Safe and near optimal controller synthesis for Stochastic Hybrid Games

Richard Valentín Yantas Alcantara

Advisor: PhD. Marco Muñiz

Committee Members:

Thesis submitted to the
Department of Computer Science
in partial fulfillment of the requirements for the degree of
Master in Computer Science.

Universidad Católica San Pablo – UCSP
September of 2021 – Arequipa – Peru

To my parents, Mario and Maria, who never stop giving of themselves in countless ways. To my brothers and sister, who encourage and support me.

Abbreviations

SHG Stochastic Hybrid Game

SWH Solar Water Heating

MPC Model Preodictive Controller

MDP Markov Decision Process

SS Switched System

Acknowledgments

First and foremost, I want to thank God for having guided me throughout these two years of study.

I would like to thank my family for their continuous support and encouragement during these years of study.

I would like to thank in a special way to the National Council for Science, Technology and Technological Innovation (CONCYTEC-PERU) and to the National Fund for Scientific Development, Technological and Technological Innovation (FONDECYT-CIENCIACTIVA), which through the Management Agreement 234-2015-FONDECYT have allowed the grant and financing of my studies in the Master Program in Computer Science at Universidad Católica San Pablo (UCSP).

I would like to express my gratitude and appreciation to my advisor Marco Muñiz for giving his guidance and support throughout the Master program and the preparation of this thesis.

Finally, I thank all the people who directly or indirectly helped me in the preparation and presentation of this work, either by exchanging ideas, giving me advice and recommendations or by encouraging me to continue.

Abstract

Stochastic hybrid systems are widely used in the energy industry, problems such as optimization and safety are some of the most important. We implement a methodology to improve a real time stochastic hybrid system combining model checking, reinforcement learning and boundary computation. We apply formal methods to synthetize (i.e derive automatically) a safe and near optimal controller. To verify the methodology, we model a solar water heating, including the human interaction to access the water in it as stochastic. Additionally, we use the tool Uppaal Stratego to synthetize a control strategy minimizing a cost function composed by energy consumption(*optimization*) and temperature distance (*safety*) while the last one also guaranteeing no overflow a max and min temperature defined previously.

Keywords: Stochastic Hybrid Systems, Machine Learning and Model Checking.

Resumen

Los sistemas híbridos estocásticos son ampliamente usados en la industria energética, problemas como optimización y seguridad son algunos de los mas imporantes. Implementamos una metodologia para mejorar un sistema hibrido estocastico en tiempo real combinando verificacion de modelos, aprendizaje por refuerzo y computacion de vector de intervalos. Aplicamos metodos formales para sintetizar (ej. derivar automaticamente) un controlador seguro y cercano al óptimo. Para verificar la metodología Se modeló un calentador de agua solar híbrido, que incluye la interacción humana con el sistema para su consumo de agua caliente, como una variable estocástica. Adicionalmente, se empleo Uppaal Stratego como una herramienta de optimización que nos permite sintetizar una estrategia de control optimo, minimizando la distancia de separación de la temperatura deseada con la controlada asi como garantizar limites para su evolución en el tiempo.

Palabras clave: Sistemas Híbridos Estocásticos, Aprendizaje Automático y Verificación de Modelos.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this thesis, we present a control synthesis method for a special form of hybrid systems with stochastic perturbations that we model as a SHG. Such models have been recently used in various domains such as solar energy industry and power electronics (e.g power converters). They are continuous-time systems with discrete switching events. More precisely, these systems are described by piecewise continuous dynamics called *modes*; the change of modes takes place instantaneously at so-called *switching instants*. We suppose that switching instants occur *periodically* at constant sampling period τ *sampled switching systems*. The *control synthesis* problem consist in finding a *switching* rule σ in order to satisfy a given specification. At each sampling time $\tau, 2\tau, ..$, according to the state of the system, the rule σ selects an appropiate mode to fulfill the specification. SHG has challenging problems such as optimization and safety that are some of those which hybrid system community are focused. For this reason, we implement a methodology to improve a real time SHG that combines model checking, reinforcement learning and constraint computation. We use formal methods to synthetize a safe and near optimal controller. In Section 4.1, we model a Solar Water Heating (SWH), including the human interaction to access the water in it as stochastic. In Section 3.3, we define use the tool Uppaal Stratego to synthetize a control strategy minimizing a cost function composed by energy consumption (*optimization*) and temperature distance (*safety*) while the last one also guaranteeing no overflow a max and min temparature defined previously. In Section 3.2, We define a SHG

1.1 Motivation and Context

Stochastic Hybrid systems are widely used in engineering applications and its importance has grown up considerably these last years, because of their ease of implementation for controlling cyber-physical systems. A SHG is a Stochastic hybrid model which represent a set of dynamical systems supervised by a switching rule σ, whose values are in a finite set $\mathbb{C}$(See Liberzon (2003)) and a stochastic perturbation u, which together and also with the state variable x determine a configuraion $\gamma = (c, u, x)$ as defined in Section 3.2. However, due to the composition of many switched systems together with stochastic perturbation, the global switched systems has ups and downs over each variables in the state without boundaries thus a risk in the system behaviour and the environment. For this reason, in order to guarantee to synthesize a safe and optimal strategy in each control step followed by a safe and near optimal control a long the real time system behaviour. SHG have numerous applications in control of mechanical systems, the automotive industry, and many other fields.

1.2 Problem Statement

Nowadays, there is a large number of methods to face issues related to switched systems; however in case of stochastic hybrid systems we notice some advances either only optimization or only safety but none of them propose a method that concatenate these.

1.3 Objectives

General Objective

Our main objective is to propose a methodology to synthesize switched systems controller that guarantee safety and near optimal switching in real time.

Specific Objectives

To achieve our main objective, we have the following specific objectives:

- Define a case of study as a stochastic hybrid game and get its mathematical model.
- Implement and explore safe patterns for the model.
- Implement an API between the simulator and UPPAAL to Optimize the model.
- Compare results with traditional methods with our methodology.

1.4 Contributions

This thesis proposes a novel approach to solve switched systems guaranteeing safety and optimal controller synthesis.

- Find safe patterns for our case study using decomposition algorithm (Le Coënt et al. (2017)).
  - This paper implement a method to guarantee safe controller. This methods consider three regions to have reachability and safety in the system.
  - This also demonstrate the utility of the proposed method comparing with traditional methods.
- Find a near optimal patterns to online control(Larsen et al. (2016)).
  - Implement an API between simulator and UPPAAL.
  - Build a timed automata Markov Decision Process (MDP) in UPPAAL for a real time MPC.

1.5 Outline

This thesis document is divided into five chapters. After this introduction and problem formulation, in Chapter 2 we survey the literature about the recent research in safety controllers and near optimal online controller synthesis. Chapter 3 presents some basic concepts about the hybrid systems, traditional controllers, stability criteria and optimal controller technique. Next, in Chapter x we describe in detail the corpus, techniques used by our pipeline and their evaluation results. Finally, the limitations, future works, and conclusions of this work are presented in Chapter.

Chapter 2

Related Works

Our work draws on prior research in the areas of hybrid systems, safe patterns and online control synthesis.

2.1 Online and Compositional Learning of Controllers with Application to Floor Heating

This work has proposed one method to perform *optimal controller synthesis for stochastic hybrid games* SHG e.g. a floor heating system in a house Larsen et al. (2016). Is proposed 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, is proposed an on-line synthesis methodology, where it periodically compute the controller only for the near future based on the current sensor readings. UPPAAL, is an integrated tool environment for modelling, validation and verification of real-time systems modeled as networks of timed automata, extended with data types. STRATEGO facilitates generation, optimization, comparison as well as consequence and performance exploration of strategies for stochastic priced timed games in a user-friendly manner. The online approach may be seen as a instance of model predictive control MPC or receding time horizon control for hybrid systems. The application of MPC to hybrid systems subject to disturbances or uncertainties gives rises to uncertain optimization problems. Many robust MPC schemes are based on the min-max strategy originally.

2.2 An Improved Algorithm for the Control Synthesis of Nonlinear Sampled Switched Systems

In this paper is presented and algorithm for the control synthesis for nonlinear SS using and existing procedure of state-space and made available for nonlinear systems with the help of guaranteed integration, the algorithm has been improved to be able to consider longer patterns of modes with a better running approach. This approach permits to deal with stability, reachability, safety and forbidden region constraints. The approach was numerically validated on several examples taken from the literature Le Coënt et al. (2017). Additionally, the search of patterns that guarantee stability, reachability and safety is given by an *decomposition* and *find patterns* algorithms; the decomposition algorithm divide each box in small box in order to find pattern that satisfy stability definition at Section 3.5. This methods are widely discussed in (Fribourg et al., 2014) for instance operators such as *posteriors* and *tubes* that determine the state after τ time and sequences of states respectively. Some applications are also proposed the boost DC-DC converter, helicopter motion, two-room building heating and multilevel converter. The control with disturbances is one of the very interesting features in the algorithm due to is only necessary replace the disturbances with an interval with its respective zonotope to get the pattern. The zonotope is an small region that was divided with the objective to verify stability and safety, these zonotopes was proposed to as polytopes with several interesting properties: these can be encoded efficiently, *stability* property require *reachibility* property which is discussed and analized in (Girard, 2005)

2.3 Final Considerations

This chapter presented some recent proposals related to our thesis work. Some research works have been focused on analyzing safety controller over switched systems. On the other hand, other works have focused on optimize controller synthesis using some techniques related to model checking. The next chapter will present some concepts needed to understand better our work, those concepts are related to the modelling physical system over our case of study and represent as a state space.

Chapter 3

Background

In this chapter we present basic concepts that are needed to understand better our work. In Section 3.1 we describe definitions about switched systems and the difference between hybrid system a basic concept that we will use for our analysis, Sanfelice (2020). In Section 3.5 we explains the method to find patterns. In Section 3.3 we describe Uppaal Stratego. In Section 3.2 we describe a method to control stochastic hybrid games using online controller. In Section 3.6 we explain the methodology to join the safe and optimal approach. Lastly in Section 3.7, we show a general diagram that explain how the system works.

3.1 Switched systems

Switched systems are loosely defined as dynamical system whose state has two components, one of which evolves in a continuous set such as $\mathbb{R}$ while the other evolves in a discrete set such as $\mathbb{N}$ according to some transition logic based rule. Hybrid System is an abstraction of SS which is a system which changes acording to switch rule, this rule comprises of discrete events,(Branicky, 1994).

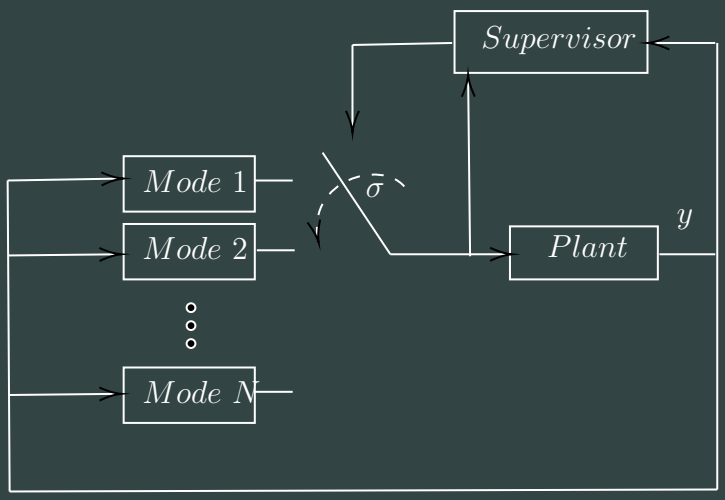


Figure 3.1: A control diagram of SS with a supervisor that determines a switching rule σ

Definition 1.(Hybrid System). A hybrid system $\mathcal{H}$ is a continuous-discrete mathematical abstraction of switched system that represents a tuple (τ, σ, x, f) where defined for all $t \geqslant 0$:
1. τ *is a time period for switching between modes*
2. $x \in \mathbb{R}^M$ *is a vector state of size* M
3. $\sigma \in \mathbb{R}^N$ *is a vector of* N *modes that determines a switching rule that will occur at time* $\tau, 2\tau, ..$ depend on time and state (SS only depend on time)
4. f *is a non-linear function subject to a switching rule* σ *and disturbances* $d(t)$

$$\frac{d}{dt}x(t) = f_\sigma(x(t), d(t)) \tag{3.1}$$

3.2 Online Stochastic Hybrid Controller Synthesis

We use a one-room heating control problem as a running example to demonstrate the technique 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 $15 minutes$ and then it has two options: either to turn the heater on (mode "HeatOn") and keep it there or switch the heater off (mode "HeaterOff"). 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 trough the walls and the window to the outside environment. In short, the temperature dynamics can be described by the following differential equation.

$$\frac{d}{dt}T(t) = (T_e(t) - T(t))A(t) + H(t) \tag{3.2}$$

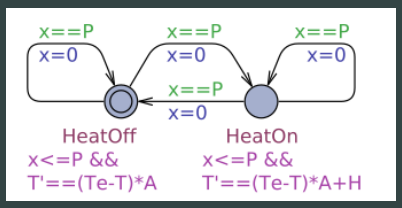
Where $T(t)$ is the room temperature at time $t, T_e(t)$ is the outdoor temperature, $A(t)$ is the heat exchange factor specific to the walls and windows, and $H(t)$ is the power of the heater.

Figure 3.2b 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 the guard $x == P$ over clock x with 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 v variables (their types and initial values) are declared as C structures in Figure 3.2a. The window step function $A(t)$ is modelled In Figure 3.2c 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

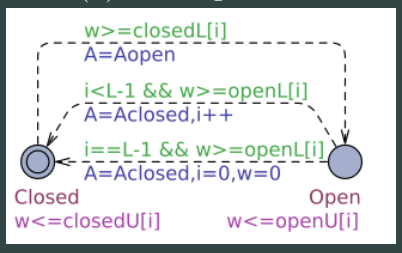
```
const double Tg = 21.0; // goal temp. in deg.C
const double Te = 15.0; // env. temperature
const double H = 0.04; // power of heater
const double Aclosed=0.002;// window closed
const double Aopen = 0.004;// window open
const int P = 15; // switching period, 15min
const int h = 60; // 1hour = 60min
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 };
```

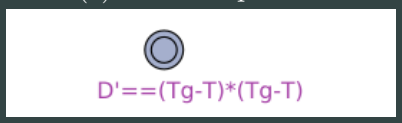
(a) Variable declarations.



(b) Heating modes.



(c) Window profile.



(d) Distance to goal temp.

Figure 3.2: Uppaal Stratego model of one room with one window

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 12 and 13, etc.

The whole system model is then a composition of the controlled heating process with the stochatic 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 2.(Stochastic Hybrid Game). A stochastic hybrid game $\mathcal{G}$ is a tuple $(\mathcal{C}, \mathcal{U}, \mathcal{X}, \mathcal{F}, \delta)$ 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. $\mathcal{X} = \{x_1, x_2, ...x_n\}$ *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,* $\delta_\gamma : U \to [0,1]$*, where* $\gamma = (c, u, v) \in C \times U \times \mathbb{R}^X$ *is a global configuration. More precisely,* $\delta_\gamma(u')$ *is the probability that*

$u \in \mathcal{U}$ in the global configuration $\gamma = (c, u, x)$ will change to the uncontrollable mode u' after a delay of τ [1].

We shall assume that among the continuous variables $\mathcal{X}$, there is a variable **time** measuring global time, i.e $\mathcal{F}_{c,u}(\tau, v)(\mathbf{time})$ for any mode-configuration (c, u). In the above definition, the syntatic specification of flow functions—e.g. using ODEs—has been left open. In the game $\mathcal{G}$, the controller $\mathbb{C}$ 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 $\mathbb{U}$ will change its uncontrollable mode according to the family of density function 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-functions definition.

Now let $\mathbb{C}$ denote the set of global configurations $C \times U \times \mathcal{R}^X$ of the game $\mathbb{G}$. 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 epression $\sigma(\gamma)$ 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:

$$\pi = \gamma_0 :: \tau_1 :: \gamma_1 :: \tau_2 :: \lambda_2 :: \tau_3 :: \gamma_3 ...$$

where $\gamma_i = (c_i, u_i, v_i), \tau_i \in \mathbb{R}_{\geqslant 0}$ and for all n there exist i st. $\sum_{j \geqslant i} \tau_j = nP$. 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 $\sum_{j \geqslant i+1} \tau_j$ is 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 of the game $\mathcal{G}$ becomes a completly stochastic process $\mathcal{G}|\sigma$, 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

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

$\mathbb{E}^{\mathcal{G},\gamma}_{\sigma,H}(D) \in \mathbb{R}_{\geqslant 0}$ is the expected value of D with respect to random runs of $\mathcal{G}|\sigma$ of length H starting in the configuration γ. We want to obtain a strategy σ^H which minimizes (or maximizes) this expected value.

3.3 Uppaal Stratego

We use the modelling tool Uppaal Stratego for control synthesis. It integrates Uppaal with the two branches Uppaal SMC(statistical model checking for stochastic hybrid systems) and Uppaal Tiga (synthesis for timed games). Therefore, Uppaal is able to synthetize safe and optimal strategy σ_{opt}. We use Dynbex and safe algorithms first we abstract the SHG ignoring all stochasticity. Subsequently, reinforcement learning is used to obtain an optimal sub-strategy σ_{opt} based on $\mathcal{G}|\sigma_{safe}$ and the given random optimization variable. Several learning algorithms are at the modelers disposal in Uppaal Stratego. Recently,

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** shows a summary of the evaluation of various methods on two variants of a one-room example: the purely dynamical model is show in **Fig1** and another one that has an extra counter incremented at each period P. The result is that among that 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** 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.

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 T_g temparatures, and then integrate it to achieve a distance function over complete trajectories. Conveniently, our distance function is modelled using differential equation in Figure 3.2d as a separate process. Before we synthetize anything, we can inspect how does a uniform random choice fare in *fig*: the temparature curve is at the top and heating and window mode trajectories are

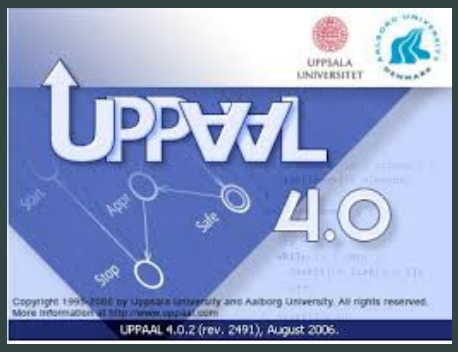


Figure 3.3: Uppaal Stratego, facilitates generation, optimization, comparison as well as consequence and performance exploration of strategies for stochastic priced timed games in a user-friendly manner.

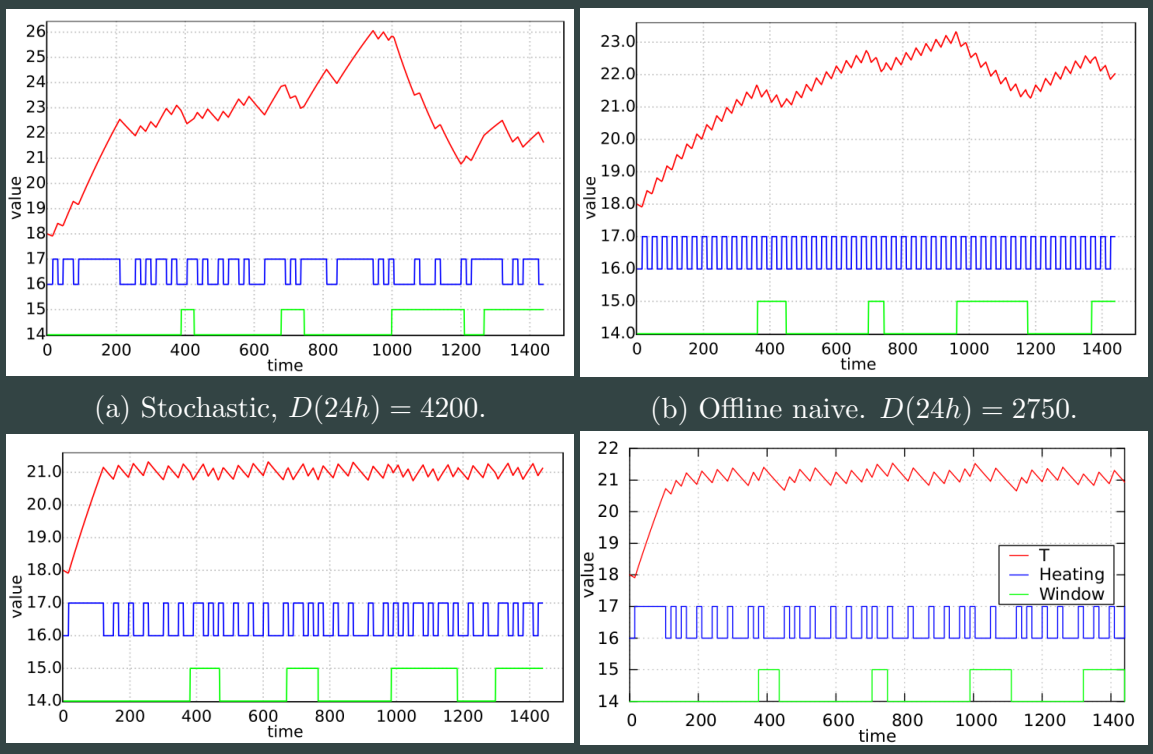
bellow and they jump up when the heating is on the window is open respectively. The result is that the room temperature responds to the mode changes and varies widely, tending to overshoot above the goal, and hence the distance function after $24h$ period is about 4200 on average. In order to synthetize a strategy we pose the following query in Uppaal Stratego:

```
strategy opt = minE (D) [<=24h]: <> 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 synthetized 24h strategy using the ”naive” learning method yields the distance od 2750 on average as shown in *Fig.* The result is even more improved by the ”splitting” learning method in *Fig* where the temperature oscillates around the goal very closely.



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

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

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

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

Figure 3.4: One-room 24h trajectories of various control strategies

3.4 Online Synthesis

Uppaal Stratego [5,6] provides a method for aproximating $\mathbb{E}^{\mathcal{G},\gamma}_{\sigma,H}(D) \in \mathbb{R}_{\geqslant 0}$ by computing a near-optimal strategy σ^H for a given horizon H using 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 $2^1 1 = 2048$ distinct options at each 15min period and thus Uppaal Stratego manages to compute sensible heating configuration only for the first two periods (yieldings $2048^2 = 4194304$ 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 nP not only on the configuration at that point (γ_n) but also on the configuration (γ_{n-1}) at time $(n-1)P$[2], which will be used as the basis for online learning of short-horizon ($h << H$) strategies.

Table 3.1: Performance evaluation of one room controller synthesis: offline-3(-6) methods synthetize strategy for entire 72 hours (144 hours respectively) at once, strategy distance is evaluated on 70 simulations; online-3 methods synthetize a strategy for 5 periods of 15 min ahead and repeat synthesis and execution until 72 hours are covered, the distance is averaged over 70 online simulations.

	Synthesis	**Purely dynamical model**			**Extra period counter**		
	method	**Distance**	**cpu,s**	**mem,kB**	**Distance**	**cpu,s**	**mem,kB**
Offline-3	naive	10227.8	1555.15	11884	3671.84	566.4	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	19668.8	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

Formally:

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

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

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

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 synthetize and evaluate the computed strategy. The following steps are involved:

1. Synthetize a strategy capable of adapting for 5 periods ahead where **LastTime** starts with 0: **strategy S = minE(D) [<= 5*P]: <> == 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 switched together to produce a continuous trajectory of the entire 3 day simulation. An example result of the first $24h$ is displayed in Figure 3.4d which is also comparable to other strategies. The online-3 section Table 3.1 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 seconds and the overall quality of any online synthesis method is very close to the best and expensive offline-3 outcome (the offline "splitting" method).

3.5 Safe patterns computation

In this part is presented some definitions based on an existing procedure of state-space bisection and made available for nonlinear systems with the help of guaranteed integration, the algorithm has been improved to be able to consider longer patterns of modes with a better pruning approach. The method consist of given a objective region R, a set S and a forbidden reigion B, the method find a pattern π_i, $i \in \mathbb{N}$. Le Coënt et al. (2016)

Definition 3*(Safe Control Synthesis).* Let us consider a hybrid system defined in Equation 3.1. Given three sets R, S and B, with $R \cup B \in S$ and $R \cap B = \varnothing$. The system is safe if there is a rule $\sigma : \mathbb{R}^+ \to U$ such that, for any $x_0 \in R$ and any perturbation $p : \mathbb{R}^+ \to U$ the following holds:

- *τ-stability: $x(t)$ return in R infinitely often, at some multiples of sampling time τ.*
$$\forall k \in \mathbb{N}, x(k\tau, t_0, x_0, \sigma, w) \in R \tag{3.3}$$
- *safety: $x(t)$ always stays in S/B.*
$$\forall t \in \mathbb{R}, x(t, t_0, x_0, \sigma, w) \in S,\ x(t, t_0, x_0, \sigma, w) \notin B \tag{3.4}$$

Definition 4(Inclusion function). *Consider a function $f : \mathbb{R}^n \to \mathbb{R}^m$, then* $[f] : \mathbb{R}^n \to \mathbb{R}^m$ is said to be an extension of f to intervals if

$$\forall [x] \in \mathbb{R}, [f]([x]) \subseteq f(x), x \in [x].$$

It is possible to define inclusion functions for all elementary functions such as x,sin,cos,exp, and so on. The *natural* inclusion function is the simplest to obtain: all occurrences of the real variables are replaced by their interval counterpart and all arithmetic operations are evaluated using interval arithmetic. More sophisticated inclusion functions such as the centered form, or the taylor inclusion function may also be used. We now introduce the Initial

One of main features of this approach is the Initial Value Problem, which is introduced as follow:

Definition 5 Initial Value Problem(IVP). Consider an ODE with a given initial condition.

$$\begin{gathered} \dot{x} = f_{\sigma(t)}(x(t), d(t)), \\ with, x(0) \in X_0, d(t) \in [d], \end{gathered}$$

with $f : \mathbb{R}^+ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ assumed to be continuous in t and d and globally Lipschitz in x. we assume that parameters d are bounded (used to represent the disturbance). An IVP consists in finding a function $x(t)$ described by 3.5 for all $d(t)$ lying in $[d]$ and for all the initial conditions in X_0.

Definition 6 Let $X \in \mathbb{R}^n$ be a box of the state space. Let $\pi = (i_1, i_2, .., i_k) \in U^k$. The successor set of X via π, denoted by $Post_\pi(X)$, is the (over-approximation of the) images of X induced by application of the pattern π, i.e, the solution at time $t = k\tau$ of

$$\begin{aligned} \dot{x} &= f_{\sigma(t)}(x(t), d(t)), \\ x(0) &= x_0 \in X, \\ \forall t \geqslant 0, d(t) &\in [d], \\ \forall j \in \{1, .., k\}, \sigma(t) = i_j \in U \text{ for } t &\in [(j-1)\tau, j\tau > \end{aligned}$$

Definition 7 Let $X \in \mathbb{R}^n$ be a box of the state space. Let $\pi = (i_1, i_2, .., i_k) \in U^k$. We denote by $Tube_\pi(X)4$ the union of boxes covering the trajectories of IVP, whose construction is detailed in

Remark 1. $Post_\pi(X)$ is an over-approximation at time $t = k\tau$ of the set of states originated from the set X at time $t = 0$, while $Tube_\pi(X)$ is an over-approximation of the whole set of states between $t = 0$ and $t = k\tau$ originated from the set X at time $t = 0$.

Algorithm 1 Decomposition

```
1: procedure DECOMPOSITION(W, R, S, B, D, K)        ▷ List of set of patterns
2: Input: A box W, a box R, a box S, a box B, a degree D of bisection, a length
   K of input pattern
3: Output: {(V_i, π_i)}_i, True or <, False >
4: (π, b) := FindPattern(W, R, S, B, K)
5:     if b = True then
6:         return < {(W, pat)}, True >
7:     else
8:         if D = 0 then
9:             return <, False >
10:        else
11:            Divide equally W into (W_1, W_2)
12:            for i = 1, 2 do
13:                (, b_i) = Decomposition(W, R, SB, D − 1, K)
14:    return <, False >.
```

Algorithm 2 Find Pattern

1: **procedure** FINDPATTERN(W, R, S, B, K) ▷ List of set of patterns
2: **Input:** A box W, a box R, a box S, a box B, a length K of input pattern
3: **Output:** $< \prod, True >$ *or* $<, False >$
4: $\mathcal{S} = \{\emptyset\}$
5: $\mathcal{L} = \{(W, W, \emptyset)\}$
6: **while** $\mathcal{L} \neq \emptyset$ **do**
7: $e_{current} = takeHead(\mathcal{L})$
8: **for** $i \in U$ **do**:
9: **if** $Post_i(e_{current}.Y_{current}) \subseteq R$ **and** $Tube_i(e_{current}.Y_{current}) \cup B = \emptyset$ **and** $Tube_i(e_{current}.Y_{current}) \cup B \subseteq S$ **then**
10: putTail($\mathcal{S}e_{current}. \prod + i$)
11: **else**
12: **if** $Tube_i(e_{current}.Y_{current}) \cap B \neq \emptyset$ **or** $Tube_i(e_{current}.Y_{current}) \nsubseteq S$ **then**
13: discard $e_{current}$
14: **if** $Tube_i(e_{current}.Y_{current}) \cap B = \emptyset$ **and** $Tube_i(e_{current}.Y_{current}) \subseteq S$ **then**
15: **if** $Length(\prod) + 1 < K$ **then**
16: putTail($\mathcal{L}, (e_{current}.Y_{init}.Post_i(e_{current}.Y_{current}), e_{current}. \prod + i)$)
17: **return** $<, False >$ ▷ if no solution is found, or $< \prod, True >$, $\prod$ beaing any pattern validated in *Solution*.

3.6 Safe and optimal strategies for stochastic hybrid games

For a given SHG, a nondeterministic *strategy* σ. Formally, a strategy is a function $\sigma : \mathbb{C} \to 2^C$,where we denote $\mathbb{C}$ the set of global configurations $\gamma = (c, u, x) \in C \times U \times \mathbb{R}^X$ that returns a nonempty set of allowed modes in a configuration γ. The behavior of the SHG $\mathcal{G}$ under supervision of a strategy σ, denoted as the stochastic process $\mathcal{G}|\sigma$, is defined as follows. A run π of SHG is an interleaved sequene $\pi \in \mathbb{C} \times (\mathbb{R}_{\geqslant 0} \times \mathbb{C})^*$ of configurations and time-delays of some given period P, starting with initial configuration γ_0:

$$\pi = \gamma_0 :: P :: \gamma_1 :: P :: \gamma_2 :: P :: \gamma_3 ...$$

where $\gamma_i = (c_i.u_i, x_i)$, for all n there exist i such that $\sum_{i \geqslant n}(\tau_i) = nP$, and for all i there exists some isntantaneous changes after a time P such as:

1. The flow function is used to update the state values $\gamma_{i-1} = (c_{i-1}, u_{i-1}, x_{i-1}) \xrightarrow{P} (c_{i-1}, u_{i-1}, x_i)$
2. Some perturbations that destabilize the system occur, c_{i-1}, u_i, x_i where $\delta_{\alpha_i}(u_I) > 0$.
3. The controller changes mode each step we extract values from the strategy σ, i.e. $\gamma_i = (c_i, u_i, x_i)$ with $c_i \in \sigma$.

A strategy σ is safe with respect to a set of configurations $S \subset \mathbb{C}$ if for any run π according to σ all configurations encountered are within the safe set S. Note that we require $\gamma_i \in S$ for all i and also $\gamma_i' \in S$ whenever $\gamma_i \xrightarrow{\tau} \gamma_i'$ with $\tau \geqslant P$.

The optimality of a strategy can be evaluated for the stochastic process $\mathcal{G}|\sigma$ with a given optimization variable. Let $H \in \mathbb{R} \geqslant 0$ be a given time-horizon and D a random variable on finite runs, then $\mathbb{E}^{\mathcal{G},\gamma}_{\sigma,H}(D) \in \mathcal{R}_{\geqslant 0}$ is the expected value of D on the space of runs of $\mathcal{G}|\sigma$ of length H starting in configuration γ. For Example , D can be the integrated error(or deviation) of a continuous variable with respect to its desired target value.

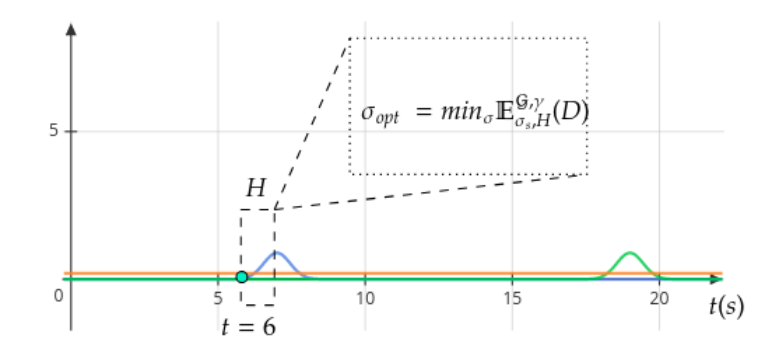


Figure 3.5: MPC with UPPAAL optimization in a horizon H subject to normal and uniform distributions as stochastic perturbations for $t = 6$ in a real time simulation.

The goal is to synthetize a safe and optimal σ_{opt} strategy for a given SHG $\mathcal{G}$, initial configuration γ, safety set S, optimization variable D, and time horizon H. To obtain σ_{opt}, the tool UPPAAL STRATEGO first a maximally permissive safe strategy σ_{safe} is synthetized with respect to S. Subsequently, σ_{opt} is a sub-strategy of σ_{safe} (i.e, $\forall \gamma \in \mathbb{C} : \sigma_{opt}(\gamma) \subseteq \sigma_{safe}(\gamma)$ that optimize (minimizes or maximizes) $\mathbb{E}^{\mathcal{G},\gamma}_{\sigma,H}(D)$. For additive random variables, the optimal sub-strategy of the maximally permissive strategy is deterministic.

3.7 Workflow

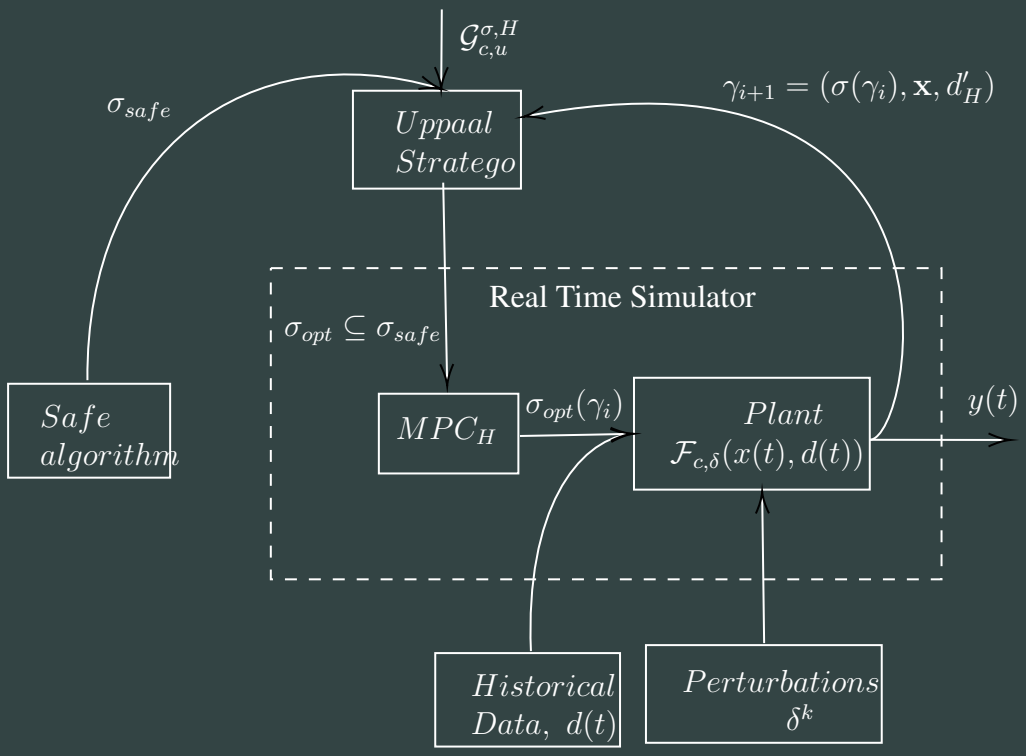


Figure 3.6: A general overview about the procedure to control a SHG

Chapter 4

Safe and Near Optimal Controller Synthesis for Stochastic hybrid Games

In this part, we propose as case study a SWH, modelled as SHG, applying the following procedure: *System modelling* using physics criteria Tsilingiris (1996) and mechanical concepts to have a SHG. Some engineering assumptions either in the model or the configuration parameters are considered to facilate our analysis. A fundamental characteristic of the model is the incorporation of uncontrollable events(perturbations) for instance the opening/closing of the valve. Secondly, our system interact with a physical environment, for this reason we set some paremeters to get the environment conditions so as understand the temporal data such as *Irradiance*, *external and internal temperatures*, these disturbances are preprocessed to interact with our simulator that use MPC to control the real-time system using strategies provided by optimizations methods Section 4.1. To establish a safe sthocastic hybrid game model, we start to compute patterns as Section 3.5 explains. It is important to bear in mind the uncontrollable events in our patterns computation to verify the safety in our system. Lastly, the computation of a near optimal controller consist of an exhaustive exploration of different strategies in order to either minimize or maximize a cost function of a random variable in a specific horizon and a model as a stochastic hybrid game, this computation use UPPAAL STRATEGO. In control experiments we found that this methodology help to concatenate the two approaches implementing a toolbox for a real time simulation in PYTHON Larsen et al. (2016).

4.1 Solar Water Heating Case study

Solar water heating (SWH) has a widely usage and applications in both domestic and industrial sectors. Solar water heating is not only environment friendly but requires minimal maintenance and operation cost compared to other solar energy Shukla et al. (2013). Extensive research has been performed to further improve the thermal efficency of SWH. This research presents a detailed review exclusively on the system modelling as stochastic hybrid game and a new way to improve the performance, these improvements are not provided from a colector design but from the system behaviour that automaticsally derived a mode of operations with this purpose.

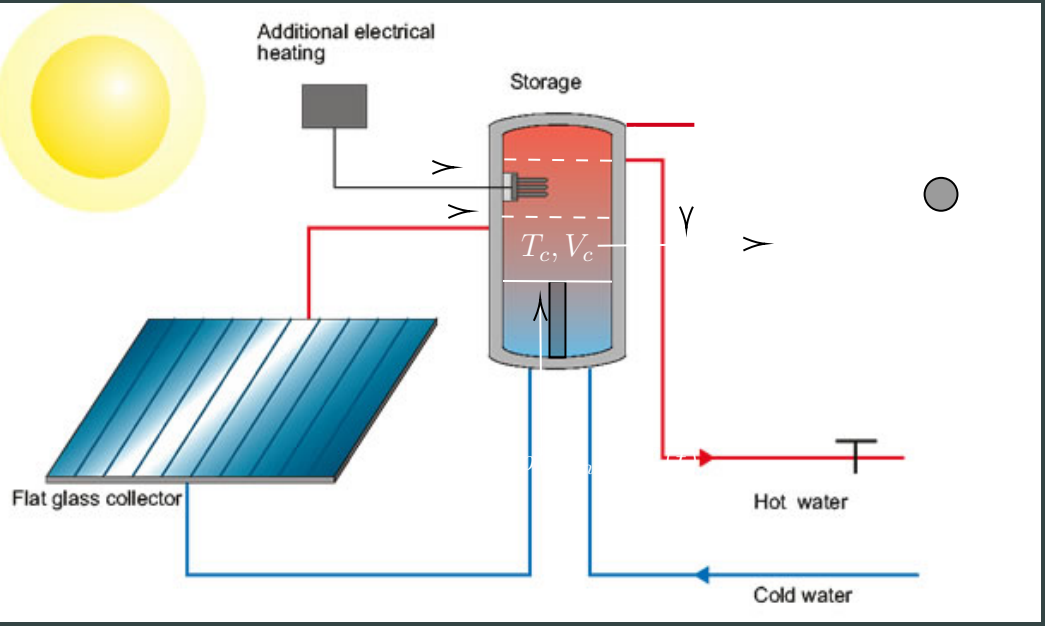


Figure 4.1: Solar Water Heating diagram used to obtain the differential equations

The system setup is about a hybrid solar water heating which involves Auxiliar energy to heat the system and also a collector that transform solar irradiance to heat energy, transfering it to the container. The system also contains a piston which change the volume, in order to simplify the analysis we devide it in three levels as shown in Figure 4.1. We also consider some asumptions in the temperature because of the gradient change along the volumen, to facilate the physical modelling we decide to work with the mean temperature as one variable in the state.

4.1.1 Data processing and generation

Stochastic events as historical data. In practice we notice unpredictable discrete events which can perturbate our dynamical system, these stochastic discrete perturbations we call uncontrallable actions and is managed using a stochastic model approach and machine learning to optimize the control. To generate these perturbations with a realistic criteria for our case study we choose three gaussian distributions with $\mu_1 = 7h$, $\mu_2 = 12h$ and $\mu_3 = 19h$ and $\sigma_1 = 12, \sigma_2 = 15, \sigma_3 = 10$ respectively. Finally we add a uniform distribution around the day.

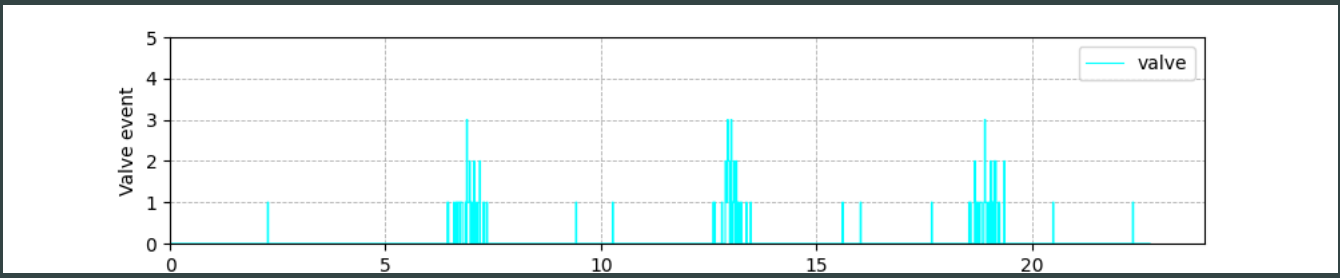


Figure 4.2: Valves perturbations generated along a day.

Solaris Data and preprocessing The data collected is from spain solaris, which consist of a set of environment features such as environment temperatures and irradiance that would be useful for the simulation. We preprocess the time series data modifying the time step from 15 minutes to 1 minute through interpolation. The data is collected from Almeria in Spain solargis (2010).

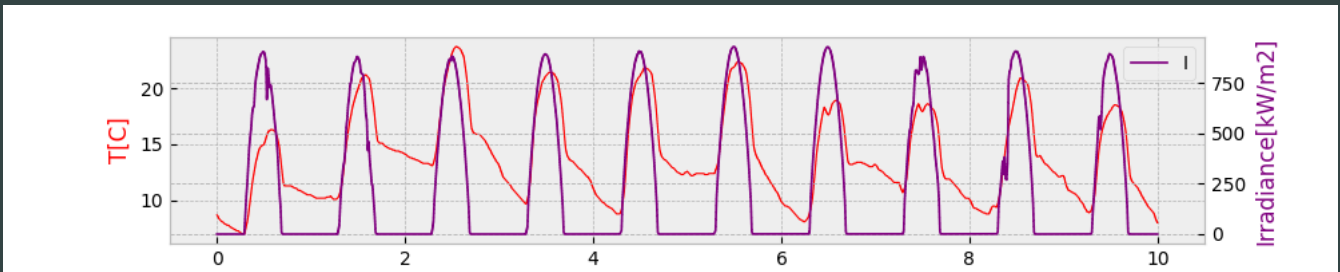


Figure 4.3: Weather time series, irradiance and temperature

Prediction Data. To integrate a MPC in our simulator we configurate a forecaster in real time in order to get a sequence of environment data on future which will be used to control future states an get the correct patterns on present. ARIMA state model is used with a realtime pivot = $24h$ prediction size = $3h$ as shown in Figure 4.4, however in the real time simulation we set a prediction size = $1h$ Jain and Mallick (2017).

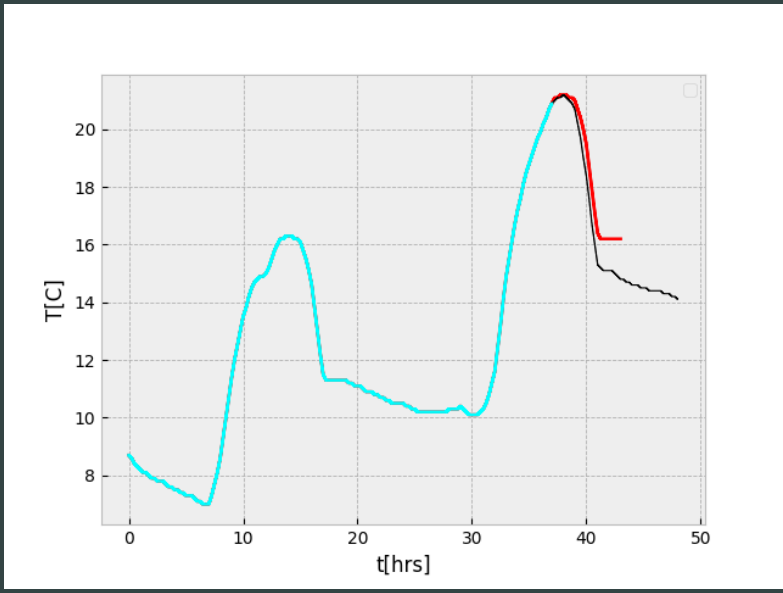


Figure 4.4: An example of prediction using ARIMA forecaster

Disturbances such as environment temperatures and irradiance affect the system either positive or negative depending of the state and desired state. *Model predictive control* is a method to control establishing in a future horizon to explore best actions control for future states.

States			
Variable	Description	Initial Value	Units
E	Energy consumption	0.0	J
V	Container Volume	0.13	m^3
T	Container Temperature	50.0	°C
Constants			
Variable	Description	Value	Units
C_e	The factor heat of the water in	4186	$J\,°C^{-1}\,kg^{-1}$
$\dot{m}$	Mass flow rate input/output	0.1	$kg\,s^{-1}$
M_c	the mass of the container in	100	kg
A_c	Area of colector in	1	m^2
A_t	Area of total of surface in	5.56	m^2
k_c	Conduction coeficient	16	$W\,m^{-1}\,°C^{-1}$
P_{aux}	Auxiliary heat power in	1000	W
Input values			
Variable	Description	Space values	
v	Controllable action to release water	$\{0, 1\}$	
r	Controllable action to heat	$\{0, 1\}$	
p	Uncontrollable action to change capacity water	$\{1, 2, 3\}$	
Disturbances			
Variable	Description	Range	Units
$T_{in}(t)$	the temperature of water input/output in	$[20 - 25]$	°C
$I_{env}(t)$	the irradiance in	$[0 - 920]$	$W\,m^{-2}$
$T_{env}(t)$	the outside temperature in	$[0 - 16.5]$	°C

Table 4.1: Parameter values used for computing c_1, c_2, c_3 and c_4.

4.1.2 Hybrid Solar Water Heating as a Sthocastic Hybrid Game

. The hybrid solar water heating scenario with 12 modes of operations is defined like this: $\mathcal{G}_{n,m} = (\mathcal{C}, \mathcal{U}, \mathcal{X}, \mathcal{F}, \delta)$, where the controller $\mathcal{C}$ has a finite set of controllable actions, given by resistance state $r \in \mathbb{B}$ and piston position $p \in \{1, 2, 3\}$. The environment $\mathcal{U}$ has a finite set of stochastic discrete perturbations $v \in \mathbb{B}$, that means the valve state for opening/closing water aperture. We assume that $\mathcal{U}$ given δ can switch among modes acording to the probabilistic distributions defined in Figure 4.2 at every period. The state variables in $\mathcal{X}$ are given by $\{E, V, T\}$, container temperature, energy used and container volumen respectively.

Given the container temperature and the volume, a controllable actions $r \in \mathbb{B}$ and $p \in \{1, 2, 3\}$, a stochastic discrete perturbation $v \in \mathbb{B}$ and a time delay τ Figure 4.1.

$$\begin{aligned}\frac{d}{dt}T(t) = \frac{1}{V(t)}[&-c_1(T(t) - T_{env}(t)) - vc_2(T(t) - T_{in}(t)) \\ &-fc_3(pV_{min} - V(t))(T(t) - T_{in}(t)) + c_4 I_{env}(t) + rc_5]\end{aligned} \tag{4.1}$$

$$\frac{d}{dt}V(t) = pV_{min} - V(t); \tag{4.2}$$

$$\frac{d}{dt}E_{used} = rc_3; \tag{4.3}$$

In 4.1, 4.2, 4.3 we note some constants $\{c_1, c_2, c_3, c_4\}$, these are computed with the parameteres defined in table 4.1 resulting $\{2.44e^{-5}, 4.77e^{-6}, 0.0024, 0.01\}$ respectively. Another important variable to consider is $f \in \{0, 1\}$, this variable affect the term $fc_3(pV_{min} - V(t))(T(t) - T_{in}(t))$ that measure the energy removed in the system due to the piston expansion when $p \xrightarrow{\tau} p', (p' > p)$ thus $f = 1$ which imply the incoming of cold water at Equation 4.1. However, if the piston is contracted, that means $p \xrightarrow{\tau} p', p \geqslant p'$ and hence $f = 0$, in this case there is no temperature changes but rather a loss of energy due to the outgoing water.

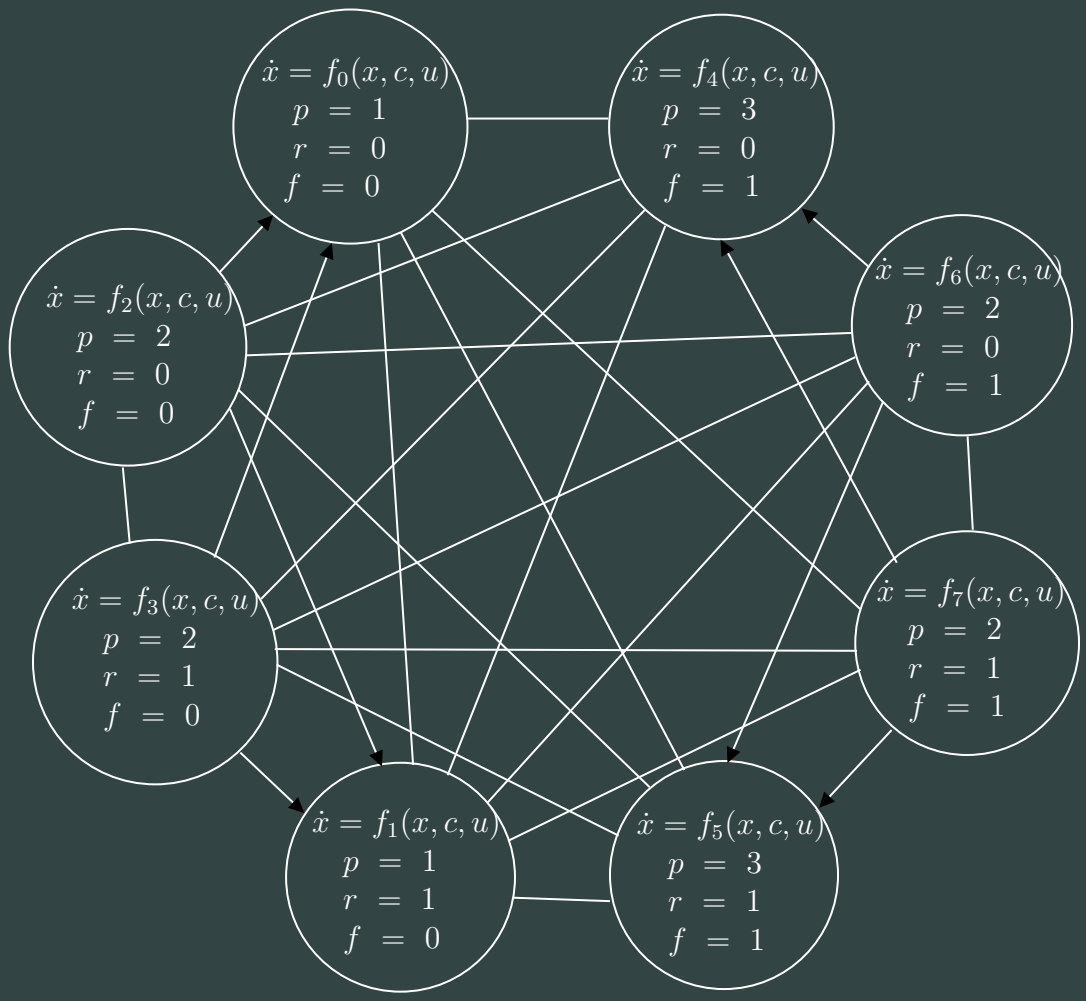


Figure 4.5: Automaton with forbidden transitions because of the addition of a discrete variable f which facilate the system modelling either on piston expansion or compression, the perturbation of the valve is not considered in this representation as discrete state but rather as interval box.

As we notice, there are eight operation modes, is because in this step we do not care the stochasticity either perturbations or disturbances will be considered as intervals assuming the best and the worst case. There are forbidden transitions in the automaton, for this reason we modify the algorithm which not filter forbidden transitions, otherwise the algorihtm ignore the direction exploring modes which can cause problems in the pattern search.

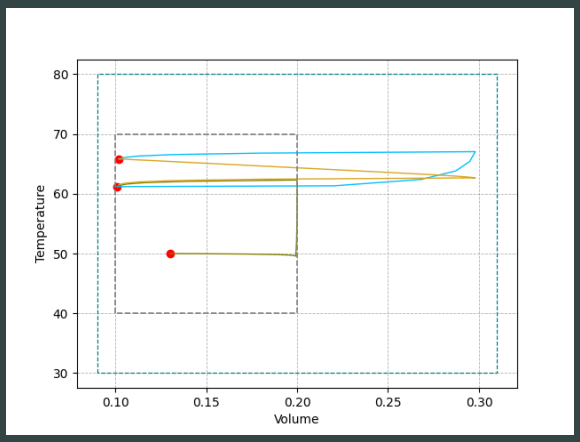
1: **procedure** FINDPATTERN(W, R, S, B, K) ▷ List of set of patterns
2: **Input:** A box W, a box R, a box S, a box B, a length K of input pattern
3: **Output:** $< \prod, True >$ *or* $<, False >$
4: $\mathcal{S} = \{\emptyset\}$
5: $\mathcal{L} = \{(W, W, \emptyset)\}$
6: **while** $\mathcal{L} \neq \emptyset$ **do**
7: $e_{current} = takeHead(\mathcal{L})$
8: **for** $i \in U$ **do**:
9: **if** $Pair(l, i) = False$ **then** ▷ l is the current mode
10: **continue** ▷ Not considered transition l, i
11: **if** $Post_i(e_{current}.Y_{current}) \subseteq R$ **and** $Tube_i(e_{current}.Y_{current}) \cup B = \emptyset$ **and** $Tube_i(e_{current}.Y_{current}) \cup B \subseteq S$ **then**
12: putTail($\mathcal{S}e_{current}.\prod + i$)
13: **else**
14: **if** $Tube_i(e_{current}.Y_{current}) \cap B \neq \emptyset$ **or** $Tube_i(e_{current}.Y_{current}) \nsubseteq S$ **then**
15: discard $e_{current}$
16: **if** $Tube_i(e_{current}.Y_{current}) \cap B = \emptyset$ **and** $Tube_i(e_{current}.Y_{current}) \subseteq S$ **then**
17: **if** $Length(\prod) + 1 < K$ **then**
18: putTail($\mathcal{L}, (e_{current}.Y_{init}.Post_i(e_{current}.Y_{current}), e_{current}.\prod + i)$)
19: **return** $<, False >$ ▷ if no solution is found, or $< \prod, True >$, $\prod$ beaing any pattern validated in *Solution*.

Figure 4.6: This algorithm is modified to ignore forbidden transitions and is improved filtering invalid transition in the line 9.

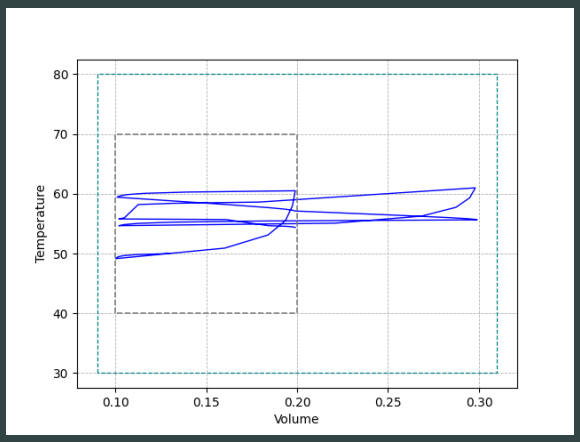
4.2 Experiments:

4.2.1 Patterns computation

To establish a safe stochastic hybrid game, we start to compute patterns as the Section 3.5 explains. It is important to bear in mind that the stochastic discrete perturbation v is considered as an interval $[0, 1]$, thus in the second term in the Equation 4.1 results $Interval(0, 1)c_2(x[0] - Ti)/(0.1p)$, which means the term becomes zero when the valve is close. However, when the valve is open the term will change to its biggest value due to the cold water incoming and warm water outgoing. Safe patterns are computed and associated to zonotopes,which are regions where patterns verify the next: Given $D = 20, K = 3$ and by constructing a law σ, such that for all $x_o \in R$, and under the unknown bounded perturbation d, there exists a $\sigma : \mathbb{C} \to C^k$ for some k such that: Such a law permits to perform a safe control guaranteeing limits for each action in each step conditioning a safe sequence of modes called *pattern* which verify stability and safety defined in Section 3.5.



(a) Simulation for the first three patterns of lengths $3, 2, 3$ respectively.



(b) Stabilization after the normal perturbation $u = 7h, \sigma = 1.6$ between $6h$ and $8h$.

We define the boundaries for the SWH such as $R = \begin{bmatrix} 40 & 70 \\ 0.1 & 0.2 \end{bmatrix}$, $S = \begin{bmatrix} 30 & 80 \\ 0.09 & 0.31 \end{bmatrix}$, $W = \begin{bmatrix} 40 & 70 \\ 0.1 & 0.2 \end{bmatrix}$, $B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$ that are required for the decomposition algorithm developed in Liberzon (2003). We can see the states returning to the R region after finish the pattern satisfying stability and safety in the simulation Figure 4.7a. However, the algorithm is able to stabilize even when a perturbation occur as Figure 4.7b.

4.2.2 Real Time Model Predictive Controller

PYTHON SIMULATOR. We implement a *toolbox* which incorporate sthocastic hybrid models $\mathcal{G}_{n,m}$ and controllers $\mathcal{C}$. The models works with seasonal data *disturbances* for this reason we incorporate modules to preprocess the data. In the initial step of the model predictive algorithm to control MPC. The application of MPC to stochastic hybrid games subject to disturbances and stochastic perturbations gives rise to uncertain optimization problems. We use the first pattern corresponding to the current zonotope. In the subsequent iterations we use the STRATEGO to find a near optimal strategy from a selected set of safe patterns as show 4.2.3.2. It is very important to bear in mind the way of model predictive controler operate which is an asynchrnous call before finish the last mode in the current pattern, sending to STRATEGO the current dynamical states such as *mode, volume, temperature* and so on with the corresponding environment predictions 4.4. After the near optimal pattern computation for stochastic hybrid game by STRATEGO the process repeat.

4.2.3 Controllers Optimizations:

Regarding our experiments, we implement two controller synthesis optimization over a real time simulator which run a MPC and synthetize a switching rule σ written in PYTHON and a number of controllers associated to two optimization, including a greedy method and the ones produced by UPPAAL STRATEGO. The simulator use our case study a SWH as plant and a SHG $\mathcal{G}_{n,m}$ as model in the second optimization. We introduce the optimizations which we use in our experiments. We present the current controller operating in the house, one controller proposed by engineers and another one controller synthetized using online synthesis and UPPAAL-STRATEGO as follows.

4.2.3.1 *Greedy optimization*

The greddy controller explore a set of patterns that correspond to the current zonotope and is chosen the pattern that minimize the cost function. Due to the use of a complete pattern, this approach is able to guaranteeing a safe behaviour. Additionally, this approach only optimize partially due to the stochasticity is ignored thus it can be optimized even more.

$$J_{\sigma_{safe},H} = \|x(\sigma_{safe}, t, d, u) - x_d(\sigma_{safe}, t, d, u)\|$$

$$\sigma_{opt} = \underset{\sigma \subseteq \sigma_{safe}}{\operatorname{argmin}}(J_{\sigma,H})$$

4.2.3.2 *Reinforcement learning optimization*

(Stratego-ON) The controller is synthetized by Uppaal Stratego using the online strategy synthesis methodology introduced in Section 3.2 with a short horizon H ahead and $P = \tau$. The aim is to learn the optimal control actions configurations for several steps ahead using machine learning methods dealing with stochastic discrete perturbations through a set of patterns which guaranteeing a safe behaviour and hence synthetize controllers that avoid overflow/underflow.

For our experiments, in the simulator we fix a time horizon H of 75 minutes with a period P of 5 minutes. As in the real house, every 5 minutes, the simulator outputs the current container values such as temperature T, volume V, energy E, mode and predicted environment data in order to Uppaal can use these to calculate the optimal strategy. Subsequently, each operation mode that correspond to the strategy is used mapping it as control actions such as resistor r, piston p which are used to control the states during the real time MPC simulation for the next 5 minutes. The house has a desired temperature T^g and an alpha parameter which denotes the importance to minimize either the distance $T - T_g$ or energy consumption E. Our goal is to optimize the comfort for the water consumption reducing the energy consumption. To define this cost function that is subject to controller (strategy) σ and $\mathcal{G}_{n,m}$ of the form $\pi = \gamma_0 \xrightarrow{t_1} \gamma_1 \xrightarrow{t_2} ... \xrightarrow{t_{k-1}} \gamma_{k-1} \xrightarrow{t_k} \gamma_k$ where $k = H/P$ is the number of control steps in the run π. Let $T(\gamma_j)$ denote the container temperature T at configuration γ_j. Then the cost function is defined by

$$\sigma^H_{opt} = \underset{\sigma \subseteq \sigma_{safe}}{\operatorname{argmin}}(\mathbb{E}^{\mathcal{G},\gamma}_{\sigma_{safe},H}(\alpha(E) + (1-\alpha)(T - T_g))) \tag{4.4}$$

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

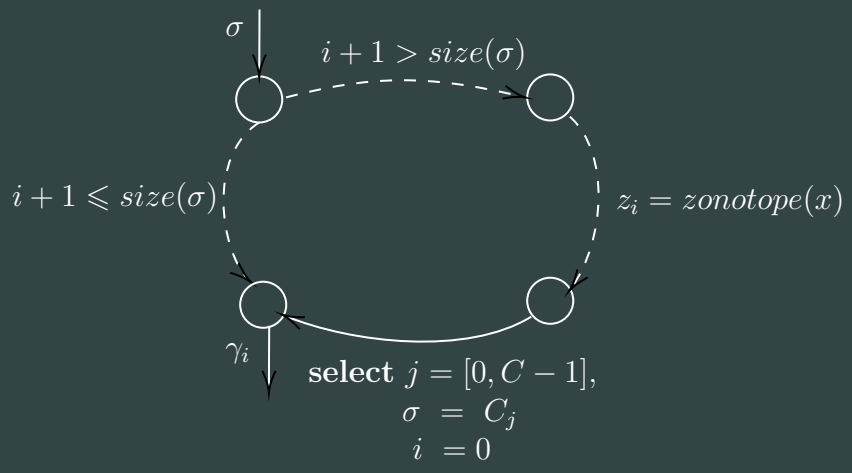


Figure 4.8: The model of the controller. The **select** statement is a simplification of having a set of edges with one for each value in the select range.

The first part in the algorithm consist on mapping the current sate to the corresponding pairs such as zonotope and set of patterns, the *select* statement represent a set of edges each of these with its corresponding pattern; the output is the new strategy when a whole modes in the current pattern were visited otherwise it just move to next mode, each mode is associated to a configuration γ_i.

Subsequently, we define distributions δ_i^k which estimates events in the system as stochastic discrete perturbations, similar to the defined in the real time MPC the distributions are composed by 3 normal distributions and 1 uniform distribution as described in Figure 4.2. It is important to bear in mind, the event accumulation which means the number of valves open depending of the probability, after to define the perturbations in the $x_{clock} < P$ we reset the $x_{clock} = 0$. Then, we implement an *Euler method* to evolve the system each period P, this method update the state x using the flow function $\mathcal{F}_{c,u}(x(t), d(t))$ and defined in Equation 4.1,Equation 4.2, Equation 4.3 respectively, this updating is subject to a cost function defined in Equation 4.4. Finally, We defined another y_{clock} to limit the processing until it reach the horizon H.

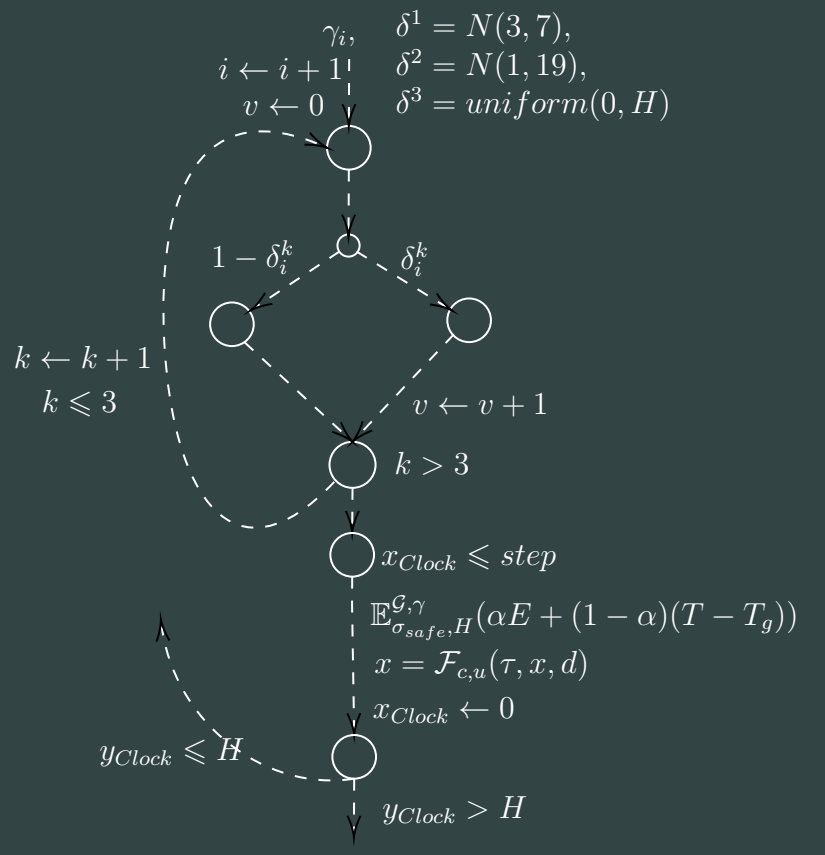
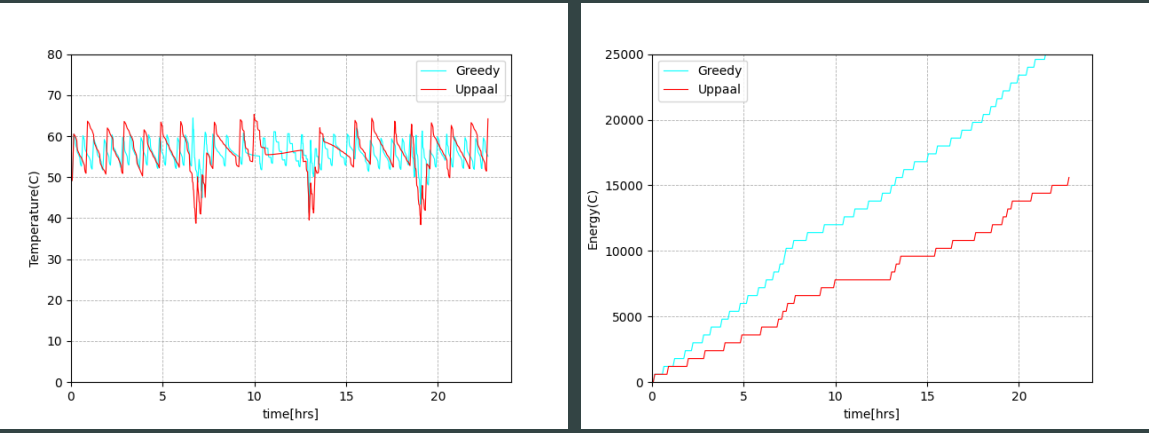


Figure 4.9: The model generating stochastic perturbations as valve aperturing and closing

After to define the automaton as SHG UPPAAL STRATEGO we use the command 1 to find the strategy which minimize the cost function in a horizon H. Then, simulate the system for a horizon in order to get modes. These modes are used in the real time simulator which use MPC to optimize strategies subject to safe patterns for $24h$ as shown in Figure 4.10a.

```
1. strategy Opt = minE (pareto) [<=horizon]:
<> GTime>=horizon
2. simulate 1 [<=horizon] {mode} under Opt
```

We notice a better performace with online control using STRATEGO because of the stochastic aproach which explore the best set of modes in a *horizon* H STRATEGO not only verify the system correcteness but also explore strategies which optimize the cost function, which include the energy consumption and temperature distance.



(a) Temparatures comparison between safe greedy and safe stratego controller.

(b) Energy comparison between safe greedy and safe online stratego controller.

Figure 4.10: Solar water heating 24h trajectories of two control strategies

For $\alpha = 0.5$, the temperature mean error in the greedy method Figure 4.10a is : 1.8 and the reinforcement learning optimization is: 3.37 and also the saved energy consumption using uppaal with respect to a greedy method is $\triangle E = 310.8Kw$ as Figure 4.10b shows.

Chapter 5

Discussion and Conclusions

5.1 Conclusions

We showed that the *Hybrid Solar Water Heating* case study can be modeled as a stochastic hybrid game and apply formal controller synthesis to automatically derive controllers for our SWH, such that symbolic and reinforced learning techniques from UPPAAL STRATEGO are used. This evaluation is subject to certain operation modes called patterns wich ensure a safe coontrol.

The use of zonotopes allows an efficient implementation of the stochastic controller synthesis with STRATEGO. The experimental evaluation showed that this method can compute a safe control strategy in a short horizon 75 minutes and also the cost function was significantly optimized comparable in performance to the *greedy* controller. The framework developed is adapted to an online model-predictive control setting and helps engineers not only guarantee a safe controller but also apply formal controller synthesis to automatically derive controllers through reinforcement learning given by UPPAAL STRATEGO.

5.2 Limitations and Future Work

Future research directions; Although it is true that we use real environment data, temperature of input water is not provided. However, considering not variations we set a constant temperature, which in fact change in time, would be interesting

to test the simulator with real input water temperatures. Additionally, the distributions of perturbations was modelled using 1 uniform distribution and 2 normal distribution, around $7h$ and $19h$. However, a more realistic distributions can be obtained through historical data provided by the interaction between members of family in a house and our algorithm.

Bibliography

Branicky, M. S. (1994). Stability of switched and hybrid systems. In *Proceedings of 1994 33rd IEEE Conference on Decision and Control*, volume 4, pages 3498–3503. IEEE.

Fribourg, L., Kühne, U., et al. (2014). Finite controlled invariants for sampled switched systems. *Formal Methods in System Design*, 45(3):303–329.

Girard, A. (2005). Reachability of uncertain linear systems using zonotopes. In *International Workshop on Hybrid Systems: Computation and Control*, pages 291–305. Springer.

Jain, G. and Mallick, B. (2017). A study of time series models arima and ets. *Available at SSRN 2898968*.

Larsen, K. G., Mikučionis, M., et al. (2016). Online and compositional learning of controllers with application to floor heating. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 244–259. Springer.

Le Coënt, A., dit Sandretto, J. A., et al. (2017). An improved algorithm for the control synthesis of nonlinear sampled switched systems. *Formal Methods in System Design*, pages 1–21.

Le Coënt, A., Fribourg, L., et al. (2016). Distributed synthesis of state-dependent switching control. In *International Workshop on Reachability Problems*, pages 119–133. Springer.

Liberzon, D. (2003). *Switching in systems and control*. Springer Science & Business Media.

Sanfelice, R. G. (2020). *Hybrid Feedback Control*. Princeton University Press.

Shukla, R., Sumathy, K., et al. (2013). Recent advances in the solar water heating systems: A review. *Renewable and Sustainable Energy Reviews*, 19:173–190.

solargis (2010). MS Windows NT kernel description. https://solargis.com/es/products/time-series-and-tmy-data/useful-resources. accessed: 2010-09-30.

Tsilingiris, P. (1996). Solar water-heating design—a new simplified dynamic approach. *Solar Energy*, 57(1):19–28.