Quantified Differential Invariants*

André Platzer
Carnegie Mellon University
Computer Science Department
Pittsburgh, PA, USA
aplatzer@cs.cmu.edu

ABSTRACT

We address the verification problem for distributed hybrid systems with nontrivial dynamics. Consider air traffic collision avoidance maneuvers, for example. Verifying dynamic appearance of aircraft during an ongoing collision avoidance maneuver is a longstanding and essentially unsolved problem. The resulting systems are not hybrid systems and their state space is not of the form $\mathbb{R}^n$. They are distributed hybrid systems with nontrivial continuous and discrete dynamics in distributed state spaces whose dimension and topology changes dynamically over time. We present the first formal verification technique that can handle the complicated nonlinear dynamics of these systems. We introduce quantified differential invariants, which are properties that can be checked for invariance along the dynamics of the distributed hybrid system based on differentiation, quantified substitution, and quantifier elimination in real-closed fields. This gives a computationally attractive technique, because it works without having to solve the infinite-dimensional differential equation systems underlying distributed hybrid systems. We formally verify a roundabout maneuver in which aircraft can appear dynamically.

Categories and Subject Descriptors

F.3.1 [**Logics and Meanings of Programs**]: Specifying and Verifying and Reasoning about Programs; D.2.4 [**Software Engineering**]: Software/Program Verification

General Terms

Verification, Theory, Algorithms

Keywords

Distributed hybrid systems, verification logic, quantified differential equations, quantified differential invariants

*This material is based upon work supported by the National Science Foundation under Grant Nos. CNS-0926181, CNS-0931985, and CNS-1035800.

HSCC'11, April 12–14, 2011, Chicago, Illinois, USA.

1. INTRODUCTION

Hybrid systems are everywhere [1, 9]. But several hybrid systems applications are really more than just hybrid systems. They involve multiple participants that communicate with each other to achieve their respective control objectives. The set of participants may even change over time whenever participants join or leave the networked control region. The resulting system dynamics has several aspects of hybrid systems, including their joint discrete and continuous dynamics. But that's not all there is to it.

When multiple participants are involved, the system has a high-dimensional hybrid dynamics. Then, the overall system behavior depends on the effects of how the system participants communicate and on structural properties like the communication topology, which may change over time. The most tricky part is when participants can join or leave the system at runtime, leading to reconfigurable system behavior and dynamics with varying dimensionality (appearance and disappearance add or remove state variables during a system evolution). The continuous system dynamics can no longer be described by ordinary differential equations. It already requires arbitrary-dimensional quantified differential equations [17]. Even without appearance and disappearance, flat representations of the system as a fixed set of all participants in a gigantic hybrid automaton becomes computationally infeasible for moderately-sized systems already.

The class of systems that share the above aspects is called *distributed hybrid systems* [6, 17], based on models for reconfigurable hybrid systems [5, 12]. Distributed hybrid systems evolve according to a joint discrete, continuous, structural, and dimensionality-changing dynamics. Hybrid systems [20, 21, 24, 18, 16] cannot represent the distributed aspects nor those of dynamic appearance and disappearance of participants. They would need infinitely many variables and differential equations to describe the system. Distributed systems [2] cannot represent the continuous system dynamics. We need to join both worlds to faithfully represent the system. But we also need verification techniques to handle the complicated resulting dynamics.

Prominent examples of distributed hybrid systems arise, e.g., in air traffic control, where multiple aircraft are flying to their respective goals. They use collision avoidance maneuvers as needed to prevent collisions resulting from traffic conflicts in imperfect flight trajectory planning, which have gone unnoticed by pilots and air traffic controllers.

As an example, consider the roundabout collision avoidance maneuver [19, 26] that aircraft x and y are following in Fig. 1. Here, the hybrid system dynamics comes from

the continuous movement of the aircraft and the discrete flight decisions by pilots and autopilots. Even though this has usually been ignored in formal analysis, air traffic control systems are really distributed hybrid systems, because the set of aircraft participating in a coordinated flight maneuver may be large and may change over time when more aircraft come near a traffic conflict or safely leave the dense area. Aircraft z may suddenly come too close to the collision avoidance maneuver that x and y are still performing in Fig. 1. Then z appears new into the horizon of relevance for x and y. This can happen when z changes its flight direction unaware of the avoidance intentions of x and y, when the collision avoidance of x and y takes longer than expected, or when z was forced to do collision avoidance with another aircraft a and now approaches the x, y roundabout. This is especially common in crowded airspace where global planning of all trajectories is computationally infeasible. Verifying the dynamic appearance of aircraft in the horizon of relevance during an air traffic control maneuver is an essentially unsolved problem. One major reason why distribution effects have been ignored in analysis so far is that formal verification techniques did not previously support distributed hybrid systems, especially not with such complicated dynamics as required in air traffic control. Previous techniques needed ad-hoc tricks to discuss the problem away but could not verify it.

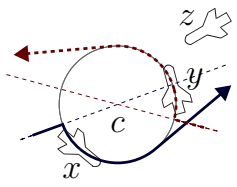


Figure 1: New appearance during collision avoidance

This paper is set out to correct this analytic deficiency by developing verification techniques for distributed hybrid systems with complicated nonlinear dynamics. For this purpose, we present a generalization of differential invariants [15, 18, 16]. These symbolic safety certificates, which we call *quantified differential invariants* can be checked for invariance along the dynamics of the distributed hybrid system based on differentiation, quantified substitution, and quantifier elimination in real-closed fields. This gives a computationally attractive technique, especially because it works without having to solve the infinite- or arbitrary-dimensional quantified differential equation systems underlying distributed hybrid systems. Our approach bears some resemblance to Lyapunov functions. But it works for safety instead of stability, it supports arbitrary logical formulas instead of just a single function, and we have extended it appropriately to cover distributed hybrid systems.

Differential invariants [15, 18, 16] are formulas F that do not change their truth-value along the dynamics of a differential equation describing a continuous system transition in a hybrid system; see Fig. 2. To prove that F is a differential invariant, it is sufficient to check a condition on the directional derivatives of all terms of the formula [18]. Like almost all other verification techniques, differential invariants only work for hybrid systems, not for distributed hybrid systems. Here we present an extension to distributed hybrid systems. In a sense made precise by our complete axiomatization relative to quantified differential equations [17], we can focus mostly on verification techniques for quantified differential equations, here, because our previous proof calculus lifts any such verification technique completely to distributed hybrid systems.

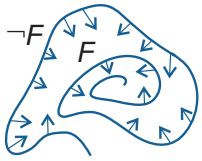


Figure 2: Differential invariant F

Our main contributions are as follows. We introduce a generalization of differential invariants (called *quantified differential invariants*), which can verify quantified differential equations. In combination with our previous work [17], this is the first formal verification technique for distributed hybrid systems with nontrivial continuous dynamics. We prove that our verification technique is formally sound. As an application scenario, we develop and verify a collision avoidance maneuver in air traffic control, where dynamic appearance of aircraft has been a major unsolved problem before. We do not claim to solve all issues in how to fly the best maneuver. Yet, we present the first verification technique with which these maneuvers can be verified. We believe this to be an important step in the development of formally assured air traffic control maneuvers and other distributed designs of cyber-physical systems.

2. RELATED WORK

The importance of understanding dynamic / reconfigurable distributed hybrid systems was recognized in modeling languages SHIFT [5] and R-Charon [12]. They focused on simulation and compilation [5] or the development of a semantics [12], so that no verification is possible yet. Stochastic simulation has been proposed [13], but soundness has not been proven, because ensuring coverage is difficult by simulation.

For distributed hybrid systems, even giving a formal semantics is very challenging [3, 22, 12, 27]! A formal semantics has been defined for a hybrid version of CSP [3] and a hybrid version of the π-calculus [22]. Rounds [22] also presented a semantics for a spatial logic for these processes. But from the semantics alone, no verification is possible in these approaches, except perhaps by manual semantic reasoning.

Other process-algebraic approaches, like χ [27], have been developed for modeling and simulation. Verification is still limited to small fragments that can be translated directly to other verification tools for (non-distributed) hybrid systems.

Our focus, instead, is on sound formal verification, not on simulation. And it is on distributed hybrid systems.

Approaches for distributed systems [2] do not cover hybrid systems, because the addition of differential equations to distributed systems is even more challenging than the addition of differential equations to discrete dynamics.

Multi-party distributed control has been suggested for air traffic control [26, 25, 14, 10]. Due to limits in previous verification technology, no full formal verification of the distributed hybrid dynamics has been possible for these systems yet. Random simulation has been proposed [14], but that does not guarantee that the maneuver always works safely.

Ad-hoc informal arguments like "the distributed effects can be ignored when we hope that at most 5 aircraft are close" are neither general nor formal verification. They also do not show whether the system stays safe if, nevertheless, a sixth aircraft a approaches. If each 5-tuple of aircraft around the sixth aircraft u assumes that u cannot appear and has to go some place else, then the system will still be unsafe, because u would have to stop moving (and drop out of the sky) to comply with all those assumptions.

Lyapunov-style verification has been used successfully for hybrid systems, including barrier certificates [20, 21], tem-

plate equations [24, 23], differential invariants [15, 18, 16] and a constraint-based template approach [7]. Here, we present an extension for distributed hybrid systems.

3. DISTRIBUTED HYBRID SYSTEMS

We review and extend a system model that we have recently introduced for modeling distributed hybrid systems [17]. The basic idea is to parameterize hybrid system models by agents and provide a means of quantifying over all affected agents (of a certain type, e.g., A for aircraft). Hence, instead of a primitive state variable like position $x:\mathbb{R}$, distributed hybrid systems have a state function $x:A\to\mathbb{R}$ so that $x(i)$ will be the position of aircraft i and $x(j)$ denotes the position of aircraft j. In order to express that all aircraft i evolve simultaneously, we will use quantified differential equations like $\forall i\, x(i)'=\theta$ with a universal quantifier $\forall i$ (i.e., for all aircraft i) and some term θ for the flight equation for each i. For the purpose of simplifying the presentation, we ignore typing information (like A for aircraft, C for cars, and $\mathbb{R}$ for reals), because it will be clear from the context. Typing is needed for modelling multi-agent hybrid systems with agents of different kinds (a mixed systems with both aircraft and cars, not just all aircraft). We refer to [17] for details on typing.

Quantified Hybrid Programs.

As a system model for distributed hybrid systems, we use *quantified hybrid programs* (QHP) [17]. QHPs are a Kleene algebra with tests [11] based on quantified assignments and quantified differential equation systems for describing distributed hybrid dynamics. QHPs are defined by the following grammar (α,β are QHPs, θ a term, i a variable, f is a function symbol, and H is a formula of first-order logic):

$$\alpha,\beta \;::=\; \forall i\, f(i):=\theta \mid \forall i\, f(i)'=\theta \,\&\, H \mid i:=\mathsf{new} \mid ?H \mid \alpha\cup\beta \mid \alpha;\beta \mid \alpha^*$$

The effect of *quantified assignment* $\forall i\, f(i):=\theta$ is an instantaneous discrete jump assigning θ to $f(i)$ simultaneously for all objects i. Usually variable i occurs in the term θ. The effect of *quantified differential equation* $\forall i\, f(i)'=\theta\,\&\,H$ is a continuous evolution where, for all objects i, all differential equations $f(i)'=\theta$ hold *and* (written &) formula H holds throughout the evolution (the state remains in the region described by H). Again, variable i usually occurs in the term θ. The dynamics of QHPs changes the interpretation of terms over time: $f(i)'$ is intended to denote the derivative of the interpretation of the term $f(i)$ over time during continuous evolution, not the derivative of $f(i)$ by its argument i. For $f(i)'$ to be defined, we assume f is an $\mathbb{R}$-valued function symbol. Time itself is implicit and can be axiomatized by a differential equation when needed. The effect of $i:=\mathsf{new}$ is to add a new object i (say a new aircraft of type A) to the system. Then $x(i)$ could denote its position, $d(i)$ its direction, and so on.

The effect of *test* $?H$ is a *skip* (i.e., no change) if formula H is true in the current state and *abort* (blocking the system run by a failed assertion), otherwise. Tests can be used to define if-then-else. *Nondeterministic choice* $\alpha\cup\beta$ is for alternatives in the behavior of the distributed hybrid system. In the *sequential composition* $\alpha;\beta$, QHP β starts after α finishes (β never starts if α continues indefinitely). *Nondeterministic repetition* α^* repeats α an arbitrary number of times, possibly zero times.

QHPs can be extended [17] to systems of quantified differential equations, simultaneous assignments to multiple functions f,g, or statements with multiple quantifiers ($\forall i\,\forall j\,\ldots$).

We use logical formulas with operators $\wedge$ (and), $\vee$ (or), $\to$ (implies), $\forall$ (forall) and $\exists$ (exists) to express properties of QHPs. In addition to all formulas of first-order real arithmetic, we allow formulas of the form $[\alpha]\phi$ with a QHP α and formula ϕ. Formula $[\alpha]\phi$ is true in a state ν iff ϕ is true in all states that are reachable from ν by following the transitions of α. We only need to define the reachability relation of QHPs in order to define a semantics for these formulas [17].

Case Study: Distributed Roundabout Maneuvers.

Before we discuss the semantics in detail, we first give an intuitive example that we will also use as a case study for distributed hybrid systems verification in Section 7. We consider the roundabout collision avoidance maneuver for air traffic control [26, 19, 16]. In the literature, formal verification of the hybrid dynamics of air traffic control focused on a fixed number of aircraft, usually two. In reality, many more aircraft are in the same flight corridor, even if not all of them participate in the same maneuver. They may be involved in multiple distributed maneuvers at the same time, however. Perfect global trajectory planning quickly becomes infeasible then. The verification itself also becomes much more complicated for three aircraft already. Explicit replication of the system dynamics n times is computationally infeasible for larger n. Yet, collision avoidance maneuvers need to work for an (essentially) unbounded number of aircraft. Because global trajectory planning is infeasible, the appearance of other aircraft into a local collision avoidance maneuver always has to be expected and managed safely. See Fig. 1 for a general illustration of roundabout-style collision avoidance maneuvers and the phenomenon of dynamic appearance of some new aircraft z into the horizon of relevance.

The resulting flight control system has several characteristics of hybrid dynamics. But it is not a hybrid system and does not even have a fixed finite number of variables in a fixed finite-dimensional state space. The system forms a distributed hybrid system, in which all aircraft fly at the same time and new aircraft may appear from remote areas into the local flight scenario. Each aircraft i has a position $x(i)=(x_1(i),x_2(i))$ and a velocity vector $d(i)=(d_1(i),d_2(i))$. We model the continuous dynamics of an aircraft i that follows a flight curve with an angular velocity $\omega(i)$ by the (function) differential equation:

$$x_1(i)'=d_1(i),\, x_2(i)'=d_2(i),\quad d_1(i)'=-\omega(i)d_2(i),\, d_2(i)'=\omega(i)d_1(i) \qquad (\mathcal{F}_{\omega(i)}(i))$$

This differential equation, which we denote by $\mathcal{F}_{\omega(i)}(i)$, is the standard equation for curved flight from the literature [26, 19, 16], but lifted to function symbols that are parameterized by aircraft i. Now, the quantified differential equation $\forall i\,\mathcal{F}_{\omega(i)}(i)$ characterizes that *all* aircraft i fly along their respective (function) differential equation $\mathcal{F}_{\omega(i)}(i)$ according to their respective angular velocities $\omega(i)$ at the same time. This quantified differential equation captures what no finite-dimensional differential equation system could ever do. It characterizes the simultaneous movement of an unbounded, arbitrary, and even growing or shrinking set of aircraft.

$$\psi \equiv \forall i,j\, \mathcal{P}(i,j) \rightarrow [drm]\forall i,j\, \mathcal{P}(i,j)$$
$$drm \equiv (free;\ \ newac;\ \ entry;\ \ \forall i\, \mathcal{F}_{\omega(i)})^*$$
$$free \equiv \forall i\, \mathcal{F}_{\omega(i)}(i)\,\&\,\forall i,j\, \mathcal{P}(i,j)$$
$$newac \equiv (n := \mathsf{new};\ ?\forall i\, \mathcal{P}(i,n)) \cup (?true)$$
$$entry \equiv c_1 := *; c_2 := *; \omega := *;$$
$$\forall i\, d_1(i) := -\omega(x_2(i) - c_2);\ \forall i\, d_2(i) := \omega(x_1(i) - c_1)$$

Figure 3: Distributed roundabout collision avoidance maneuver for air traffic control

Two aircraft i and j have violated the safe separation property if they falsify the following formula

$$\mathcal{P}(i,j) \ \equiv\ i = j \vee (x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2 \geq p^2$$

which says that aircraft i and j are either identical or separated by at least the protected zone p (usually 5mi). For the aircraft control system to be safe, all aircraft have to be safely separated, i.e., need to satisfy $\forall i,j\, \mathcal{P}(i,j)$. Yet, it is not enough that this formula holds in the beginning of the system evolution. It has to hold always. We express this safety property for collision avoidance by the logical formula ψ shown in Fig. 3. It uses the formula $[drm]\forall i,j\, \mathcal{P}(i,j)$ to say that safe separation $\mathcal{P}(i,j)$ holds for all aircraft i,j *always* after following the dynamics of the distributed hybrid system *drm* arbitrarily long, if separation holds in the beginning of the evolution (the assumption before $\rightarrow$).

We present a QHP for the distributed roundabout maneuver *drm* in Fig. 3. First, all aircraft perform free flight (*free*). They fly arbitrarily with their respective angular velocity $\omega(i)$, but can only do so as long as the system stays safe in the evolution domain $\forall i,j\, \mathcal{P}(i,j)$. That is, free flight is only permitted when the aircraft are still far enough apart. It is easy to generalize the maneuver by allowing the aircraft to repeatedly change their angular velocity $\omega(i)$ in *free*. For space reasons, we do not pursue this here. Secondly, a new aircraft may appear by the operation $n := \mathsf{new}$ in *newac*. Yet, it may only appear at positions satisfying the subsequent test $?\forall i\, \mathcal{P}(i,n)$ of n having sufficient distance to all other aircraft i. The system cannot possibly be safe if the sensors and communication is so bad that an aircraft could suddenly appear 5 feet before a collision. If no aircraft appears (the choice after operator $\cup$ in *newac*), there is no further test (trivial test $?true$). Thirdly, the aircraft (including the new one, if created during *newac*) coordinate their flight paths locally and initiate collision avoidance as necessary (*entry*). For *entry*, we use a simplified procedure based on what has been used in classical roundabouts [26, 15]. Operation *entry* chooses a center $c = (c_1, c_2)$ like in Fig. 1 and a common angular velocity ω for the roundabout. It then directs all aircraft ($\forall i$) into the appropriate directions $d(i)$. Such random assignments $c_1 := *$ that choose an arbitrary value for c_1 are definable [17]. In the last step of *drm*, all aircraft follow coordinated roundabout circles by evolving along $\forall i\, \mathcal{F}_\omega(i)$ with the common ω. Overall, *drm* repeats collision avoidance maneuvers when necessary, as indicated by the repetition *.

It might look as if new aircraft were only allowed to appear immediately before a collision avoidance maneuver happens, not in free flight. That is not the case, though, because it is perfectly okay to follow the subsequent collision avoidance circle $\forall i\, \mathcal{F}_\omega(i)$ for zero time units if the system is still safe, because it can then re-enter *free* immediately since it still satisfies its evolution domain restriction $\forall i,j\, \mathcal{P}(i,j)$.

Semantics and Differential State Flows.

What is the behavior of a QHP? A system *state* ν associates a function $\nu(x)$ of appropriate type with each function symbol x, which associates the x-value $\nu(x)(o)$ to each object o. The set of all states is denoted by $\mathcal{S}$. For representing appearance and disappearance of objects in our semantic model, we use an existence function $\mathsf{E}()$ that has value $\mathsf{E}(o) = 1$ if object o exists. For an elaboration of how we represent (dis)appearance and the new operator, we refer to previous work [17].

For a formula ϕ of first-order logic, we write $\nu \models \phi$ iff ϕ is true at state ν. We write $\nu \models [\alpha]\phi$ iff $\tau \models \phi$ for all states τ reachable from ν by a transition of QHP α. For this, we subsequently define the reachability relation $(\nu,\tau) \in \rho(\alpha)$.

We write $\nu[\![\theta]\!]$ to denote the value of term θ in state ν. Further, $\nu_i^e[\![\theta]\!]$ denotes the value of term θ in state ν_i^e, i.e., in a state like ν, but with the interpretation of variable i changed to e. Discrete assignments can be given a semantics based on the effect they have on the state, i.e., how they transform the current state ν to the next state τ. We define the behavioral semantics of quantified discrete assignments as taking an effect on all objects (say aircraft) at once. For differential equations, however, this is more difficult, because a single state is not sufficient for giving a semantics to differential equations, let alone quantified differential equations.

For a semantics for quantified differential equations, we first have to give a meaning to differential function symbols like $x(i)'$. At isolated states, differential symbols do not have a well-defined semantics, because derivatives are not defined. Thus, we consider a flow φ of the system and define a semantics for $x(i)'$ at every state along this flow.

Definition 1. (Differential state flow). A function $\varphi : [0,r] \rightarrow \mathcal{S}$ is called *(differential) state flow* of duration $r \geq 0$ if φ is componentwise continuous on $[0,r]$, i.e., for all $x \in \Sigma$ and all u, $\varphi(\zeta)(x)(u)$ is continuous in ζ. Then, the *differentially augmented state* $\bar{\varphi}(\zeta)$ of φ at $\zeta \in [0,r]$ agrees with $\varphi(\zeta)$ except that it assigns values to some of the extra differential function symbols $x^{(1)}$: If $\varphi(t)(x)(u)$ is continuously differentiable in t at ζ for all u, then $\bar{\varphi}(\zeta)(x^{(1)})$ is defined as the function $u \mapsto \frac{\mathsf{d}\varphi(t)(x)(u)}{\mathsf{d}t}(\zeta)$ that maps u to the time derivative $\frac{\mathsf{d}\varphi(t)(x)(u)}{\mathsf{d}t}(\zeta)$ at ζ; otherwise the value of $x^{(1)}$ is not defined for $\bar{\varphi}(\zeta)$. The value of $x(i)'$ at $\bar{\varphi}(\zeta)$ is defined to be the same as the value of $x^{(1)}(i)$ at $\bar{\varphi}(\zeta)$.

A state flow φ of duration r is called *state flow of the order of quantified differntial equation* $\forall i\, f(i)' = \theta\,\&\,H$ iff the value of each differential symbol occurring in it is defined on $[0,r]$. For a formula F with differential function symbols, and a state flow φ of the order of F (all differential symbols are defined), we write $\varphi \models F$ iff for all $\zeta \in [0,r]$, $\bar{\varphi}(\zeta) \models_{\mathbb{R}} F$ using the standard semantics $\models_{\mathbb{R}}$ of first-order real arithmetic. In particular, $\varphi \models \forall i\, f(i)' = \theta\,\&\,H$ iff, at each time $\zeta \in [0,r]$ and for each interpretation e of i:

- All differential equations hold and derivatives exist (trivial for $r = 0$):

$$\frac{\mathsf{d}\,(\varphi(t)_i^e[\![f(i)]\!])}{\mathsf{d}t}(\zeta) = (\varphi(\zeta)_i^e[\![\theta]\!])$$

- And the evolution domain is respected: $\varphi(\zeta)_i^e \models H$.

The *transition relation,* $\rho(\alpha) \subseteq \mathcal{S} \times \mathcal{S}$, of QHP α specifies which state $\tau \in \mathcal{S}$ is reachable from $\nu \in \mathcal{S}$ by running QHP α. It is defined inductively:

1. $(\nu, \tau) \in \rho(\forall i\, f(i) := \theta)$ iff state τ is identical to ν except that for each interpretation e of i, the value of f changes as follows: $\tau(f)\big(e\big) = \nu_i^e[\![\theta]\!]$.
2. $(\nu, \tau) \in \rho(\forall i\, f(i)' = \theta \,\&\, H)$ iff there is a differential state flow $\varphi{:}[0, r] \to \mathcal{S}$ for some $r \geq 0$ with $\varphi(0) = \nu$ and $\varphi(r) = \tau$ such that $\varphi \models \forall i\, f(i)' = \theta \,\&\, H$.
3. $\rho(?H) = \{(\nu, \nu) \,:\, \nu \models H\}$
4. $\rho(\alpha \cup \beta) = \rho(\alpha) \cup \rho(\beta)$
5. $\rho(\alpha; \beta) = \{(\nu, \tau) \,:\, (\nu, z) \in \rho(\alpha)$ and $(z, \tau) \in \rho(\beta)$ for a state $z\}$
6. $(\nu, \tau) \in \rho(\alpha^*)$ iff there is an $n \in \mathbb{N}$ with $n \geq 0$ and states $\nu = \sigma_0, \ldots, \sigma_n = \tau$ such that $(\sigma_i, \sigma_{i+1}) \in \rho(\alpha)$ for all $0 \leq i < n$.

4. DERIVATIONS & DIFFERENTIATION

Now that we have defined a semantics, we know what the behavior of the distributed hybrid system models of QHPs is. The next important question is how we can prove properties of this behavior, e.g., that a QHP never leaves a safe region. The trouble is that we cannot rely on having good solutions of the quantified differential equations that distributed hybrid systems follow. First of all, quantified differential equations do not have a fixed dimension and are thus more complicated than ordinary differential equations. Secondly, "most" differential equations (quantified or not) either have solutions that are outside decidable classes of arithmetic or have no closed-form solutions at all. The flight equation $\mathcal{F}_{\omega(i)}(i)$, for instance, has trigonometric solutions with first-order functions, which is not even semi-decidable. In either case, we cannot rely on using the solutions for verification purposes. Instead, we are looking for a different way of proving properties about the behavior of quantified differential equations by working with the quantified differential equations, not their solutions.

In preparation for that, we define the notions that we will use as crucial reasoning primitives in Section 5. We will define quantified differential invariants of formulas using syntactic total derivations and their relation to analytic differentiation. We first introduce these notions and generalize them to quantified differential equations of distributed hybrid systems. We generally assume all formulas to be given in *prefix disjunctive normal form*, that is they are of the form $Q \bigvee_i \bigwedge_j F_{i,j}$ with a prefix Q of quantifiers and atomic formulas $F_{i,j}$ that have no quantifiers or logical operators.

Derivations.

Hybrid systems evolve along trajectories that are defined in terms of the effects of discrete transitions and in terms of solutions of their differential equations. A (vectorial) function f is a solution of a differential equation system if the function f satisfies the differential equation, in which we replace f' by its analytic differentiation. This is a good mathematical definition, but not computational enough for verification purposes, because analytic differentiation is defined as a limit process at infinitely many points in time. This becomes computationally even worse for quantified differential equations, because those amount to an essentially infinite-dimensional differential equation system, which we cannot easily solve simultaneously for all positions at once.

In order to turn this mathematically precise definition into a computationally tractable algorithm, we, instead, define an entirely syntactic and algebraic total derivation and then prove that its valuation along differential flows coincides with analytic differentiation. We can easily compute the syntactic total derivation algebraically. And then we show that its value coincides with the result of analytic differentiation. In addition, we generalize total derivations to quantified formulas and constraints with first-order function symbols, as necessary for quantified differential equations.

Definition 2. (DERIVATION). The operator D that is defined as follows on terms is called *syntactic (total) derivation*:

$$D(r) = 0 \qquad \text{for } r \in \mathbb{Q} \tag{1a}$$

$$D(x(s)) = x(s)' \qquad \text{for function symbol } x : C \to \mathbb{R} \text{ with } C \neq \mathbb{R} \text{ discrete} \tag{1b}$$

$$D(a + b) = D(a) + D(b) \tag{1c}$$

$$D(a - b) = D(a) - D(b) \tag{1d}$$

$$D(a \cdot b) = D(a) \cdot b + a \cdot D(b) \tag{1e}$$

$$D(a/b) = (D(a) \cdot b - a \cdot D(b))/b^2 \tag{1f}$$

We extend D to first-order formulas F in prefix disjunctive normal form as follows:

$$\begin{aligned} D(\forall i\, F) &\equiv \forall i\, D(F) \\ D(\exists i\, F) &\equiv \forall i\, D(F) \\ D(F \wedge G) &\equiv D(F) \wedge D(G) \\ D(F \vee G) &\equiv D(F) \wedge D(G) \\ D(a \geq b) &\equiv D(a) \geq D(b) \quad \text{accordingly for } <, >, \leq, = . \end{aligned}$$

In the aircraft example, consider the safe separation formula $\forall i, j\, \mathcal{P}(i, j)$. We compute its syntactic total derivation $D(\forall i, j\, \mathcal{P}(i, j))$ to be the following (differential) expression:

$$\begin{aligned} \forall i, j \big(& i' = j' \wedge \\ & 2(x_1(i) - x_1(j))(x_1(i)' - x_1(j)') \\ & \quad + 2(x_2(i) - x_2(j))(x_2(i)' - x_2(j)') \geq 0\big) \end{aligned}$$

For making use of this syntactic total derivation for verification purposes, we need to define how we can understand its differential function symbols $x_1(i)', x_2(j)', i'$ and so on. These differential function symbols do not even have a well-defined semantics if we try to evaluate or prove the above expression at an isolated state. But the first thing we need to do is to understand what the above expression could mean at all. Def. 2 is entirely syntactical (the $x_1(i)'$ are just symbols), which is good, because then we can compute it algebraically during verification. But what is its meaning? What is its relationship to the results of real analytic differentiation $\frac{\mathrm{d}}{\mathrm{d}t}$ that define the behavioral system semantics?

Differentiation.

In the following key lemma, we show that the syntactic derivation D directly coincides with analytic differentiation

$\frac{\mathsf{d}}{\mathsf{d}t}$, even for terms with differential function symbols, as occurring in quantified differential equations.

LEMMA 1 (DERIVATION LEMMA). *The valuation of terms is a* differential homomorphism*: Let θ be a term and let $\varphi : [0,r] \to \mathcal{S}$ be any state flow of the order of $D(\theta)$ and of duration $r > 0$ along which the value of θ is defined (as no divisions by zero occur). Then we have for all $\zeta \in [0,r]$ that*

$$\frac{\mathsf{d}\,\varphi(t)[\![\theta]\!]}{\mathsf{d}t}(\zeta) = \bar{\varphi}(\zeta)[\![D(\theta)]\!].$$

In particular, $\varphi(t)[\![\theta]\!]$ is continuously differentiable (where θ is defined) and its derivative exists on $[0,r]$.

PROOF. The proof is an inductive consequence of the correspondence of the semantics of differential symbols and analytic derivatives in state flows (Def. 1). It uses the assumption that the flow φ remains within the domain of definition of θ and is continuously differentiable in all variables of θ. In particular, all denominators are nonzero during φ.

- If θ is a function term $x(s)$, the lemma holds by Def. 1:

$$\begin{aligned}\frac{\mathsf{d}\,\varphi(t)[\![x(s)]\!]}{\mathsf{d}t}(\zeta) &= \frac{\mathsf{d}\,\varphi(t)(x)(\varphi(t)[\![s]\!])}{\mathsf{d}t}(\zeta)\\ &\overset{(!)}{=} \frac{\mathsf{d}\,\varphi(t)(x)(\varphi(\zeta)[\![s]\!])}{\mathsf{d}t}(\zeta) = \bar{\varphi}(\zeta)(x')(\varphi(\zeta)[\![s]\!])\\ &= \bar{\varphi}(\zeta)[\![x'(s)]\!] = \bar{\varphi}(\zeta)[\![D(x(s))]\!].\end{aligned}$$

  The equation marked (!) holds, because the argument s of x has type $C \neq \mathbb{R}$, which is equipped with the discrete topology. Consequently, only the neighborhood $\{\zeta\}$ is relevant in the limit process of the derivative, because s only has constant local dynamics. The derivative exists because the state flow is of order 1 in x and, thus, (continuously) differentiable for x.

- If θ is of the form $a + b$, the desired result can be obtained by using the properties of derivatives, derivations (Def. 2), and evaluation $\nu[\![\cdot]\!]$ of terms:

$$\begin{aligned}&\frac{\mathsf{d}}{\mathsf{d}t}(\varphi(t)[\![a+b]\!])(\zeta)\\ &= \frac{\mathsf{d}}{\mathsf{d}t}(\varphi(t)[\![a]\!] + \varphi(t)[\![b]\!])(\zeta) && \nu[\![\cdot]\!] \text{ homomorphic}\\ &= \frac{\mathsf{d}}{\mathsf{d}t}(\varphi(t)[\![a]\!])(\zeta) + \frac{\mathsf{d}}{\mathsf{d}t}(\varphi(t)[\![b]\!])(\zeta) && \frac{\mathsf{d}}{\mathsf{d}t} \text{ is linear}\\ &= \bar{\varphi}(\zeta)[\![D(a)]\!] + \bar{\varphi}(\zeta)[\![D(b)]\!] && \text{induction hyp.}\\ &= \bar{\varphi}(\zeta)[\![D(a) + D(b)]\!] && \nu[\![\cdot]\!] \text{ homomorphic}\\ &= \bar{\varphi}(\zeta)[\![D(a+b)]\!] && D(\cdot) \text{ derivation}\end{aligned}$$

- The case where θ is of the form $a \cdot b$ or $a - b$ is similar, using Leibniz product rule (1e) or subtractivity (1d) of Def. 2, respectively.

- The case where θ is of the form a/b uses (1f) of Def. 2 and depends on the assumption that $b \neq 0$ along φ. This assumption holds as the value of θ is assumed to be defined all along state flow φ.

- The values of numbers $r \in \mathbb{Q}$ do not change during a state flow (in fact, they are not affected by the state at all); hence their derivative is $D(r) = 0$. □

With Lemma 1, syntactic total derivations are directly related to the behavior during continuous flows of the system, which is good. But how can we use them in a proof? Expressions like $D(\forall i, j\, \mathcal{P}(i,j))$ are related to expressions with a meaning along a continuous flow. But we do not want to reason explicitly about what happens at a point in time during a continuous flow, otherwise we would need to know their solutions for verification and be back at square one. What prevents us from making sense of an expression like $D(\forall i, j\, \mathcal{P}(i,j))$ is the occurrence of differential function symbols like $x_2(i)'$ in it, which are only well-defined along a flow. So instead, we find a way to get rid of the differential function symbols without changing the meaning, i.e., the link to the behavioral semantics of the distributed hybrid system.

Consider some quantified differential equation $\forall i\, f(i)' = \theta$. What is the relationship between this quantified differential equation and a quantified assignment $\forall i\, f(i) := \theta$ to the function symbol $f(i)$? Obviously that the quantified differential equation takes effect continuously where term θ describes the rate of change of $f(i)$ along a flow φ, yet the quantified assignment just has an instant effect at a single state ν of changing $f(i)$ to the new value θ once and then leaving $f(i)$ alone. This is quite a fundamental difference.

But now, what is the relationship between the quantified differential equation $\forall i\, f(i)' = \theta$ and a quantified assignment $\forall i\, f(i)' := \theta$ to the *differential function symbol* $f(i)'$? This question is more tricky. We cannot really understand the latter quantified assignment at a single state ν. First of all, the semantics of differential function symbols is only well-defined along a flow φ, not at an isolated state ν. The semantics is further defined locally per (differentially augmented) state $\bar{\varphi}(\zeta)$ for each time ζ. So, instead, we consider a flow φ and one of its local differential states $\bar{\varphi}(\zeta)$. At this local differential state, we perform the quantified assignment $\forall i\, f(i)' := \theta$ and consider its effect on a differential term υ. That is we consider $[\forall i\, f(i)' := \theta]\upsilon$ at $\bar{\varphi}(\zeta)$. Now the interesting point is that the effect of this operation corresponds directly to the local effect of the quantified differential equation. That is, if the flow φ respects the quantified differential equation $\forall i\, f(i)' = \theta$, then the quantified assignment $\forall i\, f(i)' := \theta$ does not alter the value of any differential terms.

That is, we show that a quantified assignment of the right-hand side θ of the differential equation to the differential term $f(i)'$ on its left-hand side does not change the value of arbitrary differential terms along a flow φ that already respects this quantified differential equation. This is an interesting generalized differential substitution property. It shows that quantified differential equations have consequences that correspond to substitutions by quantified assignments along their flows. Hence, locally, there is a way of understanding the effects of a quantified differential equations expressed as quantified assignments. We make this formally precise in the following lemma.

LEMMA 2 (DIFFERENTIAL SUBSTITUTION PROPERTY). *If φ is a state flow satisfying $\varphi \models \forall i\, f(i)' = \theta \,\&\, H$, then the property $\varphi \models \upsilon = [\forall i\, f(i)' := \theta]\upsilon$ holds for all (differential) terms υ that include only differential symbols of the form $f(i)'$ for some variable i.*

PROOF. The proof is by induction on the structure of υ. Consider a point in time ζ during the flow φ.

1. If υ is a differential symbol, then, by assumption, it is

$$(DI)\ \frac{H\rightarrow[\forall i\, f(i)' := \theta]D(F)}{F\rightarrow[\forall i\, f(i)' = \theta \,\&\, H]F}$$

$$(DC)\ \frac{F\rightarrow[\forall i\, f(i)' = \theta \,\&\, H]G \qquad F\rightarrow[\forall i\, f(i)' = \theta \,\&\, H \wedge G]F}{F\rightarrow[\forall i\, f(i)' = \theta \,\&\, H]F}$$

$$([:=])\ \frac{\mathsf{if}\ \exists i\, i = [\mathcal{A}]u\ \mathsf{then}\ \forall i\, (i = [\mathcal{A}]u \rightarrow \phi(\theta))\ \mathsf{else}\ \phi(f([\mathcal{A}]u))\ \mathsf{fi}}{\phi([\forall i\, f(i) := \theta]f(u))}\ {}^{1}$$

$$([\cup])\ \frac{[\alpha]\phi \wedge [\beta]\phi}{[\alpha \cup \beta]\phi}$$

[1]Occurrence $f(u)$ in $\phi(f(u))$ is not in scope of a modality and we abbreviate quantified assignment $\forall i\, f(i) := \theta$ by $\mathcal{A}$.

Figure 4: Proof rules using quantified differential invariants for distributed hybrid systems

of the form $f(i)'$. Then

$$\bar{\varphi}(\zeta)[\![f(i)']\!] = \bar{\varphi}(\zeta)[\![\theta]\!] = \bar{\varphi}(\zeta)[\![[\forall i\, f(i)' := \theta]f(i)']\!]$$

because $f(i)'$ and θ have the same value along any φ satisfying the assumption $\varphi \models \forall i\, f(i)' = \theta \,\&\, H$.

2. If υ is a non-differential function symbol $g(i)$ with a variable i, then it is not affected by assigning to differential symbols. Thus, $\varphi \models g(i) = [\forall i\, f(i)' := \theta]g(i)$.

3. If υ is a function term of the form $f(s)$ for a function symbol f and (possibly vectorial) term s, then f itself is not affected by the assignment, but s might be. Then

$$\begin{aligned}\bar{\varphi}(\zeta)[\![[\forall i\, f(i)' := \theta]f(s)]\!] &= \bar{\varphi}(\zeta)[\![f([\forall i\, f(i)' := \theta]s)]\!]\\ &= \bar{\varphi}(\zeta)[\![f(s)]\!]\end{aligned}$$

The last equation holds, because, by induction hypothesis, $\varphi \models s = [\forall i\, f(i)' := \theta]s$, which directly implies that $\bar{\varphi}(\zeta) \models s = [\forall i\, f(i)' := \theta]s$. □

5. QUANTIFIED DIFFERENTIAL INVARIANTS

Based on the notions introduced in the last sections, we can now describe our new verification approach for distributed hybrid systems with nontrivial continuous dynamics. Our verification approach is based on logic and automated theorem proving techniques. For each kind of operator that can occur in a QHP describing a distributed hybrid system, we need to give a proof rule that takes care of it. That is, for the operators $;, \cup, ^*, ?$ and, most importantly, for quantified assignments and quantified differential equations. We list our proof rules for verifying distributed hybrid systems in Fig. 4 and explain them subsequently. For the operators $;, \cup, ^*, ?$ of Kleene algebras with tests [11], there are classical proof rules, which we have previously shown to apply to hybrid systems [15]. It can be shown easily [17] that we can still use the rules for Kleene algebras in distributed hybrid systems. For instance, the proof rule $[\cup]$ axiomatizes nondeterministic choice $\cup$. This rule expresses that, when we want to prove the formula $[\alpha \cup \beta]\phi$ below the inference bar (*conclusion*), it is sufficient to prove the formula $[\alpha]\phi \wedge [\beta]\phi$ above the inference bar (*premiss*). And this makes sense, because if the premiss holds, that is, if all behavior of QHP α safely stays in the region described by formula ϕ (i.e., $[\alpha]\phi$ holds) and, independently, all behavior of β stays in ϕ (i.e., $[\beta]\phi$ holds), then all behavior of the compound system $\alpha \cup \beta$, which can choose between following any behavior of α and any behavior of β, stays safely in ϕ (i.e., the conclusion holds). This proof rule, like all of our other proof rules, decomposes a property of a compound system $\alpha \cup \beta$ into properties of simpler subsystems. This compositional verification principle is beneficial for scalability purposes, because it helps taming the system complexity by recursively reducing the system to its parts during verification.

Most importantly, we need proof rules for quantified differential equations in order to be able to prove properties of the form $[\forall i\, f(i)' = \theta \,\&\, H]F$. One option is to assume that we know an explicit closed-form solution of the quantified differential equation and use that solution as a quantified assignment $\forall i\, f(i) := \theta$ to verify that property F holds at all times when following the solution, while staying in the evolution domain H. This is the option we have pursued in previous work [17]. It is correct, but the problem with that approach is that it only works if we can find a simple closed-form solution of the quantified differential equation. It does not work, however, if the quantified differential equation has no closed-form solution that we can write down, not if it has one but we cannot compute it, and not if it only has solutions that fall into undecidable classes of arithmetic. Flight equations for curved flight, for instance, have solutions with undecidable arithmetic [16].

Here, we thus follow an entirely different approach that is not limited to working with closed-form solutions of differential equations. We present an approach that has some resemblance to Lyapunov functions. But it works for safety instead of stability, and it supports arbitrary formulas instead of just a single function. Most importantly, we have extended the approach appropriately to cover distributed hybrid systems and their arbitrary-dimensional dynamics, including appearance and disappearance of participants.

The primary insight is that, when we want to verify a property of a differential equation, we do not have to know the solution for proving a property about it. If we want to know whether formula $[\forall i\, f(i)' = \theta \,\&\, H]F$ holds, i.e., whether we always safely stay in the region described by formula F when following that continuous dynamics, then we do not need to know global solutions of where exactly each point of the state space will evolve to when following the dynamics. All we need to know is whether we can possibly ever go from somewhere safe to somewhere unsafe. This is the intuition illustrated in Fig. 2. We check if the local continuous dynamics always pushes the system state in a direction where F is becoming "more" true, not in a direction where it could become false. Then, if the system also starts safe (in F), it will always stay safe no matter where we go.

For quantified differential equations, one of the extra challenges is that the system does not have a fixed finite dimension but can be arbitrary-dimensional. Consequently, there is not even a finite vector space in which the local directions of the vector field of the differential equations can be described and checked. Instead we need a criterion that captures our verification approach based on implicit properties of the local dynamics at uncountably infinitely many points in an essentially infinite-dimensional vector field. The cardinality of this set is at least that of $\mathbb{R}^{\mathbb{N}}$ (vaguely: "∞^∞").

In Section 4, we have introduced entirely symbolic notions of syntactic total derivations and the differential substitution property, which we use to turn the above intuitions into the

formally precise and rigorous proof rule DI. For a formula F if we can prove the premiss of rule DI, i.e., that, after a differential substitution $[\forall i\, f(i)' := \theta]$, its total derivative $D(F)$ is valid in the evolution domain region H, then the conclusion of DI is valid, i.e., that the system stays in region F when it starts in F (left assumption in conclusion). It is important that we add the quantified assignment $\forall i\, f(i)' := \theta$ in the premiss, because, otherwise, the premiss of DI is not even a logical formula that would have a well-defined semantics when evaluated in a state. Unlike F, the total derivative $D(F)$ will contain differential function symbols like $f(i)'$, which do not have a semantics in isolated states but only along a flow. The quantified assignment, however, defines a value for those differential function symbols, which has a well-defined correspondence to the local dynamics of quantified differential equations by way of Lemma 2. The quantified assignment resulting in the premiss of rule DI can be handled subsequently by rule [:=].

We call formula F in rule DI a *quantified differential invariant* for quantified differential equation $\forall i\, f(i)' = \theta \,\&\, H$. Note that stronger assumptions than H are generally unsound for the premiss of DI; see previous work for details [15, 16]. Even though stronger assumptions than H have been proposed for hybrid systems [20, 7], they are generally unsound even there. That is the reason why we take extra care in this paper to make sure our proof system is actually sound (cannot prove invalid formulas). Instead, we use a proof rule called differential cut (DC) that can be used to accumulate more knowledge and extra assumptions about the dynamics successively in a sound way. The right premiss proves that the property holds when assuming G as an additional restriction on the evolution domain region, and the left premiss proves that G is actually an invariant so that restricting the dynamics to G on the right branch is just a pseudo-restriction.

Rule [:=] handles quantified assignments [17]. Their effect depends on whether $\forall i\, f(i) := \theta$ *matches* $f(u)$, i.e., there is a choice for i such that $f(u)$ is affected by the assignment, because u is of the form i for some i. If it matches, the premiss uses the term θ assigned to $f(i)$ instead of $f(u)$. Otherwise, the occurrence of f in $\phi(f(u))$ will be left unchanged. Rule [:=] makes a case distinction on matching by if-then-else. In either case, the original quantified assignment $\forall i\, f(i) := \theta$, which we abbreviate by $\mathcal{A}$, will be applied to u in the premiss, because the value of argument u may also be affected by $\mathcal{A}$, recursively. The side condition on rule [:=] makes sure that we use rule [:=] in the appropriate order.

We use classical proof rules for the operators of the Kleene algebra with tests, and refer to the literature for details [11, 17]. We also use a proof rule (written $\mathbb{R}$) for real arithmetic based on quantifier elimination in real-closed fields [4]. Because we do not need the details of real arithmetic for the purpose of this paper, we consider it as a black box and refer to previous work for an elaboration of real arithmetic [17].

As a simple example, consider the following proof:

$$\cfrac{\cfrac{\cfrac{}{\mathit{true}}\,\mathbb{R}}{\forall i\, 3(x(i)^2 + x(i)^4 + 2) \geq 0}\,[:=]}{[\forall i\, x(i)' := x(i)^2 + x(i)^4 + 2]\forall i\, 3x(i)' \geq 0}\,DI$$
$$\forall i\, 3x(i) \geq 1 \rightarrow [\forall i\, x(i)' = x(i)^2 + x(i)^4 + 2]\forall i\, 3x(i) \geq 1$$

This simple proof shows that $\forall i\, 3x(i) \geq 1$ is a (quantified differential) invariant of the quantified differential equation $\forall i\, x(i)' = x(i)^2 + x(i)^4 + 2$. Note that this differential equation is difficult to solve and the solution falls into undecidable classes of arithmetic. At the bottom the proof starts with rule DI that reduces verification to a check on the total differential of the formula after an assignment of the differential function symbol to the right hand side of the quantified differential equation. The proof then uses rule [:=] to handle the quantified assignment by substitution and finally can be proven by quantifier elimination for real arithmetic (marked by $\mathbb{R}$). In this simple example, the quantifier for i can be handled in a very simple modular way. We consider a more complicated example in our case study in Section 7.

6. SOUNDNESS

The proof rules in Fig. 4 would be entirely useless if they were unsound, because they could then be used to claim counterfactual properties as "proven" that do not hold in reality. In order to show the appropriateness of the proof rules, we, thus, prove that all provable properties are actually true.

THEOREM 1 (SOUNDNESS). *The quantified differential invariant proof rules in Fig. 4 are sound, i.e., every formula they prove is a valid property (of the distributed hybrid system that the formula refers to), i.e., it is true in all states.*

PROOF. We prove soundness of each proof rule.

DC Rule DC can be proven sound using the fact that the left premise implies that every flow φ that satisfies the quantified differential equation and evolution domain restriction H also satisfies G *all along* the flow. Thus, $\varphi \models \forall i\, f(i)' = \theta \,\&\, H$ implies $\varphi \models \forall i\, f(i)' = \theta \,\&\, H \wedge G$ so that the right premise entails the conclusion.

DI Assume that the premise is valid, i.e., true in all states. We have to show that the conclusion is valid too. Let ν be a state that satisfies the assumption F of the conclusion as, otherwise, there is nothing to show. We prove soundness by induction on the structure of F. First, we assume F to be quantifier-free in disjunctive normal form and consider any disjunct G of F that is true at ν. In order to show that F is invariant during the continuous evolution, it is sufficient to show that each conjunct of G is. We can assume these conjuncts to be of the form $c \geq 0$ (or $c > 0$ where the proof is similar). Now let $\varphi : [0, r] \to \mathcal{S}$ be any state flow with $\varphi \models \forall i\, x(i)' = \theta \,\&\, H$ beginning in $\varphi(0) = \nu$. By antecedent, $\nu \models F$. We assume duration $r > 0$, because the other case is immediate ($\nu \models F$ already holds). We show that F holds all along the flow φ, i.e., $\varphi \models F$.

Consider the case where F is of the form $c \geq 0$ (or $c > 0$ where the proof is similar). Suppose there was a $\zeta \in [0, r]$ where $\varphi(\zeta) \models c < 0$; this will lead to a contradiction. Then the function $h : [0, r] \to \mathbb{R}$ defined as $h(t) = \varphi(t)[\![c]\!]$ satisfies $h(0) \geq 0 > h(\zeta)$, because the antecedent shows $\nu \models c \geq 0$. Now, φ is of the order of $D(c)$, because: φ is of order 1 for all symbols $x(i)$, and trivially of order ∞ for variables that do not change during the differential equation. The value of c is defined all along φ, because we have assumed H to guard against zeros of denominators. Thus, by Lemma 1, h is continuous on $[0, r]$ and differentiable at every $\xi \in (0, r)$. By mean value theorem there is a $\xi \in (0, \zeta)$ such that $\frac{\mathrm{d}h(t)}{\mathrm{d}t}(\xi) \cdot (\zeta - 0) = h(\zeta) - h(0) < 0$.

In particular, since $\zeta \geq 0$, we can conclude $\frac{\mathrm{d}h(t)}{\mathrm{d}t}(\xi) < 0$. Lemma 1 implies that $\frac{\mathrm{d}h(t)}{\mathrm{d}t}(\xi) = \bar{\varphi}(\xi)[\![D(c)]\!] < 0$. And $\bar{\varphi}(\xi)[\![D(c)]\!] = \bar{\varphi}(\xi)[\![[\forall i\, x(i)' := \theta]D(c)]\!]$ by Lemma 2, as $\varphi \models \forall i\, x(i)' = \theta \,\&\, H$. This, however, is now a contradiction, because the premise actually implies that $\varphi \models H \rightarrow [\forall i\, x(i)' := \theta]D(c) \geq 0$. In particular, since $\bar{\varphi}(\xi) \models H$ holds by definition of the semantics, we have $\bar{\varphi}(\xi) \models [\forall i\, x(i)' := \theta]D(c) \geq 0$.

Second, consider the case where F is of the form $\forall j\, G$ for a fresh variable j that we can assume to occur only in G by renaming. Again let $\varphi : [0, r] \to \mathcal{S}$ be any state flow with $\varphi \models \forall i\, x(i)' = \theta \,\&\, H$ beginning in $\varphi(0) = \nu$ with $\nu \models F$. Consider *any* value e for the quantified variable j, then $\nu_j^e \models G$, i.e., $\nu \models G_j^e$. Premiss $H \rightarrow [\forall i\, x(i)' := \theta]D(F)$ is provable and, thus, by induction hypothesis valid. Then $H \rightarrow [\forall i\, x(i)' := \theta]D(G_j^e)$ is valid too, because $F \equiv \forall j\, G$ implies $D(F) \equiv \forall j\, D(G)$ and $[\forall i\, x(i)' := \theta]\forall j\, D(G)$ entails $[\forall i\, x(i)' := \theta]D(G_j^e)$ by the definition of the semantics. Consequently, G_j^e satisfies all assumptions of rule DI and the induction hypothesis implies that $G_j^e \rightarrow [\forall i\, x(i)' = \theta \,\&\, H]G_j^e$ is valid. Since assumption $\nu \models G_j^e$ holds, we know that G_j^e holds at all times when following $\forall i\, x(i)' = \theta \,\&\, H$. Now e was arbitrary, thus $\forall j\, G \rightarrow [\forall i\, x(i)' = \theta \,\&\, H]\forall j\, G$ is valid. If F is of the form $\exists j\, G$, the proof is similar, except for the last step.

[:=] For a proof of the soundness of rule [:=] and [∪], we refer to earlier work [17]. □

7. DISTRIBUTED ROUNDABOUT FLIGHT VERIFICATION

As an example of a distributed hybrid system, we have verified collision freedom in a roundabout flight collision avoidance maneuver; see Fig. 3. Unlike classical versions considered in the literature, we verify the roundabout maneuver for arbitrarily many aircraft and for an unbounded number of new aircraft that may appear into the horizon of relevance during the collision avoidance maneuver; see Fig. 1. In Fig. 5, we show the most important part of the proof for the collision-freedom property defined in Fig. 3. This part of the proof shows that the circle phase of the roundabout maneuver stays collision-free indefinitely for an arbitrary number of aircraft. That is the most crucial part, because we have to know that the aircraft remain safe during the actual roundabout collision avoidance circle. In other flight modes (e.g., *free*), the aircraft are safe by construction, because the evolution domain $\forall i, j\, \mathcal{P}(i, j)$ forces them to switch to a roundabout collision avoidance circle when the aircraft come too close. The condition $\forall i, j\, \mathcal{T}(i, j)$ characterizes compatible tangential maneuvering choices and can be proven to hold after *entry*. Without a condition like $\mathcal{T}(i, j)$, roundabouts can be unsafe [15, 16] so it is crucial that *entry* establishes it. For a systematic derivation of how to construct $\mathcal{T}(i, j)$, we refer to previous work [15, 16].

Note that the maneuver cannot be proven using any hybrid systems verification technique, because the dimension is parametric and unbounded and may even change dynamically during the remainder of the maneuver. The single proof in Fig. 5 corresponds to infinitely many proofs for systems with n aircraft for all n (plus unbounded dynamic changes of n in the proof for flight phase *newac*).

Our proof shows that the distributed roundabout maneuver safely avoids collisions for arbitrarily many aircraft (even with dynamic appearance of new aircraft). The above maneuver still requires all aircraft in the horizon of relevance to participate in the collision avoidance maneuver. In fact, we can show that this is unnecessary for aircraft that are far enough away and that may be engaged in other roundabouts. For space reasons, a discussion of these phenomena is beyond the scope of this paper, however.

8. CONCLUSIONS

Many cyber-physical systems are really distributed hybrid systems, with joint discrete, continuous, structural, and dimensional dynamics. This makes them challenging for formal verification. With hybrid systems verification, we cannot understand the distributed aspects of these systems nor aspects of dynamic appearance and disappearance of participants. With distributed systems verification, we cannot understand the continuous system dynamics. We present the first verification technique for distributed hybrid systems with nontrivial dynamics, which captures all these kinds of dynamics at once. We introduce quantified differential invariants for verifying properties of quantified differential equations. These quantified differential invariants are computationally attractive, because they can be used to verify distributed hybrid systems without having to solve their quantified differential equation systems. In particular, quantified differential invariants can be used even if the solutions cannot be computed, fall into undecidable classes of arithmetic, or do not even exist in closed form. We prove soundness of our verification approach and formally verify collision-freedom in a distributed roundabout maneuver in which new aircraft can appear dynamically at runtime.

Future work includes a more detailed study of the distributed roundabout maneuver and improving automation. In particular a number of assumptions in our distributed roundabout maneuver would be interesting to relax in future work (e.g., overly simplistic entry procedure, perfect communication, and synchronicity). Our verification approach does not depend on these assumptions, but the example does.

9. ACKNOWLEDGMENTS

I would like to thank the anonymous referees for their helpful comments.

10. REFERENCES

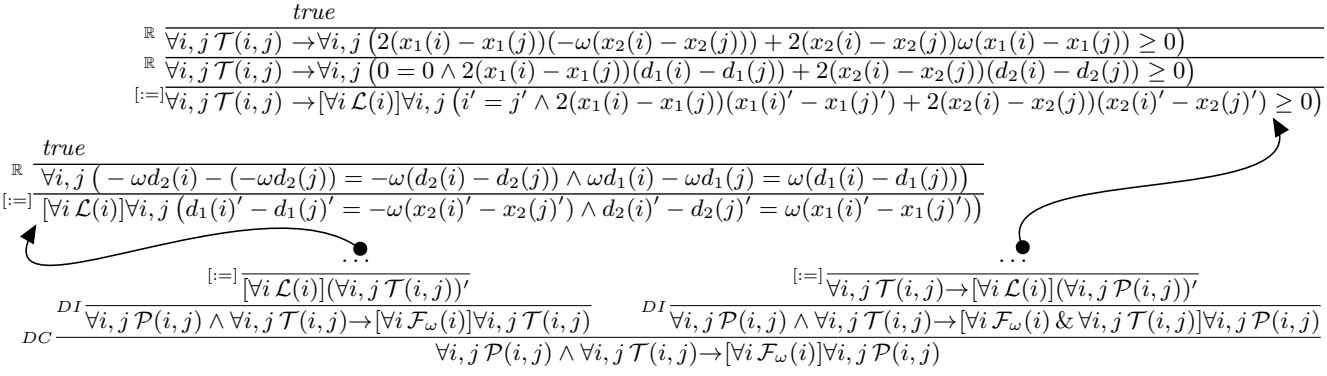
[1] R. Alur and G. J. Pappas, editors. *Hybrid Systems: Computation and Control*, volume 2993 of *LNCS*. Springer, 2004.

[2] P. C. Attie and N. A. Lynch. Dynamic input/output automata: A formal model for dynamic systems. In K. G. Larsen and M. Nielsen, editors, *CONCUR*, volume 2154 of *LNCS*, pages 137–151. Springer, 2001.

[3] Z. Chaochen, W. Ji, and A. P. Ravn. A formal description of hybrid systems. In R. Alur, T. A. Henzinger, and E. D. Sontag, editors, *Hybrid Systems*, volume 1066 of *LNCS*, pages 511–530. Springer, 1995.

[4] G. E. Collins and H. Hong. Partial cylindrical algebraic decomposition for quantifier elimination. *J. Symb. Comput.*, 12(3):299–328, 1991.

[5] A. Deshpande, A. Göllü, and P. Varaiya. SHIFT: A formalism and a programming language for dynamic



Abbreviations: $\mathcal{T}(i,j) \equiv d_1(i)-d_1(j)=-\omega(x_2(i)-x_2(j))\wedge d_2(i)-d_2(j)=\omega(x_1(i)-x_1(j))$

$\mathcal{L}(i) \equiv x_1(i)' := d_1(i), x_2(i)' := d_2(i), d_1(i)' := -\omega d_2(i), d_2(i)' := \omega d_1(i)$

Figure 5: Proof for collision freedom of roundabout collision avoidance maneuver circle

networks of hybrid automata. In P. J. Antsaklis, W. Kohn, A. Nerode, and S. Sastry, editors, *Hybrid Systems*, volume 1273 of *LNCS*, pages 113–133. Springer, 1996.

[6] S. Gilbert, N. Lynch, S. Mitra, and T. Nolte. Self-stabilizing robot formations over unreliable networks. *ACM Trans. Auton. Adapt. Syst.*, 4(3):1–29, 2009.

[7] S. Gulwani and A. Tiwari. Constraint-based approach for analysis of hybrid systems. In Gupta and Malik [8], pages 190–203.

[8] A. Gupta and S. Malik, editors. *Computer Aided Verification*, volume 5123 of *LNCS*. Springer, 2008.

[9] J. P. Hespanha and A. Tiwari, editors. *Hybrid Systems: Computation and Control*, volume 3927 of *LNCS*. Springer, 2006.

[10] I. Hwang, J. Kim, and C. Tomlin. Protocol-based conflict resolution for air traffic control. *Air Traffic Control Quarterly*, 15(1):1–34, 2007.

[11] D. Kozen. Kleene algebra with tests. *ACM Trans. Program. Lang. Syst.*, 19(3):427–443, 1997.

[12] F. Kratz, O. Sokolsky, G. J. Pappas, and I. Lee. R-Charon, a modeling language for reconfigurable hybrid systems. In Hespanha and Tiwari [9], pages 392–406.

[13] J. Meseguer and R. Sharykin. Specification and analysis of distributed object-based stochastic hybrid systems. In Hespanha and Tiwari [9], pages 460–475.

[14] L. Pallottino, V. G. Scordio, E. Frazzoli, and A. Bicchi. Decentralized cooperative policy for conflict resolution in multi-vehicle systems. *IEEE Trans. on Robotics*, 23(6):1170–1183, 2007.

[15] A. Platzer. Differential-algebraic dynamic logic for differential-algebraic programs. *J. Log. Comput.*, 20(1):309–352, 2010.

[16] A. Platzer. *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Springer, Heidelberg, 2010.

[17] A. Platzer. Quantified differential dynamic logic for distributed hybrid systems. In A. Dawar and H. Veith, editors, *CSL*, volume 6247 of *LNCS*, pages 469–483. Springer, 2010.

[18] A. Platzer and E. M. Clarke. Computing differential invariants of hybrid systems as fixedpoints. In Gupta and Malik [8], pages 176–189.

[19] A. Platzer and E. M. Clarke. Formal verification of curved flight collision avoidance maneuvers: A case study. In A. Cavalcanti and D. Dams, editors, *FM*, volume 5850 of *LNCS*, pages 547–562. Springer, 2009.

[20] S. Prajna and A. Jadbabaie. Safety verification of hybrid systems using barrier certificates. In Alur and Pappas [1], pages 477–492.

[21] S. Prajna, A. Jadbabaie, and G. J. Pappas. A framework for worst-case and stochastic safety verification using barrier certificates. *IEEE T. Automat. Contr.*, 52(8):1415–1429, 2007.

[22] W. C. Rounds. A spatial logic for the hybrid π-calculus. In Alur and Pappas [1], pages 508–522.

[23] S. Sankaranarayanan. Automatic invariant generation for hybrid systems using ideal fixed points. In K. H. Johansson and W. Yi, editors, *HSCC*, pages 221–230. ACM, 2010.

[24] S. Sankaranarayanan, H. Sipma, and Z. Manna. Constructing invariants for hybrid systems. In Alur and Pappas [1], pages 539–554.

[25] C. Tomlin, G. Pappas, J. Košecká, J. Lygeros, and S. Sastry. Advanced air traffic automation: A case study in distributed decentralized control. In B. Siciliano and K. Valavanis, editors, *Control Problems in Robotics and Automation*, volume 230 of *Lecture Notes in Control and Information Sciences*, pages 261–295. Springer, 1998.

[26] C. Tomlin, G. J. Pappas, and S. Sastry. Conflict resolution for air traffic management: a study in multi-agent hybrid systems. *IEEE T. Automat. Contr.*, 43(4):509–521, 1998.

[27] D. A. van Beek, K. L. Man, M. A. Reniers, J. E. Rooda, and R. R. H. Schiffelers. Syntax and consistent equation semantics of hybrid Chi. *J. Log. Algebr. Program.*, 68(1-2):129–210, 2006.