} (\hat{x}_{ij}^k)^2 - 2\sum_{i}^{N_e}\sum_{j}^{N_c} h_i \Omega_{j3} \hat{x}_{ij}^k x_{ij}^k \tag{6b}$$

$$x_{ij}^k = x_{oi}^k + \theta^k h_i \sum_{k'}^{N_c} \Omega_{k'j} \dot{x}_{ik'}^k \tag{6c}$$

$$x_{oi}^k - \left\{ x_{o,i-1}^k + \theta^k h_{i-1} \sum_{j}^{N_c} \Omega_{j3} \dot{x}_{i-1,j}^k \right\} \leq S_{i1}^k \tag{6d}$$

$$\dot{x}_{ij}^1 + \left\{ \hat{x}_{ij}^1 e^{\hat{x}_{ij}^1 - 1} + e^{\hat{x}_{ij}^1 - 1} \right\} x_{ij}^1 - u_{ij}^1 + \left\{ \hat{x}_{ij}^1 e^{\hat{x}_{ij}^1 - 1} - (\hat{x}_{ij}^1 e^{\hat{x}_{ij}^1 - 1} + e^{\hat{x}_{ij}^1 - 1}) \hat{x}_{ij}^1 \right\} y_{ij}^1 \leq S_{ij}^1 \tag{6e}$$

$$\dot{x}_{ij}^2 - 0.075(\hat{x}_{ij}^2)^2 x_{ij}^2 - 0.05 u_{ij}^2 + 0.05(\hat{x}_{ij}^2)^3 y_{ij}^2 \leq S_{ij}^2 \tag{6f}$$

$$\dot{x}_{ij}^3 + \left\{ \hat{x}_{ij}^3 e^{\hat{x}_{ij}^3 - 1} + e^{\hat{x}_{ij}^3 - 1} \right\} x_{ij}^3 - u_{ij}^3 + \left\{ \hat{x}_{ij}^3 e^{\hat{x}_{ij}^3 - 1} - (\hat{x}_{ij}^3 e^{\hat{x}_{ij}^3 - 1} + e^{\hat{x}_{ij}^3 - 1}) \hat{x}_{ij}^3 \right\} y_{ij}^3 \leq S_{ij}^3 \tag{6g}$$

$$-u_b y_{ij}^k \leq x_{ij}^k \leq u_b y_{ij}^k \tag{6h}$$

$$-u_b y_{ij}^k \leq u_{ij}^k \leq u_b y_{ij}^k \tag{6i}$$

$$-u_b y_{ij}^k \leq \dot{x}_{ij}^k \leq u_b y_{ij}^k \tag{6j}$$

$$-u_b y_{i1}^k \leq x_{oi}^k \leq u_b y_{i1}^k \tag{6k}$$

$$-u_b y_{ij}^k \leq S_{ij}^k \leq u_b y_{ij}^k \tag{6l}$$

$$\theta_{ij} - \theta_s^1 \leq M(1 - y_{ij}^1) \tag{6m}$$

$$\theta_s^1 - \theta_{ij} \leq M y_{ij}^1 \tag{6n}$$

$$\theta_{ij} - \theta_s^2 \leq M y_{ij}^3 \tag{6o}$$

$$\theta_s^2 - \theta_{ij} \leq M(1 - y_{ij}^3) \tag{6p}$$

$$\sum_{k}^{N_s} y_{ij}^k = 1 \tag{6q}$$

$$x_{o,1}^1 = x_{init}^1 \tag{6r}$$

$$x_{o,t'+1}^2 = x_{o,t',N_c}^1 \tag{6s}$$

$$x_{o,t''+1}^3 = x_{o,t'',N_c}^2 \tag{6t}$$

Equation 6a is the definition of the objective function. It comprises two terms: the first term is given by the transition objective as defined in Equation 3 for each one of the manufacturing stages. The remaining terms contain the slack variables for each state along the transition trajectory (S_{ij}^k) times a weighting factor α_s that was taken equal to 1000. N_s, and N_e are the number of manufacturing stages and the number of finite elements, respectively. Slack variables are used to meet linearizations of the nonlinear terms embedded in the mathematical model of the underlying system. Equation 6b is the approximated numerical quadrature of the V linearized objective function given in Equation 3 for each one of the manufacturing stages. Equations 6c-6g are the discretization of the underlying system using the transcription approach. In particular Equations 6e-6g are the linear equivalents of the nonlinear dynamic mathematical models for each one of the stages as represented by Equations 1-2. As previously mentioned, in this set of equations $\hat{x}_{ij}^k$ stands for the value of the states trajectory along which the linearization was done, whereas y_{ij}^k is a binary variable to enforce or to remove a given stage and its associated mathematical model. Notice that u_{ij}^k stands for the control action at each stage and along the transition trajectory and $\dot{x}_{ij}^k$ is the first derivative of the state variable. Constraints 6h- 6l are used to set to zero a given variable when this is not used or to locate its value between given bounds (denoted by u_b). Equations 6m-6q are used to switch between the different processing stages and to assign proper values of the binary variables. In this set of equations θ_{ij} is the discretized time, θ_s^1 and θ_s^2 are the desired switching times among production periods and M is an enough large constant. Finally, Equations 6r-6t represent the initial values of the states at the beginning of each manufacturing period.

As can be seen from Table 1, using 30 finite elements and three internal collocation points inside each manufacturing stage, the proposed algorithm is able to find the optimal solution in just a single iteration. Of course, the quality of the initial points around which the linearization of the model is done plays an important role in the convergence of the method. To get the initial points for the linearization of the underlying system open-loop dynamic tests were carried out using different values of the control variable until enough good initial values were found that enabled to get the optimal solution. After some trials we found that the smallest CPU time was obtained when the value of

Iteration	Z_{MILP}	Z_{NLP}
1	-10.1061	-10.1061

Table 1: Summary of major iterations for the proposed logic OA in Example 1.

the control action $u(t)$ was slightly perturbed from an initial value $u(0) = 4$. Commonly, for systems featuring steady-state conditions the linearization process is performed around the nominal steady-state. For systems with no steady-state conditions, linearization around a given system trajectory is performed. As previously stated, system trajectories can be easily obtained running open-loop tests deploying different values of the control action. In Figure 3 the optimal control action and the system response results are displayed for each one of the processing stages. Problems statistics are as follows: (a) for the MILP master problem the number of equations is 3392, the number of integer variables is 270, and the CPU time is 0.1 s, (b) for the NLP problem the number of equations is 2402 and the CPU time is 0.01 s. In both cases the number of continuous variables is 1444. It must be stressed that the large number of binary variables is due to the fact that one binary variable was defined for each collocation point inside each one of the finite elements.

Hybrid Operation of an Isothermal Semibatch Reactor

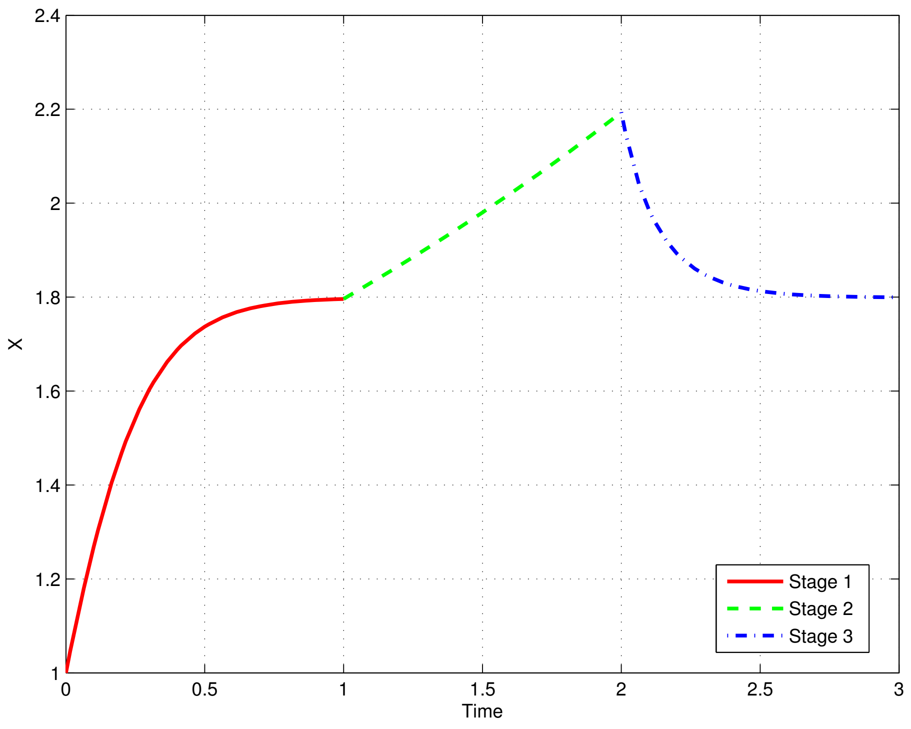
The following set of consecutive chemical reactions take place in an isothermal well mixed semi-batch reactor for manufacturing a set of products denoted by B and C:

$$A + R_1 \xrightarrow{k_1} B \tag{7}$$

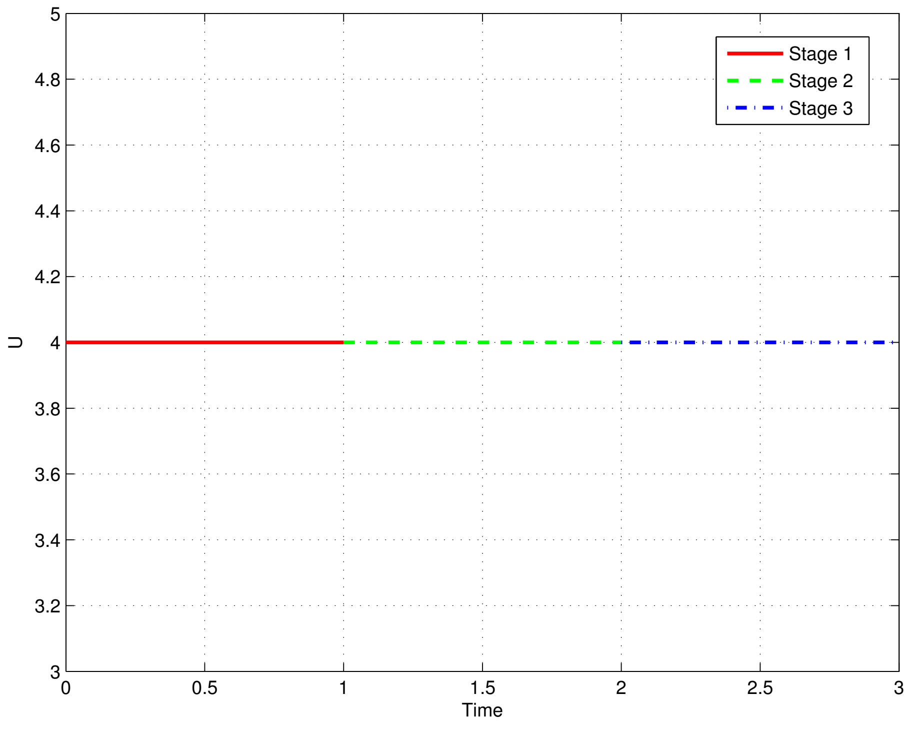
$$B + R_2 \xrightarrow{k_2} C \tag{8}$$

Starting from an initial charge of reactant A the aim of the reaction system consists in maximizing the production of products B and C while minimizing the amount of reactants R_1, R_2 at the end of their respective manufacturing stages (see Figure 4). For achieving this goal the following objective function is used,

$$max \;\; \Omega = \left\{ N_B^1(t_s) - N_A(t_s) - N_{R1}(t_s) \right\} + \left\{ N_C(t_f) - N_B^2(t_f) - N_{R2}(t_f) \right\} \tag{9}$$



(a)



(b)

Figure 3: (a) Optimal system response and (b) control actions for example 1.

where t_s stands for the switching time between the two manufacturing stages, t_f is the final processing time at the second stage and the super index 1 and 2 stands for the number of the production stage. The feed stream flow rates of reactants R_1 and R_2 are the manipulated variables. The reaction system goes trough two reaction stages. In the first reaction stage only reactant R_1 is continuously fed, whereas in the second reaction stage, only reactant R_2 is continuously fed. The switching time t_s is fixed.

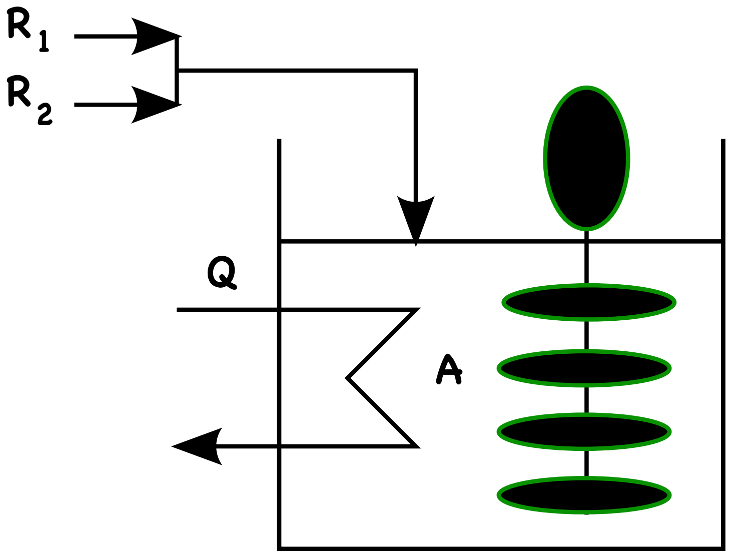


Figure 4: Isothermal semibatch reactor featuring two reaction states. In the first reaction stage only the reaction $A + R_1 \rightarrow B$ takes places, whereas in the second reaction stage the reaction $B + R_2 \rightarrow C$ takes place.

The dynamic model of the system reads as follows,

$$\frac{dN_A}{dt} = -k_1 C_A C_{R_1} V \tag{10}$$

$$\frac{dN_{R_1}}{dt} = F_{R_1} - k_1 C_A C_{R_1} V \tag{11}$$

$$\frac{dN_B}{dt} = k_1 C_A C_{R_1} V - k_2 C_B C_{R_2} V \tag{12}$$

$$\frac{dN_{R_2}}{dt} = F_{R_2} - k_2 C_B C_{R_2} V \tag{13}$$

$$\frac{dN_C}{dt} = k_2 C_B C_{R_2} V \tag{14}$$

$$\frac{dV}{dt} = \frac{F_{R_1}}{C^o_{R_1}} + \frac{F_{R_2}}{C^o_{R_2}} \tag{15}$$

where N stands for the number of moles, C is the concentration, F is the molar flow rate, V is the reactor volume, k_1 and k_2 are the rate constants of the first and second reactions, respectively. The subindex A, B, C, R_1 and R_2 stands for the set of reactant and products. Finally, the superscript " ^{o} " stands for processing conditions related to the two feed streams of reactants R_1 and R_2. The numerical value of the reactor parameters are shown in Table 2, whereas the initial conditions are: $N_A(0) = 8000$, $N_i(0) = 0$ $(i = B, C, R_1, R_2)$ and $V(0) = 1000$ L.

The DAGDP formulation reads as follows,

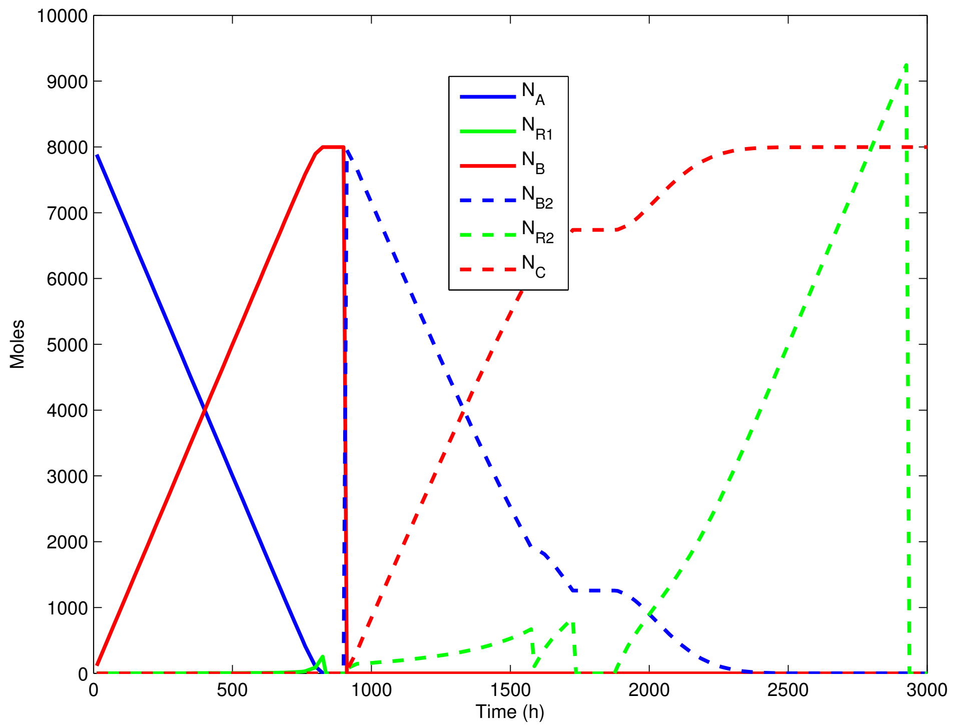
$$\underset{F_{R1}(t),F_{R2}(t)}{\text{maximize}} \ \Omega = \left\{N_B^1(t_s) - N_A(t_s) - N_{R1}(t_s)\right\} + \left\{N_C(t_f) - N_B^2(t_f) - N_{R2}(t_f)\right\} \tag{16}$$

s.t.

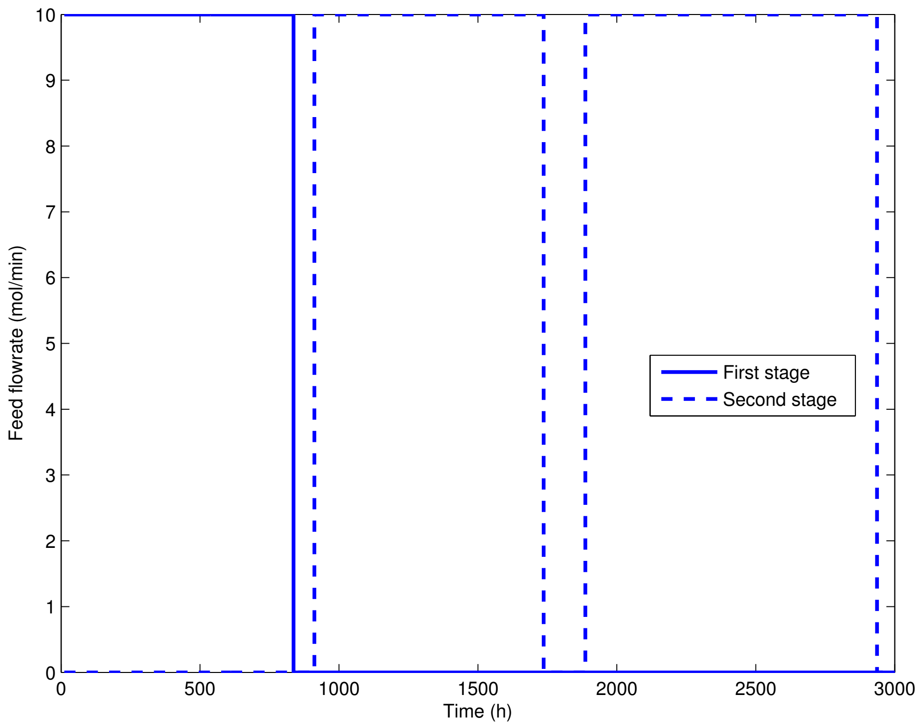
$$\begin{bmatrix} Y_1(t) \\ \frac{dN_A}{dt} = -k_1 C_A C_{R_1} V \\ \frac{dN_{R_1}}{dt} = F_{R_1} - k_1 C_A C_{R_1} V \\ \frac{dN_B}{dt} = k_1 C_A C_{R_1} V - k_2 C_B C_{R_2} V \\ \frac{dN_{R_2}}{dt} = F_{R_2} - k_2 C_B C_{R_2} V \\ \frac{dN_C}{dt} = k_2 C_B C_{R_2} V \\ \frac{dV}{dt} = \frac{F_{R_1}}{C^o_{R_1}} + \frac{F_{R_2}}{C^o_{R_2}} \\ 0 \le t \le t_s \end{bmatrix} \veebar \begin{bmatrix} Y_2(t) \\ \frac{dN_A}{dt} = -k_1 C_A C_{R_1} V \\ \frac{dN_{R_1}}{dt} = F_{R_1} - k_1 C_A C_{R_1} V \\ \frac{dN_B}{dt} = k_1 C_A C_{R_1} V - k_2 C_B C_{R_2} V \\ \frac{dN_{R_2}}{dt} = F_{R_2} - k_2 C_B C_{R_2} V \\ \frac{dN_C}{dt} = k_2 C_B C_{R_2} V \\ \frac{dV}{dt} = \frac{F_{R_1}}{C^o_{R_1}} + \frac{F_{R_2}}{C^o_{R_2}} \\ t_s \le t \le t_f \end{bmatrix} \tag{17}$$

Notice that, as was previously discussed, the proposed DAGDP formulation allow us to embed differential-algebraic equations within the disjunctions but also to include other differential-equations that are always enforced. This fact clearly demonstrates the flexibility of the proposed framework for addressing the optimal control of hybrid systems.

Using 20 finite elements and 3 internal collocation points inside each reaction stage to approximate the dynamic behavior of each system state, the hybrid dynamic optimal results are shown in Figure 5. Upper and lower bounds on the flow rate of the R_1 and R_2 reactants were set at 10 and 0 mol/min, respectively. The initial linearization of each one of the reaction stages was carried out around the



(a)



(b)

Figure 5: (a) Optimal system response and (b) control actions for example 2.

Variable	Value	Units
$C^o_{R_1}$	7	mol/L
$C^o_{R_2}$	6	mol/L
k_1	1.5	L/mol-min
k_2	0.02	L/mol-min
t_s	800	min

Table 2: Operating parameters, kinetic and thermodynamic information, t_s stands for the switching time between the first and second reaction stages. The lower and upper bounds of the reactants flow rates were set at 10 and 0 mol/min, respectively.

Iteration	Z_{MILP}	Z_{NLP}
1	1600	1599.2945

Table 3: Summary of major iterations for the proposed logic OA in Example 2.

dynamic trajectory obtained by slightly perturbing the reactant flow rates around their maximum values. Even when the shape of the manipulated variables is not easy to guess the proposed logic outer-approximation algorithm is able to find the optimal solution in just a single step as seen from the results shown in Table 3. The quality of the optimal solution displayed in Figure 5 is good since as seen there the amount of the B and C target products is maximized while the amount of the deployed reactants (A, R_1, R_2) is minimized. Problem statistics are as follows: the number of constraints is 6323, the number of continuous variables is 2334 and the number of discrete variables is 720. The CPU times were less than 1 s for MILP master problem using CPLEX and 3 s for the NLP problem using CONOPT. Once again, since a binary variable was used for each one of the discretization points, this leads to a large number of binary variables.

Six stages dynamic model switching

For dealing with complex decision switching among different models along multistage dynamic environments without feedback among them, the sequential multistage formulation shown in Figure 6 has been proposed [34]. As seen from Figure 6 the idea lies in splitting the processing time into a series of production stages. Within a given stage we need to select only a single dynamic mathematical model (among several candidates) representing the optimal production schedule in that stage.

This decision process is repeated along all the production schedules comprising the overall optimal production sequence.

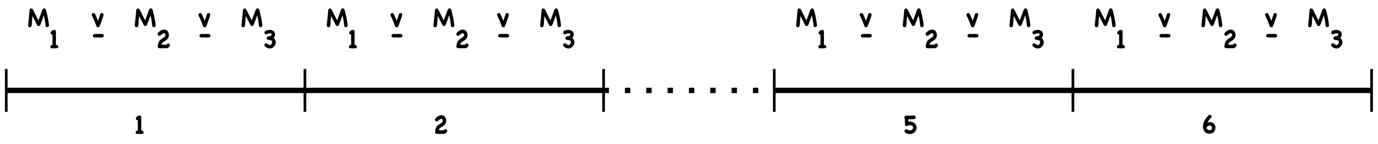


Figure 6: Production system comprised of 6 manufacturing stages. In each stage only one out of three different models is enforced. The time horizon of each production stage is 1 time unit.

This problem is a more complex version of the first case study. It deals with the switching of three different models in a six stages dynamic environment [34]. The dynamic mathematical models read as follows,

Model 1 (M_1)

$$\frac{dx}{dt} = -2tx + u \tag{18}$$

Model 2 (M_2)

$$\frac{dx}{dt} = \frac{x+u}{t+10} \tag{19}$$

Model 3 (M_3)

$$\frac{dx}{dt} = -4xe^{-t} + u \tag{20}$$

in the above set of equations x is the single system state, t is the time and u is the control action. The control actions are bounded in the interval [-4,4]. The goal of this problem is to find the optimal parameter $u \in [-4, 4]$ and the optimal sequence of modes that drive the scalar system state from a given initial to its final state over a fixed time period $t \in [t_0, t_6]$ so as to minimize the following objective function:

$$V(x,u) = 10 \int_{t_0}^{t_6} x^2(t)dt \tag{21}$$

the processing time of each one of the stages is 1 time unit. As done previously in the first example, we provide full details of both the Master MILP and NLP problem formulations.

The problem is formulated as a DAGDP problem as follows:

$$\underset{u}{\text{minimize}} \qquad 10\int_{t_0}^{t_6} x^2(t)dt \tag{22}$$

s.t.

$$\begin{bmatrix} Y_{s1} \\ \\ \frac{dx(t)}{dt} = -2tx(t) + u \end{bmatrix} \veebar \begin{bmatrix} Y_{s2} \\ \\ \frac{dx(t)}{dt} = \frac{x(t)+u}{t+10} \end{bmatrix} \veebar \begin{bmatrix} Y_{s3} \\ \\ \frac{dx(t)}{dt} = -4x(t)e^{-t} + u \end{bmatrix}, \quad s = 1, \ldots, 6$$
$$t \in [t_{s-1}, t_s] \tag{23}$$

$t_0 = 0, t_s = s, \quad s = 1, \ldots, 6$

$x(t_0) = 1$

$p \in [-4, 4], x \in \mathbb{R}, Y_{s1}, Y_{s2}, Y_{s3} \in \{True, False\}, s = 1, \ldots, 6$

where s dentotes the index set of stages. We use a disjunctive multistage model that contains Boolean variables Y_{sm} which are associated to the differential-algebraic constraints of each mode m in process stage s. The above optimization problem can be transformed into a discretized GDP problem by orthogonal discretization. We use a simultaneous method that fully discretizes the DAE system by approximating the control and state variables as piecewise polynomials functions over finite elements [15]. Accordingly, at each collocation point the state variable is represented by:

$$x_{sik} = x_{si}^0 + h_{si}\sum_{j=1}^{K}\Omega_j(\tau_k)\dot{x}_{sij}, \quad s = 1, \ldots, S,\ i = 1, \ldots, I,\ k = 1, \ldots, K \tag{24}$$

where x_{si}^0 is the value of the state variable at the beginning of element i in stage s, $\dot{x}_{sij}$ is the value of its first derivative in element i at the collocation point k in stage s, h_{si} is the length of element i in stage s, and $\Omega_j(\tau_k)$ is the interpolation polynomial of order K for collocation point j, satisfying:

$\Omega_j(0) = 0 \qquad \text{for } j = 1, \ldots, K$

$\Omega_j(\tau_k) = \delta_{jk} \qquad \text{for } j, k = 1, \ldots, K$

where τ_k is non-dimensional time coordinate with the location of the kth collocation point within each element.

We enforce continuity of the state variable across finite element boundaries in each stage by:

$$x_{si}^{0} = x_{s,i-1,K}, \qquad s = 1, \ldots, S,\ i = 2, \ldots, I \tag{25}$$

The multistage model contains additional stage transition conditions, which map the differential state variable values across the stage boundaries. This feature adds flexibility to the model to allow for changing the number of the state variables from one stage to the next. If all model stages contain the same number of differential state variables, as in our example (there is only one state variable), it is normal practice to assume that the values of these variables are continuous across the stage boundary. The physical reasoning for this assumption is that the differential variables represent conserved quantities (such as mass, energy, or momentum) or are directly related to them. In particular, the mapping conditions for our example read as follows:

$$\begin{aligned} x_{s1}^{0} - x_{s-1,I,K} &= 0, \quad s = 2, \ldots, S \\ x_{11}^{0} - x^{initial} &= 0 \end{aligned} \tag{26}$$

These state mapping conditions also include an equality constraint with the initial value for the differential state variable in the first stage x_{11}^{0}, which is equal to given initial value ($x^{initial}$).

For each stage, the collocation method requires the time to be discretized over each finite element at the selected collocation points t_{sik}:

$$t_{sik} = t_{si}^{0} + h_{si}\tau_k, \quad s = 1, \ldots, S,\ i = 1, \ldots, I,\ k = 1, \ldots, K \tag{27}$$

where t_{si}^{0} is the value of the time at the beginning of element i in stage s. Time continuity between stages (Eq. 28) and between elements within a stage (Eq. 29) is also enforced by the following constraints:

$$t_{s,1}^{0} = t_{s-1,I,K}, \quad s = 2, \ldots, S \tag{28}$$

$$t_{si}^{0} = t_{s,i-1,K}, \quad s = 1, \ldots, S,\ i = 2, \ldots, I \tag{29}$$

In our example the initial time $t^{initial}$ and final time t^{final} are given leading to the following constraints:

$$
\begin{aligned}
t_{11}^{0} - t^{initial} &= 0 \\
t^{final} - t_{S,I,K} &= 0
\end{aligned} \tag{30}
$$

Our example has explicit discontinuities, which means that the system switches from one continuous stage to the next at given times (denoted by t_s^{fixed}). Therefore, the switching function for each stage (denote with φ_s) takes a very simple form, which is independent of the state variable and parameter values:

$$
\varphi_s := t_{s1}^{0} - t_s^{fixed} = 0, \quad s = 2, \dots, S \tag{31}
$$

Hence, by a full orthogonal collocation discretization in time domain and using the Eqs.(24)-(31), we convert the time-continuous DAGDP problem into a finite dimensional discretized GDP problem:

$$
\underset{x_{sik}, \dot{x}_{sik}, x_{si}^{0}, h_{si}, t_{sik}, p, Y_{1s}, Y_{2s}, Y_{3s}}{\text{minimize}} \qquad 10 \sum_{s=1}^{S} \sum_{i=1}^{I} \sum_{k=1}^{K} h_{si} \omega_k x_{sik}^2 \tag{DGDP}
$$

s.t.

State discretization, Eq.(24)

State continuity element, Eq.(25)

State mapping, Eq.(26)

Time discretization, Eq.(27)

Time continuity, Eqs.(28,29)

Time boundary conditions, Eq.(30)

Switching functions, Eq.(31)

$$
\begin{bmatrix} Y_{s1} \\ \dot{x}_{sik} = -2t_{sik}x_{sik} + u, \ i = 1, \dots, I, \ k = 1, \dots, K \end{bmatrix}
$$

$$
\veebar \begin{bmatrix} Y_{s2} \\ \dot{x}_{sik} = \frac{x_{sik}+u}{t_{sik}+10}, \ i = 1, \dots, I, \ k = 1, \dots, K \end{bmatrix}
$$

$$
\veebar \begin{bmatrix} Y_{s3} \\ \dot{x}_{sik} = -4x_{sik}e^{-t_{sik}} + u, \ i = 1, \dots, I, \ k = 1, \dots, K \end{bmatrix}
$$

$s = 1, \dots, 6$

$t^{initial} = 0,\ t^{final} = S,\ t_s^{fixed} = s - 1,\ s = 2, \dots, S$

$x^{initial} = 1$

$u \in [-4, 4], x_{sik} \in \mathbb{R}, \dot{x}_{sik} \in \mathbb{R}, t_{sik} \in \mathbb{R}_+, Y_{s1}, Y_{s2}, Y_{s3} \in \{True, False\}$

where ω_k are Radau quadrature weights, $\omega_k = \dot{\Omega_j(\tau_K)}$.

4.1 Logic-based discretized NLP subproblem

We use the Logic-based Outer Approximation algorithm to fully exploit the structure of the GDP representation of our problem. The Logic-Based OA shares the main idea of the traditional OA for MINLP, which is to solve iteratively a master problem given by a linear GDP, leading to a lower bound of the solution, and an NLP subproblem, which provides an upper bound. In our example, for fixed values of the Boolean variables Y_{sm} the corresponding discretized NLP subproblem is as follows:

$$\underset{x_{sik}, \dot{x}_{sik}, x_{si}^0, h_{si}, t_{sik}, u}{\text{minimize}} \qquad Z^l\left(Y_{sm}^l\right) = 10 \sum_{s=1}^{S} \sum_{i=1}^{I} \sum_{k=1}^{K} h \omega_k x_{sik}^2 \tag{SNLP}$$

s.t.

State discretization, Eq.(24)

State continuity element, Eq.(25)

State mapping, Eq.(26)

Time discretization, Eq.(27)

Time continuity, Eqs.(28,29)

Time boundary conditions, Eq.(30)

Switching functions, Eq.(31)

$$\left.\dot{x}_{sik} = -2t_{sik}x_{sik} + u, \quad i = 1, \dots, I,\ k = 1, \dots, K\right\} \text{ for } Y_{s1}^l = True$$
$$\left.\dot{x}_{sik} = \tfrac{x_{sik}+u}{t_{sik}+10}, \quad i = 1, \dots, I,\ k = 1, \dots, K\right\} \text{ for } Y_{s1}^l = True$$
$$\left.\dot{x}_{sik} = -4x_{sik}e^{-t_{sik}} + u, \quad i = 1, \dots, I,\ k = 1, \dots, K\right\} \text{ for } Y_{s3}^l = True$$

$s = 1, \dots, S$

$$t^{initial} = 0,\ t^{final} = S, t_s^{fixed} = s-1,\ s = 2, \dots, S$$
$$x^{initial} = 1$$
$$u \in [-4, 4], x_{sik} \in \mathbb{R}, \dot{x}_{sik} \in \mathbb{R}, t_{sik} \in \mathbb{R}_+$$

It is important to note that only the constraints that belong to the active terms (i.e. associate Boolean variable $Y_{sm}^l = True$) are imposed. This leads to a substantial reduction in the size of the NLP subproblem compared to the direct application of the traditional OA method on the MINLP reformulation. Indeed, the reduction ratio of the NLP subproblem size is equal to the number of alternative modes per stage in the problem (for our example, $M = 3$).

4.2 Logic-based discretized Master problem

We formulate the discretized Master problem with the accumulated linearizations of the nonlinear constraints from previous solutions of each discretized NLP subproblem $l = 1, \dots, L$. This linear approximation yields an underestimation of the objective function and an overestimation of the feasible region, and thus a lower bound Z_{LB} to the solution of the optimization problem.

$$\underset{x_{sik}, \dot{x}_{sik}, x_{si}^0, h_{si}, t_{sik}, u, Y_{sm}}{\text{minimize}} \qquad Z_{LB}^L = \alpha \tag{MLGDP}$$

s.t.

$$\begin{aligned} \alpha \geq f(\mathbf{x}^l, \mathbf{h}^l) &+ \sum_{s=1}^{S}\sum_{i=1}^{I}\sum_{k=1}^{K} \nabla_{x_{sik}} f(\mathbf{x}^l, \mathbf{h}^l)\left(x_{sik} - x_{sik}^l\right) \\ &+ \sum_{s=1}^{S}\sum_{i=1}^{I} \nabla_{h_{si}} f(\mathbf{x}^l, \mathbf{h}^l)\left(h_{si} - h_{si}^l\right), \qquad l = 1, \dots, L \end{aligned} \tag{32}$$

$$\begin{aligned} & t_{sik}^{g,l}\left[\nabla_{x_{sik}} g_k(\theta_{sik}^l)\left(x_{sik} - x_{sik}^l\right) + \nabla_{\dot{x}_{si1}} g_k(\theta_{sik}^l)\left(\dot{x}_{si1} - \dot{x}_{si1}^l\right) + \ldots + \nabla_{\dot{x}_{sij}} g_k(\theta_{sik}^l)\left(\dot{x}_{sij} - \dot{x}_{sij}^l\right)\right. \\ & \left. + \ldots + \nabla_{\dot{x}_{ijK}} g_k(\theta_{sik}^l)\left(\dot{x}_{siK} - \dot{x}_{siK}^l\right) + \nabla_{x_{si}^0} g_k(\theta_{sik}^l)\left(x_{si}^0 - x_{si}^{0,l}\right) + \nabla_{h_{si}} g_k(\theta_{sik}^l)(h_{si} - h_{si}^l)\right] \leq 0 \\ & \qquad s = 1, \dots, S,\ i = 1, \dots, I,\ k = 1, \dots, K,\ l = 1, \dots, L \end{aligned} \tag{33}$$

State continuity element, Eq.(25)

State mapping, Eq.(26)

Time discretization, Eq.(27)

Time continuity, Eqs.(28,29)

Time boundary conditions, Eq.(30)

Switching functions, Eq.(31)

$$\underset{m \in D_s}{\veebar} \begin{bmatrix} Y_{sm} \\ t^{r,l}_{sikm} \left[\nabla_{x_{sik}} r_m(\xi^l_{sik}, t^l_{sik}) \left(x_{sik} - x^l_{sik} \right) + \nabla_{\dot{x}_{sik}} r_m(\xi^l_{sik}, t^l_{sik}) \left(\dot{x}_{sik} - \dot{x}^l_{sik} \right) \right. \\ \left. + \nabla_{t_{sik}} r_m(\xi^l_{sik}, t^l_{sik})(t_{sik} - t^l_{sik}) + \nabla_u r_m(\xi^l_{sik}, t^l_{sik})(u - u^l) \right] \leq 0 \\ i = 1, \ldots, I \quad k = 1, \ldots, K, \quad l \in L_{sm} \end{bmatrix},$$

$$m = 1, \ldots, M, \; s = 1, \ldots, S \tag{34}$$

$$\begin{array}{l} x^{initial} = 1, \quad t^{initial} = 0, \; t^{final} = S, \; t^{fixed}_s = s - 1, \; s = 2, \ldots, S \\ \mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_s, \ldots, \mathbf{x}_S\} \in \mathbb{R}^{S \times I \times K} \\ \mathbf{x}_s = \{x_{s1}, \ldots, x_{si}, \ldots, x_{sI}\} \in \mathbb{R}^{I \times K}, \mathbf{x}_{si} = \{x_{si1}, \ldots, x_{sik}, \ldots, x_{siK}\} \in \mathbb{R}^{S \times I} \\ \theta_{sik} = \{x_{sik}, \dot{\mathbf{x}}_{si}, x^0_{si}, h_{si}\} \in \mathbb{R}^{3+K}, \dot{\mathbf{x}}_{si} = \{\dot{x}_{si1}, \ldots, \dot{x}_{sij}, \ldots, \dot{x}_{siK}\} \in \mathbb{R}^K \\ \xi_{sik} = \{x_{sik}, \dot{x}_{sik}, u\} \in \mathbb{R}^3 \\ \mathbf{h} = \{\mathbf{h}_1, \ldots, \mathbf{h}_s, \ldots, \mathbf{h}_S\} \in \mathbb{R}^{S \times I}, \; \mathbf{h}_s = \{h_{s1}, \ldots, h_{si}, \ldots, h_{sI}\} \in \mathbb{R}^I \\ f : \mathbb{R}^{S \times I \times (K+1)} \to \mathbb{R}, \; g_k : \mathbb{R}^{3+K} \to \mathbb{R}, \; r_m : \mathbb{R}^4 \to \mathbb{R} \\ u \in [-4, 4], x_{sik}, \dot{x}_{sik} \in \mathbb{R}^1, t_{sik} \in \mathbb{R}^+, Y_{sm} \in \{True, False\} \end{array}$$

where t^g and t^r are assigned with either the value 1, 0 or -1, depending the sign of the Lagrange multiplier of the corresponding nonlinear constraint.

Linearizations of the objective function (Eq. 32) and the state discretization constraints Eq. (33) are accumulated in each major iteration L. The constraints of a certain disjunction term (mode m in stage s) are only included in the master problem if the corresponding Boolean variable Y_{sm} is True, whereas linearizations of temporally inactive terms are simply discarded. Formally, for each disjunction term this is expressed by the subset $L^L_{sm} := \left\{ l : Y^l_{sm} = True, \; l = 1, \ldots, L \right\}$. Note that this property constitutes again a major difference to the standard OA method.

4.3 MILP reformulation of the discretized master problem

The master problem of the logic-based OA algorithm can be reformulated as an MILP using either Big-M(BM) or Hull Reformulation (HR) formulations. If we apply the Hull Reformulation, then the disjunctions of the discretized master problem (Eqs. 34) are as follows:

$$
\begin{gathered}
t_{sikm}^{r,l}\left[\nabla_{x_{sik}} r_m\left(\xi_{sik}^l, t_{sik}^l\right) x_{sikm} + \nabla_{\dot{x}_{sik}} r_m\left(\xi_{sik}^l, t_{sik}^l\right) \dot{x}_{sikm}\right. \\
\left. + \nabla_{t_{sik}} r_m\left(\xi_{sik}^l, t_{sik}^l\right) t_{sikm} + \nabla_u r_m\left(\xi_{sik}^l, t_{sik}^l\right) u_m\right] \leq \\
t_{sikm}^{r,l}\left[\nabla_{x_{sik}} r_m\left(\xi_{sik}^l, t_{sik}^l\right) x_{sik}^l + \nabla_{\dot{x}_{sik}} r_m\left(\xi_{sik}^l, t_{sik}^l\right) \dot{x}_{sik}^l + \right. \\
\left. \nabla_{t_{sik}} r_m\left(\xi_{sik}^l, t_{sik}^l\right) t_{sik}^l + \nabla_u r_m\left(\xi_{sik}^l, t_{sik}^l\right) u^l\right] y_{sm} + s_{sikm}^r \\
i = 1, \ldots, I \quad k = 1, \ldots, K, \quad m = 1, \ldots, M, \quad l \in L_{sm}, \quad s = 1, \ldots, S
\end{gathered} \tag{35}
$$

$$
\begin{gathered}
\left.\begin{aligned}
x_{sik} &= \sum_{m=1}^{M} x_{sikm} \\
\dot{x}_{sik} &= \sum_{m=1}^{M} \dot{x}_{sikm} \\
t_{sik} &= \sum_{m=1}^{M} t_{sikm}
\end{aligned}\right\} s = 1, \ldots, S, \ i = 1, \ldots, I, \ k = 1, \ldots, K \\
u = \sum_{m=1}^{M} u_m
\end{gathered} \tag{36}
$$

$$
\begin{aligned}
\left.\begin{aligned}
\underline{x_s} y_{sm} \leq x_{sik}^m \leq \bar{x}_s y_{sm} \\
\underline{\dot{x}_s} y_{sm} \leq \dot{x}_{sik}^m \leq \overline{\dot{x}_s} y_{sm} \\
\underline{t_s} y_{sm} \leq t_{sik}^m \leq \overline{t_s} y_{sm}
\end{aligned}\right\} s &= 1, \ldots, S, \ i = 1, \ldots, I, \ k = 1, \ldots, K, \ m = 1, \ldots, M \\
\underline{u} y_{sm} &\leq u_m \leq \bar{u} y_{sm}, \ s = 1, \ldots, S, \ m = 1, \ldots, M
\end{aligned} \tag{37}
$$

where a variable with an underline and overline denote lower and upper bounds, respectively. Each variable inside the disjunctive terms are disaggregated into M new variables as described in the Eqs. 36. The upper and lower bounds for all the disaggreagated variables and the binary variables y_{sm} are used in Eqs. 37 to force the variables to zero when the mode m in not selected in stage s.

Alternatively the big-M formulation can be applied to the same disjunctive constraints (Eq. 34) as follows:

$$t^{r,l}_{sikm}\left[\nabla_{x_{sik}} r_m(\xi^l_{sik}, t^l_{sik})\left(x_{sik} - x^l_{sik}\right) + \nabla_{\dot{x}_{sik}} r_m(\xi^l_{sik}, t^l_{sik})\left(\dot{x}_{sik} - \dot{x}^l_{sik}\right) + \right.$$
$$\left. + \nabla_{t_{sik}} r_m(\xi^l_{sik}, t^l_{sik})(t_{sik} - t^l_{sik}) + \nabla_u r_m(\xi^l_{sik}, t^l_{sik})(u - u^l)\right] \leq M^{BM}_m(1 - y_{sm}) + s^r_{sik}$$
$$i = 1, \ldots, I,\ k = 1, \ldots, K,\ m = 1, \ldots, M,\ l \in L_{sm},\ s = 1, \ldots, S \tag{38}$$

where M^{BM}_m is a sufficiently large upper bound for the corresponding constraint. For $y_{sm} = 1$, the linear inequality constraint (and thus the disjunctive term) is enforced. Otherwise the associated constrain becomes redundant.

For both reformulations, the exclusive OR logic operator in the disjunction is transformed into the following linear constraint:

$$\sum_{m=1}^{M} y_{sm} = 1,\ s = 1, \ldots, S \tag{39}$$

Furthermore, we add a set of integer cuts to exclude the previous solution for the binary variables:

$$\sum_{(s,m)\in B^l} y_{sm} - \sum_{(s,m)\in N^l} y_{sm} \leq \left|B^l\right| - 1, \qquad l = 1, \ldots, L \tag{40}$$

where B^l is the subset defined for each NLP subproblem l that stores the binary variables y_{sm} with a value of 1, and N^l is the subset that collects the remaining binary variables for that NLP subproblem (i.e., $.B^l = \left\{(s, m) : Y^l_{sm} = True\right\}$ and $N^l = \left\{(s, m) : Y^l_{sm} = False\right\}$).

To avoid infeasible master problems caused by the nonconvexity of the discretized GDP problem, we relax the linear inequality constraints for both reformulations, Eq. (35) and Eq. (38) by introducing positive slack variables s^r_{sikm} and s^r_{sik}, respectively [35]. As the common state constraint (Eq. 24) is also nonlinear, we also add the slack variable s^g_{sik} to the RHS of the linearized state constraint Eq. (33). These slack variables are included in the objective function through a penalty term with weights w^r_m and w^g chosen to be sufficiently large. Accordingly, the objective function of the discretized Master Problem is rewritten as:

$$\underset{x_{sik}, \dot{x}_{sik}, x^0_{si}, h_{si}, t_{sik}, u, y_{sm}}{\text{minimize}} \quad Z^L_{LB} = \alpha + \sum_s \sum_i \sum_k \sum_m w^r_m s^r_{sikm} + \sum_s \sum_i \sum_k w^g s^g_{sik}$$

The optimal results of this case study are shown in Figure 6, where we discretize the time domain using 3 collocation points per finite element and 3 finite elements per stage. The correctness of this discretization is ensured by the equality in the state variable profiles foretold by the orthogonal collocation (square marker in Figure 6) and a commercial solver for ordinary differential equations (ode15s from MATLAB), which can calculate the state varible profile after fixing the sequence of modes according to the optimal values of the binary variables.

Table 4 shows the main results obtained by the logic-based OA for both formulations, BM and HR. The optimum is obtained in eight and twelve major iterations for the BM and HR formulations, respectively. To avoid that the algorithm stops early due to the non-convex constraints, we use a stopping criterion based on the heuristic: stop as two consecutive NLP subproblem worsen. It is worth noting that BM formulation attains the global optimum (we verify that the solution 1.306952 is indeed globally optimal by explicit enumeration of the $3^6 = 729$ structurally different process sequences). Furthermore, BM formulation has better performance than HR in terms of computation time.

To compare the performance of the dynamic extension of the logic-based OA algorithm, we also solved this case study using the MIDO approach for addressing the solution of hybrid optimal control problems, which consists in using an MINLP solver after system discretization and directly applying either the BM or HR formulations to handle embedded logic decisions. As the constraints inside the terms of the disjunctions are nonconvex, we use the BM formulation for the disjunctions of the discretized GDP problem:

$$\underset{x_{sik},\dot{x}_{sik},x^0_{si},h_{si},t_{sik},u,y_{1s},y_{2s},y_{3s}}{\text{minimize}} \quad 10\sum_{s=1}^{S}\sum_{i=1}^{I}\sum_{k=1}^{K} h_{si}\omega_k x_{sik}^2 \qquad \text{(MINLP-BM)}$$

s.t.

State discretization, Eq.(24)

State continuity element, Eq.(25)

State mapping, Eq.(26)

Time discretization, Eq.(27)

Time continuity, Eqs.(28,29)

	GDP to MINLP (MIDO approach)		Logic-based Outer Approximation	
	Big M		Big-M	Hull Reformulation
MINLP Solver	DICOPT	SBB	-	-
Objective value	4.108938	3.703904	**1.306952**	1.307100
Sequence of modes	1-1-1-1-1-1	1-2-2-1-1-1	**3-2-2-1-1-1**	3-2-1-1-1-1
Best Subproblem (major iteration)	-	-	8	12
Time to solve the model (CPU seconds)	2.25	28.66	11.45	19.33

Table 4: Statistics of the solution obtained by the MIDO approach and the logic-based Outer Approximation algorithm.

Time boundary conditions, Eq.(30)

Switching functions, Eq.(31)

$$\left.\begin{array}{r}\dot{x}_{sij} \leq -2t_{sik} + u + M(1-y_{s1})\\ \dot{x}_{sij} \geq -2t_{sik} + u - M(1-y_{s1})\\ \dot{x}_{sik}(t_{sik}+10) \leq x_{sik} + u + M(1-y_{s2})\\ \dot{x}_{sik}(t_{sik}+10) \geq x_{sik} + u - M(1-y_{s2})\\ \dot{x}_{sik} \leq -4x_{sik}e^{-t_{sik}} + u + M(1-y_{s3})\\ \dot{x}_{sik} \geq -4x_{sik}e^{-t_{sik}} + u - M(1-y_{s3})\end{array}\right\},\ i=1,\dots,I,\ k=1,\dots,K,\ s=1,\dots,S$$

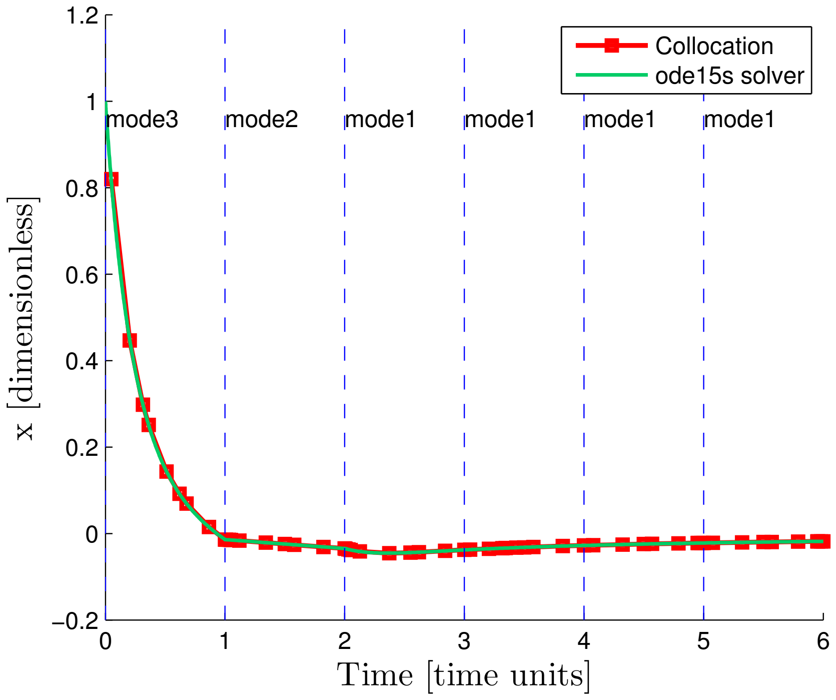
$$\sum_{m=1}^{M} y_{sm} = 1, \quad s=1,\dots,S$$

$$t^{initial}=0,\ t^{final}=S,\ t_s^{fixed}=s-1,\ s=2,\dots,S$$

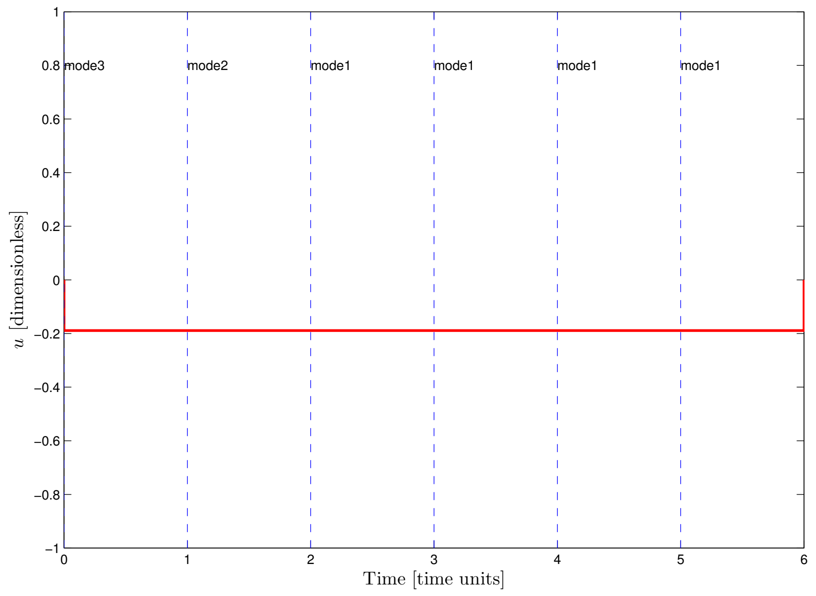
$$x^{initial}=1$$

$$p \in [-4,4], x_{sik} \in \mathbb{R}, \dot{x}_{sik} \in \mathbb{R}, t_{sik} \in \mathbb{R}_+, \quad y_{s1}, y_{s2}, y_{s3} \in \{0,1\}$$

Table 4 compares the performance of two conventional MINLP solvers applied to the direct conversion of the DAGDP into an MINLP (MINLP-BM). Solver DICOPT finds a better solution than solver SBB, and use less computational time. Nevertheless, both solvers are not able to find the global optimum, and SBB has a higher computation time than the worst formulation (i.e. HR) for the Logic-based OA. This is attributed to the fact that the MIDO approach gives rise to a larger nonconvex problem in which the likelihood of getting trapped in suboptimal solutions is greater.



(a)



(b)

Figure 7: Production system comprised of 6 manufacturing stages. In each stage a different model is enforced. The time horizon of each production stage is 1 time unit.

5 Results and Discussion

In this work, 3 case studies of wide complexity related to the hybrid dynamic optimization of processing systems have been solved. These problems feature nonlinear dynamic behavior and two of them have been reported in the open research literature [33], [34] for addressing the numerical solution of hybrid optimal control problems. In all the cases an initial trajectory of the dynamic decision variables must be supplied for linearization purposes. In the case of dynamic continuous systems, such an initial trajectory is easy to obtain since it can be taken as the value of the input variables under steady-state conditions. For noncontinuous systems, as the ones addressed in the present work, the system linearization was performed along an initial guessed trajectory: an initial value of the input variables was fixed and then the dynamic system response was recorded. Such a system response was deployed as the trajectory for linearization purposes. Of course, during the solution procedure instead of guessing the value of the input variables we could just use the values of such variables obtained in a previous iteration.

Regarding the dynamic system behavior during the switching times, special care must be taken to avoid numerical convergence issues due to model switching. To illustrate this point, consider the optimal dynamic response of the first case study shown in Figure 3. We have observed that if the value of the α_s weighting factor used in Eqn 6a is not properly chosen, sometimes it becomes difficult to find an optimal solution because the set of discretized dynamic model equations 6e-6g become infeasible. Anyway, once those tuning parameters are well chosen, obtaining the hybrid optimal solution becomes rather easy.

It must be highlighted that during the past years the field of hybrid optimal control has seen numerous and important contributions [27], [36]. However, some of the solutions approaches require the use of a linear dynamic system. This is normally done by Taylor expansion of the original nonlinear model around a given operating condition. Using this approach has important implications in terms of the ability of solving the underlying optimal control problems since the resulting problems are cast in terms of a mixed-integer linear programming problem for which efficient numerical solution codes are available [37]. Whether or not the nonlinear behavior can be neglected is a matter of discussion, and it depends on the specific analyzed case. It is quite clear that when such a nonlinear behavior cannot be neglected the optimal results obtained from using linear dynamic models will lead to get suboptimal solutions. Therefore, improved optimal solutions can demand the use of nonlinear models and the consideration of nonlinear behavior as a key component when formulating models that better reflect the way a given system behave. Numerical optimization techniques, as the one proposed in the present work, can be helpful to improve the dynamic optimization of hybrid and highly nonlinear systems when the featured nonlinearities cannot be neglected.

6 Conclusions

In this work we have extended the Logic Outer-Approximation algorithm [16] originally proposed for dealing with the optimization of algebraic logic systems to include logic hybrid dynamic system described in terms of differential-algebraic equations featuring switching conditions. Although the dynamic behavior of most of hybrid systems refers to the switching among control actions, in the present work we are more interested in switching some parts of a given dynamic mathematical model that become irrelevant under specific processing conditions. By doing so, the robustness of the logic outer approximation algorithm for addressing the solution of hybrid control problems improves since the redundant parts of the dynamic model are removed, and therefore they are not taken into account during the solution procedure. The quality of the results clearly indicate that converging LMIDO problems could benefit from using the proposed extension of the logic outer-approximation algorithm to deal with logic dynamic hybrid control problems in comparison to the approach commonly used for solving such problems, which consists in the straightforward application of the big-M or convex-hull relation followed by the use of a MINLP algorithm. By taking advantage of the underlying mathematical structure of optimization problems improved optimization solution procedures, as the one described in the present manuscript, can be formulated.

References

[1] R.G.E. Franks. *Modeling and Simulation in Chemical Engineering*. Wiley Interscience, 1972.

[2] W. Bequette. *Process Dynamics*. Prentice-Hall, 1998.

[3] K.J. Astrom and B. Wittenmark. *Computer Controlled Systems*. Prentice-Hall, 1996.

[4] P. Barton. *The Modelling and Simulation of Combined Discrete/Continuous Processes*. PhD thesis, Imperial College of Science, Technology and Medicine, 1992.

[5] Arvind U. Raghunathan, M. Soledad Diaz, and Lorenz T. Biegler. An MPEC Formulation for Dynamic Optimization of Distillation Operations. *Comput. Chem. Eng.*, 28:2037–2052, 2004.

[6] Sebastian Terrazas-Moreno, Antonio Flores-Tlacuahuac, and Ignacio E. Grossmann. A Lagrangean Heuristic Approach for the Simultaneous Cyclic Scheduling and Optimal Control of Multi-Grade Polymerization Reactors. *AIChE J.*, 54(1):163–182, 2008.

[7] A. Kroll W. Marquardt A. Prata, J.Oldenburg. Integrated Scheduling and Dynamic Optimization of Grade Transitions for a Continuous Polymerization Reactor. *Comput. Chem. Eng.*, 32:463–476, 2008.

[8] Jan Oldenburg, Wolfgang Marquardt, Dieter Heinz, and Daniel B. Leineweber. Mixed-Logic Dynamic Optimization Applied to Batch Distilation Process Design. *AIChE J.*, 49(11):2900–2916, 2003.

[9] Antonio Flores-Tlacuahuac and Ignacio E. Grossmann. Simultaneous Cyclic Scheduling and Control of a Multiproduct CSTR. *Ind. Eng. Chem. Res.*, 45(20):6175–6189, 2006.

[10] C. Chatzidoukas, C. Kiparissidis, J.D. Perkins, and Pistikopoulos E.N. Optimal Grade Transition Campaign Scheduling in a Gas Phase Polyolefin FBR Using Mixed Integer Dynamic Optimization. Process System Engineering, pages 744–747. Elsevier, 2003.

[11] A. Flores-Tlacuahuac and Lorenz T. Biegler. Simultaneous Mixed-Integer Dynamic Optimization for Integrated Design and Control. 2007.

[12] C. Chatzidoukas, J.D. Perkins, E.N. Pistikopoulos, and C. Kiparissides. Optimal Grade Transition and Selection of Closed-Loop Controllers in a Gas-Phase Olefin Polymerization Fluidized Bed Reactor. *Chem. Eng. Sci.*, 58:3643–3658, 2003.

[13] V. Bansal, J.D. Perkins, and E.N. Pistikopoulos. A Case Study in Simultaneous Design and Control Using Rigurous, Mixed-Integer Dynamic Optimization Models. *Ind. Eng. Chem. Res.*, 41:760–778, 2002.

[14] V. Sakisliz, J.D. Perkins, and E.N. Pistikopoulos. Recent Advances in Optimization-Based Simultaneous Process and Control Design. *Comput. Chem. Eng.*, 28:2069–2086, 2004.

[15] L.T. Biegler. *Nonlinear Programming: Concepts, Algorithms and Applications to Chemical Engineering*. SIAM, 2010.

[16] Turkay, M. and Grossmann, I.E. Logic-Based MINLP Algorithms for the Optimal Synthesis of Process of Process Networks. *Comput. Chem. Eng.*, 20(8):959–978, 1996.

[17] H.P. Williams. *Model Building in Mathematical Programming*. Wiley, 1999.

[18] L.T. Biegler, I.E. Grossmann, and A.W. Westerberg. *Systematic Methods for Chemical Process Design*. Prentice-Hall, 1997.

[19] L.T. Biegler. An overview of simultaneous strategies for dynamic optimization. *Chemical Engineering and Processing*, 46(11):1043–1053, 2007.

[20] J. Betts. *Practical Methods for Optimal Control Using Nonlinear Programming*. SIAM, 2001.

[21] M.A. Duran and I.E. Grossmann. An outer approximation algorithm for a class of mixed integer nonlinear programs. *Math.Prog.*, 36:307, 1986.

[22] A.M. Geoffrion. Lagrangean Relaxation for Integer Programming. *Mathematical Programming Study*, 2:82–114, 1974.

[23] P. Bonami, L.T. Biegler, A. Conn, G. Cornuejols, I.E. Grossmann, C. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Waechter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 5(2):186–204, 2008.

[24] V. Bansal, V. Sakizlis, R. Ross, J.D. Perkins, and E.N. Pistikopoulos. Towards and efficient numerical procedure for mixed integer optimal control. *Comput. Chem. Eng.*, 21:S457–S462, 1997.

[25] B.T. Baumrucker, J.G. Renfro, and L.T. Biegler. MPEC problem formulations and solution strategies with chemical engineeering applications. *Comput. Chem. Eng.*, 32(12):2903–2913, 2008.

[26] B.T. Baumrucker and L.T. Biegler. MPEC strategies for cost optimization of pipeline operations. *Comput. Chem. Eng.*, 34(6):900–913, 2010.

[27] Alberto Bemporad and Manfred Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, 1999.

[28] B.T. Baumrucker and L.T. Biegler. MPEC strategies for optimization of a class of hybrid dynamic systems. *Journal of Process Control*, 19(8):1248–1256, 2009.

[29] S.B. Lee and I.E. Grossmann. A Global Optimization Algorithm for nonconvex generalized disjunctive programming and applications to process systems . *Comput. Chem. Eng.*, 25:1675–1697, 2001.

[30] Ramesh Raman and Ignacio E. Grossmann. Relation Between MILP Modelling and Logical Inference for Chemical Process Synthesis. *Comput. Chem. Eng.*, 15(2):73–84, 1991.

[31] H. Yeomans and I.E. Grossmann. Optimal Design of Complex Distillation Columns Using Rigurous Tray-by-Tray Disjunctive Programming. *Ind. Eng. Chem. Res.*, 39:4326–4335, 2000.

[32] I.E. Grossmann and J.P. Ruiz. Generalized Disjunctive Programming: A Framework for Formulation and Alternative Algorithms for MINLP Optimization. The IMA Volumes in Mathematics and its Applications 154. Springer, 2011.

[33] P. I. Barton and C.K. Lee. Design of process operations using hybrid dynamic optimization. *Comput. Chem. Eng.*, 28:955–969, 2004.

[34] J. Oldenburg. *Logic-based Modeling and Optimization of Discrete-Continuous Dynamic Systems*. PhD thesis, RWTH Aachen University, 2006.

[35] J. Viswanathan and I.E. Grossmann. A Combined Penalty Function and Outer-Approximation Method for MINLP Optimization. *Comput. Chem. Eng.*, 14(7):769–782, 1990.

[36] M.L. Tyler and M. Morari. Propositional logic in control and monitoring problems. *Automatica*, 35:565–582, 1999.

[37] A. Brooke, D. Kendrick, Meeraus, and R. A. Raman. *GAMS: A User's Guide*. GAMS Development Corporation, 1998, http://www.gams.com.