Undecidability of Quantized State Feedback Control for Discrete Time Linear Hybrid Systems

Federico Mari, Igor Melatti, Ivano Salvo, and Enrico Tronci

Computer Science Department, Sapienza University of Rome, Italy
{mari,melatti,salvo,tronci}@di.uniroma1.it

Abstract. We show that the existence of a quantized controller for a given *Discrete Time Linear Hybrid System* (DTLHS) is undecidable. This is a relevant class of controllers since *control software* always implements a quantized controller. Furthermore, we investigate the relationship between dense time modelling and discrete time modelling by showing that any *Rectangular Hybrid Automaton* (and thus, any *Timed Automaton*) can be modelled as a DTLHS.

1 Introduction

Many embedded systems are software based control systems. A software based control system consists of two main subsystems: the *controller* and the *plant*. Typically, the plant is a physical system consisting, for example, of mechanical or electrical devices while the controller consists of *control software* running on a microcontroller. In an endless loop, each T seconds (sampling time), the controller, after an *Analog-to-Digital* (AD) conversion (*quantization*), reads sensor outputs from the plant and, possibly after a *Digital-to-Analog* (DA) conversion, sends commands to plant actuators. The controller selects commands in order to guarantee that the closed loop system (that is, the system consisting of both plant and controller) meets given safety and liveness properties, i.e. system level specifications.

Formal verification of system level specifications for software based control systems requires modelling both continuous systems (typically, the plant) as well as discrete systems (the controller). This is typically done using *Hybrid Systems* (e.g., see [3, 2, 11, 14, 9]).

In [15], we presented a constructive necessary condition and a constructive sufficient condition for the existence of a (*quantized sampling*) controller for a software based control system when the plant is modelled using a *Discrete Time Linear Hybrid System* (DTLHS).

From [12] we know that the existence of a sampling controller is undecidable even for relatively simple linear hybrid automata. Considering that, given a *quantization schema* (i.e. number of bits used in AD conversion), the number of quantized sampling controllers is finite and that when using DTLHSs also the plant is modelled using a discrete model of time, one may be led to think that the existence of a quantized sampling controller might be decidable. In this paper we show that this problem is also undecidable.

Furthermore, we investigate the relationship between dense time modelling and discrete time modelling by showing that the class of *Rectangular Hybrid Automata* (RHA) [13] (and thus, the class of *Timed Automata* (TA) [3, 14]) can be encoded into the class of DTLHSs.

Our Main Contributions A DTLHS (e.g., see [5, 15] and citations thereof) is a discrete time hybrid system whose dynamics is defined as a linear predicate, i.e. a boolean combination (without negation) of linear constraints on its continuous as well as discrete variables. A large class of hybrid systems, including mixed-mode analog circuits, can be modelled using DTLHSs. System level safety as well as liveness specifications may be modelled as sets of states defined in turn as linear predicates. In our setting, as always in control problems, liveness constraints define the set of states that any evolution of the closed loop system should eventually reach (*goal states*). Our main contributions are the following.

First, we show that the existence of a quantized sampling controller for DTLHSs, meeting given safety and liveness specifications is undecidable (Section 5). We prove such a result by showing that any two-counter machine can be coded as a DTLHS thereby extending to DTLHSs the proof technique in [12].

Despite that, the non-complete algorithm in [15] usually succeeds in control software synthesis for meaningful hybrid systems. The main ingredient of our approach in [15] is to reduce the nondeterminism of a finite state abstraction of a given DTLHS: in Section 6, we show that also finding the "best" abstraction (in order to maximize the possibilities of finding a controller) involves to solve an undecidable problem.

Finally, we show that any RHA can be modelled as a DTLHS (Section 7). Since a TA is also an RHA, this implies that any TA can be modelled as a DTLHS. Such an embedding sheds light on how, by exploiting availability of real valued state and input variables, dense time behaviours can be modelled using discrete time behaviours.

Related Work TAs [3, 14] are a subset of RHAs [13] which, in turn, are a subset of *Linear Hybrid Automata* (LHA) [2, 11]. Undecidability results of the control synthesis problem for dense as well as discrete time linear hybrid systems have been presented in [13, 12, 19, 4]. A more general problem is considered in [7], namely the discrete time control with unknown sampling rate, that is undecidable even for TA. Moreover, we note that none of the above papers addresses the issue of quantized control. In [15], we presented a non-complete algorithm for DTLHS quantized sampling control synthesis from formal system level specifications, without addressing the issue of decidability.

Indeed, to the best of our knowledge, no previously published work has addressed the issue of decidability of existence of a quantized sampling controller for DTLHSs.

The relationship between dense time models and discrete time models has been extensively studied in control engineering (e.g., see [6]) with the goal of *approximating* dense time dynamics with discrete time ones. Here we present an *exact* representation of RHA as DTLHSs thus showing that, as long as real valued variables are available, interesting dense time behaviors can also be *exactly* modelled using a discrete time approach.

2 Background

We denote with $[n]$ the initial segment $\{1, \ldots, n\}$ of the natural numbers. We denote with $X = x_1, \ldots, x_n$ a finite sequence of distinct variables, that we may regard, when convenient, as a set. Each variable x ranges on a known (bounded or unbounded) interval $\mathcal{D}_x$ either of the reals (*continuous variables*) or of the integers (*discrete variables*). We denote with $\mathcal{D}_X$ the set $\prod_{x \in X} \mathcal{D}_x$. If $X = \varnothing$ then $\mathcal{D}_X = \{\epsilon\}$, where ϵ is an arbitrary constant. Boolean variables are discrete variables ranging on the set $\mathbb{B} = \{0,\ 1\}$. If x is a boolean variable, we write $\bar{x}$ for its complement. We denote with X^r (resp. X^d, X^b) the sequence of real (resp. discrete, boolean) variables in X.

A *linear expression* $L(X)$ over a sequence of variables X is a linear combination $\sum_i a_i x_i$ of variables in X with rational coefficients. A *linear constraint* over X (or simply a *constraint*) is an expression of the form $L(X) \bowtie b$ where $L(X)$ is a linear expression over X, $\bowtie$ is one of $\leq$, $\geq$, $=$ and b is a rational constant.

Predicates are inductively defined as follows. A constraint $C(X)$ over a sequence of variables X is a predicate on X. If $A(X)$ and $B(X)$ are predicates on X, then $(A(X) \wedge B(X))$ and $(A(X) \vee B(X))$ are predicates on X. Parentheses may be omitted, assuming usual associativity and precedence rules of logical operators. A *conjunctive predicate* is a conjunction of constraints.

Let $P(X)$ be a predicate. A variable $x \in X$ is said to be *bounded* in P if there exist a, $b \in \mathcal{D}_x$ such that $P(X)$ implies $a \leq x \leq b$. In such a case, we denote a with $\inf(x)$ and b with $\sup(x)$. A predicate P is *bounded* if all its variables are bounded. Let a be a rational number and x be a bounded variable. We write $\sup(ax)$ (resp. $\inf(ax)$) for $a \sup(x)$ (resp. $a \inf(x)$) if $a \geq 0$ and for $a \inf(x)$ (resp. $a \sup(x)$) if $a < 0$. We write $\sup(L(X))$ for $\sum_{i=1}^{n} \sup(a_i x_i)$ and $\inf(L(X))$ for $\sum_{i=1}^{n} \inf(a_i x_i)$.

A *valuation* over a sequence of variables X is a function v that maps each variable $x \in X$ to a value $v(x)$ in $\mathcal{D}_x$. We also call valuation the sequence of values $X^* = v(x_1), \ldots, v(x_n)$. A *satisfying assignment* to a predicate P over X is a valuation X^* such that $P(X^*)$ holds. Two predicates P and Q over X are *equivalent* if they have the same set of satisfying assignments. They are equisatisfiable, if P is satisfiable if and only if Q is satisfiable.

Given a constraint $C(X)$ and a fresh boolean variable $y \notin X$, the *guarded constraint* $y \to C(X)$ (if y then $C(X)$) denotes the predicate $((y = 0) \vee C(X))$. Similarly, we use $\bar{y} \to C(X)$ to denote the predicate $((y = 1) \vee C(X))$. A *guarded predicate* is a conjunction of either constraints or guarded constraints. A bounded guarded predicate can be transformed into a conjunctive predicate, by observing that a guarded constraint $z \to (L(X) \leq b)$ (resp. $\bar{z} \to (L(X) \leq b)$) is equivalent to the constraint $(\sup(L(X)) - b)z + L(X) \leq \sup(L(X))$ (resp. $(b - \sup(L(X)))z + L(X) \leq b$). Therefore, the following proposition holds.

Proposition 1. *For each bounded guarded predicate $P(X)$, there exists an equivalent conjunctive predicate $Q(X)$.*

3 Labeled Transition Systems

In this section we define the reachability and the control problem for *Labeled Transition Systems* (LTSs), by extending to possibly infinite LTSs the definitions in [18, 8] for finite LTSs.

An LTS $\mathcal{S}$ is a tuple (S, A, T) where S is a possibly infinite (even possibly uncountable) set of *states*, A is a possibly infinite (even possibly uncountable) set of *actions*, and $T : S \times A \times S \to \mathbb{B}$ is the *transition relation* of $\mathcal{S}$. Given a state $s \in S$ and an action $a \in A$, we denote with $\text{Adm}(\mathcal{S}, s)$ the set of actions admissible in s, that is $\text{Adm}(\mathcal{S}, s) = \{a \in A \mid \exists s' T(s, a, s')\}$ and with $\text{Img}(\mathcal{S}, s, a)$ the set of next states from s via a, that is $\text{Img}(\mathcal{S}, s, a) = \{s' \in S \mid T(s, a, s')\}$. $\mathcal{S}$ is said to be *deterministic* if, for all $s \in S, a \in A$, $|\text{Img}(\mathcal{S}, s, a)| \leq 1$. We call *self-loop* a transition of the form $T(s, a, s)$.

Given two LTSs $\mathcal{S}_1 = (S_1, A_1, T_1)$ and $\mathcal{S}_2 = (S_2, A_2, T_2)$, we say that $\mathcal{S}_1$ and $\mathcal{S}_2$ are *isomorphic*, notation $\mathcal{S}_1 \simeq \mathcal{S}_2$, if there exist two bijective maps $f_S : S_1 \to S_2$ and $f_A : A_1 \to A_2$ such that for all $s \in S_1$, for all $a \in A_1$ $T_1(s, a, s')$ holds if and only if $T_2(f_S(s), f_A(a), f_S(s'))$ holds.

Given two LTSs $\mathcal{S}_1 = (S, A, T_1)$ and $\mathcal{S}_2 = (S, A, T_2)$, we say that $\mathcal{S}_1$ *refines* $\mathcal{S}_2$ (notation $\mathcal{S}_1 \sqsubseteq \mathcal{S}_2$) iff $T_1(s, a, s')$ implies $T_2(s, a, s')$ for each state $s, s' \in S$ and action $a \in A$. The refinement relation is a partial order on LTSs. Informally speaking, the LTS $\mathcal{S}_1$ refines the LTS $\mathcal{S}_2$ if the set of transitions of $\mathcal{S}_1$ is a subset of the set of transitions of $\mathcal{S}_2$.

A *run* or *path* for an LTS $\mathcal{S}$ is a sequence $\pi = s_0, a_0, s_1, a_1, s_2, a_2, \ldots$ of states s_t and actions a_t s. t. $\forall t \geq 0$ $T(s_t, a_t, s_{t+1})$. The length $|\pi|$ of a finite run is the number of actions in π. The t-th state element of π is denoted by $\pi^{(S)}(t)$, and $\pi^{(A)}(t)$ denotes the t-th action element of π, that is $\pi^{(S)}(t) = s_t$, and $\pi^{(A)}(t) = a_t$.

Definition 1. *A* reachability problem *is a triple $(\mathcal{S}, I, G)$, where $\mathcal{S}$ is an LTS (S, A, T), and $I, G \subseteq S$. G is* reachable *from I if there exists a run π of $\mathcal{S}$ such that $\pi^{(S)}(0) \in I$ and $\pi^{(S)}(t) \in G$ for some $t \in \mathbb{N}$.*

3.1 LTS Control Problem

A *controller* for an LTS $\mathcal{S}$ is used to restrict the dynamics of $\mathcal{S}$ so that all states in the initial region will reach in one or more steps the goal region. A *strong controller* ensures that the closed loop system meets liveness specifications under a pessimistic view of nondeterminism (worst case distance J_s defined below), whereas a *weak controller* assumes an optimistic view of nondetermism (best case distance J_w defined below). In the following, we formalize such concepts by defining strong and weak solutions to an LTS control problem. In what follows, let $\mathcal{S} = (S, A, T)$ be an LTS, I, $G \subseteq S$ be, respectively, the *initial* and *goal* regions of $\mathcal{S}$.

Definition 2. *A* controller *for $\mathcal{S}$ is a function $K : S \times A \to \mathbb{B}$ such that $\forall s \in S$, $\forall a \in A$, if $K(s, a)$ then $\exists s'$ $T(s, a, s')$. The* domain *of K is the set* $\text{dom}(K)$ *of all states for which at least a control action is enabled. Formally,* $\text{dom}(K) = \{s \in S \mid \exists a\ K(s, a)\}$.

$\mathcal{S}^{(K)}$ *denotes the* closed loop system*, that is the LTS $(S, A, T^{(K)})$, where $T^{(K)}(s, a, s') = T(s, a, s') \wedge K(s, a)$.*

We call a path π *fullpath* if either it is infinite or its last state $\pi^{(S)}(|\pi|)$ has no successors (i.e. $\mathrm{Adm}(\mathcal{S}, \pi^{(S)}(|\pi|)) = \varnothing$). We denote with $\mathrm{Path}(s, a)$ the set of fullpaths starting in state s with action a, i.e. the set of fullpaths π such that $\pi^{(S)}(0) = s$ and $\pi^{(A)}(0) = a$.

Given a path π in $\mathcal{S}$, we define $j(\mathcal{S}, \pi, G)$ as follows. If there exists $n > 0$ such that $\pi^{(S)}(n) \in G$, then $j(\mathcal{S}, \pi, G) = \min\{n \mid n > 0 \wedge \pi^{(S)}(n) \in G\}$. Otherwise, $j(\mathcal{S}, \pi, G) = +\infty$. We require $n > 0$ since our systems are non-terminating and each controllable state (including a goal state) must have a path of positive length to a goal state. Taking $\sup \varnothing = +\infty$ and $\inf \varnothing = +\infty$, the *worst case distance* (pessimistic view) of a state s from the goal region G is $J_s(\mathcal{S}, G, s) = \sup\{j_s(\mathcal{S}, G, s, a) \mid a \in \mathrm{Adm}(\mathcal{S}, s)\}$, where $j_s(\mathcal{S}, G, s, a) = \sup\{j(\mathcal{S}, G, \pi) \mid \pi \in \mathrm{Path}(s, a)\}$. The *best case distance* (optimistic view) of a state s from the goal region G is $J_w(\mathcal{S}, G, s) = \sup\{j_w(\mathcal{S}, G, s, a) \mid a \in \mathrm{Adm}(\mathcal{S}, s)\}$, where $j_w(\mathcal{S}, G, s, a) = \inf\{j(\mathcal{S}, G, \pi) \mid \pi \in \mathrm{Path}(s, a)\}$.

Definition 3. *A* control problem *for $\mathcal{S}$ is a triple $\mathcal{P} = (\mathcal{S}, I, G)$. A* strong *(resp.* weak*) solution to $\mathcal{P}$ is a controller K for $\mathcal{S}$, such that $I \subseteq \mathrm{dom}(K)$ and for all $s \in \mathrm{dom}(K)$, $J_s(\mathcal{S}^{(K)}, G, s)$ (resp. $J_w(\mathcal{S}^{(K)}, G, s)$) is finite.*

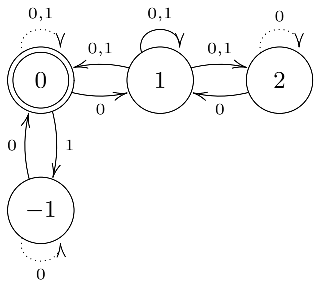

Fig. 1. The LTS $\mathcal{S}_1$ in Example 1.

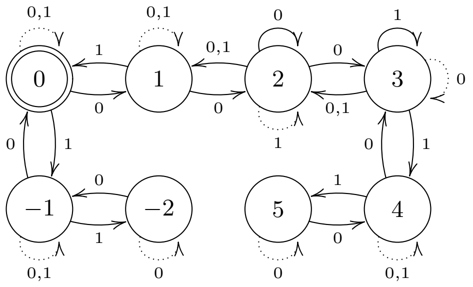

Fig. 2. The LTS $\mathcal{S}_2$ in Example 1.

Example 1. Let $\mathcal{S}_1 = (S_1, A_1, T_1)$ be the LTS in Fig. 1 and let $\mathcal{S}_2 = (S_2, A_2, T_2)$ be the LTS in Fig. 2. S_1 is the integer interval $[-1, 2]$ and $S_2 = [-2, 5]$. $A_1 = A_2 = \{0, 1\}$ and the transition relations T_1 and T_2 are defined by all continuous arrows in the pictures (dotted arrows will be considered later in Example 4). Let $I_1 = S_1$, $I_2 = S_2$ and let $G = \{0\}$.

There is no strong solution to the control problem $(\mathcal{S}_1, I_1, G)$. Because of the self-loops of the state 1, we have that both $j_s(\mathcal{S}_1, G, 1, 0) = +\infty$ and $j_s(\mathcal{S}_1, G, 1, 1) = +\infty$. On the other hand, the controller K_1, defined by $K_1(s, a) \equiv a = 0$, that enables action 0 in all states, is a weak solution.

The controller K_2, defined by $K_2(s, a) \equiv ((s = 2 \vee s = 1) \wedge a = 1) \vee (s \neq 1 \wedge s \neq 2 \wedge a = 0)$ is a the most general optimal strong solution to the control problem $(\mathcal{S}_2, I_2, G)$.

We end this section, by recalling a well-known result that relates strong and weak solutions that will be useful in the sequel.

Proposition 2. *Let $(\mathcal{S}, I, G)$ be a control problem. Then each strong solution is also a weak solution. If $\mathcal{S}$ is deterministic, then each weak solution is also a strong solution.*

4 Discrete Time Linear Hybrid Systems

Discrete Time Linear Hybrid Sytems (DTLHSs) can effectively model linear algebraic constraints involving both continuous as well as discrete variables. Many embedded control systems can be modeled as DTLHSs. As an example, in [15] it is provided a DTLHS model of a buck DC-DC converter, i.e. a mixed-mode analog circuit that converts the DC input voltage to a desired DC output voltage. The dynamics of a DTLHS is given in terms of a suitable LTS.

Definition 4. *A DTLHS $\mathcal{H}$ is a tuple* (X, U, Y, N) *where:*

$X = X^r \cup X^d$ *is a finite sequence of real* (X^r) *and discrete* (X^d) present state *variables. We denote with X' the sequence of* next state *variables obtained by decorating with $'$ all variables in X.*

$U = U^r \cup U^d$ *is a finite sequence of* input *variables, that models* controllable inputs.

$Y = Y^r \cup Y^d$ *is a finite sequence of* auxiliary *variables. Auxiliary variables are typically used to model* modes *(e.g., from switching elements such as diodes) or* uncontrollable inputs *(e.g., disturbances).*

$N(X, U, Y, X')$ *is a predicate over $X \cup U \cup Y \cup X'$ defining the* transition relation *(*next state*) of the system.*

$\mathcal{H}$ is bounded *if N is a bounded predicate. It is* conjunctive *if N is a conjunctive predicate. It is* deterministic *iff $N(x, u, y, x') \wedge N(x, u, \tilde{y}, \tilde{x}')$ implies $x' = \tilde{x}'$.*

Definition 5. *Let $\mathcal{H} = (X, U, Y, N)$ be a DTLHS. The dynamics of $\mathcal{H}$ is defined by the labeled transition system* $\mathrm{LTS}(\mathcal{H}) = (\mathcal{D}_X, \mathcal{D}_U, \bar{N})$ *where: $\bar{N} : \mathcal{D}_X \times \mathcal{D}_U \times \mathcal{D}_X \to \mathbb{B}$ is a function s.t. $\bar{N}(x, u, x') = \exists\, y \in \mathcal{D}_Y\ N(x, u, y, x')$. A* state *$x$ for $\mathcal{H}$ is a state x for* $\mathrm{LTS}(\mathcal{H})$ *and a* path *for $\mathcal{H}$ is a path for* $\mathrm{LTS}(\mathcal{H})$.

4.1 DTLHS Reachability and Control Problem

Definition 6. *Let $\mathcal{H} = (X, U, Y, N)$ be a DTLHS and let I and G be linear predicates over X. The* DTLHS reachability problem $\mathcal{R} = (\mathcal{H}, I, G)$ *is defined as the LTS reachability problem* $(\mathrm{LTS}(\mathcal{H}), I, G)$.

Similarly, the DTLHS control problem $(\mathcal{H}, I, G)$ *is defined as the LTS control problem* $(\mathrm{LTS}(\mathcal{H}), I, G)$.

Example 2. Let T be the positive constant $1/10$ (sampling time). We define the DTLHS $\mathcal{H} = (\{x\}, \{u\}, \varnothing, N)$ where x is a continuous variable, u is a boolean variable, and $N(x, u, x') \equiv [\overline{u} \to x' = x + (5/4 - x)T] \wedge [u \to x' = x + (x - 7/4)T]$. Finally, let $I(x) \equiv -1 \leq x \leq 5/2$ and $G(x) \equiv 0 \leq x \leq 1/2$.

Let us consider the control problem $\mathcal{P} = (\mathcal{H}, I, G)$. A controller may drive the system into the goal G, by enabling a suitable action in such a way that $x' < x$ when $x > 1/2$ and $x' > x$ when $x < 0$. Indeed, the controller: $K(x, u) = (-1 \leq x < 0 \wedge \overline{u}) \vee (0 \leq x < 2 \wedge u) \vee (1 \leq x \leq 5/2 \wedge \overline{u})$ is a weak solution to P. K is not a strong controller, because it allows infinite paths to be executed. For example, K enables the action $u = 0$ in the state $x = 5/4$. Since $N(5/4, 0, 5/4)$ holds, the closed loop system $\mathcal{H}^{(K)}$ may loop forever along the path $5/4, 0, 5/4, 0 \ldots$.

A strong controller K' for $\mathcal{H}$ is $K'(x,u) = (-1 \leq x < 0 \ \wedge \ \overline{u}) \ \vee \ (0 \leq x < {}^3/_2 \ \wedge \ u) \ \vee \ ({}^3/_2 \leq x \leq {}^5/_2 \ \wedge \ \overline{u})$.

4.2 Quantized Control Problem

In order to manage real variables, in classical control theory the concept of *quantization* is introduced (e.g., see [10]). Quantization is the process of approximating a continuous interval by a set of integer values. A quantized feedback control system uses two converters to translate continuous variables into discrete variables (AD converter) and vice versa (DA converter). In the following we formally define a quantized feedback control problem for DTLHSs.

A *quantization function* $\gamma : \mathbb{R} \mapsto \mathbb{Z}$ is a non-decreasing function, such that for any bounded interval $I = [a,b] \subset \mathbb{R}$, $\gamma(I)$ is a bounded integer interval. We will denote $\gamma(I)$ as $\hat{I} = [\gamma(a), \gamma(b)]$. For ease of notation, we extend quantizations to integer intervals, by stipulating that in such a case the quantization function is the identity function.

Definition 7. *Let $\mathcal{H} = (X,U,Y,N)$ be a DTLHS, and let $W = X \cup U \cup Y$. A* quantization *$\mathcal{Q}$ for $\mathcal{H}$ is a pair $(\mathcal{A}, \Gamma)$, where:*

$\mathcal{A}$ is a predicate over W that explicitely bounds each variable in W. For each $w \in W$ $\mathcal{A}_w = \{w^* \mid \exists w_1, \ldots, w_n \mathcal{A}(w_1, \ldots, w^*, \ldots, w_n)\}$ denotes the admissible region of *w, and $\mathcal{A}_W = \prod_{w \in W} \mathcal{A}_w$ denotes the admissible region of Γ.*

Γ is a set of maps $\{\gamma_w \mid w \in W$ and γ_w is a quantization function $\}$.

Let $W = [w_1, \ldots w_k]$ and $v = [v_1, \ldots v_k] \in \mathcal{A}_W$. We write $\Gamma(v)$ for the tuple $[\gamma_{w_1}(v_1), \ldots \gamma_{w_k}(v_k)]$.

A control problem admits a *quantized* solution if control decisions can be made by just looking at quantized values. This enables a software implementation for a controller.

Definition 8. *Let $\mathcal{H} = (X,U,Y,N)$ be a DTLHS, $\mathcal{Q} = (\mathcal{A}, \Gamma)$ be a quantization for $\mathcal{H}$ and $\mathcal{P} = (\mathcal{H}, I, G)$ be a DTLHS control problem. A $\mathcal{Q}$* Quantized Feedback Control *(QFC) strong (resp. weak) solution to $\mathcal{P}$ is a strong (resp. weak) solution $K(x,u)$ to $\mathcal{P}$ such that $K(x,u) = \hat{K}(\Gamma(x), \Gamma(u))$ where $\hat{K} : \Gamma(\mathcal{A}_X) \times \Gamma(\mathcal{A}_U) \to \mathbb{B}$.*

Example 3. Let $\mathcal{P}$, K and K' be as in Example 2. Let us consider the quantizations $\mathcal{Q}_1 = (\mathcal{A}_1, \Gamma_1)$, where $\mathcal{A}_1 = I$, $\Gamma_1 = \{\gamma_x\}$ and $\gamma_x(x) = \lfloor x \rfloor$. The set $\Gamma_1(\mathcal{A}_x)$ of quantized states is the integer interval $[-1, 2]$. Let $\hat{K}(s,a) = (a = 0 \ \wedge \ s \neq 0) \vee (a = 1 \ \wedge \ s \in \{0,1\})$. The controller $K''(x,u) = \hat{K}(\Gamma_1(x), \Gamma_1(u))$ is exactly K, and therefore it is a QFC weak solution to $\mathcal{P}$.

No $\mathcal{Q}$ QFC strong solution can exist, because in state 1 either enabling action 1 or action 0 allows infinite loops to be potentially executed in the closed loop system.

The strong controller K' in Example 2 can be obtained as a quantized controller decreasing the quantization step, for example, by considering the quantization $\mathcal{Q}_2 = (\mathcal{A}_2, \Gamma_2)$, where $\mathcal{A}_2 = \mathcal{A}_1$, $\Gamma_2 = \{\tilde{\gamma}_x\}$ and $\tilde{\gamma}_x(x) = \lfloor 2x \rfloor$.

5 Quantized Feedback Control Problem Undecidability

In this section we prove the undecidability of the DTLHS quantized feedback control problem. Along the same lines of similar undecidability proofs [13, 12], we first show that a two-counter machine M can be encoded as a deterministic DTLHS $\mathcal{H}_M$ without controllable inputs in such a way that M halts if and only if $\mathcal{H}_M$ reaches a goal region. This immediately implies that DTLHS reachability is undecidable. Since $\mathcal{H}_M$ has no controllable inputs, existence of a weak controller is equivalent to a reachability problem. For the same reason, actions enabled by any controller for $\mathcal{H}_M$ do not depend on state variables. As a consequence, a quantized weak control problem is equivalent to a DTLHS control problem. Finally, by Proposition 2, weak solutions to deterministic LTS control problems are also strong solutions. Therefore, since $\mathcal{H}_M$ is deterministic, the quantized strong control problem for DTLHS is undecidable, too.

Two-Counter Machines. A *two-counter machine* [16] M consists of two counters that store unbounded natural numbers and a finite control that is a finite sequence of statements $\langle 1 : \mathit{stmt}_1, \ldots, n : \mathit{stmt}_n\rangle$, where $\mathit{stmt} ::= \mathsf{inc}\ i\ k \mid \mathsf{dec}\ i\ k \mid \mathsf{beq}\ i\ k \mid \mathsf{halt}$, with $i \in \{0, 1\}$. Computations start from the statement labeled 1. The execution of $j : \mathsf{inc}\ i\ k$ increments the counter i and then jumps to the statement labeled k. Similarly, the execution of $j : \mathsf{dec}\ i\ k$ decrements the counter i (leaving it unchanged if it is 0) and then jumps to the statement labeled k. If the counter i is 0, the execution of $j : \mathsf{beq}\ i\ k$ causes a jump to the statement labeled k. Otherwise, the statement labeled $j + 1$ will be executed. Finally, the execution stops if a halt statement is executed. The halting problem for two-counter machine is undecidable [16].

Lemma 1. *For any two-counter machine M, there exists a* bounded, conjunctive, *and* deterministic *DTLHS $\mathcal{H}_M$, and two predicates I and G such that M halts if and only if G is reachable from I in $\mathcal{H}_M$.*

Proof. Let M be a two-counter machine and let $\mathcal{H}_M$ be the DTLHS (X, U, Y, N), where $X^r = \{x_0, x_1\}$, $X^d = \{l, g\}$, and $U = Y = \varnothing$. Since we are dealing with bounded DTLHSs, we use two real variables x_0 and x_1 to encode values stored in counters. Each natural number m is encoded by the rational number $1/2^m$. Variables x_i are both bounded by the predicate $0 \leq x_i \leq 1$. A discrete variable l stores the label of the statement currently under execution and it is bounded by $0 \leq l \leq n$, where n is the number of statements in the finite control of M. Finally, the boolean variable g encodes termination of the computation of M. The transition relation N encodes the execution of the control program. Let $U(X)$ be the predicate $\bigwedge_{x \in X} x' = x$. A program $\langle 1 : \mathit{stmt}_1, \ldots, n : \mathit{stmt}_n\rangle$ is encoded by the predicate $N = \bigwedge_{j=1}^{n} [\![j : \mathit{stmt}_j]\!]$, where:

$$\begin{aligned}
[\![j : \mathsf{dec}\ i\ k]\!] &\equiv (l \neq j) \lor (((x_i = 1) \lor (x'_i = 2x_i)) \land \\
&\quad \land ((x_i \neq 1) \lor (x'_i = 1)) \ \land\ (l' = k) \ \land\ U(x_{1-i}, g)) \\
[\![j : \mathsf{inc}\ i\ k]\!] &\equiv (l \neq j) \lor ((x'_i = x_i/2) \ \land\ (l' = k) \ \land\ U(x_{1-i}, g)) \\
[\![j : \mathsf{beq}\ i\ k]\!] &\equiv (l \neq j) \lor (((x_i \neq 1) \lor (l' = k)) \land \\
&\quad \land ((x_i = 1) \lor (l' = l + 1)) \land U(x_{1-i}, g)) \\
[\![j : \mathsf{halt}]\!] &\equiv (l \neq j) \lor ((l' = j) \ \land\ (g' = 1) \land U(x_0, x_1))
\end{aligned}$$

We observe that we use negation as syntactic sugar to improve readability. Indeed, since x_i can assume only values of the form $1/2^m$ for some $m \in \mathbb{N}$, the condition $x_i \neq 1$ can be replaced by the constraint $x_i \leq 1/2$. Moreover, since l is a discrete variable, the condition $l \neq j$ can be replaced by the predicate $(l \leq j-1) \vee (l \geq j+1)$.

It is possible to check that $N(\{l, 1/2^m, 1/2^p, g\}, \epsilon, \{l', 1/2^{m'}, 1/2^{p'}, g'\})$ if and only if after executing the statement labeled l with m and p as counter values, M will execute the statement labeled l' with m' and p' as counter values. Moreover if $g = 0$, g' will be 1 if and only if the statement labeled l is a halt statement.

Let I be the predicate $l = 1 \wedge g = 0$ and G be the predicate $g = 1$. G is reachable from I in $\mathcal{H}_M$ if and only if the computation of M terminates.

Finally, we show that N can be written as a conjunctive predicate. Any predicate $P(X)$ can be written as an equivalent DNF $\bigvee_{i=1}^{q} \bigwedge_{j=1}^{m_i} C_{ij}(X)$, where $C_{ij}(X)$ are constraints. By introducing q fresh boolean auxiliary variables z_1, $\ldots, z_q$ this is equisatisfiable to $\bigwedge_{i=1}^{q}(z_i \rightarrow \bigwedge_{j=1}^{m_i} C_{ij}(X)) \wedge \sum_{i=1}^{q} z_i \geq 1$, which in turn is equivalent to $\bigwedge_{i=1}^{q} \bigwedge_{j=1}^{m_i}(z_i \rightarrow C_{ij}(X)) \wedge \sum_{i=1}^{q} z_i \geq 1$. Since N is bounded, by Proposition 1 this can be transformed into a conjunctive predicate.

For example we have:

$$\begin{aligned}\llbracket j : \mathsf{halt} \rrbracket \equiv\ & (z_{j,1} \rightarrow (l \geq j+1)) \wedge (z_{j,2} \rightarrow (l \leq j-1)) \wedge (z_{j,3} \rightarrow (l' = j)) \wedge \\ & \wedge (z_{j,3} \rightarrow (g' = 1)) \wedge (z_{j,3} \rightarrow (x'_0 = x_0)) \wedge (z_{j,3} \rightarrow (x'_1 = x_1)) \wedge \textstyle\sum_{i=1}^{3} z_{j,i} \geq 1\end{aligned}$$

An immediate consequence of Lemma 1 is the undecidability of the DTLHS reachability problem.

Theorem 1. *The reachability problem for bounded and conjunctive DTLHSs is undecidable.*

Theorem 2. *Existence of strong and weak solutions to a control problem for a bounded and conjunctive DTLHS is undecidable.*

Proof. For any two-counter machine M, the DTLHS $\mathcal{H}_M$ has no controllable actions. Let K be the controller that enables all actions, i.e. such that $\forall x \in \mathcal{D}_X$ $K(x, \epsilon)$ holds. K is a weak solution to the control problem $(\mathcal{H}_M, I, G)$ if and only if G is reachable from I (observe that states in G are controlled by K). Moreover, since the transition relation of $\mathcal{H}_M$ is deterministic, by Proposition 2, K is a weak solution to $(\mathcal{H}_M, I, G)$ if and only if it is a strong solution.

Theorem 3. *Existence of QFC strong and weak solutions to a DTLHS control problem is undecidable.*

Proof. The controller K considered in the proof of Theorem 2 is a quantized controller. Indeed, for any quantization $\mathcal{Q} = (A, \Gamma)$, let $\hat{K}$ be defined by $\forall s \in \Gamma(A_X)$ $\hat{K}(s, \epsilon)$. We have that $K(x, \epsilon) = \hat{K}(\Gamma(x), \epsilon)$.

6 Abstraction Based Control Synthesis

A typical approach to the automatic synthesis of controllers consists of building a suitable finite state abstraction $\hat{\mathcal{H}}$ of a hybrid system $\mathcal{H}$, computing an abstraction $\hat{I}$ (resp. $\hat{G}$) of the initial (resp. goal) region I (resp. G) so that any solution to the LTS control problem $(\hat{\mathcal{H}}, \hat{I}, \hat{G})$ is a finite representation of a solution to $(\mathcal{H}, I, G)$. For example, this can be done by giving conditions ensuring that the abstract system satisfies some equivalence relation with respect to the concrete system (e.g. see [17] or [1]).

In our approach, the abstraction induced by a quantization is a design constraint rather than a methodological tool, since it depends on the number of bits used by AD/DA conversions. In [15], we give a constructive sufficient condition ensuring that the controller computed for $\hat{\mathcal{H}}$ is indeed a quantized controller for $\mathcal{H}$. Such a condition stems from the notion of *control abstraction*. Control abstractions form a family of abstractions induced by a given quantization.

In this section, we show that finding the "best" control abstraction (in order to maximize the possibilities of finding a solution to the original control problem) is also undecidable.

We start by briefly summarizing some definitions and results of [15].

Definition 9. *Let $\mathcal{H} = (X, U, Y, N)$ be a DTLHS and $\mathcal{Q} = (\mathcal{A}, \Gamma)$ be a quantization for $\mathcal{H}$.*

An action $u \in \mathcal{A}_U$ is $\mathcal{A}$-admissible *in $x \in \mathcal{A}_X$ if for all x', $(\exists y \in \mathcal{A}_Y : N(x, u, y, x'))$ implies $x' \in \mathcal{A}_X$.*

An action $a \in \Gamma(\mathcal{A}_U)$ is $\mathcal{Q}$-admissible *in $s \in \Gamma(\mathcal{A}_X)$ if for all $x \in \Gamma^{-1}(s)$, $u \in \Gamma^{-1}(a)$, u is $\mathcal{A}$-admissible for x in $\mathcal{H}$.*

The $\mathcal{Q}$-abstraction *of $\mathcal{H}$ is the LTS $\hat{\mathcal{H}} = (S, A, T)$ such that $\Gamma(\mathcal{A}_X) = S$, $\Gamma(\mathcal{A}_U) = A$, and for all $s, s' \in S$, $a \in A$ we have $T(s, a, s')$ iff there exists $x \in \Gamma^{-1}(s)$, $x' \in \Gamma^{-1}(s')$, $u \in \Gamma^{-1}(a)$, $y \in \mathcal{D}_y$ such that $N(x, u, y, x')$ and a is $\mathcal{Q}$-admissible in s.*

The $\mathcal{Q}$ abstraction could be a highly non-deterministic LTS, thus making problematic the existence of a strong solution to the (abstract) control problem. In particular, for small values of the sampling time, the $\mathcal{Q}$-abstraction may contain a large number of self-loops.

Example 4. Let $\mathcal{H}$ be the DTLHS of Example 2, and let $\mathcal{Q}_1 = (\mathcal{A}_1, \Gamma_1)$ and $\mathcal{Q}_2 = (\mathcal{A}_2, \Gamma_2)$ be quantizations in Example 3. Then, the $\mathcal{Q}_1$-abstraction of $\mathcal{H}$ is the LTS $\mathcal{S}_1'$, obtained from the LTS $\mathcal{S}_1$ in Example 1, by adding all dotted self-loops in Fig. 1. The $\mathcal{Q}_2$-abstraction of $\mathcal{H}$ is the LTS $\mathcal{S}_2'$, obtained from the LTS $\mathcal{S}_2$ in Example 1, by adding all dotted self-loops in Fig. 2.

Let I_1, I_2, and G as in Example 1. Because of self-loop nondeterminism, no strong solution exists for control problems $(\mathcal{S}_1', I_1, G)$ and $(\mathcal{S}_2', I_2, G)$.

On the other hand, if by repeatedly performing an action a in an abstract state s, it is guaranteed that the system will leave the region represented by s after a finite number of steps, a self–loop $T(s, a, s)$ can be eliminated and the action a can be enabled by a strong controller in state s.

Definition 10. *Let $\mathcal{H} = (X, U, Y, N)$ be a DTLHS, and let $\hat{\mathcal{H}} = (S, A, T)$ be its $\mathcal{Q}$-abstraction.*

A self–loop $T(s, a, s)$ is non-eliminable *if there exists at least an infinite run $\pi = x_0 u_0 x_1 u_1 x_2 \ldots$ in $\mathcal{H}$ such that $\forall t \in \mathbb{N}$ $x_t \in \Gamma^{-1}(\hat{s})$ and $a_t \in \Gamma^{-1}(\hat{a})$.*

Otherwise, a self-loop $T(s, a, s)$ not satisfying the above property is said to be an eliminable self loop*.*

Definition 11. *Given the $\mathcal{Q}$-abstraction $\hat{\mathcal{H}}$ of $\mathcal{H}$, we call $\mathcal{Q}$-*control abstraction *any refinement $\mathcal{C} \sqsubseteq \hat{\mathcal{H}}$ that omits some eliminable self–loops.*

The following theorem [15] states that it is correct to consider control abstractions when looking for a QFC strong solution to a DTLHS control problem.

Theorem 4. *Let $\mathcal{H} = (X, U, Y, N)$ be a DTLHS, $\mathcal{Q} = (\mathcal{A}, \Gamma)$ be a quantizantion and let the LTS $\hat{\mathcal{H}}$ be a $\mathcal{Q}$ control abstraction of $\mathcal{H}$. If $I \subseteq \Gamma^{-1}(\hat{I})$ and $G \supseteq \Gamma^{-1}(\hat{G})$, then a strong solution $\hat{K}$ to the control problem $(\hat{\mathcal{H}}, \hat{I}, \hat{G})$ is a quantized solution to $(\mathcal{H}, I, G)$.*

Since self–loop nondeterminism is an obstruction in finding a strong solution to an LTS control problem, and the set of control abstractions is a finite lattice with respect to the refinement relation $\sqsubseteq$, it would be convenient considering the *minimum control abstraction* when looking for a quantized strong solution to a DTLHS control problem.

Theorem 5. *Finding the minimum control abstraction is undecidable.*
Proof. We will show that it is undecidable to state if a self–loop is non-eliminable.

Let M be a two-counter machine. We encode M in a DTLHS $\mathcal{H}_M = (X, U, Y, N)$, where $X^r = \{x_0, x_1, l\}$, $X^d = \{g\}$, and $U = Y = \varnothing$. $N = (\bigvee_{j=1}^{n} l = j) \wedge (\bigvee_{j=1}^{n} l' = j) \wedge \bigwedge_{j=1}^{n} [\![j : stmt_j]\!]$, where $[\![j : stmt_j]\!]$ is defined as in the proof of Lemma 1.

Let $\mathcal{Q} = (A, \Gamma)$ be the quantization defined as follows: $A_{x_0} = A_{x_1} = [0, 1]$, $A_l = [1, n]$, $A_g = \mathbb{B} = \{0, 1\}$, $A_U = \{0\}$, $\gamma_{x_0}(x) = \gamma_{x_1}(x) = \gamma_l(x) = 1$. Note that we have only two abstract states: $\langle \hat{x}_0, \hat{x}_1, \hat{l}, g\rangle = \langle 1, 1, 1, 0\rangle$ and $\langle \hat{x}_0, \hat{x}_1, \hat{l}, g\rangle = \langle 1, 1, 1, 1\rangle$. Then, the self–loop $(\langle 1, 1, 1, 0\rangle, 0, \langle 1, 1, 1, 0\rangle)$ is non-eliminable iff there exists an infinite run on M. Being the latter an undecidable problem, we cannot decide if a self–loop is eliminable or non-eliminable.

Example 5. Let us consider again the DTLHS $\mathcal{H}$, and the quantizations $\mathcal{Q}_1$ and $\mathcal{Q}_2$ in Example 4. The LTS $\mathcal{S}_1$ (resp. $\mathcal{S}_2$) in Example 1 is the minimal $\mathcal{Q}_1$ (resp. $\mathcal{Q}_2$) control abstractions of $\mathcal{H}$, where all eliminable self-loops have been eliminated.

Self loops $T(1, 0, 1)$ and $T(1, 1, 1)$ in $\mathcal{S}_1$ are not eliminable because of the infinite paths $5/4, 0, 5/4, 0, 5/4 \ldots$ and $7/4, 1, 7/4, 1, 7/4 \ldots$. The same concrete paths make abstract self-loops $T(2, 0, 2)$ and $T(3, 1, 3)$ not eliminable in $\mathcal{S}_2$.

7 Dense Time Rectangular Hybrid Automata as DTLHSs

In this section we show that DTLHSs are expressive enough to faithfully encode a relevant class of dense time hybrid systems, namely *Rectangular Hybrid Automata* (RHA) [13], a proper superclass of Timed Automata [3]. More precisely, we show that for every RHA $\mathcal{A}$ there exists a DTLHS $\mathcal{H}_{\mathcal{A}}$ that has the same dynamics, i.e. such that $LTS(\mathcal{H}_{\mathcal{A}}) \simeq LTS(\mathcal{A})$ (Theorem 6). As a byproduct of this encoding, we obtain alternative proofs of Theorems 1 and 2.

Rectangular Hybrid Automata. We define RHA following the presentation in [13]. Given a positive $n > 0$, a subset of $\mathbb{R}^n$ is called a *region*. A closed and bounded region is called a *compact*. A region $R \subseteq \mathbb{R}^n$ is *rectangular* if it is a cartesian product of (possibly unbounded) intervals (finite endpoints are rationals). We write R_i for the projection of R on the i-th coordinate, so that $R = \prod_{i\in[n]} R_i$. The set of rectangular regions in $\mathbb{R}^n$ is denoted by $\mathcal{R}^n$.

An n-dimensional RHA $\mathcal{A}$ consists of a finite directed multigraph (V, E), a finite *observation alphabet* Σ, three vertex labeling functions $init : V \to \mathcal{R}^n$, $inv : V \to \mathcal{R}^n$, and $flow : V \to \mathcal{R}^n$, and four edge labeling functions $pre : E \to \mathcal{R}^n$, $post : E \to \mathcal{R}^n$, $jump : E \to \mathcal{P}([n])$, and $obs : E \to \Sigma$. The set V of vertices is the set of *control modes*, and the set E of edges is the set of *control switches*.

A variable x_i is *bounded* if for every control mode v, the region $inv(v)_i$ is a bounded interval. A variable x_i is *monotone* if for every control mode v, either $flow(v)_i \subset \mathbb{R}_{<0}$ or $flow(v)_i \subset \mathbb{R}_{>0}$. A variable x_i is *closed* if for every control mode and every control switch e, the intervals $inv(v)_i$, $flow(v)_i$, $init(v)_i$, $pre(e)_i$, and $post(e)_i$ are closed intervals. A rectangular automata is bounded (resp. monotone, closed) if all its variables are bounded (resp. monotone, closed).

The rectangular automaton $\mathcal{A}$ defines a labeled transition system $LTS(\mathcal{A}) = (S, A, T)$, where:

States: The set of states S is $V \times \mathbb{R}^n$. Each subset $Z \subseteq S$ is called a *zone* of $\mathcal{A}$. A state (v, x) is an *initial state* of $\mathcal{A}$ if $x \in init(v)$. The *initial zone* of $\mathcal{A}$, denoted by $Init(\mathcal{A})$, is the set of all initial states of $\mathcal{A}$.

Actions: The set of actions A is $\Sigma \cup \mathbb{R}^+$. Each transition labeled with $a \in \Sigma$ corresponds to a jump step, whose observation is a. Each transition labeled with $t \in \mathbb{R}^+$ corresponds to a flow step, whose duration is $t \geq 0$.

Transition Relation: The transition relation T is defined by *jump* and *flow* transitions as follows. For each edge $e = (v, w)$ of $\mathcal{A}$, $T^e((v, x), a, (w, y))$ holds iff $x \in pre(e)$, $y \in post(e)$, for every $i \notin jump(e)$ we have $x_i = y_i$, and $a = obs(e)$. For all $t \in \mathbb{R}^+$, $T^{flow}((v, x), t, (v, y))$ holds iff either $t = 0$ and $x = y$ or $t > 0$ and $(y - x)/t \in flow(v)$. Finally, the transition relation T of $\mathcal{A}$ is $\bigcup_{e\in E} T^e \cup T^{flow}$.

We observe that, thanks to convexity of rectangular regions, we have that a flow transition $T^{flow}((v, x), t, (v, y))$ can be performed if and only if there exists a smooth function $f : [0, t] \to inv(v)$ with first derivative f' such that $f(0) = x$, $f(t) = y$, and for all $s \in (0, t)$ $f'(s) \in flow(v)$. In the following, for the sake of readability, we consider the case $\Sigma = E$ and $obs(e) = e$.

Theorem 6. *For any closed RHA $\mathcal{A}$ there exists a DTLHS $\mathcal{H}_{\mathcal{A}}$ such that $LTS(\mathcal{A}) \simeq LTS(\mathcal{H}_{\mathcal{A}})$.*

Proof. Let $\mathcal{A}$ be a closed n dimensional RHA. First, we define a DTLHS $\mathcal{H}_{\mathcal{A}}$ that encodes $\mathcal{A}$. Let V be the set of m vertices, and E be the set of l edges of $\mathcal{A}$. Let $|\cdot|_V : V \to [m]$ and $|\cdot|_E : E \to [l]$ be two encoding functions of the set of vertices and the set of edges into initial segments of natural numbers. Since both vertex and edge labeling functions define rectangular regions, they can be easily represented as conjunctive predicates. Let $inv(v) = \prod_{i\in[n]}[\underline{\alpha}_{v,i}, \overline{\alpha}_{v,i}]$, $init(v) = \prod_{i\in[n]}[\underline{\beta}_{v,i}, \overline{\beta}_{v,i}]$, $pre(e) = \prod_{i\in[n]}[\underline{\beta}_{v,i}, \overline{\beta}_{v,i}]$, and $post(e) = \prod_{i\in[n]}[\underline{\alpha}_{e,i}, \overline{\alpha}_{e,i}]$. We define the following predicates:

$$\begin{array}{ll} \mathsf{inv}_v(x) \equiv \bigwedge_{i\in[n]} \underline{\alpha}_{v,i} \le x_i \le \overline{\alpha}_{v,i} & \mathsf{init}_v(x) \equiv \bigwedge_{i\in[n]} \underline{\beta}_{v,i} \le x_i \le \overline{\beta}_{v,i} \\ \mathsf{pre}_e(x) \equiv \bigwedge_{i\in[n]} \underline{\alpha}_{e,i} \le x_i \le \overline{\alpha}_{e,i} & \mathsf{post}_e(x) \equiv \bigwedge_{i\in[n]} \underline{\beta}_{e,i} \le x'_i \le \overline{\beta}_{e,i} \end{array}$$

The DTLHS $\mathcal{H}_{\mathcal{A}} = (X, U, Y, N)$ is defined as follows:

State Variables: The set of present state variables is $X{=}X^r \cup X^d$, where $X^r = \{x_1, \ldots, x_n\}$ and $X^d = \{q\}$. Each x_i encodes one continuous variable of $\mathcal{A}$, and q encodes the set of vertices V of $\mathcal{A}$. Continuous variables are bounded by inv_v (see the definition of N below), and q ranges over $[m]$.

Input Variables: The set of input variable is $U{=}U^r \cup U^d$, where $U^r = \{t\}$ and $U^d = \{r\}$. The variable $t \ge 0$ encodes flow transition durations. The variable $r \in \{0, \ldots, l\}$ encodes the edge taken in a jump transition. The variable r assumes the value 0 when a flow transition is taken.

Transition Relation: The transition relation predicate N is defined as follows. Let $flow(v) = \prod_{i\in[n]}[\underline{\gamma}_{v,i}, \overline{\gamma}_{e,i}]$. For each vertex $v \in V$, we define the predicate flow_v as follows:

$$\mathsf{flow}_v(q, x, t, q', x') \equiv \mathsf{inv}_v(x') \ \wedge \ q' = q \ \wedge \ \bigwedge_{i\in[n]} x_i + \underline{\gamma}_{v,i}t \le x'_i \le x_i + \overline{\gamma}_{v,i}t$$

For each edge $e = (v, w) \in E$, we define the predicate jump_e as follows:

$$\mathsf{jump}_e(q, x, q', x') \equiv q = |v|_V \ \wedge \ q' = |w|_V \wedge \ \mathsf{pre}_e(x) \ \wedge \ \mathsf{post}_e(x') \\ \wedge \ \bigwedge_{i\notin jump(e)} x_i = x'_i$$

Finally, we define the transition relation N as follows:

$$N(x, q, t, x', q') \equiv ((r \ne 0) \ \vee \ (\bigwedge_{v\in V}(q \ne |v|_V) \vee \mathsf{flow}_v(q, x, t, q', x'))) \\ \wedge \ \bigwedge_{e\in E}((r \ne |e|_E) \ \vee \ \mathsf{jump}_e(q, x, q', x'))$$

Now we show that $LTS(\mathcal{A}) = (S, A, T)$ is isomorphic to $LTS(\mathcal{H}_{\mathcal{A}}) = (\mathcal{D}_X, \mathcal{D}_U, N)$. Let us consider the map $f_S : V \times \mathbb{R}^n \to [m] \times \mathbb{R}^n$ defined by $f_S(v, x) = (|v|_V, x)$ and the map $f_A : \Sigma \cup \mathbb{R} \to \{0, \ldots, l\} \times \mathbb{R}$ defined by $f_A(a) = (0, a)$ if $a \in \mathbb{R}$ and $f_A(a) = (|a|_E, 0)$ if $a \in E$. We have:

Flow Transitions: For all $v \in V$, $t \ge 0$ we have $T^{flow}((v, x), t, (w, y))$ if and only if $v = w$ and either $t = 0$ and $x = y$ or $t > 0$ and $(y - x)/t \in flow(v)$, i.e. for all $i \in [n]$ $(y_i - x_i)/t \in flow(v)_i$. In turn, this is equivalent to $\mathsf{flow}_v(|v|_V, x, t, |w|_V, y)$ (observe that $t = 0$ implies $x = x'$), and hence if and only if $N((|v|_V, x), (0, t), (|w|_V, y))$.

Jump Transitions: For all $e = (v, w) \in E$ we have $T^e((v, x), t, (v, y))$ if and only if $x \in pre(e)$, $y \in post(e)$, and for all $i \in [n]$ such that $i \notin jump(e)$, $x_i = y_i$. Again, this is equivalent to $\mathsf{jump}_\mathsf{e}(|v|_V, x, |w|_V, y)$ and hence if and only if $N((|v|_V, x), (|e|_E, 0), (|w|_V, y))$.

Corollary 1. *Let $\mathcal{A}$ be a closed RHA and let $\mathcal{H}_{\mathcal{A}}$ be the DTLHS that encodes $\mathcal{A}$. If $\mathcal{A}$ is bounded, then $\mathcal{H}_{\mathcal{A}}$ is bounded and conjunctive.*

Proof. If the RHA $\mathcal{A}$ is monotone and bounded, then the DTLHS $\mathcal{H}_{\mathcal{A}}$ is bounded. Each continuous variable x_i is bounded by $\bigwedge_{v \in V} \mathsf{inv}_v(x_i)$. If $inv(v)_i$ is bounded, then $\mathsf{inv}_v(x_i)$ is bounded.

If $\mathcal{A}$ is monotone and bounded, for every mode v there is an upper bound T to flow transition durations. Therefore, the predicate N implies the constraint $0 \leq t \leq T$.

If $\mathcal{A}$ is bounded but not monotone, a bit more involved definition of $\mathcal{H}_{\mathcal{A}}$ is required. Without going into details, in such a case the definition of $\mathcal{H}_{\mathcal{A}}$ stems from the fact that a flow transition $T^{flow}((v, x), t, (v, y))$ is equivalent to a sequence of flow transitions $T^{flow}((v, x_1), t_1, (v, x_2))$, $T^{flow}((v, x_2), t_2, (v, x_3))$, $\ldots$, $T^{flow}((v, x_n), t_n, (v, x_{n+1}))$, with $x_1 = x$, $x_{n+1} = y$, and $\sum_{i=1}^{n} t_i = t$.

If $\mathcal{H}_{\mathcal{A}}$ is bounded, then N can be transformed into a conjunctive predicate as discussed in the proof of Lemma 1.

Undecidability Results Revisited.

The reachability problem for RHAs is a pair $(\mathcal{A}, Z)$ where $\mathcal{A}$ is an RHA and Z is a zone of $\mathcal{A}$ and it is defined as the LTS reachability problem $(LTS(\mathcal{A}), init(\mathcal{A}), Z)$.

The reachability problem is undecidable for a restricted class of RHA, namely *Simple Rectangular Automata* (SRA) [13]. Since SRA are bounded rectangular automata, Corollary 1 gives immediately an alternative proof of Theorem 1.

Given an SRA $\mathcal{S}$, the DTLHS $\mathcal{H}_{\mathcal{S}}$ obtained by applying the encoding in the proof of Theorem 6 has a unique initial state and it is deterministic. In such a case, finding weak and strong solutions can be easily reduced to a reachability problem, thus obtaining an alternative proof of Theorem 2. On the other hand, a proof for Theorem 3 does not follow immediately.

8 Conclusions

We have shown that, for DTLHSs, existence of a quantized sampling controller meeting given (safety and liveness) system level specifications is undecidable. The relevance of such a problem stems from the fact that the *control software* implementing the controller in a software based control system always yields a quantized sampling controller.

Furthermore, we have shown that *Rectangular Automata* (RA), and thus *Timed Automata* (TA), can be modelled as DTLHSs. This shows how, by exploiting availability of real valued variables, dense time behaviors can be modelled using discrete time behaviors.

Investigating interesting classes of (discrete time) hybrid systems for which quantized sampling control is decidable appears to be an interesting future work.

Acknowledgments

We are grateful to our anonymous referees for their helpful comments. Our work has been partially supported by: MIUR project DM24283 (TRAMP) and by the EC FP7 project GA218815 (ULISSE).

References

1. Agrawal, M., Thiagarajan, P.S.: The discrete time behavior of lazy linear Hybrid Automata. In: HSCC. pp. 55–69. LNCS 3414 (2005)
2. Alur, R., Courcoubetis, C., Halbwachs, N., Henzinger, T.A., Ho, P.H., Nicollin, X., Olivero, A., Sifakis, J., Yovine, S.: The algorithmic analysis of Hybrid Systems. Theoretical Computer Science 138(1), 3 – 34 (1995)
3. Alur, R.: Timed Automata. In: CAV. pp. 8–22. LNCS 1633 (1999)
4. Asarin, E., Bouajjani, A.: Perturbed Turing Machines and Hybrid Systems. In: LICS. pp. 269–278 (2001)
5. Bemporad, A., Morari, M.: Verification of Hybrid Systems via mathematical programming. In: HSCC. pp. 31–45. LNCS 1569 (1999)
6. Brogan, W.L.: Modern Control Theory (3rd ed.). Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1991)
7. Cassez, F., Henzinger, T.A., Raskin, J.F.: A comparison of control problems for Timed and Hybrid Systems. In: HSCC. pp. 134–148 (2002)
8. Cimatti, A., Roveri, M., Traverso, P.: Strong planning in non-deterministic domains via Model Checking. In: AIPS. pp. 36–43 (1998)
9. Frehse, G.: Phaver: algorithmic verification of Hybrid Systems past HYTECH. Int. J. Softw. Tools Technol. Transf. 10(3), 263–279 (2008)
10. Fu, M., Xie, L.: The sector bound approach to quantized feedback control. IEEE Trans. on Automatic Control 50(11), 1698–1711 (2005)
11. Henzinger, T., Ho, P.H., Wong-Toi, H.: Hytech: A model checker for Hybrid Systems. STTT 1(1), 110–122 (1997)
12. Henzinger, T.A., Kopke, P.W.: Discrete-time control for rectangular Hybrid Automata. In: ICALP. pp. 582–593 (1997)
13. Henzinger, T.A., Kopke, P.W., Puri, A., Varaiya, P.: What's decidable about Hybrid Automata? J. of Computer and System Sciences 57(1), 94–124 (1998)
14. Larsen, K.G., Pettersson, P., Yi, W.: Uppaal: Status & developments. In: CAV. pp. 456–459. LNCS 1254 (1997)
15. Mari, F., Melatti, I., Salvo, I., Tronci, E.: Synthesis of quantized feedback control software for Discrete Time Linear Hybrid Systems. In: CAV. pp. 180–195. LNCS 6174 (2010)
16. Minsky, M.L.: Recursive unsolvability of Post's problem of "tag" and other topics in theory of Turing Machines. The Annals of Mathematics 74(3), pp. 437–455 (1961)
17. Pola, G., Girard, A., Tabuada, P.: Approximately bisimilar symbolic models for nonlinear control systems. Automatica 44(10), 2508–2516 (2008)
18. Tronci, E.: Automatic synthesis of controllers from formal specifications. In: ICFEM. pp. 134–143. IEEE (1998)
19. Vidal, R., Schaffert, S., Shakernia, O., Lygeros, J., Sastry, S.: Decidable and semidecidable controller synthesis for classes of Discrete Time Hybrid Systems. In: CDC. pp. 1243–1248. IEEE Computer Society (2001)