Computers and Chemical Engineering 32 (2008) 3254–3263

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

Hybrid fuzzy predictive control based on genetic algorithms for the temperature control of a batch reactor

Javier Causa[a], Gorazd Karer[b,*], Alfredo Núñez[a], Doris Sáez[a], Igor Škrjanc[b], Borut Zupančič[b]

[a] Facultad de Ciencias Físicas y Matemáticas, Universidad de Chile, Chile
[b] Faculty of Electrical Engineering, University of Ljubljana, Slovenia

ARTICLE INFO

Article history:
Received 23 September 2007
Received in revised form 20 May 2008
Accepted 29 May 2008
Available online 5 June 2008

Keywords:
Process control
Model predictive control
Fuzzy systems
Hybrid systems

ABSTRACT

In this paper we describe the design of hybrid fuzzy predictive control based on a genetic algorithm (GA). We also present a simulation test of the proposed algorithm and a comparison with two hybrid predictive control methods: Explicit Enumeration and Branch and Bound (BB). The experiments involved controlling the temperature of a batch reactor by using two on/off input valves and a discrete-position mixing valve. The GA-hybrid predictive control strategy proved to be a suitable method for the control of hybrid systems, giving similar performance to that of typical hybrid predictive control strategies and a significant saving with respect to the computation time.

1. Introduction

Model Based Predictive Control (MBPC) consists of optimizing the process behavior to obtain optimal future control actions. In the MBPC framework the use of non-linear models with continuous and/or discrete variables has been considered in order to obtain better representations of the process non-linearities. Thus, we present the more recent reports related to fuzzy predictive control, hybrid predictive control and the new hybrid predictive control approach.

Firstly, simplified solutions of non-linear fuzzy predictive control methods were developed, such as the fuzzy predictive controller based on the Takagi–Sugeno fuzzy model linearization proposed by Roubos, Babuska, Bruijn, and Verbruggen (1998). In that paper a linear model for every sampling time is derived by evaluating the fuzzy model premises and a linear predictive controller is designed. The method provides a sub-optimal solution, but requires less computational time, as Kim et al. point out in a similar study (Kim & Huh, 1998). Espinosa, Vandewalle, and Wertz (2005) and Espinosa and Vandewalle (1998) propose a fuzzy predictive control algorithm based on an approximation of the free and forced responses of the fuzzy model. In such an approach the predictive control analytical solution is similar to the one obtained with the linear MBPC algorithm. Hadjili and Wertz (1999), Espinosa and Vandewalle (1999) and Nounou and Passino (1999) describe similar predictive controllers, where the fuzzy predictor is linearly approximated by using constant satisfaction degrees for the future horizons and an analytical solution of a linear MBPC is applied. These algorithms compare favorably with Roubos's algorithms (Roubos et al., 1998).

More robust solutions of the fuzzy predictive control strategy have been proposed. Babuska (1998) and Babuska, Sousa, and Verbruggen (1999) developed a multi-step fuzzy predictor for longer prediction horizons. However, these solutions require a longer computation time. Mahfouf, Kandiah, and Linkens (2002) consider a Takagi–Sugeno (T–S) fuzzy model with different fuzzy partitions of the input space.

Recently, in order to appropriately control processes that contain discrete and/or continuous variables (hybrid systems), hybrid predictive control techniques were developed. Slupphaug, Vada, and Foss (1997) and Slupphaug and Foss (1997) describe a predictive controller with continuous and integer input variables solved by non-linear mixed integer programming.

Bemporad and Morari (1999) and Bemporad, Borrelli, and Morari (2002) present a predictive control scheme for hybrid systems solved by using Mixed Integer Quadratic Programming (MIQP). The proposed algorithm was applied to a gas-supply system that considers quantized manipulated variables. The main problem with MIQP is its computational complexity, which increases the time required to find the solution. Bemporad, Giovanardi, and Torrisi (2000) present a predictive control design for Piece-wise affine (PWA) systems, as these are models for describing both non-

* Corresponding author.
E-mail address: gorazd.karer@fe.uni-lj.si (G. Karer).

0098-1354/$ – see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compchemeng.2008.05.014

linear and hybrid systems. In this case, reachability conditions are established. In addition, Bemporad, Heemels, and Schutter (2002) established that a hybrid system with the predictive control based on a quadratic objective function and linear constraints is a subclass of the Mixed Logical Dynamical (MLD) hybrid system. In other words, the closed-loop system corresponds to a hybrid system. This result opens up the use of robustness and stability tools developed for the hybrid model classes, to study the closed-loop properties of hybrid predictive control.

Borrelli, Baotic, Bemporad, and Morari (2003) and Borrelli (2003) propose a finite-time optimal control solution for PWA systems with a quadratic performance criterion. The controller is based on a dynamic programming recursion and a multiparametric quadratic programming solver. Thus, the optimization problem is solved for each partition of the PWA system. Baotic, Christophersen, and Morari (2003) present a linear criterion for the proposed algorithm that results in a reduced computation time. Thomas, Dumur, and Buisson (2004) propose a hybrid predictive controller partitioning in the state-space domain. In every partition some variables change, while the others remain constant. This approach also reduces the computation time. Beccuti, Geyer, and Morari (2003) present a hybrid predictive approach based on a temporal decomposition scheme. In this case duality properties are used to translate the original optimal control problem into a temporal sequence of independent subproblems with a smaller dimension. This solution approximates the optimal, but the computation time is significantly reduced. On the other hand, Potočnik, Mušič, and Zupančič (2004) propose a hybrid predictive control algorithm with discrete inputs based on a reachability analysis. The computation time is reduced by building and pruning an evolution tree.

Škrjanc, Blažič, and Agamenonni (2005) present modelling and identification using the interval fuzzy model (INFUMO). This approach is useful for describing a family of uncertain non-linear functions or when systems with uncertain physical parameters are observed.

In a recent study, Núñez, Saez, Oblak, and Škrjanc (2006) present a hybrid predictive control strategy based on a fuzzy model. The key element of the fuzzy identification is the detection and estimation of switching regions by combining fuzzy clustering and a principal component analysis. The non-linear NP-hard optimization problem was solved efficiently, by using the genetic algorithms, in terms of accuracy and computation time.

A self-adaptive supervisory predictive functional control for applications in a semi-batch reactor in which the optimal operation is to follow the reference trajectory without significant overshoot is presented in Škrjanc (2008).

Both fuzzy and hybrid predictive controllers correspond to non-linear predictive control strategies that are required to solve an NP-hard problem given by the non-linear optimization problem associated with the predictive objective function and the non-linear predictive model (fuzzy and/or hybrid model). Note that neural network predictive control also corresponds to non-linear predictive control. To solve these kinds of NP-hard problems, evolutionary algorithms have been proposed.

Recently, in order to obtain a good solution in a reasonable time for the fuzzy predictive control optimization problem, Sarimveis and Bafas (2003) propose the specialized genetic algorithm (GA) optimization method for fuzzy predictive control based on Takagi–Sugeno models.

GAs have also been applied for a non-linear predictive controller based on neural networks. Shin and Park (1998) describe a neural predictive controller based on a GA that shows better performance than a Quasi-Newton optimization technique. Woolley, Kambhampati, Sandoz, and Warwick (1998) describe a GA predictive controller based on radial basis functions (RBF). Furthermore, as a new, efficient and optimization evolutionary algorithm, particle swarm optimization (PSO) (Kennedy & Eberhart, 2001) has been developed. This is inspired by the social behavior of animals and insects and is defined by the behavior of a swarm of particles in a multidimensional search space.

Coelho, de Moura Oliveira, and Cunha (2005) present a predictive controller based on recursive linear models where the optimization problem is solved by PSO. The good performance of PSO is demonstrated in comparison with a GA and classical Quasi-Newton methods. Wang and Xiao (2005) describe a PSO-based predictive controller based on a radial basis function (RBF) neural network model by obtaining slightly better results than a GA and a Quasi-Newton method.

Solis, Saez, and Estevez (2006) propose the application of PSO as an efficient tool for the design of fuzzy predictive control (FPC) strategies. The performance of the proposed method is successfully evaluated in terms of the accuracy and the computation time.

In this study, the design of Hybrid Fuzzy Predictive Control based on a GA (HFPC-GA) is described and applied to a batch reactor (Karer, Mušič, Škrjanc, & Zupančič, 2007a, 2007b; Karer, Škrjanc, & Zupančič, in press). In Section 2, we formally describe the HFPC-GA formulation, which includes a prediction based on a hybrid fuzzy model of the process. Two approaches for solving such an NP-hard problem are described: Branch and Bound (BB) and the GA. The former corresponds to the non-linear optimization algorithm used in (Karer et al., 2007a), and the second, the newly proposed approach, HFPC-GA. In Section 3 the batch reactor process and its corresponding hybrid fuzzy modelling are described. In Section 4, numerical examples are presented to show the benefits of applying hybrid fuzzy predictive control based on a GA, compared with BB. Finally, Section 5 concludes with an analysis, comments and further research directions.

2. Model predictive control of systems with discrete inputs based on a reachability analysis

Model predictive control is an approach where a model of the system is used to predict the future evolution of the system (Camacho & Bordons, 1998; Maciejowski, 2002). The most appropriate input vector is established and applied for every time step. Its determination is an optimization problem that is solved within a finite horizon N_u, i.e., for a pre-specified number of time steps ahead. For each time step k a sequence of optimal input vectors (1) is acquired; this minimizes the selected cost function while considering the eventual constraints of the inputs, outputs and system states. However, only the first vector of the optimal sequence is actually applied during the current time step. In the next time step, a new optimal sequence is determined, etc.

$$U_k^{k+N_u-1} = \{u(k), u(k+1), \ldots, u(k+N_u-1)\} \quad (1)$$

The Hybrid Fuzzy Predictive Control (HFPC) strategy is a generalization of Model Predictive Control (MPC), where the prediction model includes both discrete-integer and continuous variables. In this study we propose a hybrid fuzzy prediction model.

In general, a hybrid predictive control design minimizes the following generic objective function. This particular case corresponds to the most common objective function used for predictive control

purposes.

$$\min_{\{u(k),u(k+1),\ldots,u(k+N_u-1)\}} J = J_1 + \lambda J_2$$
$$J_1 = \sum_{h=N_1}^{N_y} (\hat{y}(k+h) - r(k+h))^2, \quad J_2 = \sum_{h=N_1}^{N_u} \Delta u(k+h-1)^2 \tag{2}$$
subject to
$$\hat{y}(k+h) = f(\hat{y}(k+h-1), \ldots, u(k+h-1))$$
$$u(k+h-1) \in U$$

Here J is the objective function, $\hat{y}(k+h)$ corresponds to the h-step-ahead prediction for the controlled variable, $r(k+h)$ is the reference, $\Delta u\ (k+h-1)$ is the increment of the control action and λ is the weighting factor. N_1, N_y and N_u are the prediction horizons and the control horizon, respectively. $U_k^{k+N_u-1} = \{u(k), \ldots, u(k+N_u-1)\}$ represents the control action sequence, which corresponds to the optimization variables. U is a set of discrete input values.

For the hybrid fuzzy predictive control design proposed, the prediction model is given by a non-linear function as a T–S fuzzy hybrid model and the manipulated variable and/or state variable are integer/discrete. This non-linear optimization problem corresponds to NP-hard and, therefore, we propose two approaches: a BB method and a GA.

2.1. The Branch and Bound approach (HFPC-BB)

The control algorithm used in this paper is thoroughly described in Karer et al. (2007a) and Potočnik, Mušič, and Zupančič (2005). Since it is limited to systems with discrete inputs only, the possible evolution of the system over time steps h up to a maximum prediction horizon N_u can be illustrated by a tree of evolution, as shown in Fig. 1 for $N_u = 4$ and 3 input vectors. The nodes of the tree represent reachable states, and branches connect two nodes if a transition exists between the corresponding states.

For a given root-node V_1, representing the initial states $\underline{x}_i = \underline{x}(k|k)$ and $q_i = q(k|k)$, the reachable states are computed and inserted in the tree as nodes V_i, where i indices the nodes as they were successively computed. The notation $(k_1|k_2)$ denotes the time step of the current node in the MPC algorithm (k_1) and the time step in which the algorithm started (k_2), i.e., the actual time step in the control process.

A cost value J_i is associated with each new node, and based on the cost value the most promising node is selected. After labelling the node as explored, new reachable states emerging from the selected node are computed. The construction of the tree of evolution continues upwards first, until one of the following conditions occurs:

- The value of the cost function at the current node is greater than the current optimal one ($J_i \geq J_{\text{opt}}$).[1]
- The maximum step horizon is reached ($h = N_u$).

If the first condition occurs, the node is labelled as non-promising and thus eliminated from further exploration. On the other hand, if the node satisfies the second condition only, it becomes the new current optimal node ($J_{\text{opt}} = J_i$), whereas the sequence of input vectors leading to it becomes the current optimal one.

The exploration continues from the topmost step horizon, where unexplored nodes can be found, etc., until all the nodes are explored and the optimal input vector $\underline{u}_{\text{opt}}(k)$ can be derived from the current optimal sequence. The optimal input vector is applied to the system $\underline{u}_{\text{opt}}(k)$ and the whole procedure is repeated at the next time step $k+1$.

For an insight into the computational complexity issues and the approaches and properties used for dealing with them, see Karer et al. (2007a).

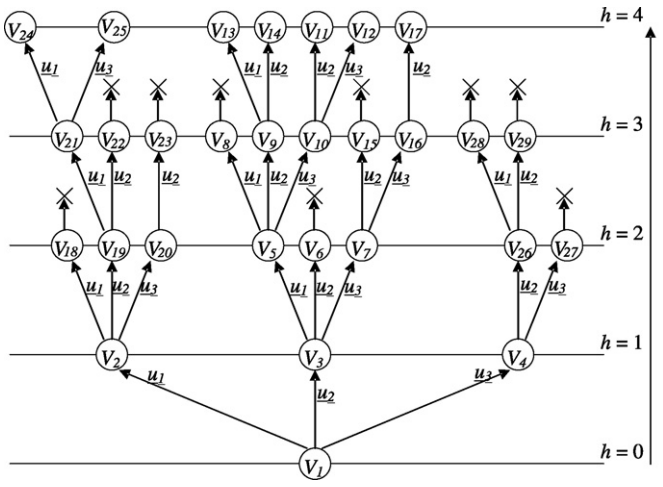

Fig. 1. Example of an explored tree of evolution. The optimal node is V_{14}, therefore the input $\underline{u}_{\text{opt}} = \underline{u}_2$ is selected and applied.

2.2. Optimization based on a genetic algorithm (HFPC-GA)

The GA method is suitable for NP-hard optimization problems with discrete or integer variables, and therefore the binary codification is not necessary. In other words, the genes of the individuals (feasible solutions) are given directly by the integer optimization variables. In addition, gradient computations are not necessary, as in conventional non-linear optimization solvers, which allows us to save a significant amount of computation time.

The optimization based on a GA (Man, Tang, & Kwong, 1998), presented in Fig. 2, can be described by the following steps:

(1) Initialize a random population of individuals corresponding to the feasible solutions.
(2) Evaluate the objective function for each individual of the current population.
(3) Select random parents from the current population.
(4) Apply genetic operators like *crossover* and/or *mutation* to the parents, for a new generation.
(5) Evaluate the objective function for all the individuals of the generation.
(6) Choose the best individuals according to the best values of the objective function.
(7) Replace the weakest individuals of the previous generation with the best ones of the new generation obtained in Step 6.
(8) If either the value of the objective function reaches a certain tolerance or the maximum number of generations is reached, then the algorithm stops. Otherwise, go to Step 2.

In general, genetic algorithms efficiently cope with non-linear mixed/integer optimization problems with constraints, which can be input constraints or the constrains of the process states (Man et al., 1998). Another advantage is that the objective function gradient does not need to be calculated, which relaxes the computational effort.

A potential solution of the genetic algorithm is called an individual. The individual can be represented by a set of parameters related

[1] Before beginning the exploration of the tree of evolution, the initial value of the current optimal node is set to infinity $J_{\text{opt}} = \infty$.

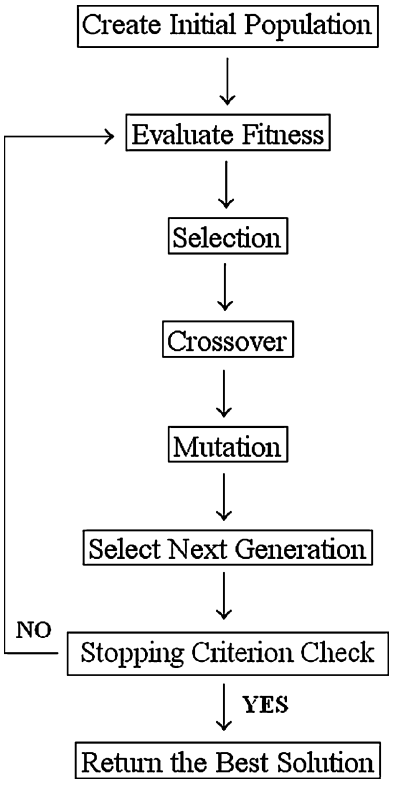

Fig. 2. GA flowchart.

to the genes of a chromosome and can be described in binary or integer form. The individual represents a possible control action sequence $U_k^{k+N_u-1} = \{u(k), \ldots, u(k+N_u-1)\}$, where each element is called a gene, and the individual length corresponds to the control horizon N_u.

Using genetic evolution, the fittest chromosome is selected to ensure the best offspring. The best parent genes are selected, mixed and recombined for the production of the offspring in the next generation. For the recombination of the genetic population, two fundamental operators are used: crossover and mutation. For the crossover mechanism, the portions of two chromosomes are exchanged with a certain probability in order to produce the offspring. The mutation operator alters each portion randomly with a certain probability (Man et al., 1998).

In summary, the proposed genetic algorithm solution provides a solution near to the optimum. The tuning parameters of the GA-method are as follows: the number of individuals which denotes the number of possible solutions; the number of generations which means the number of iterations; the crossover probability; the mutation probability; and the stopping criteria, which defines the precision of the algorithm.

The solving of constrained optimization problems using GA is a very complex issue due to the genetic operations (mutation and crossover) do not guarantee solution feasibility. Although much attention has been given to solve these issues, no general and systematic solution has been proposed.

There are many publications considering constraints in optimization problems based on GA. Back, Fogel, and Michalewics (2000), Coello (2002), and Michalewicz (1995) are excellent reviews and methods, but no general methodology has been proposed. One of the most important methods is GENOCOP proposed by Michalewicz and Nazhiyath (1995), who developed this genetic algorithm-based program for constrained and unconstrained optimization.

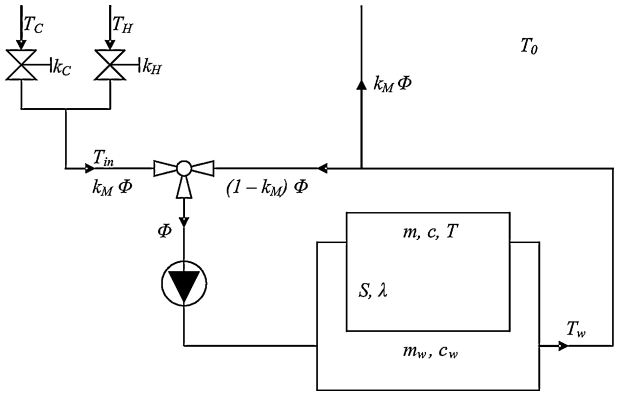

Fig. 3. Scheme of the batch reactor.

Recent work has shown promising results for a Feasible–Infeasible Two-Population (FI-2Pop) genetic algorithm for constrained optimization Kimbrough, Koehler, Lu, and Wood (2008). The FI-2Pop GA has proved to be better than standard methods for handling constraints in GAs; it has regularly produced better decisions for comparable computational effort than GENOCOP. Moreover FI-2Pop GA is a high-quality GA solver engine for constrained optimization problems generating excellent decisions for problems that cannot be handled by GENOCOP.

3. The batch reactor

The aforementioned approaches to solving the optimization problem arising from the optimal control problem were tested on a simulation example of a real batch reactor that is situated in a pharmaceutical company and is used in the production of medicines. The goal is to control the temperature of the ingredients stirred in the reactor core so that they synthesize into the final product. In order to achieve this, the temperature has to follow the reference trajectory, given in the recipe, as accurately as possible.

A scheme of the batch reactor is shown in Fig. 3. The reactor's core (temperature T) is heated or cooled through the reactor's water jacket (temperature T_w). The heating medium in the water jacket is a mixture of fresh input water, which enters the reactor through on/off valves, and reflux water. The water is pumped into the water jacket with a constant flow Φ. The dynamics of the system depend on the physical properties of the batch reactor, i.e., the mass m and the specific heat capacity c of the ingredients in the reactor's core and in the reactor's water jacket (here, the index w denotes the water jacket). λ is the thermal conductivity, S is the contact area and T_0 is the temperature of the surroundings.

The temperature of the fresh input water T_{in} depends on two inputs: the positions of the on/off valves k_H and k_C. However, there are two possible operating modes of the on/off valves. In case $k_C = 1$ and $k_H = 0$, the input water is cool ($T_{in} = T_C = 12\,^\circ C$), whereas if $k_C = 0$ and $k_H = 1$, the input water is hot ($T_{in} = T_H = 75\,^\circ C$).

The ratio of fresh input water to reflux water is controlled by the third input, i.e., by the position of the mixing valve k_M. There are six possible ratios that can be set by the mixing valve. The share of fresh input water can be either 0, 0.01, 0.02, 0.05, 0.1 or 1.

We are therefore dealing with a multivariable system with three discrete inputs (k_M, k_H and k_C) and two measurable outputs (T and T_w).

Due to the nature of the system, the time constant of the temperature in the water jacket is obviously much shorter than the time constant of the temperature in the reactor's core. Therefore, the batch reactor is considered as a stiff system.

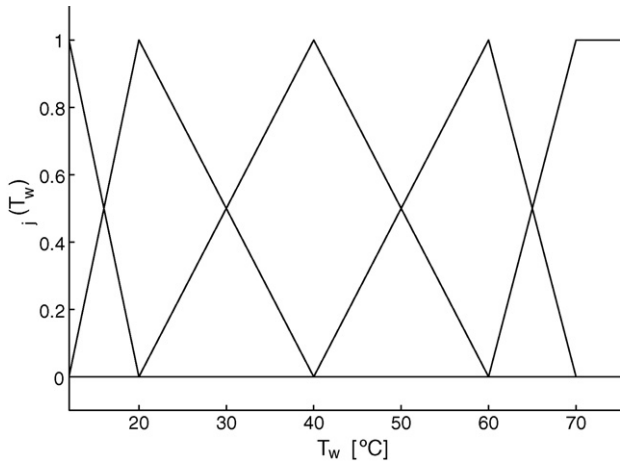

Fig. 4. Membership functions.

3.1. Modelling and identification of the batch reactor

The modelling procedure is explained in detail in Karer et al. (2007a).

The temperature in the reactor's core T is influenced only by the heat conduction between the reactor's core and the reactor's water jacket. Furthermore, we have surmised that the heat conduction is proportional to the temperature difference between the reactor's core T and the reactor's water jacket T_w.

Therefore, a first-order linear MISO submodel can be presumed, as shown in (3). The system parameters are given below.

$$\hat{T}(k+1) = \Theta_c^T [T_w(k) T(k)]^T \tag{3}$$

$$\Theta_c^T = [0.0033 \quad 0.9967] \tag{4}$$

The temperature in the reactor water jacket T_w is influenced by the temperature in the core T, the fresh input water inflow at the mixing valve k_M, and the position of the cold-water and hot-water on/off valves k_C and k_H.

Let us assume two operating modes of the subsystem ($s = 2$).

- The first operating mode ($q = 1$) is the case when the fresh input water is hot, i.e., $k_C(k) = 0$ and $k_H(k) = 1$.
- The second operating mode ($q = 2$) is the case when the fresh input water is cool, i.e., $k_C(k) = 1$ and $k_H(k) = 0$.

$$q(k) = q(k_C(k), k_H(k)) = \begin{cases} 1 & \text{if} \quad k_C(k) = 0 \wedge k_H(k) = 1 \\ 2 & \text{if} \quad k_C(k) = 1 \wedge k_H(k) = 0 \end{cases} \tag{5}$$

Next, the membership functions have to be defined. The system is fuzzyfied with regard to the temperature in the reactor's water jacket $T_w(k)$. Simple triangular functions are used, as shown in Fig. 4.

Such a form of the membership functions ensures that the normalized degrees of fulfillment $\beta_j(T_w)$ are equal to the membership values $\mu_j(T_w)$ across the whole operating range for each rule $\mathbf{R}^{jd}$, respectively. The normalized degrees of fulfillment $\beta_j(T_w)$ make up a normalized vector of fulfillment $\beta(T_w(k)) = \beta(k)$. In this case there are five membership functions ($K = 5$), with maximums at 12°, 20°, 40°, 60° and 70°, so that the whole operating range is covered.

The rule base of the hybrid fuzzy model is given in (6). We presume that a local system corresponding to an individual rule $\mathbf{R}^{jd}$ is affine.

$$\begin{aligned} &\mathbf{R}^{jd}: \\ &\text{if } q(k) \text{ is } Q_d \text{ and } T_w(k) \text{ is } A_1^j \\ &\text{then } T_w(k+1) = a_{1jd} T_w(k) + a_{2jd} T(k) + b_{1jd} k_M(k) + r_{jd} \\ &\text{for } j = 1, \ldots, 5 \text{ and } d = 1, 2 \end{aligned} \tag{6}$$

The output of the model of the temperature in the reactor's water jacket is written in compact form in (7)–(9).

$$\hat{T}_w(k+1) = \beta(k) \Theta_w^T(k) [T_w(k) T(k) k_M(k) k_M(k) 1]^T \tag{7}$$

$$\Theta_w(k) = \begin{cases} \Theta_{w1} & \text{if} \quad q(k) = 1 \\ \Theta_{w2} & \text{if} \quad q(k) = 2 \end{cases} \tag{8}$$

$$\Theta_{w1} = \begin{bmatrix} 0.9453 & 0.9431 & 0.9429 & 0.9396 & 0.7910 \\ 0.0376 & 0.0458 & 0.0395 & 0.0339 & 0.0225 \\ 19.6748 & 16.7605 & 10.5969 & 3.9536 & 1.6856 \\ 0.3021 & 0.2160 & 0.5273 & 1.2701 & 12.0404 \end{bmatrix} \tag{9}$$

$$\Theta_{w2} = \begin{bmatrix} 0.9803 & 0.9740 & 0.9322 & 0.9076 & 0.8945 \\ 0.0025 & 0.0153 & 0.0466 & 0.0466 & 0.0111 \\ -0.0704 & -0.6956 & -7.8013 & -12.2555 & -18.7457 \\ 0.2707 & 0.2033 & 0.5650 & 1.9179 & 5.6129 \end{bmatrix} \tag{10}$$

4. Results

For the hybrid predictive control optimization problem of the batch reactor we propose the cost function given by (11) (see also Karer et al., 2007a).

$$\begin{aligned} j = w_1 \sum_{h=1}^{N_y} (T(k+h) - T_{\text{ref}}(k+h))^2 + w_2 \sum_{h=1}^{N_u} k_C(k+h) k_H(k+h-1) \\ + w_3 \sum_{h=1}^{N_u} |k_M(k+h) - k_M(k+h-1)| k_H(k+h-1) \end{aligned} \tag{11}$$

$$w_1 = \frac{1}{15}, w_2 = 15, w_3 = 0.03$$

In this study the prediction horizons considered are $N_u = N_y = N = 4$, 5 and 6. The sampling time of the prediction model equals $T_s = 10$ s. Therefore, the optimization problem must be solved within 10 s. Note that the inputs are allowed to change only every 15 time steps (see Karer et al., 2007a). This means that the control action will be held during a control sampling time T_{cs}, which equals 150 s.

The set of possible input variables $u(k+h-1)$, $h = 1 \ldots N_u$, is defined in (12).

$$M = \left\{ \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0.01 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0.02 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0.05 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0.1 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \right\} \tag{12}$$

- The first row denotes the mixed valve input $k_M \in \{0, 0.01, 0.02, 0.05, 1\}$.
- The second row is the cool-water on/off valve input $k_C \in \{0, 1\}$.
- The third row denotes the hot-water on/off valve input $k_H \in \{0, 1\}$.

There are two possible output disturbances that are often encountered in real-life applications of batch reactors: we should

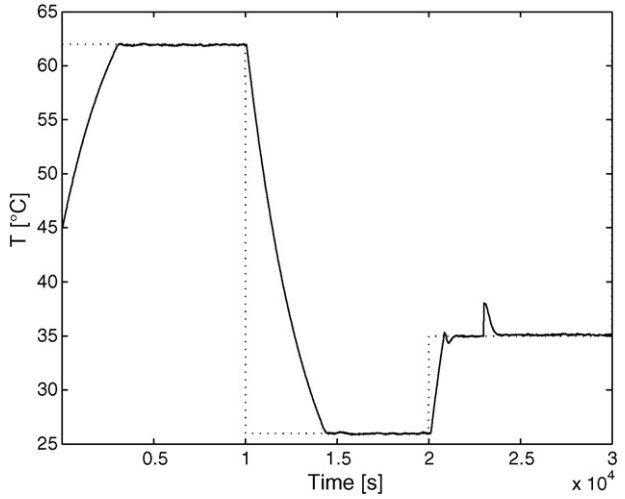

Fig. 5. Results of HFPC based on BB: core temperature T (solid line) and reference temperature T_{ref} (dotted line).

consider a disturbance added to temperature in the reactor's core T and a disturbance added to temperature in the reactor's water jacket T_w. Due to faster dynamics, it is much easier for the control algorithm to take care of the latter; only the former has thus been considered in the experiments. Therefore, at time 23,000 s, a step with amplitude 3 °C was added to the temperature in the reactor's core T as a persistent disturbance.

4.1. The Branch and Bound approach—results

The results of the experiment using the HFPC-BB approach for $N_y = N_u = 4$ are shown in Figs. 5 and 6.

The control results for $N_y = N_u = 5$ and $N_y = N_u = 6$ are almost identical in this case.

4.2. Optimization based on a genetic algorithm—results

In this case the individuals for the HFPC based on a GA are defined as feasible future control action sequences:

$$\text{individual}_j = \{u(k), \ldots, u(k + N_u - 1)\}$$

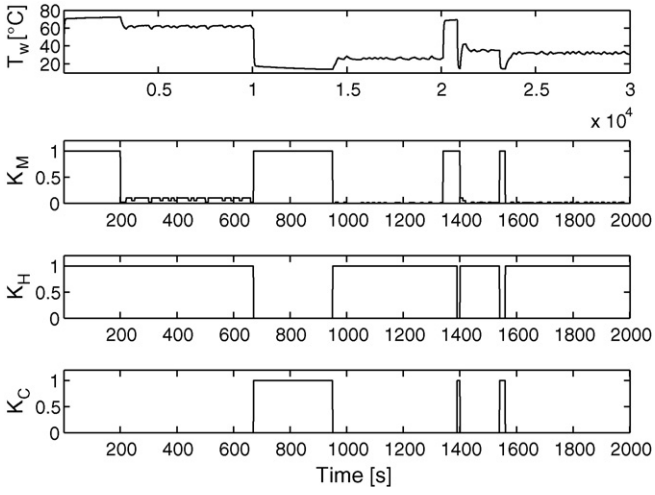

Fig. 6. Results of HFPC based on BB: other system states.

An individual consists of N_u genes and each gene represents one control action.

For simplicity, we consider the following notation for representing the seven possible control actions for the batch reactor:

$$0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, 1 = \begin{bmatrix} 0.01 \\ 0 \\ 1 \end{bmatrix}, 2 = \begin{bmatrix} 0.02 \\ 0 \\ 1 \end{bmatrix}, 3 = \begin{bmatrix} 0.05 \\ 0 \\ 1 \end{bmatrix},$$

$$4 = \begin{bmatrix} 0.1 \\ 0 \\ 1 \end{bmatrix}, 5 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}, 6 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}. \quad (13)$$

Now, the possible control action $u(k+h-1) \in \{0,1, 2, 3, 4, 5, 6\}$, which represents the possible values or the states of the input variables.

The procedure for the HFPC-GA consists of:

(1) Initialize a random population of individuals, i.e., create random-integer feasible solutions of manipulated variables for the hybrid fuzzy predictive control problem. As an example, the size of the population could be seven individuals per generation. Then, as the control horizon is 4, there are 7^4 possible individuals. However, for the GA per generation, the following population is considered:

$$\text{Population}_i = \begin{bmatrix} \text{individual}_1 \\ \text{individual}_2 \\ \text{individual}_3 \\ \text{individual}_4 \\ \text{individual}_5 \\ \text{individual}_6 \\ \text{individual}_7 \end{bmatrix} = \begin{bmatrix} 0163 \\ 2100 \\ 5423 \\ 3634 \\ 4131 \\ 2543 \\ 0301 \end{bmatrix}$$

(2) Evaluate the fitness function for all the initial individuals of the population using Eq. (11). Note that the prediction $\hat{y}(k+h)$ is calculated recursively by using the future control action. In general

$$\hat{y}(k+h) = f(\hat{y}(k+h-1), \ldots, u(k+h-1), \ldots)$$

where f is a non-linear function defined by a hybrid fuzzy model.

(3) Select random parents from the population (different vectors of the future control actions). For example, Individual 1 and Individual 5 are chosen as the parents:

$$\overbrace{\underbrace{\begin{bmatrix} 0 & 1 \end{bmatrix}}_{1A}\underbrace{\begin{bmatrix} 6 & 3 \end{bmatrix}}_{1B}}^{individual_1} \qquad \overbrace{\underbrace{\begin{bmatrix} 4 & 1 \end{bmatrix}}_{5A}\underbrace{\begin{bmatrix} 3 & 1 \end{bmatrix}}_{5B}}^{individual_5}$$

(4) Apply crossover and mutation to the parents in order to generate an offspring.

After the crossover step

$$\underbrace{\begin{bmatrix} 0 & 1 \end{bmatrix}}_{1A}\underbrace{\begin{bmatrix} 3 & 1 \end{bmatrix}}_{5B} \qquad \underbrace{\begin{bmatrix} 4 & 1 \end{bmatrix}}_{5A}\underbrace{\begin{bmatrix} 6 & 3 \end{bmatrix}}_{1B}$$

After the mutation step

$$\overbrace{\begin{bmatrix} 0 & 1 & 2 & 1 \end{bmatrix}}^{New\ Individual_1} \qquad \overbrace{\begin{bmatrix} 0 & 1 & 6 & 3 \end{bmatrix}}^{New\ Individual_2}$$

↑ ↑

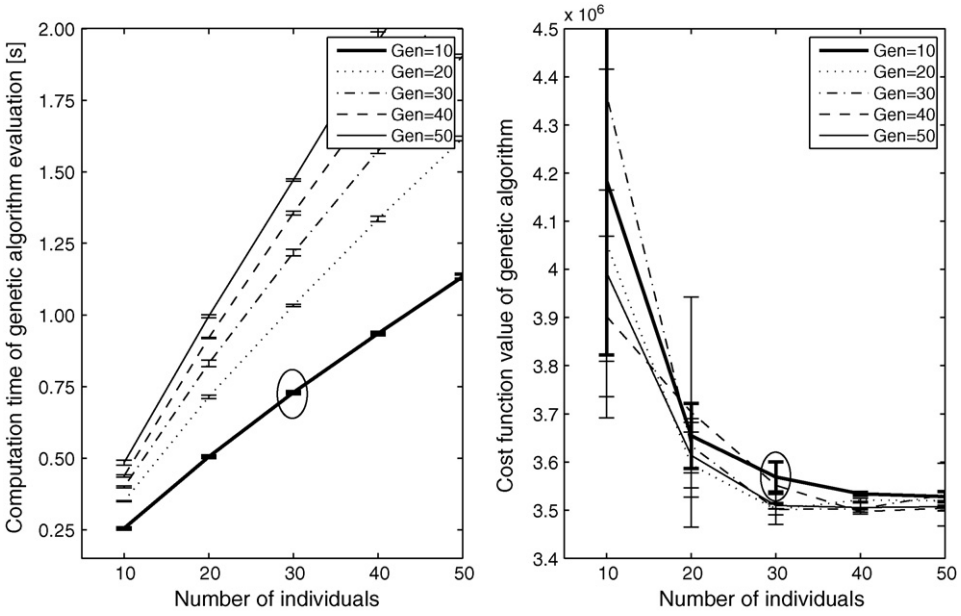

Fig. 7. Computation time and cost function, $N = 4$.

(5) Evaluate the fitness given by the objective function (11) of all the individuals of the offspring population.
(6) Select the best individuals according to the objective function.
(7) Replace the weakest individuals from the previous generation with the strongest individuals of the new generation selected in step 6.
(8) If the objective function value reaches the defined tolerance or the maximum generation number is reached (stopping criteria), then stop. Otherwise, go to step 2.

The genetic algorithm approach in the HFPC-GA provides a sub-optimal discrete control law close to the optimal one. The tuning parameters of the GA method are the number of individuals, the number of generations, the crossover probability, the mutation probability and the stopping criteria.

Figs. 7–9 present the computation time of the genetic algorithm evaluation, the value of the objective function as a function of the number of individuals and the number of generations for different prediction and control horizons ($N_u = N_y = N = 4$, 5 and 6). Based on these figures, and considering a reasonable trade off between accuracy and computational effort, 10 generations with 30 individuals are selected for $N = 4$, 30 generations with 30 individuals for $N = 5$ and 30 generations with 30 individuals for $N = 6$.

The computation time of the HFPC-GA algorithm evaluation is linearly dependent on the generation number and its slope slightly increases with the number of individuals.

For $N = 4$ with 10 generations and 30 individuals (Fig. 7), the computation time was approximately 0.75 s. For $N = 5$ with 30 generations and 30 individuals (Fig. 8), the computation time was approximately 2 s. Finally, for $N = 6$ with 30 generations and 30

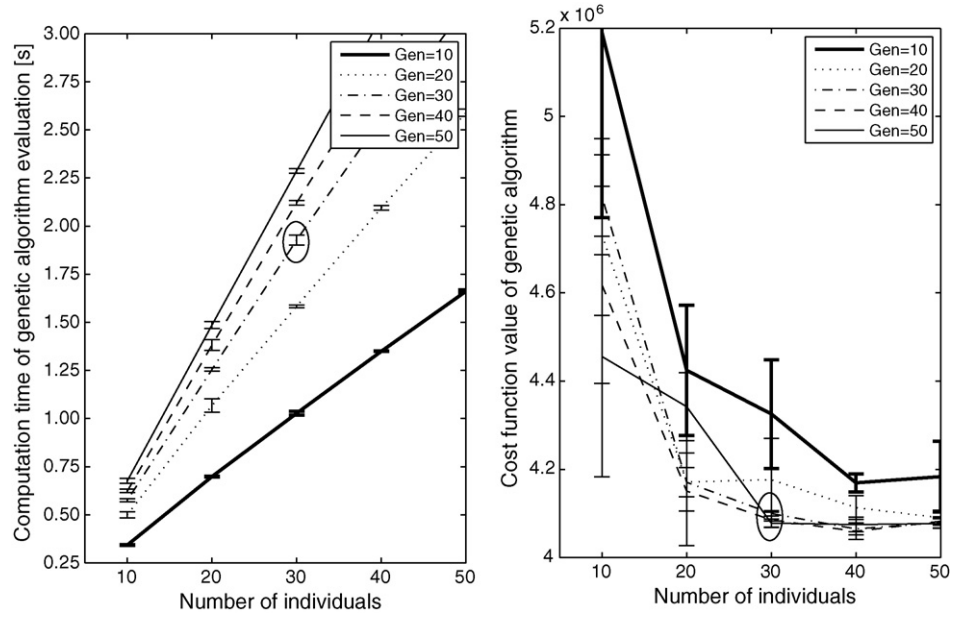

Fig. 8. Computation time and cost function, $N = 5$.

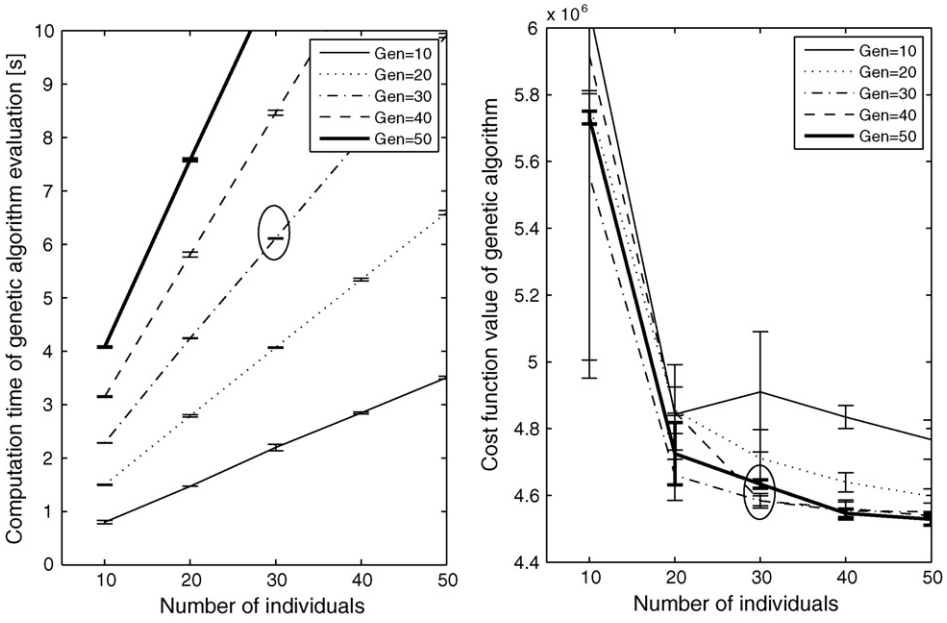

Fig. 9. Computation time and cost function, $N = 6$.

individuals (Fig. 9), the computation time was approximately 6 s. The computation time is smaller than the sampling time, which equals $T_s = 10$ s. This allows the use of the proposed HFPC-GA control strategies in real-time control problems.

The results of the experiment are shown in Figs. 10 and 11. These results are very similar to the ones obtained with the HFPC-BB (see Figs. 5 and 6).

Figs. 12–14 show the normalized computation time for both the HFPC-BB and HFPC-GA for $N = 4$, 5 and 6 respectively. Normalized computation time corresponds to the total time expended on solving the optimization problem divided by the sampling time $T_s = 10$ s. In the figures, k represents every instant when a control action is taken (instant k occurs at $150 \cdot k$ seconds).

The computation time in the case of the GA remains constant during the whole simulation. On the other hand, in the case of the BB the computation time varies significantly with set-point changes. This is the main advantage of the HFPC-GA, and makes it usable in a real-time implementation. The HFPC-GA provides solutions within a bounded computation time that is less than the sampling time. Note that the HFPC-BB with $N = 5$ and 6 cannot be used for solving the control problem in real-time, because in the case when the reference signal changes, the computation time becomes longer than the sampling time (i.e. normalized computation time is major than one). For a real-time problem, the normalized computation time should be less than one, at every instant k.

4.3. Comparison

Tables 1–3 show the computation time per algorithm iteration and the objective function values using Branch and Bound (HFPC-BB), the genetic algorithm (HFPC-GA) and, in addition, explicit enumeration (HFPC-EE).

From Tables 1–3 we obtained equal mean values for the objective function of BB and EE, and also a similar value was provided

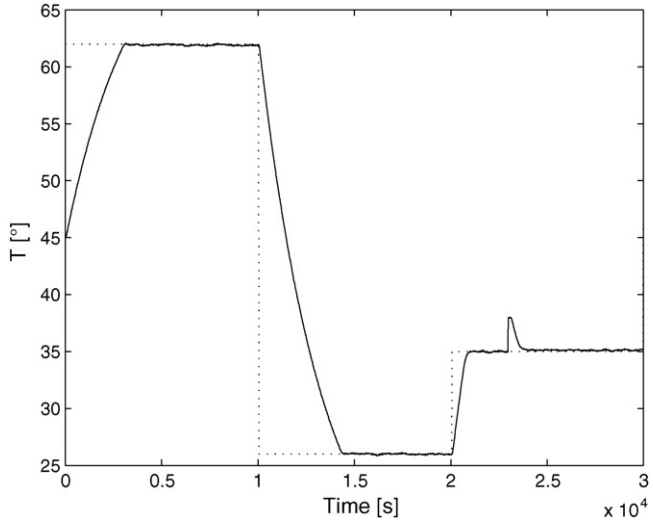

Fig. 10. Results HFPC based on GA: core temperature T (solid line) and reference temperature T_{ref} (dotted line).

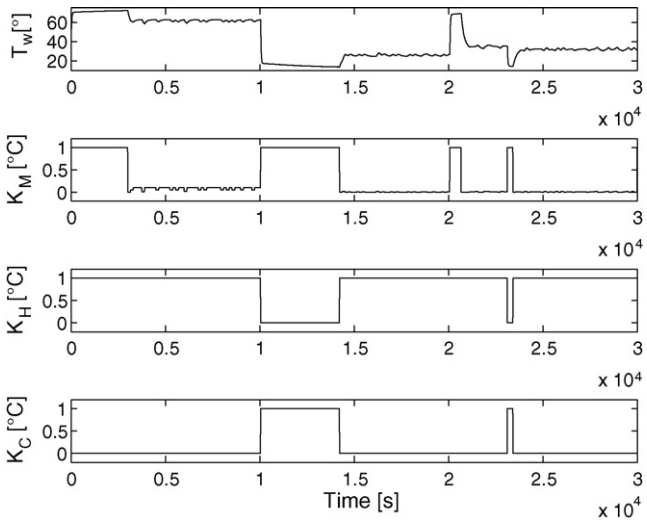

Fig. 11. Results HFPC based on GA: other system states.

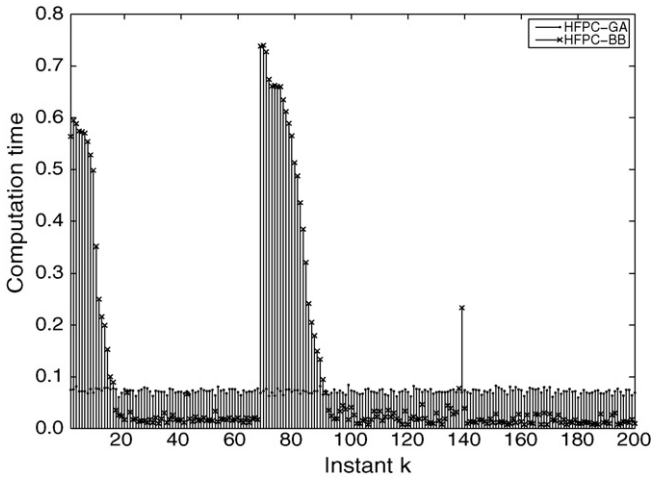

Fig. 12. Normalized computation time, $N = 4$.

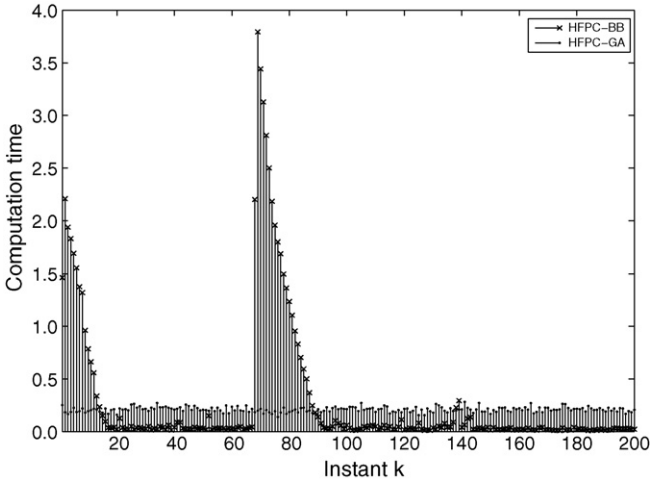

Fig. 13. Normalized computation time, $N = 5$.

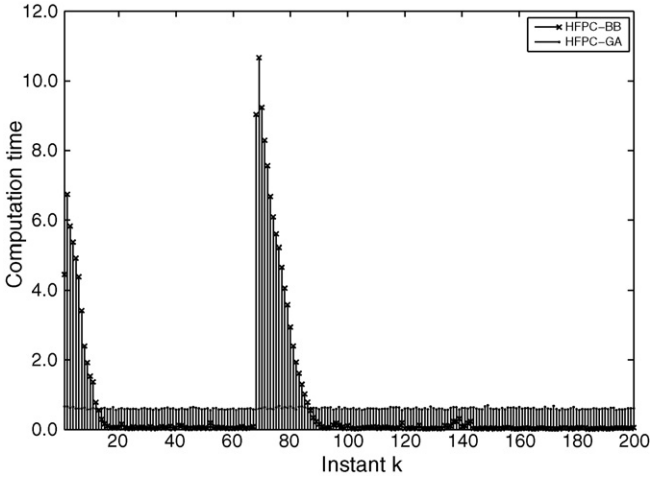

Fig. 14. Normalized computation time, $N = 6$.

by the GA. Thus, the three proposed optimization algorithms allow us to solve the HFPC strategy efficiently in terms of objective function. However, in terms of computation time, there are significant differences between them. The EE provides the global optimum at each instant k; however, the long computation time required does not allow us to ensure a real-time implementation. Regarding the HFPC-BB and HFPC-GA strategies, the mean computation time varies between 1.11 and 7.35 s and between 0.72 and 6.1 s, respectively. Therefore, a computation time saving of approximately 25% is obtained when using the GA in comparison with the BB.

Although the HFPC-BB returns an optimal solution at each instant k, the overall behavior of the controlled plant is practically identical to the HFPC-GA, which provides sub-optimal results. Furthermore, the BB approach does not require any parameter tuning. However, the GA ensures a steady and bounded computation time for each control sampling time, which is critical in real-time applications (see Figs. 12–14).

Table 1
Comparison between BB, GA, and EE, $N = 4$

	Mean time (s)	Standard time deviation (s)	Mean J	Standard deviation J
BB	1.11	0.19	3,401,524	0
GA (30,10)	0.72	0.04	3,494,511	25342
EE	17.47	0.38	3,401,524	0

Table 2
Comparison between BB, GA, and EE, $N = 5$

	Mean time (s)	Standard time deviation (s)	Mean J	Standard deviation J
BB	2.97	0.52	4,035,237	0
GA (30,30)	1.93	0.09	4,065,722	6054
EE	158.1	1.24	4,035,237	0

Table 3
Comparison between BB, GA, and EE, $N = 6$

	Mean time (s)	Standard time deviation (s)	Mean J	Standard deviation J
BB	7.35	0.93	4,406,738	0
GA (30,30)	6.1	0.11	4,432,356	11463
EE	–	–	–	–

5. Conclusion

The proposed Hybrid Predictive Control strategy allows us to regulate the temperature of a batch reactor, minimizing both the trajectory error and the control energy.

The HFPC-BB and HFPC-GA provide very similar behavior in terms of accuracy, but a 25% computation time saving is obtained by using the HFPC-GA. Moreover, for a longer prediction horizon the HFPC-GA can obtain good solutions in a shorter computation time.

In this study the HFPC-GA is presented as a heuristic, systematic and efficient algorithm that allows us to solve NP-hard problems with the HFPC strategy.

Future work will focus on extending the proposed HFPC-GA to solve predictive control with both discrete and continuous manipulated variables.

Acknowledgement

This work was supported in part by the Ministry of Science, Higher Education and Technology of the Republic of Slovenia and by Fondecyt grants 1061156 (Chile) and 7070293 (Chile-Slovenia).

References

Babuska, R. (1998). *Fuzzy modelling for control*. KAP.

Babuska, R., Sousa, J., & Verbruggen, H. (1999). Predictive control of nonlinear systems based on fuzzy and neural models. In *European control conference* (p. 667).
Back, T., Fogel, D., & Michalewics, Z. (Eds.). (2000). *Advanced algorithms and operators*. Bristol, UK: Institute of Physics Publishing.
Baotic, M., Christophersen, F., & Morari, M. (2003). A new algorithm for constrained finite time optimal control of hybrid systems with a linear performance index. In *European control conference*. UK: University of Cambridge.
Beccuti, A. G., Geyer, T., & Morari, M. (2003). Temporal Lagrangian decomposition of model predictive control for hybrid systems. In *European control conference, IEEE* (pp. 2509–2514).
Bemporad, A., Giovanardi, L., & Torrisi, F. D. (2000). Performance driven reachability analysis for optimal scheduling and control of hybrid systems. In *Proc. 39th IEEE conf. decision and control* (pp. 969–974).
Bemporad, A., & Morari, M. (1999). Control of systems integrating logic, dynamics and constraints. *Automatica*, *35*(3), 407–427.
Bemporad, A., Borrelli, F., & Morari, M. (2002). On the optimal control law for linear discrete time hybrid systems. *Hybrid Systems: Computation and Control: Lecture Notes in Computer Science*, *2289*, 105–119.
Bemporad, A., Heemels, W., & Schutter, B. D. (2002). On hybrid systems and closed-loop MPC system. *IEEE Transactions on Automatic Control*, *47*(5), 863–869.
Borrelli, F. (2003). Constrained optimal control of linear and hybrid systems, Lecture Notes in Control and Information Sciences, 290(18).
Borrelli, F., Baotic, M., Bemporad, A., & Morari, M. (2003). An efficient algorithm for computing the state feedback solution to optimal control of discrete time hybrid systems. In *American control conference* (pp. 4717–4722).
Camacho, E. F., & Bordons, C. (1998). *Model predictive control, advanced textbooks in control and signal processing*. London: Springer-Verlag.
Coelho, J. P., de Moura Oliveira, P. B., & Cunha, J. B. (2005). Greenhouse air temperature predictive control using the particle swarm optimisation algorithm. *Computers and Electronics in Agriculture*, *49*, 330–344.
Coello, C. (2002). Theoretical and numerical constraint handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, *191*(11–12), 1245–1287.
Espinosa, J., & Vandewalle, J. (1998). Predictive control using fuzzy models applied to a steam generating unit. In *3rd international FLINS workshop on fuzzy logic and intelligent technologies for nuclear science industry* (pp. 151–160).
Espinosa, J., & Vandewalle, J. (1999). Predictive control using fuzzy models—Comparative study. In *European control conference, EUCA* (p. 273).
Espinosa, J., Vandewalle, J., & Wertz, V. (2005). *Fuzzy logic, identification and predictive control*. Spriger-Verlag.
Hadjili, M., & Wertz, V. (1999). Generalized predictive control using Takagi–Sugeno fuzzy models. In *IEEE international symposium on intelligent control, intelligent systems & semiotics, IEEE* (pp. 405–410).
Karer, G., Mušič, G., Škrjanc, I., & Zupančič, B. (2007a). Hybrid fuzzy model-based predictive control of temperature in a batch reactor. *Computers and Chemical Engineering*, *31*, 1552–1564.
Karer, G., Mušič, G., Škrjanc, I., & Zupančič, B. (2007b). Hybrid fuzzy modelling for model predictive control. *Journal of Intelligent and Robotic Systems*, *50*(3), 297–319.
Karer, G., Škrjanc, I., & Zupančič, B. (in press). Self-adaptive predictive functional control of the temperature in an exothermic batch reactor, Chemical Engineering and Processing. doi:10.1016/j.cep.2008.01.015.
Kennedy, J., & Eberhart, R. (2001). *Swarm intelligence*. Morgan Kaufmann Publishers.
Kim, J., & Huh, U. (1998). Fuzzy model based predictive control. In *IEEE international conference on fuzzy systems, IEEE* (pp. 405–409).
Kimbrough, S., Koehler, G., Lu, M., & Wood, D. (2008). On a feasible–infeasible two-population (fi-2pop) genetic algorithm for constrained optimization: Distance tracing and no free lunch. *European Journal of Operational Research*, *190*(2–16), 310–327.
Maciejowski, J. M. (2002). *Predictive control: With constraints*. Harlow: Prentice Hall.
Mahfouf, M., Kandiah, S., & Linkens, D. A. (2002). Fuzzy model-based predictive control using an ARX structure with feedforward. *Fuzzy Sets and Systems*, *125*, 39–59.
Man, K., Tang, K., & Kwong, S. (1998). *Genetic algorithms, concepts and designs*. Springer-Verlag.
Michalewicz, Z. (1995). Do not kill unfeasible individuals. In Dabrowski, Michalewicz, & Ras (Eds.), *Proceedings of the fourth intelligent information systems workshop, (IIS'95)* (pp. 110–123).
Michalewicz, Z., & Nazhiyath, G. (1995). Genocop iii: a co-evolutionary algorithm for numerical optimization problems with nonlinear constraints. In *IEEE international conference on evolutionary computation, Vol. 2* (pp. 647–651).
Nounou, H., & Passino, K. (1999). Fuzzy model predictive control: Techniques, stability issues, and examples. In *IEEE international symposium on intelligent control, intelligent systems & semiotics, IEEE* (pp. 423–428).
Núñez, A., Sáez, D., Oblak, S., & Škrjanc, I. (2006). Hybrid predictive control based on fuzzy model. In *IEEE world congress on computational intelligence, IEEE* (pp. 9079–9085).
Potočnik, B., Mušič, G., & Zupančič, B. (2004). Model predictive control of systems with discrete inputs. In *12th proceedings IEEE mediterranean electrotechnical conference* (pp. 383–386).
Potočnik, B., Mušič, G., & Zupančič, B. (2005). Model predictive control of discrete time hybrid systems with discrete inputs. *ISA Transactions*, *44*(2), 199–211.
Roubos, J., Babuska, R., Bruijn, P., & Verbruggen, H. (1998). Predictive control by local linearization of a Takagi–Sugeno fuzzy model. In *Proceedings of the IEEE international conference on fuzzy systems* (pp. 37–42).
Sarimveis, H., & Bafas, G. (2003). Fuzzy model predictive control of non-linear processes using genetic algorithms. *Fuzzy Sets and Systems*, *139*, 59–80.
Shin, S. C., & Park, S. B. (1998). GA-based predictive control for nonlinear processes. *IEEE Electronics Letters*, *34*(20), 1980–1981.
Škrjanc, I. (2008). Self-adaptive supervisory predictive functional control of a hybrid semi-batch reactor with constraints. *Chemical Engineering Journal*, *136*(2–3), 312–319.
Škrjanc, I., Blažič, S., & Agamenonni, O. (2005). Identification of dynamical systems with a robust interval fuzzy model. *Automatica*, *41*, 327–332.
Slupphaug, O., & Foss, B. (1997). Model predictive control for a class of hybrid systems. In *European control conference, EUCA*
Slupphaug, O., Vada, J., & Foss, B. (1997). MPC in systems with continuous and discrete control inputs. In *American control conference*.
Solis, J., Sáez, D., & Estévez, P. (2006). Particle swarm optimization-based fuzzy predictive control strategy. In *IEEE world congress on computational intelligence, IEEE* (pp. 8525–8530).
Thomas, J., Dumur, D., & Buisson, J. (2004). Predictive control of hybrid systems under a multi-MLD formalism with state space polyhedral partition. In *American control conference*.
Wang, X., & Xiao, J. (2005). PSO-based model preditive control for nonlinear processes. *Lecture Notes in Computer Science*, *3611*, 196–203.
Woolley, I., Kambhampati, C., Sandoz, D., & Warwick, K. (1998). Intelligent control toolkit for an advanced control system. In *UKACC international conference on control, IEEE* (pp. 445–450).