Optimal Control for Simple Linear Hybrid Systems

Mahmoud A. A. Mousa, Sven Schewe, and Dominik Wojtczak
University of Liverpool, Liverpool, UK
{mahmoud.mousa, sven.schewe, dominik.wojtczak}@liverpool.ac.uk

***Abstract*—This paper studies optimal time-bounded control in a simple subclass of linear hybrid systems, which consists of one continuous variable and global constraints. Each state has a continuous cost attached to it, which is linear in the sojourn time, while a discrete cost is attached to each transition taken. We show the corresponding decision problem to be NP-complete and develop an FPTAS for finding an approximate solution. We have implemented a small prototype to compare the performance of these approximate and precise algorithms for this problem. Our results indicate that the proposed approximation schemes scale. Furthermore, we show that the same problem with infinite time horizon is in LOGSPACE.**

I. INTRODUCTION

We study a simple subclass of linear hybrid systems [1] extended with weights, which consist of one variable and global constraints for the values that the variable is allowed to take during a run of the system. The aim is to minimise the total cost over a finite-time horizon or a long-time average cost over an infinite time horizon. The cost is composed of discrete and continuous parts. In our model, each state and transition is assigned a fixed cost. Every time a transition is taken (i.e. the current state changes), a cost assigned to this transition is incurred. The continuous cost is the sum of the products of the sojourn time in each state and the cost assigned to this state. We exemplify this by applying this model to the optimal control of heating, ventilation, and air-conditioning (HVAC) systems. HVAC systems account for about 50% of the total energy cost in buildings [2], so a lot of energy can be saved by optimising their control. Many simulation programs have been developed to analyse the influence of control on the performance of HVAC system components such as TRNSYS [3], EnergyPlus [4], and the Matlab's IBPT [5]. Our approach has the advantage over the existing control theory techniques that it provides guarantees. We also study the computational complexity of the underlying decision problems. We show that the existence of a control with a cost lower than a given constant is NP-complete. At the same time we develop a simple constant factor approximation algorithm, which is then used as the basis for a fully-polynomial-time approximation scheme (FPTAS) for finding a solution with a relative performance ρ for any $\rho > 0$. Moreover, we have implemented a small prototype to compare the performance of these approximate and precise algorithms for this problem. Our results indicate that the proposed approximation schemes scale. Furthermore, we show the limit-average cost optimisation problem over infinite time horizon to be in LOGSPACE.

Related work. Our model can be seen as a weighted extension of the linear hybrid automata model ([6], [7]), but with global constraints only. Even basic questions for the general linear hybrid automata model are undecidable already for three variables and not known to be decidable for two variables [8]. Most of the research for this model has focused on qualitative objectives such as reachability. Various subclasses of hybrid systems with a decidable reachability problem were considered, see e.g. [8] for an overview. In particular, reachability in linear hybrid systems, where the derivative of each variable in each state is constant, can be shown to be decidable for one continuous variable by using the techniques from [9]. In [10], it has been shown that reachability is decidable for timed automata, which are a particular subclass of hybrid automata for which all the variables have slope equal to 1.

In this paper, we study the quantitive objective of cost optimisation for a linear hybrid automaton model with one variable, where each state and each transition is assigned a weight. [11] studies a similar model with multiple variables, but with no switching costs and only for the infinite time horizon. [12] studies a hybrid automaton model, where the dynamics are governed by linear differential equations, but again without switching costs and only with an infinite time horizon. Both of these papers show that, for any number of variables, a schedule with the optimal long-time average cost can be computed in polynomial time. In [13], [14], the same models without switching costs have been studied over the infinite time horizon, with the objective of minimising the peak cost, rather than the long-time average cost. In [15], long-time average and total cost games have been shown to be decidable for hybrid automata with strong resets, in which all variables are reset to 0 after each discrete transition. The long-time average and total cost optimisation for the weighted timed automata model have been shown to be PSPACE-complete (see e.g. [16] for an overview).

There are many practical approaches to the reduction of energy consumption and peak demand in buildings. One particularly popular one is model predictive control (MPC) [17]. In [18], stochastic MPC was used to minimise the energy consumption in a building, while [19] studies the reduction of the peak electricity demand. In [20], On-Off optimal control was considered for air conditioning and refrigeration. The drawback of using MPC is its high computational complexity and the fact that it cannot provide any worst-case guarantees. UPPAAL Stratego [21] supports the analysis of the expected cost in linear hybrid systems, but uses a stochastic semantics of these models [22], [23]. I.e. a control strategy induces

a stochastic model where the time delay in each state is uniformly or exponentially distributed. This is different to the standard nondeterministic interpretation of the model, which we use in this paper.

Structure of the paper. The paper is organised as follows. We introduce all necessary notation and formally define the model in Section II. In Section III we prove the existence of optimal schedules of a particular structure and that the infinite time horizon decision problem is in LOGSPACE. In Section IV we show that the cost optimisation decision problem for finite time horizons is NP-complete. In Section V we introduce a constant factor approximation algorithm for our problem. In Section VI we extend this algorithm to an FPTAS by a reduction to the 0-1 knapsack problem. In Section VII we compare the performance of our algorithms on randomly generated instances. Finally, we conclude in Section VIII.

II. PRELIMINARIES

A. Motivation

Our simple linear hybrid automata are motivated by the following application. Suppose we would like to keep the temperature in a room in a comfort zone, e.g. between V_{min} = 18°C and V_{max} = 22°C. We have multiple heaters at our disposal, each with a different running cost per time unit and initial set up cost. The set up cost is paid every time the heater is switched on, which models wear and tear of its multiple elements. The aim is to find a schedule with the minimum total average cost per time unit that keeps the temperature within the comfort zone at all times for a given finite or infinite time horizon. Note that, in the case of a finite time horizon, the problem is the same as minimising the total cost incurred during that finite time. As we will show later this problem is NP-complete for the finite time horizon and so we will focus later on finding an approximately optimal schedules instead.

Let $V(t)$ be the function describing the temperature in the room at time t and $V(0) = V_0$ be the initial temperature satisfying $V_{min} \leq V_0 \leq V_{max}$. The equation below, taken from [13], [14], describes the change of temperature in a room with one heater:

$$C\frac{dT}{dt} + \lambda V = Q$$

where C is the thermal capacity of the room (kJ/K), λ is the thermal conductance between the room and the ambient air (kW/K), and Q is the heat input rate of the heater (kW). If the heater is switched off then $Q = 0$. Solving this first order differential equation gives us the following formula for $V(t)$.

$$V(t) = \frac{Q}{\lambda} + \left(V_0 - \frac{Q}{\lambda}\right) e^{-\frac{\lambda}{C}t}$$

We can write down this equation as:

$$V(t) = K_1 e^{-at} + K_2$$

where $K_1 = V_0 - \frac{Q}{\lambda}$, $K_2 = \frac{Q}{\lambda}$, and $a = -\frac{\lambda}{C}$. Under the natural assumptions that the heater output is much higher than the heat loss and the comfort zone is quite narrow, this exponential behaviour can be approximated well by a linear behaviour. This is because the slope of $V(t)$ at $t = 0$ is aK_1 and the most extreme value of the slope of $V(t)$ before the boundary of the comfort zone is reached is $aK_1(1+(V_{max}-V_{min})/K_1)$.

B. Formal definition of Simple Linear Hybrid Automata

Motivated by our application, we define a simple linear hybrid automaton $\mathcal{A}$ as a tuple $(M, A, \pi_c, \pi_d, [V_{min}, V_{max}], V_0, t_{max})$ where:

- $M = \{0, 1, \ldots, K\}$ is the set of modes, where 0 is a special *idle* mode; we use M^+ to refer to the set of non-idle modes, i.e. $\{1, \ldots, K\}$.
- $A : M \to \mathbb{Q}$ is the slope of the variable in a given mode, where $A(m) > 0$ for all $m \in M^+$ and $A(0) < 0$;
- $\pi_c : M \to \mathbb{Q}_{\geq 0}$ is the cost per time unit in a given mode, where $\pi_c(0) = 0$;
- $\pi_d : M \to \mathbb{Q}_{\geq 0}$ is the cost of switching to a given mode, where $\pi_d(0) = 0$;
- V_{min} and V_{max} are the minimum and maximum allowed value of the variable, respectively;
- $V_0 \in [V_{min}, V_{max}]$ is the initial value of the variable;
- $t_{max} \in \mathbb{Q} \cup \{\infty\}$ is the end of the time horizon.

The definition of our model is motivated by the special properties idle systems have. (In our motivating example, a heating system is idle when all heaters are switched off.) Switching to the idle state is free, while changing into a mode, where the dynamics of the system are actively influenced—in our example, by switching heaters on—incurs costs. These costs come in two flavours: initial costs for starting such a mode, and continuous costs for staying in it. Investing through these 'active' states into influencing the dynamics of the system leads to the continuous variable—the temperature in our motivating example—to develop into the opposite direction compared to the idle system.

As we will show in Observation 1 and Observation 2, the decision problems for simple linear hybrid automata that we study in this paper can easily be reduced to the same ones for structurally equivalent simple linear hybrid automata with $V_{min} = V_0 = 0$.

Running example. Suppose we need to keep the temperature inside an office between 18°C and 22°C for $t_{max} = 7$ hours, and the initial temperature inside of it is 18°C. (As we will see later, we can reduce this problem to keeping the temperature inside between $V_{min} = V_0 = 0$°C and $V_{max} = 4$°C.) We have two heaters, i.e. $K = 2$, at our disposal: gas (mode 1) and electric (mode 2). Their parameters are $A(1) = 4/3\,[°\text{C}/h]$, $A(2) = 2\,[°\text{C}/h]$, and $A(0) = -4\,[°\text{C}/h]$, i.e. it takes 3 hours for the office to reach the maximum allowable temperature of 22°C when using the gas heater, but just 2 hours using the electric one. It takes 1 hour for the office to cool from 22°C to 18°C, when both of the heaters are off (mode 0). The running costs of the heaters are $\pi_c(1) = 10\,[£/h]$ and $\pi_c(2) = 20\,[£/h]$, and the initial costs of switching each heater are $\pi_d(1) = 30\,[£]$ and $\pi_d(2) = 10\,[£]$. That is, the gas heater is cheaper to run, but more expensive to turn on, e.g. due to a need for regular inspections. ◁

C. Schedules, leaps, and their cost

A *timed action* is a pair $(m, t) \in M \times \mathbb{R}_+$ of a mode m and time delay $t > 0$. A *finite schedule* σ is a finite sequence of timed actions $\sigma = \langle (m_1, t_1), (m_2, t_2), \ldots, (m_k, t_k) \rangle$, such that $\sum_{i=1}^{k} t_i = t_{max}$. An *infinite schedule* σ is an infinite sequence of timed actions $\sigma = \langle (m_1, t_1), (m_2, t_2), \ldots, (m_k, t_k), \ldots \rangle$, such that $\sum_{i=1}^{\infty} t_i = \infty$.

The *run* of a finite schedule σ is a sequence of states and timed actions $run(\sigma) = \langle V_0, (m_1, t_1), V_1, ..., (m_k, t_k), V_k \rangle$ such that for all $0 \leq i \leq k-1$, we have that $V_{i+1} = V_i + t_i A(m_i)$. A schedule and its run are called *safe* if $V_{min} \leq V_i \leq V_{max}$ holds for all $1 \leq i \leq k$. The run of an infinite schedule and its safety is defined accordingly.

The *cost* of a finite schedule $\sigma = \langle (m_1, t_1), (m_2, t_2), \ldots, (m_k, t_k) \rangle$ is defined as $\pi(\sigma) = \sum_{i=1}^{k} \pi_d(m_i) + \pi_c(m_i) t_i$. The *limit-average cost* of an infinite schedule $\sigma = \langle (m_1, t_1), (m_2, t_2), \ldots \rangle$ is defined as

$$\pi_{avg}(\sigma) = \limsup_{k \to \infty} \left(\sum_{i=1}^{k} \pi_d(m_i) + \pi_c(m_i) t_i \right) / \sum_{i=1}^{k} t_i$$

A safe finite schedule σ is *ϵ-optimal* if, for all safe finite schedules σ', we have that $\pi(\sigma') \geq \pi(\sigma) - \epsilon$. A safe finite schedule is *optimal* if it is 0-optimal. A safe infinite schedule σ is *optimal* if, for all safe infinite schedules σ', we have that $\pi_{avg}(\sigma') \geq \pi_{avg}(\sigma)$.

Running example continues. For instance $\sigma_1 = \langle (1,3), (0,1), (2,2), (0,1) \rangle$ and $\sigma_2 = \langle (1,1), (2,1), (0,\frac{1}{2}), (1,1), (0,\frac{1}{2}), (1,1), (2,1), (0,1) \rangle$ are both safe finite runs that last for 7 hours. By summing up the contribution of each mode to the overall cost, we get $\pi(\sigma_1) = (1 \cdot 30 + 3 \cdot 10) + (1 \cdot 10 + 2 \cdot 20) = 110\,[£]$ and $\pi(\sigma_2) = (3 \cdot 30 + 3 \cdot 10) + (2 \cdot 10 + 2 \cdot 20) = 180\,[£]$. Moreover, $\sigma_3 = \langle (1,3), (0,1), (2,2), (0,1), (1,3), (0,1), (2,2), (0,1), \ldots \rangle$ is a safe infinite run with the average cost $\pi_{avg}(\sigma_3) = 110/7\,[£/h]$. ◁

Given a simple linear automaton $\mathcal{A} = (M, A, \pi_c, \pi_d, [V_{min}, V_{max}], V_0, t_{max})$ with $V_{min} > 0$, consider automaton $\mathcal{A}' := (M, A, \pi_c, \pi_d, [0, V_{max} - V_{min}], V_0 - V_{min}, t_{max})$. Note that any finite (infinite) safe schedule σ in $\mathcal{A}$ is also safe in $\mathcal{A}'$ and its cost (limit-average cost, respectively) is the same. As a result we have the following observation which allows us to assume $V_{min} = 0$ from now on.

Observation 1: Any decision problem regarding (ϵ-)optimal (finite or infinite) schedules for simple linear hybrid automata, can be easily reduced to the same decision problem for simple linear hybrid automata with $V_{min} = 0$.

A *leap* is a sequence of two pairs $(m_k, t_k), (m_{k+1}, t_{k+1})$ in a schedule such that $m_{k+1} = 0$, $A(m_k) t_k \leq V_{max}$, and $A(m_k) t_k + A(m_{k+1}) t_{k+1} = 0$. A leap is of *type* $i \in M^+$ iff $m_k = i$. A *complete leap* is a leap such that $A(k) t_k = V_{max}$. By Δt_i and $\Delta \pi_i$ we denote the time duration and the cost of a complete leap of type $i \in M^+$, respectively. Note that $\Delta t_i = V_{max}/A(i) - V_{max}/A(0)$ and $\Delta \pi_i = \pi_d(i) + \pi_c(i) \cdot V_{max}/A(i)$. We also introduce $\pi_e(i) = (\Delta \pi_i - \pi_d(i))/\Delta t_i$ to be the effective continuous cost rate per time unit of using mode i as part of a leap. Note that a leap of type i that lasts for time t has the total cost of $\pi_d(i) + \pi_e(i) \cdot t$.

Running example continues. For instance $(1,3), (0,1)$ is a complete leap of type 1 and $(2,1), (0,\frac{1}{2})$ is an incomplete leap of type 2. Their costs are $(30 + 3 \cdot 10) + (0 + 1 \cdot 0) = 60\,[£]$ and $(10 + 1 \cdot 20) + (0 + \frac{1}{2} \cdot 0) = 30\,[£]$, respectively. We can calculate that $\Delta t_1 = 3\,[h]$, $\Delta t_2 = 2\,[h]$, $\Delta \pi_1 = 60\,[£]$, and $\Delta \pi_2 = 50\,[£]$. Moreover, $\pi_e(1) = 7\frac{1}{2}\,[£/h]$ and $\pi_e(2) = 13\frac{1}{3}\,[£/h]$. Note that the cost of this example incomplete leap of type 2 is $\pi_d(2) + \pi_e(2) \cdot (1 + \frac{1}{2}) = 30\,[£]$, which matches the cost that we computed earlier. ◁

D. Approximation algorithms

We study approximation algorithms for the minimisation cost problem in simple linear hybrid automata. An algorithm has a relative performance ρ iff for all inputs x the cost of the solution that it computes, $f(x)$, satisfies $OPT(x) \leq f(x) \leq (1 + \rho) \cdot OPT(x)$, where $OPT(x)$ is the optimal cost for the input x. We are particularly interested in polynomial-time approximation algorithms. A polynomial-time approximation scheme (PTAS) is an algorithm that, for every $\rho > 0$, runs in polynomial-time and has relative performance ρ. Note that the running time of a PTAS may depend in an arbitrary way on ρ. Therefore, we typically strive to find a fully polynomial-time approximation scheme (FPTAS) which is an algorithm that runs in polynomial-time in the size of the input and $1/\rho$.

In particular, the following two well-known optimisations problems have a FPTAS: the 0-1 Knapsack problem and the Unbounded Knapsack problem (see e.g. [24]). In the 0-1 Knapsack problem, we are given a knapsack with a fixed volume and a list of items, each with an integer volume and value. The aim is to pick a subset of these items that together do not exceed the volume of the knapsack and have the maximum total value. In the Unbounded Knapsack problem, the setting is the same, but there are unlimited number of copies of each item. Both problems are well-known to be NP-complete [25], but possess pseudo-polynomial algorithms and FPTASes (see e.g. [24]).

III. Optimal Schedules

We start with considering the easy case of infinite time horizons, before turning to the interesting case of finite time horizons.

A. Infinite time horizon

Let $j = \operatorname{argmin}_{i \in M^+} \Delta \pi_i / \Delta t_i$. Obviously, at all times $t = k \cdot \Delta t_j$ where $k \in \mathbb{N}$, using only complete leaps of type j is the cheapest finite schedule. Consequently, the limit superior of the average cost cannot be smaller than $\Delta \pi_j / \Delta t_j$. At the same time, the simple schedule that only uses complete leaps of type j realises this long-time average. Taking into account that $\operatorname{argmin}_{i \in M^+} \Delta \pi_i / \Delta t_i$ can be computed using logarithmic space, because multiplication, division and comparison can be [26], we get the following theorem.

Theorem 1: An optimal safe infinite schedule can be computed in deterministic LOGSPACE.

Running example continues. It is easy to check that $\sigma_4 = \langle (1,3),(0,1),(1,3),(0,1),\ldots\rangle$ is an optimal safe infinite run whose long-time average cost is $\pi_{avg}(\sigma_4) = 15\,[£/h]$. ◁

B. Finite time horizon

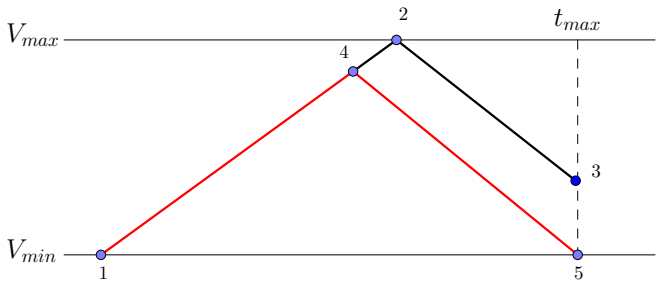

Fig. 1: Any optimal schedule can be assumed to reach value $V_{min} = 0$ at the end. Replacing timed actions $1 \to 2 \to 3$ in a finite safe schedule with timed actions $1 \to 4 \to 5$ reduces the cost of this schedule within its time horizon t_{max}.

We start with the following observation.

Proposition 1: For every safe schedule σ there exists a safe schedule σ' with the same or a lower cost and the value of the variable at the end of $run(\sigma')$ equal to $V_{min} = 0$.

Proof: We can see this illustrated in Figure 1. Let t be the first point of time during the execution of σ that the value of the variable equals $A(0) \cdot (t - t_{max})$. (Note that such a $t \in [0, t_{max}]$ exists.) We then construct σ' from σ by changing the behaviour in the last $t_{max} - t$ time units, choosing the idle mode there. As choosing the idle mode incurs no costs, this can only reduce the overall costs. □

In the remainder of this paper, we assume that all schedules have the property as stated in Proposition 1. In fact, we can show that there exists an optimal schedule of a very special form as stated by the following theorem.

Theorem 2: For every safe schedule σ there exists a safe schedule σ' consisting of a sequence of leaps where all but possibly the last one are complete and such that the cost of σ' is the same or lower than σ.

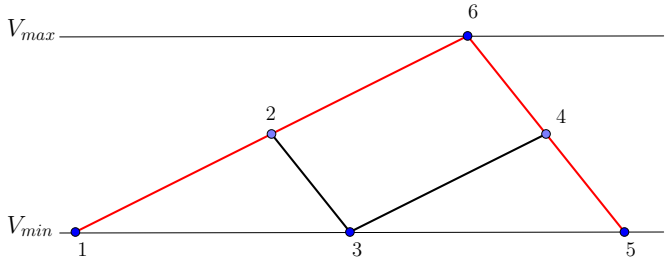

Fig. 2: Two incomplete leaps $1 \to 2 \to 3$ and $3 \to 4 \to 5$ being combined into one leap $1 \to 6 \to 5$.

Proof: Let $\sigma = \langle (m_1,t_1),(m_2,t_2),\ldots,(m_k,t_k)\rangle$ be any safe schedule. Define $T_\sigma(m) := \sum_{1\le i\le k: m_i = m} t_i$ to be the total time mode $m \in M^+$ is used for in σ. We define a schedule σ'' as follows: it starts with $\lfloor T_\sigma(m)A(m)/V_{max}\rfloor$ complete leaps of type m for each mode $m \in M^+$. At the end we add for each $m \in M^+$ an incomplete leap starting with a timed action $(m, T_\sigma(m) - A(m)\lfloor T_\sigma(m)A(m)/V_{max}\rfloor)$ if $T_\sigma(m)A(m)/V_{max}$ is not an integer. It is easy to see that σ'' is safe and no more expensive than σ, because each mode is used the same amount of time as in σ and the number of switches to any mode $m \in M^+$ is the same or smaller. To construct σ' we iterate the following until there is at most one incomplete leap left: take the first two incomplete leaps in σ'': $(m_1,t_1),(0,t_{01})$ and $(m_2,t_2),(0,t_{02})$. W.l.o.g. the continuous cost for mode m_1 is lower, i.e. $\pi_c(m_1)\cdot t_1/(t_1+t_{01}) \le \pi_c(m_2)\cdot t_2/(t_2+t_{02})$. We can then replace these two incomplete leaps by

- $(m_1,(t_1+t_2+t_{01}+t_{02})\cdot t_1/(t_1+t_{01})),(0,(t_1+t_2+t_{01}+t_{02})\cdot t_{01}/(t_1+t_{01}))$ if it is a leap, i.e. $(t_1+t_2+t_{01}+t_{02})\cdot t_1/(t_1+t_{01}) \le V_{max}$, see Figure 2
- one complete leap for m_1 and a shorter leap for m_2 such that the time delay of the two leaps is $t_1+t_2+t_{01}+t_{02}$, see Figure 3

This operation cannot increase the cost of the schedule, because the continuous cost of m_1 is the same or lower and the number of mode switches is the same or lower. At the same time the number of incomplete leaps is strictly reduced. □

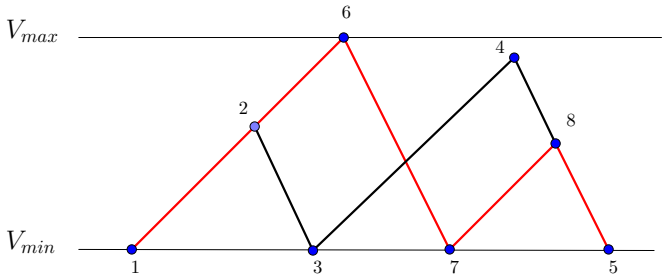

Fig. 3: Two incomplete leaps $1 \to 2 \to 3$ and $3 \to 4 \to 5$ being combined into one complete leap $1 \to 6 \to 7$ and one incomplete one $7 \to 8 \to 5$.

From Theorem 2, an optimal schedule exists, because for any fixed time horizon t_{max} there are only finitely many schedules of the form stated and no other schedule can have a better cost than all of them.

When we allow the initial value, V_0, of the variable to be non-zero at the beginning, then we can exploit a similar argument to show that it is safe to initially stay in the idle mode until either t_{max} is reached or the value of the variable has fallen to $V_{min} = 0$, whatever happens first.

Observation 2: For simple linear hybrid automata with $V_0 > V_{min}$ the following holds. For every safe schedule σ there exists a safe schedule σ' where initially the idle mode is active until the value of the variable is $V_{min} = 0$ (or, if this is earlier, for the complete duration t_{max}), followed by a sequence of leaps, where all but possibly the last one are complete and such that the cost of σ' is the same or lower cost than σ.

IV. NP-Completeness of Finite Time Horizon Optimal Control

In this section, we study the complexity of the optimal control problem for a finite time horizon. As usual, we analyse the complexity of the related decision problem:

> For a given cost C, is there a way to control the system in such a way that the total cost incurred for keeping the system in the safe zone for time t_{max} is at most C?

We show that this optimal control decision problem is NP-complete. We start by showing its hardness by a reduction from the Unbounded Knapsack problem, which is NP-complete [25]. For this reduction, it suffices to use a simpler problem, where all continuous costs π_c are 0. We refer to this problem as 0-cost rate optimal control decision problem.

Theorem 3: The 0-cost rate optimal control decision problem is NP-hard.

Proof: For the 0-cost rate optimal control problem, cost is only incurred when switching to a non-idle mode. This reduces our continuous optimisation problem to a discrete one, which is easier to relate to the Unbounded Knapsack problem.

In this setting, the natural constraint of the decision problem would be that the time horizon needs to be covered completely, which is reflected by

$$\sum_{i \in M^+} n_i \Delta t_i \geq t_{max}$$

This constraint says that the sum of the time of leaps is at least t_{max}. This includes the—possibly incomplete—last leap. Note that, in our discrete setting where the length of the leap does not influence the cost, the question of whether or not this cycle is complete is irrelevant for the total cost. For this reason, the n_i in this proof refer to all leaps, including the incomplete one.

Under this constraint, we would ask if there are natural numbers $(n_i)_{i \in M^+}$ such that $\sum_{i \in M^+} n_i \pi_d(i) \leq C$. These two constraints together are precisely the constraints used in the Unbounded Knapsack problem, where C represents the volume of the knapsack, $\pi_d(i)$ is the volume of item i—such that $\sum_{i \in M^+} n_i \pi_d(i) \leq C$ reflects the constraint volume of the knapsack—$\Delta\pi_i$ the value of item i, and t_{max} the lower bound on the overall value—such that $\sum_{i \in M^+} n_i \Delta\pi_i \geq t_{max}$ refers to the (decision version of) the optimisation criterion. □

The inclusion in NP of the general cost optimisation decision problem is straightforward, as the problem can be rewritten as an integer linear program. Assume that we know the type, j, of the incomplete leap at the end of the schedule. We can then solve the decision cost optimisation problem by solving the following integer linear constraint system.

$$\sum_{i \in M^+} n_i \Delta\pi_i + \big(t_{max} - \sum_{i \in M^+} n_i \Delta t_i\big)\pi_e(j) + \pi_d(j) \leq C$$

The first term in this expression is the total cost of the complete leaps and the other one is the total cost of the last (possibly incomplete) leap, whose duration is $t_{max} - \sum_{i \in M^+} n_i \Delta t_i$. Additionally we need the following constraints.

$$\bigwedge_{i \in M^+} n_i \in \mathbb{Z} \quad \wedge \quad \bigwedge_{i \in M^+} n_i \geq 0 \quad \wedge$$

$$t_{max} \geq \sum_{i \in M^+} n_i \Delta t_i \geq t_{max} - \Delta t_j$$

A solution to such a system of integer linear constraints, if it exists, can be guessed and verified in polynomial-time, which shows that the problem is in NP for a fixed j. Furthermore, j can be guessed at the same time, which gives us the following theorem.

Theorem 4: The finite time horizon optimal control decision problem is NP-complete.

V. Constant Factor Approximation Algorithm

We show here an approximation algorithm with a constant relative performance ≤ 2 for the cost minimisation problem in simple linear hybrid automata. We prove that it suffices to pick the cheapest schedule among the ones that only use one of the modes. Building on this constant approximation algorithm, we will show FPTAS for the same problem in the next section.

Algorithm 1 Constant factor approximation algorithm computing a finite schedule with the total cost at most twice the optimal one.

```
MinCost := ∞;  m := 0;
for i := 1 to K do
    Cost := π_e(i) · t_max + π_d(i) · k_i
    if Cost < MinCost then
        MinCost := Cost;  m := i;
    end if
end for
return schedule consisting of ⌊t_max/Δt_m⌋ complete
    leaps of type m followed by at most one more leap of
    type m for the remaining time t_max − ⌊t_max/Δt_m⌋ · Δt_m
```

Let $k_i := \lceil t_{max}/\Delta t_i \rceil$ denote the minimum number of leaps of type i that have to be used to cover the whole time horizon t_{max} by themselves. Let us introduce the following constant $\alpha := \max\{1, \max_{\{i \in M^+ | k_i \geq 2\}} k_i/(k_i - 1)\}$, where as usual $\max \emptyset = -\infty$. Note that $\alpha \leq 2$, because $k_i/(k_i - 1)$ decreases with k_i and $k_i \geq 2$.

Theorem 5: Algorithm 1 runs in deterministic LogSpace and returns an α-approximate schedule.

Proof: It is straightforward to see that the algorithm can be made to run in deterministic LogSpace, because it suffices to only store m inside the for loop and outputting the value and comparisons between arithmetic expressions can be performed in deterministic LogSpace [26].

To prove that the schedule returned has relative performance at most α, we first introduce some useful notation. Let X_j be the value of the *Cost* variable for $i = j$, i.e. $X_j = \pi_e(j) \cdot t_{max} + \pi_d(j) k_j$ for all $j \in M^+$, which is the minimum cost of a schedule that only uses leaps of type j. Let us assume

w.l.o.g. that the mode picked by Algorithm 1 is 1. Thus, for all $i \in M^+$ we have $X_1 \leq X_i$. Let σ be an optimal finite schedule of the form as described in Theorem 2. For any $i \in M^+$ let $n_i \in \mathbb{N}$ be the number of complete leaps of type i in σ. Let the last leap in σ be of type m and $0 \leq L \leq \Delta t_k$ be the time that it lasts for. Note that $L = t_{max} - \sum_{i \in M^+} n_i \Delta t_i$. From the definition of k_i we know that $k_i \Delta t_i \geq t_{max} \geq (k_i - 1)\Delta t_i$. It follows that $\Delta t_i / t_{max} \leq 1/(k_i - 1)$ for $k_i \geq 2$. If $n_i \geq 1$ then obviously $\Delta t_i / t_{max} \leq 1$, because otherwise we would have $L < 0$. Based on these, we note the following estimations:

$$1 - \frac{L}{t_{max}} = \sum_{i \in M^+} \frac{n_i \Delta t_i}{t_{max}} = \sum_{\{i \in M^+ | n_i \geq 1\}} \frac{n_i \Delta t_i}{t_{max}} =$$
$$\sum_{\{i \in M^+ | n_i \geq 1 \& k_i = 1\}} \frac{n_i \Delta t_i}{t_{max}} + \sum_{\{i \in M^+ | n_i \geq 1 \& k_i \geq 2\}} \frac{n_i \Delta t_i}{t_{max}} \leq$$
$$\sum_{\{i \in M^+ | n_i \geq 1 \& k_i = 1\}} \frac{n_i}{k_i} + \sum_{\{i \in M^+ | n_i \geq 1 \& k_i \geq 2\}} \frac{n_i}{k_i} \frac{k_i}{k_i - 1} \leq$$
$$\max\{1, \max_{\{i \in M^+ | k_i \geq 2\}} \frac{k_i}{k_i - 1}\} \sum_{i \in M^+} \frac{n_i}{k_i} = \alpha \sum_{i \in M^+} \frac{n_i}{k_i}$$

Moreover, we have the following. If $k_m \geq 2$ then $\frac{k_m L}{t_{max}} \leq \frac{k_m \Delta t_m}{t_{max}} \leq \frac{k_m}{k_m - 1} \leq \alpha$. If $k_m = 1$ then $\frac{k_m L}{t_{max}} \leq 1 \leq \alpha$, so in fact in both cases $\frac{k_m L}{t_{max}} \leq \alpha$.

We are now ready to give a lower bound on the total cost of the optimal schedule σ in terms of X_1. The total cost of σ is equal to the following expression.

$$\sum_{i \in M^+} (n_i \pi_d(i) + n_i \Delta t_i \pi_e(i)) + \pi_d(m) + L\pi_e(m) =$$
$$\sum_{i \in M^+} \frac{n_i}{k_i} (k_i \pi_d(i) + k_i \Delta t_i \pi_e(i)) + \pi_d(m) + L\pi_e(m) \geq$$
$$\sum_{i \in M^+} \frac{n_i}{k_i} (k_i \pi_d(i) + t_{max} \pi_e(i)) + \pi_d(m) + L\pi_e(m) =$$
$$\sum_{i \in M^+} \frac{n_i}{k_i} X_i + \pi_d(m) + L\pi_e(m) \geq$$
$$\sum_{i \in M^+} \frac{n_i}{k_i} X_1 + \pi_d(m) + L\pi_e(m) \geq$$
$$\frac{X_1}{\alpha} \left(1 - \frac{L}{t_{max}}\right) + \frac{k_m L}{\alpha t_{max}} \pi_d(m) + \frac{L}{\alpha} \pi_e(m) \geq$$
$$\frac{X_1}{\alpha} \left(1 - \frac{L}{t_{max}}\right) + \frac{L}{\alpha t_{max}} (k_m \pi_d(m) + t_{max} \pi_e(m)) =$$
$$\frac{X_1}{\alpha} \left(1 - \frac{L}{t_{max}}\right) + \frac{L}{\alpha t_{max}} X_k \geq$$
$$\frac{X_1}{\alpha} \left(1 - \frac{L}{t_{max}}\right) + \frac{L}{\alpha t_{max}} X_1 = \frac{X_1}{\alpha}$$

This shows that the cost of X_1 is at most α times the optimal cost, which concludes the proof. □

Running example continues. It is easy to check that $\sigma_5 = \langle (1,3), (0,1), (2,2), (0,1) \rangle$ is an optimal safe run whose cost is $\pi(\sigma_5) = 110\,[£]$. At the same time, a cheapest safe schedule consisting of leaps of type 1 is $\sigma_6 = \langle (1,3), (0,1), (1, \frac{9}{4}), (0, \frac{3}{4}) \rangle$ and of type 2 is $\sigma_7 = \langle (2,2), (0,1), (2,2), (0,1), (2, \frac{2}{3}), (0, \frac{1}{3}) \rangle$. Their costs are $\pi(\sigma_6) = 2 \cdot 20 + (3 + \frac{9}{4}) \cdot 10 = 112.5$ and $\pi(\sigma_7) = 3 \cdot 10 + 4\frac{2}{3} \cdot 20 = 123\frac{1}{3}$. Hence, Algorithm 1 will return σ_6 and the approximation ratio of this solution is 1.022. ◁

From the proof of Theorem 5 we can easily deduce the following corollary.

Corollary 1: Algorithm 1 returns an optimal schedule if $\pi_d(i) = 0$ for all $i \in M^+$.

Proof: Analysing the proof of Theorem 5 we can make the following observations. If $\pi_d(1) = \pi_d(i) = 0$ then the condition $X_1 \leq X_i$ implies that $\pi_e(1) \leq \pi_e(i)$. The cost of an optimal schedule σ is $\sum_{i \in M^+} n_i \Delta t_i \pi_e(i) + L\pi_e(m) \geq \sum_{i \in M^+} n_i \Delta t_i \pi_e(1) + L\pi_e(1) = t_{max} \pi_e(1) = X_1$. □

VI. FPTAS ALGORITHM

We show here that the cost minimisation problem for simple linear hybrid automata is in FPTAS by a polynomial time reduction to the 0-1 Knapsack problem, for which many FPTAS algorithms are available (see e.g. [24]).

Let c^* be the α-approximation, which can be computed using Algorithm 1, of the optimal cost o^*. Since $\alpha \leq 2$, to get an approximation to our optimal cost problem with a relative performance ρ, it suffices to find a solution with $c^* \rho / 2$ absolute performance. We split this into two equal parts of $\epsilon = c^* \rho / 4$. An optimal solution to the knapsack instance that we produce will provide us with a schedule with cost no greater than ϵ over the optimal one. Moreover, a solution to the knapsack instance with δ absolute error will provide a schedule with an $\epsilon + \delta$ absolute error. Therefore, it suffices to set $\delta = \epsilon$ to find a schedule with ρ relative performance. In our reduction, the value of the resulting knapsack problem is at most $4|M|$ times the optimal cost for safe schedules, so by using $\rho' = \rho / (8|M|)$, for the resulting knapsack problem we will find a near optimal solution with a relative performance ρ for simple linear hybrid automata. The running time of this procedures is in $\mathcal{O}(\mathsf{poly}(1/\rho)\mathsf{poly}(|M|)\mathsf{poly}(\text{size of the knapsack instance}))$. This suffices to establish the inclusion of the cost minimisation problem for simple linear hybrid automata in FPTAS.

Let σ be an optimal safe schedule consisting of a sequence of leaps where all but possibly the last one are complete, which has to exist due to Theorem 2. Let $m^* \in M^+$ be the mode used in the last leap in σ. Note that we can try all modes as candidates for m^*. For each mode $m \in M^+$ we build the following items for this knapsack problem instance: $\{(2^i \cdot \Delta t_m, 2^i \cdot \Delta \pi_m) \mid i \in \mathbb{N} \wedge 2^i \cdot \Delta \pi_m \leq c^* \wedge 2^i \cdot \Delta t_m \leq t_{max}\}$. Let $i^* \in \mathbb{N}$ be smallest such that $2^{-i^*} \cdot (\Delta \pi_{m^*} - \pi_d(m^*)) \leq \epsilon$. For m^* we add the following extra multiset of items: $\{(2^{-i} \cdot \Delta t_{m^*}, 2^{-i} \cdot (\Delta \pi_{m^*} - \pi_d(m^*))) \mid i \in \mathbb{Z}_+ \wedge i \leq i^* \wedge 2^{-i} \cdot (\Delta \pi_{m^*} - \pi_d(m^*)) \leq c^*\}$ and additionally $(2^{-i^*} \cdot \Delta t_{m^*}, 2^{-i^*} \cdot (\Delta \pi_{m^*} - \pi_d(m^*)))$, which is a copy of an element already in the multiset. Let t_Σ be the time span of all items in this knapsack instance. We set the volume of this 0-1 knapsack instance to be $t_\Sigma - t_{max}$.

The just produced knapsack problem has the following properties:

- the size of its description is polynomial in the size of the original problem including the relative performance
- if there is an incomplete leap of m^* in σ, it can be overestimated by stringing together the fractional copies of leaps (without start-up cost), so that we do not exceed the volume by $2^{-i^*} \cdot \Delta t_{m^*}$ or more, and if there is no incomplete leap in σ, one complete leap of m^* of σ can be replaced by all of these fractional copies of leaps of m^*; the remaining complete leaps can be replaced by sums of complete leaps of the respective type.
- The volume of these items is $\geq t_{max}$. Let v^* be the value of these items. Then $v^*+\pi_d(m^*)-\epsilon \leq o^* \leq v^*+\pi_d(m^*)$.
- Let V_Σ be the value of all items in the multiset. For any solution to the knapsack problem with value V we get a schedule σ' with cost $\leq V_\Sigma - V + \pi_d(m^*)$.

Lemma 1: Solving this knapsack instance with a relative performance of $\rho/(8|M|)$ gives us a safe schedule with relative performance of ρ.

Corollary 2: Solving the optimal control problem for simple linear hybrid automata with relative performance ρ takes

$$\mathcal{O}(\mathsf{poly}(1/\rho)\mathsf{poly}(\text{size of the instance}))$$

time and is therefore in FPTAS.

VII. Experimental Results

In this section, we compare the performance of the several algorithms that we devised for the optimal finite time horizon control problem for simple linear hybrid automata. Namely, we compare the constant factor approximation algorithm from Section V, an algorithm based on the integer programming formulation of our optimisation problem stated explicitly in this section as Algorithm 2, and the FPTAS from Section VI. All algorithms were implemented in Matlab version 2016a and all tests were run on a HP DV6-6199ee with Intel core i7 2GHz CPU and 8GB of RAM. For integer linear programming (ILP) we used the standard library provided in Matlab. For larger instances the Matlab's ILP library was running out of memory and crashing. However, it was running very fast (in less than 2 seconds) for the instances it did not crash on, which shows that the library may have some memory management issues.

We tested our algorithms on randomly generated instances with strongly correlated coefficients as defined in Section 5.5 of [24] for the 0-1 knapsack problem. As tested in [24], such instances are some of the hardest to solve for most algorithms for the 0-1 knapsack problem. In our setting, an instance of this type is defined as follows. First, for all $i \in M^+$, we pick Δt_i uniformly at random from the interval $[1, R]$, where R is some constant. We then assign $\Delta\pi_i = \Delta t_i + \frac{R}{10}$ for $i \in M^+$ as well as $\pi_d(i) = \gamma\Delta\pi_i$, where γ is picked uniformly at random from the $[0.1, 0.4]$ interval. We also set $V_{min} = 18°\text{C}$, $V_{max} = 22°\text{C}$, and $A(0) = -1$. Based on this information, we can reverse engineer all the other parameters $A(i)$ and $\pi_c(i)$ for all $i \in M^+$ of this simple linear hybrid automaton instance. For each instance we consider various lengths of the time horizon $t_{max} = h \cdot \sum_{i \in M^+} \Delta t_i$, where $h = \{0.2, 0.4, 0.6, 0.8, 1\}$. We tested our algorithm for different values of R, but since there was no significant difference in the relative performance of the algorithms, we only include the running times for $R = 6000$.

Algorithm 2 Integer Linear Programming algorithm for the optimal cost problem.

1: Solve the following ILP for all possible $j \in M^+$:

$$\text{Min} \quad \sum_{i \in M^+} n_i \Delta\pi_i + \big(t_{max} - \sum_{i \in M^+} n_i \Delta t_i\big)\pi_e(j) + \pi_d(j)$$

Subject to the following constraints:

$$\bigwedge_{i \in M^+} n_i \in \mathbb{Z} \quad \wedge \quad \bigwedge_{i \in M^+} n_i \geq 0 \quad \wedge$$

$$t_{max} \geq \sum_{i \in M^+} n_i \Delta t_i \geq t_{max} - \Delta t_j$$

2: Pick j^* and the corresponding solution $(n_i)_{i \in M^+}$ with the minimum value of the objective function.

3: **return** schedule consisting of n_i complete leaps of type i for all $i \in M^+$ followed by a leap of type j^* and duration $t_{max} - \sum_{i \in M^+} n_i \Delta t_i$

	$\forall_i \Delta t_i \in [1, 6000]$				
t_{max}	$0.2\sum_i \Delta t_i$	$0.4\sum_i \Delta t_i$	$0.6\sum_i \Delta t_i$	$0.8\sum_i \Delta t_i$	$\sum_i \Delta t_i$
K=2	20.3	29.3	21.3	31.7	19.1
K=4	41.4	57.6	45.5	70.4	43
K=6	79	93.7	—	—	—
K=8	—	123.3	—	—	—
K=10	128.4	—	101.7	—	—
K=20	—	—	—	—	—
K=30	—	—	—	—	—
K=40	—	—	—	—	—
K=50	—	—	—	—	—

TABLE I: Average running time over 1000 random instances each for the Mixed Integer Linear Programming algorithm (in milliseconds), where K is the number of modes.

Table I shows the average execution time of the optimal integer linear programming (ILP) algorithm in milliseconds. The dashed cells mean that the algorithm failed to give an answer and the system crashed. We can conclude that the algorithm performs very well and, if it does not crash, it returns the optimal schedule quickly (in less than 2 seconds). We plan to use a different ILP library in the future to check whether the crashes are just due to MatLab's ILP library implementation.

Table II shows the average execution time of the constant factor approximation algorithm in milliseconds. As the algorithm is really simple, the execution time is really small for all the instances that we tried it on. Although this algorithm in general can return a solution with twice the optimal cost in the worst-case, by comparing its solutions with the optimal ones found by the ILP algorithm, we found that for all these instances the relative performance was below 10%. Moreover, as we showed in Section V, the longer the time horizon is,

	$\forall_i \Delta t_i \in [1, 6000]$				
t_{max}	$0.2 \sum_i \Delta t_i$	$0.4 \sum_i \Delta t_i$	$0.6 \sum_i \Delta t_i$	$0.8 \sum_i \Delta t_i$	$\sum_i \Delta t_i$
K=2	0.012	0.012	0.012	0.012	0.012
K=4	0.012	0.012	0.012	0.012	0.012
K=6	0.012	0.012	0.014	0.014	0.014
K=8	0.012	0.013	0.013	0.015	0.015
K=10	0.013	0.013	0.014	0.014	0.015
K=20	0.021	0.021	0.025	0.024	0.023
K=30	0.022	0.024	0.025	0.025	0.025
K=40	0.022	0.025	0.026	0.025	0.025
K=50	0.023	0.024	0.023	0.024	0.024

TABLE II: Average running time over 1000 random instances each for the constant factor approximation algorithm (in milliseconds), where K is the number of modes.

the better are the worst-case guarantees that this algorithm provides. So if each heater has to be used at least 11 times by itself to cover the whole time horizon (i.e. $k_i \geq 11$ for all $i \in M^+$), the cost of the solution returned by this algorithm is at most $1/10 = 10\%$ higher than the optimal one.

	$\forall_i \Delta t_i \in [1, 6000]$			
t_{max}	$0.05 \sum_i \Delta t_i$	$0.1 \sum_i \Delta t_i$	$0.2 \sum_i \Delta t_i$	$\{0.4, \ldots, 1\} \sum_i \Delta t_i$
K=2	21.61	84.55	332.28	—
K>2	—	—	—	—

TABLE III: The average running time over 1000 random instances each for the FPTAS algorithm (in seconds), where K is the number of modes.

Finally, Table III shows the average execution time in seconds for the FPTAS approximation algorithm with $\rho = 10\%$. The dashes mean that the algorithm took too much time to produce an answer (with a one hour timeout). We found FPTAS to perform poorly on these very hard instances. However, on uncorrelated instances this algorithm performs very well. We leave as future work the development of an FPTAS that works well in practice for all kind of instances.

Based on these tests we can conclude that, for simple linear hybrid automata with a small number of modes, it is best to use the optimal integer programming algorithm as it runs quickly and gives the exact optimal schedule. In all other instances, the constant factor approximation algorithm is the best choice as it runs really quickly and most of the time gives a near-optimal answer.

VIII. Conclusions

Linear hybrid systems are computationally challenging. In particular, safety and reachability are undecidable already for three variables. We have identified the class of simple linear hybrid systems as a class that arises naturally when studying the optimal control of heating or cooling systems: there is only one continuous variable (the temperature in our setting) in addition to the time. Although it was to be expected that the optimal control for this model is decidable, the fact that this problem is both NP-complete and admits an FPTAS was not. Only a small number of NP-hard problems admit a FPTAS, i.e. can be approximated with relative precision ρ, in polynomial time in the size of the input and $1/\rho$. Most NP-hard problems can be shown to be inapproximable within a constant relative performance in polynomial time unless P=NP. The existence of FPTAS, besides offering a cheap approximation in every desired precision, often indicates that good standard solvers will normally behave well. In our example, this may be viewed as a reason why the ILP solver performs so well on our benchmark. (That is, when it did not crash, which was most likely due to MatLab's ILP library that we used.)

Summing up, we have identified a simple subclass of linear hybrid automata with an easy (LogSpace) optimal control problem over an infinite time horizon, and a control problem over a finite time horizon which is fast to approximate (FPTAS). We believe that this class is of interest because, broadly speaking, it is just tractable enough. Adding to the collection of classes with de-facto efficient algorithms expands the set of problems that we can handle. As the next step, we plan to analyse the model where there can be multiple modes with negative slopes (i.e. $A(i) < 0$) apart from the idle mode. Such a generalisation, however, breaks down the existence of an optimal schedule consisting of leaps, which was crucial for the development of an FPTAS algorithm for this problem. We conjecture that the optimal control problem for such a model is still decidable, but with a higher computational complexity.

Acknowledgment

This work was supported by EPSRC EP/M027287/1 grant "Energy Efficient Control".

References

[1] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, "The algorithmic analysis of hybrid systems," *Theoretical computer science*, vol. 138, no. 1, pp. 3–34, 1995.

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

[3] "TRaNsient SYstems Simulation Program," http://sel.me.wisc.edu/trnsys/.

[4] "EnergyPlus: building energy simulation program," https://energyplus.net/.

[5] "IBPT: International Building Physics Toolbox in Simulink," http://www.ibpt.org/.

[6] R. Alur, C. Courcoubetis, T. Henzinger, and P. Ho, "Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems," in *Hybrid Systems*. Springer Berlin / Heidelberg, 1993, pp. 209–229.

[7] T. A. Henzinger, "The theory of hybrid automata," in *Proc. of the 11th IEEE LICS'96 Symposium*, Washington, DC, USA, 1996.

[8] E. Asarin, V. P. Mysore, A. Pnueli, and G. Schneider, "Low dimensional hybrid systems – decidable, undecidable, don't know," *Information and Computation*, vol. 211, pp. 138–159, Feb. 2012.

[9] F. Laroussinie, N. Markey, and P. Schnoebelen, "Model checking timed automata with one or two clocks," in *CONCUR 2004-Concurrency Theory*. Springer, 2004, pp. 387–401.

[10] R. Alur and D. L. Dill, "A theory of timed automata," *Theoretical Computer Science*, vol. 126, no. 2, pp. 183–235, Apr. 1994.

[11] R. Alur, A. Trivedi, and D. Wojtczak, "Optimal scheduling for constant-rate mulit-mode systems," in *Proc. of Hybrid Systems: Computation and Control 2012*.

[12] D. Wojtczak, "Optimal Control for Linear-Rate Multi-mode Systems," in *Formal Modeling and Analysis of Timed Systems*, ser. Lecture Notes in Computer Science, V. Braberman and L. Fribourg, Eds. Springer Berlin Heidelberg, Aug. 2013, no. 8053, pp. 258–273.

[13] T. X. Nghiem, M. Behl, R. Mangharam, and G. J. Pappas, "Green scheduling of control systems for peak demand reduction," in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*. IEEE, 2011, pp. 5131–5136.

[14] T. X. Nghiem, G. J. Pappas, and R. Mangharam, "Event-based green scheduling of radiant systems in buildings," in *American Control Conference (ACC), 2013*. IEEE, 2013, pp. 455–460.

[15] P. Bouyer, T. Brihaye, M. Jurdziński, R. Lazić, and M. Rutkowski, "Average-Price and Reachability-Price Games on Hybrid Automata with Strong Resets," in *Formal Modeling and Analysis of Timed Systems*, ser. Lecture Notes in Computer Science, F. Cassez and C. Jard, Eds. Springer Berlin Heidelberg, Sep. 2008, no. 5215, pp. 63–77.

[16] P. Bouyer, "Weighted Timed Automata: Model-Checking and Games," *Electronic Notes in Theoretical Computer Science*, vol. 158, pp. 3–17, May 2006.

[17] E. Camacho, C. Bordons, E. Camacho, and C. Bordons, *Model predictive control*. Springer Berlin, 1999, vol. 303.

[18] F. Oldewurtel, A. Ulbig, A. Parisio, G. Andersson, and M. Morari, "Reducing peak electricity demand in building climate control using real-time pricing and model predictive control," in *Decision and Control (CDC), 2010 49th IEEE Conference on*, 2010, pp. 1927 –1932.

[19] Y. Ma, F. Borrelli, B. Hencey, B. Coffey, S. Bengea, and P. Haves, "Model predictive control for the operation of building cooling systems," in *American Control Conference (ACC), 2010*, 30 2010-july 2 2010, pp. 5106 –5111.

[20] B. Li and A. G. Alleyne, "Optimal on-off control of an air conditioning and refrigeration system," in *American Control Conference (ACC), 2010*. IEEE, 2010, pp. 5892–5897.

[21] A. David, P. G. Jensen, K. G. Larsen, M. Mikuionis, and J. H. Taankvist, "Uppaal Stratego," in *Tools and Algorithms for the Construction and Analysis of Systems*, ser. Lecture Notes in Computer Science, C. Baier and C. Tinelli, Eds. Springer Berlin Heidelberg, no. 9035, pp. 206–211. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-662-46681-0_16

[22] A. David, K. G. Larsen, A. Legay, M. Mikuionis, and Z. Wang, "Time for Statistical Model Checking of Real-Time Systems," in *Computer Aided Verification*, ser. Lecture Notes in Computer Science, G. Gopalakrishnan and S. Qadeer, Eds. Springer Berlin Heidelberg, no. 6806, pp. 349–355. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-22110-1_27

[23] A. David, D. Du, K. G. Larsen, A. Legay, and M. Mikuionis, "Optimizing Control Strategy Using Statistical Model Checking," in *NASA Formal Methods*, ser. Lecture Notes in Computer Science, G. Brat, N. Rungta, and A. Venet, Eds. Springer Berlin Heidelberg, no. 7871, pp. 352–367. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-38088-4_24

[24] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004.

[25] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*. San Francisco: W. H. Freeman, Apr. 1979.

[26] A. Chiu, G. I. Davida, and B. E. Litow, "Division in logspace-uniform NC^1," *ITA*, vol. 35, no. 3, pp. 259–275, 2001.