Online and Compositional Learning of Controllers with Application to Floor Heating

Kim G. Larsen, Marius Mikučionis, Marco Muñiz, Jiří Srba(✉), and Jakob Haahr Taankvist

Department of Computer Science, Aalborg University, Aalborg, Denmark
{kgl,marius,muniz,srba,jht}@cs.aau.dk

Abstract. Controller synthesis for stochastic hybrid switched systems, like e.g. a floor heating system in a house, is a complex computational task that cannot be solved by an exhaustive search though all the control options. The state-space to be explored is in general uncountable due to the presence of continuous variables (e.g. temperature readings in the different rooms) and even after digitization, the state-space remains huge and cannot be fully explored. We suggest a general and scalable methodology for controller synthesis for such systems. Instead of off-line synthesis of a controller for all possible input temperatures and an arbitrary time horizon, we propose an on-line synthesis methodology, where we periodically compute the controller only for the near future based on the current sensor readings. This computation is itself done by employing machine learning in order to avoid enumeration of the whole state-space. For additional scalability we propose and apply a compositional synthesis approach. Finally, we demonstrate the applicability of the methodology to a concrete floor heating system of a real family house.

1 Introduction

Home automation includes the centralized control of a number of functionalities in a house such as lighting, HVAC (heating, ventilation and air conditioning), appliances, security locks of gates and doors as well as other systems. The overall goal is to achieve improved convenience, comfort, energy efficiency as well as security. The popularity of home automation has increased significantly in recent years through affordable smartphone and tablet connectivity. Also the emergence of "Internet of Things" has tied in closely with the popularization of home automation. In particular, several devices may be connected through a home network to allow control by a personal computer, and may allow remote access from the internet.

The connectivity in the home enables new, intelligent and personalized control strategies or (and across) activities in the house. One novel approach which is being developed and applied in the on-going EU FP7 project CASSTING[1] is that of *game theory*. Empowered with efficient techniques and tools, game

[1] www.cassting-project.eu.

M. Chechik and J.-F. Raskin (Eds.): TACAS 2016, LNCS 9636, pp. 244–259, 2016.
DOI: 10.1007/978-3-662-49674-9_14

theory comes with the promise of automatic synthesis of improved, optimal and personalized control strategies produced on demand by the user herself. In fact, the tool Uppaal Tiga has already been successfully[2] applied to user-directed and user-demanded synthesis of control strategies for lighting in a house, and been implemented in a complete tool-chain on a Raspberry Pi [12].

Within the CASSTING project, we collaborate with the Danish company Seluxit[3] offering complete home automation solutions. The focus is on the floor-heating system of a family house, where each room of the house has its own hot-water pipe circuit. These are controlled through a number of valves based on information about room temperatures communicated wirelessly (periodically due to energy considerations) from a number of temperature sensors. In the present system, a simple "Bang-Bang"-like strategy is applied. There are though several problems with this strategy, as experienced by the house owner: it completely disregards the interaction between rooms in terms of heat-exchange, the impact of the outside temperature and weather forecast as well as information about movements in the house. Taking this knowledge into account should potentially enable the synthesis of significantly improved control strategies.

For the control synthesis of the lighting system, timed games and Uppaal Tiga proved sufficient. However, in order to control a floor-heating system, we must take into account continuous (temperature) as well as stochastic aspects (outside temperature, movements). Hence we need to be able to (efficiently) synthesize strategies for *stochastic hybrid games*.

A promising starting point is the recent branch Uppaal-Stratego [5,6], which allows for the synthesis of safe and near-optimal strategies for stochastic timed games using a combination of symbolic synthesis and reinforcement learning. The tool has recently been extended to stochastic hybrid games with a successful application to the synthesis of strategies for battery aware scheduling problems [14] as well as safe and optimal adaptive cruise controllers for cars [9].

Facing the floor heating case study of CASSTING, direct application of Uppaal-Stratego does not scale: due to the enormous number of control modes it is virtually impossible to learn optimal control. Instead, we propose a novel on-line synthesis methodology, where we periodically—and *on-line*—learn the optimal controller for the near future based on the current sensor readings. For additional scalability, we propose and apply a novel compositional synthesis approach. As we shall see this combination allows us to significantly improve upon the currently applied "Bang-Bang" control strategy.

Related Work. In [10,11] a method and tool (PESSOA) is presented for synthesizing controllers for cyber-physical systems, represented by a set of smooth differential equations and automata given a specification in a fragment of Linear Temporal Logic (LTL). In [8] a class of hybrid systems that involve random phenomena, in addition to discrete and continuous behaviour are considered, and abstraction techniques are presented and applied to the synthesis of controllers. In [13] the authors provide an abstraction-refinement method for

[2] [12] won the Embedded Thesis Award 2014 of the Federation of Danish Industry.
[3] www.seluxit.com.

synthesis of controllers for discrete, stochastic dynamical systems with respect to LTL objectives. In [7] a number of benchmarks for hybrid system verification has been proposed, including a room heating benchmark. In [3] Uppaal SMC was applied to the performance evaluation of several strategies proposed in the benchmark. In [4] a combination of Uppaal SMC with ANOVA has been made for efficient identification of optimal parameters of the various control strategies.

Our online approach may be seen as an instance of model predictive control or receding time horizon control for hybrid systems (see e.g. [2]) where the optimal solutions are already very expensive to compute. We tackle even a more general class of systems (including stochasticity in particular) and apply a learning heuristic that is cheaper on the cost but does not guarantee optimality.

The main novelty of our work, compared to the previous research, is that we address an industrial-size case with its full complexity, where the already studied methods and approaches do not scale. It is the combination of online learning approach, employment of the very recent tool support and the compositional approach that allowed us to significantly improve upon the performance of the current controller used for the floor heating system in the existing house.

2 Switched Control Synthesis

We use a one-room heating control problem as a running example to demonstrate our techniques in a simple setting: we model the problem, explain the necessary theory behind the model, show how the model fits the theory and show how Uppaal Stratego can be used to solve the problem.

The one-room system consists of a room with walls, a window, heater and its controller. The objective of the controller is to maintain the room temperature at the goal level (21 °C). Due to temperature sensor energy considerations the controller receives temperature readings only once every 15 min and then it has two options: either to turn the heater on (mode "HeatOn") and keep it there or switch the heater off (mode "HeatOff"). Consequently the temperature evolution will be different in these modes due to different energy supply from the heater. There is also a continuous leak of energy through the walls and the window to the outside environment. In short, the temperature dynamics can be described by the following differential equation:

$$\frac{d}{dt}T(t) = \big(T_e(t) - T(t)\big) \cdot A(t) + H(t)$$

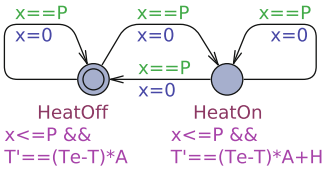
where $T(t)$ is the room temperature at time t, $T_e(t)$ is the outdoor temperature, $A(t)$ is the heat exchange factor specific to the walls and windows, and $H(t)$ is the power of the heater.

Figure 1b shows such differential equation with heater step functions modelled in Uppaal Stratego as hybrid automaton with two discrete modes. The continuous dynamics of $T(t)$ is typeset as an invariant constraint over the clock variable `T` derivative under the respective modes. The periodic behaviour of the controller is enforced by the invariant `x<=P` and guard `x==P` over clock `x` with

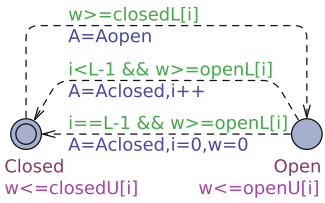
```
const double Tg = 21.0; // goal temp. in deg.C
const double Te = 15.0; // env. temperature
const double H = 0.04; // power of heater
const double Aclosed=0.002;// window closed
const double Aopen = 0.004;// window open
const int P = 15; // switching period, 15min
const int h = 60; // 1hour = 60min
clock t; // global time in minutes
clock T = 18.0; // room temperature in deg.C
clock D = 0.0; // distance between T and Tg
clock x; // controls switching period
clock w = 0.0; // controls window profile
int i = 0; // phase of window profile
double A = Aclosed; // heat to environment

const int L=4;// four phases in window profile:
const int closedL[L]={ 6*h, 11*h, 16*h, 19*h };
const int closedU[L]={ 7*h, 12*h, 17*h, 23*h };
const int openL[L] = { 7*h, 12*h, 18*h, 24*h };
const int openU[L] = { 8*h, 13*h, 21*h, 24*h };
```

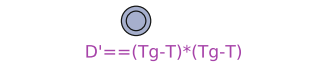
(a) Variable declarations



(b) Heating modes



(c) Window profile



(d) Distance to goal temp.

Fig. 1. Uppaal Stratego model of one room with one window

default derivative of 1. For the sake of simplicity, we assume static outdoor temperature fixed to a specific value and modelled by the constant floating point variable `Te`. All model variables (their types and initial values) are declared as C structures in Fig. 1a. The window step function $A(t)$ is modelled in Fig. 1c as stochastic automaton with transitions between "Open" and "Closed" modes and changing the floating point variable `A`. Thus the window process can change the value of `A` discretely between values `Aclosed` and `Aopen` at any moment with uniform probability distribution over time, but only at intervals specified by a user profile. The profile is stored in arrays `closedL/U` and `openL/U` denoting the lower and upper bounds of time intervals when the switch may happen. For example, one can read the profile arrays by columns: the window starts and stays closed during the night time, but it will open somewhere between 6 and 7 o'clock in the morning and close between 7 and 8 o'clock, then it will open again between 11 and 12, and close between 12 and 13, etc.

The whole system model is then a composition of the controlled heating process with the stochastic window process where temperature depends on the heating mode and the mode of the window. We use stochastic hybrid game to describe the controller synthesis formally.

Definition 1 (Stochastic Hybrid Game). *A* stochastic hybrid game $\mathcal{G}$ *is a tuple* $(\mathcal{C}, \mathcal{U}, X, \mathcal{F}, \delta)$ *where:*

1. $\mathcal{C}$ *is a controller with a finite set of (controllable) modes* C,
2. $\mathcal{U}$ *is the environment with a finite set of (uncontrollable) modes* U,
3. $X = \{x_1, \ldots, x_n\}$ *is a finite set of continuous (real-valued) variables,*

4. *for each* $c \in C$ *and* $u \in U$, $\mathcal{F}_{c,u} : \mathbb{R}_{>0} \times \mathbb{R}^X \to \mathbb{R}^X$ *is the flow-function that describes the evolution of the continuous variables over time in the combined mode* (c, u), *and*
5. δ *is a family of density functions,* $\delta_\gamma : \mathbb{R}_{\geq 0} \times U \to \mathbb{R}_{\geq 0}$, *where* $\gamma = (c, u, v) \in C \times U \times \mathbb{R}^X$. *More precisely,* $\delta_\gamma(\tau, u')$ *is the density that* $\mathcal{U}$ *in the global configuration* $\gamma = (c, u, v)$ *will change to the uncontrollable mode* u' *after a delay of* τ[4].

We shall assume that among the continuous variables X, there is a variable time measuring global time, i.e. $\mathcal{F}_{c,u}(\tau, v)(\mathsf{time}) = v(\mathsf{time}) + \tau$ for any mode-configuration (c, u). In the above definition, the syntactic specification of flow functions—e.g. using ODEs—has been left open. In the game $\mathcal{G}$, the controller $\mathcal{C}$ will only be permitted to change controllable mode at time-points being a multiple of some given period P (hence the term switched control). In contrast, the environment $\mathcal{U}$ will change its uncontrollable mode according to the family of density functions δ_γ.

Example 1. In our one-room example, the controllable modes are `HeatOff` and `HeatOn` with controllable transitions (using solid lines) between them, the uncontrollable are `Open` and `Closed` with uncontrollable transitions (using dashed lines). We also have a number of continuous variables: temperature `T` and clocks `t`, `x` and `w`. The differential equations together with discretely changing variables are part of the flow-function definition. ◁

Now let $\mathbb{C}$ denote the set of global configurations $C \times U \times \mathbb{R}^X$ of the game $\mathcal{G}$. Then a (memoryless) strategy σ for the controller $\mathcal{C}$ is a function $\sigma : \mathbb{C} \to C$, i.e. given the current configuration $\gamma = (c, u, v)$, the expression $\sigma(\gamma)$ is the controllable mode to be used in the next period.

Let $\gamma = (c, u, v)$ and $\gamma' = (c', u', v')$. We write $\gamma \xrightarrow{\tau} \gamma'$ in case $c' = c, u' = u$ and $v' = \mathcal{F}_{(c,u)}(\tau, v)$. We write $\gamma \xrightarrow{\tau}_u \gamma'$ in case $c' = c$, $v' = \mathcal{F}_{(c,u)}(\tau, v)$ and $\delta_\gamma(\tau, u') > 0$. Let $\sigma : \mathbb{C} \to C$ be a (memoryless) strategy. Consider an interleaved sequence π of configurations and relative time-delays of the form:

$$\pi = \gamma_o :: \tau_1 :: \gamma_1 :: \tau_2 :: \gamma_2 :: \tau_3 :: \gamma_3 \cdots$$

where $\gamma_i = (c_i, u_i, v_i)$, $\tau_i \in \mathbb{R}_{\geq 0}$ and for all n there exist i st. $\sum_{j \leq i} \tau_j = n \cdot P$. Then π is a run according to the strategy σ if for all i either $\gamma_i \xrightarrow{\tau_{i+1}}_u \gamma_{i+1}$ or $\sum_{j \leq i+1} \tau_j$ is a multiple of P and $\gamma_i \xrightarrow{\tau_{i+1}} (c_i, u_i, v_{i+1})$ with $c_{i+1} = \sigma((c_i, u_i, v_{i+1}))$ and $u_{i+1} = u_i$.

In fact, under a given strategy σ the game $\mathcal{G}$ becomes a completely stochastic process $\mathcal{G} \upharpoonright \sigma$, inducing a probability measure on sets of runs. Thus, if $H \in \mathbb{N}$ is a given time-horizon, and D is a random variable on runs—e.g. measuring the integrated deviation of the continuous variables wrt. given target values—then $\mathbb{E}^{\mathcal{G},\gamma}_{\sigma,H}(D) \in \mathbb{R}_{\geq 0}$ is the expected value of D with respect to random runs of $\mathcal{G} \upharpoonright \sigma$ of length H starting in the configuration γ. We want to obtain a strategy σ^H which minimizes (or maximizes) this expected value.

[4] Note that $\sum_{u'} \int_\tau \delta_{(c,u,v)}(\tau, u') d\tau = 1$ for all (c, u, v).

Example 2. The one-room controller's goal is to keep the room temperature as close as possible to the goal set point, therefore a desired controller would minimize the absolute difference $T(t) - T_g$. In order to encourage the minimization even more we use a quadratic difference function to measure the distance between the room `T` and the goal `Tg` temperatures, and then integrate it to achieve a distance function over complete trajectories. Conveniently, our distance function is modelled using differential equation in Fig. 1d as a separate process. Before we synthesize anything, we can inspect how does a uniform random choice fare in Fig. 2a: the temperature curve is at the top and heating and window mode trajectories are below and they jump up when the heating is on and window is open respectively. The result is that the room temperature responds to the mode changes and varies widely, tending to overshoot above the goal, and hence the distance function after 24 h period is about 4200 on average. In order to synthesize a strategy we pose the following query in Uppaal Stratego:

```
strategy opt = minE (D) [<=24*h]: <> t==24*h
```

which asks to find the strategy that will minimize the expected value of `D` when we reach a state with `t==24*h` while considering simulations of up to `24*h` in duration. Once the strategy is available, we may inspect it by requesting a simulation plot:

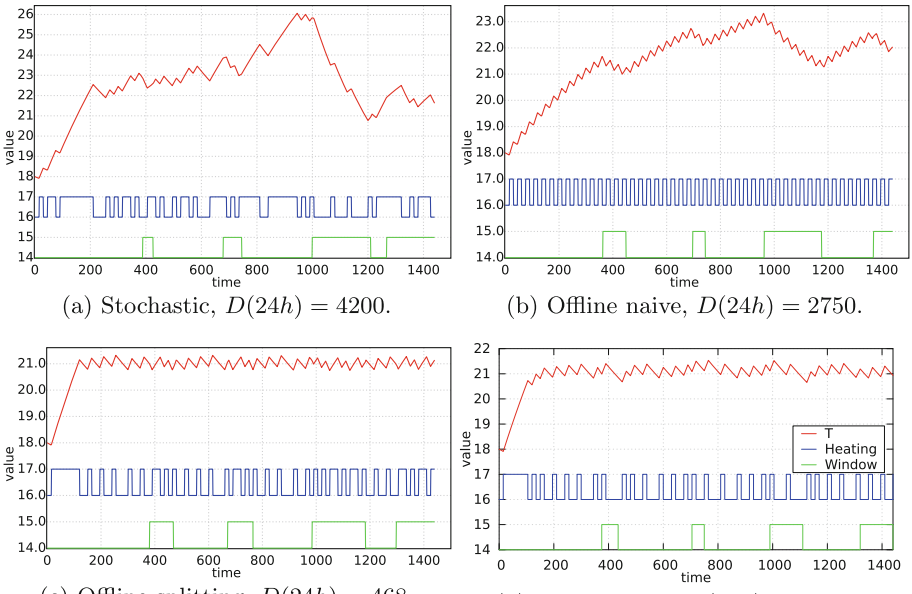
```
simulate 1 [<=24*h] {T,Window.Open+14,Room.HeatOn+16} under opt
```

For example, the synthesized 24 h strategy using the "naive" learning method yields the distance of 2750 on average as shown in Fig. 2b. The result is even more improved by the "splitting" learning method in Fig. 2c where the temperature oscillates around the goal very closely. ◁

Uppaal Stratego offers four learning methods focusing on various parts of the model, therefore we consider the quality and the cost of each method before we focus on our industrial-scale example. Table 1 shows a summary of the evaluation of various methods on two variants of a one-room example: the purely dynamical model is shown in Fig. 1 and another one that has an extra counter incremented at each period `P`. The result is that among the offline methods (discussed so far) the "splitting" method provides the smallest distance solution, however it is costlier than others in CPU time and memory. The right side Table 1 shows that if we add a period counter to our model, then other methods dominate and the "splitting" method is no longer as good and the "naive" computation costs significantly less. Offline-6 section (strategy for six days) requires twice as many resources as offline-3 (strategy for three days) which means that a linear number of resources is needed in terms of duration of the strategy while using the same number of runs, but the quality (distance) degraded almost four times with a period counter.

2.1 Online Synthesis

Uppaal Stratego [5,6] provides a method for approximating $\mathbb{E}^{\mathcal{G},\gamma}_{\sigma,H}(D) \in \mathbb{R}_{\geq 0}$ by computing a near-optimal strategy σ^H for a given horizon H using



(a) Stochastic, $D(24h) = 4200$.

(b) Offline naive, $D(24h) = 2750$.

(c) Offline splitting, $D(24h) = 468$.

(d) Online naive, $D(24h) = 471.5$.

Fig. 2. One-room 24 h trajectories of various control strategies

reinforcement learning. However, the effort needed to learn the strategy σ^H with a desired precision and confidence-level grows exponentially in the number of dimensions (variables). The quality of the learned control degrades sharply after the control options outnumber the number of simulation runs during learning, making this direct application of Uppaal Stratego limited in the time horizon. For instance, given a realistic setting of eleven heating switches as considered in our case study, the controller is faced with $2^{11} = 2048$ distinct options at each 15 min period and thus Uppaal Stratego manages to compute sensible heating configurations only for the first two periods (yielding $2048^2 = 4194304$ combinations in total) and then it simply resolves to default option of no heating at all.

Instead of learning—at great computational expense—the entire strategy σ^H, we propose a method for attentively and online (i.e. while playing the game $\mathcal{G}$ in the real setting) to compute a near-optimal strategy for controllable mode-change at the next period. More precisely, the resulting online and periodic strategy σ^O will base the mode-change at time $n \cdot P$ not only on the configuration at that point (γ_n) but also on the configuration (γ_{n-1}) at time $(n-1) \cdot P$[5], which will be used as the basis for online learning of short-horizon $(h << H)$ strategies. Formally:

$$\sigma^O(\gamma_{n-1}, \gamma_n) =^{\text{def}} \mathsf{let}\ \left(\sigma^h = \mathsf{argmin}_\sigma \mathbb{E}^{\mathcal{G},\gamma_{n-1}}_{\sigma,h}(D)\right)\ \mathsf{in}\ \sigma^h(\gamma_n)\ .$$

[5] Note that there may be several configurations between γ_{n-1} and γ_n due to the environment $\mathcal{U}$ changing the uncontrollable mode.

Table 1. Performance evaluation of one room controller synthesis: offline-3(-6) methods synthesize strategy for entire 72 h (144 h respectively) at once, strategy distance is evaluated on 70 simulations; online-3 methods synthesize a strategy for 5 periods of 15 min ahead and repeat synthesis and execution until 72 h are covered, the distance is averaged over 70 online simulations.

	Synthesis method	Purely dynamical model			Extra period counter		
		Distance	cpu,s	mem,kB	Distance	cpu,s	mem,kB
Offline-3	naive	10227.8	1555.15	11884	3671.84	566.04	9448
	splitting	517.9	1640.06	13424	2361.80	1608.48	90740
	covariance	10227.8	1298.66	11896	1091.81	1668.45	22820
	regression	10227.8	1368.34	11480	1387.84	1767.50	19196
Offline-6	naive	19668.7	1855.36	11836	8032.86	1316.08	20820
	splitting	593.7	3200.38	13112	8260.19	3120.03	167308
	covariance	20234.3	2039.30	11528	2468.91	3258.09	39580
	regression	19007.2	2525.13	12148	3425.62	3488.26	28416
Online-3	naive	584.7±1.0	1046.5±5.0	7240	526.6±0.5	1227.1±3.2	7328
	splitting	547.7±0.6	1136.4±3.6	7384	526.1±0.6	1240.8±2.5	7384
	covariance	587.5±1.2	1084.0±3.9	7272	527.1±0.6	1158.5±2.5	7624
	regression	585.3±1.0	1173.9±3.4	9052	527.9±0.5	1337.1±2.5	7380

We leave the formal definition of runs under the one-step-memory strategy σ^O to the reader (slightly more complicated version of runs under a memoryless strategy given above). However, we note that σ^O may be used for an arbitrary finite horizon H or even as a strategy of infinite horizon. To maximize the quality of σ^O, the choice of the small horizon h should be such that it *just* allows the learning of σ^h to be completed between the two configurations γ_{n-1} and γ_n, i.e. within the period P.

Example 3. We implemented the online strategy evaluation on the one-room example by repeatedly calling Uppaal Stratego to synthesize and evaluate the computed strategy. The following steps are involved:

1. Synthesize a strategy capable of adapting for 5 periods ahead where `LastTime` starts with 0: `strategy S = minE (D) [<=5*P]: <> t==LastTime+5*P`
2. Simulate the system for 1 period using the strategy `S` and record its last state: `simulate 1 [<=P] { t, T, Room.HeatOn, x, Window.Open, w, i, D }`
3. Create a copy of the original model and replace the initial state with the recorded state from the simulation above.
4. Increment `LastTime` by `P` and repeat the synthesis and simulation from step 1 until a required number of periods is simulated.
5. Record the final value of the distance variable `D`.

The short trajectories from step 2 are then stitched together to produce a continuous trajectory of the entire 3 day simulation. An example result of the first 24 h is displayed in Fig. 2d which is also comparable to other strategies. The online-3 section of Table 1 shows the averages of the recorded distances together with the overall synthesis effort for entire 3 day emulation. The encouraging result is that the short strategy synthesis takes only 4–8 s and the overall quality of any online

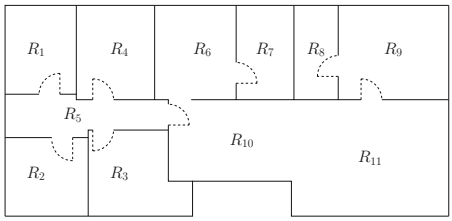


Fig. 3. Plan of the house

synthesis method is very close to the best and expensive offline-3 outcome (the offline "splitting" method). ◁

3 Floor Heating Case Study

In Fig. 3 we see the plan of the house on which we will optimize the heating strategy.

The house consists of 11 rooms, all of them heated with a floor heating system where each room has its own pipe circuit that can be either open (hot water circulates) or closed (water does not circulate). The opening and closing of the circuits is executed by a number of valves located in room R_7. Every 15 min a wireless temperature sensor in each room wakes up and reports its current reading. Currently the bang-bang strategy runs every 15 min: it collects the temperatures of all rooms, if a given room temperature is below its target temperature (setup by the user) it opens the corresponding valve and similarly if the temperature is above target it closes the valve.

The problem with this controller, as experienced by the house owner, is that it completely disregards the thermodynamics of the house, the outside temperature (and weather forecast) as well as the maximal capacity of the floor heating system. We now outline the factors affecting the heating system in this house:

1. *Heating capacity of the system.* The heating system can only provide a limited water pressure to make the water circulate within the pipes. If too many valves are open, the water will only cycle in the shortest pipes. This is especially a problem in the living room R_{11}, as it has the longest pipe circuit, and is also the most important room for the user, meaning that the temperature of this room should be maintained close to the user's wish. A smart heating system should take the heating capacity into a consideration and never exceed it.
2. *Behaviour of the doors.* The heat exchange between rooms are significantly affected by whether doors between the rooms are open or closed. The house is not equipped with door sensors, so the position of each door is unknown. This means that the control strategy has to work under a partial observability and the status of each door can be inferred only indirectly by observing the speed of heat propagation via temperature changes in the rooms.

3. *Physical layout of the pipe circuits.* Finally, as the valves are all located in room R_7, the pipes leading to some of the remote rooms necessarily pass under under rooms. Hence e.g. opening a valve for the room R_2 will contribute also to minor increase of the room temperature in rooms R_3, R_5, R_4 and R_6 under which the pipe circuit is placed.

In our thermodynamic model of the floor heating, we take all these factors in consideration. The aim of the controller program is to optimize the user comfort and satisfaction according to some measure of how far the actual temperatures are from some goal temperature.

Floor Heating Scenario as a Stochastic Hybrid Game. The floor heating scenario with n rooms and m doors is a stochastic hybrid game $\mathcal{G}_{n,m} = (\mathcal{C}, \mathcal{U}, X, \mathcal{F}, \delta)$, where the controller $\mathcal{C}$ has a finite set of controllable modes $V = \mathbb{B}^n$ given by all possible valve opening/closing combinations. The environment $\mathcal{U}$ has a finite set of uncontrollable modes $D = \mathbb{B}^m$ given by all possible door opening/closing combinations. We assume that $\mathcal{U}$ given δ can switch among modes with equal probability at every period. The state variables in X are given by the room temperatures $\{T_1, \ldots, T_n\}$ and the outside temperature T_{env}.

We will denote by vector T the room temperatures and by T_i the i-th room temperature. Given the current temperatures T, a controllable mode $v \in V$, an uncontrollable mode $d \in D$ and a time delay τ, the flow function $\mathcal{F}_{v,d}(\tau, T)$ gives the room temperatures T' (after τ time units passed) that are the solutions to the following differential equations:

$$\frac{d}{dt}T_i(t) = \sum_{j=1}^{n} A^d_{i,j}(T_j(t) - T_i(t)) + B_i(T_{\mathsf{env}}(t) - T_i(t)) + H^v_{j,i} \cdot v_j \; dt$$

where $A^d_{i,j}$ contains the heat exchange coefficients between room i and room j, given the door mode d. Note that there are 2^m matrices for the possible door modes. The vector B contains the heat exchange coefficients between the outside temperature and each room, and H^v contains the heat exchange coefficients for each pipe and the rooms it traverses. A pipe heats a room if it traverses it and valve v_j is open. There is a capacity constraint on the water pressure, if the capacity is exceed the coefficients in H^v prevent the rooms with the long pipes from been heated. Finally, $T_{\mathsf{env}}(t)$ is the current outside temperature at time t. The initial conditions are given by the current temperatures T. Hence, for a given room i, the temperature T'_i is influenced by the adjacent rooms, the door configuration (uncontrollable mode), the outside temperature T_{env}, the pipes traversing the room, and the valve configuration (controllable mode). For the thermodynamics to be realistic, the time unit is minutes.

3.1 Experiments

Regarding our experiments, we have two major components: a simulator written in MATLAB and a number of controllers, including the ones produced by UPPAAL

STRATEGO. The simulator implements the floor heating stochastic hybrid game $\mathcal{G}_{n,m}$. For our experiments, in the simulator we fix a time horizon H of 3 days with a period P of 15 min. As in the real house, every 15 min, the simulator outputs the current room temperatures T which are read by the controller. Subsequently, the controller inputs the control valves V which are used by the simulator for the next 15 min. The house has vectors of desired temperatures T^g and weights W denoting the importance of each room. Our goal is to optimize the comfort in the house. Intuitively, comfort is in proportion to the distance between the desired temperatures and the current temperatures. To measure the comfort provided by a controller (strategy) σ, we define a function dist on runs of $\mathcal{G}_{n,m} \upharpoonright \sigma$ of the form $\pi = \gamma_o \xrightarrow{t_1} \gamma_1 \xrightarrow{t_2} \ldots \xrightarrow{t_{k-1}} \gamma_{k-1} \xrightarrow{t_k} \gamma_k$ where $k = H/P$ is the number of control steps in the run π. Let $T_i(\gamma_j)$ denote the room temperature T_i at configuration γ_j. Then the distance function is defined by

$$\mathsf{dist}(\pi) = \sum_{j}^{k} \sum_{i}^{n} (T_i^g - T_i(\gamma_j))^2 \cdot W_i \ .$$

In our experiments, we evaluate a number of different controllers. The simulator uses the distance function dist to compare the different controllers.

Controllers. In the following we introduce a number of controllers which we use in our experiments. We present the current controller operating in the house, two controllers proposed by engineers and the controller synthesized using online synthesis and UPPAAL-STRATEGO.

- *Bang-Bang Controller.* The bang-bang controller is currently running in the physical house and after each reading of room temperatures T, it simply opens the valves of every room i where $T_i < T_i^g$ and leaves the remaining valves closed.
- *Capacity Aware Bang-Bang Controller.* The main problem with the bang-bang strategy is that if all rooms are below their target temperatures, it simply opens all valves in the house, violating the restriction on the maximal capacity of the floor heating system. The capacity aware bang-bang controller, at each time where a decision is to be taken, orders in descending order all rooms according to their individual distance function, given for a room i by $W_i \cdot (T_g^i - T^i)^2$ where W_i is the given priority of room i, and then opens in this order the valves of all rooms that are below their target temperatures (as the normal bang-bang controller) but only until the maximum capacity is exceeded.
- *Brute-Force Controller.* This is an online controller with short horizon 1 that for n valves by brute-force explores all possible 2^n valve combinations and selects a valve combination that minimizes the distance function. The controller operates as follows: after the current reading of the temperatures T and the valves configuration v, it guesses a random door mode d and using this information it computes the expected temperatures T' exactly after P

time units (recall that in our case study we fixed the period to $P = 15$); next the controller considers all 2^n possible valve configurations and computes the predicted room temperatures T'' at time $2P$. The controller then returns the valve configuration that minimizes the distance function dist. Note that already for the short horizon 1, the computation of the brute-force controller takes over 170 seconds, so exploring by brute-force all 2^{2n} combinations (in our case $n = 11$) needed for the short horizon of 2 is impossible due to the 15 min duration of the period.

– *Stratego Online Controller.* (Stratego-ON) The controller is synthesized by Uppaal Stratego using the online strategy synthesis methodology introduced in Sect. 2 with short horizon of 3. The aim is to learn the optimal valve configurations for several steps ahead using machine learning methods and hence avoid the exhaustive search done by the brute-force controller.

Evaluation Scenarios. In order to evaluate the performance of the different controllers described above, we fix five realistic scenarios on which we perform our experiments. We distinguish between the *stability* scenarios where the initial room temperatures are equal to the target ones ($T(0) = T^g$) and the task is to maintain these temperatures throughout the next three days relative to different weather conditions. We also study the *vacation* scenarios, where we assume that a family returns from a vacation and shortly before this the house should move from the energy-saving temperature vector $T(0)$ into the target temperature T^g vector as quickly as possible.

The stability and vacation scenarios are then subject to two different weather profiles, a *mild winter* where the outside temperature behaves according to real data from the Aalborg airport from 03.02.2015, 00:20 to 07.02.2015, 23:50 where the outside temperature ranges between 2 to 5 °C, and a *tough winter* using the data from the Aalborg airport from 14.02.2015, 00:20 to 17.02.2015, 23:50 where the outside temperature ranges between -10 and 6 °C. We also consider the *spring* scenario where the outside temperature is modelled using a sinusoid $T^{\mathsf{env}}(t) = 7 * sin(2 * pi/60 * 24 * t) + 19$ such that most of the time the outside temperature is below the target room temperatures but the peak environment temperature during the middle of the day exceeds the target room temperatures.

In all scenarios, a fixed profile when a specific door is closed or open is used, corresponding to the typical behaviour of the owner of the house. Note that none of the controllers is aware of this fixed door profile.

Controller Evaluation for 5 Rooms. We show the applicability of our online-synthesis methodology on the left part of the house consisting of rooms R_1 to R_5 and doors D_1 to D_4 (see Fig. 3), i.e. the stochastic hybrid game $\mathcal{G}_{5,4}$. We have restricted the maximum pressure capacity of the heating system to 50 %. In our simulator for $\mathcal{G}_{5,4}$, we executed all the above controllers and scenarios. The evaluation of the controllers is given in Table 2. Since we have fixed a door profile and the controllers are deterministic (except for Stratego-ON), we obtain a unique run π for every combination of scenarios and controllers.

Table 2. Evaluation of controllers for 5 and 11 rooms of the house (see Fig. 3). The simulation has a horizon H of 3 days, and a short horizon h of 3 periods. Temperatures are read every 15 min.

Scenario	Controller	5 Rooms dist	5 Rooms Time (sec.)	11 Rooms dist	11 Rooms Time (sec.)
mild winter vacation	Bang-Bang	62704	< 1	53550	< 1
	Bang-Bang-Cap-Aware	39755	< 1	31718	< 1
	Brute-Force	36489	~ 2.4	28332	~ 171
	Stratego-ON	36418	~ 2.9	31054	~ 77
	Stratego-ON-CL	—	—	29541	~ 16
tough winter vacation	Bang-Bang	248367	< 1	163635	< 1
	Bang-Bang-Cap-Aware	155090	< 1	82250	< 1
	Brute-Force	137266	~ 2.4	61897	~ 171
	Stratego-ON	137223	~ 3.0	75792	~ 78
	Stratego-ON-CL	—	—	66611	~ 17
mild winter stability	Bang-Bang	24834	< 1	9654	< 1
	Bang-Bang-Cap-Aware	18405	< 1	9430	< 1
	Brute-Force	16765	~ 2.4	9260	~ 179
	Stratego-ON	16708	~ 3.0	9972	~ 76
	Stratego-ON-CL	—	—	9025	~ 16
tough winter stability	Bang-Bang	199688	< 1	82849	< 1
	Bang-Bang-Cap-Aware	121776	< 1	37099	< 1
	Brute-Force	107065	~ 2.2	33917	~ 192
	Stratego-ON	107027	~ 3.0	42229	~ 77
	Stratego-ON-CL	—	—	34585	~ 16
spring stability	Bang-Bang	4297	< 1	4493	< 1
	Bang-Bang-Cap-Aware	4297	< 1	4419	< 1
	Brute-Force	3875	~ 2.2	2861	~ 171
	Stratego-ON	3755	~ 2.8	3239	~ 50
	Stratego-ON-CL	—	—	2819	~ 16

For a controller, the column dist is the accumulated distance $\mathsf{dist}(\pi)$ between the current temperatures and the desired temperatures during the 3 day simulation. We observe that in all scenarios, the online controller Stratego-ONhas the minimal distance, providing the best comfort among all the controllers. Our final goal is to synthesize a controller for the full house with 11 rooms. However, the corresponding state space hinders online strategy synthesis to scale with satisfactory quality of the produced control strategy. We address this issue in the next section.

4 Compositional Synthesis

Although online learning is an important step towards the scalability of our approach, it does not enable us to learn small horizon strategies of sufficient quality for the full version of the floor heating case study. Even though we have decreased the horizon, the branching factor is enormous: for each period we have to learn the optimal setting of 11 valves, i.e. the optimal of 2^{11} modes. Given a horizon h, this means that we have to learn the optimal sequence of modes out of 2^{11h} possible sequences. Clearly, this becomes infeasible for small h.

However, often the set of modes C will be a product of two (or more) sub-modes, i.e. $C = C_1 \times C_2$; e.g. in the floor heating case study we may split the

11 valves into two subsets i.e. valves 1 to 5 and valves 6 to 11. This suggests the possibility of a compositional approach for the synthesis of σ^h based on the synthesis of two sub-strategies $\sigma_1^h : \mathbb{C} \to C_1$ and $\sigma_2^h : \mathbb{C} \to C_2$, with $\sigma^h(\gamma) = (\sigma_1^h(\gamma), \sigma_2^h(\gamma))$.

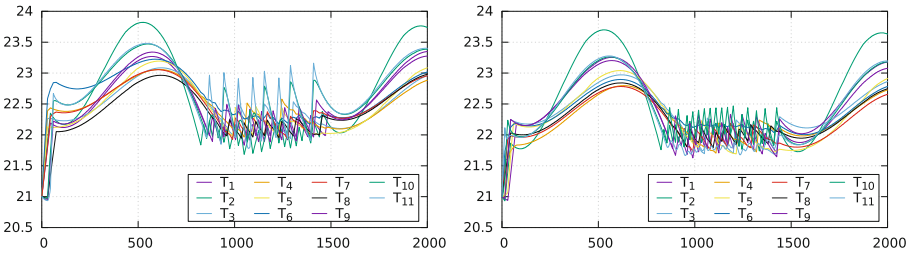
Given an initial sub-strategy $\sigma_1^0 : \mathbb{C} \to C_1$, the game $\mathcal{G}$ becomes a reduced game $\mathcal{G} \restriction \sigma_1^0$ with C_2 as remaining controllable modes. With the significant reduction in size, it may now be feasible to synthesize a near-optimal strategy, $\sigma_2^0 : \mathbb{C} \to C_2$, with horizon h for this reduced game, i.e. $\sigma_2^0 = \mathsf{argmin}_\sigma \mathbb{E}_{\sigma,h}^{\mathcal{G}\restriction\sigma_1^0,\gamma}(D)$. Now given σ_2^0, we may similarly learn an optimal sub-strategy, $\sigma_1^1 : \mathbb{C} \to C_1$, for the reduced game $\mathcal{G} \restriction \sigma_2^0$ with C_1 as remaining controllable modes. Repeating this process will generate a sequence of sub-strategies $\sigma_1^i : \mathbb{C} \to C_1$ and $\sigma_2^i : \mathbb{C} \to C_2$, with $\sigma_1^h = \sigma_1^N$ and $\sigma_2^h = \sigma_2^N$ for some a priori chosen N. Clearly, this method is a heuristic, with no guarantee of converging to the optimum overall strategy, and where the quality depends on the initial sub-strategy chosen, the choice of N as well as the game $\mathcal{G}$ itself. However, as we shall see, this heuristic may be with success applied to our floor heating case study.

Stratego Online Compositional Controller. (STRATEGO-ON-CL) This controller applies the previously introduced compositional synthesis together with online synthesis. The controller uses two UPPAAL STRATEGO models. In the first model, valves 1 to 5 are controllable and valves 6 to 11 are fixed by a Bang-Bang controller (the second model is constructed in a dual manner where the valves 1 to 5 are now fixed to the computed control strategy in the first model). At every period, distributing the valve capacity between the left and right parts of the house plays a key role. This controller dynamically assigns the maximum allowed capacity for the two parts of the house proportionally to the distance function dist of the two parts of the house.

Experiments for 11 Rooms. We implemented the floor heating stochastic hybrid game $\mathcal{G}_{11,8}$ with 11 rooms and 8 doors in our simulator and evaluated the Stratego compositional controller together with the previously defined controllers and all the scenarios described in Sect. 3.1. Table 2 presents the results. We observe that for all scenarios the Stratego online compositional controller obtains results comparable to the Brute-Force controller, however, by an order of magnitude faster.

In order to see how the STRATEGO-ON-CL controller can take weather information into account, consider Fig. 4 that illustrates the spring stability scenario. From points of time between 0 and 500 min, the outside temperature increases and exceeds the target temperature. We observe that since the STRATEGO-ON-CL controller is able to look at the weather forecast for the next 45 min, it shuts down the valves much earlier than the other controllers. This results in energy savings and increased comfort.

Comparing the Brute-Force controller with STRATEGO-ON-CL, we can see that in the vacation scenarios and tough winter scenario STRATEGO-ON-CL performs with a slightly larger discomfort due to the fact that the goal is to heat up all the rooms as quickly as possible and hence looking more time periods



(a) Bang-Bang controller (b) Stratego-ON-CL controller

Fig. 4. Room temperatures in the spring stability scenario

into the future does not help (there is only little risk of overshooting the target temperatures). On the other hand, in the remaining scenarios where looking more steps into the future can have an effect on the selected control strategy, Stratego-ON-CL has a slightly better performance. Nevertheless, Stratego-ON-CL is a clear winner in terms of the time needed to compute the strategy which will be particularly important when moving to even larger case studies.

5 Conclusion

In the floor heating case study we evaluated the existing Uppaal Stratego controller synthesis techniques and showed its limitations when applied on industrial scale models. In order to solve the scalability issues, we proposed online framework to compute and combine the short-term control strategies iteratively on demand, while connected to the real house heating system. In addition, we proposed a compositional methodology in order to scale the synthesis for more rooms needed in our real scenario. The experimental evaluation showed that the resulting strategies are outperforming the presently used controller and while comparable in performance to the Brute-Force controller, our method can compute the control strategy by an order of magnitude faster. Hence the developed framework is suitable for installation at home automation systems and we have already constructed a scaled physical model of the house with the actual hardware used by the company Seluxit, as a first step towards the industrial employment of our methodology in their products.

Acknowledgments. The research leading to these results has received funding from the EU FP7 FET projects CASSTING and SENSATION, the project DiCyPS funded by the Innovation Fund Denmark, the Sino Danish Research Center IDEA4CPS and the ERC Advanced Grant LASSO. The fourth author is partially affiliated with FI MU, Brno, Czech Republic.

References

1. Caccamo, M., Frazzoli, E., Grosu, R. (eds.): Proceedings of the 14th ACM International Conference on Hybrid Systems: Computation and Control, HSCC 2011, Chicago, IL, USA, 12–14 April 2011. ACM (2011)
2. Camacho, E., Ramirez, D., Limon, D., de la Pea, D.M., Alamo, T.: Model predictive control techniques for hybrid systems. Ann. Rev. Control **34**(1), 21–31 (2010). http://www.sciencedirect.com/science/article/pii/S1367578810000040
3. David, A., Du, D., Larsen, K.G., Mikučionis, M., Skou, A.: An evaluation framework for energy aware buildings using statistical model checking. SCIENCE CHINA Inf. Sci. **55**(12), 2694–2707 (2012). http://dx.doi.org/10.1007/s11432-012-4742-0
4. David, A., Du, D., Guldstrand Larsen, K., Legay, A., Mikučionis, M.: Optimizing control strategy using statistical model checking. In: Brat, G., Rungta, N., Venet, A. (eds.) NFM 2013. LNCS, vol. 7871, pp. 352–367. Springer, Heidelberg (2013)
5. 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: Cassez, F., Raskin, J.-F. (eds.) ATVA 2014. LNCS, vol. 8837, pp. 129–145. Springer, Heidelberg (2014)
6. David, A., Jensen, P.G., Larsen, K.G., Mikučionis, M., Taankvist, J.H.: Uppaal stratego. In: Baier, C., Tinelli, C. (eds.) TACAS 2015. LNCS, vol. 9035, pp. 206–211. Springer, Heidelberg (2015)
7. Fehnker, A., Ivančić, F.: Benchmarks for hybrid systems verification. In: Alur, R., Pappas, G.J. (eds.) HSCC 2004. LNCS, vol. 2993, pp. 326–341. Springer, Heidelberg (2004)
8. Hahn, E.M., Norman, G., Parker, D., Wachter, B., Zhang, L.: Game-based abstraction and controller synthesis for probabilistic hybrid systems. In: Eighth International Conference on Quantitative Evaluation of Systems, QEST 2011, Aachen, Germany, 5–8 September 2011, pp. 69–78. IEEE Computer Society (2011). http://dx.doi.org/10.1109/QEST.2011.17
9. Larsen, K.G., Mikucionis, M., Taankvist, J.H.: Safe and optimal adaptive cruise control. In: Meyer, R., et al. (eds.) Olderog-Festschrift. LNCS, vol. 9360, pp. 260–277. Springer, Heidelberg (2015). doi:10.1007/978-3-319-23506-6_17
10. Majumdar, R., Render, E., Tabuada, P.: Robust discrete synthesis against unspecified disturbances. In: Caccamo et al. [1], pp. 211–220. http://doi.acm.org/10.1145/1967701.1967732
11. Roy, P., Tabuada, P., Majumdar, R.: Pessoa 2.0: a controller synthesis tool for cyber-physical systems. In: Caccamo et al. [1], pp. 315–316. http://doi.acm.org/10.1145/1967701.1967748
12. Sørensen, M.G.: Automated controller synthesis in home automation. Master's thesis, Computer Science, Aalborg University (2014)
13. Svoreňová, M., Křetínský, J., Chmelík, M., Chatterjee, K., Černá, I., Belta, C.: Temporal logic control for stochastic linear systems using abstraction refinement of probabilistic games. In: Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control, pp. 259–268. HSCC 2015, ACM, New York, NY, USA (2015). http://doi.acm.org/10.1145/2728606.2728608
14. Wognsen, E.R., Haverkort, B.R., Jongerden, M., Hansen, R.R., Larsen, K.G.: A score function for optimizing the cycle-life of battery-powered embedded systems. In: Sankaranarayanan, S., Vicario, E. (eds.) FORMATS 2015. LNCS, vol. 9268, pp. 305–320. Springer, Heidelberg (2015)