GENETIC ALGORITHM BASED MODEL PREDICTIVE CONTROL FOR HYBRID SYSTEMS UNDER A MODIFIED MLD FORM

Sorin Olaru¹, Jean Thomas¹, Didier Dumur¹ and Jean Buisson²
¹ Supélec F 91192 Gif sur Yvette cedex Phone : +33 (0)1 69 85 13 75
² Supélec F 35 511 Cesson-Sévigné cedex Phone : +33 (0)2 99 84 45 43

ABSTRACT: The Mixed Logical Dynamical (MLD) formalism has proved to be an efficient modelling framework for hybrid systems described by dynamics, logic and constraints. Furthermore, it allows formulating and solving practical problems such as control and state estimation, using for example predictive strategies. However, its main drawback remains the computation load due to the complexity of the MIQPs to be solved. To overcome this problem, the presented paper proposes an alternative optimization technique based on genetic algorithms. The genetic algorithms provide suboptimal solutions in reasonable time even with a long prediction horizon. For this purpose, a modified MLD form with only discrete control actions is derived that results in a structure adequate for quadratic (0,1)-problems formalism. The transformation of continuous inputs is based on fuzzy intelligent techniques including an adaptation of the variables sets for a better precision. This approach considerably reduces the computation time, thus enabling real time implementation even with small sampling time. This strategy is applied in simulation to the control of a three tanks benchmark.

1. INTRODUCTION

Hybrid systems are an attractive field for engineers, appearing in many control applications in industry, and induce challenging research problems. Hybrid systems include both continuous and discrete variables. The discrete variables are coming from
parts described by logic such as for example on/off switches or valves that interfere with continuous time dynamics. Various approaches have been proposed for modeling hybrid systems (Branicky et al., [5]), like Automata, Petri nets, Linear Complementary (LC) models. It was shown that moving logical relations into linear constraints on integer variables provides a global modeling framework called Mixed Logical Dynamical (MLD) formalism (Bemporad and Morari, [1]). It allows describing large classes of hybrid systems, e.g. piecewise linear systems or systems with mixed continuous/discrete inputs and states.

This formalism can also provide an appropriate framework to formulate and solve practical problems such as state estimation or control. Predictive strategies in that sense offer efficient tools due to their model based formulation, which enable MLD systems to track a desired reference trajectory. The main drawback of this MLD formalism remains the computational burden related to the complexity of the derived MIQPs optimization problems, depending on the number of binary variables and on the prediction horizon. Indeed MIQPs problems are classified as NP-complete, so that in the worst case, the optimization time grows exponentially with the problem size, even if branch and bounds methods may reduce this solution time (Fletcher and Leyffer, [13]). In order to reduce the computational complexity alternative approaches have been developed, e.g. (Stursberg and Panek [21]), (Thomas et al., [22]) where different techniques are used to reduce the number of binary optimization variables.

The main idea of the paper is the elaboration of a new structure, combining predictive control of hybrid systems under a MLD form, with an optimization strategy based on genetic algorithms. This approach provides a suboptimal solution in a reasonable time even for long range prediction horizons. However, a preliminary adaptation of the MLD form is required, in order to consider only discrete control actions, which may improve the solutions of the optimization problem resulting from the genetic algorithm.

In this direction, the reformulation of the MLD form is realized replacing the continuous input variables by discrete control actions. At this step, fuzzy techniques have been introduced to improve the precision of the chosen set of discrete variables. Based on the fuzzy coefficient and on the chosen set of the discretized control variables, a reconstruction of the modified MLD model is developed at each iteration.

Afterwards, a predictive control scheme is applied to this modified MLD structure, and the optimization of the ensuing quadratic (0,1)-problems is performed under a genetic algorithm (GAs) strategy. Even though this technique provides a suboptimal
control sequence, it efficiently reduces the computation time that will now linearly increase with the number of binary optimization variables.

The paper is organized as follows: Section 2 gives a brief description of the MLD form for hybrid systems. General considerations about model predictive control (MPC) and its application to MLD systems are presented in section 3, with a brief description of the classical branch & bound optimization techniques. Section 4 is dedicated to the genetic algorithms and their application to the resolution of the optimization problems linked to the MLD form. Section 5 is the main contribution and examines the features leading to the modified MLD form. Section 6 presents the implementation details towards discrete optimization using GAs. This strategy is applied in Section 7 to the control of a three tanks benchmark. Finally, conclusions are stated in Section 8.

## 2. MLD MODEL

The Mixed Logical Dynamical (MLD) model allows describing various classes of hybrid systems, like linear hybrid systems, constrained linear systems, sequential logical systems (finite state machines, automata), some classes of discrete event systems, and non-linear dynamic systems, where nonlinearities can be expressed through logical combination. The MLD model describes the systems by linear dynamic equations subject to linear inequalities involving both real and integer variables, under the following form (see Bemporad and Morari, [1], for more details):

\[
\begin{align*}
    x(k+1) &= Ax(k) + Bu(k) + B_2\delta(k) + B_zz(k) \\
    y(k) &= Cx(k) + Du(k) + D_2\delta(k) + D_zz(k) \\
    E_2\delta(k) + E_zz(k) &\leq E_1u(k) + E_4x(k) + E_5
\end{align*}
\]

where:

\[
\begin{align*}
    x &= \begin{pmatrix} x_c \\ x_f \end{pmatrix} \in \mathbb{R}^{n_x} \times \{0,1\}^{n_y} , \\
    u &= \begin{pmatrix} u_c \\ u_f \end{pmatrix} \in \mathbb{R}^{m_u} \times \{0,1\}^{m_y} , \\
    y &= \begin{pmatrix} y_c \\ y_f \end{pmatrix} \in \mathbb{R}^{p_x} \times \{0,1\}^{m_y} , \\
    \delta &\in \{0,1\}^{\eta_x} , \\
    z &\in \mathbb{R}^{\tau_z}
\end{align*}
\]

are respectively the vectors of continuous and binary states of the system, continuous and binary (on/off) control inputs, continuous and binary output signals, auxiliary binary and continuous variables. The auxiliary variables \(\delta, z\) are introduced when translating propositional logic into linear inequalities following the scheme in Figure 1. A MLD model equation (1) thus represents logical relations and interaction between continuous and logical variables by mixed integer linear inequalities (Hayes, [15]).
The $A, B_j, C_j, D_j, E_j$ matrices in equation (1) are obtained through the specification language HYSDEL (Hybrid System Description Language) as explained in (Bemporad and Mignone, [4]).

Given the current state $x(k)$ and the input $u(k)$ for a MLD system, the auxiliary variables $\delta(k)$ and $z(k)$ can be defined from the inequalities equation of (1). An important aspect is the "well posed assumption" meaning that there is a unique solution of $\delta$ and $z$ for a given pair $(x, u)$. In other words, once known the state and the inputs, the pair of auxiliary variables can be uniquely defined.

3. MODEL PREDICTIVE CONTROL

Model predictive control (MPC) has proved to efficiently control a wide range of applications in industry. It is capable to control a great variety of processes, including systems with long delay times, non-minimum phase systems, unstable systems, multivariable systems, and constrained systems (Camacho and Bordons, [7]).

3.1 General consideration

The main idea of predictive control is the use of a plant model to predict future system outputs. Based on this prediction, a sequence of future control values is elaborated at each sampling period through an on-line optimization process, which maximizes the tracking performance while satisfying constraints. Only the first value of this optimal sequence is applied to the plant, the whole procedure is repeated again at the next sampling period according to the ‘receding’ horizon strategy (Dumur and Boucher, [12]). The cost function to be minimized is generally a weighted sum of square predicted errors and square future control values, e.g. in Generalized Predictive Control (GPC) (Clarke et al., [8]). For a certain class of hybrid systems, an adaptive GPC has also been proposed (Buisson and Bergomi, [6]).
3.2 Model Predictive Control for MLD systems

For an MLD system equation (1), the following model predictive control problem is considered. Let \( k \) be the current time, \( x(k) \) the current state, \( N+k-1 \) the final time, find \( u_k^{k+N-1} = [u(k), \ldots, u(k+N-1)]^T \) the control sequence that moves the state from \( x(k) \) to the equilibrium state \( x_e \) and minimizes the cost function:

\[
\min_{u_k^{k+N-1}} J(u_k^{k+N-1}, x(k)) = \sum_{j=0}^{N-1} \left[ \|u(j)-u_e\|^2_{Q_1} + \|\delta(j/k) - \delta_e\|^2_{Q_2} + \right. \\
+ \|z(j/k) - z_e\|^2_{Q_3} + \|x(j+1/k) - x_e\|^2_{Q_4} + \|y(j/k) - y_e\|^2_{Q_5} \right]
\]

subject to equation (1), and \( u(j) \) constant for \( j \geq N_u \), where \( N \) is the prediction horizon, \( N_u \) the control horizon, \( \delta_e, z_e \) are the auxiliary variables of the equilibrium point or the reference trajectory value, calculated by solving a MILP problem for the inequality equation. \( x(j+1/k) \approx x(k+j+1, x(k), u_k^{k+j}) \) (in a similar way for the other input and output variables), \( Q_i = Q'_i > 0 \) for \( i = 1,4 \), and \( Q_i = Q'_i \geq 0, i = 2,3,5, \)

It must be noticed that this form equation (2) is slightly different from the one proposed by (Bemporad and Morari, [1]), as the final equality condition \( x(k+N) = x_e \) has been cancelled to be sure that the optimizer can find a solution whatever the prediction horizon may be (even short). Moreover, \( N_u \) has been added to limit the future control values in order to decrease the number of optimization variables.

3.3 Mixed integer quadratic programming (MIQP)

The equality constraints related to the dynamics of the MLD form (1) can be used to replace the implicit variables from the optimization problem of equation (2). This scheme leads to MIQP problems under the form:

\[
F(\chi, x_0) = \min_{\chi} \frac{1}{2} \chi^T H \chi + f^T \chi \text{ subject to } c \chi < b
\]

where the optimization vector is:

\[
\chi = [u(k), \ldots, u(k+N-1), \delta(k), \ldots, \delta(k+N-1), z(k), \ldots, z(k+N-1)]^T
\]
and the number of binary optimization variables is \( L = N_u m_I + N_I \). In the worst case, the optimization time increases exponentially with the number of binary optimization variables (Raman and Grossmann, [20]).

3.4 Branch and Bound (B&B) optimization

As the optimization vector includes both continuous and binary variables, the B&B technique can be used (Bemporad et al., [3]), where the 0-1 combinations are explored through a binary tree, systematically partitioning the feasible region into sub-domains, and generating valid upper and lower bounds at different levels of the binary tree. Several authors agree on the fact that B&B methods are the most successful solving MIQP problems (Fletcher and Leyffer, [13]). Solving MIQP problems with B&B technique relies on the following ideas:

- Relax the constraints of binary variables, i.e. binary variables are allowed to span over the continuous interval \([0,1]\) and solve the problem as a QP problem. The solutions of those relaxed problems, if they exist, represent lower bounds on the optimal solution.

- Choose a binary variable and branch the problem to two new sub-problems by setting the binary variable to 0 or 1.

- If the solution of a relaxed problem satisfies the binary constraints, then it is an upper limit for the global optimal solution.

- If there is no feasible solution at one node, then the B&B technique cuts this branch, as all the sub-problems behind this node will give also a non-feasible solution. The same cutting could also happen if the solution of this node is higher than the upper bound of the optimal solution determined before.

Different strategies for solving the B&B tree exist, e.g. depth first strategy, where the problems are solved according to last-in first-out rules; breadth first strategy where problems at level \( K \) are solved only after solving the problems at level \( K - 1 \); outside first strategy where the problems to be solved start from outside towards inside. An B&B tree example with 3 binary variables according to depth first strategy is shown in Figure 2. Another strategy proposed by (Helmberg and Rendl, [16]) is based on a relaxation approach.
Fig. 2. B&B tree with three binary variables, according to depth first strategy.

4. GENETIC ALGORITHMS

The main drawback of the classical approach with the B&B technique is the exponential complexity with the number of binary optimization variables. In this section, an analysis of the computational difficulties is first completed, the genetic algorithm strategy is then proposed as an alternative technique to solve the optimization problem equation (2) without facing the exponential complexity.

4.1 Computational aspects

Recalling the structure of the optimization problem equations (3), (4), two important remarks should be formulated about the number of binary optimization variables $L = N_u m_f + Nr_l$, which is as mentioned earlier critical for the computational burden.

Remark 1: From the computational point of view, the main disadvantage comes from the fact that a $(\delta, z)$ constrained optimization problem in a $\mathbb{R}^{N_u m_f + Nr_l} \times \{0,1\}^{N_u m_f + Nr_l}$ dimension space has to be solved, instead of an $(\delta, z)$ relaxed optimization problem in a $\mathbb{R}^{N_u m_f} \times \{0,1\}^{N_u m_f}$ dimension space. This latter should have been sufficient as the $Nr_l + Nr_c$ variables are uniquely defined by the system inequalities of (1) due to the “well posed assumption” of the MLD form.

Remark 2: The power of predictive control schemes comes from their capability to provide optimal control sequences based on long range prediction horizons. Nevertheless, as the number of binary optimization variables $L$ is in direct dependence with the length of the receding horizon, the prediction horizon $N$ is forced for computational reasons to a small value, resulting in canceling the main advantage of the model based predictive philosophy.
These two remarks are the main arguments to search other alternative optimization techniques that can replace the classical approach (B&B method), satisfying the following principles:

- The optimization arguments are represented exclusively by the input control actions along the prediction horizon (i.e. the length of the binary part of the optimization vector should be $N_u, m_t$ corresponding to the logical control actions).
- The complexity of the optimization routine should avoid the exponential complexity with respect to the length of the prediction horizon $N$.

In the following, the genetic algorithms will be examined as a stochastic method to solve the optimization problem of MPC for the MLD form. Studies showed that even if the GAs have a limitation from the optimality point of view, offering no guarantee of it, they are suitable for a large class of optimization problems.

### 4.2 Genetic algorithms technique

The original concept of genetic algorithms was first introduced in the 1970s by John Holland of the University of Michigan (Dulay, [11]), (Haupt, [14]) and (Wall, [23]). The basic principle is as follows: a random number generator is used to generate a finite set of random design variable vectors, which are referred to as individuals, genotypes, structures, strings or chromosomes. A set of individuals is called a population. For each individual in the population, the objective function value (or fitness) is calculated. Individuals with a higher fitness are more likely to be chosen for reproduction. Single variables of the chosen individuals are then randomly mutated and crossovers between two parent individuals are performed on parts of the chromosomes. The resulting population generation is then examined as a basis for the next optimization cycle. The algorithm is stopped after a specified number of generations or a certain convergence development, and the individual that has produced the best value of the objective function is considered as output.

In the original context, genetic algorithms worked on individuals consisting of binary design variables. Michalewicz introduced a type of genetic algorithm that works directly on continuous (or floating point) individuals (Michalewicz, [19]). The two basic types only differ in the mutation and crossover operators, while initialization, selection and termination methods remain the same.

*Initialization*: it is completed by generating a random initial population. According to the kind of algorithm (binary or floating point), individuals consist of binary numbers,
or floating point numbers within the range defined for each design variable separately. For each individual, the fitness is calculated using the objective function.

**Selection:** for each generation, individuals are chosen to serve as parents for the next generations. In general, individuals with a higher fitness are more likely to be chosen for reproduction.

**Crossover:** a probability or a number of crossovers is defined a priori. A random pair of individuals is then chosen to perform crossover. Among the various strategies, the most common are:

- Simple crossover: the operator swaps randomly chosen variables from the two parents to produce two children;
- Arithmetic crossover: two chosen variables are interpolated by a random amount, thus moving the two parent values closer to each other. This structure only works with floating point representation. A visualisation of the operator functionality is presented in Figure 3(a);
- Heuristic crossover: using the two chosen variables, a randomly-sized extrapolation is performed in the direction of the variable belonging to the individual possessing the greater fitness, as shown Figure 3(b).

**Mutation:** a single individual is randomly chosen. The value of a single variable is then mutated according to the chosen mutation operator:

- Bit-Flip: the value of a binary variable is inverted from 0 to 1 or 1 to 0, respectively;
- Uniform: a random value within the variable range is chosen as a new value;
- Gaussian: a new value is chosen based on a Gaussian distribution around the parent value. Its functionality is illustrated in Figure 4(a);
- Non-Uniform: the new value is randomly chosen based on a Gaussian distribution around the parent value, its standard deviation decreases with increasing generation numbers. It is illustrated in Figure 4(b).

Bit-Flip mutation only works with binary representation while the three others only accept floating point representation.
**Termination:** in the ideal case, the genetic algorithm would determine when it has found the global optimum, would stop iterations and output the optimum. Unfortunately, the value of the global optimum (the fitness value corresponding to the best set of input variables) is almost never known. Several methods have been developed to trigger termination of the genetic algorithm optimization procedure:

- **Generation number:** execution is stopped after a predefined number of generations. This value should be based on the number of input variables and the input variables’ ranges.
- **Fitness threshold:** a threshold can be selected before execution. If a fitness value is found that exceeds the threshold, execution is terminated. This should be used in combination with the generation number termination method in case the threshold is not reached.
- **Fitness convergence:** when the difference between a filter smoothing over a large number of past generations and a filter smoothing over a small number of past generations falls below a pre-defined percentage, the program is terminated.
- **Population convergence:** if the average population fitness falls below a pre-defined percentage below the best individual, the population is considered converged.

Finally, the tuning of the GA requires the choice of the variable type and related bounds, the crossover and mutation operators, the population size (affecting the quality of the optimality), the number of maximal genetic operations and the termination criteria.

Genetic algorithms are able to process either binary or continuous individuals. However, variables of different types generally cannot be mixed due to limitations of
current genetic algorithm implementations. The classical approach is to transfer all variable types to binary design variables, either explicitly, or through a user interface, as it was realized for the Matlab Genetic Algorithm Optimization Toolbox (GAOT) (Lothrop, [18]). In this case, the value of the continuous variable is calculated as:

\[ v_c = l_{\text{min}} + \frac{l_{\text{max}} - l_{\text{min}}}{2^b} \cdot v_b \]  

where \( v_c \) is the continuous value, \( v_b \) is the binary value, \( l_{\text{min}} \) the lower variable limit, \( l_{\text{max}} \) the upper limit and \( b \) the number of bits.

In fact, this discretization realized by the interface in a transparent way is not necessary adapted to the problem considered by the user.

4.3 Genetic algorithms linked to the MLD form

In the case of predictive control for systems under the MLD form and with genetic algorithms as optimization strategy, the optimization vector is restricted to:

\[ \chi^* = [u(k), \cdots, u(k+N_u-1)]^T \]  

The parameters of the function to be minimized are the current states and the input control variables only. The evaluation function is \( F(\chi, x_0) \) defined as in equation (3). The auxiliary binary and continuous variables of \( \chi \), for evaluation of \( F(\chi, x_0) \), are determined by solving a MILP problem for the equality and inequality constraints of the form \( c\chi < b \) in equation (1), over the prediction horizon \( N \). It must be reminded that once the current states and the input sequence are known, the auxiliary variables are uniquely defined according to the “well posed” aspect. Note that in this technique, no auxiliary variables are optimized, as the GA does not depend upon the structure of the cost function, the only requirement being the possibility of evaluation of the cost function for a particular parameter combination.

The population is created on the input variables space over the chosen control horizon \( N_u \), \( u_k^{k+N_u-1} = [u(k), \cdots, u(k+N_u-1)]^T \) where the vectors \( u = [u_c \ u_f]^T \) contain the continuous and binary control actions. Any individual of this population is a candidate solution for the next iteration step.

Genetic algorithms, as optimization routine in conjunction with the predictive control strategy for MLD form, could offer a new alternative to the binary optimization problems while avoiding the exponential complexity of Branch & Bound technique. In
this point, it can be concluded that the GA methods satisfy the requirements defined at the end of section 4.1. These requirements are fulfilled by avoiding the analytical optimization techniques based on the structural properties of the optimization problem and replacing them with the particular evaluation of the cost function. The exponential complexity is dodged by providing suboptimal solutions.

Genetic algorithms can be used with the classical MLD model without any modification, i.e. having as arguments mixed variables: binary and continuous variables where the continuous variables are transformed and calculated as explained before equation (5), but this may lead to a loss of accuracy through quantification of the continuous control actions. With this remark, an intelligent discretization prior to the effective GA optimization may improve the global optimality of the solution. That is the reason why the following section describes a modified MLD form leading to a time-varying Quadratic (0,1) optimization problem instead of a MIQP structure. These transformations do not change the complexity of the GA to be solved but replace the implicit blind quantification of the continuous control variables with an adaptive discretization technique, which improve the control performances even with a small number of discretized variables.

5. MODIFIED MLD FORM

A new feature in discrete optimization problem for MPC is the introduction of a set of incremental control alternatives. Starting from the MLD form, all the continuous intervals must be converted in such sets.

5.1 Discretization of continuous variables of the MLD form

The MLD form equation (1) clearly shows that variables vectors either include a continuous part $u_c \in U \subset \mathbb{R}^n_c$, $x_c \in X \subset \mathbb{R}^m_c$, $y_c \in Y \subset \mathbb{R}^p_c$ or are completely continuous $z \in Z \subset \mathbb{R}^r_c$. The goal is to replace the continuous intervals $X, U, Y, Z$ by corresponding sets of discrete values $X^d, U^d, Y^d, Z^d$. The discretization implies that at each sampling time, the inputs $u_c^d(k)$ will lie inside a set of discrete control alternatives $U^d$ instead of a real value inside the continuous interval $U$. Discretizing the control variables in $M$ possible alternatives $u_c^d$ leads to the control actions set described by:
\[ U^d = \left\{ u^d_{c,j} \right\}^{j=1,2,\cdots,M} \]  

(7)

In order to illustrate the method, a continuous interval of control actions \( U = \left[ u_0, u_f \right] \in \mathbb{R} \) will be taken as example. It must be transformed into a set of discrete real values. An option is to split the interval in a homogeneous way. Supposing \( M = 2^n \) the number of linearly distributed values, the set \( U^d = \left\{ u^d_{c,0}, u^d_{c,1}, \ldots, u^d_{c,2^n-1} \right\} \) will replace the interval \( U \) with:

\[
u^d_{c}(k) = u_0 + (2^k - 1) \frac{u_f - u_0}{2^n - 1}
\]

(8)

where \( k = 0,1,\ldots,n \). The continuous value is then substituted by \( n \) binary variables \( [d_0 \ d_1 \ \cdots \ d_{n-1}]^T \in \{0,1\}^n \), coding \( u^d_{c}(k) \in U^d \).

It can be proved that knowing the current state \( x(k) \) and applying a discretized input, the updated state will have the possibility to evolve only in a well defined set of discrete values (“well posed” assumption). The following result concerns the implicit discretization of the sets \( X,Y,Z \).

**Proposition:** The discretization of the continuous control actions induces a discrete set for all the other continuous variables. For each logical control value combination \( u_l \), the induced discretized sets satisfy: \( \text{card} U^d \geq \text{card} X^d \), \( \text{card} U^d \geq \text{card} Y^d \), \( \text{card} U^d \geq \text{card} Z^d \).

**Proof:** At instant \( k \) the current state of the system \( x(k) \) is known together with a discretization of the control actions in \( M = 2^n \) alternatives for each input, such that \( \text{card} U^d = M \). For every \( u^d_{c}(k) \in U^d \) a unique pair \( (\delta^d(k),z^d(k)) \) can be defined from the system inequality equation:

\[
E_2\delta^d(k) + E_3z^d(k) \leq E_1 u^d(k) + E_4 x(k) + E_5
\]

(9)

due to the “well posed” assumption of the MLD form, where \( u^d = [u^d_c \ u_l]^T \). Introducing these values in the following equations, the future state and output are found subsequently:

\[
x^d(k+1) = A x^d(k) + B_1 u^d(k) + B_2 \delta^d(k) + B_3 z^d(k)
\]

\[
y^d(k) = C x^d(k) + D_1 u^d(k) + D_2 \delta^d(k) + D_3 z^d(k)
\]

(10)
The relation between the discrete inputs and all other (real or logical) variables can be rewritten as
\[
\begin{align*}
\delta \in & \mathcal{U} \rightarrow \mathcal{U} \\
\mathcal{U} \rightarrow & \mathcal{X} \times \mathcal{Y} \\
\mathcal{X} \times \mathcal{Y} \rightarrow & \mathcal{Z}
\end{align*}
\]
where \( \mathcal{X} = \{0,1\}^n \times \mathcal{Z} \times \mathcal{X} \times \mathcal{Y} \). This means that we are dealing with a mapping
\[
\mathcal{U} \rightarrow \mathcal{X} \times \mathcal{Y} \times \mathcal{Z}.
\]
Noting \( \mathcal{X} = \text{Pr}_\mathcal{X} \text{Im}(F) \), it can be seen that the MLD form is an application
\[
\mathcal{U} \rightarrow \mathcal{X}.
\]
From the surjectivity of \( T \) comes the conclusion that
\[
\text{card} \mathcal{U} \geq \text{card} \mathcal{X}.
\]
In the same way \( \mathcal{X}, \mathcal{Z} \), can be found which completes the proof.

By induction it can be proved that discretized control sets will induce for further sample times the discretization of the continuous intervals. It must be noticed that advancing in time the topology and the cardinality of these sets change as consequence of the new input alternatives available. This means that the control design procedure has to focus only on the discretization of the control actions as it leads to discretization of all other continuous sets \( \mathcal{X}, \mathcal{Y}, \mathcal{Z} \).

The time-invariant discretization by means of linearly distributed values is also the usual kind of technique that genetic algorithms employ in order to transform the mixed optimization problem in a discrete optimization one.

An important note is that the transformation of continuous control actions to a small discrete set can cause chattering or overshoots on the response. In the following subsection, a solution to this problem is proposed by scaling control actions set. Considering such techniques, the discretization is performed in an adaptive manner, allowing the genetic algorithms to approach the global optimum of the predictive control law criterion.

5.2 Fuzzy predictive filter

In order to overcome these difficulties of degraded performances, while keeping a small discretized set, a fuzzy filter is used combined with an adaptation technique. As shown in (Da Costa, [9]) there are two approaches for scaling the discrete sets: fuzzy predictive filters and discrete alternatives based on fuzzy rules. The first one is described here. The design process includes the choice of the adaptive control actions, and the construction of the fuzzy criteria for scaling purposes.

5.2.1 Adaptive distribution of control increments

One major disadvantage of the homogeneous discretization is the conservativeness of the discrete alternatives. This can be avoided by the adaptation of the control increments as a function of the previous sample time value as stated in Figure 5.
Fig. 5. Linear discretization by subintervals.

Let $u_c^d(k-1) = u_p(k-1) \in U$ represent the control action at instant $k-1$, where $U = [u_0, u_f]$ is the domain for a continuous variable. Let also the upper and lower bounds of the possible variation in the control signal at time $k$ be $\Delta_k^+$ and $\Delta_k^-$ respectively, where:

$$\Delta_k^+ = u_f - u_p(k-1), \quad \Delta_k^- = u_p(k-1) - u_0$$

The set of adaptive incremental control alternatives is:

$$U^d(k) = \{0, \gamma_j \Delta_k^+, \gamma_j \Delta_k^- \mid j = 1, 2, \ldots, l\}$$

where the coefficients $\gamma_j \in [0,1]$ are fixed and describe a linear distribution upon the continuous subsets available in relation with the previous control action. The $\gamma_j$ coefficients must thus be chosen instead of choosing fix alternatives as in equation (8). Their choice imposes the changes allowed by scaling the maximum variation $\Delta_k^+$ and $\Delta_k^-$, and $l$ represents the number of possible positive/negative control actions. For a practical implementation, these control actions are coded with $p$ binary variables such that $l = 2^{p-1} - 1$.

5.2.2 Fuzzy scaling factors

Even if the previous technique improves the quality of the control actions, once approaching steady state the control alternatives do not change, and do not permit fine control adjustments. To avoid this problem, a scaling factor $\lambda \in [0,1]$ is introduced, which will modify the subintervals length according to the tracking error (Figure 6).

Fig. 6. Linear discretization by scaled subintervals.
The scaling factor $\lambda$ is time-variant, recomputed at each instant $k$ by a simple fuzzy criterion based for instance on the predicted error $e(k + N - 1|k)$, on the error variation $\Delta e(k) = e(k) - e(k - 1)$ or on the setpoint variations, scaling the previous introduced set of control updates:

$$U^d(k) = \lambda(k) \ U^d(k), \quad \lambda(k) \in [0, 1]$$ (13)

When the error and the error variation are small, the hybrid system is close to a steady state situation, and the discretized control alternatives corresponding to the former continuous inputs should be scaled down to allow tighter control actions, i.e. $\lambda \to 0$. In the same manner when they have higher values, an important corrective variation should be available, i.e. $\lambda \to 1$. Fuzzy criteria have to be designed in order to follow the simultaneous fulfillment of the “small predicted error” and “small error variation” goals described by the membership functions $\mu_e$ and $\mu_{\Delta e}$ respectively. Figure 7 gives an example of such a membership function.

![Fig. 7. Membership function example.](image)

Generally, a weighted conjunction between these two fuzzy criteria could be used:

$$\mu_\lambda = \mu_e \land \mu_{\Delta e}$$ (14)

The gain $\lambda$ privileges large variations when the degree of fulfillment for $\mu_\lambda$ is reduced. Following this idea $\lambda$ is defined as the fuzzy complement:

$$\lambda = \overline{\mu_\lambda} = 1 - \mu_\lambda$$ (15)

The combination of an adaptive set of incremental control alternatives and fuzzy criteria for scaling them forms a fuzzy predictive filter. The filter reduces the problem introduced by the discretization while at the same time providing a set of adaptive alternatives that overcomes the lack of freedom related to a small number of discrete values. This small number of possible control actions being kept low speeds up the optimization.
6. IMPLEMENTATION ISSUES

The modified MLD form for hybrid system presented in the previous section depends on the value of the previous control actions and scaling factors that are recomputed at each sampling period. Thus denoting the MLD system at instant \( k \) by \( \Sigma_{MLD}(k) \), its evolution can be written under the following relation:

\[
\Sigma_{MLD}(k) = \Sigma_{MLD}(u_p(k-1), \lambda(k))
\] (16)

which means that the MLD form is recomputed at each sampling time. In other words, from the implementation point of view, the classical MLD form has been replaced by a new time-varying version. This can be implemented through the specification language HYSDEL (Hybrid System Description Language) (Bemporad and Mignone, [4]) as follows. The upper and lower bound of the continuous interval of each continuous control variable \( u_0 \) and \( u_f \) are defined globally (once) in the HYSDEL language, while the previous control action \( u_p(k-1) \) and the fuzzy scaling factors \( \lambda \) are defined as a real parameter. So the corresponding continuous control variable \( u_{c,i} \) is represented by:

\[
u_{c,i}(k) = u_{p,i}(k-1) + s_i(k) \lambda_i(k) \frac{\Theta_i \psi_i(k)}{2^{p_i-1}} \Delta_{ki} + (1 - s_i(k)) \lambda_i(k) \frac{\Theta_i \psi_i(k)}{2^{p_i-1}} \Delta_{ki}
\] (17)

where: \( \Theta_i = \begin{bmatrix} 2^0 & 2^1 & \cdots & 2^{p_i-2} \end{bmatrix} \), \( \psi_i(k) = \begin{bmatrix} d_0 & d_1 & \cdots & d_{p_i-2} \end{bmatrix} \in [0,1]^{p_i-1} \)

\( s_i \in \{0,1\} \) representing the sign of the control action is considered as a binary optimization variable. The direct specification of the dynamics of the MLD system using the form equation (17) is not possible because the term \( u_p(k-1) \) can not be implemented into the HYSDEL current version. Consequently, the easiest way to overcome this difficulty is to treat \( u_p(k-1) \) as an additional continuous auxiliary variable, which has no impact on the complexity of the MLD model and on the computational burden.

Once the modified MLD form completely described, a routine, which performs the evaluation for the genetic algorithm, has to be created. This routine interprets the GA at each iteration, solving a MILP problem for the equality and inequality equations of the MLD form. It results in the response to the \( \delta^d, z^d, x^d, y^d \) values and then the
evaluation of the cost function equation (2). The whole control design procedure described above is summarized in the diagram of Figure 8.

![Diagram of Figure 8. Control scheme including the modified MLD form.](image)

An important note is that the number of binary variables, coding the discrete alternatives, could be chosen as small as 2 or 3 without loss of performances. Such choices offer good precision in general cases due to the adaptive mechanism introduced by the adaptive distribution of control increment $\frac{\gamma_j \Delta t}{\Delta \gamma_j}$ and the fuzzy scaling factors $\lambda$, while decreasing the computational time and complexity.

### 7. APPLICATION

The previously described predictive control strategy is applied to the three tanks benchmark system presented in (Dolanc et al., [10]). The details of the classical MLD model for the three tanks benchmark can be found in (Bemporad et al., [2]).

![Three tanks benchmark system.](image)

#### 7.1 Benchmark description

The tanks 1 and 2 in Figure 9 are loaded through the two flow inputs denoted by $Q_1$ and $Q_2$ taking continuous values between 0 and $Q_{\text{max}}$. All the valves $V_1$, $V_2$, $V_{13}$, $V_{23}$, $V_{L1}$, $V_{L2}$ and $V_{L3}$ are on-off type. The valves $V_{L1}$ and $V_{L2}$ remain closed and $V_{L3}$ open. The objective is to maintain a predefined constant level into the tank 3. The state space vector includes the water levels in the tanks $h_1$, $h_2$ and $h_3$. The mass conservation gives the following differential equation:
\[ \dot{h}_1 = \frac{1}{A} (Q_1 - Q_{13V1} - Q_{13V1}) \]
\[ \dot{h}_2 = \frac{1}{A} (Q_2 - Q_{23V2} - Q_{23V2}) \]
\[ \dot{h}_3 = \frac{1}{A} (Q_{13V1} + Q_{13V1} + Q_{23V2} + Q_{23V2} - Q_N) \]

where \( Q_y \) are the flows and \( A \) is the section of each tank. Using Toricelli’s law and linearizing the flows leads to:

\[ Q_{13V1} \approx k_{i3} V_i (h_i - h_3) \]
\[ Q_{13V1} \approx k_i V_i (\max(h_v, h_i) - \max(h_v, h_3)) \]
\[ Q_N = k_{N3} V_{L3} h_3 \]

where: \( k_{i3} = a_z S_{i3} \sqrt{\frac{2g}{h_{\text{max}}}}, \quad i = 1,2 \), \( k_i = a_z S_i \sqrt{\frac{2g}{h_{\text{max}} - h_v}}, \quad k_{N3} = a_z S_{N3} \sqrt{\frac{2g}{h_{\text{max}}}} \)

The MLD model implies the following set of variables:

\[ x = [h_1 \ h_2 \ h_3]^T \]
\[ u = [Q_1 \ Q_2 \ V_1 \ V_2 \ V_{13} \ V_{23}]^T \]
\[ \delta = [\delta_{01} \ \delta_{02} \ \delta_{03}]^T \]
\[ z = [z_{01} \ z_{02} \ z_{03} \ z_1 \ z_2 \ z_{13} \ z_{23}]^T \]

with:

\[ [\delta_{0i}(t) = 1] \leftrightarrow [h_i(t) \geq h_v], \quad i = 1,2,3 \]
\[ z_{0i}(t) = \delta_{0i}(t) (h_i(t) - h_v), \quad i = 1,2,3 \]
\[ z_i(t) = V_i (z_{0i}(t) - z_{03}(t)), \quad i = 1,2 \]
\[ z_{13}(t) = V_{13} (h_i(t) - h_3) \quad i = 1,2 \]

7.2 Modified MLD form of the three tanks benchmark

Following the technique presented before, the inputs \( Q_1 \) and \( Q_2 \) are discretized and coded by three binary variables including the sign (\( p = 3 \)), as follows:

\[ \Delta Q_1 = \pm \lambda_1 \frac{2^1 d_{11} + 2^0 d_{12}}{(2^2 - 1)} \Delta k_1^{+/-} \]
\[ \Delta Q_2 = \pm \lambda_2 \frac{2^1 d_{21} + 2^0 d_{22}}{(2^2 - 1)} \Delta k_2^{+/-} \]
with $d_1, d_2, d_3, d_4 \in \{0,1\}$ and the scaling factors $\lambda_1, \lambda_2 \in [0,1]$ recomputed at each instant $k$ by fuzzy rules.

The variable giving the sign of the variation for the control actions, $\lambda_j \Delta_{k}^i$ or $\lambda_j \Delta_{k}^i$, represents an optimization variable. In conclusion, the modified MLD form has as input the following vector:

$$u = [s_1 \ d_{11} \ d_{12} \ s_2 \ d_{21} \ d_{22} \ V_1 \ V_2 \ V_1 \ V_3]^{T}$$

(23)

### 7.3 Simulation results

Let consider now the following specification: starting from zero levels (the three tanks being completely empty), the objective of the control strategy is to reach the liquid levels $h_1 = 0.5$ m, $h_2 = 0.5$ m and $h_3 = 0.1$ m. According to the known specified levels, fuzzy criteria based on Gaussian membership functions similar to those given in Figure 7 are designed. The genetic algorithm for the optimization problem of equation (2), under the dynamic constraints described with the modified MLD model, has been applied in simulation to reach the level specification previously given with the two prediction horizons $N = 2$ and $N_u = 2$. The results are presented on Figure 10(a) for the three tanks levels and on Figure 10(b) for the control signals.

The level in the third tank is oscillating around 0.1 m, since $h_3 = 0.1$ m is not an equilibrium point. The continuous signals $Q_1$ and $Q_2$ shown in Figure 10(b) have various values introduced by the adaptive effect, proving the freedom brought in by the intelligent discretization procedure. For a comparison purpose, Figure 11 presents results obtained for the same tuning $N = N_u = 2$ with a classical approach including B&B optimization. The sub-optimality of the GA strategy does not affect the observed performance.
The following problem, where the valves $V_{L1}$ and $V_{L2}$ are not forced to be closed and the specifications levels are $h_1 = 0.1m$, $h_2 = 0.1m$ and $h_3 = 0.2m$, is impossible to solve in reasonable time with MIQP solvers for the MLD form because it demands a large prediction horizon. Due to the restricted computational time obtained with the GA technique, this problem can now be solved for $N = 5$, $N_u = 2$, where the number of binary variables of the modified MLD model is 10 (equation (23)). The results are presented in Figure 12.
Table 1. Computational times (sec.) for predictive control problems for different prediction and control horizons

<table>
<thead>
<tr>
<th>$N$</th>
<th>$N_u$</th>
<th># of binary variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>16.20 18.37 20.58 23.20 25.21 26.66</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>X 50.66 57.25 64.15 70.86 78.58</td>
</tr>
<tr>
<td>3</td>
<td>24</td>
<td>X X 77.50 84.73 96.33 105.33</td>
</tr>
<tr>
<td>4</td>
<td>32</td>
<td>X X X 105.08 117.08 128.50</td>
</tr>
<tr>
<td>5</td>
<td>40</td>
<td>X X X X 131.87 147.15</td>
</tr>
</tbody>
</table>

From Table 1, it is clear that the genetic algorithm with the technique described before is more effective with the long prediction and control horizon, i.e. with a large number of binary variables rather than with a small number of binary optimization variables, as the complexity depends linearly on the number of binary variables.

8. CONCLUSIONS

The paper presents the discrete optimization with the genetic algorithm as a design control strategy for the predictive control of hybrid systems modeled with the MLD form. With this aim, a modified discretized MLD form has been developed using adaptive discretization applying fuzzy predictive filter and fuzzy scaling factors for the continuous control actions which induce the transformation of all continuous intervals into discrete sets. The resulting modified MLD form improves the control performance of the Genetic Algorithm.

Genetic Algorithms provide a solution for the optimization problem while avoiding the computational load inherent in the classical Branch & Bound methods for solving Quadratic (0,1)-problems. They are not taking advantage from the structural properties, but are able to give solutions for large scale problems without facing the problem of exponential growing of the computational time. Thus, it is easy to tune the predictive control law by adjusting the prediction and control horizons, which is a hard task in the classical approaches that are drastically limiting their applications.

ACKNOWLEDGMENT

The first author acknowledges the support from the E.C. (HPAW-CT-2002-80062).
REFERENCES

[21] O. Stursberg, S. Panek, Control of switched hybrid systems based on disjunctive formulations, 5th Int. workshop on hybrid systems: Computation and Control, Stanford (CA, USA), 2002.