Formal Controller Synthesis from Specifications Given by Discrete-Time Hybrid Automata

Vladimir Sinyakov, Antoine Girard

Formal Controller Synthesis from Specifications Given by Discrete-Time Hybrid Automata ⋆

Vladimir Sinyakov [a], Antoine Girard [b],

[a] *Bundeswehr University Munich, Institute of Control Engineering, 85577 Neubiberg/Munich, Germany*

[b] *Université Paris-Saclay, CNRS, CentraleSupélec, Laboratoire des Signaux et Systèmes, 91190, Gif-sur-Yvette, France*

Abstract

This paper deals with formal controller synthesis for discrete-time dynamical systems. We consider a specification provided under the form of a discrete-time hybrid automaton with external inputs, which can represent, for instance, instructions or informations received from a human user or from another system. The hybrid automaton describes the intended behavior of the system and we first consider the problem of synthesizing a controller such that the maximal trajectories of the closed-loop system are also maximal trajectories of the hybrid automaton. We show that the existence of an alternating simulation relation from the specification to the open-loop system is a necessary and sufficient condition for the existence of such controllers. To be able to solve this problem using symbolic (i.e. finite-state) abstractions, we provide a method to compute a symbolic specification that under-approximates the behavior of the hybrid automata. Then, we extend our approach to consider additional safety or reachability requirements so that some unsafe (e.g. blocking) states are avoided or some target states are reached, respectively. The originality of the problem is that these additional requirements are not formulated over the states of the system but over the states of the specification. Finally, we demonstrate the effectiveness of our approach with two illustrative examples from autonomous vehicle control.

Key words: Formal controller synthesis; alternating simulation relation; symbolic abstractions; hybrid automata

1 Introduction

Recent years saw a burst in research on formal methods in control theory (see e.g. [21,4] and the references therein). One of the key problems in this field is that of synthesizing automatically a controller for a dynamical system so that the closed-loop system has a certain desired behavior, formally described by a specification. The considered specifications can be complex and usually go beyond traditional stability properties. For example, they can be given by regular languages [16,8] or by linear temporal logic formulas [22,6,10,4]. In some other cases, the specification itself can be given under the form of a dynamical system: a trajectory is then accepted if it can be related in some sense to a trajectory of the specification system: see e.g. [21,15] where specifications are given by finite-state dynamical systems or [23,24,13] where both the system and the specification are given by linear systems. When a dynamical system has an infinite number of states, a common approach in formal methods is to transition from the original infinite system to an approximating finite-state system called symbolic abstraction (see e.g. [7,17,25,5,9,18]). To justify rigorously this transition, a formal relation must be established between the behaviors of these two systems. Different kinds of relations, such as alternating simulation relations [2,21] or feedback refinement relations [18], were introduced to formalize whether a controller for the system can be obtained from that of its symbolic abstraction.

In this paper, we consider a controller synthesis problem for discrete-time control systems where the specification is provided under the form of a discrete-time hybrid automaton [20], which is a discrete-time dynamical system with the state having discrete and continuous components, similarly to their continuous-time counterparts [1,12]. In our setting, the specification hybrid automaton may have external inputs that can represent, for example, an instruction issued by a human user or a

⋆ This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 725144).

Email addresses: vladimir.siniakov@gmail.com (Vladimir Sinyakov), Antoine.Girard@l2s.centralesupelec.fr (Antoine Girard).

message communicated by some other system. We consider the problem of synthesizing a controller such that the maximal trajectories of the closed-loop system are also maximal trajectories of the hybrid automaton. The first contribution of the paper is to show that for the considered class of controllers, the existence of an alternating simulation relation from the specification to the open-loop system is not only a sufficient but also a necessary condition for the existence of such a controller. Computing an alternating simulation relation may be difficult for a given system and specification. To address this difficulty, we present an approach based on the use of symbolic abstractions. Since both the system and the specification are given by infinite-state dynamical systems, our solution includes the computation of symbolic counterparts for both of them. While the construction of the symbolic abstraction for the system ressembles existing approaches in the literature [5,18], the proposed construction of a symbolic specification from a hybrid automaton is, to the best of our knowledge, new. Note that in this case, the direction of the alternating simulation relation has to be reversed since the behavior of the symbolic specification should "under-approximate" that of the hybrid automaton. Under mild assumptions on the hybrid automaton and on the partition of the state-space used for abstraction, we describe an approach to compute the symbolic specification. This constitutes the second main contribution of the paper.

Then, we formulate two extensions of our controller synthesis problems with additional safety or reachability requirements so that some unsafe states are avoided or some target states are reached, respectively. The originality of these problem formulations is that these additional requirements are not formulated over the states of the system but over the states of the specification. This, for instance, makes it possible to force the closed-loop system to avoid blocking states of the specification or to reach some states of the specification corresponding to the accomplishment of a given task. To the best of our knowledge, the combination of specifications under the form of a dynamical system and of a logical statement, have not been considered previously in the literature. So our solutions, based on the symbolic abstractions defined above, make another contribution of the paper. Finally, we illustrate our approach with two examples taken from autonomous vehicle control applications: we first consider adaptive cruise control, and then a takeover maneuver. The specifications are given by hybrid automata with additional safety requirements in the case of adaptive cruise control and with additional reachability requirements on the case of the takeover maneuver.

The paper is organized as follows. Section 2 formulates the first controller synthesis problem under consideration. Section 3 provides a characterization of the solution in terms of alternating simulation relations. Section 4 discusses the abstraction algorithms for the system and the specification. Section 5 extends the controller synthesis problem with additional safety or reachability requirements. Finally, Section 6 illustrates our approach with autonomous vehicle control examples.

Notations: $\|.\|_\infty$ is the infinity norm for vectors of $\mathbb{R}^n$ and the associated induced norm for matrices. For $A, B \subseteq \mathbb{R}^n$, $h(A,B)$ denotes the Hausdorff distance between A and B measured by the infinity norm. The inradius of A is $\rho^-(A) = \sup_{x\in\mathbb{R}^n}(\sup\{r \mid x + rB \subseteq A\})$, the circumradius of A is $\rho^+(A) = \inf_{x\in\mathbb{R}^n}(\inf\{r \mid A \subseteq x + rB\})$, where B is the unit ball in the infinity norm. Intuitively, the inradius and the circumradius represent the radii of the largest ball that is included in A and of the smallest ball that contains A, respectively. For a set-valued map $f\colon X \rightrightarrows Y$, its domain is defined as $\mathrm{dom}(f) = \{x \in X \mid f(x) \neq \emptyset\}$, its graph is defined as $\mathrm{graph}(f) = \{(x,y) \in X \times Y \mid y \in f(x)\}$, the image of $X' \subseteq X$ is denoted by $f(X') = \bigcup_{x\in X'} f(x)$.

2 Problem formulation

In this section, we start by introducing some preliminary definitions and then formulate the first controller synthesis problem under consideration.

2.1 Transition systems

In this paper, we consider transition systems to model in a common framework control systems and specifications.

Definition 1 *A* transition system *S is a tuple (X, U, Y, Δ, H), where X is a set of states; U is a set of inputs; Y is a set of outputs; $\Delta : X \times U \rightrightarrows X$ is a set-valued transition map; $H\colon X \longrightarrow Y$ is an output map.*

An input $u \in U$ is called *enabled* at $x \in X$ if $\Delta(x,u) \neq \emptyset$. Let $\mathrm{enab}_\Delta(x) \subseteq U$ denote the set of all inputs enabled at x. If $\mathrm{enab}_\Delta(x) = \emptyset$ the state x is called *blocking*.

Definition 2 *A* trajectory *of S is a sequence $(x_k, u_k)_{k=0}^K$, where $K \in \mathbb{N} \cup \{+\infty\}$, $x_k \in X$, $u_k \in U$, for $0 \leq k \leq K$, and $x_{k+1} \in \Delta(x_k, u_k)$, for $0 \leq k < K$. A trajectory $(x_k, u_k)_{k=0}^K$ is called* maximal *if either $K = +\infty$ or $\Delta(x_K, u_K) = \emptyset$, it is* complete *if $K = +\infty$.*

2.2 Controller synthesis problem

We consider a discrete-time control system S_1 defined as a transition system $S_1 = (X, U, X, F, H_1)$ with state $x \in X \subseteq \mathbb{R}^{n_x}$, control input $u \in U \subseteq \mathbb{R}^{n_u}$, and the output map given by $H_1(x) = x$.

We consider a specification in the form of a discrete-time hybrid automaton S_2 defined as a transition system $S_2 = (X \times P, V, X, G, H_2)$ where P and V are finite sets of modes and external inputs, and the output map

is given by $H_2(x,p) = x$. The set of continuous states of S_2 coincides with the set of states of S_1. Intuitively, the specification describes how the closed-loop system should react to external inputs $v \in V$. An example of such specification is as follows:

Example 3 *We define a hybrid automaton specifying adaptive cruise control. We consider two vehicles moving on a road: the leader is uncontrollable while the follower is controllable. The continuous state is given by* $x = (x^1, x^2, x^3)^T$ *where* x^1 *denote the relative position of the follower with respect to the leader,* x^2 *is the velocity of the follower,* x^3 *is the velocity of the leader. The set of continuous states is* $X = (-\infty, 0) \times [x^2_{\min}, x^2_{\max}] \times [x^3_{\min}, x^3_{\max}]$, *the set of discrete states* $P = \{p^1, p^2\}$ *consists of 2 modes (*p^1*: "track velocity" and* p^2*: "avoid collision") and the set of inputs* $V = \{v_1, \ldots, v_m\}$ *consists of finitely many reference velocities. The dynamics of the hybrid automaton is given by* $(x'p') \in G(x,p,v)$ *if one of the following condition holds*

(1) $p' = p^1$, $x^1 + l \leq 0$ *and* $x^{2\prime} \in [\min(x^2 + C, v - \varepsilon), \max(x^2 - C, v + \varepsilon)]$.
(2) $p' = p^2$, $x^1 + l \leq 0$, $x^2 \leq v + \varepsilon$ *and* $x^{2\prime} \in [x^2_{\min}, v + \varepsilon]$.

The parameters $l > 0$, $C > 0$ *and* $\varepsilon > 0$ *specifying the safety distance, the velocity settling speed and tolerance error. The dynamics does not impose any explicit constraints on* $x^{1\prime}$ *and* $x^{3\prime}$ *(other than* $x^{1\prime} \in (-\infty, 0)$, $x^{3\prime} \in [x^3_{\min}, x^3_{\max}]$*). When* $p' = p^1$, *the follower has to track the velocity* $v \in V$, *which is an external time-varying input that can be provided for instance by the human passenger or by the speed limit regulation. When* $p' = p^2$, *the follower is allowed to reduce its speed to avoid collision. A transition to* p^2 *is possible only if* $x^2 \leq v + \varepsilon$. *For any transition to be enabled, the following safety requirement must be satisfied* $x^1 + l \leq 0$ *(i.e. the follower is at a distance greater than* l *behind the leader).*

In this paper, we consider a class of *controllers* given by a pair of set-valued maps θ: $X \times P \times V \rightrightarrows U$ and $\pi : X \times P \times X \times V \rightrightarrows P$. Controller (θ, π) is said to be *compatible* with S_1 if for all $x \in X$, $p \in P$, $v \in V$,

$$\theta(x,p,v) \subseteq \mathrm{enab}_F(x) \text{ and}$$
$$\forall x' \in F(x, \theta(x,p,v)),\ \pi(x,p,x',v) \neq \emptyset.$$

Then, the closed-loop system is given by $S_{cl} = (X \times P, V, X, \Delta_{cl}, H_2)$ where for all $x \in X$, $p \in P$, $v \in V$,

$$\Delta_{cl}(x,p,v) = \left\{ (x',p') \;\middle|\; \begin{array}{l} x' \in F(x, \theta(x,p,v)) \\ p' \in \pi(x,p,x',v) \end{array} \right\}. \quad (1)$$

Let us remark that if (θ, π) is compatible with S_1 then $v \in \mathrm{enab}_{\Delta_{cl}}(x,p)$ if and only if $\theta(x,p,v) \neq \emptyset$. Let us emphasize that the considered controllers have an internal variable p that takes its values in P, the set of modes of the specification. Hence, the closed-loop system is also a discrete-time hybrid automaton, with the same set of states as the specification.

The first problem considered in this paper is to synthesize a controller (θ, π) and a non-empty set of controllable initial states $Z_c \subseteq X \times P$ such that every trajectory of the closed-loop system S_{cl} initialized in Z_c is a trajectory of the specification S_2.

Problem 4 *Synthesize a controller* (θ, π) *compatible with* S_1 *and a non-empty controllable set* $Z_c \subseteq X \times P$ *such that for every* $(x_0, p_0) \in Z_c$, *every maximal trajectory* $(x_k, p_k, v_k)_{k=0}^K$ *of* S_{cl} *is a maximal trajectory of* S_2.

3 Characterization using alternating simulation

In this section, we establish that the solution of Problem 4 is characterized by the existence of an alternating simulation relation [21] from the specification S_2 to the system S_1.

Definition 5 *Let* $S_a = (X_a, U_a, Y_a, \Delta_a, H_a)$ *and* $S_b = (X_b, U_b, Y_b, \Delta_b, H_b)$ *be two transition systems with* $Y_a = Y_b$. *A relation* $R \subseteq X_a \times X_b$ *is an* alternating simulation relation *from* S_a *to* S_b *if the following conditions are satisfied:*

(1) for every $(x_a, x_b) \in R$ *we have* $H_a(x_a) = H_b(x_b)$;
(2) for every $(x_a, x_b) \in R$ *and for every* $u_a \in \mathrm{enab}_{\Delta_a}(x_a)$ *there exists* $u_b \in \mathrm{enab}_{\Delta_b}(x_b)$ *such that for every* $x'_b \in \Delta_b(x_b, u_b)$ *there exists* $x'_a \in \Delta_a(x_a, u_a)$ *satisfying* $(x'_a, x'_b) \in R$.

It is said that S_b *alternatingly simulates* S_a, *denoted by* $S_a \preceq_{AS} S_b$, *if there exists an alternating simulation relation* $R \neq \emptyset$ *from* S_a *to* S_b.

Remark 6 *Let* R *be an alternating simulation relation from* S_a *to* S_b *and consider* $(x_a, x_b) \in R$. *If* x_a *is a blocking state of* S_a *then condition 2) holds automatically. If* x_b *is blocking for* S_b *then* x_a *must be blocking for* S_a.

Within the setting defined in the previous section, the existence of an alternating simulation relation can be interpreted as follows. Let us consider the repeated two-player game between the environment and the controller where the goal of the controller is to keep the specification and system outputs indistinguishable: at each time step, the environment picks the specification input v, then the controller picks the system input u, then the environment picks the system successor state x' that the controller finally has to match with a specification successor state of the form (x', p'). Intuitively, an alternating simulation relation from S_2 to S_1 defines a set from which the controller has a winning strategy.

3.1 Sufficiency

Let us assume that $R \subseteq X \times P \times X$, is an alternating simulation relation from S_2 to S_1, then let us define the set $Z_c \subseteq X \times P$ and the controller (θ, π) as follows:

$$Z_c = \{(x,p) \in X \times P \mid ((x,p),x) \in R\}, \tag{2}$$

$$\theta(x,p,v) = \left\{ u \in \mathrm{enab}_F(x) \;\middle|\; \begin{array}{l} \forall x' \in F(x,u),\ \exists p' \in P : \\ (x',p') \in G(x,p,v) \cap Z_c \end{array} \right\}, \tag{3}$$

$$\pi(x,p,x',v) = \{p' \in P \mid (x',p') \in G(x,p,v) \cap Z_c\}. \tag{4}$$

Theorem 7 *Let $R \neq \emptyset$ be an alternating simulation relation from S_2 to S_1, let set Z_c and controller (θ, π) be defined by (2), (3), (4). Then, (θ, π) and Z_c solve Problem 4.*

PROOF. The fact that $Z_c \neq \emptyset$ follows from (2) and $R \neq \emptyset$. We now prove that (θ, π) is compatible with S_1. By (3) it is clear that $\theta(x,p,v) \subseteq \mathrm{enab}_F(x)$. Moreover, by (3) and (4), we get that for all $x' \in F(x, \theta(x,p,v))$, $\pi(x,p,x',v) \neq \emptyset$. Then, let us consider $(x_0,p_0) \in Z_c$, and a maximal trajectory $(x_k,p_k,v_k)_{k=0}^K$ of S_{cl}. Then, by (1), we get that $p_{k+1} \in \pi(x_k,p_k,x_{k+1},v_k)$ for $0 \leq k < K$. By (4), we get that $(x_{k+1},p_{k+1}) \in G(x_k,p_k,v_k) \cap Z_c$ for $0 \leq k < K$. Hence, it follows that $(x_k,p_k,v_k)_{k=0}^K$ is a trajectory of S_2 and that $(x_k,p_k) \in Z_c$, for $0 \leq k \leq K$. Let us assume that $(x_k,p_k,v_k)_{k=0}^K$ is not maximal for S_2, then $v_K \in \mathrm{enab}_G(x_K,p_K)$. Since $(x_K,p_K) \in Z_c$, we get $((x_K,p_K),x_K) \in R$, which is an alternating simulation relation from S_2 to S_1. Therefore, from the condition 2) in Definition 5, there exists $u \in \mathrm{enab}_F(x_K)$ such that for all $x' \in F(x_K,u)$, there exists $(x'',p'') \in G(x_K,p_K,v_K)$ such that $((x'',p''),x') \in R$. Condition 1) of Definition 5 gives that $x'' = x'$ and therefore $(x',p'') \in Z_c$. Hence, by (3), $u \in \theta(x_K,p_K,v_K)$, which is therefore non-empty. Since (θ,π) is compatible with S_1, we obtain that $v_K \in \mathrm{enab}_{\Delta_{cl}}(x_K,p_K)$. It follows that $(x_k,p_k,v_k)_{k=0}^K$ is not maximal for S_{cl}, which leads to a contradiction. □

3.2 Necessity

While Theorem 7 shows that solutions of Problem 4 can be obtained from alternating simulation relations, the following theorem shows the converse result:

Theorem 8 *Let controller (θ, π) and set $Z_c \neq \emptyset$ solve Problem 4. Let us define a relation R by the following: $((x,p),x') \in R$ if and only if $x = x'$ and there exists $(x_0,p_0) \in Z_c$ and a trajectory $(x_k,p_k,v_k)_{k=0}^K$ of S_{cl}, with $K \in \mathbb{N}$, such that $x_K = x$, $p_K = p$. Then $R \neq \emptyset$ and is an alternating simulation relation from S_2 to S_1.*

PROOF. For all $(x,p) \in Z_c$, $((x,p),x) \in R$, then $R \neq \emptyset$ follows from $Z_c \neq \emptyset$. Condition 1) of Definition 5 does obviously hold. Now, let $((x,p),x) \in R$ and consider the corresponding trajectory $(x_k,p_k,v_k)_{k=0}^K$ of S_{cl}, such that $(x_0,p_0) \in Z_c$ and $x_K = x$, $p_K = p$. If (x_K,p_K) is blocking for S_2, then by Remark 6, condition 2) of Definition 5 holds. Otherwise, let $v \in \mathrm{enab}_G(x_K,p_K)$. Then let $v'_k = v_k$ for $0 \leq k \leq K-1$ and $v'_K = v$. Then, $(x_k,p_k,v'_k)_{k=0}^K$ is a trajectory of S_{cl} and also of S_2.Moreover, it is not maximal for S_2, which implies by Problem 4 that it is not maximal for S_{cl}. Hence, there exists $u \in \theta(x_K,p_K,v'_K)$. Let $x' \in F(x_K,u)$, since (θ,π) is compatible with S_1 we get that there exists $p' \in \pi(x_K,p_K,x',v'_K)$. By (1), we get that $(x',p') \in \Delta_{cl}(x_K,p_K,v'_K)$. Hence, $(x_k,p_k,v'_k)_{k=0}^{K+1}$ where $x_{K+1} = x'$, $p_{K+1} = p'$ and $v'_{K+1} \in V$ is a trajectory of S_{cl}. Then, $((x',p'),x') \in R$. Moreover Problem 4 implies that $(x_k,p_k,v'_k)_{k=0}^{K+1}$ is also a trajectory of S_2, which implies that $(x',p') \in G(x_K,p_K,v'_K)$. Hence, condition 2) of Definition 5 holds. □

4 Abstraction based approach

In the previous section, we have shown that solving Problem 4 is actually equivalent to computing an alternating simulation relation from S_2 to S_1. In this section, we present an approach for computing such a relation. Our approach is based on the computation of a symbolic system $\hat{S}_1$ and a symbolic specification $\hat{S}_2$ such that $S_2 \preceq_{AS} \hat{S}_2$ and $\hat{S}_1 \preceq_{AS} S_1$. Then, if we show that $\hat{S}_2 \preceq_{AS} \hat{S}_1$, we get $S_2 \preceq_{AS} S_1$, by transitivity of alternating simulation (as per Proposition 4.23 in [21]).

4.1 Abstraction for the control system

The construction of the symbolic system $\hat{S}_1$ ressembles approaches in the literature [5,18] with the slight difference that we do not assume that $\mathrm{enab}_F(x) = U$, for all $x \in X$. For this reason, we briefly present the abstraction method and state the main result without proof.

Let the state space X be covered by a finite partition $(X_q)_{q \in Q}$: $X = \cup_{q \in Q} X_q$. Let $\hat{U} \subseteq U$ be a finite subset of control inputs, and let us define an abstract system $\hat{S}_1 = (X, \hat{U}, X, \hat{F}, H_1)$ where the transition map $\hat{F}$ is given by:

$$x' \in \hat{F}(x,\hat{u}) \iff x \in X_q,\ x' \in X_{q'},\ q' \in \hat{\Delta}_1(q,\hat{u}) \tag{5}$$

with the finite transition map $\hat{\Delta}_1 : Q \times \hat{U} \rightrightarrows Q$ satisfying for all $q \in Q$

$$\mathrm{enab}_{\hat{\Delta}_1}(q) \subseteq \bigcap_{x \in X_q} \mathrm{enab}_F(x) \tag{6}$$

and for all $q, q' \in Q$, $\hat{u} \in \text{enab}_{\hat{\Delta}_1}(q)$

$$X_{q'} \cap F(X_q, \hat{u}) \neq \emptyset \implies q' \in \hat{\Delta}_1(q, \hat{u}). \tag{7}$$

Condition (6) does not appear in [5,18], where it is assumed that $\text{enab}_F(x) = U$, for all $x \in X$. Indeed, in that case (6) is automatically satisfied. With minor modifications to the proofs in these works, we can show that the identity relation on X is an alternating simulation relation and we get:

Proposition 9 *For transition system S_1 and symbolic system $\hat{S}_1$ satisfying (5), (6), (7), we have $\hat{S}_1 \preceq_{AS} S_1$.*

4.2 Abstraction for the specification

We now present the construction of the symbolic specification $\hat{S}_2$ such that $S_2 \preceq_{AS} \hat{S}_2$. Such construction is not available in the literature, since the direction of the alternating simulation relation is reversed in comparison to the system case. Let us introduce for all $p, p' \in P$, $v \in V$, the map $G^v_{p,p'} : X \rightrightarrows X$ given by

$$G^v_{p,p'}(x) = \{x' \in X |\ (x', p') \in G(x, p, v)\}.$$

Then, $G(x, p, v) = \bigcup_{p' \in P} G^v_{p,p'}(x) \times \{p'\}$.

Let $(X_q)_{q \in Q}$ be the same partition as in the previous section and let us define $\hat{S}_2 = (X \times P, V, X, \hat{G}, H_2)$ where the transition map $\hat{G}$ is given by:

$$\begin{array}{c} (x', p') \in \hat{G}(x, p, v) \iff \\ x \in X_q,\ x' \in X_{q'},\ (q', p') \in \hat{\Delta}_2(q, p, v) \end{array} \tag{8}$$

with the finite transition map $\hat{\Delta}_2 : Q \times P \times V \rightrightarrows Q \times P$ satisfying for all $q, q' \in Q$, $p, p' \in P$, $v \in V$,

$$(q', p') \in \hat{\Delta}_2(q, p, v) \iff X_q \times X_{q'} \subseteq \text{graph}(G^v_{p,p'}). \tag{9}$$

We now provide conditions under which the identity relation on $X \times P$ is an alternating simulation relation from S_2 to $\hat{S}_2$.

Assumption 10 *The transition map G satisfies the following conditions for some $L > 0$, $\delta > 0$:*

(1) for all $p, p' \in P$, $v \in V$, $x_1, x_2 \in \text{dom}(G^v_{p,p'})$,

$$h(G^v_{p,p'}(x_1), G^v_{p,p'}(x_2)) \leq L\|x_1 - x_2\|_\infty;$$

(2) for all $p, p' \in P$, $v \in V$, $x \in \text{dom}(G^v_{p,p'})$,

$$\rho^-\left(G^v_{p,p'}(x)\right) \geq \delta.$$

Assumption 10 concerns the maps $G^v_{p,p'}$. The first item requires the maps $G^v_{p,p'}$ to be Lipschitz on their domains, which is a fairly mild assumption. The second item requires that $G^v_{p,p'}(x)$ has a minimal inradius whenever it is not empty. In particular, it means that $G^v_{p,p'}(x)$ cannot be a singleton, and that the specification should enable some non-determinism.

Assumption 11 *The partition $(X_q)_{q \in Q}$ satifies:*

(1) for all $q \in Q$, $p \in P$, $v \in V$,

$$\begin{array}{l} \exists x \in X_q,\ G(x, p, v) \neq \emptyset \implies \\ \exists p' \in P,\ X_q \subseteq \text{dom}(G^v_{p,p'}); \end{array}$$

(2) for all $q \in Q$, $\rho^+(X_q) < \delta/(2 + L)$, where L and δ are the same as in Assumption 10.

Remark 12 *Let us point out that condition 1) in Assumption 11 can be replaced by the stronger (but easier to check) condition that for all $q \in Q$, $p, p' \in P$, $v \in V$,*

$$X_q \cap \text{dom}(G^v_{p,p'}) \neq \emptyset \implies X_q \subseteq \text{dom}(G^v_{p,p'}).$$

Assumption 11 concerns the partition $(X_q)_{q \in Q}$. It should be emphasized that it is always possible to choose a partition such that this assumption is satisfied. From the previous remark, condition 1) can be satisfied by having the regions X_q either fully inside or outside $\text{dom}(G^v_{p,p'})$. Condition 2) can be satisfied by choosing the regions X_q to be small enough. It is important to remark that if the maps $G^v_{p,p'}$ were deterministic then condition 2) of Assumption 10 would only hold for $\delta = 0$. Then, it would become impossible to find a finite partition $(X_q)_{q \in Q}$ such that condition 2) of Assumption 11 is satisfied. We now establish the following instrumental result:

Lemma 13 *Under Assumptions 10 and 11, we have for all $x \in X$, $p \in P$, $\text{enab}_G(x, p) = \text{enab}_{\hat{G}}(x, p)$.*

PROOF. Consider an arbitrary (x, p, v) such that $G(x, p, v) \neq \emptyset$. Let q be such that $x \in X_q$, then from condition 1) of Assumption 11, there exists $p' \in P$, such that $X_q \subseteq \text{dom}(G^v_{p,p'})$. From conditions 1) and 2) in Assumption 10, it follows that

$$\rho^-\left(\bigcap_{x \in X_q} G^v_{p,p'}(x)\right) \geq \delta - L\rho^+(X_q).$$

By condition 2) of Assumption 11, we get for all $q' \in Q$,

$$\rho^-\left(\bigcap_{x \in X_q} G^v_{p,p'}(x)\right) > 2\rho^+\left(X_{q'}\right),$$

which implies since $(X_q)_{q\in Q}$ is a partition that there exists $q' \in Q$ such that $X_{q'} \subseteq \bigcap_{x\in X_q} G^v_{p,p'}(x)$. Then, $X_q \times X_{q'} \subseteq \text{graph}(G^v_{p,p'})$ and $\hat{G}(x,p,v) \neq \emptyset$. Thus, $\text{enab}_G(x,p) \subseteq \text{enab}_{\hat{G}}(x,p)$. The inclusion $\text{enab}_{\hat{G}}(x,p) \subseteq \text{enab}_G(x,p)$, is a direct consequence of (8) and (9). □

Proposition 14 *Under Assumptions 10 and 11, for transition system S_2 and symbolic system $\hat{S}_2$ satisfying (8), (9), we have $S_2 \preceq_{AS} \hat{S}_2$.*

PROOF. Consider the identity relation on $X \times P$ and let us prove that it is an alternating simulation relation. Condition 1) of Definition 5 does obviously hold. Condition 2) reads: for every $(x,p) \in X \times P$ and every $v_1 \in \text{enab}_G(x,p)$ there exists $v_2 \in \text{enab}_{\hat{G}}(x,p)$ such that $\hat{G}(x,p,v_2) \subseteq G(x,p,v_1)$. Let us verify a stricter statement when $v_2 = v_1 = v$. Consider an arbitrary $(x,p) \in X \times P$, $v \in \text{enab}_G(x,p)$, then from Lemma 13, $v \in \text{enab}_{\hat{G}}(x,p)$. Let $(x',p') \in \hat{G}(x,p,v)$, let $q, q' \in Q$ be such that $x \in X_q$, $x' \in X_{q'}$. Then $(x,x') \in \text{graph}(G^v_{p,p'})$. Therefore, $(x',p') \in G(x,p,v)$. □

We now briefly describe a practical approach to compute the abstract specification $\hat{S}_2$ when the original specification S_2 is described by a piecewise affine hybrid automaton. Let us assume that for all $p,p' \in P$, $v \in V$, $\text{dom}(G^v_{p,p'}) = D^v_{p,p'}$, and that for all $x \in \text{dom}(G^v_{p,p'})$ $G^v_{p,p'}(x) = A^v_{p,p'}x + W^v_{p,p'}$, where $A^v_{p,p'}$ is a matrix in $\mathbb{R}^{n_x \times n_x}$, $D^v_{p,p'}$, $W^v_{p,p'}$ are closed convex polytopes in $\mathbb{R}^{n_x}$. For Assumption 10 to hold, it is sufficient that the sets $W^v_{p,p'}$ have non-empty interior for all $p,p' \in P$, $v \in V$. Then, L and δ satisfying Assumption 10 are given by

$$L = \max_{p,p'\in P, v\in V} \|A^v_{p,p'}\|, \ \delta = \min_{p,p'\in P, v\in V} \rho^-(W^v_{p,p'}).$$

Let us assume that the partition $(X_q)_{q\in Q}$ is chosen so as to satisfy Assumption 11. Let us remark that if X is a polytope it is always possible to choose such a partition where for all $q \in Q$, X_q is a polytope. Then according to (9), $(q',p') \in \hat{\Delta}_2(q,p,v)$ if and only if

$$X_q \subseteq D^v_{p,p'} \text{ and } X_{q'} - A^v_{p,p'}X_q \subseteq W^v_{p,p'}.$$

These inclusions can be checked by verifying that for all x and x' belonging to the set of vertices of X_q and $X_{q'}$ respectively, $x \in D^v_{p,p'}$ and $x' - A^v_{p,p'}x \in W^v_{p,p'}$.

In this section, we have shown how to build a symbolic system $\hat{S}_1$ and a symbolic specification $\hat{S}_2$ such that $S_2 \preceq_{AS} \hat{S}_2$ and $\hat{S}_1 \preceq_{AS} S_1$. In both cases, the alternating simulation relation is given by the identity relation. Then, according to Proposition 4.23 in [21], if $R \subseteq X \times P \times X$, $R \neq \emptyset$, is an alternating simulation relation from $\hat{S}_2$ to $\hat{S}_1$, we get that R is also an alternating simulation relation from S_2 to S_1, and provides a solution to Problem 4 as per Theorem 7. The use of a symbolic system and a symbolic specification is the only source of conservatism of the proposed solution to Problem 4. This conservatism can be reduced by choosing finer partitions of X, at the expense of an increased computational complexity.

We do not discuss the computation of the alternating simulation relation from $\hat{S}_2$ to $\hat{S}_1$, which can be done either by extending (to the alternated case) the algorithms for discrete simulation games presented in [21], or by using the approach presented in Section 5.1 to solve Problem 16 of which Problem 4 is a special case.

Remark 15 *Rigorously speaking, $\hat{S}_1$ and $\hat{S}_2$ are infinite transition systems since they have infinite sets of states. However, their transition relations are finitely represented by the maps $\hat{\Delta}_1 : Q \times \hat{U} \rightrightarrows Q$ and $\hat{\Delta}_2 : Q \times P \times V \rightrightarrows Q \times P$. For this reason, we refer to $\hat{S}_1$ and $\hat{S}_2$ as symbolic system and specification, respectively. Using Q and $Q \times P$, instead of X and $X \times P$, as sets of states for $\hat{S}_1$ and $\hat{S}_2$, would make it impossible to get $S_2 \preceq_{AS} \hat{S}_2$ and $\hat{S}_1 \preceq_{AS} S_1$ with X as output set (without resorting to set valued output maps). Using X as the output set instead of Q is important since it makes the formulation of Problem 4 independent of the choice of the partition. It is important to note that the controller synthesis algorithms presented in the next section, only operates on the finite objects Q, P, $\hat{U}$, V, $\hat{\Delta}_1$ and $\hat{\Delta}_2$.*

5 Safety and reachability requirements

By solving Problem 4, we guarantee that the closed-loop system S_{cl} behaves in the same way as the specification S_2. In this section, we show how to take into account additional requirements such as (generalized) safety and reachability properties.

Safety requirements can often be encoded directly in S_2 by blocking transitions starting from unsafe states: e.g. in Example 3, there is no transition enabled if $x^1 + l > 0$, (i.e. the follower is at a distance less than l from the leader). However, Problem 4 does not prevent reaching such blocking states: actually, from Remark 6 and Theorem 7, all blocking states of S_2 are elements of the controllable set Z_c. Hence, to take into account safety requirements, one needs to modify the formulation of Problem 4 to prevent S_{cl} from reaching blocking states of S_2. We actually formulate a more general problem, which allows not only to enforce safety requirements but also has Problem 4 as a special case. For that purpose, let us consider a set of terminal states given by $Z_f \subseteq X \times P$.

Problem 16 (Safety requirements) *Synthesize a controller (θ, π) compatible with S_1 and a controllable*

set $Z_c \subseteq X \times P$ such that for every $(x_0, p_0) \in Z_c$, for every maximal trajectory $(x_k, p_k, v_k)_{k=0}^K$ of S_{cl}, one of the following condition holds:

(1) $(x_k, p_k, v_k)_{k=0}^K$ is a trajectory of S_2, $K \in \mathbb{N}$ and $(x_K, p_K) \in Z_f$;
(2) $(x_k, p_k, v_k)_{k=0}^K$ is a maximal trajectory of S_2, and either $K = +\infty$, or $K \in \mathbb{N}$ and $\text{enab}_G(x_K, p_K) \neq \emptyset$.

Some explanations on Problem 16 are in order. Let us consider the following cases:

- If $Z_f = \emptyset$, condition 1) is never satisfied and only condition 2) is relevant. In that case, all states that are reachable by trajectories of S_{cl} are non-blocking states of S_2 (even though we can have $v_K \notin \text{enab}_G(x_K, p_K)$). Then, if the sequence of external inputs is enabled in the specification (i.e. if $v_k \in \text{enab}_G(x_k, p_k)$ for all $0 \leq k \leq K$), then all maximal trajectories of S_{cl} are complete trajectories of S_2. Hence, as explained above, solving Problem 16 with $Z_f = \emptyset$ makes it possible to enforce the safety requirements encoded in S_2.
- If $Z_f = \{(x, p) \in X \times P | \text{ enab}_G(x, p) = \emptyset\}$ (i.e. Z_f is the set of blocking states of S_2), then in condition 1) the trajectory is also maximal for S_2. Hence, all trajectories that are maximal for S_{cl} are also maximal for S_2, which is the requirement of Problem 4. Hence, the solution provided in this section to solve Problem 16 can be readily used to solve Problem 4.
- In the general case, we have that all states outside Z_f that are reachable by trajectories of S_{cl} are non-blocking states of S_2. Moreover, if the sequence of external inputs is enabled in the specification, then all maximal trajectories of S_{cl} are trajectories of S_2 and are either complete or end in a terminal state $Z_f \subseteq X \times P$.

Problem 16 does not allow to enforce reachability properties. Indeed, from condition 2), complete trajectories of S_2 that never reach Z_f are always accepted. Reachability requirement can be taken into account as follows:

Problem 17 (Reachability requirements)
Synthesize a controller (θ, π) compatible with S_1 and a controllable set $Z_c \subseteq X \times P$ such that for every $(x_0, p_0) \in Z_c$, for every maximal trajectory $(x_k, p_k, v_k)_{k=0}^K$ of S_{cl}, one of the following condition holds:

(1) $(x_k, p_k, v_k)_{k=0}^K$ is a trajectory of S_2, $K \in \mathbb{N}$ and $(x_K, p_K) \in Z_f$;
(2) $(x_k, p_k, v_k)_{k=0}^K$ is a maximal trajectory of S_2, and $K \in \mathbb{N}$ and $\text{enab}_G(x_K, p_K) \neq \emptyset$.

From condition 2) of Problem 17, all states outside Z_f that are reachable by trajectories of S_{cl} are non-blocking states of S_2. Moreover, if the sequence of external inputs is enabled in the specification, then from condition 1), all maximal trajectories of S_{cl} are trajectories of S_2 and end in a terminal state $Z_f \subseteq X \times P$.

It should be noted that any solution of Problem 17 is also a solution of Problem 16. Intuitively, Problem 17 requires that S_{cl} behaves as S_2 until a final state in Z_f is reached. Problem 16 allows additionally for the possibility of not reaching Z_f. In that case, S_2 must behave as S_{cl} forever. Essentially, the difference between Problem 17 and Problem 16 is the same as that between "until" and "weak until" in Linear Temporal Logic [3]. Let us remark that for $Z_f = \emptyset$, Problem 17 does not admit any solution while for Problem 16, this case corresponds to pure safety requirements.

In the following, we provide solutions to Problems 16 and 17 based on symbolic abstractions. Let $\hat{S}_1$ and $\hat{S}_2$ be the symbolic system and specification defined in Section 4. Given a controller (θ, π), we denote $\hat{S}_{cl}$ the closed-loop abstract system with transition relation $\hat{\Delta}_{cl}$, defined as in (1). We first show the following result, which shows that if (θ, π) and Z_c solve Problem 16 or 17 for $\hat{S}_1$ and $\hat{S}_2$, they also provide a solution for system and specification S_1 and S_2.

Lemma 18 *If a controller (θ, π) is compatible with $\hat{S}_1$ then it is compatible with S_1. Any maximal trajectory of S_{cl} is also a maximal trajectory of $\hat{S}_{cl}$. Under Assumptions 10 and 11, any (maximal) trajectory of $\hat{S}_2$ is a (maximal) trajectory of S_2.*

PROOF. From (5), (6) and (7) it follows that for all $x \in X$, $\text{enab}_{\hat{F}}(x) \subseteq \text{enab}_F(x)$ and that for all $u \in \text{enab}_{\hat{F}}(x)$, $F(x, u) \subseteq \hat{F}(x, u)$. Therefore, (θ, π) being compatible with $\hat{S}_1$ implies that it is also compatible with S_1. Moreover, for all $x \in X$, $p \in P$, $\text{enab}_{\hat{\Delta}_{cl}}(x, p) = \text{enab}_{\Delta_{cl}}(x, p)$ and consists of inputs $v \in V$ such that $\theta(x, p, v) \neq \emptyset$. Then, we get that for all $v \in \text{enab}_{\Delta_{cl}}(x, p)$, $F(x, \theta(x, p, v)) \subseteq \hat{F}(x, \theta(x, p, v))$, which implies that $\Delta_{cl}(x, p, v) \subseteq \hat{\Delta}_{cl}(x, p, v)$. Hence, any maximal trajectory of S_{cl} is also a maximal trajectory of $\hat{S}_{cl}$. From Lemma 13 and from the proof of Proposition 14, we get that for all $x \in X$, $p \in P$, $\text{enab}_G(x, p) = \text{enab}_{\hat{G}}(x, p)$ and that for all $v \in \text{enab}_{\hat{G}}(x, p)$, $\hat{G}(x, p, v) \subseteq G(x, p, v)$. Hence, any (maximal) trajectory of $\hat{S}_2$ is also a (maximal) trajectory of S_2. □

Before providing detailed solutions to Problems 16 and 17, we define the following operator:

Definition 19 *For $\hat{Z} \subseteq Q \times P$ the* controllable predecessor *of $\hat{Z}$ is* $\text{Pre}\ \hat{Z} \subseteq Q \times P$ *where $(q, p) \in \text{Pre}\ \hat{Z}$ if and only if*

(1) $\text{enab}_{\hat{\Delta}_2}(q,p) \neq \emptyset$, *and*
(2) $\forall v \in \text{enab}_{\hat{\Delta}_2}(q,p), \exists u \in \text{enab}_{\hat{\Delta}_1}(q)$:
$\forall q' \in \hat{\Delta}_1(q,u), \exists p' \in P : (q',p') \in \hat{\Delta}_2(q,p,v) \cap \hat{Z}$.

Intuitively, Pre $\hat{Z}$ consists of states from which the controller can enforce the successor state to belong to $\hat{Z}$. Let us remark that since $Q \times P$ is a finite set, the computation of Pre $\hat{Z}$ does not present any difficulty. We also define the set of symbolic terminal states:

$$\hat{Z}_f = \{(q,p) \in Q \times P \mid X_q \times \{p\} \subseteq Z_f\}.$$

5.1 Safety requirements

We start by describing a solution approach to Problem 16. Let $\hat{Z}^\infty$ denote the limit of the sequence $(\hat{Z}^k)_{k\in\mathbb{N}}$ defined by:

$$\hat{Z}^0 = Q \times P,\ \hat{Z}^{k+1} = \hat{Z}_f \cup \text{Pre}\ \hat{Z}^k. \tag{10}$$

Since for all $k \in \mathbb{N}$, $\hat{Z}^k$ is finite and $\hat{Z}^{k+1} \subseteq \hat{Z}^k$, it follows that the fixed point $\hat{Z}^\infty$ is reached in finite time. Let Z_c be defined as follows:

$$Z_c = \left\{(x,p) \in X \times P \,\middle|\, x \in X_q,\ (q,p) \in \hat{Z}^\infty \right\}. \tag{11}$$

Then we may define the controller (θ, π) as follows:

$$\theta(x,p,v) = \left\{ u \in \text{enab}_{\hat{F}}(x) \,\middle|\, \begin{array}{l} \forall x' \in \hat{F}(x,u),\ \exists p' \in P : \\ (x',p') \in \hat{G}(x,p,v) \cap Z_c \end{array} \right\}, \tag{12}$$

$$\pi(x,p,x',v) = \{p' \in P \mid (x',p') \in \hat{G}(x,p,v) \cap Z_c\}. \tag{13}$$

Theorem 20 *Under Assumptions 10 and 11, let the set Z_c and the controller (θ,π) be defined by (11), (12), (13). Then, (θ,π) and Z_c solve Problem 16.*

PROOF. Using the same arguments as in the beginning of the proof of Theorem 7, we get that (θ,π) is compatible with $\hat{S}_1$ and that any maximal trajectory $(x_k,p_k,v_k)_{k=0}^K$ of $\hat{S}_{cl}$ with $(x_0,p_0) \in Z_c$ is a trajectory of $\hat{S}_2$ and that $(x_k,p_k) \in Z_c$ for all $0 \leq k \leq K$. If $K = +\infty$ or if $K \in \mathbb{N}$ and $(x_K,p_K) \in Z_f$, the requirements of Problem 16 are met for $\hat{S}_1$ and $\hat{S}_2$. Then, let us assume that $K \in \mathbb{N}$ and $(x_K,p_K) \notin Z_f$. Since $(x_K,p_K) \in Z_c \setminus Z_f$, it follows by (11), (10) and (8) that $\text{enab}_{\hat{G}}(x_K,p_K) \neq \emptyset$. Let us assume that $(x_k,p_k,v_k)_{k=0}^K$ is not maximal for $\hat{S}_2$, then $v_K \in \text{enab}_{\hat{G}}(x_K,p_K)$ and from (11), (10), (8), (5), we get that there exists $u \in \text{enab}_{\hat{F}}(x_K)$ such that for all $x' \in \hat{F}(x_k,u)$, there exits $p' \in P$ such that $(x',p') \in \hat{G}(x,p,v) \cap Z_c$. Then, from (12), $u \in \theta(x_K,p_K,v_K)$, which is therefore nonempty. Since (θ,π) is compatible with $\hat{S}_1$, we obtain that $v_K \in \text{enab}_{\hat{\Delta}_{cl}}(x_K,p_K)$. It follows that $(x_k,p_k,v_k)_{k=0}^K$ is not maximal for $\hat{S}_{cl}$, which leads to a contradiction. Hence, it follows that (θ,π) and Z_c solve Problem 16 for $\hat{S}_1$ and $\hat{S}_2$. By Lemma 18, they also solve Problem 16 for S_1 and S_2. □

5.2 Reachability requirements

We now present a solution approach to Problem 17. Let $\hat{Z}^\infty$ denote the limit of the sequence $(\hat{Z}^k)_{k\in\mathbb{N}}$ defined by:

$$\hat{Z}^0 = \hat{Z}_f,\ \hat{Z}^{k+1} = \hat{Z}_f \cup \text{Pre}\ \hat{Z}^k. \tag{14}$$

Since $Q \times P$ is finite and for all $k \in \mathbb{N}$, $\hat{Z}^k \subseteq \hat{Z}^{k+1} \subseteq Q \times P$, it follows that the fixed point $\hat{Z}^\infty$ is reached in finite time. Let Z_c be defined by (11), and for $k \in \mathbb{N}$, let

$$Z^k = \left\{(x,p) \in X \times P \,\middle|\, x \in X_q,\ (q,p) \in \hat{Z}^k \right\}.$$

Then, for $(x,p) \in Z_c$ we define

$$\kappa(x,p) = \min\left\{k \in \mathbb{N} \middle|\ (x,p) \in Z^k\right\}. \tag{15}$$

Then, let (θ,π) be given, for $(x,p) \in Z^0$, by $\theta(x,p,v) = \emptyset$, $\pi(x,p,x',v) = \emptyset$, and for $(x,p) \in Z_c \setminus Z^0$, by

$$\theta(x,p,v) = \left\{ u \in \text{enab}_{\hat{F}}(x) \,\middle|\, \begin{array}{l} \forall x' \in \hat{F}(x,u),\ \exists p' \in P : \\ (x',p') \in \hat{G}(x,p,v) \cap \\ Z^{\kappa(x,p)-1} \end{array} \right\}, \tag{16}$$

$$\pi(x,p,x',v) = \{p' \in P \mid (x',p') \in \hat{G}(x,p,v) \cap Z^{\kappa(x,p)-1}\}. \tag{17}$$

Theorem 21 *Under Assumptions 10 and 11, let the set Z_c and the controller (θ,π) be defined by (11), (16), (17). Then, (θ,π) and Z_c solve Problem 17.*

PROOF. Using the same arguments as in the beginning of the proof of Theorem 7, we get that (θ,π) is compatible with $\hat{S}_1$ and that any maximal trajectory $(x_k,p_k,v_k)_{k=0}^K$ of $\hat{S}_{cl}$ with $(x_0,p_0) \in Z_c$ is a trajectory of $\hat{S}_2$ and that $\kappa(x_k,p_k)$ is strictly discreasing and nonnegative. Then, necessarily $K \in \mathbb{N}$. If $(x_K,p_K) \in Z_f$, the requirements of Problem 16 are met for $\hat{S}_1$ and $\hat{S}_2$. Then, let us assume that $(x_K,p_K) \notin Z_f$. Then, $(x_K,p_K) \notin Z^0$ and $\kappa(x_K,p_K) > 0$. Then, it follows by (15), (14) and (8) that $\text{enab}_{\hat{G}}(x_K,p_K) \neq \emptyset$. Let us assume that $(x_k,p_k,v_k)_{k=0}^K$ is not maximal for $\hat{S}_2$, then $v_K \in \text{enab}_{\hat{G}}(x_K,p_K)$ and from (15), (14), (8),

(5), we get that there exists $u \in \text{enab}_{\hat{F}}(x_K)$ such that for all $x' \in \hat{F}(x_k, u)$, there exits $p' \in P$ such that $(x', p') \in \hat{G}(x, p, v) \cap Z^{\kappa(x_K, p_K)-1}$. Then, from (16), $u \in \theta(x_K, p_K, v_K)$, which is therefore non-empty. The end of the proof is identical to that of Theorem 20. □

We end this section by highlighting the duality between Problems 16 and 17 whose respective solutions are obtained by computing the greatest and least fixed points of the same operator (see (10) and (14)). The implication on computational complexity is that Problems 16 and 17 can be solved in linear time of problem data by adapting the method presented in [11].

6 Autonomous vehicle examples

In this section, we provide illustrations of our approach by showing instances of Problems 16 and 17 in the context of autonomous vehicle control. We first consider adaptive cruise control and then a takeover maneuver. Both problems have been solved using the MATLAB toolbox Co4Pro [1], which can perform automatic abstractions of systems and specifications as well as symbolic controller synthesis. We ran the algorithms on Intel Core i7 2.8GHz CPU, 16 GB RAM.

6.1 Safety: adaptive cruise control

Let us consider the problem that was introduced in Example 3. Let the system S_1 of two vehicles be described by discrete-time equations:

$$\begin{aligned} x^1_{k+1} &= x^1_k + (x^2_k - x^3_k)T_0, \\ x^2_{k+1} &= \chi\left(x^2_k + \alpha(u_k, x^2_k)T_0,\ [x^2_{\min}, x^2_{\max}]\right), \\ x^3_{k+1} &= \chi\left(x^3_k + w_k T_0,\ [x^3_{\min}, x^3_{\max}]\right). \end{aligned}$$

where $T_0 > 0$ is the sampling period, $\alpha(u, x) = u - M^{-1}(f_0 + f_1 x + f_2 x^2)$ and $\chi(x, [x_1, x_2]) = \min\{\max\{x, x_1\}, x_2\}$. The vector of parameters $f = (f_0, f_1, f_2) \in \mathbb{R}^3_+$ describes the road friction and the vehicle aerodynamics whose numerical values are taken from [14]: $f_0 = 51\ N$, $f_1 = 1.2567\ Ns/m$, $f_2 = 0.4342\ Ns^2/m^2$. Here $u_k \in [u_{\min}, u_{\max}]$ is the torque applied to the wheels of the follower and is the control input, $w_k \in [w_{\min}, w_{\max}]$ is the acceleration of the leader and is an uncertain input. We recall that the state space, already defined in Example 3 is $X = (-\infty, 0) \times [x^2_{\min}, x^2_{\max}] \times [x^3_{\min}, x^3_{\max}]$ We choose the following parameters: $M = 1370\ kg$, $x^2_{\min} = x^3_{\min} = 10\ m/s$, $x^2_{\max} = 30\ m/s$, $x^3_{\max} = 25\ m/s$, $u_{\min} = w_{\min} = -3\ m/s^2$ $u_{\max} = w_{\max} = 3\ m/s^2$, $T_0 = 0.5\ s$.

[1] https://github.com/girardan/Co4Pro

The specification S_2 is given as in Example 3. The safety requirement that $x^1_k + l \leq 0$ for all $k \in \mathbb{N}$ is directly encoded in S_2. To enforce the safety requirement, we then solve Problem 16 with an empty set of terminal states $Z_f = \emptyset$, to avoid reaching blocking state of the specification (in particular those where $x_1 + l > 0$). The numerical values of the specification parameters are $C = 0.4\ m/s$, $\varepsilon = 1\ m/s$ and $l = 10\ m$. The external input take values in the set of reference velocities $V = \{15, 20, 25\}$. Let us remark that this specification S_2 can be written as a piecewise affine hybrid automaton. We then choose an appropriate partition of the state space such that Assumptions 10 and 11 are satisfied.

A controller (θ, π) has been synthesized by the approach presented in Section 5.1. The partition of X is obtained using partitions on each subspace. The partition of $(-\infty, 0)$ is given by $(-\infty, -100)$ and by a uniform partition of $[-100, 0)$ of step 2. We also use uniform partitions of $[x^2_{\min}, x^2_{\max}]$ and $[x^3_{\min}, x^3_{\max}]$ of step 0.5. The resulting symbolic set Q has 61 200 states, the symbolic set of inputs is $\hat{U} = \{-3, -2, \ldots, 3\}$. The computation time for this example is 3 119 seconds with 13 seconds spent on abstracting the system, 2623 seconds spent on abstracting the specification and 483 seconds spent on controller synthesis.

Next, for the synthesized controller (θ, π), we simulate a trajectory of the closed-loop system S_{cl}. Note that the controller is non-deterministic so we use the following priorities: for θ, pick the first valid input in the ordered list $\{0, 1, -1, 2, -2, 3, -3\}$; for π pick the first valid mode on the ordered list $\{p^1, p^2\}$. Figure 1 depicts the time evolution with respect to $t = kT_0$ of the relative distance x^1, follower and leader velocities x^2, x^3, reference velocity v, control input u and mode p. One can check that initially the follower tracks the desired velocity until the

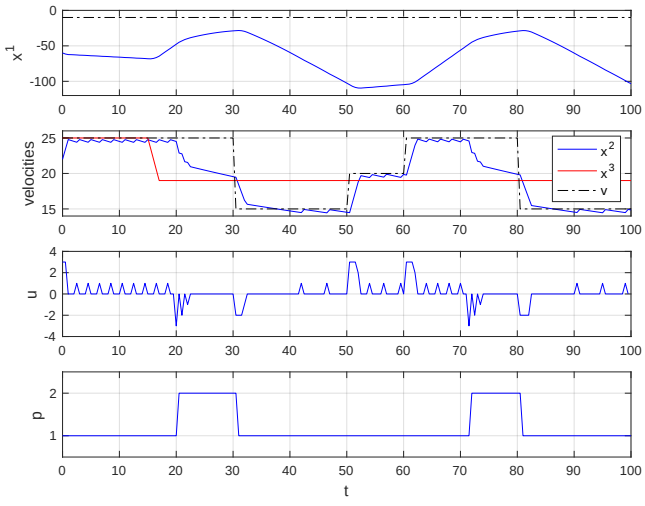

Fig. 1. Adaptive cruise control: relative distance x^1, follower and leader velocities x^2, x^3, reference velocity v, control input u, mode p.

distance between the two vehicles becomes insufficient and the controller moves to mode p^2, reducing the velocity of the follower to avoid collision, when the reference velocity is decreased, the controller moves to mode p^1 and so on. It should be noted that the safety requirement $x_k^1 + l \leq 0$ holds for all k. Let us remark that adaptive cruise control has already been tackled by formal methods, most notably in [14], where the specification is provided by a linear temporal logic formula. In comparison, our specification formalism allows to account for variation of the reference velocity using the input of the hybrid automaton, and to specify the speed at which this target velocity is reached using the parameter C in the dynamics of hybrid automaton.

6.2 Reachability: takeover maneuver

Consider now a similar setting with two vehicles on a road with two lanes where the controllable vehicle is able to perform a takeover maneuver. The state of the system S_1 of two vehicles is given by $x = (x_1, x_2, x_3, x_4)^T$ where x_1, x_2, x_3 have the same meaning as in the previous example and $x_4 \in \{1, 2\}$ corresponds to the lane number of the controlled vehicle. The state space of S_1 is $X = (-\infty, +\infty) \times [x^2_{\min}, x^2_{\max}] \times [x^3_{\min}, x^3_{\max}] \times \{1, 2\}$. The dynamics is given by the following discrete-time equations:

$$\begin{aligned} x^1_{k+1} &= x^1_k + \beta(x^2_k, x^3_k, x^4_k, u^2_k)T_0, \\ x^2_{k+1} &= \chi\left(x^2_k + \alpha(u_k, x^2_k)T_0,\ [x^2_{\min}, x^2_{\max}]\right), \\ x^3_{k+1} &= \chi\left(x^3_k + w_k T_0,\ [x^3_{\min}, x^3_{\max}]\right) \\ x^4_{k+1} &= u^2_k \end{aligned}$$

where T_0, α and χ are the same as before and

$$\beta(x^2, x^3, x^4, u^2) = \begin{cases} x^2 - x^3 & \text{if } x^4 = u^2 \\ \frac{1}{T_0}\sqrt{(x^2 T_0)^2 - r^2} - x^3 & \text{if } x^4 \neq u^2 \end{cases}$$

The second case in β describes a lane-changing maneuver where parameter r denote the distance between lanes. In the following we consider $r = 2$ m and it is assumed for simplicity that this maneuver takes time T_0. The control inputs are $u^1_k \in [u_{\min}, u_{\max}]$, which is the torque applied to the wheels of the controlled vehicle and $u^2_k \in \{1, 2\}$ which selects the lane number of the controlled vehicle at the next time instant. As before $w_k \in [w_{\min}, w_{\max}]$ is the acceleration of the uncontrolled vehicle and is an uncertain input. The numerical values of the parameters are the same as before.

The specification S_2 for the takeover maneuver is described by a discrete-time automaton with 3 modes $P = \{p^1, p^2, p^3\}$ and no external input. In mode p^1, the controlled vehicle follows the uncontrolled vehicle. In mode p^2, the controlled vehicle is in lane 2 and is overtaking the uncontrolled vehicle. In mode p^3, the controlled vehicle leads the uncontrollable vehicle. Formally, the dynamics of S_2 is given by $(x'p') \in G(x, p, v)$ if one of the following condition holds:

(1) $p = p^1$, $p' = p^1$, $x_1 + l \leq 0$, $x_4 = 1$ and $x^{1\prime} \geq x^1$,
(2) $p = p^1$, $p' = p^2$, $x_1 + l \leq 0$, $x_4 = 2$ and $x^{1\prime} \geq x^1$,
(3) $p = p^2$, $p' = p^2$, $x_4 = 2$ and $x^{1\prime} \geq x^1$,
(4) $p = p^2$, $p' = p^3$, $x_1 - l \geq 0$, $x_4 = 2$ and $x^{1\prime} \geq x^1$.

The dynamics does not impose any explicit constraints on $x^{2\prime}$, $x^{3\prime}$ and $x^{4\prime}$ (other than $x^{2\prime} \in [x^2_{\min}, x^2_{\max}]$, $x^{3\prime} \in [x^3_{\min}, x^3_{\max}]$ and $x^{4\prime} \in \{1, 2\}$). The parameters $l > 0$ specifies the safety distance and its numerical value is taken as $l = 10$ m. The condition in all transitions that $x^{1\prime} \geq x^1$ enforces that the relative distance separating the controlled and the uncontrolled vehicle grows continuously. Note that the specification is a piecewise affine hybrid automaton. We then choose an appropriate partition of the state space such that Assumptions 10 and 11 are satisfied. The takeover maneuver ends when the controllable vehicle is in the first lane leading the uncontrollable vehicle with a minimal relative distance of l. This corresponds to the terminal set

$$Z_f = \{(x, p) \mid p = p^3,\ x^1 - l \geq 0, x^4 = 2\}.$$

The maneuver needs to end in finite time, so it is a reachability requirement as in Problem 17.

A controller $\{\theta, \pi\}$ has been synthesized by the approach presented in Section 5.2. The partition of X is obtained using partitions on each subspace. The partition of $(-\infty, +\infty)$ is given by $(-\infty, -70)$, $[30, +\infty)$ and by a uniform partition of $[-70, 30)$ of step 2. We also use uniform partitions of $[x^2_{\min}, x^2_{\max}]$ and $[x^3_{\min}, x^3_{\max}]$ of step 0.5, and $\{1, 2\}$ is kept as it is. The symbolic set Q has 124 800 states, the symbolic set of inputs is $\hat{U} = \{-3, -2, \ldots, 3\}$. The computation time for this example is 1 076 seconds with 61 seconds spent on abstracting the system, 498 seconds spent on abstracting the specification and 517 seconds spent on controller synthesis.

Next, for the synthesized controller (θ, π), we simulate a trajectory of the closed-loop system S_{cl}. Note that the controller is non-deterministic so we use the following priorities: for θ, pick the first valid input in the ordered list $\{0, 1, -1, 2, -2, 3, -3\}$; for π pick the first valid mode on the ordered list $\{p^1, p^2, p^3\}$. Figure 1 depicts the time evolution with respect to $t = kT_0$ of the relative distance x^1, velocities of the controlled and uncontrolled vehicles x^2, x^3, control input u and mode p. One can check that the controllable vehicle is in lane 2 when the overtaking takes place and that x^1 the relative distance between the controlled and the uncontrolled vehicle grows continuously.

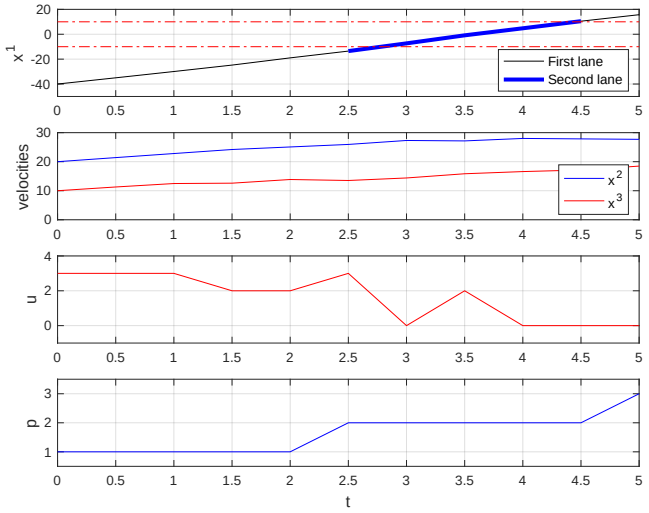

Fig. 2. Takeover maneuver: relative distance x^1, velocities of the controlled and uncontrolled vehicles x^2, x^3, control input u, mode p.

7 Conclusion and future work

In this paper, we presented an approach to formal controller synthesis from specifications given by discrete-time hybrid automata, enriched with safety and reachability requirements. Using illustrations from autonomous vehicle control, we have shown the usefulness of our specification formalism. In addition, we proposed a fully automatic approach to controller synthesis using symbolic abstractions of the system and of the specification. These algorithms are available within the toolbox Co4Pro. In our future work, we will use these two types of problems as building blocks for control programs specifying even more complex behaviors (see e.g. [19]).

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*, 138(1):3–34, 1995.

[2] R. Alur, T. A. Henzinger, O. Kupferman, and M.Y. Vardi. Alternating refinement relations. In *Concurrency Theory*, pages 163–178, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[3] C. Baier and J.-P. Katoen. *Principles of model checking*. MIT press, 2008.

[4] C. Belta, B. Yordanov, and E.A. Gol. *Formal Methods for Discrete-Time Dynamical Systems*. Springer, 2017.

[5] M. Coogan, S.and Arcak. Efficient finite abstraction of mixed monotone systems. In *International Conference on Hybrid Systems: Computation and Control*, pages 58–67. ACM, 2015.

[6] G.E. Fainekos, A. Girard, H. Kress-Gazit, and G.J. Pappas. Temporal logic motion planning for dynamic robots. *Automatica*, 45(2):343–352, 2009.

[7] A. Girard, G. Pola, and P. Tabuada. Approximately bisimilar symbolic models for incrementally stable switched systems. *IEEE Transactions on Automatic Control*, 55(1):116–126, 2009.

[8] E.A. Gol, M. Lazar, and C. Belta. Language-guided controller synthesis for linear systems. *IEEE Transactions on Automatic Control*, 59(5):1163–1176, 2014.

[9] J. Liu and N. Ozay. Finite abstractions with robustness margins for temporal logic-based control synthesis. *Nonlinear Analysis: Hybrid Systems*, 22:1–15, 2016.

[10] J. Liu, N. Ozay, U. Topcu, and R.M. Murray. Synthesis of reactive switching protocols from temporal logic specifications. *IEEE Transactions on Automatic Control*, 58(7):1771–1785, 2013.

[11] X. Liu and S.A. Smolka. Simple linear-time algorithms for minimal fixed points. In *International Colloquium on Automata, Languages, and Programming*, pages 53–66. Springer, 1998.

[12] J. Lygeros, K.H. Johansson, S.N. Simić, J.Zhang, and S.S. Sastry. Dynamical properties of hybrid automata. *IEEE Transactions on Automatic Control*, 48(1):2–17, 2003.

[13] N.Y. Megawati and A. van der Schaft. Abstraction and control by interconnection of linear systems: A geometric approach. *Systems & Control Letters*, 105:27–33, 2017.

[14] P. Nilsson, O. Hussien, A. Balkan, Y. Chen, A.D. Ames, J.W. Grizzle, N. Ozay, H. Peng, and P. Tabuada. Correct-by-construction adaptive cruise control: Two approaches. *IEEE Transactions on Control Systems Technology*, 24(4):1294–1307, 2015.

[15] G. Pola, A. Borri, and M.D. Di Benedetto. Integrated design of symbolic controllers for nonlinear systems. *IEEE Transactions on Automatic Control*, 57(2):534–539, 2011.

[16] G. Pola and M.D. Di Benedetto. Control of cyber-physical-systems with logic specifications: A formal methods approach. *Annual Reviews in Control*, 47:178–192, 2019.

[17] G. Reißig. Computing abstractions of nonlinear systems. *IEEE Transactions on Automatic Control*, 56(11):2583–2598, 2011.

[18] G. Reißig, A. Weber, and M. Rungger. Feedback refinement relations for the synthesis of symbolic controllers. *IEEE Transactions on Automatic Control*, 62(4):1781–1796, 2016.

[19] V. Sinyakov and A. Girard. Formal synthesis from control programs. In *IEEE Conference on Decision and Control*, pages 2138–2145, 2020.

[20] T. Stauner. Discrete-time refinement of hybrid automata. In *Hybrid Systems: Computation and Control*, pages 407–420. Springer, 2002.

[21] P. Tabuada. *Verification and Control of Hybrid Systems: a symbolic approach*. Springer, 2008.

[22] P. Tabuada and G.J. Pappas. Linear time logic control of discrete-time linear systems. *IEEE Transactions on Automatic Control*, 51(12):1862–1877, 2006.

[23] A.J. van der Schaft. Equivalence of dynamical systems by bisimulation. *IEEE Transactions on Automatic Control*, 49(12), 2004.

[24] H. Vinjamoor and A.J. van der Schaft. The achievable dynamics via control by interconnection. *IEEE Transactions on Automatic Control*, 56(5):1110–1117, 2010.

[25] M. Zamani, G. Pola, M. Mazo, and P. Tabuada. Symbolic models for nonlinear control systems without stability assumptions. *IEEE Transactions on Automatic Control*, 57(7):1804–1809, 2012.