End-to-End Heat-Pump Control Using Continuous Time Stochastic Modelling and Uppaal Stratego

Imran Riaz Hasrat, Peter Gjøl Jensen, Kim Guldstrand Larsen, and Jiří Srba

Department of Computer Science, Aalborg University, Aalborg, Denmark
{imranh, pgj, kgl, srba}@cs.aau.dk

Abstract. Heatpump-based floor-heating systems for domestic heating offer flexibility in energy-consumption patterns, which can be utilized for reducing heating costs—in particular when considering hour-based electricity prices. Such flexibility is hard to exploit via classical Model Predictive Control (MPC), and in addition, MPC requires a priori calibration (i.e. model identification) which is often costly and becomes outdated as the dynamics and use of a building change. We solve these shortcomings by combining recent advancements in stochastic model identification and automatic (near-)optimal controller synthesis. Our method suggests an adaptive model-identification using the tool CTSM-R, and an efficient control synthesis based on Q-learning for Euclidean Markov Decision Processes via Uppaal Stratego. On a virtual Danish family-house from the OpSys project, we demonstrate up to 33% reduction in heating cost while retaining comparable comfort to a standard bang-bang controller. Furthermore, we show the flexibility of our method by computing the Pareto-frontier that visualizes the cost/comfort tradeoff.

Keywords: Model identification · strategy syntheses · heat-pump control · floor heating.

1 Introduction

Space heating is the primary source of energy consumption in residential and commercial buildings, consuming more than 40% of the energy delivered to such facilities [8]. Intelligent control of heating systems promises to reduce this energy use, thereby countering global warming effects and CO2 emissions.

Model Predictive Controllers (MPCs) have been demonstrated as an efficient method for control of domestic heating systems with the potential for both energy and cost reductions [1,18,26]. In the classical setting of MPC, an approximate system model under control is constructed before application and paired with a control objective. In the setting of a domestic heating system, such a model describes heat dynamics of the house and external effects that may impact the temperature of the house, for instance, residential behaviour, outdoor temperature or solar radiation contributing to the temperature of the rooms.

This model also defines the possible actions of the controller along with the impact of these control-actions on the behaviour of the system. When the given model is paired with a control objective (e.g. minimize cost and maximize comfort), (model, objective)-pairing induces an *optimization*-problem. Depending on the formalism used to describe the model and control objective, different *solvers* can be used to obtain the next optimal control action to use. As a result, one can apply the MPC-controller in a tight and periodic loop of "observe, solve, act" to control the system.

This paper targets a floor-heating heat-pump control problem for a standard family house. In the case of domestic heating systems, three significant challenges appear in the classical MPC application:

- the heating dynamics of a house is not known a priori,
- the behaviour and dynamics of a house change over the lifetime of the house,
- the design, tuning and deployment of MPC for learning near-optimal control strategies to minimize a desired objective function is not straightforward.

To address these issues, we propose a framework for model-identification for use with the tool UPPAAL Stratego extending on the concept presented by Larsen et. al. [18]. In particular, we introduce the model estimation into the loop of the regular MPC-control. We employ the tool CTSM-R (continues time stochastic modelling in R) [17] to identify the house model by utilizing the *historical data* of the house. We further propose an online strategy synthesis approach where the controller periodically predicts the control decisions with the current room temperatures and weather forecast knowledge. The controller explores partial state space and learns near-optimal strategies within the given price budget based on the selected learning method. Under the learned strategies, the controller optimizes and computes every decision ahead of time to make it ready to be applied for the next time interval. The main contributions of the method are as follows.

1. Developing data-driven thermal dynamics and constant coefficient estimations using CTSM-R.
2. Modelling a case-house in Stratego using the estimated heat transfer coefficients and thermal dynamics.
3. Employing Stratego MPC to learn near-optimal control strategies for operating the heat-pump.
4. Analysing the efficiency of the Stratego controller to handle the trade-off between heating cost and user comfort.

Related Work: Several domestic heating systems have been discussed in the literature. Vogler-Finck et. al. [26] study the house dynamics where the control objective is given in the restricted form of linear equations. They apply *grey-box* modelling facility from MATLAB System Identification toolbox [20] to identify a model for predictive control. They examine three family houses of different ages and the results witness reduced carbon footprint of heating by MPC optimization based on energy and CO2. However, they use a simple deterministic model for

MPC whereas we apply a stochastic model for MPC with an online strategy synthesis approach. An alternative approach is adopted by Larsen et. al. [18] who utilize the tool Stratego [6,7] to obtain *near-optimal* control-optimization for switch-controlled hybrid stochastic systems. The authors suggest an online and compositional synthesis approach for a family-house floor heating system, focusing purely on maximizing comfort and disregarding energy consumption. In our work, we instead optimize for multiple objectives (comfort and cost) and introduce a model identification approach into the loop of an MPC control.

For large and complex systems, model identification becomes an important aspect of MPC control to identify a relatively simple and reduced-order model to make it practically controllable in real-time. However, Prívara et. al. [22] describe that model identification is one of the main practical obstacles for large scale use of MPC. The *grey-box* modelling is one of the commonly used approaches that require a small data-set for model identification. Ferracuti et. al. [9] and Fonti et. al. [10] utilise low-order *grey-box* models to detect short-term (15min, 1h and 3h horizons) thermal behaviour of a real building while Reynders et. al. [23] used *grey-box* approach to identify reduced-order energy building models. However, these works do not offer a controller synthesis for the identified models. We create a specialised model for a fully automatic heat-pump case where we first identify the model and then use it for strategy synthesis and evaluation.

Vinther et. al. [25] propose *black-box* approach to estimate MPC models from multiple Artificial Neural Network (ANN) techniques with a Genetic Algorithm (GA) to predict the future set-points (for room temperatures) in an existing floor heating system. Furthermore, Nassif et. al. [21] also use ANN with GA for optimisation of a HVAC system. Harasty et. al. [11] apply a differential evolution (DE) algorithm with ANN for MPC control and optimisation of room temperature for the conservation of cultural heritage. These approaches are based on offline learning, but we offer an online learning approach where the control decisions are predicted periodically for the near future by considering the real-time, making them applicable in practice even for long-lasting horizons.

2 Usecase and Method

We demonstrate our approach on a $150m^2$ experimental family house sketched in Figure 1. The house consists of four rooms of different sizes and material properties: *Room 1* is the designated living room with a build-in kitchen, *Room 2* and *Room 4* are bedrooms, and *Room 3* serves as the bathroom with a light concrete floor as opposed to the light wooden floors of the other rooms. The system uses hot water as a means of heat distribution. The house in question has a high-fidelity Dymola model, which has been constructed and compared to a physical model through the OpSys project [15]. As it is standard for existing houses, the heating system has two layers of control: room thermostats (with a fixed mechanical bang/bang controller), and a heat-pump (for which we derive a controller). The model reflects a realistic scenario where an existing building is retrofitted with an intelligent heat-pump. Notice here that each room thermostat

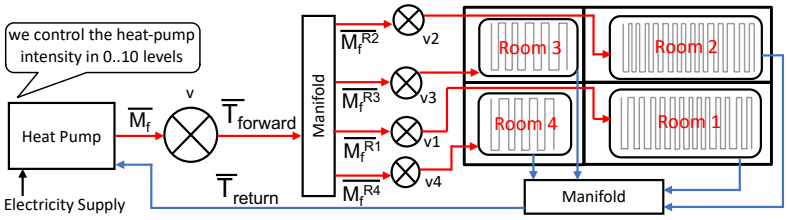


Fig. 1: An Overview of the four-rooms family house equipped with floor-heating.

acts independently of the heat-pump and the other thermostats; it is simply concerned with opening the flow of hot water when the room temperature drops below a certain set point (fixed to 22° in our experiments). This implies that the only control-point of concern is the intensity of the heat-pump.

The system is designed as a closed-loop system with a fixed distribution-key. This means that the ingoing massflow of water ($\overline{M_f}$) must be distributed to the rooms via the manifold. Conventionally the manifold is calibrated s.t. a fixed proportion of water is designated to each room. This distribution-key we denote $\overline{M_f}^{R_x}$ for a room x. The individual room thermostats regulate the binary valves v_x. This implies that the massflow is re-distributed (proportionally) to the remaining open valves if certain valves are shut. Furthermore, we denote by $\overline{T}_{forward}$ and $\overline{T}_{return}$ the forward and return water temperature.

The main purpose of the heat-pump is to balance the difference between the room temperatures $\tilde{T}_r^x$ and a given set point T_g with the cost of heating $cost(\tau) = price_\tau \cdot w_\tau$, which is a time-dependent function of the energy consumption of the heat-pump w_τ (determined directly by the controller) and the market electricity price ($price_\tau$) which varies on an hourly basis and is known 24 hours in advance.

Naturally, a consumer wishes to weight comfort against cost, and we can thus state the optimization criteria and introduce the $0 \leq W_{comf} \leq 1$ weighting factor that allows for such tuning. This allows us to derive the following (W_{comf}-parameterized) fitness function for our controllers for a period τ_0 to τ_n given that the heat-pump settings, energy prices and room temperatures (denoted $\tilde{T}_r^i(\tau)$) are known for the duration for k rooms as:

$$F(\tau_0, \tau_n) = \int_{\tau_0}^{\tau_n} \left(((1 - W_{comf}) \cdot cost(x)) + W_{comf} \cdot \sqrt{\sum_i^k (T_g - \tilde{T}_r^i(x))^2} \right) dx \tag{1}$$

Notice that this function penalizes more significant deviations of temperatures in a squared fashion while the cost increase has a linear impact on fitness.

2.1 Methodological Overview

While a high-fidelity model for Dymola is provided, this model is infeasible for practical experiments both due to its high computational effort caused by the high fidelity and due to licensing issues making massive parallel experiments

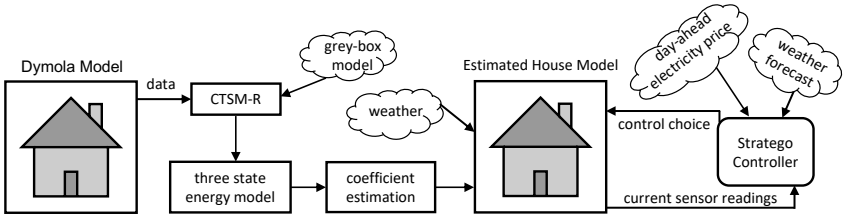


Fig. 2: Estimation and control process using observed data

unfeasible. The overview of our suggested approach can be seen in Figure 2. We initially derive a low-fidelity model of the thermal dynamics of the building via a data-driven *grey-box* model estimation using the CTSM-R software [16]. The CTSM-R library allows finding best-fit parameters for a stochastic model based on samples of a (virtual) house. The specific grey-box formulation is given in Section 3 along with an evaluation of its performance. Notice that the model-identification problem is constructed s.t. the estimation can be computed in a decomposed manner, i.e. a model is computed individually for each room, and the set of models is later recomposed. We do so to avoid an explosion in computational effort from a large set of variables to estimate. While this *grey-box* model yields us the internal heat coefficients (how the energy dissipates between rooms or a heater), it does not capture the heat transfer from the hot water flowing through the pipes. Therefore, we derive a model capturing the relationship between internal temperature changes and the drop in the heat-carrying water, yielding us the return water temperature ($\overline{T}_{return}$).

Given these two thermal models, we can instantiate a Euclidean Markov Decision Process (EMDP) [13] from which Stratego MDP model can learn near-optimal control strategies. This model simulates both the behaviour of the house but also contains models of the expected weather and the future energy price prediction. We use this model for both evaluation and predictive control in our experimental setup, including an experiment with varying degrees of noise in the model used for predictive control. For instance, we replay a historical weather scenario during the evaluation while the predictive controller only has access to an approximate weather forecast.

3 Thermal Model Identification

The CTSM-R software [17] allows for continuous-time *grey-box* model identification. The general method of CTSM-R is based around maximum likelihood estimation and a gradient-decent approach for convergence. It identifies Continuous Time Stochastic Model and estimates the embedded parameters. CTSM-R has been successfully applied for the identification and estimation of physical system models, e.g., heat thermodynamics of buildings and walls, thermostats and radiators, and more [5, 24, 27].

We model the thermal dynamics of the house as presented in Equations (2)–(4) from Figure 3. For readability, we annotate the system in the following way: predicted state-variables by a tilde, e.g. $\tilde{T}_r^i$, inputs directly affected by the two

Data	Description
$\tilde{T}_r^i$	room i air temp.[°C]
$\overline{M^i}$	room i water mass flow [kg/s]
$\overline{T_f}$	water temp. exiting heat-pump [°C]
$\dot{T}_a$	outside temp. [°C]
$\overline{T^i}_{return}$	water temp. exiting room i [°C]
$\dot{S}^i$	solar heat to room i [Watt]

(a) Measured data

Constant	Description
α_h^i	floor to room air
α_e^i	envelope to room
α_s^i	solar radiation to room
α_w^i	pipes to floor
α_a^i	envelope to outdoors
α_n^i	room i to room n
β_h^i	floor heat capacity
β_h^e	envelope heat capacity

(b) Heat exchange coefficients

$$\frac{d\tilde{T}_r^i}{dt} = \alpha_h^i(\tilde{T}_h^i - \tilde{T}_r^i) + \alpha_e^i(\tilde{T}_e^i - \tilde{T}_r^i) + \alpha_s^i \cdot \dot{S}^i \tag{2}$$

$$\frac{d\tilde{T}_h^i}{dt} = \frac{\alpha_h^i}{\beta_h^i}(\tilde{T}_r^i - \tilde{T}_h^i) + \alpha_w^i \cdot \overline{M^i}(\overline{T_f} - \tilde{T}_h^i) \tag{3}$$

$$\frac{d\tilde{T}_e^i}{dt} = \frac{\alpha_e^i}{\beta_e^i}(\tilde{T}_r^i - \tilde{T}_e^i) + \alpha_a^i(\dot{T}_a - \tilde{T}_e^i) + \sum_{\substack{n=1 \\ n \neq i}}^{N} \alpha_n^i(\hat{T_r^n} - \tilde{T}_e^i) \tag{4}$$

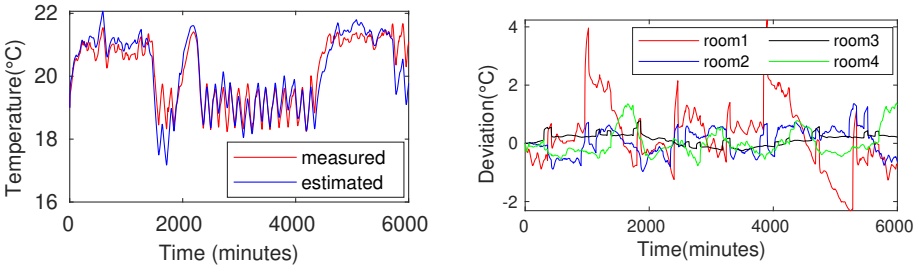
(c) Proposed three state energy model

Fig. 3: The thermodynamics system with the input data and the heat transfer coefficients where $i \in \{1, 2, 3, 4\}$ and, (α), (β) are "resistance" and "capacity"

levels of controls (i.e. the *heat-pump* or the *room thermostat*) with an overline, e.g. $\overline{M^i}$, impact of nature with a dot, e.g. $\dot{T}_a$, and constants (to be estimated) are left without decoration.

The thermal dynamics model follows a classical three-state framework consisting of room temperature ($\tilde{T}_r^i$), heater temperature ($\tilde{T}_h^i$) and envelope temperature ($\tilde{T}_e^i$). This approach is similar to what was presented in [16]; however, we here consider the rooms as individual model identification problems to overcome instability in the model identification when the number of coefficients to be estimated grows. Here the room temperature ($\tilde{T}_r^i$) is directly affected by the heater (relative to the coefficient α_h^i), the envelope (relative to the coefficient α_e^i) and the direct solar radiation of the room ($\dot{S}^i$ relative to the coefficient α_s^i). The room temperature directly impacts the heater temperature ($\tilde{T}_h^i$), but also receives energy from the in-flowing hot water which is proportional to the flow-rate $\overline{M^i}$ and the relative difference to the forward temperature of the water ($\overline{T_f}$). The envelope ($\tilde{T}_h^i$) models the energy transfer through the walls surrounding the room. The envelope thus exchanges energy with the room itself ($\tilde{T}_r^i$), the outside world $\dot{T}_a$ (proportional to the coefficient α_a^i) and with the other rooms of the house (the last sum term of Equation (4)).

A legend of the variables used can be seen in Figure 3a for data-variables and Figure 3b for the transfer coefficients. Given a time-series of historical data of the



(a) Room 4 (b) Deviations for each room

Fig. 4: Predicted and and measured indoor temperatures

variables presented in Figure 3a (excluding $\tilde{T}_h^i$ and $\tilde{T}_e^i$), the CTSM-R software can estimate the heat transfer coefficients presented in Figure 3b. The model is estimated s.t. the suggested coefficients allow us to predict $\tilde{T}_r^i$ with a high accuracy given known values for the massflow and temperature of the feed-in water and the environmental influences such as sun and outdoor temperature.

Decomposed Grey-Box Modelling. Notice here that the presented model only considers a single room. This implies that the identification can be done on a room-to-room basis without the explosion in the coefficients to be estimated. In general, this allows us for faster model identification. This implies repeating the same model-identification scheme four times but with different indexing. As we shall see later, four such identified models can be recomposed to form a complete model with high predictive power.

Evaluation of the Estimations. We shall now assess the quality of our proposed model-identification. The model is identified on a time-series consisting of data generated by the high-fidelity Dymola model using historical weather input from February 05, 2009, from Aalborg, Denmark and five days forward at a sample rate of 60 seconds.We note that the model estimation for the values is completed in less than 7 minutes for each room. Remark here that the data used for the estimation contains some effects not captured by Equations (2)–(4); notably occupants and cooking activities, contributing to significant noise in the system.

To estimate the quality of the model, we compute a 6000 minutes time-series in Dymola and compare the predicted room temperatures of each of the identified room-models to the reference computed by Dymola. In Figure 4a we see a head-to-head comparison for Room 4 where we can observe that the identified model demonstrates good predictive power. In particular, we note that no divergent behaviour is observed. The deviation of each room from measured data can be seen in Figure 4b. Excluding Room 1, the largest average deviation observed for all rooms is less than 1 C°, and in general, kept below 0.5 C° of difference. However, in Room 1 in the following intervals 1000-1400, 2400-2900, 3800-4300, 4700-5400 minutes, significant deviation occurs, which is traced back to the unaccounted contribution of heat from cooking activities which are

expected to be unobservable in a real application—and not captured during the model identification.

Estimating Return-Water Temperatures. The model created so far is only concerned with the internal transfer of energy in the house; however, in Figure 1 we can see that the return-water of the rooms is given as input to the heat-pump. We thus need to estimate a second model for predicting the return-water temperature. We here again adopt a data-driven approach and derive a model of this temperature-drop directly from data. We here assume a simple linear model; the increase in temperature of a room is assumed to be primarily due to energy dissipation from the heater. While this assumption is true in a closed system (due to the principle of conservation of energy), we here know that the assumption is incorrect: external influences occur, such as loss and gain of energy from neighboring rooms. This assumption allows us to establish the relationship described in Equations (5)–(6), and using historical data for $\overline{T}_{forward}$, $\tilde{T}_h^i$ and $\tilde{T}_r^i$ we can estimate the coefficients α, β and intercept γ. Again, these coefficients allow us to later predict the value of $\overline{T}_{return^i}$ for use in our full predictive model. Notice that prior to model-identification, we filter out data points where the water is not flowing (i.e. the local thermostat has shut off the supply). It is observed that while the return temperatures are all above 32 C°, the absolute mean error in estimating them is for all the rooms between 0.74 C° and 0.95 C°.

$$Energy_{loss}^i = \overline{T}_{forward} - \tilde{T}_h^i \tag{5}$$

$$Energy_{gain}^i = \tilde{T}_h^i - \tilde{T}_r^i \tag{6}$$

$$\overline{T}_{forward} - \overline{T}_{return^i} \approx \alpha \cdot Energy_{loss}^i + \beta \cdot Energy_{gain}^i + \gamma \tag{7}$$

4 Modelling in Uppaal Stratego

We can create a complete predictive model given the two identified thermal models. For doing so, we use the Uppaal tool suit [2–4, 19] which has been successfully applied in many industrial projects for verification, performance analysis, and strategy synthesis. Uppaal Stratego [7] is a branch of the tool that provides machine learning-based techniques for strategy synthesis and cost optimization of different controllers from Priced Timed MDPs. It has a rich modelling formalism for stochastic and hybrid games and control synthesis exploiting efficient reinforcement learning facilities. In Uppaal systems are modelled as networks of finite-state automata processes. The processes communicate with each other through channels or shared variables, and real-valued clocks facility is available in the tool to capture critical timing aspects of a system. In addition, Stratego provides C-library support [14] which offers a convenient way to construct complex interactions with other libraries and historical data, for instance, Stratego itself.

The overall composition of the system as a model in Stratego can be seen in Figure 5. We design sub-parts of the system as separate templates (parameterizable automata). The dashed-line areas in Figure 5 represent that the model has four templates: *Room*, *HeatPumpController*, *DataReader* and *ObjectiveFunction*.

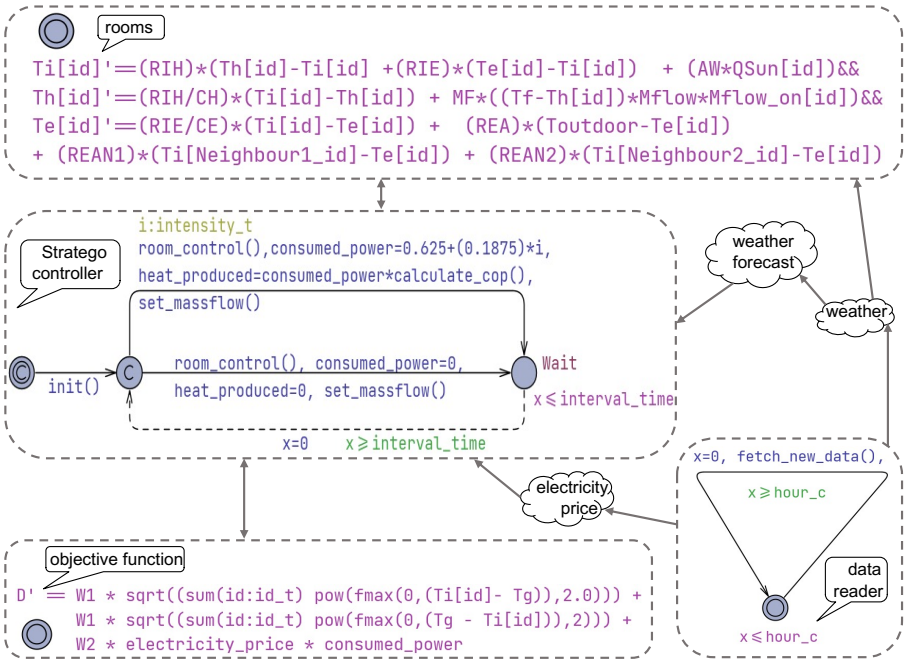


Fig. 5: Overall system composition

In the *Room* template, we express the continuous variables $\tilde{T}_r^i$, $\tilde{T}_h^i$ and $\tilde{T}_e^i$ (developed in Equations 2–4) as real-time clocks which evolve with time, but with rate-expressions matching the identified models. The *DataReader* template reads the *weather* and *day-ahead electricity price* information from the data file into the model. The *ObjectiveFunction* template implements the *fitness/optimization* function (see Equation 1). The *HeatPumpController* template implements the control mechanism of the controller where choices for using different energy levels for heat-pump are made.

Stratego handles this heating problem as a *stochastic hybrid game*. In this model, solid edges indicate controllable actions, and dashed lines indicate uncontrollable (controlled by the environment) actions. Circles are denoted locations, and the double-circle indicate the initial location of automata and arrows are called *edges*. An (instantiated) set of templates constitute a Network of (Hybrid Stochastic) Timed Game Automata (which semantically gives us an EMDP) where a state is given by a location vector (one for each template instance) and an assignment of concrete values to any variable. The system may then evolve by respecting `guards` and `invariants` – in particular, those that restrict the continuous development of clocks wrt our house dynamics. The initial location is committed (marked by a C in the locations) forcing the model to initially call the **`init()`** function, initializing the *weather*, *electricity price* and *solar radiations* values. At the following location, the controller may take any of the two edges depending on the amount of energy it decides to operate the heat-pump in an interval; in the lower transition, the heat-pump is shut off while in the

upper uses the *select* statement to compactly implement 10 different intensity levels of the controller (encoded in the type `intensity_t`). I.e. the controller assigns the temporary variable i a discrete value from *intensity* (range between 0 and 10) to reflect different intensity level choices to operate the heat-pump. The function `roomControl()` implements the bang/bang controllers of the room thermostats, which decides whether to periodically open or close the valves leading to each room, this directly gives a way to compute the distribution of the ongoing flow by the `setMassFlow()` function. The variable `consumed_power` contains the amount of electricity selected by the controller for each intermediate interval during strategy synthesis while `heat_produced` is the produced heat for every interval. The function `calculate_cop()` calculates the *Coefficient of Performance* (COP) value used to measure the produced heat. Notice that the COP value, for a given return-water temperature and outdoor temperature, provides a gearing of input to output energy; i.e. at a COP value of 2, a single kWh of power yields 2kWh of energy. Such gearing is normally found in the technical specifications of a heat-pump. This gearing, along with the pump intensity-setting, massflow and the return-water temperature, allows for the computation of the *forward* temperature and using the equations for return-water temperature. The return-water temperature is computed based on the previous time-step, and the relative changes in the indoor temperatures allow us to predict this value using the developed equations from Section 3. At the `wait` location, the invariant `x<=interval` bounds the system to stay there until the value of clock `x` (local clock) reaches `interval` i.e. 15-minute. The constant `interval` is the control frequency of the heat-pump. The guard `x>=interval` prevents the system from making this transition before spending 15 minutes at the `wait` location, after which the clock `x` is reset to zero, allowing for timing a new interval.

Evaluation using Stratego*:* We use the provided model for both learning and evaluation in our experiments. However, in the evaluation phase, the control choices of the Stratego controller template are restricted to follow those suggested by a call to an external library, implying that the controller is invoked for every simulated 15 minutes. When we evaluate the system under the control of a bang-bang controller, this call returns either full intensity or the 0 intensity value. When the Stratego itself is used as a controller instead, the call instantiates the above templates—but instead of evaluating, it will synthesize a controller by repeated sampling.

4.1 Learning by Stratego

To attain a predictive controller, Stratego trains on the instantiated model, using the initial state estimate by exploiting repeated sampling and Q-learning. This allows the tool to factor in temporal changes such as weather and price changes of the near future. However, four problems arise:

1. not all variables are observable, specifically only the room-temperatures ($\tilde{T}_r^i$), the weather forecast and the price projection can be observed,

2. controllers take time to compute, leading to a delay from observation to reaction,
3. the development of the real world (or evaluation model) can deviate over time, and
4. both weather forecasts and prices have a limited horizon.

The latter points we shall discuss in Section 4.2, and let us instead here address the first issue.

Initial state estimation: In our three-state thermodynamics model, the changes in room temperature ($\tilde{T}_r^i$) are directly dependent on two hidden states, i.e., floor temperature ($\tilde{T}_h^i$) and envelope temperature ($\tilde{T}_e^i$). The states are unobservable and they intuitively track the abstract value of "energy stored in the heater" and "energy stored in the wall" respectively. This implies that the tool works on a partially observable model. To attain reasonable estimates, we can predict these hidden variables—and similarly the future value of observable values to accommodate for the observational delay. Assuming that a full state is known at τ_t and that a control choice has already been made, then the state at τ_{t+15} can be captured by forward simulation of the model. Experiments show that this method of approximating the hidden variables $\tilde{T}_e^i$ and $\tilde{T}_h^i$ allowed a drift of no more than $0.1C^\circ$ between the repeated estimates of the variables in the model used for learning and the model used for evaluation.

4.2 Online Synthesis

Given that energy costs are only known 24 hours in advance and that weather forecasts are known to have degrading predictive power the further into the future they look, we utilize an online synthesis procedure. At the same time, as our model used for controller synthesis has hidden variables, a rapid recomputation of a strategy allows us to adjust for errors made in the initial state-estimation and potential discrepancies with the real house (or evaluation model in our case).

We thus propose a method where at each 15-minute interval, the volatile variables (energy price, weather and measurable variables of the house) are monitored and transferred to the controller. This allows the controller to instantiate the method of Section 4.1 with the most recent measurements. We can then let Stratego synthesize a controller on. An overview of this flow is given in Figure 6. Each day has four 6-hour periods, and each period has 24 of 15-minute intervals. The reuse interval (k) is 24, which means a single strategy is used for k intervals. In the first interval, Stratego gets the current *room temperatures*, *weather forecast* and *day-ahead electricity price*, estimates the initial state and synthesizes a strategy that minimizes the fitness function F by sampling 12-hours (learning horizon) ahead in future. The controller saves the first strategy (*S1*). The heat-pump uses *S1* during intervals 2–25 and makes another strategy during interval 25, which is then used for the next 24 intervals (i.e. from 26–49).

Notice here that it is assumed that the synthesis procedure is *not* instantaneous, implying that a synthesized controller can only be applied in the subsubsequent interval. To overcome this issue, we apply the initial state-estimation

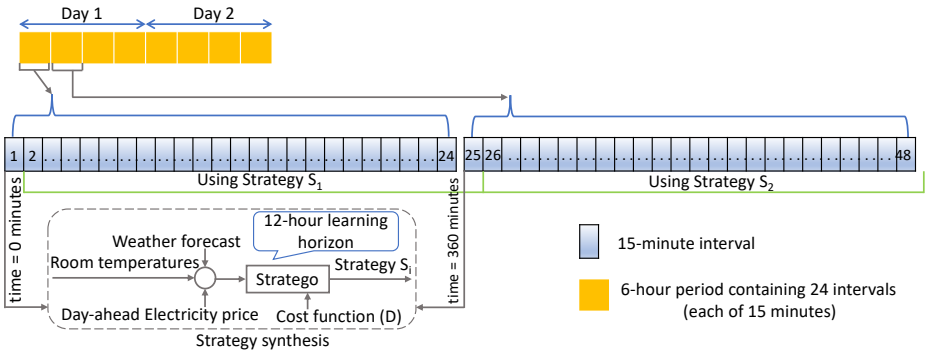


Fig. 6: Online synthesis approach shown on a two day period.

as described in Section 4.1 to estimate the house state at a time-point $\tau + 15$ from measurements obtained at τ. We set the training horizon of the method to be 12 hours, implying that the learning method will only sample the system up to a horizon of 12 hours from the starting measurement. This can aptly be done in STRATEGO using the query presented in Equation 8. We here see that the controller is trained to only react on the observable variables `minute_clock` (tracking real-time), `Ti[i]` (tracking indoor room temperatures) and `Toutdoor_forecast` (tracking weather forecast). The result of the synthesis procedure is a reactive strategy that for an assignment of the input variables yields a (near-)optimal action to take [6, 13, 18]

```
strategy S = minE (F) [minute_clock <= 12*60)]
          {} -> {minute_clock, Ti[0], Ti[1], Ti[2], Ti[3],
                 Toutdoor_forecast}: <> minute_clock==(12*60)
```
(8)

Notice also that Equation 8 defines a feature vector (observable variables of the state-space) to be only directly measurable variables of the original system. This makes the strategy directly applicable for several intervals (up to the learning horizon) in the actual system under control. We can exploit this to lower the overall computational effort of deploying our approach, which directly impacts the overall energy impact of running the smart heating system. Essentially, by re-using a strategy for several intervals, a single server can act as a controller for a group of houses, i.e. if a strategy only needs to be computed every six intervals, six houses can share the computational resource.

5 Evaluation

To validate our approach, we experiment with controlling the OpSys house for a week with weather conditions matching February 4-10, 2018. As a reference, we use the estimated model presented in Section 4. The goal of any controller is to minimize the parameterized cost-criteria defined in Equation 1 for the period

under control. In all experiments, we compare different variations of the online Stratego-based controller against a standard bang-bang (BB) controller.

We conduct three series of experiments:

1. a series under realistic assumptions on observability and time,
2. a study of sensitivity to various degrees of measurement noise, and
3. a study of impact of changes in the learning parameters of Stratego.

All experiments are conducted on AMD EPYC 7551 limited a single core and 2GB of memory[1]. We limit the controller to 15-minute control intervals which is sufficient for systems with as slow dynamic as floor-heating. For each reported configuration, the experiment is repeated 10 times and the mean value is reported with standard deviation intervals reported in bar-charts. A reproducibility package is available in [12].

In all series, we observe changes to the results when the weights of comfort ($0 \leq W_{comf} \leq 1$) changes; we modify these values in steps of 0.1. To make this weighting independent of the given week and to allow for this proportional weighting between two units (kWh and squared degrees Celsius), we compute a normalization-factor *norm* based on the BB controller. This normalization-factor is computed by estimating the performance of the BB controller on the week leading up to February 4-10 in terms of *cost* and *comfort*. The normalization-factor is then given directly by $norm = \frac{comfort}{cost}$.

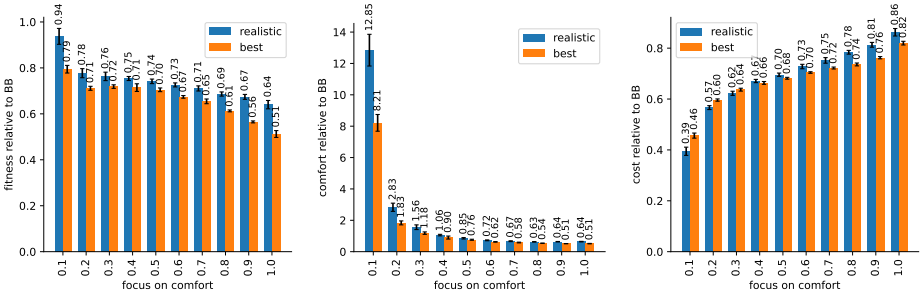
In our plots we report *relative* performance to that of the BB controller; we do so as the behaviour of the BB controller in the model (when replayed with historical weather) is deterministic. We omit to report values for a pure emphasis on cost as this leads the controller to (as expected) turn off all heat.

Realistic Evaluation: In this series, we emulate a realistic setup; this implies that hidden variables are estimated, the weather forecast is stochastic, and the controller only can be applied with a 15 minutes delay from the variable observation-time, as described in Section 4.

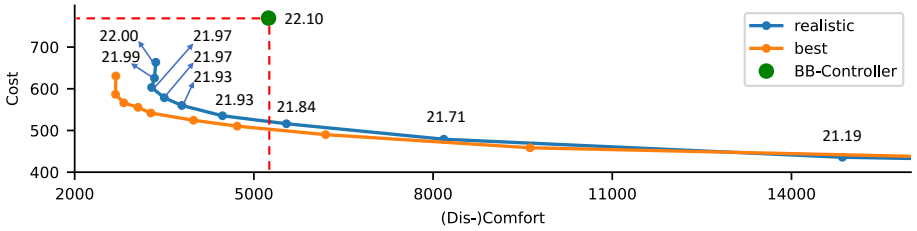
We experiment with two configurations and compare them with the standard BB controller. In one, we fix the learning parameters s.t. the wall-clock computation time is not exceeding 15 minutes on given hardware; namely 600 samples (depicted blue in Figure 7), and a re-use interval of 24 (6 hours) limiting the computation effort—denoted the *realistic* configuration. In a more hypothetical application, we double the training budget to 1200 samples (leading to an approx 30 min computation time) and reduce the re-use interval to 6 (depicted orange in Figure 7)—named the *best* configuration.

We see that the BB controller is dominated in all ways by any of the two Stratego controllers. In particular we see that at $W_{comf} = 0.4$ either Stratego controller achieves comparable comfort to the BB controller (Figure 7b), but at a 33-34% reduction in cost (Figure 7c). In Figure 7a, we observe for any setting of the parameters the Stratego-based controllers are dominating in the distance measure and more so with an increased comfort weight. In Figure 7d we

[1] Actual memory usage expected to be significantly lower, but not recorded.



(a) Relative fitness (b) Relative discomfort (c) Relative cost



(d) The cost/comfort tradeoff with different W_{comf} settings (red line separates the points on its left side where STRATEGO is outperforming BB w.r.t both comfort and cost)

Fig. 7: February week control under partial observability with 24 reuse intervals and 600 training samples (blue) and 6 reuse intervals and 1200 training samples.

have also annotated the average temperature experienced throughout the week for the *realistic* controller to provide a tangible perspective on the discomfort measure. With a less than $0.16C°$ deviation in the average temperature from the setpoint with $W_{comf} \geq 0.4$, again resulting in a 33% reduction in cost.

As the BB controller is incapable of adjusting to the time-varying cost function, it is expected that the STRATEGO controller can outperform it with a focus on cost. More interesting, the benefit of using STRATEGO grows with an increased focus on comfort. Notice that Equation 1 penalizes overshooting. We hypothesize that the binary mode of the BB entails periods of large overshooting of the target temperature and others with undershooting due to the 15-minute control interval [2].The STRATEGO controllers can instead compensate gradually and react more subtly.

While the *realistic* controller in general trails the *best* controller, find it to be well-performing in general, trailing with no more than 4% points in comfort for settings with $W_{comf} \geq 0.4$ and a loss in comfort of no more than 13% points in the same range.

Sensitivity Analysis of Impact of Measurement Noise: As our experimental setup is virtual, we can study the impact of stochastic weather-prediction, hidden variables and delayed controller response; this comparison is seen in Figure 8. The

[2] Due to wear and tear on heat-pumps it is undesirable to change states too often.

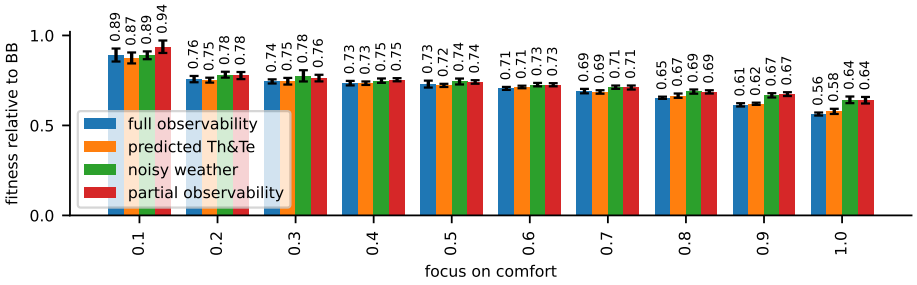


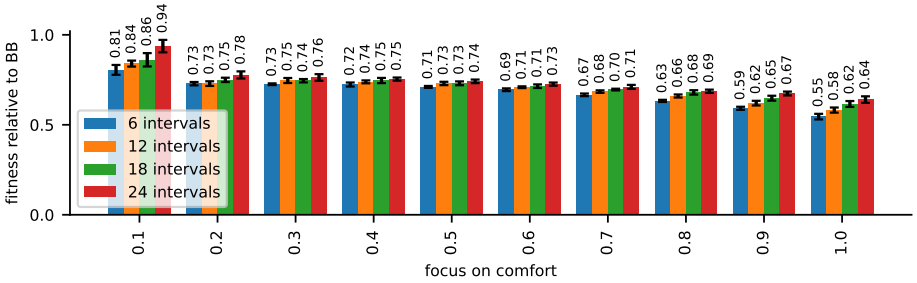
Fig. 8: February week control under different observability situations with 24 reuse intervals and 600 learning runs

experimental setup is similar to the *Realistic* series apart from the specified changes. We experiment with four configurations: 1. Full observability, allowing for delayed controller response, perfect weather-prediction and direct observation of hidden variables (i.e. an ideal scenario), 2. Predicted $\tilde{T}_e^i$ and $\tilde{T}_h^i$, otherwise as full observability, 3. Noisy weather, otherwise as full observability, and 4. Partial observability, exactly the same setup as in the *Realistic Evaluation* series. In Figure 8, we observe that while a high emphasis on cost (which is a fully observable variable), the impact of noisy weather prediction and predicted unobservable variables has only a modest effect (2-3%), however with a ≥ 0.8 focus on comfort, we see the uncertainty of the weather significantly impacting the performance (up to 8%), indicating a sensitivity of the controller towards accurate forecasts. We observe an anomaly with a 0.1 emphasis on comfort where the full observability configuration is significantly worse than the predicted scenario. While the discrepancy appears to be within measurement noise, it still warrants further investigation. We hypothesize that the squaring of the difference of the target and the actual temperature in Equation 1 leads to rare spikes in the response affecting the partition-refinement scheme deployed of Stratego.

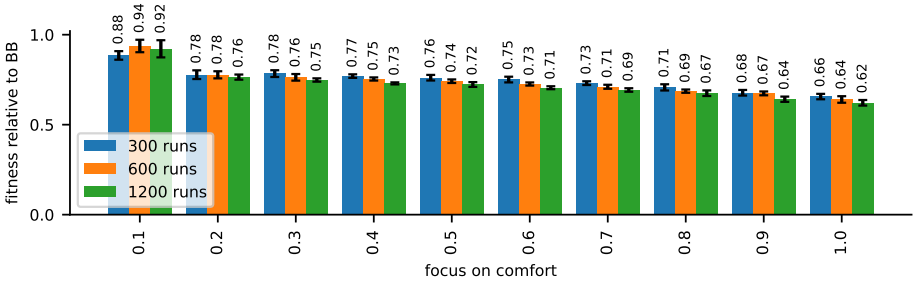
Sensitivity Analysis of Learning Parameters: Stratego is a sampling-based tool, and the performance (in terms of quality of the controller synthesized) is dependent on the number of samples provided. This is given directly in the number of runs (simulations) the engine is allowed to conduct.

In Figure 9b we see that an increase in the number of runs (in general) improves the performance. However, with a doubled effort of 1200 runs (approx 30 minutes of computation pr. strategy), the performance in general improved by 2% and in a rare case by 3%. We can see a similar improvement in performance with an increase from 300 to 600 runs. However, a similar anomaly is observed with an 0.1 emphasis on comfort as was found in the *Measurement Noise*-series of experiments. We conjecture that this effect manifests to a higher degree with an increased learning effort.

We here also experiment with reducing the overall computational effort needed for control. In Section 4.2 we introduced the *reuse interval* which allows for applying a given control strategy for an extended period. In Figure 9a we observe that a tighter re-computation cycle of 6 intervals (every $1\frac{1}{2}$ hours) allows us to gain 3% performance compared to our *Realistic* series of experiments. Towards



(a) February week with different reuse intervals and 600 learning runs



(b) February week with different number of runs with 24 reuse intervals

Fig. 9: Effect of choosing different sets of reuse intervals and number of runs

a reuse of 24 periods, we observe a drop in performance when the reuse interval is extended.

6 Conclusion

We presented a tool-chain for controlling a heat-pump system in a floor heating case study. The tool-chain offers an end-to-end solution for floor heating applications by establishing an automatic procedure for the identification of house thermodynamics and designing Uppaal Stratego controller. We compare the performance of Stratego controller against the traditionally used *bang-bang* controller. Experimental results show that our controller offers significant improvements in both user *comfort* as well as energy *cost*, even when realistic limitations on computation effort are taken into account. In particular, Stratego saves 33% energy while preserving the same comfort as the standard bang-bang controller. We also analyse the cost-comfort trade-off paradigm, which shows that we can save energy costs by slightly compromising the comfort. We believe that the results can be further improved by introducing a heat buffer and that the computational effort can be reduced by techniques such as ensemble learning.

Acknowledgements. We would like to thank Simon Thorsteinsson for his extensive help with Dymola and acquiring base data for model identification. This research is partly funded by the ERC Advanced Grant LASSO, the Villum Investigator Grant S4OS as well as DIREC: Digital Research Centre Denmark.

References

1. Agesen, M.K., Larsen, K.G., Mikučionis, M., Muñiz, M., Olsen, P., Pedersen, T., Srba, J., Skou, A.: Toolchain for user-centered intelligent floor heating control. In: IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society. pp. 5296–5301 (2016), 10.1109/IECON.2016.7794040
2. Behrmann, G., Cougnard, A., David, A., Fleury, E., Larsen, K.G., Lime, D.: Uppaal-tiga: Time for playing games! In: International Conference on Computer Aided Verification. pp. 121–125. Springer (2007)
3. Behrmann, G., David, A., Larsen, K.G., Håkansson, J., Pettersson, P., Yi, W., Hendriks, M.: Uppaal 4.0. IEEE Computer Society (2006)
4. Bulychev, P., David, A., Larsen, K.G., Mikučionis, M., Poulsen, D.B., Legay, A., Wang, Z.: Uppaal-SMC: Statistical model checking for priced timed automata. arXiv preprint arXiv:1207.1272 (2012)
5. Carrascal, E., Garrido, I., Garrido, A.J., Sala, J.M.: Optimization of the heating system use in aged public buildings via model predictive control. Energies **9**(4), 251 (2016)
6. David, A., Jensen, P.G., Larsen, K.G., Legay, A., Lime, D., Sørensen, M.G., Taankvist, J.H.: On time with minimal expected cost! In: International Symposium on Automated Technology for Verification and Analysis. pp. 129–145. Springer (2014)
7. David, A., Jensen, P.G., Larsen, K.G., Mikučionis, M., Taankvist, J.H.: Uppaal stratego. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems. pp. 206–211. Springer (2015)
8. Dixit, M.K., Fernández-Solís, J.L., Lavy, S., Culp, C.H.: Identification of parameters for embodied energy measurement: A literature review. Energy and buildings **42**(8), 1238–1247 (2010)
9. Ferracuti, F., Fonti, A., Ciabattoni, L., Pizzuti, S., Arteconi, A., Helsen, L., Comodi, G.: Data-driven models for short-term thermal behaviour prediction in real buildings. Applied Energy **204**, 1375–1387 (2017)
10. Fonti, A., Comodi, G., Pizzuti, S., Arteconi, A., Helsen, L.: Low order grey-box models for short-term thermal behavior prediction in buildings. Energy Procedia **105**, 2107–2112 (2017)
11. Harasty, S., Lambeck, S., Cavaterra, A.: Model predictive control for preventive conservation using artificial neural networks. In: 12th REHVA World Congress. Aalborg, Denmark (2016)
12. Hasrat, I., Jensen, P., Larsen, K., Srba, J.: Artefact for "End-to-End Heat-Pump Control Using Continuous Time Stochastic Modelling and Uppaal Stratego" (May 2022), https://doi.org/10.5281/zenodo.6548367
13. Jaeger, M., Bacci, G., Bacci, G., Larsen, K.G., Jensen, P.G.: Approximating euclidean by imprecise markov decision processes. In: International Symposium on Leveraging Applications of Formal Methods. pp. 275–289. Springer (2020)
14. Jensen, P.G., Larsen, K.G., Legay, A., Nyman, U.: Integrating tools: Co-simulation in uppaal using fmi-fmu. In: 2017 22nd International Conference on Engineering of Complex Computer Systems (ICECCS). pp. 11–19. IEEE (2017)
15. Østergaard Jensen, S.: OPSYS tools for investigating energy flexibility in houses with heat pumps. https://www.annex67.org/media/1838/report-opsys-flexibilitet.pdf (2018)
16. Juhl, R., Kristensen, N.R., Bacher, P., Kloppenborg, J., Madsen, H.: Grey-box modeling of the heat dynamics of a building with CTSM-R (2017), http://ctsm.info/pdfs/examples/building2.pdf

17. Juhl, R., Møller, J.K., Madsen, H.: CTSMR - Continuous Time Stochastic Modeling in R. arXiv (2016), 10.48550/ARXIV.1606.00242
18. Larsen, K.G., Mikučionis, M., Muñiz, M., Srba, J., Taankvist, J.H.: Online and compositional learning of controllers with application to floor heating. In: Chechik, M., Raskin, J.F. (eds.) Tools and Algorithms for the Construction and Analysis of Systems. pp. 244–259. Springer Berlin Heidelberg, Berlin, Heidelberg (2016)
19. Larsen, K.G., Pettersson, P., Yi, W.: Uppaal in a nutshell. International journal on software tools for technology transfer **1**(1-2), 134–152 (1997)
20. Ljung, L.: Matlab system identification toolbox—getting started guide r2016a. Mathworks, Ed., Mathworks (2016)
21. Nassif, N.: Modeling and optimization of hvac systems using artificial neural network and genetic algorithm. In: Building Simulation. vol. 7(3), pp. 237–245. Springer (2014)
22. Privara, S., Cigler, J., Váňa, Z., Oldewurtel, F., Sagerschnig, C., Žáčeková, E.: Building modeling as a crucial part for building predictive control. Energy and Buildings **56**, 8–22 (2013)
23. Reynders, G., Diriken, J., Saelens, D.: Quality of grey-box models and identified parameters as function of the accuracy of input and observation signals. Energy and Buildings **82**, 263–274 (2014)
24. Thilker, C.A., Bergsteinsson, H.G., Bacher, P., Madsen, H., Calì, D., Junker, R.G.: Non-linear model predictive control for smart heating of buildings. In: E3S Web of Conferences. vol. 246, p. 09005. EDP Sciences (2021)
25. Vinther, K., Green, T., Jensen, S.Ø., Bendtsen, J.D.: Predictive control of hydronic floor heating systems using neural networks and genetic algorithms. IFAC-PapersOnLine **50**(1), 7381–7388 (2017)
26. Vogler-Finck, P., Wisniewski, R., Popovski, P.: Reducing the carbon footprint of house heating through model predictive control – a simulation study in danish conditions. Sustainable Cities and Society **42**, 558 – 573 (2018), http://www.sciencedirect.com/science/article/pii/S2210670718301173
27. Yu, X., You, S., Cai, H., Georges, L., Bacher, P.: Data-driven modelling and optimal control of domestic electric water heaters for demand response. In: The International Symposium on Heating, Ventilation and Air Conditioning. pp. 77–86. Springer (2019)