Deductive Control Synthesis for Alternating-Time Logics

Rayna Dimitrova
MPI-SWS
rayna@mpi-sws.org

Rupak Majumdar
MPI-SWS
rupak@mpi-sws.org

ABSTRACT

Algorithmic design of control laws for continuous systems for complex temporal specifications is a key step toward automatic synthesis of controllers for cyber-physical systems. Current approaches either abstract the dynamical system to a finite-state approximation or search for certificates that imply invariance or reachability properties (barriers and Lyapunov functions, respectively). The first approach is limited by an exponential blow-up in the abstraction process; the second in the properties that can be controlled for.

We present a deductive proof system for the control of alternating-time temporal properties on continuous systems. We show that reasoning about temporal logic constraints in ATL*, an expressive branching-time logic that allows for quantification over control strategies, can be reduced effectively to reasoning about combinations of barrier certificates and Lyapunov functions. Our approach enables the application of existing constraint-based techniques for finding barriers and Lyapunov functions to the design of controllers for complex temporal properties, while sidestepping the exponential cost of computing finite-state abstractions.

Categories and Subject Descriptors

C.3 [**Special-purpose and application-based systems**]: Real-time and embedded systems; D.2.4 [**Software verification**]: Formal methods

General Terms

Design, Verification

Keywords

Control synthesis, Deductive proof systems, Barrier certificates, Lyapunov functions

1. INTRODUCTION

The automatic synthesis of continuous controllers for complex temporal requirements is a key challenge toward the correct-by-construction design of embedded control systems. Currently, algorithmic synthesis of continuous controllers follows two main routes. In the first, one constructs a finite-state abstraction of a continuous system, using for example an approximate bisimulation or simulation relation, and applies techniques from finite-state reactive synthesis to design a discrete controller, which is then refined to a continuous one [17, 8, 25, 7, 12, 32, 30, 6, 29]. In the second, one sets up a constraint system, for example in the form of a barrier certificate or a control Lyapunov function, whose solution provides the description of a controller satisfying a temporal requirement [19, 18, 26, 9]. For example, constraint-based techniques based on density functions [22] reduce to efficient convex optimization problems and have been used for control synthesis for safety [23] and stability [20].

Unfortunately, both techniques have limitations. While finite-state abstractions enable the synthesis of controllers for expressive ω-regular properties, the finite abstraction is exponential in the size of the system and constructing it explicitly constitutes a computational bottleneck. On the other hand, while constraint-based techniques reduce to efficient semi-definite programming techniques, they are limited so far to invariance properties (barrier certificates) or to eventuality properties (Lyapunov functions). Finally, both approaches have been applied to (subclasses of) linear-time temporal requirements. As we show below, there are interesting properties of systems that cannot be stated in these logics.

In this paper, we present a deductive system for the design of controllers for alternating-time temporal properties. Alternating-time temporal logics (e.g., ATL*) are expressive logics for multi-agent systems [1]. They generalize linear-time temporal logics such as LTL [16] and branching-time temporal logics such as CTL [4]. In contrast to LTL, which assumes universal quantification over trajectories, or CTL, which allows existential or universal quantification, alternating-time logics explicitly model multiple agents, and allow quantification over outcomes of certain games played among teams of agents. For example, an alternating-time logic formula can state *receptiveness* requirements, such as on every path, a controller has a strategy to ensure a liveness property. For example, for a quadrotor, a receptiveness condition might state that along all points of the trajectory, the quadrotor has a strategy to return to its starting point no matter how the environment behaves. It is well known that such properties cannot be expressed in LTL or CTL.

One possible approach to model checking alternating-time logics, which subsumes controller synthesis, is again to con-

ESWEEK '14, October 12 - 17 2014, New Delhi, India

ACM 978-1-4503-3052-7/14/10 ...$15.00.
http://dx.doi.org/10.1145/2656045.2656054

struct a finite-state abstraction and then to apply finite-state model checking algorithms for alternating-time logics. It is known that (approximate) alternating bisimulation [2, 17] characterizes alternating-time logics in the same way that (approximate) bisimulation characterizes linear- or branching-time logics. Moreover, under certain stability assumptions on the dynamics, one can show finite approximate bisimulations exist. However, as mentioned before, this approach requires the exponential computation of the abstraction up front.

Instead, we take the constraint-based approach. We show that control for alternating-time properties expressed in the logic ATL* [1] can be reduced to constraint systems that search for sequences of barrier certificates and Lyapunov functions. Thus, constraint-based techniques such as sum-of-squares techniques [19, 18] or semi-algebraic sets [27, 13, 26], can be applied to model checking alternating-time temporal logic specifications on continuous dynamical systems. Our constraint system is based on deductive proof systems for CTL* and ATL* studied in the context of discrete systems [10, 24]. While our proof system is incomplete (see Section 5.2 for discussions), our proof rules are simple and generalize barriers and Lyapunov functions.

In particular, we describe a certificate for linear-temporal logic conditions, represented as parity conditions. Our construction is based on the notion of progress measures for parity properties [11], and the resulting constraints combine barrier- and Lyapunov-like conditions.

While we describe our technique for continuous-time dynamical systems, our deductive proof system generalizes to hybrid systems.

Other Related Work. By now, as we have discussed above, there is a rich literature in techniques bridging the gap between control theory and reactive synthesis with applications to various domains such as flight control for quadrotors or robotics. The focus in most of this work has been on safety or (repeated) reachability properties, or fragments of LTL that admit efficient synthesis. Our results extend these techniques to alternating-time requirements, and also provide a deductive approach to full LTL properties.

Our proof system builds on existing systems for deductive verification for temporal logics [14, 10, 24], replacing invariants and rank functions in these approaches with their continuous-time analogues. The logical foundations of dynamical systems with respect to alternating-time specifications on finite trajectories has been explored by Platzer [15, 21]. Our contribution is to make the explicit connection between constraint-based techniques and deductive methods for alternating-time logics.

Recently, an approach for LTL verification that combines automata theory and barrier certificates has been presented in [31]. The method in [31] depends heavily on structural constraints on safety automata, and produces an "unknown" result even on simple formulas like $\Diamond p$ whose negation is given as a two state Büchi automaton. In contrast, since we mix barriers and Lyapunov functions, we can find controllers for ATL* specifications.

A template-based approach for the synthesis of switching logic for safety and reachability objectives has been proposed in [28]. While our technique also relies on template polynomials provided by the user and reduces the problem to solving exists-forall constraints, our main contribution is an extension of template-based techniques to handle full ATL*.

2. SYSTEMS AND LOGICS

2.1 Notation

We write $\mathbb{R}$ and $\mathbb{N}$ for the set of real numbers and the set of natural numbers, respectively. We write $\mathbb{R}^n$ for the n-dimensional Euclidean space. For a vector $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$, we write $\|x\|$ for the infinity norm of x, i.e., $\|x\| = \max\{|x_1|, \ldots, |x_n|\}$ where $|x|$ denotes the absolute value of x. For a set $V \subseteq \mathbb{R}^n$ we denote with $\mathrm{cl}(V)$ and ∂V respectively the closure and boundary of V, and with $\overline{V}$ the complement $\mathbb{R}^n \setminus V$ of V. For a set Σ we denote with Σ^ω the set of infinite sequences over Σ and if $\rho = \sigma_0\sigma_1 \ldots \in \Sigma^\omega$ and $i \geq 0$, we define $\rho[i] = \sigma_i$ and $\rho[i, \infty) = \sigma_i\sigma_{i+1} \ldots$.

Given a measurable function $f : \mathbb{R}_0^+ \to \mathbb{R}^n$, the (essential) supremum of f is denoted by $\|f\|_\infty$; we recall that $\|f\|_\infty := (\mathrm{ess})\sup\{\|f(t)\|, t \geq 0\}$ and $\|f\|_{[0,\tau)} := (\mathrm{ess})\sup\{\|f(t)\|, t \in [0, \tau)\}$. A function f is essentially bounded if $\|f\|_\infty < \infty$. For a given time $\tau \in \mathbb{R}^+$, define f_τ so that $f_\tau(t) = f(t)$, for any $t \in [0, \tau)$, and $f_\tau(t) = 0$ elsewhere; f is said to be locally essentially bounded if for any $\tau \in \mathbb{R}^+$, f_τ is essentially bounded. A function $f : \mathbb{R}^n \to \mathbb{R}_0^+$ is called radially unbounded if $f(x) \to \infty$ as $\|x\| \to \infty$.

2.2 Continuous systems

A *continuous system* is a tuple $\mathcal{C} = (\mathbb{R}^n, W, \mathcal{W}, f)$, where

- $\mathbb{R}^n$ is the state space;
- $W = U \times D$ is the input space, where $U \subseteq \mathbb{R}^m$ is the space of control inputs and $D \subseteq \mathbb{R}^d$ is the space of disturbance inputs;
- $\mathcal{W} = \mathcal{U} \times \mathcal{D}$ is a subset of the set of all measurable and locally essentially bounded functions of time from intervals of the form $(a, b) \subseteq \mathbb{R}$ to W with $a < 0$ and $b > 0$;
- $f : \mathbb{R}^n \times W \to \mathbb{R}^n$ is a continuous *flow function* satisfying the following Lipschitz assumption: for every compact $K \subseteq \mathbb{R}^n$, there exists a constant $L > 0$ such that
$$\|f(x, w) - f(y, w)\| \leq L\|x - y\|$$
for all $x, y \in K$ and all $w \in W$.

An absolutely continuous curve $\mathbf{x} : (a, b) \to \mathbb{R}^n$ is a *trajectory* of $\mathcal{C}$ if there exists $\mathbf{w} \in \mathcal{W}$ such that
$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{w}(t))$$
for almost all $t \in (a, b)$. We say that $\mathbf{x}$ is *compatible* with $\mathbf{u}$ and $\mathbf{v}$, where $\mathbf{w} = (\mathbf{u}, \mathbf{v})$. While trajectories are defined over open intervals, we will refer to trajectories $\mathbf{x} : [0, \tau] \to \mathbb{R}^n$ defined on closed domains $[0, \tau]$ for some $\tau > 0$, with the understanding that there is a trajectory $\mathbf{z} : (a, b) \to \mathbb{R}^n$, for $a < 0$ and $b > \tau$, such that $\mathbf{x}$ is the restriction of $\mathbf{z}$ to $[0, \tau]$, i.e., $\mathbf{x}(t) = \mathbf{z}(t)$ for $t \in [0, \tau]$. We write $\mathbf{x}(t, x_0, \mathbf{u}, \mathbf{v})$ to denote the state reached at time $t \in (a, b)$ under the input $(\mathbf{u}, \mathbf{v}) \in \mathcal{W}$ from the initial state x_0. Notice that this state is uniquely defined, since the assumptions on f ensure existence and uniqueness of trajectories.

In the following, we assume that the continuous systems are *forward complete*, that is, every trajectory is defined

on an interval of the form (a, ∞). Sufficient and necessary conditions for a continuous system to be forward complete can be found in [3].

2.3 Strategies

We can interpret the interaction between the control and the disturbance in a continuous system $\mathcal{C} = (\mathbb{R}^n, W, \mathcal{W}, f)$ as a *two-player game*. A *strategy* for the control is a function $\mathbf{u} \in \mathcal{U}$ and a strategy for the disturbance is a function $\mathbf{v} \in \mathcal{D}$.

Suppose that F is a set of functions such that $F = \emptyset$, or $F = \{\mathbf{u}\}$, or $F = \{\mathbf{v}\}$ or $F = \{\mathbf{u}, \mathbf{v}\}$, where $\mathbf{u} \in \mathcal{U}$ and $\mathbf{v} \in \mathcal{D}$. Then, given a set $I \subseteq \mathbb{R}^n$ of states, the set $\mathsf{Trajectories}(\mathcal{C}, I, F)$ consists of all trajectories $\mathbf{x} : [0, \infty) \to \mathbb{R}^n$ such that $\mathbf{x}(0) \in I$ and if $\mathbf{u} \in F \cap \mathcal{U}$, then $\mathbf{x}$ is compatible with $\mathbf{u}$, and if $\mathbf{v} \in F \cap \mathcal{D}$, then $\mathbf{x}$ is compatible with $\mathbf{v}$. We write $\mathsf{Trajectories}(\mathcal{C}, I)$ instead of $\mathsf{Trajectories}(\mathcal{C}, I, \emptyset)$.

2.4 Alternating-time temporal logic

We use an alternating time temporal logic to specify properties of continuous systems. More specifically, we use Alternating-time Temporal Logic (ATL*) [1] without the next operator. For readability, we denote the logic (without the next operator) as ATL* rather than ATL$^* \setminus \mathsf{X}$; the logic in [1] has an additional next operator.

2.4.1 Syntax

Let $\mathcal{P}$ be a set of propositions, where each proposition $p \in \mathcal{P}$ defines a subset $[\![p]\!] \subseteq \mathbb{R}^n$. We assume that for each $p \in \mathcal{P}$, there is a dual $\bar{p} \in \mathcal{P}$ such that $[\![\bar{p}]\!] = \mathbb{R}^n \setminus [\![p]\!]$.

Let $Agts = \{\mathsf{u}, \mathsf{d}\}$ be the set of *control* and *disturbance* agents.[1] The formulas of ATL* are of two types: *state formulas* and *path formulas*. The set of state formulas is defined by the following grammar, where $p \in \mathcal{P}$ is a proposition, Φ_1 and Φ_2 are state formulas, $A \subseteq Agts$, and φ is a path formula:

$$\Phi \;=\; p \mid \neg p \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid [\![A]\!]\varphi \mid \langle\!\langle A \rangle\!\rangle \varphi.$$

The set of path formulas is defined by the grammar below, where Φ is a state formula, and φ_1 and φ_2 are path formulas:

$$\varphi = \Phi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \,\mathcal{U}\, \varphi_2 \mid \varphi_1 \,\mathcal{W}\, \varphi_2.$$

We define $true = p \vee \neg p$ for some $p \in \mathcal{P}$ and $false = \neg true$, and the usual additional temporal operators *globally* $\Box\varphi = \varphi \,\mathcal{W}\, false$ and *finally* $\Diamond\varphi = true \,\mathcal{U}\, \varphi$. We write $\langle\!\langle\rangle\!\rangle$ and $[\![]\!]$ instead of $\langle\!\langle\emptyset\rangle\!\rangle$ and $[\![\emptyset]\!]$ respectively.

The linear time temporal logic LTL consists of those path formulas that do not contain the quantifiers $\langle\!\langle\cdot\rangle\!\rangle$ and $[\![\cdot]\!]$. A state formula Φ is called *basic state formula* iff it is of the form $\Phi = \langle\!\langle A \rangle\!\rangle \varphi$ or $\Phi = [\![A]\!]\varphi$ where φ is an LTL formula.

For a path formula φ, we denote with $\mathsf{StateForm}(\varphi)$ the set of state formulas occurring as subformulas of φ. Given formulas ψ and ψ', $\varphi[\psi'/\psi]$ is the formula obtained from φ by replacing each occurrence of ψ in φ with ψ'.

2.4.2 Traces

Let $\Sigma \subseteq 2^{\mathbb{R}^n}$ be a set of subsets of $\mathbb{R}^n$, $\mathcal{C} = (\mathbb{R}^n, W, \mathcal{W}, f)$ be a continuous system and $\mathbf{x}$ be a trajectory of $\mathcal{C}$. A *trace* for $\mathbf{x}$ over Σ is an infinite sequence $\sigma_0, \sigma_1, \ldots \in \Sigma^\omega$ such that there exists an infinite sequence of time points $\tau_0, \tau_1, \ldots$ for which it holds that:

- $\tau_0 = 0$, $\tau_i < \tau_{i+1}$, and τ_i goes to ∞ as i goes to ∞,
- for each $i \in \mathbb{N}$ there exists $\tau \in [\tau_i, \tau_i + 1]$ such that:
  - for all $t \in [\tau_i, \tau)$ we have $\mathbf{x}(t) \in \sigma_i$,
  - for all $t \in (\tau, \tau_{i+1}]$ we have $\mathbf{x}(t) \in \sigma_{i+1}$,
  - for some $\sigma \in \{\sigma_i, \sigma_{i+1}\}$ we have $\mathbf{x}(\tau) \in \sigma$.

We write $\mathsf{Traces}(\mathbf{x}, \Sigma)$ for the set of traces for $\mathbf{x}$ over Σ.

2.4.3 Semantics

Let $\mathcal{C} = (\mathbb{R}^n, W, \mathcal{W}, f)$ be a continuous system. We interpret ATL* state formulas over the states of the system and ATL* path formulas over (traces of) trajectories. The respective satisfaction relations $\models_{\mathcal{C}}$ are defined below. For a state formula Φ we denote $[\![\Phi]\!] = \{x \in \mathbb{R}^n \mid x \models_{\mathcal{C}} \Phi\}$.

For every path formula φ we define the set

$$\Sigma_\varphi = \Big\{\sigma \mid \exists \Pi \subseteq \mathsf{StateForm}(\varphi).\ \sigma = \bigcap_{\Phi \in \Pi} [\![\Phi]\!] \cap \bigcap_{\Phi \notin \Pi} \overline{[\![\Phi]\!]}\Big\}.$$

- If $\Phi = p$, then $x \models_{\mathcal{C}} \Phi$ iff $x \in [\![p]\!]$.
- If $\Phi = \neg\Phi'$, then $x \models_{\mathcal{C}} \Phi$ iff $x \not\models_{\mathcal{C}} \Phi'$.
- If $\Phi = \Phi_1 \wedge \Phi_2$, then $x \models_{\mathcal{C}} \Phi$ iff $x \models_{\mathcal{C}} \Phi_1$ and $x \models_{\mathcal{C}} \Phi_2$.
- If $\Phi = \Phi_1 \vee \Phi_2$, then $x \models_{\mathcal{C}} \Phi$ iff $x \models_{\mathcal{C}} \Phi_1$ or $x \models_{\mathcal{C}} \Phi_2$.
- If $\Phi = [\![A]\!]\varphi$, then $x \models_{\mathcal{C}} \Phi$ iff for all sets of strategies $\{f_a\}_{a \in A}$ for the agents in A it holds that there exists $\mathbf{x} \in \mathsf{Trajectories}(\mathcal{C}, \{x\}, \{f_a\}_{a \in A})$ such that $\mathbf{x} \models_{\mathcal{C}} \varphi$.
- If $\Phi = \langle\!\langle A \rangle\!\rangle \varphi$, then $x \models_{\mathcal{C}} \Phi$ iff there exists a set of strategies $\{f_a\}_{a \in A}$ for the agents in A, such that $\mathbf{x} \models_{\mathcal{C}} \varphi$ for all trajectories $\mathbf{x} \in \mathsf{Trajectories}(\mathcal{C}, \{x\}, \{f_a\}_{a \in A})$.
- $\mathbf{x} \models_{\mathcal{C}} \varphi$ iff $\rho \models_{\mathcal{C}} \varphi$ for all $\rho \in \mathsf{Traces}(\mathbf{x}, \Sigma_\varphi)$.
- If $\varphi = \Phi$, then $\rho \models_{\mathcal{C}} \varphi$ iff $\rho[0] \subseteq [\![\Phi]\!]$.
- If $\varphi = \varphi_1 \vee \varphi_2$, then $\rho \models_{\mathcal{C}} \varphi$ iff $\rho \models_{\mathcal{C}} \varphi_1$ or $\rho \models_{\mathcal{C}} \varphi_2$.
- If $\varphi = \varphi_1 \wedge \varphi_2$, then $\rho \models_{\mathcal{C}} \varphi$ iff $\rho \models_{\mathcal{C}} \varphi_1$ and $\rho \models_{\mathcal{C}} \varphi_2$.
- If $\varphi = \varphi_1 \,\mathcal{U}\, \varphi_2$, then $\rho \models_{\mathcal{C}} \varphi$ iff there exists $i \geq 0$ such that $\rho[i, \infty) \models_{\mathcal{C}} \varphi_2$ and $\rho[j, \infty) \models_{\mathcal{C}} \varphi_1$ for all $0 \leq j < i$.
- If $\varphi = \varphi_1 \,\mathcal{W}\, \varphi_2$, then $\rho \models_{\mathcal{C}} \varphi$ iff $\rho \models_{\mathcal{C}} \varphi_1 \,\mathcal{U}\, \varphi_2$ or $\rho[i, \infty) \models_{\mathcal{C}} \varphi_1$ for all positions $i \geq 0$.

The *ATL* verification problem* asks, given a continuous system $\mathcal{C}$, a state x, and an ATL* state formula Φ, if $x \models_{\mathcal{C}} \Phi$.

EXAMPLE 1. *As a simple example, consider an ATL* formula describing the desired behavior of a quadrotor. In this case, the set Agts contains four control agents, one for each motor. That is, $Agts = A_u \cup \{\mathsf{d}\}$, where $A_u = \{\mathsf{u}_1, \mathsf{u}_2, \mathsf{u}_3, \mathsf{u}_4\}$. The specification states that the coalition of all controllers has a strategy that forces every trajectory of the quadrotor to reach a state within some safe altitude range, and to fulfil an additional requirement that depends on whether or not a fault occurs in some rotor. Formally,*

$$\Phi = \langle\!\langle A_u \rangle\!\rangle \Diamond \big(safe \wedge \varphi_{correct} \wedge \varphi_{faulty}\big).$$

[1] The logic in [1] is parameterized over arbitrary sets of agents. We can extend our logic to the same setting, by explicitly defining inputs for each agent in the definition of continuous systems, and considering $U = U_1 \times \ldots U_k$ if there are k agents. We restrict our logic to control and disturbance agents to keep the notation simple.

In the case when no fault occurs, the trajectory must remain safe and eventually reach a given goal region:

$$\varphi_{correct} = \big(\Box \bigwedge_{i=1}^{4} \overline{faulty_i}\big) \rightarrow \big(\Box\, safe \wedge \Diamond\, goal\big).$$

Otherwise, the system is required to tolerate one faulty rotor. In our example this means that the remaining three rotors should have a strategy to keep the vehicle within the safe range until it is lowered in preparation for landing:

$$\varphi_{faulty} = \Box\Big(\bigwedge_{i=1}^{4} \big(faulty_i \rightarrow \langle\!\langle A_u \setminus \{u_i\}\rangle\!\rangle (safe\, \mathcal{U}\, low)\big)\Big).$$

The propositions safe, goal and low are defined in terms of the position $(X, Y, Z)^T$ of the center of gravity of the quadrotor. We model fault occurrence as part of the disturbance input and thus, the proposition $faulty_i$ for $i \in \{1, \ldots, 4\}$ is defined in terms of the disturbance signal D_i. Formally:

$$\begin{array}{lcl} safe & = & \underline{z}_s \leq Z \wedge Z \leq \overline{z}_s, \\ low & = & \underline{z}_l \leq Z \wedge Z \leq \overline{z}_l, \\ goal & = & \underline{x}_g \leq X \wedge X \leq \overline{x}_g \wedge \underline{y}_g \leq Y \wedge Y \leq \overline{y}_g, \\ faulty_i & = & \underline{d}_i \leq D_i \wedge D_i \leq \overline{d}_i, \end{array}$$

where $\underline{z}_s, \overline{z}_s, \underline{z}_l, \overline{z}_l, \underline{x}_g, \overline{x}_g, \underline{y}_g, \overline{y}_g$ are real constants defining the sets of safe, low and goal positions of the quadrotor. For each $i \in \{1, \ldots, 4\}$, $\underline{d}_i$ and $\overline{d}_i$ are real constants defining the range of the disturbance signal that indicates a fault.

The system state x contains components for the position of the quadcopter in space, its linear and angular velocity, the roll, pitch and yaw angles. The control inputs are the angular velocities of the four rotors. The dynamics can be described by a differential equation of the form $\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{w}(t))$, which we omit here.

Notice that ATL* subsumes the branching time logic CTL* (an existential path quantifier $\exists\varphi$ is equivalent to $\langle\!\langle \mathsf{u}, \mathsf{d}\rangle\!\rangle\varphi$ and a universal path quantifier $\forall\varphi$ is equivalent to $\langle\!\langle\rangle\!\rangle\varphi$).

2.4.4 Automata for LTL formulas

A *deterministic parity automaton* is a tuple $\mathcal{A} = (L, \Sigma, \delta, l_0, C, \lambda)$, where L is a finite set of states, Σ is an input alphabet, $\delta : L \times \Sigma \rightarrow L$ is a transition relation, $l_0 \in L$ is the initial state, $C \subseteq N$ is a finite set of colors and $\lambda : L \rightarrow C$ is a function that assigns a color to each state. A run of $\mathcal{A}$ on an infinite word $\omega \in \Sigma^\omega$ is a sequence $\eta \in L^\omega$ of states such that $\eta[0] = l_0$ and $\eta[i+1] = \delta(\eta[i], \sigma[i])$ for all $i \geq 0$. For a deterministic automaton $\mathcal{A}$, there is at most one run of $\mathcal{A}$ on a given word ω. A run η on ω is accepting if $\min\{k \in C \mid \forall i \geq 0. \exists i' > i.\ \lambda(\eta[i']) = k\}$, i.e., the minimal color occurring infinitely often, is even. A word ω is accepted by $\mathcal{A}$ iff there exists an accepting run of $\mathcal{A}$ on ω. We consider w.l.o.g. automata with total transition function, i.e. there is an infinite run on each infinite word.

Given an LTL formula φ we can construct a deterministic parity automaton $\mathcal{A}_\varphi = (L, 2^{\mathsf{StateForm}(\varphi)}, \delta, l_0, C, \lambda)$, such that a word $\omega \in \left(2^{\mathsf{StateForm}(\varphi)}\right)^\omega$ is accepted by $\mathcal{A}$ iff $\rho \models_{\mathcal{C}} \varphi$, where $\rho \in \Sigma_\varphi^\omega$ is such that $\rho[i] = \bigcap_{\Phi \in \omega[i]} [\![\Phi]\!] \cap \bigcap_{\Phi \notin \omega[i]} \overline{[\![\Phi]\!]}$ for each $i \geq 0$. Note that each $\rho \in \Sigma_\varphi$ also uniquely defines a word $\omega \in \left(2^{\mathsf{StateForm}(\varphi)}\right)^\omega$ with this correspondence. We thus sometimes use $\mathsf{Traces}(\mathbf{x}, 2^{\mathsf{StateForm}(\varphi)})$ instead of $\mathsf{Traces}(\mathbf{x}, \Sigma_\varphi)$.

3. CERTIFICATES

In this section we present the concepts of *barrier certificates*, *Lyapunov-like* functions, and *certificates for parity conditions* used for proving respectively *safety*, *eventuality*, and *parity* properties of continuous systems. While barrier certificates and Lyapunov-like functions are standard tools for analysis of dynamical systems, we show how they can be generalized to give certificates for parity conditions, which allow reasoning about general LTL specifications.

We first present the notions which provide their respective guarantees for the system's trajectories resulting from *all possible values of the control and disturbance inputs* in the sets U and V respectively. Then, we adapt the definitions to control synthesis, namely when the set of trajectories results from a given control (respectively disturbance) function.

Let $\mathcal{C} = (\mathbb{R}^n, W, \mathcal{W}, f)$ be a given continuous system.

3.1 Barrier certificates

Barrier certificates were introduced in [19] for safety verification of hybrid systems. The definition we present here is slightly more general than the original one and is useful for establishing weak until properties. The notion is parameterized by a set of states R, such that the safety requirement we are interested in is released if a state in R is reached.

A *barrier certificate* for $\mathcal{C}$ and the sets of states $Init \subseteq \mathbb{R}^n$, $Err \subseteq \mathbb{R}^n$ and $R \subseteq \mathbb{R}^n$ is a continuously differentiable function $B : \mathbb{R}^n \rightarrow \mathbb{R}$ that satisfies the following conditions:

1. $B(x) \leq 0$ for all $x \in Init$,
2. $B(x) > 0$ for all $x \in Err$,
3. $\frac{\partial B}{\partial x}(x) f(x, w) < 0$ for all $x \in \mathbb{R}^n$ and $w \in W$ such that $B(x) = 0$ and $x \notin R$.

3.2 Lyapunov functions

Lyapunov functions are a classical tool for reasoning about stability of continuous and hybrid systems. The definition we employ here for reasoning about the until operator is close to the one used in [18] for proving eventuality.

A *Lyapunov function* for $\mathcal{C}$ and the sets of states $S \subseteq \mathbb{R}^n$, $Init \subseteq S$, and $Target \subseteq S$ is a continuously differentiable function $L : \mathbb{R}^n \rightarrow \mathbb{R}$ such that there exists $\epsilon \in \mathbb{R}^+$ where:

1. $L(x) \leq 0$ for all $x \in Init$,
2. $L(x) > 0$ for all $x \in \mathrm{cl}(\partial S \setminus \partial\, Target)$,
3. $\frac{\partial L}{\partial x}(x) f(x, w) \leq -\epsilon$ for all $x \in S \setminus Target$ and $w \in W$.

3.3 Certificate functions for parity conditions

We introduce certificates for parity conditions in order to be able to establish that the traces generated by a given continuous system are all accepted by a given parity automaton. Unlike barrier certificates and Lyapunov functions, these are tuples $\langle V_0, V_1, \ldots V_c\rangle$ of $c{+}1$ functions, where c is the number of colors in the acceptance condition of the parity automaton. The function V_0 plays the role of a barrier certificate and guarantees that a given set of states is never left. The function V_k for $k \in \{1, \ldots, c\}$ is similar to a Lyapunov function for the k-th color. For an odd color k, the properties of the function V_k guarantee that the set of states with that color is left in finite time. As the value of such a function cannot increase when visiting states with higher color, we get that the minimal color visited infinitely often is even.

Let $P_1, \ldots, P_c \subseteq \mathbb{R}^n$ be sets of states such that $\bigcup_{k=1}^{c} P_k = \mathbb{R}^n$. A *parity certificate* $\langle V_0, V_1, \ldots V_c \rangle$ for $\mathcal{C}$, the sets of states $S \subseteq \mathbb{R}^n$ and $Init \subseteq S$ and the sets $P_1, \ldots, P_c$ consists of $c+1$ continuously differentiable functions $V_k : \mathbb{R}^n \to \mathbb{R}$ for $k = 0, \ldots, c$, for which there exists $\epsilon \in \mathbb{R}^+$ such that:

1. $V_0(x) \leq 0$ for all $x \in Init$,
2. $V_0(x) > 0$ for all $x \in \mathbb{R}^n \setminus S$,
3. $\frac{\partial V_0}{\partial x}(x) f(x,w) < 0$ for $(x,w) \in \mathbb{R}^n \times W$ with $V_0(x) = 0$,
4. for all $i \in \{1, \ldots, c\}$, $x \in P_i \cap S$ and $w \in W$:
   (a) for all $j \leq i$ it holds that $\frac{\partial V_j}{\partial x}(x) f(x,w) \leq 0$,
   (b) if i is odd, $\frac{\partial V_j}{\partial x}(x) f(x,w) \leq -\epsilon$ for some $j \leq i$.

3.4 Certificates for control

When the goal is to show that there exists a strategy for one of the agents (coalition of agents) that enforces some *universal* property against all possible strategies of the remaining agents, it suffices to assume that the choices of the other agents do not have to follow particular strategies. In this case we perform simultaneous search for a strategy function and certificate function(s) with the respective guarantees. As is commonly done, we search for control functions in state feedback form. That is, we suppose that $\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{v}(t))$, where $f(x,v) = h(x,v) + g(x,v)\tilde{\mathbf{u}}(x)$ and look for a control function $\mathbf{u}(t) = \tilde{\mathbf{u}}(\mathbf{x}(t))$ that is in $\mathcal{U}$.

Let $A \subseteq Agts$ be a set of agents, and $\{f_a\}_{a \in A}$ be strategies for the agents in A. We denote with $W_{\overline{A}}$ the set of inputs of $\mathcal{C}$ that belong to the agents in $Agts \setminus A$. Furthermore, we let $f_{\overline{A}}(x, w_{\overline{A}}) = f(x, w_{\overline{A}}, w_A)$ for each $x \in \mathbb{R}^n$, each $w_{\overline{A}} \in W_{\overline{A}}$ and w_A consisting of the values of $f_a(x)$ for $a \in A$.

3.4.1 Barrier certificates for control

Given a set $A \subseteq Agts$ of agents and sets of states $Init \subseteq \mathbb{R}^n$, $Err \subseteq \mathbb{R}^n$ and $R \subseteq \mathbb{R}^n$, the set $\mathsf{Barrier}_A(\mathcal{C}, Init, Err, R)$ consists of all tuples $(\{f_a\}_{a \in A}, B)$, where the functions f_a are strategies for the agents in A, and $B : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function that satisfies conditions 1 and 2 from the definition in Section 3.1 as well as condition 3' below:

3' $\frac{\partial B}{\partial x}(x) f_{\overline{A}}(x, w_{\overline{A}}) < 0$ for all $x \in \mathbb{R}^n$ and $w_{\overline{A}} \in W_{\overline{A}}$ such that $B(x) = 0$ and $x \notin R$.

PROPOSITION 1. *Let $\mathcal{C} = (\mathbb{R}^n, W, \mathcal{W}, f)$ be a continuous system and $Init \subseteq \mathbb{R}^n$, $Err \subseteq \mathbb{R}^n$ and $R \subseteq \mathbb{R}^n$. Suppose that $(\{f_a\}_{a \in A}, B) \in \mathsf{Barrier}_A(\mathcal{C}, Init, Err, R)$ for some $A \subseteq Agts$. Then, for every trajectory $\mathbf{x} \in \mathsf{Trajectories}(\mathcal{C}, \{x \in \mathbb{R}^n \mid B(x) \leq 0\}, \{f_a\}_{a \in A})$ it holds that $\mathbf{x}$ stays forever in the set of states $\overline{Err}$, or it stays there until it reaches the set R.*

PROOF. Assume that $\mathbf{x} \in \mathsf{Trajectories}(\mathcal{C}, \{x \in \mathbb{R}^n \mid B(x) \leq 0\}, \{f_a\}_{a \in A})$ is a trajectory that does not satisfy the claim. That is, there exists $T \in \mathbb{R}_0^+$ such that $\mathbf{x}(T) \in Err$ and for each $t \in [0,T]$ it holds that $\mathbf{x}(t) \notin R$.

By the choice of $\mathbf{x}$, $B(\mathbf{x}(0)) \leq 0$. Since for every $t \in [0,T]$ $\mathbf{x}(t) \notin R$, the third condition of the definition of barrier certificate implies that for every $t \in [0,T]$ it holds that $B(\mathbf{x}(t)) \leq 0$. This contradicts $B(\mathbf{x}(T)) > 0$, which is implied by the second condition of the definition. □

3.4.2 Lyapunov functions for control

Given a set $A \subseteq Agts$ of agents and sets of states $S \subseteq \mathbb{R}^n$, $Init \subseteq S$, and $Target \subseteq S$, the set $\mathsf{Lyapunov}_A(\mathcal{C}, Init, S, Target)$ consists of all tuples $(\{f_a\}_{a \in A}, L)$, where the functions f_a are strategies for the agents in A, and $L : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function for which there exists $\epsilon \in \mathbb{R}^+$ such that conditions 1 and 2 in Section 3.2 and 3' below are satisfied:

3' $\frac{\partial L}{\partial x}(x) f(x, w_{\overline{A}}) \leq -\epsilon$ for $x \in S \setminus Target$ and $w_{\overline{A}} \in W_{\overline{A}}$.

PROPOSITION 2. *Let $\mathcal{C} = (\mathbb{R}^n, W, \mathcal{W}, f)$ be a continuous system and $S \subseteq \mathbb{R}^n$, $Init \subseteq S$ and $Target \subseteq S$. Suppose that the set S is bounded and $(\{f_a\}_{a \in A}, L) \in \mathsf{Lyapunov}_A(\mathcal{C}, Init, S, Target)$ for some $A \subseteq Agts$. Then, every $\mathbf{x} \in \mathsf{Trajectories}(\mathcal{C}, \{x \in S \mid L(x) \leq 0\}, \{f_a\}_{a \in A})$ eventually reaches the set Target and stays in S until then.*

PROOF. Assume that $\mathbf{x} \in \mathsf{Trajectories}(\mathcal{C}, \{x \in S \mid L(x) \leq 0\}, \{f_a\}_{a \in A})$ is a trajectory that does not satisfy the claim. That is, either $\mathbf{x}$ never exits the set $S \setminus Target$, or it exits the set S without entering the set $Target$.

Since the set $\mathrm{cl}(S \setminus Target)$ is closed and bounded, the continuous function L is bounded from below on this set. According to the third requirement in the definition, the function $\frac{\partial L}{\partial x}(x) f(x,d)$ has a maximum, which is a negative number. Therefore, the value of L along $\mathbf{x}$ must go to minus infinity which contradicts the fact that L is bounded from below in $\mathrm{cl}(S \setminus Target)$. Hence, $\mathbf{x}$ leaves $S \setminus Target$ in finite time.

If we now assume that $\mathbf{x}$ leaves S without entering $Target$, this means that there exists $T \in \mathbb{R}_0^+$ such that $\mathbf{x}(T) \notin Target$ and one of the following holds:

- $\mathbf{x}(T) \notin S$ and for each $t \in [0,T)$, $\mathbf{x}(t) \in S \setminus Target$,
- for each $t \in [0,T]$, $\mathbf{x}(t) \in S \setminus Target$, and there exists $\epsilon' > 0$ where for each $0 < \epsilon'' \leq \epsilon'$, $\mathbf{x}(T + \epsilon'') \notin S$.

The second condition of the definition of Lyapunov function implies that $L(\mathbf{x}(T)) > 0$. From the fact that $L(\mathbf{x}(0)) \leq 0$ and the third condition in the definition, it follows that $L(\mathbf{x}(T)) \leq 0$, which is a contradiction. □

3.4.3 Parity certificates for control

Given $A \subseteq Agts$, sets of states $P_1, \ldots, P_c \subseteq \mathbb{R}^n$ with $\bigcup_{k=1}^{c} P_k = \mathbb{R}^n$, and sets of states $S \subseteq \mathbb{R}^n$ and $Init \subseteq S$, the set $\mathsf{Parity}_A(\mathcal{C}, Init, S, P_1, \ldots, P_c)$ consists of all tuples $(\{f_a\}_{a \in A}, \langle V_0, V_1, \ldots V_c \rangle)$, where the functions f_a are strategies for the agents in A, and $V_k : \mathbb{R}^n \to \mathbb{R}$ for $k = 0, \ldots, c$ are continuously differentiable functions for which there exists $\epsilon \in \mathbb{R}^+$ such that conditions 1 and 2 in Section 3.3 and conditions 3' and 4' below are satisfied:

3' $\frac{\partial V_0}{\partial x}(x) f(x, w_{\overline{A}}) < 0$ for $(x, w_{\overline{A}}) \in \mathbb{R}^n \times W_{\overline{A}}$, $V_0(x) = 0$,

4' for all $i \in \{1, \ldots, c\}$, $x \in P_i \cap S$ and $w_{\overline{A}} \in W_{\overline{A}}$:
   1. for all $j \leq i$ it holds that $\frac{\partial V_j}{\partial x}(x) f(x, w_{\overline{A}}) \leq 0$,
   2. if i is odd, $\frac{\partial V_j}{\partial x}(x) f(x, w_{\overline{A}}) \leq -\epsilon$ for some $j \leq i$.

Given a set of states $I \subseteq \mathbb{R}^n$, a deterministic parity automaton $\mathcal{A} = (L, \Sigma, \delta, l_0, C, \lambda)$, and a state $x \in \mathbb{R}^n$ we define the set $\mathsf{Colors}(\mathcal{C}, I, \mathcal{A}, x)$ of colors in C associated with x as follows. For any $k \in C$ we let $k \in \mathsf{Colors}(\mathcal{C}, I, \mathcal{A}, x)$ iff there exist $\mathbf{x} \in \mathsf{Trajectories}(\mathcal{C}, I)$, $\rho \in \mathsf{Traces}(\mathbf{x}, \Sigma)$ and a function $\alpha : \mathbb{R}_0^+ \to \mathbb{N}$, that satisfy the following conditions:

- $\alpha(0) = 0$, $\alpha(t_1) \leq \alpha(t_2)$ for $t_1 \leq t_2$, $\alpha(t) \underset{t \to +\infty}{\longrightarrow} +\infty$,
- for every $t \in \mathbb{R}_0^+$ it holds that $\mathbf{x}(t) \in \rho[\alpha(t)]$,
- $x \in I$ and $\lambda(l_0) = k$, or there exist $t \in \mathbb{R}_0^+$ such that $\mathbf{x}(t) = x$ and $\lambda(\eta[\alpha(t)+1]) = k$, for the run η on ρ.

Intuitively, for each $k \in C$, the set $S_k = \{x \in \mathbb{R}^n \mid k \in \mathsf{Colors}(C, I, \mathcal{A}, x)\}$ consists of the states of $\mathcal{C}$ that can have color k associated with them in the composition of the continuous system $\mathcal{C}$ and the automaton $\mathcal{A}$.

PROPOSITION 3. *Let $\mathcal{C} = (\mathbb{R}^n, W, \mathcal{W}, f)$ be a continuous system and let $S \subseteq \mathbb{R}^n$ and $Init \subseteq S$. Let $\mathcal{A} = (L, \Sigma, \delta, l_0, C, \lambda)$ be a deterministic parity automaton and let $p_1, \ldots, p_c$ be propositions such that for each $x \in \mathbb{R}^n$ and each $k \in \{1, \ldots, c\}$, if $k \in \mathsf{Colors}(\mathcal{C}, Init, \mathcal{A}, x)$, then $x \models p_k$. Suppose that for some $A \subseteq Agts$, $(\{f_a\}_{a \in A}, \langle V_0, \ldots, V_c\rangle) \in \mathsf{Parity}_A(\mathcal{C}, Init, S, P_1, \ldots, P_c)$, where $P_k = [\![p_k]\!]$ for each $k \in \{1, \ldots, c\}$. Then, if the set S is bounded it holds that for every trajectory $\mathbf{x} \in \mathsf{Trajectories}(\mathcal{C}, Init, \{f_a\}_{a \in A})$, every trace $\rho \in \mathsf{Traces}(\mathbf{x}, \Sigma)$ is accepted by $\mathcal{A}$.*

PROOF. Assume that $\mathbf{x} \in \mathsf{Trajectories}(\mathcal{C}, Init, \{f_a\}_{a \in A})$ is a trajectory that violates the claim. That is, there exists a trace $\rho \in \mathsf{Traces}(\mathbf{x}, \Sigma)$ such that for the run η of $\mathcal{A}$ on ρ it holds that the number $m = \min\{k \in \{1, \ldots, c\} \mid \forall j \in \mathbb{N}. \exists j' \in \mathbb{N}. j' > j \wedge \lambda(\eta[j']) = k\}$ is odd.

Using the reasoning from the proof of Proposition 1 and the requirements on the function V_0 we can show that for every time instance $t \in \mathbb{R}_0^+$ it holds that $\mathbf{x}(t) \in S$.

Since $\mathrm{cl}(S)$ is closed and bounded, the continuous function V_k is bounded from below on this set for $k = 1, \ldots, c$.

The definition of m implies that if $k \in \{1, \ldots, c\}$ is such that for every $i \in \mathbb{N}$ there exists $i' \in \mathbb{N}$ such that $i' > i$ and $\lambda(\eta[i']) = k$, then $k \geq m$. Thus, there exist $\widehat{i} \in \mathbb{N}$ such that for every $i \geq \widehat{i}$ it holds that $\lambda(\eta[i]) \geq m$. That is, there is a position $\widehat{i}$ from which on, only colors greater or equal to m occur in the run η of $\mathcal{A}$ on ρ, and m occurs infinitely often.

Since the set of states P_k overapproximates the set of states that can be associated with color k, we have that there exists a $T \in \mathbb{R}_0^+$ such that for every $t \geq T$ there exists $k \geq m$ such that $\mathbf{x}(t) \in P_k$. Furthermore, for every $t \in \mathbb{R}_0^+$ there exists $t' > t$ such that $\mathbf{x}(t') \in P_m$. Therefore, by the fourth condition on the functions $V_1, \ldots, V_c$ it holds that there exists a $j \leq m$ such that the value of V_j along $\mathbf{x}$ goes to minus infinity as t goes to infinity. This contradicts the fact that V_j is bounded from below on the set $\mathrm{cl}(S)$. □

4. PROOF SYSTEM FOR ATL*

In this section we present a proof system for deductive verification of continuous systems against ATL* properties. The idea is to decompose the problem into a set of simpler verification (synthesis) problems that can be solved by constructing suitable certificate functions.

The proof system, which we call $\mathbb{P}$, consists of a set of inference rules. The types of rules are similar to those used in [24] for proving ATL* properties of *discrete* infinite-state systems (and which extend to the alternating setting the rules for deductive verification of CTL* presented in [10]).

In the following, let $\mathcal{C} = (\mathbb{R}^n, W, \mathcal{W}, f)$ be a continuous system. For a ATL* state formula Φ, we write $\mathcal{C} \models \Phi$ iff $x \models_{\mathcal{C}} \Phi$ for every state x of $\mathcal{C}$. Our proof system works on statements of the form $\mathcal{C} \models \varphi$, where φ is an ATL* formula.

4.1 Rule BASIC-STATE

Let Φ be an ATL* state formula. The following proof rule allows us to reduce the verification of Φ to the verification of formulas of the form $p \to \Psi$, where p is a proposition and Ψ is a basic state formula. According to this rule, a basic state formula Ψ occurring one or more times in Φ can be replaced by a proposition p which underapproximates the set of states in which the state formula Ψ is true.

$$\frac{\begin{array}{l}\Psi \text{ is a basic state formula occurring in } \Phi \\ p \text{ is an auxiliary proposition} \\ \mathcal{C} \models \Phi[p/\Psi] \\ \mathcal{C} \models p \to \Psi\end{array}}{\mathcal{C} \models \Phi}\ \text{BASIC-STATE}$$

The soundness of this rule follows from the soundness of the respective rule given in [24] which extends the one in [10].

EXAMPLE 2. *Consider the formula Φ from Example 1. Each of the subformulas $\Psi_i = \langle\langle A_u \setminus \{u_i\}\rangle\rangle(safe\ \mathcal{U}\ low)$ is a basic state formula. Thus, we can successively apply rule* BASIC-STATE *and reduce the verification of $\mathcal{C} \models \Phi$ to the verification of $\mathcal{C} \models \Phi[p_{\Psi_1}/\Psi_1, p_{\Psi_2}/\Psi_2, p_{\Psi_3}/\Psi_3, p_{\Psi_4}/\Psi_4]$ and the statements $\mathcal{C} \models p_{\Psi_i} \to \Psi_i$, where p_{Ψ_i} is an auxiliary proposition for each $i = 1, \ldots, 4$.*

4.2 Positive BASIC-PATH rule

The positive BASIC-PATH rule is applied to formulas of the form $\Phi = p \to \langle\langle A\rangle\rangle\varphi$ where φ is an LTL formula.

The idea is to translate the LTL formula φ to a deterministic parity automaton and establish $\mathcal{C} \models \Phi$ using a parity certificate corresponding to this automaton.

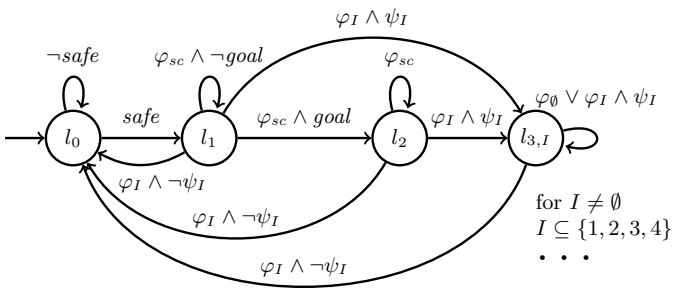
Rule POS-BASIC-PATH states that if the premises listed below are satisfied, then $\mathcal{C} \models \Phi$.

1. $\mathcal{A}_\varphi = (L, 2^{\mathsf{StateForm}(\varphi)}, \delta, l_0, C, \lambda)$ is a deterministic parity automaton for φ, with colors $C = \{1, \ldots, c\}$.
2. p' and q' are auxiliary propositions with $[\![p']\!] \subseteq [\![q']\!]$.
3. $p_1, \ldots, p_c$ are propositions such that $\bigcup_{k=1}^{c} P_k = \mathbb{R}^n$, where $P_k = [\![p_k]\!]$ for each $k \in \{1, \ldots, c\}$.
4. $[\![p']\!] \subseteq [\![p_{\lambda(l_0)}]\!]$ and for every $i, j \in \{1, \ldots, c\}$ and every $l, l' \in L$, $\sigma \in \Sigma$, $x, x' \in \mathbb{R}^n$ that are such that
   (a) $l' = \delta(l, \sigma)$, $\lambda(l) = i$, $\lambda(l') = j$, $x' \in \sigma$ and
   (b) $x' = \mathbf{x}(t, x, \mathbf{u}, \mathbf{v})$ for some $t \in \mathbb{R}_0^+$ and $(\mathbf{u}, \mathbf{v}) \in \mathcal{W}$,

   it holds that $x \models p_i$ implies $x' \models p_j$.
5. $(\{f_a\}_{a \in A}, \langle V_0, V_1, \ldots, V_c\rangle)$ is an element of the set $\mathsf{Parity}_A(\mathcal{C}, Init, S, P_1, \ldots, P_c)$ for $Init = [\![p']\!]$, $S = [\![q']\!]$.
6. The set $S = [\![q']\!]$ is bounded.
7. It holds that $\mathcal{C} \models p \to p'$.

Premise 4 above requires checking that pairs of states x and x' such that x' is a state on some trajectory of $\mathcal{C}$ originating from x satisfy a certain property. Given a characterization of (an overapproximation) of the set of such pairs as a logical formula, the condition can be checked by a procedure for checking validity for the respective logical theory.

The proposition p' underapproximates the set of states of $\mathcal{C}$ that satisfy the state formula $\langle\langle A\rangle\rangle\varphi$.



$$
\begin{array}{lcl}
\varphi_{sc} & = & safe \wedge \bigwedge_{i=1}^{4} \overline{faulty_i}, \\
\varphi_I & = & \big(\bigwedge_{i\in I} faulty_i\big) \wedge \big(\bigwedge_{i\notin I} \overline{faulty_i}\big) \text{ for } I \subseteq \{1,2,3,4\}, \\
\psi_I & = & \big(\bigwedge_{i\in I} p_{\Psi_i}\big) \text{ for } I \subseteq \{1,2,3,4\},
\end{array}
$$

$\lambda(l_0) = 1, \lambda(l_1) = 1, \lambda(l_2) = 2, \lambda(l_{3,I}) = 2.$

Figure 1: Deterministic parity automaton for φ.

In case p is *true*, the formula Φ reduces to $\langle\langle A\rangle\rangle\varphi$ for φ in LTL. The rule POS-BASIC-PATH is then simplified by replacing the last premise with $\mathcal{C} \models p'$. We call the resulting rule POS-BASIC-PATH-1.

EXAMPLE 3. *The formula resulting from rule* BASIC-STATE *in Example 2,* $\Phi[p_{\Psi_1}/\Psi_1, p_{\Psi_2}/\Psi_2, p_{\Psi_3}/\Psi_3, p_{\Psi_4}/\Psi_4]$ *is* $\langle\langle A_u\rangle\rangle\varphi$*, where* $\varphi = \Diamond(safe \wedge \varphi_{correct} \wedge \widehat{\varphi_{faulty}})$*, is an LTL formula. We can thus apply the* POS-BASIC-PATH-1 *rule.*

Figure 1 depicts part of the deterministic parity automaton $\mathcal{A}_\varphi$ *with set of colors* $\{1,2\}$ *for the formula*

$$
\begin{array}{rcl}
\varphi = \Diamond\Big(safe & \wedge & \big((\Box\bigwedge_{i=1}^{4}\overline{faulty_i}) \rightarrow (\Box\, safe \wedge \Diamond\, goal)\big) \\
& \wedge & \Box\big(\bigwedge_{i=1}^{4}(faulty_i \rightarrow p_{\Psi_i})\big)\Big).
\end{array}
$$

Suppose that the propositions p'*,* q'*,* p_1 *and* p_2 *are such that*

- $[\![p']\!] \subseteq [\![q']\!]$ *and* $[\![p_1]\!] \cup [\![p_2]\!] = \mathbb{R}^n$*;*
- $[\![p']\!] \subseteq [\![p_1]\!]$ *and for* $x, x' \in \mathbb{R}^n$ *such that* $x' = \mathbf{x}(t, x, \mathbf{u}, \mathbf{v})$ *for some* $t \in \mathbb{R}_0^+$ *and* $(\mathbf{u}, \mathbf{v}) \in \mathcal{W}$*, it holds that:*
  - *if* $x \models p_1$ *and* $x' \models \varphi_{sc} \wedge goal$*, then* $x' \models p_2$*;*
  - *if* $x \models p_1$ *and* $x' \models \varphi_I \wedge \psi_I$*, then* $x' \models p_2$*;*
  - *if* $x \models p_2$ *and* $x' \models \varphi_I \wedge \neg\psi_I$*, then* $x' \models p_1$*;*
- *The set* $S = [\![q']\!]$ *is bounded.*

Then, if f_1, f_2, f_3 *and* f_4 *are strategies for* $\mathsf{u}_1, \mathsf{u}_2, \mathsf{u}_3$ *and* u_4 *respectively, and* V_0, V_1 *and* V_2 *are functions such that the pair* $(\{f_1, f_2, f_3, f_4\}, \langle V_0, V_1, V_2\rangle)$ *is an element of the set* $\mathsf{Parity}_{A_u}(\mathcal{C}, [\![p']\!], [\![q']\!], [\![p_1]\!], [\![p_2]\!])$*, we reduce the verification of* $\mathcal{C} \models \langle\langle A_u\rangle\rangle\varphi$ *to the verification of* $\mathcal{C} \models p'$*.*

PROPOSITION 4. *Rule* POS-BASIC-PATH *is sound.*

PROOF. First, note that conditions 3 and 4 of the rule guarantee that the propositions $p_1, \ldots, p_c$ satisfy the preconditions of Proposition 3. According to the last premise of POS-BASIC-PATH, every state in $\mathcal{C}$ that satisfies p also satisfies p'. By Proposition 3, if a state x_0 belongs to $Init = [\![p']\!]$, then for every trajectory $\mathbf{x} \in \mathsf{Trajectories}(\mathcal{C}, \{x_0\}, \{f_a\}_{a\in A})$ it holds that every trace $\rho \in \mathsf{Traces}(\mathbf{x}, \Sigma)$ is accepted by $\mathcal{A}$. Thus, since $\mathcal{A}_\varphi$ is an automaton for φ, we have that every state that satisfies p also satisfies $\langle\langle A\rangle\rangle\varphi$, meaning that $\mathcal{C} \models p \rightarrow \langle\langle A\rangle\rangle\varphi$. □

4.3 Negative BASIC-PATH rule

The negative BASIC-PATH rule is applied to formulas of the form $\Phi = p \rightarrow [\![A]\!]\varphi$, where φ is an LTL formula. It relies on the fact that $\mathcal{C} \models \langle\langle Agts \setminus A\rangle\rangle\varphi$ implies $\mathcal{C} \models [\![A]\!]\varphi$.

$$\frac{\mathcal{C} \models p \rightarrow \langle\langle Agts \setminus A\rangle\rangle\varphi}{\mathcal{C} \models p \rightarrow [\![A]\!]\varphi}\ \text{NEG-BASIC-PATH}$$

It is a known fact that $\mathcal{C} \models \langle\langle Agts \setminus A\rangle\rangle\varphi$ implies $\mathcal{C} \models [\![A]\!]\varphi$, and hence the rule NEG-BASIC-PATH is sound.

By setting p to *true*, the above rule can also be applied to formulas of the form $[\![A]\!]\varphi$ where φ is an LTL formula. We call the resulting rule NEG-BASIC-PATH-1.

The following two rules POS-INV and POS-UNTIL allow us to handle directly invariance and eventuality properties, and thus they can be applied to the respective special cases of formulas of the form $\Phi = p \rightarrow \langle\langle A\rangle\rangle\varphi$ instead of applying the more general and complex POS-BASIC-PATH rule.

4.4 Positive proof rule for invariance

Consider a state formula $\Phi = p \rightarrow \langle\langle A\rangle\rangle(q\ \mathcal{W}\ r)$, where p, q and r are propositions. Rule POS-INV states that if the premises listed below are satisfied, then $\mathcal{C} \models \Phi$.

1. p' is an auxiliary proposition such that $[\![p']\!] \subseteq [\![q \wedge \neg r]\!]$.
2. $(\{f_a\}_{a\in A}, B) \in \mathsf{Barrier}_A(\mathcal{C}, Init, Err, R)$ for $\mathcal{C}$ and the sets of states $Init = [\![p']\!]$, $Err = [\![\neg q]\!]$ and $R = [\![r]\!]$.
3. It holds that $\mathcal{C} \models p \rightarrow r \vee B(X) \leq 0$.

The proposition $r \vee B(X) \leq 0$ underapproximates the set of states of the system that satisfy $\langle\langle A\rangle\rangle(q\ \mathcal{W}\ r)$.

PROPOSITION 5. *Rule* POS-INV *is sound.*

PROOF. According to the last premise of POS-INV, every state in $\mathcal{C}$ that satisfies p also satisfies $B(X) \leq 0$ or r. By Proposition 1, if a state x_0 satisfies $B(X) \leq 0$, then for every trajectory $\mathbf{x} \in \mathsf{Trajectories}(\mathcal{C}, \{x_0\}, \{f_a\}_{a\in A})$ it holds that $\mathbf{x}$ stays forever in $\overline{Err} = [\![q]\!]$, or it stays there until it reaches the set $R = [\![r]\!]$. Thus, by the semantics of the operator $\mathcal{W}$, every state that satisfies p also satisfies $\langle\langle A\rangle\rangle(q\ \mathcal{W}\ r)$, meaning that $\mathcal{C} \models p \rightarrow \langle\langle A\rangle\rangle(q\ \mathcal{W}\ r)$. □

In the special case when p is *true*, the above rule can also be applied to formulas of the form $\Phi = \langle\langle A\rangle\rangle(q\ \mathcal{W}\ r)$, where q and r are propositions. In this case, the third premise reduces to $\mathcal{C} \models r \vee B(X) \leq 0$. We call the resulting rule POS-INV-1.

4.5 Positive proof rule for eventuality

Consider a state formula $\Phi = p \rightarrow \langle\langle A\rangle\rangle(q\ \mathcal{U}\ r)$, where p, q and r are propositions. Rule POS-UNTIL states that if the premises listed below are satisfied, then $\mathcal{C} \models \Phi$.

1. p' is an auxiliary proposition such that $[\![p']\!] \subseteq [\![q \wedge \neg r]\!]$.

2. q' is a proposition such that $[\![p']\!] \subseteq [\![q']\!] \subseteq [\![q]\!]$

3. The set $S = [\![q' \vee r]\!]$ is bounded.

4. $(\{f_a\}_{a\in A}, L) \in \mathsf{Lyapunov}_A(\mathcal{C}, \mathit{Init}, S, \mathit{Target})$ for $\mathcal{C}$ and the sets $\mathit{Init} = [\![p']\!]$, $S = [\![q' \vee r]\!]$ and $\mathit{Target} = [\![r]\!]$.

5. It holds that $\mathcal{C} \models p \to r \vee (q' \wedge L(X) \leq 0)$.

The proposition $r \vee (q' \wedge L(X) \leq 0)$ underapproximates the set of states of $\mathcal{C}$ that satisfy the state formula $\langle\langle A\rangle\rangle(q\,\mathcal{U}\,r)$.

EXAMPLE 4. *Each of the formulas $p_{\Psi_i} \to \Psi_i$ resulting from the application of rule* BASIC-STATE *in Example 2 is in a form to which we can apply the* POS-UNTIL *rule.*
Let, for instance, $i = 4$ and consider the statement

$$\mathcal{C} \models p_{\Psi_4} \to \langle\langle A_u \setminus \{u_4\}\rangle\rangle(\mathit{safe}\,\mathcal{U}\,\mathit{low}).$$

Suppose that the propositions p' and q' are such that

- $[\![p']\!] \subseteq [\![\mathit{safe} \wedge \neg \mathit{low}]\!]$, $[\![p']\!] \subseteq [\![q']\!] \subseteq [\![\mathit{safe}]\!]$,
- *the set $S = [\![q' \vee \mathit{low}]\!]$ is bounded.*

Taking, for instance, $p' = \mathit{safe} \wedge \neg \mathit{low}$ and $q' = \mathit{safe}$ fulfils these conditions in our example.
Suppose further that f_1, f_2, f_3 are strategies for the controls u_1, u_2 and u_3, and that L is such that $(\{f_1, f_2, f_3\}, L) \in \mathsf{Lyapunov}_{\{\mathsf{u}_1,\mathsf{u}_2,\mathsf{u}_3\}}(\mathcal{C}, [\![p']\!], S, [\![\mathit{low}]\!])$. Then, verifying the statement $\mathcal{C} \models p_{\Psi_4} \to \Psi_i$ reduces to verifying the statement $\mathcal{C} \models p_{\Psi_4} \to \mathit{low} \vee (q' \wedge L(X) \leq 0)$.
If we succeed in finding a set of strategies and a corresponding Lyapunov function for propositions $p' = \mathit{safe} \wedge \neg \mathit{low}$ and $q' = \mathit{safe}$, then the set of states $[\![\mathit{low} \vee (\mathit{safe} \wedge L(x) \leq 0)]\!]$ underapproximates the set of states in which the first three controllers have a strategy to keep the quadrotor's altitude within the safe interval until it lowers for landing.

PROPOSITION 6. *Rule* POS-UNTIL *is sound.*

PROOF. According to the last premise of POS-UNTIL, every state in $\mathcal{C}$ that satisfies p also satisfies $q' \wedge L(X) \leq 0$ or r. By Proposition 2, if a state y satisfies $q' \wedge L(X) \leq 0$, then for every trajectory $\mathbf{x} \in \mathsf{Trajectories}(\mathcal{C}, \{y\}, \{f_a\}_{a\in A})$ it holds that $\mathbf{x}$ eventually reaches a state in $\mathit{Target} = [\![r]\!]$ and stays in the set $S = [\![q' \vee r]\!] \subseteq [\![q]\!] \cup [\![r]\!]$ until then. Thus, by the semantics of the until operator, every state that satisfies p also satisfies $\langle\langle A\rangle\rangle(q\ \mathcal{U}\ r)$, meaning that $\mathcal{C} \models p \to \langle\langle A\rangle\rangle(q\,\mathcal{U}\,r)$. □

The above rule can also be applied to formulas of the form $\langle\langle A\rangle\rangle(q\ \mathcal{U}\ r)$, where q and r are propositions. In this case, the last premise simplifies to $\mathcal{C} \models r \vee (q' \wedge L(X) \leq 0)$. We name the resulting rule POS-UNTIL-1.

4.6 Generalization rule

The following proof rule allows us to derive that a proposition p holds in every state of the $\mathcal{C}$ by establishing its validity using a procedure for the corresponding logical theory.

$$\frac{\forall x \in \mathbb{R}^n.\ x \in [\![p]\!]}{\mathcal{C} \models p}\ \text{GEN}$$

$$\dfrac{\dfrac{\dfrac{\forall x. p_\Psi \to \varphi_2}{\mathcal{C} \models p_\Psi \to \varphi_2}\text{GEN}}{\mathcal{C} \models p_\Psi \to \Psi}(2) \quad \dfrac{\dfrac{\forall x. \varphi_3}{\mathcal{C} \models \varphi_3}\text{GEN}}{\mathcal{C} \models \langle\langle\rangle\rangle \Box p_\Psi}(3)}{\mathcal{C} \models \langle\langle\rangle\rangle \Box \langle\langle A_u\rangle\rangle \Diamond \mathit{start}}(1)$$

Figure 2: Deduction tree for receptiveness property.

4.7 ATL* verification

The proof system $\mathbb{P}$ consists of the inference rules BASIC-STATE, POS-BASIC-PATH, POS-BASIC-PATH-1, NEG-BASIC-PATH, NEG-BASIC-PATH-1, POS-INV, POS-INV-1, POS-UNTIL, POS-UNTIL-1 and GEN. After applying the rule BASIC-STATE as much as possible to the input and resulting state formulas, we obtain a set of verification conditions together with the obligation to verify a set of formulas of the form $p \to \Psi$, where p is a proposition and Ψ is a basic state formula. To such formulas the positive and negative basic path rules POS-BASIC-PATH, POS-BASIC-PATH-1, NEG-BASIC-PATH, NEG-BASIC-PATH-1 can then be applied.
The soundness of the proof system follows from the soundness of the individual rules established above. The proof system is not complete, which we discuss in Section 5.

THEOREM 1. *The proof system $\mathbb{P}$ is sound, i.e., if there exists a proof for the claim $\mathcal{C} \models \Phi$, where $\mathcal{C}$ is a continuous system and Φ is a state formula, then $\mathcal{C} \models \Phi$.*

The proof rules in $\mathbb{P}$ have three types of premises: those that require temporal reasoning and are handled by applying subsequent rules, those that require checking validity of propositions and are handled by decision procedures for the underlying assertion language, and those that require computing appropriate certificate functions. We address the problem of searching for certificate functions in Section 5.1.
In the following example we show a deduction for a simple ATL* property and the set of generated constraints.

EXAMPLE 5. *We consider again the quadrotor example and the ATL* formula $\Phi = \langle\langle\rangle\rangle \Box \langle\langle A_u\rangle\rangle \Diamond \mathit{start}$, which specifies the receptiveness property, namely, that from every state on every possible trajectory, the controller has a strategy to return to its starting region described by the proposition start, in terms of the coordinates of the vehicle.*
Figure 2 shows a deduction tree of $\mathcal{C} \models \Phi$, which reduces the problem to computing a barrier certificate and a Lyapunov function together with a set of control strategies.

1. *Let $\Psi = \langle\langle A_u\rangle\rangle \Diamond \mathit{start}$. We apply rule* BASIC-STATE *with auxiliary proposition p_Ψ.*

2. *Since $\Psi \equiv \langle\langle A_u\rangle\rangle \mathit{true}\ \mathcal{U}\ \mathit{start}$, we apply rule* POS-UNTIL *with auxiliary propositions p'_2 and q'_2 such that:*

   - $[\![p'_2]\!] \subseteq [\![\neg \mathit{start}]\!]$ *and* $[\![p'_2]\!] \subseteq [\![q'_2]\!]$,
   - *the set $S_2 = [\![q'_2 \vee \mathit{start}]\!]$ is bounded.*

   This reduces the problem to:

   (a) *solving $\mathsf{Lyapunov}_{A_u}(\mathcal{C}, [\![p'_2]\!], S_2, [\![\mathit{start}]\!])$,*
   (b) *verifying the statement $\mathcal{C} \models p_\Psi \to \varphi_2$, where $\varphi_2 = \mathit{start} \vee (q'_2 \wedge L_2(x) \leq 0)$ for the Lyapunov function L_2 from condition 2a.*

3. *Since* $\Box p_\Psi \equiv p_\Psi \ \mathcal{W}\ false$ *we apply rule* POS-INV-1 *with auxiliary proposition* p'_3 *such that* $\langle\!\langle p'_3 \rangle\!\rangle \subseteq \langle\!\langle p_\Psi \rangle\!\rangle$ *and thus reduce the problem to:*
   (a) *solving* $\mathsf{Barrier}_\emptyset(\mathcal{C}, \langle\!\langle p'_3 \rangle\!\rangle, \langle\!\langle \neg p_\Psi \rangle\!\rangle, \emptyset)$,
   (b) *checking* $\mathcal{C} \models \varphi_3$, *where* $\varphi_3 = B_3(x) \leq 0$ *for the barrier certificate* B_3 *from condition 3a.*

5. DISCUSSION

5.1 Reduction to polynomials

We now consider the special case where the dynamics and the predicates are defined by polynomials and semi-algebraic constraints, respectively. Let $\mathcal{C} = (\mathbb{R}^n, W, \mathcal{W}, f)$ be a continuous system. We write $x_1, \ldots, x_n$ for the state variables, $u_1, \ldots, u_m$ for the input variables, and $v_1, \ldots, v_d$ for the disturbance variables. The continuous system $\mathcal{C}$ is a *polynomial* system if f is a polynomial over the state, input, and disturbance variables.

Recall that a polynomial constraint is of the form $p(\mathbf{x}) \sim 0$, where $\sim \in \{\leq, <, =, >, \geq\}$, and a semi-algebraic set is a set of the form $\{\mathbf{x} \mid \varphi(\mathbf{x})\}$, where φ is a Boolean combination of polynomial constraints over the variables $\mathbf{x}$. Consider a polynomial system $\mathcal{C}$ and let $\mathcal{P}$ be a set of propositions such that $\langle\!\langle p \rangle\!\rangle$ can be defined by a semi-algebraic constraint for each $p \in \mathcal{P}$.

We can reduce the ATL* verification problem to a constraint satisfaction problem over polynomials in the following way. We search for control functions and certificates that can be represented as polynomials over the state variables. For each barrier, Lyapunov, or parity certificate in the deduction tree, we fix a *template polynomial* over the state variables for the certificate (in case of a parity condition, we fix k template polynomials, one for each co-ordinate), as well as a template polynomial over the state variables for the control action. Recall that a template polynomial over a set of variables $\mathbf{x}$ is a polynomial whose coefficients are unknown parameters.

We now substitute the template polynomials in the conditions for barrier, Lyapunov, and parity certificates. This gives rise to a semi-algebraic constraint $\varphi(\mathbf{x}, \mathbf{c})$, whose free variables consist of the state variables $\mathbf{x}$ as well as the unknown parameters $\mathbf{c}$ representing the coefficients in the templates. Then, the original formula is verified if the following constraint is satisfiable for $\mathbf{c}$:

$$\forall \mathbf{x}.\varphi(\mathbf{x}, \mathbf{c})$$

The satisfiability constraint can be checked in various ways: by invoking a decision procedure for the theory of reals (or incomplete implementations for non-linear arithmetic constraints) [13, 26, 5] or by reducing to sum-of-squares optimization [18].

EXAMPLE 6. *After fixing template polynomials for the Lyapunov function* L_2, *the barrier certificate* B_3, *and the control functions for step (2) in Example 5, we obtain a constraint* $\varphi(\mathbf{x}, \mathbf{c})$, *which is a conjunction of the constraints generated by the premises of the proof rules applied in the deduction tree shown in Figure 2. For instance:*

- $\langle\!\langle p'_2 \rangle\!\rangle \subseteq \langle\!\langle \neg start \rangle\!\rangle$ *gives* $p'_2(\mathbf{x}) \rightarrow \neg start(\mathbf{x})$;
- $\mathcal{C} \models \varphi_3$ *gives* $p_x(\mathbf{x}) \rightarrow t_3(\mathbf{x}, \mathbf{c})$, *where* p_x *is a conjunction of equalities describing the state* x *and* t_3 *is the template polynomial for the barrier certificate* B_3;
- *the requirement to find a barrier certificate from the set* $\mathsf{Barrier}_\emptyset(\mathcal{C}, \langle\!\langle p'_3 \rangle\!\rangle, \langle\!\langle \neg p_\Psi \rangle\!\rangle, \emptyset)$ *results in constraints obtained from substituting* t_3, *and the propositions* p'_3, $\neg p_\Psi$ *and false in the conditions 1, 2 and 3' in the definition of Barrier certificate for control.*

5.2 Incompleteness

The proof system $\mathbb{P}$ presented in the previous section is incomplete. Firstly, the rule NEG-BASIC-PATH is not relatively complete, since it is possible that for a system $\mathcal{C}$ and a state x, we have that $x \models_\mathcal{C} [\![A]\!]\varphi$ holds and $x \models_\mathcal{C} \langle\!\langle Agts \setminus A \rangle\!\rangle\varphi$ does not hold. The reason is, that the semantics of $[\![A]\!]$ allows the behavior of the agents in $Agts \setminus A$ to depend on the chosen strategies of the agents in A, while the strategy quantifier $\langle\!\langle Agts \setminus A \rangle\!\rangle$ restricts their behavior to strategies in the respective set of functions defined by $\mathcal{C}$.

One of the premises of each of the rules BASIC-PATH, POS-INV, POS-UNTIL requires the existence of *memoryless* strategies for the agents in the set A, which depend only on the current state of the continuous system, as we pointed out in Section 3.4. Since in general strategies of this form may not exist, this restriction is a source of incompleteness. The proof system in [24] contains an additional proof rule that introduces history variables to the alternating discrete system to allow for memoryless strategies in the resulting game structure. An interesting question is how to derive conditions for certificates for controllability that allow for considering more general types of control functions.

The rule POS-INV relies on the existence of a barrier certificate whose Lie derivative is strictly negative at points where the barrier certificate is 0. This condition is known to result in a proof rule that is not relatively complete. On the other hand, making the inequality non-strict leads to unsoundness. We refer to Taly and Tiwari [27] for further discussions.

Finally, in the rule POS-UNTIL, which requires the existence of a Lyapunov function whose Lie derivative is bounded away from 0 over a certain set. Since it is possible that a function goes to minus infinity as its argument goes to infinity while its Lie derivative is not bounded away from 0, this rule is also not relatively complete.

Clearly, due to these reasons, the more general proof rule BASIC-PATH is also not relatively complete.

5.3 Extension to hybrid systems

Our proof system can be extended to handle hybrid systems in the following way. The only significant change that has to be made is in the way the premises that require finding a barrier certificate, a Lyapunov function, or certificate functions for a parity condition. This is a standard extension [18], typically done by associating a function with each discrete location of the hybrid system and relating the different functions accordingly to ensure properties are preserved by jump transitions.

Acknowledgements. We thank Jyo Deshmukh, Jim Kapinski, and the participants of the Dagstuhl seminar on Verification of Cyber-physical Systems (Seminar 14122) for useful discussions. This research was partially sponsored by a contract from Toyota Motors North America.

6. REFERENCES

[1] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, 2002.

[2] R. Alur, T. A. Henzinger, O. Kupferman, and M. Y. Vardi. Alternating refinement relations. In D. Sangiorgi and R. de Simone, editors, *CONCUR*, volume 1466 of *Lecture Notes in Computer Science*, pages 163–178. Springer, 1998.

[3] D. Angeli and E. D. Sontag. Forward completeness, unboundedness observability, and their lyapunov characterizations. *Systems & Control Letters*, 38(4 - 5):209 – 217, 1999.

[4] E. Clarke and E. Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In *Logic of Programs*, volume 131 of *LNCS*, pages 52–71. Springer, 1981.

[5] S. Gao, J. Avigad, and E. M. Clarke. δ-complete decision procedures for satisfiability over the reals. *Automated Reasoning*, pages 286–300, 2012.

[6] A. Girard. Controller synthesis for safety and reachability via approximate bisimulation. *Automatica*, 48(5):947–953, 2012.

[7] A. Girard and G. Pappas. Approximate bisimulation: A bridge between computer science and control theory. *Eur. J. Control*, 17(5-6):568–578, 2011.

[8] A. Girard, G. Pola, and P. Tabuada. Approximately bisimilar symbolic models for incrementally stable switched systems. *IEEE Trans. Automat. Contr.*, 55(1):116–126, 2010.

[9] J. Kapinski, J. Deshmukh, S. Sankaranarayanan, and N. Arechiga. Simulation-guided lyapunov analysis for hybrid dynamical systems. In *HSCC*, pages 133–142. ACM, 2014.

[10] Y. Kesten and A. Pnueli. A compositional approach to CTL* verification. *Theor. Comput. Sci.*, 331(2-3):397–428, 2005.

[11] N. Klarlund and D. Kozen. Rabin measures and their applications to fairness and automata theory. In *LICS*, pages 256–265, 1991.

[12] M. Kloetzer and C. Belta. Dealing with nondeterminism in symbolic control. In *HSCC*, LNCS 4981, pages 287–300. Springer, 2008.

[13] J. Liu, N. Zhan, and H. Zhao. Computing semi-algebraic invariants for polynomial dynamical systems. In *EMSOFT*, pages 97–106. ACM, 2011.

[14] Z. Manna and A. Pnueli. *Temporal Verification of Reactive Systems: Safety*. Springer-Verlag, 1995.

[15] A. Platzer. The complete proof theory of hybrid systems. In *LICS*, pages 541–550. IEEE, 2012.

[16] A. Pnueli. The temporal logic of programs. In *FOCS*, pages 46–57. IEEE Computer Society, 1977.

[17] G. Pola and P. Tabuada. Symbolic models for nonlinear control systems: Alternating approximate bisimulations. *SIAM J. Control and Optimization*, 48(2):719–733, 2009.

[18] S. Prajna. *Optimization-based Methods for Nonlinear and Hybrid Systems Verification*. PhD thesis, CalTech, Pasadena, CA, USA, 2005. AAI3185641.

[19] S. Prajna and A. Jadbabaie. Safety verification of hybrid systems using barrier certificates. In *HSCC*, LNCS 2993, pages 477–492. Springer, 2004.

[20] S. Prajna, P. Parrilo, and A. Rantzer. Nonlinear control synthesis by convex optimization. *IEEE Trans. Automat. Contr.*, 49(2):310–314, 2004.

[21] J.-D. Quesel and A. Platzer. Playing hybrid games with KeYmaera. In *IJCAR*, LNCS 7364, pages 439–453. Springer, 2012.

[22] A. Rantzer. A dual to Lyapunov's stability theorem. *Systems & Control Letters*, 42(3):161 – 168, 2001.

[23] A. Rantzer and S. Prajna. On analysis and synthesis of safe control laws. In *Proceedings of the Allerton Conference on Communication, Control, and Computing*. Allerton Conference, 2004.

[24] M. Slanina, H. B. Sipma, and Z. Manna. Deductive verification of alternating systems. *Form. Asp. Comput.*, 20(4-5):507–560, 2008.

[25] P. Tabuada. *Verification and Control of Hybrid Systems - A Symbolic Approach*. Springer, 2009.

[26] A. Taly, S. Gulwani, and A. Tiwari. Synthesizing switching logic using constraint solving. *STTT*, 13(6):519–535, 2011.

[27] A. Taly and A. Tiwari. Deductive verification of continuous dynamical systems. In *FSTTCS*, volume 4 of *LIPIcs*, pages 383–394, 2009.

[28] A. Taly and A. Tiwari. Switching logic synthesis for reachability. In *EMSOFT '10*, pages 19–28. ACM, 2010.

[29] E. Wolff, U. Topcu, and R. Murray. Optimal control of non-deterministic systems for a computationally efficient fragment of temporal logic. In *CDC*, pages 3197–3204. IEEE, 2013.

[30] K. Wong, C. Finucane, and H. Kress-Gazit. Provably-correct robot control with LTLMoP, OMPL and ROS. In *IROS*, page 2073. IEEE, 2013.

[31] T. Wongpiromsarn, U. Topcu, and A. G. Lamperski. Automata theory meets barrier certificates: Temporal logic verification of nonlinear systems. *CoRR*, abs/1403.3524, 2014.

[32] B. Yordanov, J. Tumova, I. Cerna, J. Barnat, and C. Belta. Temporal logic control of discrete-time piecewise affine systems. *IEEE Trans. Automat. Contr.*, 57(6):1491–1504, 2012.