Model Predictive Control of Hybrid Systems with Applications to Supply Chain Management

Alberto Bemporad, Stefano Di Cairano, and Nicolò Giorgetti

Dipartimento di Ingegneria dell'Informazione,
Università di Siena,
Via Roma 56, 53100 Siena, Italy
{bemporad,dicairano,giorgetti}@dii.unisi.it

Abstract

Hybrid systems are dynamical systems whose behavior is determined by the interaction of continuous and discrete dynamics. Such systems arise in many real contexts, including automotive systems, chemical processes, communication networks, and supply chain management. A supply chain, whose goal is to transform ideas and raw materials into delivered products and services, is an example of a heterogeneous interconnection between continuous dynamics (inventory levels, material flows, etc.) and discrete dynamics (connection graphs, precedences, priorities, etc.). In general, in order to maximize a certain benefit or minimize certain costs, we have to *optimally* control all the heterogeneous components of the hybrid system. *Model predictive control* (MPC) is a well-known technique used in industry to (sub)optimally control dynamical processes, and is usually based on linear models. This paper presents an overview of MPC techniques for hybrid systems. After giving a brief introduction to hybrid system models, model predictive control, and standard computation techniques, the paper summarizes recent results in using symbolic techniques and event-based formulations that exploit the particular structure of the hybrid process to come up with improved numerical computation schemes. The concepts are illustrated through application examples in centralized management of supply chains.

Keywords: hybrid systems, model predictive control, logic-based methods, event-driven approaches, supply chain management.

1 Introduction

Over the last few years we have witnessed a growing interest in the study of dynamical processes of a mixed continuous and discrete nature, denoted as hybrid systems, both in academia and in industry. Hybrid systems are characterized by the interaction of continuous models, describing continuous variables governed by differential or difference

equations, and of discrete models, describing symbolic variables governed by logic rules, switching mechanisms, and other discrete behaviors. Hybrid systems can switch between many operating modes where each mode is governed by its own characteristic continuous dynamical laws. Mode transitions may be endogenous (variables crossing specific thresholds), or exogenous (discrete commands directly given to the system).

The interest in hybrid systems is motivated not only by theoretical challenges, but mainly by their ability to model, analyze and synthesize controllers in a large variety of application areas [1], including manufacturing systems [2, 3].

Most of the control synthesis problems are addressed as optimal control problems. For continuous-time hybrid systems, researchers either studied necessary conditions for a trajectory to be optimal, or focused on the computation of optimal/suboptimal solutions by means of dynamic programming or the maximum principle [4, 5, 6]. The hybrid optimal control problem becomes less complex when the dynamics are expressed in discrete-time, as the main source of complexity becomes the combinatorial (yet finite) number of possible switching sequences. In particular, in [7, 8] the authors have solved optimal control problems for discrete-time hybrid systems by transforming the hybrid model into a set of linear equalities and inequalities involving both real and (0-1) variables, so that the optimal control problem can be solved by a mixed-integer programming (MIP) solver [9, 10, 11].

Coming from a different viewpoint, MIP was used in the context of supply chain optimization in [12], where a static model is proposed to support the elaboration of the networking and resource deployment strategy of an enterprise, therefore not taking into account the dynamics of supply chains. Other supply chain management strategies that use MIP methods are reported in [13, 14, 15].

In general an MIP solver provides the solution after solving a sequence of relaxed standard linear (or quadratic) problems (LP,QP). A potential drawback of MIP is (1) the need for converting the discrete/logic part of the hybrid problem into mixed-integer inequalities, therefore losing most of the original discrete structure, and (2) the fact that its efficiency mainly relies upon the tightness of the continuous LP/QP relaxations.

Such drawbacks are not suffered by techniques for solving constraint satisfaction problems (CSP), i.e., the problem of determining whether a set of constraints over discrete variables can be satisfied. Under the class of CSP solvers we mention constraint logic programming (CLP) [16] and satisfiability (SAT) solvers [17], the latter specialized for the satisfiability of Boolean formulas.

The approach of [18], reviewed in this paper, combines MIP and CSP techniques in a cooperative way. In particular, convex programming (e.g. linear, quadratic, etc. [19]) for optimization over real variables, and SAT solvers for determining the satisfiability of Boolean formulas (or logic constraints) are combined in a single branch and bound solver. The benefits of such a combination for optimal control of hybrid systems are exemplified in the context of supply chain management.

Although mixed integer programming is the common standard for solving optimal control problems of the numerical optimization techniques for hybrid MPC mentioned so far are based on discrete-time models. In general, sampling a continuous-time dynamics introduces errors, that in case of hybrid dynamics might become large, due to discontinuities of the state-update equations. One way of reducing this error is to increase

the sampling rate, but this requires more samples to control the system during a given time interval, and hence the computation complexity to obtain the optimal control profile may grow significantly (it is typically exponential with respect to the number of sampling steps). In this paper we review the approach of [20] where time-driven control is substituted by event-driven control. In contrast with discrete-time methods, in this approach events do not happen when a fixed amount of time (the sampling period) elapses, but rather when certain *events* occur (such as the continuous state overpassing a given threshold). The main advantages of the approach are the minimization of the number of optimization variables required to compute the hybrid MPC control action, and the elimination of modeling errors due to events occurring between two sampling instants.

The paper is organized as follows. An abstract hybrid modeling framework and an example of a supply chain management problem are introduced in Section 2. Section 3 presents model predictive control and the corresponding solution methods are analyzed in Section 4. The event-based modeling and control techniques are reviewed in Section 5. A brief description of a Matlab toolbox developed for modeling and simulating hybrid systems and for designing MPC controllers is presented in Section 6.

2 Hybrid Models

Several modeling formalisms have been developed to describe hybrid systems, among them the class of discrete hybrid automata (DHA) introduced in [8], where examples of real-world applications that can be naturally modeled within the DHA framework are also reported. DHA result from the interconnection of a finite state machine (FSM), which is the discrete dynamics of the hybrid system, with a switched affine system (SAS), which is the continuous dynamics, see Figure 1. The interaction between the two is based on two connecting elements: the event generator (EG) and the mode selector (MS). The EG triggers logic signals according to conditions over continuous state and continuous input signals. These logic events and other exogenous logic inputs affect the logic states of the FSM. The MS combines all the logic variables (states, inputs, and events) to choose the "mode" of the continuous dynamics of the SAS. Continuous dynamics are expressed as linear affine difference equations.

DHA models are a mathematical abstraction of the features provided by other computational oriented and domain specific hybrid frameworks, including mixed logical dynamical (MLD) models [7], piecewise affine (PWA) systems [21], and other models described in [22], where the authors also show that all these modeling frameworks are equivalent, under additional mild assumptions.

DHA are formulated in discrete time, because computation is efficiently tractable for discrete-time models. As anticipated, DHA generalize many computational oriented models for hybrid systems and therefore represent the starting point for solving complex synthesis and analysis problems for hybrid systems. In particular the MLD and PWA frameworks allow one to recast reachability/observability analysis and optimal control as mixed-integer linear or quadratic optimization problems. Reachability analysis algorithms were developed in [23] for MLD and PWA systems. For feedback control, in [7] the authors propose a model predictive control scheme (see Section 3) which is able to stabilize MLD systems on desired reference trajectories while fulfilling operating constraints, and possibly

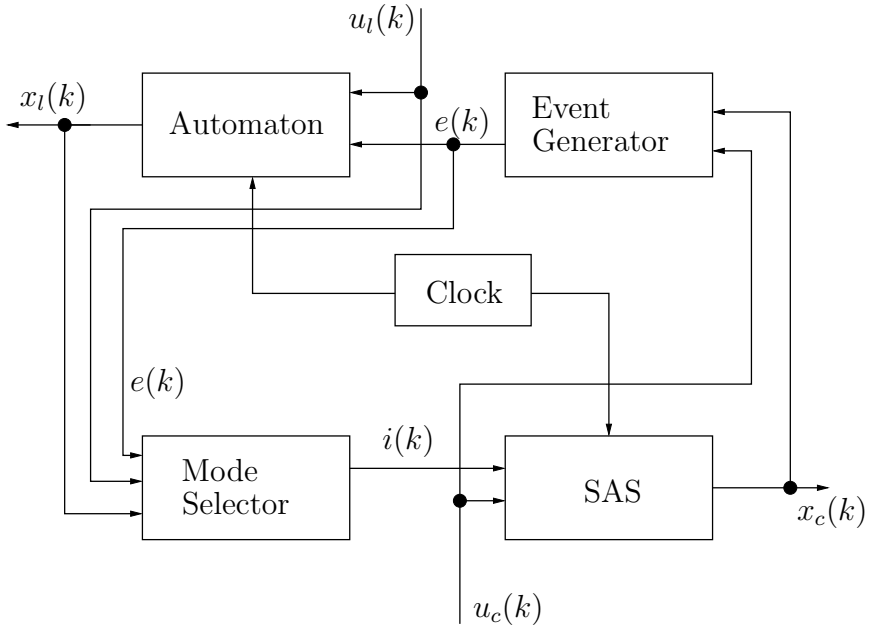


Figure 1: Discrete-time hybrid automaton

take into account previous qualitative knowledge in the form of heuristic rules.

A modeling language to describe DHA models, called HYbrid System DEscription Language (HYSDEL) was proposed in [8] and is currently included in the *Hybrid Toolbox* for Matlab [24] (see Section 6), that can be freely downloaded from `http://www.dii.unisi.it/hybrid/toolbox`.

2.1 Example: Supply Chain Management

The DHA modeling concept is illustrated on a supply chain management (SCM) problem (the reader is referred to [18] for further details).

Supply chain management is the planning and execution of supply chain activities, ensuring a coordinated flow within the enterprise and among integrated companies [25, 26]. These activities include the sourcing of raw materials and parts, manufacturing and assembly, warehousing and inventory tracking, order entry and order management, distribution across all channels and, ultimately, delivery to the customer. The primary objectives of SCM are to reduce supply costs, improve product margins, increase manufacturing throughput, and improve return on investment [14, 15].

In the example of SCM presented in [18] there are F factories, P products can be produced in each factory, and there are $R > P$ retailers in which products are sold. The amount of product p stored in retailer r is updated with a discrete-time continuous dynamics. The quantity of product p delivered to retailer r by all the factories depends by the presence of a connection between a factory f and retailer r. This connection is represented by a logic function. Furthermore, there are a set of rules to satisfy: each retailer cannot store more than a bound Q of products, every factory can produce at most one product at time, at most S_p products can be sold by a retailer, a factory can serve at most MAX_r retailers, and a retailer must be served by at least MIN_f factories.

There are also some logic constraints to fulfill:

Near retailers. Near retailers cannot have the same products. This is to spread the products and to avoid competition among neighboring retailers.

Black-list retailers. A retailer r is in a black list of factory f if the retailer r is not able to sell the products produced in two time units. In this case the factory stops the connection with the retailer for two time units.

Market appreciation. If a product p is appreciated by the market it has to be produced by at least one factory.

The supply chain described above can be viewed as a hybrid system in DHA form. The (discrete-time) continuous dynamics of the retailers represent the SAS part of the DHA. The connection between a factory f and a retailer r described as a logic function represents the mode selector part of the SAS. The logic constraints on the near retailers, black list of retailers and more appreciated products on the market represent the automaton part of the DHA. The goal of SCM is to track as much as possible the demand forecast on a T-step horizon. This goal can be described as the minimization of a linear cost function. In the next section we will show how the posed SCM problem can be solved numerically.

3 Model Predictive Control

For complex constrained multivariable control problems, model predictive control (MPC) has become the accepted standard in the process industries [27, 28]. MPC is based on the so-called *receding horizon* philosophy: at each sampling time, an open-loop optimal control problem starting at the current state, is solved over a finite horizon; however, only the first command input of the optimal sequence is applied to the process, because at the next time step a new optimal control problem based on new measurements is solved over a shifted horizon. In most MPC schemes the optimal solution relies on a linear dynamic model of the process, respects all input and output constraints, and minimizes a performance figure. This is usually expressed as a *quadratic* or a *linear* criterion, so that the resulting optimization problem can be cast as a quadratic program (QP) or linear program (LP), respectively, for which a rich variety of efficient solvers are available.

MPC ideas can be applied to control hybrid DHA models by formulating the finite-time optimal control problem as follows:

$$\min_{\{y(k+1),u(k)\}_{k=0}^{T-1}} \sum_{k=0}^{T-1} \|Q_y(y(k+1) - r_y(k+1))\|_p + \|Q_u(u(k) - r_u(k))\|_p \tag{1a}$$

subject to

DHA dynamics, (1b)

design constraints, (1c)

where T is the control horizon, r_y, r_u are the references of the output and input respectively, Q_y, Q_u are weighting matrices of suitable dimensions, and in (1a) $\|Qx\|_p = x'Qx$

for $p = 2$ and $\|Qx\|_p = \|Qx\|_\infty$ for $p = \infty$. The constraints of the optimization problem (1) are the DHA dynamics and the *design constraints* imposed by the designer to fulfill certain required specifications.

Assuming that the optimal solution $\mathcal{U}_t^* = \{u^*(0), \ldots, u^*(T-1)\}$ of problem (1) exists, according to the receding horizon philosophy, the hybrid MPC algorithm sets the current manipulated input

$$u(t) = u^*(0), \tag{2}$$

disregards the subsequent optimal inputs $u^*(1), \ldots, u^*(T-1)$, and repeats the whole optimization procedure at time $t+1$.

4 MPC Computation

4.1 Mixed-Integer Programming

If we consider the MLD representation for the DHA model the MPC formulation (1) can be rewritten as a *Mixed Integer Quadratic Program* (MIQP) when the squared Euclidean norm $p = 2$ is used, or as a *Mixed Integer Linear Program* (MILP), when $p = \infty$.

Despite the fact that very effective methods exist to compute the (global) optimal solution of both MIQP and MILP problems (see Section 4.1.1 below), in the worst-case the complexity of computing the control action $u(t)$ in (2) on line at each time t depends exponentially on the number of integer variables. In principle, this limits the scope of application of the proposed method to relatively slow systems, since the sampling time should be large enough for real-time implementation to allow the worst-case computation. However, in order to achieve closed-loop stability, it is not required that the evaluated control sequence $\mathcal{U}_t^* = \{u(0), \ldots, u(T-1)\}$ is a global optimum, as long as the value of the objective function is decreased with respect to time $t-1$. Thus the solver can be interrupted at any intermediate step to obtain a suboptimal solution $\mathcal{U}_t$ which satisfies the cost-decreasing condition. For instance, when branch & bound methods are used to solve an MIQP problem, the new control sequence $\mathcal{U}_t$ can be selected as the solution to a QP subproblem which is integer-feasible and ensures a sufficient decrease of the value function. Obviously, suboptimal choices lead to the deterioration of the closed-loop performance.

4.1.1 Mixed Integer Program Solvers

With the exception of particular structures, mixed-integer programming problems involving 0-1 variables are classified as $\mathcal{NP}$-complete, which means that in the worst case, the solution time grows exponentially with the problem size [29]. Despite this combinatorial nature, several packages exist for solving such problems, including [11] for solving MILPs, and [9, 10] for solving MILPs and MIQPs.

A numerical study comparing different MILP/MIQP approaches and solvers is reported in [30].

As described by [31], the branch&bound (B&B) algorithm for MILP/MIQP consists of solving and generating new LP/QP problems in accordance with a tree search, where the nodes of the tree correspond to LP/QP subproblems. Branching is obtained by generating child-nodes from parent-nodes according to branching rules, which can be

based for instance on a-priori specified priorities on integer variables, or on the amount by which the integer constraints are violated. Nodes are labeled as either pending, if the corresponding LP/QP problem has not been solved yet, or fathomed, if the node has already been (implicitly or explicitly) fully explored in depth. The algorithm stops when all nodes have been fathomed. The success of the branch and bound algorithm relies on the fact that whole subtrees can be excluded from further exploration by fathoming the corresponding root nodes. This happens if the corresponding LP/QP subproblem is either infeasible or an integer solution is obtained. In the second case, the corresponding value of the cost function serves as an upper bound on the optimal solution of the MILP/MIQP problem, and is used to further fathoming other nodes having greater optimal value or lower bound.

4.2 Multiparametric Programming

Model predictive control provides a systematic control design procedure for various classes of dynamical systems. However, it might not be possible to solve an on-line optimization problem at each time instant when the system has fast dynamics. By using *multiparametric programming*, in [32] it is shown that the receding horizon control law can be explicitly represented in a piecewise affine form defined over polyhedral partitions. Therefore the design of the controller can be performed in two steps. First, the receding horizon controller is tuned in simulation using MILP/MIQP solvers, until the desired performance is achieved. Then, for implementation purposes, the explicit piecewise affine form of the receding horizon law is computed off-line by using a multiparametric solver. The value of the resulting piecewise affine control function is identical to the one which would be calculated by the MPC controller designed in the first phase, but the on-line complexity is reduced to the simple function evaluation instead of the solution of on-line optimization.

4.3 Logic-based Methods

CSP and optimization are similar enough to make their combination possible, and yet different enough to make it profitable. The two fields evolved more or less independently until a few years ago. However, they have complementary strengths, and the last few years have seen growing efforts to combine them [33, 34, 35]. CSP, for example, offers a more flexible modeling framework than mathematical programming. It not only permits more succinct models, but the models allow one to exploit the structure of the problem itself to direct the search. CSP relies on logic-based methods such as domain reduction and (Boolean) constraint propagation to accelerate the search for the feasible solution.

While CSP methods are superior to MIP approaches for determining if a given problem has a feasible (discrete-valued) solution, the main drawback is their inefficiency for solving optimization, as they do not have the ability of MIP approaches to solve continuous relaxations (e.g., linear programming relaxations) of the problem in order to get upper and lower bounds to the optimum value. For this reason, it seemed extremely interesting to integrate the two approaches into one single cooperative solver. Some efforts have been done in this direction [34, 33, 35, 36], showing that such "hybrid" solution methods have a tremendous performance in solving mathematical programs with continuous (quantita-

tive) and discrete (logical/symbolic) components, compared to MIP or CSP individually. Such successful results have stimulated also the industrial interest of worldwide leaders in commercial software for combinatorial optimization.

In this section we illustrate the SAT-based hybrid algorithm of [18] for merging convex optimization with satisfiability into a single problem-solving technology and its application to optimal control of DHA.

4.3.1 SAT Problems

An instance of a satisfiability (SAT) problem is a Boolean formula that has three components:

- A set of n variables: $x_1, x_2, \ldots, x_n$.
- A set of literals. A literal is a variable ($Q = x$) or a negation of a variable ($Q = \neg x$).
- A set of m distinct clauses: $C_1, C_2, \ldots, C_m$. Each clause consists of only literals combined by just logical "*or*" ($\vee$) connectives.

The goal of the satisfiability problem is to determine whether there exists an assignment of truth values to variables that makes the following Conjunctive Normal Form (*CNF*) formula satisfiable:

$$C_1 \wedge C_2 \wedge \ldots \wedge C_m,$$

where $\wedge$ is the logical "*and*" connective. The CNF form is widely used, since any Boolean formula can be expressed in CNF form [34]. For a survey on SAT problems and related solvers the reader is referred to [17].

SAT solvers are much more efficient than MIP solvers for solving satisfiability problems. To support this statement, in [18] we compared the time spent for solving a feasibility problem with a SAT solver and with an MIP solver on the equivalent mixed-integer formulation, obtained by translating the CNF formula into a set of linear inequalities [8].

4.3.2 Logic-based Solution of MPC

Problem (1) can be reformulated as follows:

$$\min_{\{z,\mu,\nu\}} c(z,\mu) \tag{3a}$$

subject to

$$f(z) \leq 0, \qquad \text{(continuous constraints)} \tag{3b}$$

$$g(z,\mu) \leq 0, \qquad \text{(mixed constraints)} \tag{3c}$$

$$h(\mu,\nu) = \texttt{TRUE}, \qquad \text{(logic constraints)} \tag{3d}$$

where $z \in \mathbb{R}^{n_z}$ are the continuous variables, $\mu \in \{0,1\}^{n_\mu}$ are (0-1) variables which appear both in the mixed and logic constraints, ν are (0-1) variables which appear only in the logic constraints, c, f, g are convex functions, and h is a Boolean function or its CNF representation.

4.3.3 The SAT-based Algorithm

The basic ingredients for an integrated approach are (1) a solver for convex problems obtained from relaxations over continuous variables of mixed integer convex programming problems of the form (3a)-(3b)-(3c), and (2) a SAT solver for testing the satisfiability of Boolean formulas of the form (3d). The relaxed model is used to obtain a solution that satisfies the constraint sets (3b) and (3c) and optimizes the objective function (3a). The optimal solution of the relaxation may fix some of the (0-1) variables to either 0 or 1. If all the (0-1) variables in the relaxed problem have been assigned (0-1) values, the solution of the relaxation is also a feasible solution of the mixed integer problem (3a)-(3b)-(3c). More often, however, some of the (0-1) variables have fractional parts, so that further "branching" and solution of further relaxations is necessary. To accelerate the search of feasible solutions one may use the fixed (0-1) variables to "infer" new information on the other (0-1) variables by solving a SAT problem obtained by constraint (3d). In particular, when an integer solution of μ is found from convex programming, a SAT problem then verifies whether this solution can be completed with an assignment of ν that satisfies (3d).

The basic B&B strategy for solving mixed integer problems can be extended to the present "hybrid" setting where both convex optimization and SAT solvers are used. The B&B method requires the solution of a series of convex subproblems obtained by branching on integer variables. Here, the non-integer variable to branch on is chosen by selecting the variable with the largest fractional part (i.e., the one closest to 0.5), and two new convex subproblems are formed with that variable fixed at 0 and at 1, respectively. In the hybrid algorithm at hand when an integer feasible solution of the relaxed problem is obtained, an additional SAT problem is solved to ensure that the integer solution is feasible for the constraints (3d) and to find an assignment for the other logic variables ν that appear in (3d). If an integer solution exists and has a better value of the objective function the current best integer solution is updated. The value of the objective function for an integer feasible solution of the whole problem is an upper bound (UB) of the objective function, which may be used to rule out branches where the optimum value attained by the relaxation is larger than the current upper bound.

4.3.4 Example: Supply Chain Management (continued)

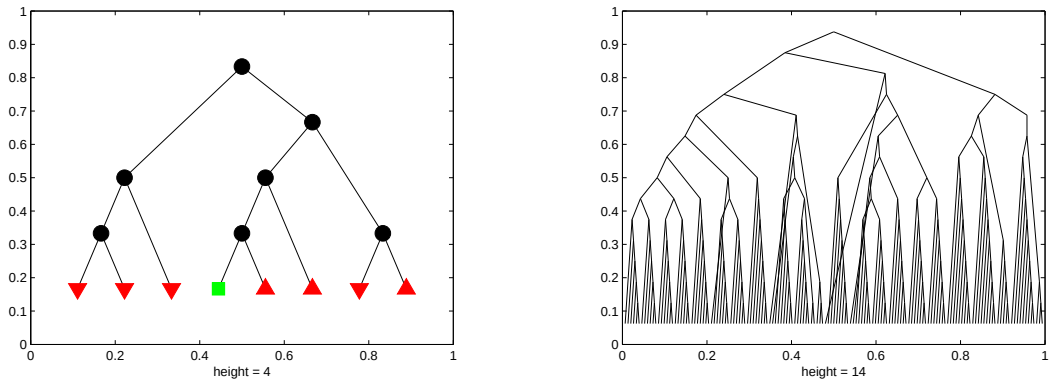
The SCM problem described in Section 2.1 can be solved by using the aforementioned SAT-based approach [18]. Each part of the supply chain problem is managed by either the SAT solver or the convex solver: the linear cost function, the continuous and mixed constraints derived by the DHA model are managed by the LP solver, the logic part is managed by the SAT solver.

The performances of SAT-based B&B is always better than the one of commercial MILP solvers[1], see Table 1. The increase of performance is introduced by the SAT inference, which let the SAT-based B&B algorithm to solve a much smaller number of LP problems than an MILP solver. This result is shown in Figure 2 where the tree gener-

[1]Our simulations were carried out on a Pentium Mobile 1.2 GHz with 640 Mb RAM, by describing and solving the problem within Matlab 7.0 environment and by calling zCHAFF [37] for SAT and CPLEX 9.01 [9] for LP through MEX interfaces available at `http://www.dii.unisi.it/~giorgetti/downloads.html`

T	Bool.	SATbB&B			CPLEX		Naive MILP	
	Vars	(s)	LPs	SATs	(s)	LPs	(s)	LPs
5	210	0.401	8	10	0.61	74	3.45	193
10	420	1.430	10	16	1.802	216	8.91	312
20	840	7.825	33	40	10.301	632	18.160	748
30	1260	11.510	78	98	23.081	692	114.53	1021
40	1680	79.318	264	304	125.930	934	813.23	1404
100	4200	207.840	2306	2409	403.020	3657	> 1200	–

Table 1: Optimal control solution: comparison among SAT-based B&B, CPLEX 9.0 MILP, and a naive MILP implementation



(a) SAT-based algorithm tree (b) Naive MILP algorithm tree

Figure 2: Comparison of the trees generated by the SAT-based and naive MILP algorithms when solving the problem with $T = 10$.

ated by the SAT-based B&B algorithm is compared with the tree generated by a "naive MILP", obtained from the SAT-based B&B by simply disabling the SAT inference.

5 Event-based MPC

Although efficient computation techniques were described in the previous sections, they are based on discrete-time models. Time-discretization might be a critical operation when dealing with hybrid systems, as one has to consider that the system operates in different modes, and thus the sampling period must be chosen with respect to the fastest dynamics. The risk is to oversample the dynamics when the system is operating in the slowest mode, with a consequent unnecessary computation effort due to several binary variables introduced in the optimal control problem. In addition, discrete-time hybrid models assume that discrete-events such as mode switches and discrete-state transitions can occur only at the sampling instants, which might not be a realistic hypothesis. Finally, operating constraints are enforced in discrete-time models only at sampling instants, so that constraint violation between two sampling instants is possible in principle.

To avoid such problems a different approach was proposed in [20]. A continuous-time hybrid system can be modelled as an event-driven system, in which the events are the

T_s	horizon	t_{cpu} (sec.)	t_u
event-driven	6 events	1.23	0
20.00	5 steps	0.45	40.00
16.66	6 steps	1.46	33.32
10.00	10 steps	8.58	20.00
5.00	20 steps	747.14	10.00

Table 2: A numerical comparison of the event-driven and discrete-time techniques

mode switches and the changes of the inputs, provided that the SAS is a set of switched integrators (i.e., $\dot{x}(t) = B_i u(t) + f_i$). While the main reason for focusing the attention to integral dynamics is computational, such class of continuous state dynamics has been widely exploited for modelling and verification of hybrid systems [38], showing to be powerful enough for modelling many practical problems.

The optimal-control problem of the event-driven approach of [20] searches for the sequence of input values and input durations that bring the state to a desired target, while respecting the design constraints and while minimizing different continuous-time criteria, such as the arrival time at the target, the input effort and the maximum displacement from the target.

The optimal control problem can be defined as follows:

$$\min_{\substack{\{x(k+1),\, v(k),\\ t(k+1),\, q(k)\}_{k=0}^{T-1}}} \sum_{k=0}^{T-1} \|Q_x(x(k+1) - r_x(k+1))\|_p + \|Q_v(v(k) - r_v(k))\|_p + \\ + \|R_t(t(k+1) - r_t(k+1))\|_p + \|R_q(q(k) - r_q(k))\|_p \tag{4a}$$

subject to

$$\text{DHA dynamics,} \tag{4b}$$

$$\text{design constraints,} \tag{4c}$$

$$\text{event-generation constraints.} \tag{4d}$$

Problem (4) is similar to the standard discrete-time problem (1). In (4) the state x is weighted, the time instant t at which an event occurs is also weighted and it is an additional state variable, and both the durations q between two events and the input integrals $v = u \cdot q$ are optimization variables. The additional constraints (4d) force the controller to change its action when an event occurs.

In the above event-driven approach the time elapsed between two different control actions can change arbitrarily, since it is a variable in the optimization problem, thus sampling is not uniform as in discrete-time approaches. From a computational point of view this may lead to a reduction of the amount of computation required for the control action. In addition, errors due to unmodeled mode switches between two sampling instants are avoided. Finally, constraints on system states and inputs are enforced along the whole continuous-time trajectory.

The advantages of the event-driven optimal control over its discrete-time counterpart for solving a finite-horizon optimal control problem are highlighted in Table 2, which

refers to the problem of controlling the speed of a train and the closure of a gate in such a way that the train crosses the gate, when this is closed, in minimum time [20]. In column t_{cpu} the time for computing the optimal trajectory is reported, and in column t_u a measure of the risk of being in an unsafe situation (i.e., the time period the train can be in the crossing area while the gate is open) is reported. In order to reduce the risk of unsafety, the discrete-time approach requires the reduction of the sampling period T_s, with a consequent increase of the number of variables and thus of the problem complexity.

In the above event-based formulation, MPC is achieved as follows: given the current state $x(t)$, the corresponding optimal control problem of the form (4) is solved, and the first optimal input value $u(t) = v^*(0)/q^*(0)$ is applied for the whole optimal period duration $q^*(0)$ (which is the time separation from the next predicted event), after which a new event-based optimal control problem is solved.

The adaptation of the event-based approach to the system dynamics is clear. When the system evolves quickly, the next event is predicted to be close in time, so that the new input is computed after a short time period, resulting in a tighter control action. When the system evolves slowly, the predicted time separation with the next event is larger, so that the new input is computed after a longer time period, resulting in a looser control action. Excessively large and small time periods between two events can be enforced through linear upper and lower bounds on durations q.

The event-based techniques can be efficiently applied to the SCM problem, since the continuous dynamics of the supply chain (e.g., inventory levels) are typically integral. Different operating modes can be activated by thresholds conditions on the inventory levels and by logic constraints on near retailers, black lists of retailers, and appreciated products on the market. In this case the event-based approach allows one to avoid the fixed sampling period, thus leading to a simpler and more flexible centralized control/scheduling strategy for the supply chain.

6 Matlab Tools — The Hybrid Toolbox

The Hybrid Toolbox for Matlab/Simulink is a set of tools for modeling, simulating, verifying and designing controllers for hybrid dynamical systems [24].

For hybrid model design and simulation, the toolbox handles mixed logical dynamical (MLD) systems as Matlab objects, obtained from HYSDEL models. MLD objects can be automatically converted into piecewise affine (PWA) objects, and MLD or PWA objects can be simulated in Matlab or in Simulink. Safety properties can be verified through reachability analysis.

Regarding control design, model predictive controllers based on on-line optimization (MILP/MIQP, as described in Section 4.1) can be designed for hybrid systems and for linear systems subject to constraints, with support for both quadratic costs and infinity-norm objectives.

Explicit piecewise affine controllers can be designed via offline optimization (multi-parametric programming, as described in Section 4.2) for linear systems with constraints and for hybrid systems. MPC controllers designed for linear constrained systems with the new Model Predictive Control Toolbox for Matlab [28] can be also converted into piecewise affine form via multiparametric quadratic programming. Explicit controllers

can be easily exported as C-code for direct implementation and rapid prototyping.

The toolbox provides Simulink blocks for controllers based on on-line optimization, explicit piecewise linear controllers, MLD and PWA models, and supports several solvers for linear, quadratic, and mixed-integer optimization.

The toolbox can be freely downloaded from `http://www.dii.unisi.it/hybrid/toolbox`. Functions and methods for implementing the logic-based approach of Section 4.3 and of the event-based approach of Section 5 will be made available by the authors in future releases of the toolbox.

7 Conclusions

In this paper we have reviewed different model predictive control techniques for hybrid dynamical systems and their corresponding numerical computation schemes. Such techniques are particularly suitable for solving supply chain management problems in a centralized manner.

References

[1] P.J. Antsaklis. A brief introduction to the theory and applications of hybrid systems. *Proc. IEEE, Special Issue on Hybrid Systems: Theory and Applications*, 88(7):879–886, jul 2000.

[2] D.L. Pepyne and C.G. Cassandras. Optimal control of hybrid systems in manufacturing. *Proc. IEEE*, 88(7):1108–1123, 2000.

[3] F. Balduzzi, A. Giua, and C. Seatzu. Modelling automated manufacturing systems with hybrid automata. In *Workshop on Formal Methods and Manufacturing*, pages 33–48, Zaragoza, Spain, September 1999.

[4] Xuping Xu and P J Antsaklis. Optimal control of switched systems based on parameterization of the switching instants. *IEEE Transactions on Automatic Control*, 49(1):475–482, Jan 2004.

[5] B. Lincoln and A. Rantzer. Optimizing linear system switching. pages 2063–2068, Orlando, FL, USA, 2001.

[6] M. S. Shaikh and P. E. Caines. On the optimal control of hybrid systems: Optimization of trajectories, switching times, and location schedules. In *Hybrid Systems: Computation and Control*, number 2623 in Lecture Notes in Computer Science, pages 466–481, 2003.

[7] A. Bemporad and M.Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407427, March 1999.

[8] F.D. Torrisi and A. Bemporad. HYSDEL - A tool for generating computational hybrid models. *IEEE Trans. Contr. Systems Technology*, 12(2):235–249, March 2004.

[9] ILOG, Inc. *CPLEX 9.0 User Manual*. Gentilly Cedex, France, 2004.

[10] Dash Optimization Ltd. *Xpress User Manual*. Blisworth, UK, 2005.

[11] A. Makhorin. *GNU Linear Programming Kit (GLPK), Reference manual*, January 2005. `http://www.gnu.org/software/glpk`.

[12] S. Lakhal, A. Martel, O. Kettani, and M. Oral. On the optimization of supply chain networking decisions. 129:259–270, 2001.

[13] E. Melachrinoudis, H. Min, and A. Messac. The relocation of a manufacturing/distribution facility from supply chain perspectives: A physical programming approach. *Multi-Criteria Applications*, 10:15–39, 2000.

[14] R.A. Lancioni. New developments in supply chain management for the millennium. *Industrial Marketing Management*, (29):1–6, 2000.

[15] M. W. Braun, D. Rivera, W.M. Carlyle, and K.G. Kempf. Application of model predictive control to robust management of multiechelon demand networks in semiconductor manufacturing. *SIMULATION*, 79(3):139–156, March 2003.

[16] K. Marriot and P.J. Stuckey. *Programming with constraints: an introduction*. MIT Press, 1998.

[17] J. Gu, P.W. Purdom, J. Franco, and B. Wah. Algorithms for the satisfiability (SAT) problem: A survey. In *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, number 35, pages 19–151. American Mathematical Society, 1997.

[18] A. Bemporad and N. Giorgetti. Logic-based solution methods for optimal control of hybrid systems. *IEEE Transactions on Automatic Control*, 2005. In press.

[19] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[20] A. Bemporad, S. Di Cairano, and J. Júlvez. Event-driven optimal control of integral continuous-time hybrid automata. In *IEEE Conference on Decision and Control*, Seville, Spain, 2005. To Appear: `http://www.dii.unisi.it/ dicairano/papers/cdc05.pdf`.

[21] E.D. Sontag. Nonlinear regulation: The piecewise linear approach. *IEEE Transactions on Automatic Control*, 26(2):346358, April 1981.

[22] W.P.M.H. Heemels, B. De Schutter, and A. Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37(7):10851091, July 2001.

[23] F.D. Torrisi and A. Bemporad. Discrete-time hybrid modeling and verification. In *Proc. 40th IEEE Conf. on Decision and Control*, pages 2899–2904, Orlando, Florida, 2001.

[24] A. Bemporad. *Hybrid Toolbox – User's Guide*. January 2004. `http://www.dii.unisi.it/hybrid/toolbox`.

[25] H. Min and G. Zhou. Supply chain modeling: Past, present and future. *Computer & Chemical Engineering*, (43):231–249, 2002.

[26] E. Perea, I. Grossmann, E. Ydstie, and T. Tahmanebi. Dynamic modeling and classical control theory for supply chain management. *Computer & Chemical Engineering*, (24):1143–1149, 2000.

[27] S. J. Qin and T. A. Badgwell. A survey of industrial model predictive control technology. 11:733–764, 2003.

[28] A. Bemporad, M. Morari, and N. L. Ricker. *Model Predictive Control Toolbox for Matlab – User's Guide*. The Mathworks, Inc., 2004. `http://www.mathworks.com/access/helpdesk/help/toolbox/mpc/`.

[29] R. Raman and I.E. Grossmann. Relation between MILP modeling and logical inference for chemical process synthesis. *Computers Chem. Engng.*, 15(2):73–84, 1991.

[30] Hans Mittelmann. *Benchmarks for Optimization Software*. `http://plato.asu.edu/bench.html`.

[31] R. Fletcher and S. Leyffer. Numerical experience with lower bounds for MIQP branch-and-bound. *SIAM J. Optim.*, 8(2):604–616, May 1998.

[32] F. Borrelli, M. Baotic, A. Bemporad, and M. Morari. Constrained optimal control of discrete-time linear hybrid systems. In *American Control Conference*, 2003.

[33] A. Bockmayr and T. Kasper. Branch and infer: A unifying framework for integer and finite domain constraint programming. *INFORMS Journal on Computing*, 10(3):287–300, Summer 1998.

[34] J. Hooker. *Logic-based methods for Optimization*. Wiley-Interscience Series, 2000.

[35] R. Rodosek, M. Wallace, , and M. Hajian. A new approach to integrating mixed integer programming and constraint logic programming. *Annals of Oper. Res.*, 86:63–87, 1997.

[36] I. Harjunkoski, V. Jain, , and I.E. Grossmann. Hybrid mixed-integer/constraint logic programming strategies for solving scheduling and combinatorial optimization problems. *Comp. Chem. Eng.*, 24:337–343, 2000.

[37] M. Moskewicz, C. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff: Engineering an efficient sat solver. In *39th Design Automation Conference*, June 2001. `http://www.ee.princeton.edu/~chaff/zchaff.php`.

[38] T A Henzinger, P H Ho, and H. Wong-Toi. HyTech: A model checker for hybrid systems. *International Journal on Software Tools for Technology Transfer*, 1(1–2):110–122, 1997.