Learning Near-optimal Decision Rules for Energy Efficient Building Control

Alexander Domahidi[1], Fabian Ullmann[1], Manfred Morari[1] and Colin N. Jones[2]

Abstract— **Recent studies suggest that advanced optimization based control methods such as model predictive control (MPC) can increase energy efficiency of buildings. However, adoption of these methods by industry is still slow, as building operators are used to working with simple controllers based on intuitive decision rules that can be tuned easily on-site. In this paper, we suggest a synthesis procedure for rule based controllers that extracts prevalent information from simulation data with MPC controllers to construct a set of human readable rules while preserving much of the control performance. The method is based on the ADABOOST algorithm from the field of machine learning. We focus on learning binary decisions, considering also the ranking and selection of measurements on which the decision rules are based. We show that this feature selection is useful for both complexity reduction and decreasing investment costs by pruning unnecessary sensors. The proposed method is evaluated in simulation for six different case studies and is shown to maintain the high performance of MPC despite the tremendous reduction in complexity.**

I. INTRODUCTION

About 20-40% of total primary energy usage is spent for the heating, ventilation and cooling (HVAC) of commercial buildings and private housing [1]. While currently most buildings are equipped, if at all, with simple automatic control systems, many studies have indicated the merits of advanced building automation over current state of the art control practice in terms of energy usage and comfort regulation, see e.g. [2]–[5] and the references therein.

Particularly the field of model predictive control (MPC) offers a systematic framework for optimal operation of buildings. Incorporating a model of the building, these controllers can combine measurements available through on-site sensors with weather forecast data to control the building such that energy consumption is minimized while respecting specifications (constraints) on occupancy comfort [6], [7]. These advanced controllers generally outperform current state of the art solutions and thus have a huge potential in reducing green house gas emissions when widely applied.

However, the field only slowly adapts to these advanced optimization based control strategies mainly due to two practical reasons. First, the resulting optimization problem, which needs to be re-solved after new measurements are available, requires substantial IT infrastructure both in terms of hardware and software, and particularly, the latter comes with subtle issues such as licensing and maintenance. Second, commission engineers are not trained, and will not be in the foreseeable future, to set up complex control systems, tune them and respond adequately to eventual malfunctions. Contrary to the petrochemical industry, where optimization based control of large scale dynamic systems started in the early 1970s, a building is usually operated with no on-site engineers carefully monitoring and supervising the correct functioning of the employed control system.

It therefore remains a major challenge to derive control schemes which allow both an *energy efficient* operation of the building and *simple implementation* on the controller side, with concise computer code and intuitive parameters for tuning. In this work, we present a framework which allows to extract near-optimal decision rules for binary controlled variables *automatically* from simulation data generated with advanced control schemes such as MPC, and thereby to significantly decrease the complexity of implementation. The resulting controller code can be run on any existing embedded controller hardware. Furthermore, the synthesized set of rules is readable by humans and can be easily adjusted on-site if the controller behavior is unsatisfactory.

Moreover, the proposed scheme automatically provides a ranking of the most important measurements needed for accurate control by pruning sensors or data streams which contain information irrelevant to the decisions being made. This process of *feature selection* significantly saves investment costs, as it determines the most important sensors. We show that near-optimal control is achieved with only a small number of sensors and weather predictions, and that the extracted rules can be in fact meaningful in practice. Note that, as with other model based approaches, the practical performance of the method will be limited by the accuracy of the simulation model.

Linear interpolation of MPC control laws for *continuous* inputs is the subject of [8]. Rule extraction for binary inputs in a similar context has been considered in [9] with a logistic regression model. Here, we propose to use a boosting algorithm (ADABOOST, [10]) from the field of *machine learning*. One very successful application of ADABOOST is, for instance, face detection [11].

The paper is organized as follows. In Section II, we outline the MPC formulation to be approximated. Next, we describe ADABOOST along with feature selection in Section III. The main contribution of the paper is Section IV, presenting simulation studies with rule based controllers synthesized by ADABOOST, which are shown to have a near-optimal performance despite a significant reduction in complexity.

[1] Automatic Control Laboratory, Department of Information Technology and Electrical Engineering, ETH Zurich, 8092 Zurich, Switzerland. domahida|ullmannf|morari@ee.ethz.ch

[2] Automatic Control Laboratory, École Polytechnique Fédérale de Lausanne (EPFL), 1015 Lausanne, Switzerland. colin.jones@epfl.ch

II. Hybrid Model Predictive Building Control

A. Hierarchical control structure

In this work, we consider a 2-level hierarchical control scheme that employs a high level supervisory controller (HLC) for setpoint calculation and a low level controller (LLC) for tracking. The HLC computes an optimal plan for a certain prediction horizon subject to parameters such as energy price and comfort constraints, as well as measurements of the current state of the building and weather forecast data. In order to predict the building's behavior, the HLC employs a mathematical model of the building. The resulting setpoint commands are then issued to the LLC, which ensures tracking of the given setpoints despite disturbances such as varying occupancy of the building and imperfect weather forecast. The LLC is in practice usually a proportional-integral-derivative (PID) controller or a rule based controller (RBC) [3, §3]. The LLC is however not the focus of this paper, and good tracking of the setpoints provided by the HLC is assumed in the following. At the next sampling instance of the HLC, the optimization is repeated based on the current state of the system, and new setpoint trajectories are computed. This scheme is repeated in each sampling time, which coined the term *receding horizon control* (RHC) widely used for MPC in literature.

There are two fundamentally different types of control inputs computed by the HLC: discrete ones, which are referred to as *binary decisions* in the rest of the paper, and continuous ones. Examples for binary decisions used throughout the paper are *energy recovery* and *free cooling* systems. Energy recovery controls mechanical ventilation and heat exchangers that extract energy from exhaust air, while free cooling controls the amount of available chilled water generated with a wet cooling tower. Both systems have two modes of operation, `LOAD` and `UNLOAD`, for increasing and decreasing the thermal energy stored in the building. The HLC decides upon these modes and issues the commands to the LLC. An example for continuous inputs is the blind position.

B. Hybrid model predictive control formulation

In this section, we describe the main parts of our hybrid model predictive control formulation for the high-level building controller. Details are given in [12].

1) Dynamic building model: We model the building as one zone using a resistance-capacitance network model, which is common to obtain a mathematical description of the building dynamics [3], [5], [7]. The resulting discrete-time model is *bilinear* (cf. [3, §5]):

$$\begin{aligned} x^+ &= Ax + B_u u + B_v v + \sum_{j=1}^{n_u} \left(B_{vu_j} v + B_{xu_j} x \right) u_j \\ y &= Cx + D_u u + D_v v + \sum_{j=1}^{n_u} D_{vu_j} v \, u_j \end{aligned} \tag{1}$$

where $x \in \mathbb{R}^{n_x}$ is the current state of the system, $x^+ \in \mathbb{R}^{n_x}$ the successor state at the next sampling time, $u = [u_c^T \ d^T]^T \in \mathbb{R}^{n_{u,c}} \times \{0,1\}^{n_d}$ with $n_u = n_{u,c} + n_d$ denotes the continuous input variables u_c and binary input variables d, respectively, and $v \in \mathbb{R}^{n_v}$ is the disturbance (weather and occupancy) acting on the system. The matrices A, B_u, B_v, B_{vu_j}, B_{xu_j} C, D_u, D_v, D_{vu_j} of corresponding size are obtained in the modeling process. We linearize around the current operating point to obtain the affine prediction model

$$\begin{aligned} x_{n+1} &= \tilde{A} x_n + \tilde{B}_c u_{c,n} + \tilde{B}_d d_n + \tilde{B}_v v_n + \tilde{p} \\ y_n &= \tilde{C} x_n + \tilde{D}_c u_{c,n} + \tilde{D}_d d_n + \tilde{D}_v v_n + \tilde{q} \end{aligned} \tag{2}$$

for $n = 0, \ldots, N-1$, where N is the prediction horizon. The output trajectory $\boldsymbol{y} := [y_0, \ldots, y_{N-1}]$ is a function of the current state $x \equiv x_0$, the input trajectories $\boldsymbol{u_c} := [u_{c,0}, \ldots, u_{c,N-1}]$ and $\boldsymbol{d} := [d_0, \ldots, d_{N-1}]$ and the disturbance trajectory $\boldsymbol{v} := [v_0, \ldots, v_{N-1}]$.

2) Constraints: MPC inherently supports system constraints. We employ input constraints of the form

$$H_u u_{c,n} \leq h_u \qquad n = 0, \ldots, N-1 \ , \tag{3a}$$

$$d_n \in \{0,1\}^{n_d} \qquad n = 0, \ldots, N-1 \ , \tag{3b}$$

while *soft constraints* are used for the output,

$$H_y y_n \leq h_y + s_n, \quad s_n \geq 0, \quad n = 0, \ldots, N-1 \ , \tag{3c}$$

in order to ensure feasibility of the problem at all times. In (3c), slack variables s_n are used to measure the violation of occupant comfort specifications, for example minimal room temperature or maximum admissible CO_2 level. Inequality (3a) models actuator limitations such as minimum and maximum blind positions.

3) Cost function: The performance index reflects a trade-off among various control objectives. In building control, the cost function is usually an economic performance index such as the cost of non-renewable primary energy (NRPE) usage. In this paper, we consider the following quadratic performance index to be minimized:

$$J\left(x, \boldsymbol{v}, \boldsymbol{u_c}, \boldsymbol{d}, \boldsymbol{s}\right) := \sum_{n=0}^{N-1} c_c^T u_{c,n} + c_d^T d_n + s_n^T S s_n \ , \tag{4}$$

with c_c and c_d being the cost weights for continuous and discrete inputs, respectively. The quadratic penalty term on s with $S \succ 0$ allows one to trade off comfort constraint violations vs. NRPE usage.

4) Hybrid MPC – Summary: We now summarize the finite time constrained optimal control problem, which the HLC has to solve at every time instant. The input trajectories $\boldsymbol{u_c^*}$ and $\boldsymbol{d^*}$ are determined by minimizing (4) subject to an *assumed* disturbance trajectory $\hat{\boldsymbol{v}}$, e.g. a weather forecast profile and occupancy predictions:

$$\begin{aligned} J^*(x, \hat{\boldsymbol{v}}) := \min_{\boldsymbol{u_c}, \boldsymbol{d}} \ & J(x, \hat{\boldsymbol{v}}, \boldsymbol{u_c}, \boldsymbol{d}, \boldsymbol{s}) \\ \text{s.t.} \quad & x_0 = x, \\ & (2), (3) \end{aligned} \tag{5}$$

Due to the integrality constraint (3b), problem (5) is in the class of *hybrid MPC (HMPC)* problems.

Remark II.1 *Problem* (5) *neglects stochasticity in the system and is by no means the only way to formulate a sensible optimization problem for the control task at hand, cf. [13] for stochastic formulations for instance. However, the proposed approach of rules learning can be applied to any other MPC formulation involving binary decision variables.*

C. Problem complexity and suboptimal controllers

Any form of (5) becomes essentially a hard problem due to the combinatorial nature of the discrete variables involved [14]. Such problems can take a very long time to solve even on powerful computers, as it belongs to the class of *mixed-integer quadratic programs* (MIQPs), for which currently no polynomial time algorithm exists. However, while the time to solve (5), together with the aforementioned practical problems, is prohibitive for an on-line implementation, offline simulation can be carried out using software tools with advanced heuristics such as CPLEX[1], which enable the optimal solution of the problem at hand in acceptable development and run time.

The main underlying idea of this paper is therefore to *approximate* the optimal decisions recorded in simulation such that the resulting controller as good as possible reproduces the behavior of the HMPC. Latter is then replaced by the approximate or *suboptimal* controller in the actual implementation. We call the process of approximating a controller from data *learning*.

Remark II.2 *Since it is the binary decisions which make problem* (5) *hard to solve online, we focus in this work on learning binary decisions only. Learning of continuous controllers has been investigated by the authors in [15].*

Once the binary decisions have been made, i.e. the value of the binary variables in (5) has been determined, the resulting optimization problem is a quadratic program (QP), for which very efficient methods exists that have been showed to be applicable on embedded control systems [16], [17]. In the following, we describe one method to synthesize simple decision rules representing an approximate controller for binary decisions in (5).

III. Learning: Classification using AdaBoost

Assume that M data points from simulation,

$$\chi_i := \left[x_i^T, \hat{\boldsymbol{v}}_i^T\right] \in \mathcal{P} := \mathcal{X} \times \mathcal{V} \ , \tag{6}$$

as well as the corresponding optimal binary solutions $d^*_{0,i}$ to the MPC problem (5), are given. The index 0 denotes that we are considering the approximation of the first control move only, since the other computed inputs are actually never applied to the system in a receding horizon framework. Instead of representing these decisions as zeros and ones, we map a zero to minus one, i.e.

$$z_i = 2d^*_{0,i} - \mathbf{1} \in \{-1, 1\}^{n_d} \ , \tag{7}$$

to match common learning algorithm formulations. We denote the tuple (χ_i, z_i) as a *sample*.

The problem of learning a controller is then as follows: We would like to infer n_d functions $g_m : \mathcal{P} \to \{-1, 1\}$ from the data with minimal error for any *unseen* sample χ. This is generally a difficult problem as the optimization would operate in an infinite dimensional *function* space, which makes the problem fundamentally more difficult than finding a vector in an Euclidian space, for example. Hence one has to restrict to approximations of the learning problem that parametrize the learned function by a finite dimensional parameter vector [18].

Next, we are going to describe AdaBoost, which synthesizes special types of discriminant functions.

A. Adaptive Boosting

AdaBoost is a classification algorithm which emerged with the seminal paper by Freund and Schapire, [10]. The underlying idea is that the learning problem might be too complex for a single learning machine. Instead, one can *boost* the performance of many simple learning machines, so-called *weak learners*, by combining them in an appropriate way to a single powerful learning machine. Each weak learner has to do only slightly better than random for convergence of the algorithm, and it can be shown that the algorithm minimizes an exponential loss function on wrong decisions.

The weak learners are (simple) discriminant functions $h_j : \mathcal{P} \to \{-1, 1\}$, which can be evaluated very fast but usually have mediocre performance. The final discriminant function is a linear combination of L weak learners,

$$g(\chi) = \sum_{j=1}^{L} a_j h_j(\chi) \ , \tag{8}$$

which yields a strong classifier. The binary decision is obtained by taking the sign of the discriminant function and shifting back,

$$d(\chi) = \frac{1}{2}\left[\text{sign}\left(g(\chi)\right) + 1\right] \ . \tag{9}$$

An example for such a classifier is given in Fig. 1.

1) Training procedure: Training an AdaBoost classifier is an iterative process, and the following assumption has to be met by the weak learners.

Assumption III.1 *The training procedure of the j^{th} weak learner returns a function h_j and a training error rate ϵ_j, i.e. the fraction of misclassified points, given the training samples (χ_i, z_i) and* importance weights *$w_{i,j}$, $i = 1, \dots, M$, that are used to give a relative relevance to the sample points.*

At the start of the algorithm, all points are equally important, i.e. $w_{i,1} = 1 \ \forall i = 1, \dots, M$. After training the j^{th} weak learner, its weight a_j in (8) is computed according to

$$a_j(\epsilon_j) = \frac{1}{2} \log\left(\frac{1 - \epsilon_j}{\epsilon_j}\right) \ . \tag{10}$$

Intuitively, the weighting rule rates weak learners that perform well higher. Next, the importance weights w_i are

[1] www.ibm.com/software/integration/optimization/cplex-optimizer/

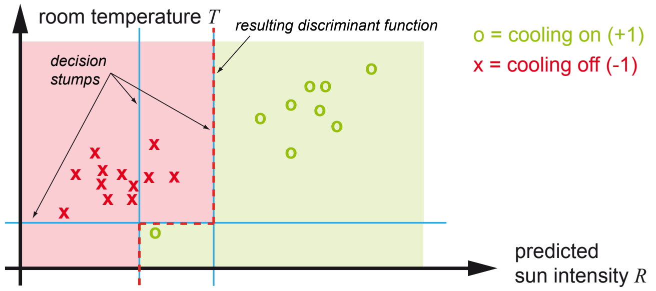

Fig. 1. ADABOOST example. The classifier decides on two *features*, R and T, whether to switch on or off the cooling system. The data points represent exemplary optimal decisions from an MPC controller. The final discriminant function results from the combination of weak learners.

adjusted in order to emphasize wrongly classified data while attenuating correctly classified samples. The data weighting factors for weak learner $j+1$ are given by

$$w_{i,j+1}\left(a_j,\chi_i,z_i\right)=\exp\left(-a_j z_i h_j\left(\chi_i\right)\right) \quad . \qquad (11)$$

Then, the next weak learner $h_{j+1}\left(\chi\right)$ is trained using the reweighted data. This process of weak learner weight computation (10), data reweighting (11) and weak learner training is repeated until L weak learners have been trained.

2) Choice of weak learners: Once the parameters of the weak learners h_j and the corresponding weights a_j have been found, evaluation of the control law boils down to the evaluation of the weak learners. It is therefore desirable to have functions h_j which can be evaluated quickly.

The simplest class of weak learners which work very well with ADABOOST are so-called *decision stumps*. We use these in our case studies.

Definition III.2 (Decision stumps) *A decision stump is defined by*

$$h\left(\chi\right)=sign\left(e_k^T\chi-d\right) \quad , \qquad (12)$$

where e_k is one of the unit vectors. Its parameters in terms of learning are the feature index k *and the* offset d*.*

It turns out that, for our purposes, decision stumps are a very good fit because of the inherent similarity of the resulting controller to current building control practice and because decisions stumps allow a straightforward feature selection. We detail on both aspects in the following.

3) Resulting controller structure: It is worth noting that each decision stump decides on one feature only, which can be viewed as a simple if-then-else rule:

if $\chi_k > d$ **then** $h\left(\chi\right)=+1$ **else** $h\left(\chi\right)=-1$ **end if**

One rule could for example read "if room temperature greater than 24° C then switch on the cooling system". Recall that each weak learner represents such a rule, so combining them with the associated weights results in a very simple majority voting system for the overall decision:

```
vote = 0;
for j = 1 to L do
  if Feature[j] > Threshold[j] then
    vote += weight[j];
  else
    vote -= weight[j];
  end if
end for
```

In the pseudocode above, `Feature[j]` denotes $e_k^T\chi$ and `Threshold[j]` denotes the offset d of the j^{th} weak learner in (12), respectively. Moreover, `weight[j]` is the weight associated to the j^{th} weak learner, a_j in (8). Once the "votes" have been collected, the resulting decision is simply made by evaluating the sign of the variable,

```
if vote > 0 then
  return SWITCH_ON_COOLING
else
  return SWITCH_OFF_COOLING
end if
```

The majority voting system is very similar to state of the art *rule based controllers* (RBCs) [3, §3], where a set of rules is combined to yield a final decision. These rules are usually defined and tuned by experienced building control engineers, often in a long trial and error procedure. In contrast, the proposed rule based controller (8) can be synthesized automatically as a chain of if-then-else rules. Nevertheless, the resulting decision rules are readable and adjustable by humans, and thus allow intuitive insight and manual tuning, should the controller behave unsatisfactorily on-site.

4) Feature selection with ADABOOST*:* The weak learners perform an inherent feature selection by the choice of the feature index k. The weight assigned to the weak learner is therefore a direct indication of how important that very decision on feature k is, and therefore weak learners with small relative weight can be discarded from the overall decision, effectively performing a feature selection using the weights obtained by ADABOOST training.

IV. LEARNING BINARY DECISIONS OF HIGH LEVEL MPC CONTROLLERS: SIMULATION RESULTS

In this section, we present the main results of this paper based on a simulation study. We start by outlining the simulation setup and case studies in Section IV-A. Learning and feature selection are discussed in Section IV-B before presenting closed-loop simulation results with learned rule based controllers in Section IV-C. Figure 2 depicts the particular workflow.

A. Test scenario and data generation

In order to assess the performance of the proposed method, we simulate different case studies using *BACLab*, a MATLAB-based modeling and simulation environment for building climate control developed within the OptiControl[2] project, which focuses on the development of predictive control strategies for building climate control. We consider six test cases with different building types and integrated room automation (IRA) equipment at two different locations in Europe. These test cases have been identified to have an energy savings potential of 9-33% when using predictive control schemes (cf. [3, §7]), while at the same time, they

[2] www.opticontrol.ethz.ch

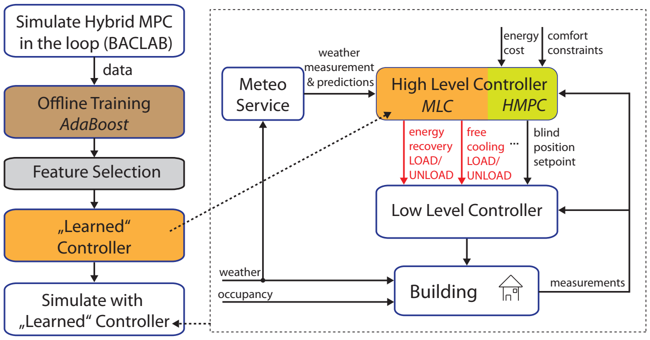

Fig. 2. Simulation strategy used to evaluate the performance of the synthesized rule based controllers. The suboptimal controller is learned using simulation data, overriding the binary decisions of the high level controller.

represent typical buildings across continental Europe. The cases are based on two building types: passive house standard with a window area fraction of 80% on the facade, and houses having a Swiss average insulation level and a low window area fraction of 30%. For more details on the test cases the reader is referred to [12].

We formulate the hybrid MPC controller as a mixed-logic-dynamical (MLD) system using HYSDEL [19] and YALMIP [20], taking into account the dynamics of the building and those of the low level controller [12]. Latter is implemented as a one step MPC controller, which is adjusted according to the modes set by the HLC (cf. [3, §3.3]). Since the continuous part of the building model is bilinear, the optimal input sequence is computed iteratively by solving a sequence of mixed-integer problems (5) and updating the linearization of the dynamics (2) around the current iterate. We use version 10 of CPLEX to solve the MIQP subproblems.

To generate training data for ADABOOST, one year of building operation under the control loop as depicted in Fig. 2 on the right has been simulated with a step size of one hour. In this simulation, the hybrid MPC controller acts as the HLC, employing a prediction horizon of 24 hours. Real weather data from the year 2007 was used to drive the simulation and generate disturbances on the models. The weather measurements were obtained from the *Swiss Federal Office of Meteorology and Climatology (MeteoSwiss)*, while weather predictions needed by the MPC controller were obtained by MeteoSwiss' weather forecast system, COSMO-7 (cf. [3, §6]). During simulation, we recorded state and output measurements as well as the binary inputs generated by the HLC. We were particularly interested in learning the decisions on *energy recovery* and *free cooling*.

B. *Controller approximation by learning & feature selection*

We have synthesized seven different approximate controllers: One for each of the six test cases, denoted as MLC-1 to MLC-6, and one controller which was trained with *all* the data and should serve as a controller for *all* cases. We denote this controller by MLC-A. All controllers were learned using the freely available software package *GML AdaBoost Matlab Toolbox* [21] with decision stumps as weak learners. We have carried out $L = 200$ boosting steps.

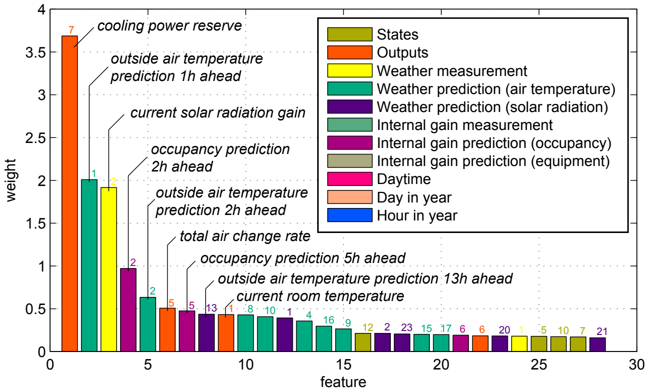

Fig. 3. Feature ranking by accumulated weights for ADABOOST-based controller for case study 4 (energy recovery mode).

The decision of the HLC depends on 126 variables: state and output measurements such as room temperatures, weather predictions for the next 72 hours, occupancy predictions, day and week, time in year etc. In order to limit the complexity of the resulting controllers by relying only on the most important information for the decision to be made, feature selection has been carried out according to the weak learners' accumulated absolute weights, i.e. if two weak learners decide on the same feature, their absolute weights are added (cf. Section III-A.4).

In the following, we present and discuss results for feature selection for MLC-4, the other cases were qualitatively similar. The resulting ranking is shown in Fig. 3, where we have annotated the first nine features. The most important feature, the so-called *cooling power reserve*, is an output of the system indicating the margin of supply air temperature to the minimum temperature of 16° C required by comfort specifications. Other highly ranked features are the *current solar radiation*, *weather and occupancy predictions 1-2 hours ahead* and the *total air change rate*. Latter is the sum of mechanical ventilation and natural convection terms. In the training data for MLC-A, the building type index is included as an additional feature, but it is not ranked very important. We keep the 50 most important out of the 200 weak learners for the closed-loop simulations presented in Section IV-C.

The selection of features seems reasonable from a practical point of view, for both the cooling power reserve and the total air change rate being important system quantities regarding the ability of actuators to change the total amount of thermal energy stored in the building. Moreover, short-term predictions are more important than those further in the future, which seems reasonable regarding the employed receding horizon scheme. We also have found that most numerical values of the thresholds could be justified in practice. For example, the decisions taken on various temperatures use a threshold of about 22° C. This seems reasonable since the minimum room temperature is 21° C according to the specification.

To summarize, feature selection allows one to automat-

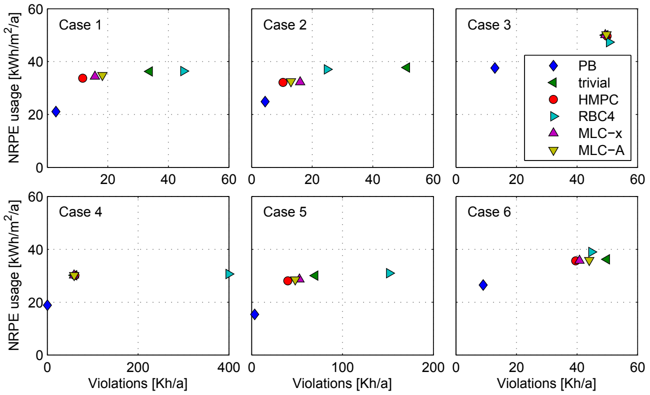

Fig. 4. Closed loop simulation results. MLC-x denotes the individual controllers MLC-1 to MLC-6, applied in the corresponding test case.

ically select important measurements while neglecting less important ones and thereby significantly reduces the control function complexity. The selected features and thresholds in our case study indicate that the synthesized rules are indeed sensible in practice.

C. Closed loop performance

In order to assess the impact of suboptimality of the approximate controllers, we simulate test cases 1-6 with the learned controller overriding the decisions of the hybrid MPC for the two modes *energy recovery* and *free cooling*. All other decisions were obtained from the hybrid MPC. The results are shown in Fig. 4. The horizontal axis counts the comfort constraint violations in Kelvin-hours per year, while the vertical axis depicts the NRPE usage in kWh per year and square meter ground area of the building. If the weather, i.e. the disturbance acting on the system, was known perfectly in advance, the performance bound (PB) would be the best any controller could achieve [6]. The PB therefore serves as a reference point – the closer any controller to the PB point, the better its performance.

We compare the learned controllers MLC-1-6 and MLC-A to other high level controllers, namely the hybrid MPC (HMPC), the "trivial" controller (always `UNLOAD`) and the current state of the art rule based controller (RBC4). We see that, generally, the closed loop performance of the learned controllers is very similar to the hybrid MPC controller, while the "trivial" controller performs significantly worse on most cases. This demonstrates that the learning is indeed effectively capturing the behavior of the optimal controller. All learned controllers perform significantly better than RBC4, which relies on manually designed rules. Moreover, while both RBC4 and HMPC are building-dependent, MLC-A performs equally well on *all* building types *with the same 50 decision rules*. This could significantly simplify the deployment of advanced controllers in practice and shows the effectiveness of the proposed approach.

Acknowledgement

The authors would like to thank Frauke Oldewurtel for discussions on MPC for buildings and Melanie N. Zeilinger for her helpful comments on this manuscript. The research leading to these results has received funding from the EU in FP7 via EMBOCON (ICT-248940).

References

[1] L. Pérez-Lombard, J. Ortiz, and C. Pout, "A review on buildings energy consumption information," *Energy and Buildings*, vol. 40, no. 3, pp. 394 – 398, 2008.

[2] P. J. Lute, *The use of predictions in temperature control in buildings*. Delft University of Technology, Faculty of Mechanical Engineering and Marine Technology, 1992.

[3] D. Gyalistras and M. Gwerder (eds.), "Use of weather and occupancy forecasts for optimal building climate control (opticontrol): Two years progress report," ETH Zurich, Switzerland and Siemens Building Technologies Division, Siemens Switzerland Ltd., Tech. Rep., 2009. [Online]. Available: http://www.opticontrol.ethz.ch/

[4] S. Prívara, J. Široký, L. Ferkl, and J. Cigler, "Model predictive control of a building heating system: The first experience," *Energy and Buildings*, vol. 43, pp. 564 – 572, 2011.

[5] Y. Ma, A. Kelman, A. Daly, and F. Borrelli, "Predictive control for energy efficient buildings with thermal storage: Modeling, stimulation, and experiments," *Control Systems, IEEE*, vol. 32, no. 1, pp. 44 –64, Feb. 2012.

[6] F. Oldewurtel, A. Parisio, C. Jones, D. Gyalistras, M. Gwerder, V. Stauch, B. Lehmann, and M. Morari, "Use of model predictive control and weather forecasts for energy efficient building climate control," *Energy and Buildings*, 2011.

[7] A. Kelman and F. Borrelli, "Bilinear model predictive control of a hvac system using sequential quadratic programming," in *Proc. of the IFAC World Congress*, 2011.

[8] B. Coffey, "Using building simulation and optimization to calculate lookup tables for control," Ph.D. dissertation, UC Berkeley: Center for the Built Environment, 2012.

[9] P. May-Ostendorp, G. P. Henze, C. D. Corbin, B. Rajagopalan, and C. Felsmann, "Model-predictive control of mixed-mode buildings with rule extraction," *Building and Environment*, vol. 46, no. 2, pp. 428 – 437, 2011.

[10] Y. Freund and R. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *Computational Learning Theory*, ser. Lecture Notes in Computer Science, P. Vitányi, Ed. Springer Berlin / Heidelberg, 1995, vol. 904, pp. 23–37.

[11] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, pp. 137–154, 2004.

[12] F. Ullmann, "Embedding predictive control in hierarchical integrated room automation systems," Semester thesis, Automatic Control Laboratory, ETH Zurich, June 2009. [Online]. Available: http://www.opticontrol.ethz.ch

[13] P. J. Goulart, E. C. Kerrigan, and J. M. Maciejowski, "Optimization over state feedback policies for robust control with constraints," *Automatica*, vol. 42, no. 4, pp. 523 – 533, 2006.

[14] M. Garey and D. Johnson, *Computers and intractability*. San Francisco: W. H. Freeman, 1979.

[15] A. Domahidi, M. Zeilinger, M. Morari, and C. Jones, "Learning a feasible and stabilizing explicit model predictive control law by robust optimization," in *Proc. of the 50th Conference on Decision and Control*, Orlando, FL, USA, Dec. 2011, pp. 513–519.

[16] A. Domahidi, A. Zgraggen, M. Zeilinger, M. Morari, and C. Jones, "Efficient interior point methods for multistage problems arising in receding horizon control," in *Proc. of the 51st Conference on Decision and Control*, Maui, Hawaii, USA, Dec. 2012.

[17] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 2, pp. 267 –278, March 2010.

[18] F. Cucker and D.-X. Zhou, *Learning theory: an approximation theory viewpoint*. Cambridge: Cambridge University Press, 2007.

[19] F. Torrisi and A. Bemporad, "Hysdel - a tool for generating computational hybrid models for analysis and synthesis problems," *IEEE Transactions on Control Systems Technology*, vol. 12, no. 2, pp. 235 – 249, March 2004.

[20] J. Löfberg, "Yalmip: A toolbox for modeling and optimization in Matlab," in *Proceedings of the CACSD*, Taipei, Taiwan, 2004. [Online]. Available: http://users.isy.liu.se/johanl/yalmip/

[21] A. Vezhnevets, "GML AdaBoost Toolbox for Matlab," 2010. [Online]. Available: http://www.inf.ethz.ch/personal/vezhneva/index.html