Differential-Algebraic Dynamic Logic for Differential-Algebraic Programs

André Platzer

University of Oldenburg, Department of Computing Science, Germany
platzer@informatik.uni-oldenburg.de

Abstract. We generalise dynamic logic to a logic for differential-algebraic programs, i.e., discrete programs augmented with first-order differential-algebraic formulas as continuous evolution constraints in addition to first-order discrete jump formulas. These programs characterise interacting discrete and continuous dynamics of hybrid systems elegantly and uniformly. For our logic, we introduce a calculus over real arithmetic with discrete induction and a new *differential induction* with which differential-algebraic programs can be verified by exploiting their differential constraints algebraically without having to solve them. We develop the theory of differential induction and differential refinement and analyse their deductive power. As a case study, we present parametric tangential roundabout maneuvers in air traffic control and prove collision avoidance in our calculus.

Keywords: dynamic logic, differential constraints, sequent calculus, verification of hybrid systems, differential induction, theorem proving

1 Introduction

Verification of Hybrid Systems. As a model for complex control systems, *hybrid systems* [23, 6, 12] are dynamical systems [45] that are governed by interacting discrete and continuous dynamics. For discrete transitions, the hybrid system changes state instantaneously and possibly discontinuously. During continuous transitions, the system state is a continuous function of continuous time and varies according to a differential equation, which is possibly subject to domain restrictions or algebraic relations resulting from physical circumstances or the interaction of continuous dynamics with discrete control.

For example, flight maneuvers in air traffic control [47, 28, 29, 14, 11, 34, 18, 25] give hybrid systems with challenging dynamics. There the continuous dynamics results from continuous movement of aircraft in space, and the discrete dynamics is caused by the instantaneous switching of maneuvering modes or by discrete aircraft controllers that decide when and how to initiate flight maneuvers. Proper functioning of these systems is highly safety-critical with respect to spatial separation of aircraft during all flight maneuvers, especially collision avoidance maneuvers. Their analysis, however, is challenging due to the superposition of involved continuous flight dynamics with nontrivial discrete control, causing hybrid systems like these to be neither amenable to mere continuous reasoning nor to verification techniques for purely discrete systems. Since, especially in the presence of parameters, hybrid systems cannot be verified numerically [34, 9], we present a purely symbolic approach using combined deductive and algebraic verification techniques.

Journal of Logic and Computation, 20(1), pp. 309-352, 2010.

doi:10.1093/logcom/exn070

In practice, correctness of hybrid systems further depends on the choice of parameters that naturally arise from the degrees of freedom of how a part of the system can be instantiated or how a controller can respond to input [47, 12, 10, 34, 37, 25]. For instance, correct angular velocities, proper timing, and compatible maneuver points are equally required for safe air traffic control [47, 34]. Additionally, relevant correctness properties for hybrid systems include safety, liveness, and mixed properties like reactivity [37], all of which can possibly involve (alternating) quantifiers or free variables for parameters [31, 10, 32, 37]. As a uniform approach for specifying and verifying these heterogeneous properties of hybrid systems with symbolic parameters, we introduce an extension of first-order logic and dynamic logic [22] for handling correctness statements about hybrid systems in the presence of (quantified) parameters, thereby extending our previous results [31, 32]. These combinations can even be used to discover constraints on the free parameters that are required for system correctness.

Logic for Hybrid Systems. The aim of this paper is to present logic-based techniques with which hybrid systems with interacting discrete and continuous dynamics can be specified and verified in a coherent logical framework. To this end, we introduce the *differential-algebraic dynamic logic* (DA-logic or DAL for short) as the logic of general hybrid change. As an elegant and uniform operational model for hybrid systems in DAL, we introduce *differential-algebraic programs* (DA-programs). These programs combine *first-order discrete jump constraints* (DJ-constraints) to characterise discrete transitions with support for *first-order differential-algebraic constraints* (DA-constraints) to characterise continuous transitions. DA-constraints provide a convenient way for expressing continuous system evolution constraints and give a uniform semantics to differential evolutions, systems of differential equations [48], switched systems [6], invariant constraints [23, 12], triggers [6], and differential-algebraic equations [19]. In DJ-constraints and DA-constraints, first-order quantifiers further give a natural and semantically well-founded way of expressing unbounded discrete or continuous nondeterminism in the dynamics, including nondeterminism resulting from internal choices or external disturbances. In interaction with appropriate control structure, DJ-constraints and DA-constraints can be combined to form DA-programs as uniform operational models for hybrid systems. With this, DA-programs are a generalised program notation for the standard notation of hybrid systems as hybrid automata [23].

As a specification and verification logic for hybrid systems given as DA-programs, we design the first-order dynamic logic DAL. In particular, we generalise discrete dynamic logic [22] to hybrid control and support DA-programs as actions of a first-order multi-modal logic [16], such that its modalities can be used to specify and verify correctness properties of hybrid systems. For instance, the DAL formula $[\alpha]\phi$ expresses that all traces of DA-program α lead to states satisfying the DAL formula ϕ. Likewise, $\langle\alpha\rangle\phi$ says that there is at least one state reachable by α which satisfies ϕ. Similarly, $\exists p\,[\alpha]\langle\beta\rangle\phi$ says that there is a choice of parameter p such that for all possible behaviour of DA-program α there is a reaction of DA-program β that ensures ϕ.

Deductive Verification and Differential Induction. As a means for verifying hybrid systems by proving corresponding DAL formulas, we introduce a sequent calculus. It uses side deductions [31] as a simple and concise, yet constructive, modular technique to integrate real quantifier elimination with calculus

rules for modalities. For handling discrete transitions, we present a first-order generalisation of standard calculus rules [22, 5]. Interacting continuous transitions are more involved. Formulas with very simple differential equations can be verified by using their solutions [17, 30, 3, 31, 32]: Linear differential equations with nilpotent constant coefficients (i.e., $x' = Ax$ for a matrix A with $A^n = 0$ for some n) have polynomial solutions so that arithmetic formulas about these solutions can be verified by quantifier elimination [8]. This approach, however, does not scale to hybrid systems with more sophisticated differential constraints because their solutions do not support quantifier elimination (e.g., when they involve transcendental functions), cannot be given in closed form [48], are not computable [38], or do not even exist [48, 27]. Solutions of differential equations are much more complicated than the original equations and can become transcendental even for simple linear differential equations like $x' = -y, y' = x$, where the solutions will be trigonometric functions.

Instead, as a logic-based technique for verifying DA-programs with more general differential-algebraic constraints, we introduce *first-order differential induction* as a fully algebraic form of proving logical statements about DA-constraints using their differential-algebraic constraints in a differential induction step instead of using their solutions in a reachability computation. Unlike in discrete induction, the invariant is a *differential invariant*, i.e., a property that is closed under total differentiation with respect to the differential constraints. There, the basic idea for showing invariance of a property F is to show that F holds initially and its total derivative F' holds always along the dynamics (with generalisations of total differentials to logical formulas and corresponding generalisations for quantified DA-constraints). This analysis considers all non-Zeno executions, i.e., where the system cannot switch its mode infinitely often in finite time. In addition, we introduce *differential strengthening* as a technique for refining the system dynamics by differential invariants until the property becomes provable for the refined dynamics, which we show to be crucial in practical applications.

Comparison. In our previous work [31, 32] we have introduced logics and calculi for verifying *hybrid programs*, which is the quantifier-free subclass of DA-programs without propositional connectives (see Table 1 for examples). In a companion paper [33], we have extended our base calculus for the logic $\mathsf{d}\mathcal{L}$ [31] to a free variable calculus with skolemisation, where we have focused on integrating arithmetic reasoning into the modal calculus. Further, we have proven this calculus to be complete *relative to* the handling of differential equations [31]. Complementary, in this paper, we address the question how sophisticated differential constraints themselves can be specified and verified in a way that lifts to hybrid systems, and how these techniques can be integrated seamlessly into a logic.

To this end, we design differential-algebraic programs as the *first-order completion* of hybrid programs [31–33], and we augment both the logic and the calculus with means for handling DA-constraints. In particular, we extend our logic $\mathsf{d}\mathcal{L}$ [31, 33] to the logic DAL with general first-order differential constraints plus first-order jump formulas and introduce differential induction for verifying differential-algebraic programs. Specifically, the continuous evolutions which can be handled by differential induction are strictly more expressive than those that previous calculi [49, 42, 12, 31–33] are able to handle. DAL even supports differential-algebraic equations [19]. Consequently, the DAL calculus can verify much more general scenarios, including the dynamics of aircraft maneuvers,

Table 1: Comparison of DAL with DA-programs versus $\mathsf{d}\mathcal{L}$ with hybrid programs

	$\mathsf{d}\mathcal{L}$/hybrid programs	DAL/DA-programs
expressive power	single assignments $x := 1$ differential equations $x_1' = d_1, x_2' = d_2$	propositional and quantified DJ-constraints $x > 0 \rightarrow \exists a\,(a < 5 \wedge x := a^2 + 1)$ propositional and quantified DA-constraints $\exists \omega \leq 1\,(d_1' = -\omega d_2 \wedge d_2' = \omega d_1) \vee (d_1' \leq d_2' \leq 2d_1)$
verification technology	substitutions *polynomial* solutions	quantifier elimination and substitutions first-order differential induction
quantifier integration	free variables, skolemisation	side deductions
scope of applications	nilpotent dynamics, e.g., trains in $\mathbb{R}^1$	algebraic dynamics and polynomial differential constraints, e.g., curved aircraft flight

which were out of scope for approaches that require polynomial solutions [17, 30–33]. Table 1 summarises the differences in syntactic expressiveness, discrete and continuous verification technology, arithmetic quantifier integration approach, and overall scope of applicability. The DAL extensions presented in this paper are both complementary to and compatible with our $\mathsf{d}\mathcal{L}$ calculus extensions for integrating arithmetic as presented in a companion paper [33].

Contributions. The first contribution of this paper is the generalised differential-algebraic dynamic logic DAL for differential-algebraic programs. DAL provides a uniform semantics and a concise language for specifying and verifying correctness properties of general hybrid systems with sophisticated (possibly quantified) first-order dynamics. The main contribution is a verification calculus for DAL including uniform proof rules for differential induction along first-order differential-algebraic constraints with differential invariants, differential variants, and differential strengthening. Our main theoretical contribution is our analysis of the deductive power of differential induction for classes of differential invariants. As an applied contribution, we introduce a generalised tangential roundabout maneuver in air traffic control and we demonstrate the capabilities of our approach by verifying collision avoidance in the DAL calculus. To the best of our knowledge, this is the first formal proof for (unbounded) safety of the hybrid dynamics of an aircraft maneuver with curved flight dynamics and the first sound verification result for collision avoidance with curved aircraft dynamics.

Related Work. Hybrid automata [23] are the predominant graphical notation for hybrid systems. In much the same way as finite automata can be represented as while-programs, or timed automata [1] have a notation as real-time programs [24], hybrid automata can be represented as hybrid programs [31–33]. Since hybrid programs are a subclass of DA-programs, hybrid automata can also be represented as DA-programs. Essentially, each continuous evolution state of a hybrid automaton is represented as a DA-constraint and each transition edge corresponds to a DJ-constraint. Yet, DA-programs are more expressive so that a single DA-constraint can even represent multiple continuous evolutions, for instance. For verification, the uniform compositional semantics of DA-programs can be exploited to devise a compositional calculus that reduces verification of statements about DA-programs to properties of their parts, which is an important aspect for integrating differential induction into the verification of hybrid

systems. In addition, DA-programs provide first-order dynamics with propositional connectives and quantifiers to express finite or infinite nondeterminism in discrete and continuous transitions, which gives a more expressive model than hybrid automata [23].

Most verification approaches for hybrid systems follow the model checking paradigm for hybrid automata and use approximations or abstraction refinement, e.g., [23, 7, 4], because reachability is undecidable for hybrid automata [23]. We have shown in previous work [34] that even reachability problems for fairly restricted classes of single continuous transitions are not decidable using numerical computations. Thus, we follow a purely symbolic approach in this paper. Moreover, we introduce the logic DAL, which gives a more expressive specification and verification language than reachability in model checking. In addition, using quantifiers, DAL is capable of handling quantified parametric properties.

There are other logics for hybrid systems. Zhou et al. [49] presented a duration calculus with mathematical expressions in derivatives of state variables. They use a multitude of rules and an oracle that requires separate mathematical reasoning about derivatives and continuity, which is not suitable for practical verification.

Rönkkö et al. [42] presented a guarded command language with differential relations and gave a semantics in higher-order logic with built-in derivatives. Without providing a means for verification of this higher-order logic, the approach is still limited to providing a notational variant of classical mathematics.

Davoren and Nerode [12] extended the propositional modal μ-calculus with a semantics in hybrid systems and examine topological aspects. They provided Hilbert-style calculi to prove formulas that are valid for all hybrid systems simultaneously. Thus, only limited information can be obtained about a particular system: In propositional modal logics, system behaviour needs to be axiomatised in terms of abstract actions a, b, c of *unknown effect*, which is not always possible.

The strength of our logic primarily is that it is a *first-order* dynamic logic: It handles actual operational models of hybrid systems like $x := x + 1; x' = 2x$ instead of abstract propositional actions of unknown effect. Further, we provide a calculus for actually verifying hybrid systems consisting of general DA-programs including quantified state change and differential induction techniques.

Logics for real-time systems, e.g., [24, 44], are not expressive enough to capture the dynamics of hybrid systems, particularly their differential equations, which are the main focus of this paper. For instance, Schobbens et al. [44] give complete axiomatisations of two decidable dense time propositional linear temporal logics. Unfortunately, in these propositional logics one cannot even express that relevant separation properties like $(x_1 - y_1)^2 + (x_2 - y_2)^2 \geq p^2$ hold always during the flight of aircraft guided by specific flight controllers.

Several authors [43, 41, 39, 40] argue that invariant techniques scale to more general dynamics than explicit reach-set computations or techniques that require solutions of the differential equations [17, 30–32]. Among them, there are model checking approaches [43, 41] that use equational polynomial invariants based on Gröbner basis computations. Still, the approach of Rodríguez-Carbonell and Tiwari [41] requires closed-form solutions and is restricted to linear dynamics. The major limitation of these approaches [43, 41], however, is that they only work for equational invariants of fully equation-definable hybrid systems, including equational initial sets and switching surfaces. Yet, this assumes highly regular systems without tolerances and only works for null sets. In practice, the set of initial states usually does not have measure zero, though. A thorough analysis

of collision avoidance maneuvers, for instance, should consider all initial flight paths in free flight instead of just a single restricted position corridor.

Prajna et al. [39, 40] have generalised Lyapunov functions to barrier certificates, i.e., a function B decreasing along the dynamics whose zero set separates initial from unsafe states. Further, they focus on stochastic extensions. DAL provides barrier certificates as a special case using $B \leq 0$ as a differential invariant. In a similar vein, criticality functions [10] generalise Lyapunov-functions from stability to safety, which DAL provides as a special case of differential invariants.

We generalise purely equational invariants [43, 41] and single polynomial expressions [43, 39, 40, 10] to general differential induction with real arithmetic formulas. In practice, such more general differential invariants are needed for verifying sophisticated hybrid systems including aircraft maneuvers. Further, unlike other approaches [43, 41, 39, 40], DAL leverages the full deductive power of logic, combining differential induction with discrete induction to lift these proof techniques uniformly to hybrid systems. In addition, dynamic logic can be used to prove sophisticated statements involving quantifier and modality alternations for parametric verification [31]. Finally, the DAL calculus supports combinations with differential variants for liveness properties or combinations with differential strengthening, which we show to be crucial in verifying realistic aircraft maneuvers.

In air traffic control, Tomlin et al. [47] analyse competitive aircraft maneuvers game-theoretically using Hamilton-Jacobi-Isaacs partial differential equations. They derive saddle solutions for purely angular or purely linear control actions. They propose roundabout maneuvers and give bounded-time verification results for trapezoidal straight-line approximations. Our symbolic techniques avoid exponential state space discretisations that are required for complicated PDEs and are thus more scalable for automation. Further, we handle fully parametric cases, even for more complicated curved flight dynamics.

Hwang et al. [25] have presented a straight-line aircraft conflict avoidance maneuver that involves optimisation over complicated trigonometric computations, and validate it on random numerical simulation. They show examples where the decisions of the maneuver change only slightly for small perturbations. Hwang et al. do not, however, prove that their proposed maneuver is safe with respect to actual hybrid flight dynamics.

Dowek et al. [14] and Galdino et al. [18] consider straight-line maneuvers and formalise geometrical proofs in PVS. Like in the work of Hwang et al. [25], they do not, however, consider curved flight paths nor verify actual hybrid dynamics but work with geometrical meta-level reasoning, instead.

In all these approaches [14, 18, 25], it remains to be proven separately that the geometrical meta-level considerations actually fit to the hybrid dynamics and flight equations. In contrast, our approach directly works for the hybrid flight dynamics and we verify roundabout maneuvers with curves instead of straight-line maneuvers with non-flyable instant turns only. A few approaches [11, 29] have been undertaken to modelcheck discretisations of roundabout maneuvers, which indicate avoidance of orthogonal collisions. However, the counterexamples found by our model checker in previous work [34] show for these maneuvers that collision avoidance does not extend to other initial flight paths.

Structure of this Paper. In Section 2, we introduce syntax and semantics of the differential-algebraic logic DAL. In Section 3, we introduce tangential

roundabout maneuvers in air traffic control as a case study and running example. Further, we introduce a sequent calculus with differential induction for DAL in Section 4, prove soundness and analyse the deductive power of differential induction. Using the DAL calculus, we prove, in Section 5, safety of the tangential roundabout maneuver in air traffic control. Finally, we draw conclusions and discuss future work in Section 6.

2 Syntax and Semantics of Differential-Algebraic Logic

In this section, we introduce the *differential-algebraic logic* (DAL) as a specification and verification logic for *differential-algebraic programs* (DA-programs). DA-programs constitute an elegant and uniform model for hybrid systems. We start with an informal introduction that motivates the definitions to come. DA-programs have three basic characteristics:

Discrete jump constraints. Discrete transitions, which can possibly lead to discontinuous change, are represented as discrete jump constraints (*DJ-constraints*), i.e., first-order formulas with instantaneous assignments of values to state variables as additional atomic formulas. DJ-constraints specify what new values the respective state variables assume by an instant change. For instance, $d_1 := -d_2$ specifies that the value of variable d_1 is changed to the value of $-d_2$. Multiple discrete changes can be combined conjunctively with simultaneous effect, e.g. $d_1 := -d_2 \wedge d_2 := d_1$, which assigns the previous value of $-d_2$ to d_1 and, simultaneously, the previous value of d_1 to d_2. This operation instantly rotates the vector $d = (d_1, d_2)$ by $\pi/2$ to the left. Using $d := d^\perp$ as a short vectorial notation for this jump, the DJ-constraint $(d_1 > 0 \to d := d^\perp) \wedge (d_1 \leq 0 \to d := -d^\perp)$ specifies that the direction of the rotation depends on the initial value of d_1. Finally, the DJ-constraint $\exists a\,(\omega := a^2 \wedge a < 5)$ assigns the square of some number less than 5 to ω.

Differential-algebraic constraints. Continuous dynamics is represented with differential-algebraic constraints (*DA-constraints*) as evolution constraints, i.e., first-order formulas with differential symbols x', e.g., in differential equations or inequalities. DA-constraints specify how state variables change continuously over time. For instance, $x_1' = d_1 \wedge x_2' = d_2$ says that the system continuously evolves by moving the vector $x = (x_1, x_2)$ into direction $d = (d_1, d_2)$ along the differential equation system $(x_1' = d_1, x_2' = d_2)$. Likewise, $d_1' = -\omega d_2 \wedge d_2' = \omega d_1 \wedge d_1 \geq 0$ specifies that the vector d is continuously rotating with angular velocity ω, so that (in conjunction with $x_1' = d_1 \wedge x_2' = d_2$), the direction where point x is heading to changes over time. By adding $d_1 \geq 0$ conjunctively to the DA-constraint, we express that the curving will only be able to continue while $d_1 \geq 0$. This evolution will have to stop before $d_1 < 0$. The evolution is impossible if $d_1 \geq 0$ already fails to hold initially. The DA-constraint $\exists\omega\,(d_1' = -\omega d_2 \wedge d_2' = \omega d_1 \wedge -1 \leq \omega \leq 1)$ characterises rotation with *some* angular velocity $-1 \leq \omega \leq 1$, which may even change over time, in contrast to $d_1' = -\omega d_2 \wedge d_2' = \omega d_1 \wedge -1 \leq \omega \leq 1 \wedge \omega' = 0$ or DA-constraint $d_1' = -\omega d_2 \wedge d_2' = \omega$ where ω is not allowed to change.

Differential-algebraic programs. As an operational model for hybrid systems, DJ-constraints and DA-constraints, which represent general discrete and

continuous transitions, respectively, can be combined to form a DA-program using regular expression operators $(\cup, ^*, ;)$ of regular discrete dynamic logic [22] as control structure. For example, $\omega := 1 \cup \omega := -1$ describes a controller that can either choose to set angular velocity ω to a left or right curve, by a nondeterministic choice $(\cup)$. Similarly, sequential composition $\omega := \omega + 1;\ d_1' = -\omega d_2 \wedge d_2' = \omega d_1$ says that the system first increases its angular velocity by a discrete transition and then switches to a mode in which it follows a continuous rotation with this angular velocity.

Discussion. Not all constraints involving $x := \theta$ or x' qualify as reasonable ways of characterising elementary system transitions. Unlike positive occurrences, negative occurrences of assignments like in $\neg(x := 5)$ are pointless, because they impose no meaningful transition constraints on which new value x actually assumes (but only on which value it is not assigned to). Likewise, negative occurrences of differential constraints as in $\neg(x' = 5)$ would be pointless as they do not constrain the overall evolution but allow arbitrary transitions.

Further, we disallow duplicate constraints that constrain the same variable in incompatible ways at the same time as, e.g., in $x := 2 \wedge x := 3$ or $x' = 2 \wedge x' = 3$. At any state during a system evolution, variable x can only assume one value at a time, not both 2 and 3 at once. Similarly, variables cannot evolve with contradictory slopes at the same time for any positive duration.

Finally, $\forall a\, x := a$ would be equivalent to false, because it is impossible to assign all possible choices for a (hence all reals) simultaneously to x, which can only assume one value at a time. Likewise $\forall a\, x' = a$ would be equivalent to false, because x' can only equal one real value at a time. Dually, $\exists a\, a := \theta$ is equivalent to true, because the DJ-constraint imposes no constraints nor has any visible effects (the scope of the quantified a ends with the DJ-constraint). The situation with $\exists a\, a' = \theta$ is similar.

Even though a semantics and proof rules for these cases can be defined, the respective transitions are degenerate and their technical handling is not very illuminating. Hence, in the sequel, we define DJ-constraints and DA-constraints to avoid these insignificant cases altogether. Note that the syntactical restrictions are non-essential but simplify the presentation by allowing us to focus on the interesting cases.

2.1 Syntax of Differential-Algebraic Logic

The formulas of DAL are built over a signature Σ of real-valued function and predicate symbols. The signature Σ contains the usual function and predicate symbols for real arithmetic: $+, -, \cdot, /, =, \leq, <, \geq, >$ and number symbols such as $0, 1$. State variables are represented as real-valued function symbols of arity zero (constants) in Σ. These state variables are *flexible* [5], i.e., their interpretation can change from state to state while following the transitions of a DA-program. Observe that there is no need to distinguish between discrete and continuous variables in DAL. The set $\mathrm{Term}(\Sigma)$ of *terms* is defined as in classical first-order logic, yielding rational expressions over the reals. The set of formulas of first-order logic is defined as common, giving first-order real arithmetic.

Although we are primarily interested in polynomial cases, our techniques generalise to the presence of division. Yet to avoid partiality in the semantics, we only allow to use p/q when $q \neq 0$ is present or ensured. Any formula or

constraint ϕ containing a term of the form p/q is taken to mean $\phi \wedge \neg(q = 0)$. Note that, in a certain sense, divisions cause less difficulties for the calculus than for the semantics. Particularly, our calculus uses indirect means of differential induction to conclude properties of solutions of DA-constraints, thereby avoiding the need to handle singularities in these solutions explicitly as caused by divisions by zero.

Differential-Algebraic Programs. DA-programs consist of first-order discrete jump formulas and first-order differential-algebraic formulas as primitive operations, which interact using regular control structure.

Reflecting the discussion before Section 2.1, we characterise reasonable occurrences for changes like $x := \theta$ or x' as follows. We call a formula G an *affirmative subformula* of a first-order formula F iff:

1. G is a positive subformula of F, i.e., it occurs with an even number of negations, and
2. no variable y that occurs in G is in the scope of a universal quantifier $\forall y$ of a positive subformula of F (or $\exists y$ of a negative subformula of F).

Definition 1 (Discrete jump constraint). *A* discrete jump constraint (DJ-constraint) *is a formula $\mathcal{J}$ of first-order real arithmetic over Σ with additional atomic formulas of the form $x := \theta$ where $x \in \Sigma$, $\theta \in \mathrm{Term}(\Sigma)$. The latter are called assignments and are only allowed in affirmative subformulas of DJ-constraints that are not in the scope of a quantifier for x of $\mathcal{J}$. A DJ-constraint without assignments is called* jump-free*. A variable x is (possibly)* changed *in $\mathcal{J}$ iff an assignment of the form $x := \theta$ occurs in $\mathcal{J}$.*

The effect of $(x_1 := \theta_1 \wedge \ldots \wedge x_n := \theta_n \wedge x_1 > 0) \vee (x_1 := \vartheta_1 \wedge \ldots \wedge x_n := \vartheta_n \wedge x_1 < 0)$ is to simultaneously change the interpretations of the variables x_i to the respective θ_i if $x_1 > 0$, and to change the x_i to ϑ_i if, instead, $x_1 < 0$. If neither case applies ($x_1 = 0$), the DJ-constraint evaluates to false as no disjunct applies so that no jump is possible at all, which will prevent the system from continuing any further. In particular, a jump-free DJ-constraint like $x \geq y$ corresponds to a test. It completes without changing the state if, in fact, $x \geq y$ holds true in the current state, and it aborts system evolution otherwise (deadlock). Especially, unlike the assignment $x := \theta$, which changes the value of x to that of θ, the test $x = \theta$ fails by aborting the system evolution if x does not already happen to have the value θ. If cases overlap, as in $(x := x - 1 \wedge x \geq 0) \vee x := 0$, either disjunct can be chosen to take effect by a nondeterministic choice.

Quantifiers within DJ-constraints express *unbounded discrete nondeterministic choices*. For instance, the following quantified DJ-constraint assigns *some* vector $u \in \mathbb{R}^2$ to e such that the rays spanned by $d = (d_1, d_2)$ and $u = (u_1, u_2)$ intersect:

$$\exists u_1 \exists u_2 \ (e_1 := u_1 \wedge e_2 := u_2 \wedge \exists \lambda{>}0 \, \exists \mu{>}0 \, (\lambda d_1 = \mu u_1 \wedge \lambda d_2 = \mu u_2)) \ .$$

We informally use *vectorial notation* when no confusion arises. Using vectorial quantifiers, equations, arithmetic, and assignments, the latter DJ-constraint simplifies to:

$$\exists u \, (e := u \wedge \exists \lambda{>}0 \, \exists \mu{>}0 \, \lambda d = \mu u) \ .$$

Definition 2 (Differential-algebraic constraints). *A* differential-algebraic constraint (DA-constraint) *is a formula $\mathcal{D}$ of first-order real arithmetic over $\Sigma \cup \Sigma'$, in which symbols of Σ' only occur in affirmative subformulas that are not in the scope of a quantifier of $\mathcal{D}$ for that symbol. Here Σ' is the set of all* differential symbols $x^{(n)}$ *with $n \in \mathbb{N}$ for state variables $x \in \Sigma$. A DA-constraint without differential symbols is called* non-differential*. A variable x is (possibly)* changed *in $\mathcal{D}$ iff $x^{(n)}$ occurs in $\mathcal{D}$ for an $n \geq 1$.*

Syntactically, $x^{(n)}$ is like an ordinary function symbol of arity 0 but only allowed to occur within DA-constraints not in any other formula. The intended semantics of a differential symbol $x^{(n)}$ is to denote the n-th time-derivative of x, which is used to form differential equations (or differential inequalities). We write x' for $x^{(1)}$ and x'' for $x^{(2)}$ and, sometimes, $x^{(0)}$ for the non-differential symbol x. The (partial) *order* $\operatorname{ord}_x \mathcal{D}$ of a DA-constraint $\mathcal{D}$ *in* x is the highest order $n \in \mathbb{N}$ of a differential symbol $x^{(n)}$ occurring in $\mathcal{D}$, or is not defined if no such $x^{(n)}$ occurs. The notion of order is accordingly for terms instead of DA-constraint.

The effect of a DA-constraint $\mathcal{D}$ is an ongoing continuous evolution respecting the differential and non-differential constraints of $\mathcal{D}$ during the whole evolution. For instance, the effect of $(x' = \theta \wedge x > 0) \vee (x' = -x^2 \wedge x < 0)$ is that the system evolves along $x' = \theta$ while $x > 0$, and evolves along $x' = -x^2$ when $x < 0$. This evolution can stop at any time but is never allowed to enter the region where neither case applies anymore ($x = 0$).

More generally, the differential constraints of $\mathcal{D}$ describe how the valuations of the respective state variables change continuously over time while following $\mathcal{D}$. The non-differential constraints of $\mathcal{D}$ can be understood to express domain restrictions or invariant regions of these evolutions for which the differential equations apply or within which the evolution resides. For instance, in the DA-constraint $d_1' = -\omega d_2 \wedge d_2' = \omega d_1 \wedge d_1 \geq 0$, the differential equations $d_1' = -\omega d_2$ and $d_2' = \omega d_1$ describe the change and $d_1 \geq 0$ the invariance region or maximal domain of evolution. Overlapping cases are resolved like in DJ-constraints, i.e., by nondeterministic choice. Likewise, a DA-constraint where no case applies abort the system evolution as it does not satisfy the DA-constraint. Hence, non-differential DA-constraints and jump-free DJ-constraints are both equivalent to pure tests [22]. Except for such tests, we need to distinguish DA-constraints from DJ-constraints: Only DA-constraints can have evolutions of non-zero duration and only DJ-constraints can lead to discontinuous changes.

Quantifiers within DA-constraints express *continuous nondeterministic choices*. For instance, $\exists u\,(d_1' = -(\omega + u)d_2 \wedge d_2' = (\omega + u)d_1 \wedge -0.1 \leq u \leq 0.1)$ expresses that the system follows a continuous evolution in which, at each time, the differential equations are respected for *some* choice of u in $-0.1 \leq u \leq 0.1$. In particular, the choice of u can be different at each time so that u amounts to a bounded nondeterministic disturbance during the rotation in the above DA-constraint.

When using constraint formulas to characterise system transitions, we face the usual frame problem: Typically, one does not expect variables to change their values unless the respective constraint explicitly specifies how. In this paper, we indicate constant variables explicitly so that no confusion arises. In practical applications, however, it can be quite cumbersome to have to specify $z := z$ or $z' = 0$ explicitly for all variables z that are not supposed to change. To account for that, we will define the DAL semantics so that variables that are not changed by a DJ-constraint or DA-constraint keep their value. Since free nondeterministic change of variable y is expressible using $\exists a\, y := a$ or $\exists a\, z' = a$, respectively, we

expect the changes of all changed variables to be specified explicitly in all cases of the constraints to improve readability:

Definition 3 (Homogeneous constraints). *A DA-constraint or DJ-constraint $\mathcal{C}$ is called* homogeneous *iff, in each of the disjuncts of a disjunctive normal form of $\mathcal{C}$, every changed variable of $\mathcal{C}$ is changed exactly once.*

Note that Lemma 3 from Section 4.1 will show that DA-constraints are equivalent to their disjunctive normal forms. Throughout the paper, we assume that all DA-constraints and DJ-constraints are homogeneous, thereby ensuring that all changed variables receive a new value in all cases of the respective constraint (or stay constant because they are changed nowhere in the constraint) and that no change conflicts occur.

Hence, variable y does not change during the DA-constraint $x' = -x \wedge x \geq y$ but works as a constant lower bound for the evolution of x, because no differential symbol $y^{(n)}$ with $n \geq 1$ occurs so that $y' = 0$ is assumed. If, instead, y is intended to vary, yet its variation is not specified by a differential equation but y varies according to some algebraic relation with x, then quantified DA-constraints can be used to represent such *differential-algebraic equations* [19]. For instance, the differential-algebraic equation $x' = -x, y^2 = x$, in which $y^2 = x$ is an algebraic variational constraint specifying how y changes over time, is expressible as the DA-constraint $x' = -x \wedge \exists u\,(y' = u \wedge y^2 = x)$. There, the quantified differential constraint on y essentially says that y can change arbitrarily (with arbitrary disturbance u) but only so that it always respects the relation $y^2 = x$.

Now we can define DA-programs as regular combinations of DJ-constraints and DA-constraints.

Definition 4 (Differential-algebraic programs). *The set* DA-program(Σ) *of* differential-algebraic programs*, with typical elements α, β, is inductively defined as the smallest set such that:*

- *If $\mathcal{J}$ is a DJ-constraint over Σ, then $\mathcal{J} \in$* DA-program(Σ).
- *If $\mathcal{D}$ is a DA-constraint over $\Sigma \cup \Sigma'$, then $\mathcal{D} \in$* DA-program(Σ).
- *If $\alpha, \beta \in$* DA-program(Σ) *then $(\alpha \cup \beta) \in$* DA-program(Σ).
- *If $\alpha, \beta \in$* DA-program(Σ) *then $(\alpha; \beta) \in$* DA-program(Σ).
- *If $\alpha \in$* DA-program(Σ) *then $(\alpha^*) \in$* DA-program(Σ).

Choices $\alpha \cup \beta$ are used to express behavioural alternatives between α and β, i.e., the system either follows α or it follows β. In particular, the difference between the DA-constraint $\mathcal{D} \vee \mathcal{E}$ and the DA-program $\mathcal{D} \cup \mathcal{E}$ is that the system has to commit to one choice of $\mathcal{D}$ or $\mathcal{E}$ in $\mathcal{D} \cup \mathcal{E}$, but it can switch back and forth multiple times between $\mathcal{D}$ and $\mathcal{E}$ in $\mathcal{D} \vee \mathcal{E}$. The sequential composition $\alpha; \beta$ says that DA-program β starts executing after α has finished (β never starts if α does not terminate, e.g., due to a failed test in α). Observe that, like repetitions, continuous evolutions within α can take longer or shorter. This nondeterminism is inherent in hybrid systems and as such reflected in DA-programs. Additional restrictions on the permitted duration of evolutions can simply be specified using auxiliary clocks, i.e., variables of derivative $\tau' = 1$. For instance, $\tau := 0; x' = -x^2 \wedge \tau' = 1 \wedge \tau \leq 5; \tau \geq 2$ specifies that the system only follows those evolutions along $x' = -x^2$ that take at most 5 but at least 2 time units. Repetition α^* is used to express that the hybrid process α repeats any number of times, including zero. With this, the repetition of hybrid automata transitions [23] can be represented, see [33] for details.

Purely conjunctive DA-constraints correspond to continuous dynamical systems [45]. DA-constraints with disjunctions correspond to switched continuous dynamical systems [6]. DA-programs without DA-constraints correspond to discrete dynamical systems or, when restricted to domain $\mathbb{N}$ (which is definable in DAL), to discrete while programs [22]. Regular combinations of DJ-constraints form a complete basis of discrete programs [22]. Finally, general DA-programs correspond to (first-order generalisations of) hybrid dynamical systems [6, 23, 12].

Formulas of Differential-Algebraic Logic. The set of *formulas* of DAL is defined as common in first-order dynamic logic [22]. They are built using propositional connectives and, in addition, if α is a DA-program and ϕ is a DAL formula, then $[\alpha]\phi, \langle\alpha\rangle\phi$ are DAL formulas. The intuitive reading of $[\alpha]\phi$ is that every run of DA-program α leads to states satisfying ϕ. Dually, $\langle\alpha\rangle\phi$ expresses that there is at least one run of α leading to such a state.

Definition 5 (DAL formulas). *The set* $\mathrm{Fml}(\Sigma)$ *of* DAL formulas*, with typical elements* ϕ, ψ*, is inductively defined as the smallest set with:*

- *If* $\theta_1, \theta_2 \in \mathrm{Term}(\Sigma)$ *are terms, then* $(\theta_1 \geq \theta_2) \in \mathrm{Fml}(\Sigma)$*, and accordingly for* $=, \leq, <, >$*.*
- *If* $\phi, \psi \in \mathrm{Fml}(\Sigma)$*, then* $\neg\phi, (\phi \wedge \psi), (\phi \vee \psi), (\phi \rightarrow \psi) \in \mathrm{Fml}(\Sigma)$*.*
- *If* $\phi \in \mathrm{Fml}(\Sigma)$ *and* $\alpha \in \text{DA-program}(\Sigma)$*, then* $[\alpha]\phi, \langle\alpha\rangle\phi \in \mathrm{Fml}(\Sigma)$*.*

Quantifiers in DAL formulas are definable in terms of DA-constraints or quantified DJ-constraints. We consider quantifiers as abbreviations:

$$\begin{aligned} \forall x\, \phi &\equiv [\exists a\, x := a]\phi \equiv [x' = 1 \vee x' = -1]\phi \\ \exists x\, \phi &\equiv \langle \exists a\, x := a\rangle\phi \equiv \langle x' = 1 \vee x' = -1\rangle\phi \ . \end{aligned}$$

The DAL formula $[\exists a\, x := a]\phi$ considers *all* possibilities of assigning *some* value a to x, which amounts to universal quantification. Likewise, $\langle\exists a\, x := a\rangle\phi$ considers some such choice, which is existential quantification. Similarly, the indeterminate continuous evolution $x' = 1 \vee x' = -1$ reaches all values, which amounts to the respective quantifier when combined with the appropriate modality.

One common pattern for representing safety statements about hybrid control loops is to use DAL formulas of the form $\phi \rightarrow [(\mathit{controller}; \mathit{plant})^*]\psi$ for specifying that the system satisfies property ψ whenever the initial state satisfies ϕ. There, the system repeats a controller-plant feedback loop, with a DA-constraint *plant* describing the continuous plant dynamics and a discrete DA-program *controller* describing the control decisions. The controller plant interaction repeats as indicated by the repetition star. Still, more general forms of systems and properties can be formulated and verified in DAL as well.

2.2 Semantics of Differential-Algebraic Logic

The semantics of DAL is a Kripke semantics with possible states of a hybrid system as possible worlds, where the accessibility relation between worlds is generated by the discrete or continuous transitions of DA-programs. A potential behaviour of a hybrid system corresponds to a succession of states that contain the observable values of system variables during its hybrid evolution.

Transition Semantics. A *state* is a map $\nu : \Sigma \to \mathbb{R}$; the set of all states is denoted by $\mathrm{State}(\Sigma)$. The function and predicate symbols of real arithmetic are interpreted as usual.

Definition 6 (Valuation of terms). *The* valuation $val(\nu, \cdot)$ of terms *with respect to state ν is defined by*

1. $val(\nu, x) = \nu(x)$ *if $x \in \Sigma$ is a variable.*
2. $val(\nu, \theta_1 + \theta_2) = val(\nu, \theta_1) + val(\nu, \theta_2)$ *and accordingly for $-, \cdot$.*
3. $val(\nu, \theta_1/\theta_2) = val(\nu, \theta_1)/val(\nu, \theta_2)$ *if $val(\nu, \theta_2) \neq 0$.*

Note that we do not need the semantics of θ_1/θ_2 for $val(\nu, \theta_2) = 0$, because we have assumed the presence of constraints ensuring $\neg(\theta_2 = 0)$ for divisions.

The interpretation of discrete jump constraints is defined as in first-order real arithmetic with the addition of an interpretation for assignment formulas.

Definition 7 (Interpretation of discrete jump constraints). *The* interpretation *of DJ-constraint $\mathcal{J}$ for the pair of states (ν, ω), denoted as $(\nu, \omega) \models \mathcal{J}$, is defined as follows, where $val(\omega, z) = val(\nu, z)$ for all variables z that are not changed in $\mathcal{J}$:*

1. $(\nu, \omega) \models x := \theta$ *iff* $val(\omega, x) = val(\nu, \theta)$.
2. $(\nu, \omega) \models \theta_1 \geq \theta_2$ *iff* $val(\nu, \theta_1) \geq val(\nu, \theta_2)$, *and accordingly for* $=, \leq, <, >$.
3. $(\nu, \omega) \models \phi \wedge \psi$ *iff* $(\nu, \omega) \models \phi$ *and* $(\nu, \omega) \models \psi$. *Accordingly for* $\neg, \vee, \rightarrow$.
4. $(\nu, \omega) \models \forall x\, \phi$ *iff* $(\nu_x, \omega) \models \phi$ *for all states ν_x that agree with ν except for the value of x.*
5. $(\nu, \omega) \models \exists x\, \phi$ *iff* $(\nu_x, \omega) \models \phi$ *for some state ν_x that agrees with ν except for the value of x.*

To give a semantics to DA-constraints, differential symbols $x' \in \Sigma'$ must get a meaning. However, a DA-constraint like $d_1' = -\omega d_2 \wedge d_2' = \omega d_1$ cannot be interpreted in a single state ν, because derivatives are not defined in isolated points. Instead, DA-constraints are constraints that have to hold for an evolution of states over time. Along such a *flow* function $\varphi : [0, r] \to \mathrm{State}(\Sigma)$, DA-constraints can again be interpreted locally by assigning to the formal differential symbol d_1' the analytic time-derivative of the value of d_1 along φ at the respective points in time. As we assumed DA-constraints to avoid zero divisions, analytic derivatives are well-defined for $r > 0$ as $\mathrm{State}(\Sigma)$ is isomorphic to a finite-dimensional real space with respect to the finitely many differential symbols occurring in the DA-constraint. We give a uniform definition for all durations $r \geq 0$ and defer the discussion of the understanding for $r = 0$ until the DA-constraint semantics has been presented in full. The philosophy behind hybrid systems is to isolate discontinuities in discrete transitions. Thus we assume that state variables (and their differential symbols, if present) always vary continuously along continuous evolutions over time.

Definition 8 (Differential state flow). *A function $\varphi : [0, r] \to \mathrm{State}(\Sigma)$ is called* state flow *of duration $r \geq 0$, if φ is componentwise continuous on $[0, r]$, i.e., for all $x \in \Sigma$, $\varphi(\zeta)(x)$ 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 further assigns values to some of the differential symbols $x^{(n)} \in \Sigma'$: If $\varphi(t)(x)$ is n-times continuously differentiable in t at ζ, then $\bar{\varphi}(\zeta)$ assigns the n-th time-derivative $\frac{\mathsf{d}^n \varphi(t)(x)}{\mathsf{d}t^n}(\zeta)$ of x at ζ to differential symbol $x^{(n)} \in \Sigma'$, otherwise the value of $x^{(n)} \in \Sigma'$ is not defined.*

For a DA-constraint $\mathcal{D}$, a state flow φ of duration r is called *state flow of the order of* $\mathcal{D}$, iff the value of each differential symbol occurring in $\mathcal{D}$ is defined on $[0,r]$, i.e., $\varphi(\zeta)(x)$ is n-times continuously differentiable in ζ on $[0,r]$ for $n = \operatorname{ord}_x \mathcal{D}$.

Definition 9 (Interpretation of differential-algebraic constraints). *The* interpretation of DA-constraint $\mathcal{D}$ *with respect to a state flow* φ *of the order of* $\mathcal{D}$ *and duration* $r \geq 0$ *is defined by:* $\varphi \models \mathcal{D}$ *iff, for all* $\zeta \in [0,r]$*,*

1. $\bar{\varphi}(\zeta) \models_{\mathbb{R}} \mathcal{D}$ *using the standard semantics* $\models_{\mathbb{R}}$ *of first-order real arithmetic, and*
2. $val(\bar{\varphi}(\zeta), z) = val(\bar{\varphi}(0), z)$ *for all variables* z *that are not changed by* $\mathcal{D}$*.*

Observe that, along the state flows for a DA-constraint $\mathcal{D}$, only those variables whose differential symbols occur in $\mathcal{D}$ have to be continuously differentiable of the appropriate order. Quantified variables can change more arbitrarily (even discontinuously) during the evolution, because the semantics does not directly relate the value of a quantified variable like u in $\exists u\, x' = u^2$ at time ζ with the values that u assumes at later times. Quantified variables may be constrained indirectly by their relations, though: In $\exists u\, x' = u^2$, the value of u^2 (but not that of u) also varies continuously over time, because x' varies continuously.

As a consequence of Picard-Lindelöf's theorem a.k.a. Cauchy-Lipschitz theorem [48, Theorem 10.VI], and using that DAL terms are continuously differentiable on the open domain where divisors are non-zero, the flows of explicit quantifier-free DA-constraints of the form $x_1' = \theta_1 \wedge \cdots \wedge x_n' = \theta_n \wedge \chi$ with non-differential χ are unique (as long as they exist): For each duration and initial value, there is at most one state flow φ. Yet, this is *not* the case for disjunctive DA-constraints, differential inequalities, quantified DA-constraints, or DA-constraints in implicit form like $x'^2 - 1 = 0$, which has solutions $x(t) = x(0) + t$ and $x(t) = x(0) - t$. Finally, a non-differential χ imposes no change but only tests whether χ holds. Hence, without differential constraints, a non-differential DA-constraint χ only has constant flows (if any), i.e., $\varphi(\zeta) = \varphi(0)$ for all ζ.

Restrictions of differential state flows to a prefix are again state flows. In particular, for all differential equations, the restriction to the point interval $[0,0]$ yields a trivial flow of no effect. For such point duration $r = 0$, however, derivatives and differentiability are not defined. To admit trivial flows nevertheless, the understanding of a DA-constraint is that its differential terms take no effect for flows of zero duration. That is, for trivial flows, atomic formulas with differential symbols are defined to evaluate to *true* as they occur only positively in DA-constraints. Thus, only the non-differential constraints of $\mathcal{D}$ impose constraints for trivial flows. A state flow of duration zero satisfying $\mathcal{D}$ and starting in some state ν exists iff ν satisfies the non-differential part of $\mathcal{D}$, which acts as a test condition.

Now we can define the transition semantics, $\rho(\alpha)$, of a DA-program α. The semantics of a DA-program is captured by the discrete or continuous transitions that are possible by following this DA-program. For DJ-constraints this transition relation holds for pairs of states that satisfy the jump constraints. For DA-constraints, the transition relation holds for pairs of states that can be interconnected by a (continuous) state flow respecting the DA-constraint.

Definition 10 (Transition semantics of differential-algebraic programs). *The* valuation, $\rho(\alpha)$, of a DA-program α*, is a* transition relation *on states. It specifies which state* ω *is reachable from a state* ν *by operations of the hybrid system* α *and is defined as:*

1. $(\nu,\omega) \in \rho(\mathcal{J})$ *iff* $(\nu,\omega) \models \mathcal{J}$ *according to Def. 7, when* $\mathcal{J}$ *is a DJ-constraint.*
2. $\rho(\mathcal{D}) = \{(\varphi(0),\varphi(r)) : \varphi$ *is a state flow of the order of* $\mathcal{D}$ *and some duration* $r \geq 0$ *such that* $\varphi \models \mathcal{D}\}$*, when* $\mathcal{D}$ *is a DA-constraint, see Def. 9.*
3. $\rho(\alpha \cup \beta) = \rho(\alpha) \cup \rho(\beta)$
4. $\rho(\alpha;\beta) = \{(\nu,\omega) : (\nu,z) \in \rho(\alpha), (z,\omega) \in \rho(\beta)$ *for some state* $z\}$
5. $(\nu,\omega) \in \rho(\alpha^*)$ *iff there are an* $n \in \mathbb{N}$ *and* $\nu = \nu_0,\ldots,\nu_n = \omega$ *such that* $(\nu_i,\nu_{i+1}) \in \rho(\alpha)$ *for all* $0 \leq i < n$.

Now, the interpretation of formulas is defined as usual for first-order modal logic [16, 22], with the transition semantics, $\rho(\alpha)$, of DA-programs for modalities.

Definition 11 (Interpretation of DAL formulas). *The* interpretation $\models$ *of* DAL *formulas with respect to state* ν *is defined as*

1. $\nu \models \theta_1 \geq \theta_2$ *iff* $val(\nu,\theta_1) \geq val(\nu,\theta_2)$*, and accordingly for* $=,\leq,<,>$.
2. $\nu \models \phi \wedge \psi$ *iff* $\nu \models \phi$ *and* $\nu \models \psi$*. For* $\neg,\vee,\rightarrow$*, the definition is accordingly.*
3. $\nu \models [\alpha]\phi$ *iff* $\omega \models \phi$ *for all states* ω *with* $(\nu,\omega) \in \rho(\alpha)$.
4. $\nu \models \langle\alpha\rangle\phi$ *iff* $\omega \models \phi$ *for some state* ω *with* $(\nu,\omega) \in \rho(\alpha)$.

Time Anomalies. Hybrid systems evolve along piecewise continuous trajectories, which consist of a sequence of continuous flows interrupted by discontinuous discrete jumps. A common phenomenon in hybrid system models is that their semantics and analysis is more controversial when discrete and continuous behaviour are allowed to interact without certain regularity assumptions [26, 42, 12, 23]. Zeno-anomalies occur when the hybrid system is allowed to take infinitely many discrete transitions in finite time.

Consider the DA-program $(a' = -1 \wedge d \leq a;\, d := d/2)^*$ starting in a state where $c > d > 0$ and c and d progress towards goal 0. The (inverse) clock variable a decreases continuously, yet d bounds the maximum duration of each continuous evolution phase. At the latest when $a = d$, variable d decreases by a discrete transition. This Zeno system generates infinitely many transitions in finite time and it is impossible for clock a to finally reach 0, because $a \geq d > 0$ will always remain true. Yet this behaviour is, in a certain sense, counterfactual, because it fails to obey divergence of time: Real time diverges, whereas clock a converges to 0. Further, systems with Zeno-anomalies cannot be realised [26, 42, 12, 23] so that corresponding regularity assumptions can be justified for practical purposes.

To avoid pitfalls of time anomalies, we define the DAL semantics so that it only refers to well-defined system behaviour with finitely many transitions in finite time: We restrict the semantics of DA-constraints and disallow infinite numbers of switches between differential equations in bounded time. With DA-constraint $\mathcal{D}$ defined as $(x \geq 0 \rightarrow x'' = -1) \wedge (x < 0 \rightarrow x'' = 1) \wedge y' = 1$, the DAL formula $\exists e\, \langle\mathcal{D}\rangle[\mathcal{D}](y > e \rightarrow x \leq d)$ expresses that, after some time, the system can stabilise such that it always remains within the region $x \leq d$ when $y > e$ for some choice of e. For such a stability property, we do not analyse what happens *after* there have been infinitely many switches from $x'' = 1$ to $x'' = -1$ within the first second. Instead, our semantics is such that our calculus reveals what happens for *any finite* number of switches. Accordingly, we restrict the semantics of DA-constraints to only accept non-Zeno evolutions:

Definition 12. *A state flow* φ *for a DA-constraint* $\mathcal{D}$ *is called* non-Zeno*, if there only is a finite number of points in time where some variable needs to*

obey another differential constraint of $\mathcal{D}$ than before the respective point in time: Let $\mathcal{D}_1 \vee \cdots \vee \mathcal{D}_n$ be a disjunctive normal form of $\mathcal{D}$, then flow $\varphi : [0, r] \to \mathrm{State}(\Sigma)$ is non-Zeno iff there are an $m \in \mathbb{N}$ and $0 = \zeta_0 < \zeta_1 < \cdots < \zeta_m = r$ and indices $i_1, \ldots, i_m \in \{1, \ldots, n\}$ such that φ respects $\mathcal{D}_{i_k}$ on the interval $[\zeta_{k-1}, \zeta_k]$, i.e., $\varphi|_{[\zeta_{k-1},\zeta_k]} \models \mathcal{D}_{i_k}$ for all $k \in \{1, \ldots, m\}$.

The semantics of DA-programs entails that runs with non-Zeno state flows are non-Zeno, because α^* does not accept infinitely many switches.

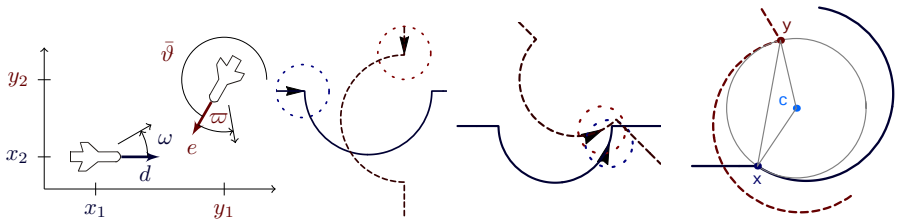
3 Collision Avoidance in Air Traffic Control

As a case study, which will serve as a running example, we show how succinctly collision avoidance maneuvers in air traffic control can be described in DAL. In Section 5, we will verify such maneuvers in the DAL calculus.

Flight Dynamics. Assuming, for simplicity, aircraft remain at the same altitude, an aircraft is described by its planar position $x = (x_1, x_2) \in \mathbb{R}^2$ and angular orientation ϑ. The dynamics of an aircraft is determined by its linear velocity $v \in \mathbb{R}$ and angular velocity ω, see Fig. 1a (with $\vartheta = 0$). When neglecting wind or gravitation, which is appropriate for analysing cooperation in air traffic control [47, 28, 29, 11, 34], the in-flight dynamics of an aircraft at x can be described by the following differential equation system, see, e.g., [47] for details:

$$x_1' = v\cos\vartheta \qquad x_2' = v\sin\vartheta \qquad \vartheta' = \omega \ . \tag{1}$$

Differential Axiomatisation. Unlike for straight-line flight ($\omega = 0$), such nonlinear dynamics is difficult to analyse [47, 28, 29, 11, 34] for $\omega \neq 0$, especially due to the trigonometric expressions which are generally undecidable. Solving (1) already requires the Floquet-theory of differential equations with periodic coefficients [48, Theorem 18.X] and yields mixed polynomial expressions with multiple trigonometric functions. A true challenge, however, is verifying properties of the states that the aircraft reach by following these solutions, which requires proving that complicated formulas with mixed polynomial arithmetic and trigonometric functions hold true for all values of state variables and all possible evolution durations. By Gödel's incompleteness theorem, however, the resulting first-order real arithmetic with trigonometric functions is not semidecidable, because the roots of sin characterise an isomorphic copy of natural numbers.



1a: Aircraft dynamics 1b: Roundabout 1c: Counterexample 1d: Tangential Roundabout

Fig. 1: Roundabout maneuvers for collision avoidance in air traffic control

To obtain polynomial dynamics, we axiomatise the trigonometric functions in the dynamics differentially and reparametrise the state correspondingly. Instead of angular orientation ϑ and linear velocity v, we use the linear speed vector $d = (d_1, d_2) := (v\cos\vartheta, v\sin\vartheta) \in \mathbb{R}^2$, which describes both the linear speed $\|d\| := \sqrt{d_1^2 + d_2^2} = v$ and orientation of the aircraft in space, see Fig. 1a. Substituting this coordinate change into (1), we immediately have $x_1' = d_1$ and $x_2' = d_2$. With the coordinate change, we further obtain differential equations for d_1, d_2 from differential equation system (1) by simple symbolic differentiation:

$$\begin{aligned} d_1' &= v'\cos\vartheta + v(-\sin\vartheta)\vartheta' = -(v\sin\vartheta)\omega = -\omega d_2 \\ d_2' &= v'\sin\vartheta + v(\cos\vartheta)\vartheta' \quad = \quad (v\cos\vartheta)\omega = \quad \omega d_1 \end{aligned}$$

The middle equality holds for constant linear velocity ($v' = 0$), which we assume, because only limited variations in linear speed are possible and cost-effective during the flight [47, 28] so that ω is the primary control parameter in air traffic control. Hence, equations (1) can be restated as the DA-constraint $\mathcal{F}(\omega)$:

$$x_1' = d_1 \wedge x_2' = d_2 \wedge d_1' = -\omega d_2 \wedge d_2' = \omega d_1 \qquad (\mathcal{F}(\omega))$$
$$y_1' = e_1 \wedge y_2' = e_2 \wedge e_1' = -\varpi e_2 \wedge e_2' = \varpi e_1 \qquad (\mathcal{G}(\varpi))$$

DA-constraint $\mathcal{F}(\omega)$ expresses that position x changes according to the linear speed vector d, which in turn rotates according to ω. Simultaneous movement together with a second aircraft at $y \in \mathbb{R}^2$ having linear speed $e \in \mathbb{R}^2$ (also indicated with angle $\bar{\vartheta}$ in Fig. 1a) and angular velocity ϖ corresponds to the DA-constraint $\mathcal{F}(\omega) \wedge \mathcal{G}(\varpi)$. Such DA-constraints capture simultaneous dynamics of multiple traffic agents succinctly using conjunction. By this differential axiomatisation, we thus obtain polynomial differential equations, even though their solutions still involve the same complicated nonlinear trigonometric expressions. Since the solutions involve trigonometric functions, previous approaches [49, 23, 17, 42, 12, 30–33] were not able to handle such dynamics.

Aircraft Collision Avoidance Maneuvers. Due to possible turbulence or collisions, a flight configuration is unsafe if another aircraft is within a protected zone of radius p, i.e., $\|x - y\|^2 < p^2$. Guiding aircraft by collision avoidance maneuvers to automatically resolve conflicting flight paths that would lead to possible loss of separation, is a major challenge both for air traffic control and verification [47, 28, 29, 14, 11, 34, 18, 25]. Several different classes of collision avoidance maneuvers for air traffic control have been suggested [47, 28, 29, 14, 18, 25]. The classical traffic alert and collision avoidance system (TCAS) [28] directs one aircraft on climbing routes the other on descending routes to resolve conflicts at different altitudes but keeps otherwise unmodified straight-line flight paths. While the simplistic TCAS maneuver has several benefits, it does not scale up easily to multiple aircraft or dense traffic situations nearby airports. As a more scalable alternative, Tomlin et al. [47] suggested roundabout maneuvers on circular paths, see Fig. 1b, where, even at the same altitude, several aircraft can participate in collision avoidance maneuvers. Because the continuous dynamics of curved flights with $\omega \neq 0$ is quite intricate, Tomlin et al. [47] and Massink and De Francesco [29] have analysed trapezoidal straight-line ($\omega = 0$) approximations of roundabouts, instead, which consist only of a series of two to five straight-line segments connected by several instant turns. Unfortunately, the discontinuities in instant turns are not flyable by aircraft.

As a more realistic model, we investigate curved roundabout maneuvers proposed by Tomlin et al. [47]. Roundabouts have proper flight curves with nonzero angular velocities ω (Fig. 1b). We have shown previously [34] that roundabout maneuvers with fixed turns [47, 28, 29, 11] are unsafe for non-orthogonal initial flight paths (see Fig. 1c for a counterexample) and we have proposed a tangential roundabout maneuver [34] with position-dependent evasive actions to overcome these deficiencies. However, because of general limits of numerical approximation techniques [34, 9], we could not actually verify the tangential roundabout maneuver numerically. In this paper, we introduce a generalised class of tangential roundabout maneuvers with curved flight paths and formally verify this maneuver in the purely symbolic DAL calculus. Our main motivation for studying roundabouts are their curved flight paths, which constitute a substantial challenge for verification of hybrid systems with nontrivial dynamics and an important part of realistic flight maneuvers.

Tangential Roundabout Maneuver. In the tangential roundabout maneuver, sketched in Fig. 1d, the idea is that the aircraft agree on some common angular velocity ω and common centre c around which both can circle safely without coming closer to each other (their linear velocities can differ, though, to compensate for different cruise speeds). Note that neither, c nor ω need to be discovered by complicated online trajectory predictions. Instead, we present in Section 5 a simple characterisation of safe choices for the parameters of the tangential roundabout maneuver and determine safety of the resulting flight paths using formal proofs in the DAL calculus.

In Fig. 2, we introduce the DAL model for the tangential roundabout maneuver, which is a simplified and more uniform generalisation of our previous work [34]. Observe how concisely complicated aircraft maneuvers can be specified in DAL. There, safety property ψ for aircraft maneuvers expresses that protected zones are respected during the flight (specified by separation property ϕ). The flight controller (trm^*) performs collision avoidance maneuvers by tangential roundabouts and repeats these maneuvers any number of times as needed. During each *trm* phase, the aircraft first perform arbitrary free flight (*free*) by (repeatedly) independently adjusting their angular velocities ω and ϖ while they are safely separated, which is expressed by conjunct ϕ of the DA-constraint. Observe that, unlike in $\exists u\,(\omega := u);\ \mathcal{F}(\omega)$, angular velocities can be (re-)adjusted continuously during free flight in $\exists\omega\,\mathcal{F}(\omega)$, rather than just once. In particular, *free* includes piecewise constant choices as in $(\exists u\,(\omega := u) \wedge \exists u\,(\varpi := u);\ \mathcal{F}(\omega) \wedge \mathcal{G}(\varpi))^*$. Due to invariant region ϕ of *free*, the tangential roundabout maneuver must be initiated (by a tangential initiation controller *entry*) before the flight paths become unsafe. Then, the tangential roundabout maneuver itself is carried out by the DA-constraint $\mathcal{F}(\omega) \wedge \mathcal{G}(\omega)$ according to some common angular velocity ω determined by *entry*. Finally, the collision avoidance roundabouts can be left again by repeating the loop trm^* and entering arbitrary free flight at any time. When further conflicts occur during free flight, the controller in Fig. 2 again enters roundabout conflict resolution maneuvers.

In summary, property ψ of Fig. 2 expresses that the aircraft remain safe during the flight, especially during evasive roundabout maneuvers. For the maneuver in Fig. 2, it is easy to see that ϕ also holds during *free*, because ϕ is specified as an invariant region of *free*. In this paper, we do not formally investigate temporal properties like "always ϕ", but refer to [32] for appropriate extensions of our logic. In Section 5, we will determine a constraint on the parameter adjustment

$$\begin{aligned}
\psi &\equiv \phi \to [trm^*]\phi \\
\phi &\equiv \|x-y\|^2 \geq p^2 \equiv (x_1-y_1)^2+(x_2-y_2)^2 \geq p^2 \\
trm &\equiv free;\ \ entry;\ \ \mathcal{F}(\omega) \wedge \mathcal{G}(\omega) \\
free &\equiv \exists\omega\, \mathcal{F}(\omega) \wedge \exists\varpi\, \mathcal{G}(\varpi) \wedge \phi \\
entry &\equiv \text{will be derived in Section 5}
\end{aligned}$$

Fig. 2: Flight control with tangential roundabout collision avoidance maneuvers

by *entry* such that the roundabout maneuver is safe, and we give a simple choice for *entry* respecting this parameter constraint.

4 Verification Calculus for Differential-Algebraic Logic

In this section, we introduce a sequent calculus for proving DAL formulas. The basic idea is to symbolically compute the effects of DA-programs and successively transform them into simpler logical formulas describing their effects. The calculus consists of standard propositional rules, dedicated rules for handling DA-program-modalities, including differential induction rules for sophisticated differential constraints, and side deduction rules for integrating real quantifier elimination.

In our calculus, we use *substitutions*: The result of applying to ϕ the substitution that replaces x by θ is defined as usual; it is denoted by ϕ_x^θ. Likewise, in a simultaneous substitution $\phi_{x_1}^{\theta_1}\ldots_{x_n}^{\theta_n}$ the x_i are replaced simultaneously by the respective θ_i, see [5] for details.

4.1 Derivations and Differentiation

As a purely algebraic device for proving properties about continuous evolutions in our calculus, we define syntactic derivations of terms and show that their valuation corresponds with analytic differentiation (the *total differential*). With this, we can build proof rules for verifying DA-programs fully algebraically by a differential form of induction without the need to carry out analytic reasoning about analytic limits or similar concepts that would require higher-order logic.

Definition 13 (Derivation). *The map* $D : \mathrm{Term}(\Sigma\cup\Sigma') \to \mathrm{Term}(\Sigma\cup\Sigma')$ *that is defined as follows is called* syntactic (total) derivation

$$D(r) = 0 \qquad \text{if } r \in \mathbb{Q} \text{ is a rational number} \tag{2a}$$

$$D(x^{(n)}) = x^{(n+1)} \qquad \text{if } x \in \Sigma \text{ is a state variable}, n \geq 0 \tag{2b}$$

$$D(a+b) = D(a) + D(b) \tag{2c}$$

$$D(a-b) = D(a) - D(b) \tag{2d}$$

$$D(a\cdot b) = D(a)\cdot b + a\cdot D(b) \tag{2e}$$

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

For a first-order formula F*, we define the following abbreviations:*

$$D(F) \equiv \bigwedge_{i=1}^{m} D(F_i) \qquad \text{where } \{F_1,\ldots,F_m\} \text{ is the set of all literals of } F$$

$$D(a \geq b) \equiv D(a) \geq D(b) \quad \text{and accordingly for } <, >, \leq, = \text{ or negative literals.}$$

To illustrate the naturalness of this definition, we briefly align it in terms of the structures from differential algebra and refer to [27] for details. Case (2a) defines number symbols as differential constants, which do not change during continuous evolution. Equation (2c) and the Leibniz rule (2e) are defining conditions for derivation operators on rings. Equation (2d) is a derived rule for subtraction according to $a-b=a+(-1)\cdot b$. In addition, equation (2b) uniquely defines D on the differential polynomial algebra spanned by the differential indeterminates $x\in\Sigma$. Equation (2f) canonically extends D to the differential field of quotients. As the base field $\mathbb{R}$ has no zero divisors, the right hand side of (2f) is defined whenever the division a/b can be carried out, which, as we assumed, is guarded by $b\neq 0$. The resulting structure $\mathrm{Term}(\Sigma\cup\Sigma')$, together with the derivation D, corresponds to the *differential field of rational fractions* with state variables as *differential indeterminates* over $\mathbb{R}$ and with rational numbers as *differential constants*.

The conjunctive definition of the formula $D(F)$ in Def. 13 corresponds to the joint total derivative of all atomic subformulas of F and will be an important tool for differential induction rules of our calculus.

The following central lemma, which is the differential counterpart of the substitution lemma, establishes the connection between syntactic derivation of terms and semantic differentiation as an analytic operation to obtain analytic derivatives of valuations along state flows. It will allow us to draw analytic conclusions about the behaviour of a system along differential equations from the truth of purely algebraic formulas obtained by syntactic derivation.

Lemma 1 (Derivation lemma). *The valuation of* DAL *terms is a* differential homomorphism*: Let $\theta\in\mathrm{Term}(\Sigma)$ and let $\varphi:[0,r]\to\mathrm{State}(\Sigma)$ be any state flow of the order of $D(\theta)$ and 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*

$$\frac{\mathsf{d}\,val(\varphi(t),\theta)}{\mathsf{d}t}(\zeta)=val(\bar\varphi(\zeta),D(\theta))\quad.$$

In particular, $val(\varphi(t),\theta)$ 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. 8). 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 non-zero during φ.

– If θ is a variable x, the conjecture holds immediately by Def. 8:

$$\frac{\mathsf{d}\,val(\varphi(t),x)}{\mathsf{d}t}(\zeta)=\frac{\mathsf{d}\,\varphi(t)(x)}{\mathsf{d}t}(\zeta)=\bar\varphi(\zeta)(x')=val(\bar\varphi(\zeta),D(x))\quad.$$

There, 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. 13), and valuations (Def. 6):

$$\begin{aligned}
&\frac{\mathsf{d}}{\mathsf{d}t}(val(\varphi(t), a+b))(\zeta) \\
&= \frac{\mathsf{d}}{\mathsf{d}t}(val(\varphi(t), a) + val(\varphi(t), b))(\zeta) && val(\nu,\cdot) \text{ homomorph for } + \\
&= \frac{\mathsf{d}}{\mathsf{d}t}(val(\varphi(t), a))(\zeta) + \frac{\mathsf{d}}{\mathsf{d}t}(val(\varphi(t), b))(\zeta) && \frac{\mathsf{d}}{\mathsf{d}t} \text{ is a (linear) derivation} \\
&= val(\bar{\varphi}(\zeta), D(a)) + val(\bar{\varphi}(\zeta), D(b)) && \text{by induction hypothesis} \\
&= val(\bar{\varphi}(\zeta), D(a) + D(b)) && val(\nu,\cdot) \text{ homomorph for } + \\
&= val(\bar{\varphi}(\zeta), D(a+b)) && D(\cdot) \text{ is a syntactic derivation}
\end{aligned}$$

– The case where θ is of the form $a \cdot b$ or $a - b$ is accordingly, using Leibniz product rule (2e) or subtractiveness (2d) of Def. 13, respectively.
– The case where θ is of the form a/b uses (2f) of Def. 13 and further depends on the assumption that $b \neq 0$ along φ. This 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 $D(r) = 0$. □

The *principle of substitution* [16] can be lifted to differential equations, i.e., differential equations can be used for equivalent substitutions along state flows respecting the corresponding differential constraints.

Lemma 2 (Differential substitution principle). *If φ is a state flow with $\varphi \models x_1' = \theta_1 \wedge \cdots \wedge x_n' = \theta_n \wedge \chi$, then $\varphi \models \mathcal{D} \leftrightarrow (\chi \rightarrow \mathcal{D}_{x_1'}^{\theta_1} \ldots_{x_n'}^{\theta_n})$ holds for all DA-constraints $\mathcal{D}$.*

Proof. By using the substitution lemma for first-order logic on the basis of $val(\bar{\varphi}(\zeta), x_i') = val(\bar{\varphi}(\zeta), \theta_i)$ and $\bar{\varphi}(\zeta) \models \chi$ at each time ζ in the domain of φ. □

The following lemma captures that the semantics of DA-constraints is not sensitive to how the DA-constraint is presented. It also plays its part in the soundness proof of our calculus, because it immediately makes all implicational and equivalence transformations of real-arithmetic available for DA-constraints.

Lemma 3 (Differential transformation principle). *Let $\mathcal{D}$ and $\mathcal{E}$ be DA-constraints (with the same changed variables). If $\mathcal{D} \rightarrow \mathcal{E}$ is a tautology of (non-differential) first-order real arithmetic (that is, when considering $x^{(n)}$ as a new variable independent from x), then $\rho(\mathcal{D}) \subseteq \rho(\mathcal{E})$.*

Proof. Let the first-order formulas ϕ and ψ be obtained from $\mathcal{D}$ and $\mathcal{E}$, respectively, by replacing all x' by new variable symbols X (accordingly for higher-order differential symbols $x^{(n)}$). Using vectorial notation, we write $\phi_X^{x'}$ for the formula obtained from ϕ by substituting all variables X by x'. Thus, $\phi_X^{x'}$ is $\mathcal{D}$ and $\psi_X^{x'}$ is $\mathcal{E}$. Let $\phi \rightarrow \psi$ be valid in (non-differential) real arithmetic. Let $(\nu, \omega) \in \rho(\mathcal{D})$ according to a state flow φ. Then φ also is a state flow for $\mathcal{E}$ that justifies $(\nu, \omega) \in \rho(\mathcal{E})$: For any $\zeta \in [0, r]$, we have $\bar{\varphi}(\zeta) \models \mathcal{D}$ hence $\bar{\varphi}(\zeta) \models \mathcal{E}$, because $\bar{\varphi}(\zeta) \models \phi_X^{x'}$ immediately implies $\bar{\varphi}(\zeta) \models \psi_X^{x'}$ by validity of $\phi \rightarrow \psi$. The assumption on $\mathcal{D}$ and $\mathcal{E}$ having the same set of changed variables is only required for compatibility with condition 2 of Def. 9, which enforces that unchanged variables z remain constant. It can be established easily by adding constraints of the form $z' = 0$ as required □

DA-constraints $\mathcal{D}$ and $\mathcal{E}$ are *equivalent* iff $\rho(\mathcal{D}) = \rho(\mathcal{E})$. In particular, the semantics of DA-programs is preserved when replacing a DA-constraint by another DA-constraint that is equivalent in non-differential first-order real arithmetic (similarly for DJ-constraints).

4.2 Differential Reduction and Differential Elimination

Using the expressive power of DA-constraints, several reductions can be performed to simplify the syntactic form of DA-constraints. With quantified DA-constraints, we can reduce differential inequalities to quantified differential equations equivalently:

Lemma 4 (Differential inequality elimination). *DA-constraints admit* differential inequality elimination*, i.e., to each DA-constraint $\mathcal{D}$, an equivalent DA-constraint without differential inequalities can be effectively associated that has no other free variables.*

Proof. Let $\mathcal{E}$ be obtained from $\mathcal{D}$ by replacing all differential inequalities $\theta_1 \leq \theta_2$ by a quantified differential equation $\exists u\,(\theta_1 = \theta_2 - u \wedge u \geq 0)$ with a new variable u for the quantified disturbance (accordingly for $\geq, >, <$). By Lemma 3, the equivalence of $\mathcal{D}$ and $\mathcal{E}$ is a simple consequence of the corresponding equivalences in first-order real arithmetic. □

In the sequel, we assume this transformation has been applied such that we can focus on DA-constraints with differential equations, i.e., where differential symbols only occur in differential equations, and where inequalities do not contain differential symbols. Yet, the DA-constraint resulting from Lemma 4 could become inhomogeneous when multiple differential equations are produced for the same variable that result from multiple differential inequalities. For instance, $\theta_1 \leq x' \leq \theta_2$ produces $\exists u\,\exists v\,(x' = \theta_1 + u \wedge x' = \theta_2 - v \wedge u \geq 0 \wedge v \geq 0)$. To rehomogenise this DA-constraint, we use the following:

Lemma 5 (Differential equation normalisation). *DA-constraints admit* differential equation normalisation*, i.e., to each DA-constraint $\mathcal{D}$, an equivalent DA-constraint with at most one differential equation for each differential symbol can be effectively associated that has no other free variables. Furthermore, this differential equation is* explicit*, i.e., of the form $x^{(n)} = \theta$ where $\operatorname{ord}_x \theta < n$.*

Proof. For each differential symbol $x^{(n)} \in \Sigma'$ occurring in $\mathcal{D}$, we introduce a new non-differential variable $X_n \in \Sigma$. Let $\mathcal{D}^{X_n}_{x^{(n)}}$ denote the result of substituting X_n for $x^{(n)}$ in $\mathcal{D}$. By Lemma 3, the equivalence of $\mathcal{D}$ and $\exists X_n\,(x^{(n)} = X_n \wedge \mathcal{D}^{X_n}_{x^{(n)}})$ is a simple consequence of the corresponding equivalence in first-order logic. Proceeding inductively for all such $x^{(n)} \in \Sigma'$ in $\mathcal{D}$ gives the desired result. □

Similarly, higher-order differential constraints reduce to first-order constraints by introducing new non-differential auxiliary variables X_n for each of the higher-order differential symbols $x^{(n)}$. For $1 \leq \operatorname{ord}_x \theta < n$, we can replace a higher-order differential equation $x^{(n)} = \theta$ by:

$$x' = X_1 \wedge X_1' = X_2 \wedge \ldots \wedge X_{n-2}' = X_{n-1} \wedge X_{n-1}' = \theta^{X_1}_{x'} \ldots {}^{X_{n-1}}_{x^{(n-1)}}\, {}^{X_{n-1}'}_{x^{(n)}}$$

4.3 Rules of the Calculus for Differential-Algebraic Logic

A *sequent* is of the form $\Gamma \vdash \Delta$, where the *antecedent* Γ and *succedent* Δ are finite sets of formulas. Its semantics is that of the formula $\bigwedge_{\phi\in\Gamma}\phi \rightarrow \bigvee_{\psi\in\Delta}\psi$, and a sequent $\Gamma \vdash \Delta$ will simply be treated as an abbreviation for that formula.

The DAL calculus uses simultaneous substitutions $\phi^{\theta_1}_{x_1}\ldots^{\theta_n}_{x_n}$ that take effect within formulas and programs [5]. In the DAL calculus, only admissible substitutions are applicable, which is crucial for soundness. We assume α-conversion for renaming as needed.

Definition 14 (Admissible substitution). *An application of a substitution σ is* admissible *if no replaced variable x occurs in the scope of a quantifier or modality binding x or a variable of the replacement σx. A modality* binds *x if its DA-program (possibly) changes x, i.e., if it contains a DJ-constraint containing $x := \theta$ or a DA-constraint containing $x^{(n)} \in \Sigma'$ for an $n \geq 1$.*

As usual in sequent calculus—although the direction of entailment is from *premisses* (above rule bar) to *conclusion* (below)—the order of reasoning and reading is *goal-directed* in practice: Rules are applied in tableau-style, that is, starting from the desired conclusion at the bottom (goal) to the resulting premisses (subgoals). To highlight the logical essence of the DAL calculus, Fig. 3 provides rule *schemata* to which the following definition associates the calculus rules that are applicable during a DAL proof. The calculus consists of propositional rules (P-rules: P1–P10), first-order quantifier rules (F-rules: F1–F4), rules for dynamic modalities (D-rules: D1–D15), and global rules (G-rules: G1–G6).

Definition 15 (Rules). *The* rule schemata *in Fig. 3 induce* calculus rules *by:*

1. *If*
$$\frac{\Phi_1 \vdash \Psi_1 \quad \ldots \quad \Phi_n \vdash \Psi_n}{\Phi_0 \vdash \Psi_0}$$
is an instance of one of the rule schemata in Fig. 3, then
$$\frac{\Gamma,\Phi_1 \vdash \Psi_1,\Delta \quad \ldots \quad \Gamma,\Phi_n \vdash \Psi_n,\Delta}{\Gamma,\Phi_0 \vdash \Psi_0,\Delta}$$
can be applied as a proof rule of the DAL *calculus, where Γ,Δ are arbitrary finite sets of context formulas (including empty sets).*
2. *Symmetric schemata can be applied on either side of the sequent: If*
$$\frac{\phi_1}{\phi_0}$$
is an instance of one of the symmetric rule schemata D1–D12 in Fig. 3, then
$$\frac{\Gamma \vdash \phi_1,\Delta}{\Gamma \vdash \phi_0,\Delta} \qquad \textit{and} \qquad \frac{\Gamma,\phi_1 \vdash \Delta}{\Gamma,\phi_0 \vdash \Delta}$$
can both be applied as proof rules of the DAL *calculus, where Γ,Δ are arbitrary finite sets of context formulas (including empty sets).*

P-Rules. For propositional logic, standard P-rules are listed in Fig. 3. Rule P10 is a cut rule that can be used for case distinctions for any additional formula ϕ. We use cuts to derive simple rule dualities. Typically, they are not needed in practical verification examples from traffic domains [31, 33].

$$\text{(P1)}\ \frac{\phi\vdash}{\vdash\neg\phi}\quad \text{(P3)}\ \frac{\vdash\phi,\psi}{\vdash\phi\vee\psi}\quad \text{(P5)}\ \frac{\vdash\phi\quad\vdash\psi}{\vdash\phi\wedge\psi}\quad \text{(P7)}\ \frac{\phi\vdash\psi}{\vdash\phi\to\psi}\quad \text{(P9)}\ \frac{}{\phi\vdash\phi}$$

$$\text{(P2)}\ \frac{\vdash\phi}{\neg\phi\vdash}\quad \text{(P4)}\ \frac{\phi\vdash\quad\psi\vdash}{\phi\vee\psi\vdash}\quad \text{(P6)}\ \frac{\phi,\psi\vdash}{\phi\wedge\psi\vdash}\quad \text{(P8)}\ \frac{\vdash\phi\quad\psi\vdash}{\phi\to\psi\vdash}\quad \text{(P10)}\ \frac{\vdash\phi\quad\phi\vdash}{\vdash}$$

$$\text{(F1)}\ \frac{\mathrm{QE}(\forall x\ \bigwedge_i(\Gamma_i\vdash\Delta_i))}{\Gamma\vdash\Delta,\forall x\,\phi}\qquad \text{(F3)}\ \frac{\mathrm{QE}(\exists x\ \bigwedge_i(\Gamma_i\vdash\Delta_i))}{\Gamma\vdash\Delta,\exists x\,\phi}$$

$$\text{(F2)}\ \frac{\mathrm{QE}(\exists x\ \bigwedge_i(\Gamma_i\vdash\Delta_i))}{\Gamma,\forall x\,\phi\vdash\Delta}\qquad \text{(F4)}\ \frac{\mathrm{QE}(\forall x\ \bigwedge_i(\Gamma_i\vdash\Delta_i))}{\Gamma,\exists x\,\phi\vdash\Delta}$$

$$\text{(D1)}\ \frac{\langle\alpha\rangle\langle\beta\rangle\phi}{\langle\alpha;\beta\rangle\phi}\quad \text{(D5)}\ \frac{\exists x\,\langle\mathcal{J}\rangle\phi}{\langle\exists x\,\mathcal{J}\rangle\phi}\quad \text{(D9)}\ \frac{\chi\wedge\phi^{\theta_1\ldots\theta_n}_{x_1\ldots x_n}}{\langle x_1:=\theta_1\wedge\ldots\wedge x_n:=\theta_n\wedge\chi\rangle\phi}$$

$$\text{(D2)}\ \frac{[\alpha][\beta]\phi}{[\alpha;\beta]\phi}\quad \text{(D6)}\ \frac{\forall x\,[\mathcal{J}]\phi}{[\exists x\,\mathcal{J}]\phi}\quad \text{(D10)}\ \frac{\chi\to\phi^{\theta_1\ldots\theta_n}_{x_1\ldots x_n}}{[x_1:=\theta_1\wedge\ldots\wedge x_n:=\theta_n\wedge\chi]\phi}$$

$$\text{(D3)}\ \frac{\langle\alpha\rangle\phi\vee\langle\beta\rangle\phi}{\langle\alpha\cup\beta\rangle\phi}\quad \text{(D7)}\ \frac{\langle\mathcal{J}_1\cup\ldots\cup\mathcal{J}_n\rangle\phi}{\langle\mathcal{J}\rangle\phi}\quad \text{(D11)}\ \frac{\langle(\mathcal{D}_1\cup\ldots\cup\mathcal{D}_n)^*\rangle\phi}{\langle\mathcal{D}\rangle\phi}$$

$$\text{(D4)}\ \frac{[\alpha]\phi\wedge[\beta]\phi}{[\alpha\cup\beta]\phi}\quad \text{(D8)}\ \frac{[\mathcal{J}_1\cup\ldots\cup\mathcal{J}_n]\phi}{[\mathcal{J}]\phi}\quad \text{(D12)}\ \frac{[(\mathcal{D}_1\cup\ldots\cup\mathcal{D}_n)^*]\phi}{[\mathcal{D}]\phi}$$

$$\text{(D13)}\ \frac{\vdash[\mathcal{E}]\phi}{\vdash[\mathcal{D}]\phi}\quad \text{(D14)}\ \frac{\vdash\langle\mathcal{D}\rangle\phi}{\vdash\langle\mathcal{E}\rangle\phi}\quad \text{(D15)}\ \frac{\vdash[\mathcal{D}]\chi\quad\vdash[\mathcal{D}\wedge\chi]\phi}{\vdash[\mathcal{D}]\phi}$$

$$\text{(G1)}\ \frac{\vdash\forall^\alpha(\phi\to\psi)}{[\alpha]\phi\vdash[\alpha]\psi}\quad \text{(G2)}\ \frac{\vdash\forall^\alpha(\phi\to\psi)}{\langle\alpha\rangle\phi\vdash\langle\alpha\rangle\psi}\quad \text{(G3)}\ \frac{\vdash\forall^\alpha(\phi\to[\alpha]\phi)}{\phi\vdash[\alpha^*]\phi}\quad \text{(G4)}\ \frac{\vdash\forall^\alpha(\varphi(x)\to\langle\alpha\rangle\varphi(x-1))}{\exists v\,\varphi(v)\vdash\langle\alpha^*\rangle\exists v{\leq}0\,\varphi(v)}$$

$$\text{(G5)}\ \frac{\vdash\forall^\alpha\forall y_1\ldots\forall y_k\,(\chi\to F'^{\theta_1\ldots\theta_n}_{x'_1\ldots x'_n})}{[\exists y_1\ldots\exists y_k\,\chi]F\vdash[\exists y_1\ldots\exists y_k\,(x'_1=\theta_1\wedge\ldots\wedge x'_n=\theta_n\wedge\chi)]F}$$

$$\text{(G6)}\ \frac{\vdash\exists\varepsilon{>}0\,\forall^\alpha\forall y_1\ldots y_k\,(\neg F\wedge\chi\to(F'\geq\varepsilon)^{\theta_1\ldots\theta_n}_{x'_1\ldots x'_n})}{[\exists y_1\ldots y_k\,(x'_1=\theta_1\wedge\ldots\wedge x'_n=\theta_n\wedge\sim F)]\chi\vdash\langle\exists y_1\ldots y_k\,(x'_1=\theta_1\wedge\ldots\wedge x'_n=\theta_n\wedge\chi)\rangle F}$$

In all rule schemata, *all* substitutions need to be admissible. In D7–D8, $\mathcal{J}_1\vee\cdots\vee\mathcal{J}_n$ is a disjunctive normal form of the DJ-constraint $\mathcal{J}$. In D11–D12, $\mathcal{D}_1\vee\cdots\vee\mathcal{D}_n$ is a disjunctive normal form of the DA-constraint $\mathcal{D}$. The rules D9–D10 and G5–G6 can be applied for any reordering of the conjuncts of the DA-constraint or DJ-constraint, where χ is non-differential or jump-free, respectively. In D13 and D14, $\mathcal{D}$ implies $\mathcal{E}$, i.e., satisfies the assumptions of Lemma 3. In G5–G6, F is first-order without negative equations, and F' abbreviates $D(F)$, with z' replaced by 0 for unchanged variables. In G6, F does not contain equations and the differential equations are Lipschitz-continuous. For F-rules, the $\Gamma_i\vdash\Delta_i$ are obtained from the resulting sub-goals of a side deduction, see ($\star$) in Fig. 4. The side deduction is started from the goal $\Gamma\vdash\Delta,\phi$ at the bottom (or $\Gamma,\phi\vdash\Delta$ for F2 and F4), where x is assumed not to occur in Γ,Δ using renaming. In the resulting sub-goals $\Gamma_i\vdash\Delta_i$, variable x is assumed to occur in first-order formulas only, as quantifier elimination (QE) is then applicable.

Fig. 3: Rule schemata of the DAL proof calculus

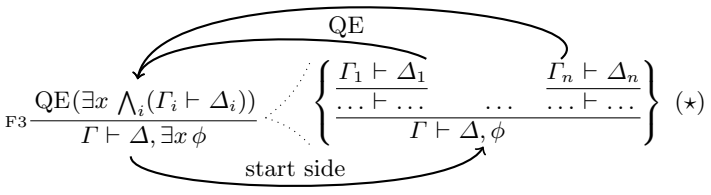


Fig. 4: Side deduction for quantifier elimination rules

Journal of Logic and Computation, 20(1), pp. 309-352, 2010.

doi:10.1093/logcom/exn070

F-Rules. Unlike in uninterpreted first-order logic [15, 16], quantifier rules have to respect the specific semantics of real arithmetic. Thus, our rules handle real quantifiers using quantifier elimination (QE) over the reals [8]. Unfortunately, QE is only defined in first-order real arithmetic and cannot handle DAL-modalities, where variables evolve along hybrid trajectories over time. We establish compatibility with dynamic modalities using side deductions for the F-rules, as illustrated in Section 4.4. Alternatively, the F-rules can be replaced by the quantifier rules of our companion paper [33], which generalise free variables, Skolemisation, and Deskolemisation to real arithmetic for integrating quantifier elimination with modal rules. Instead, here, we use side deductions that we have introduced in previous work [31] as a very intuitive and simple approach for handling real quantifiers.

D-Rules. The D-rules transform a DA-program into simpler logical formulas. Rules D1–D4 are as in discrete dynamic logic [22, 5]. D7 and D8 normalise DJ-constraints to their disjunctive normal form such that the jump alternatives can be read off easily. Similarly, D5 and D6 lift quantified choices in DJ-constraints to DAL quantifiers, which are, in turn, handled by F-rules. Then, D9 and D10 use generalised simultaneous substitutions [5] for handling discrete change and check the jump-free constraint χ.

Likewise, D11–D12 normalise DA-constraints to a form where their differential evolution alternatives are readily identifiable. Unlike for D7–D8, however, continuous evolutions take time so that the system can switch back and forth between the various cases of the DA-constraint, hence the repetition. Observe that finitely many repetitions are sufficient for non-Zeno flows (Def. 12), which can only switch finitely often in finite time.

Rules D13–D15 are weakening and strengthening rules for DA-constraints, respectively. In D13–D14, $\mathcal{D}$ implies $\mathcal{E}$ in real arithmetic according to Lemma 3, which is easy to decide by QE in practice. Note that D13–D14, are sound for *any* such combination of $\mathcal{D}$ and $\mathcal{E}$. Their primary practical purpose is to use D13 for overapproximating individual variable evolutions and D14 for refining nondeterministic variable evolutions to specific differential equations. In particular, we use D13 to project conjunctive differential constraints $\mathcal{D}$ to their non-differential constraints. As we illustrate in Section 5, this gives a powerful verification technique in combination with strengthening (D15), which allows to refine the system dynamics by auxiliary constraints. We address the problem of automatically determining the respective strengthenings χ in a follow-up paper [35], where we derive automatic verification algorithms from the results presented in this paper. Furthermore, D13–D14 make all equivalence transformations on DA-constraints from Section 4.2 available as proof rules, including index reduction techniques for differential-algebraic equations [19].

Note that DAL does not need rules for handling negation in DA-constraints or DJ-constraints, as—possibly after applying D7–D8 or D11–D12, respectively—negations only occur in jump-free or non-differential parts, because assignments and differential symbols only occur positively by Def. 1 and 2. Similarly, no rules for universal quantifiers within DA-constraints or DJ-constraints are needed. Like other propositional operators or quantifiers, negation and universal quantifiers are allowed without restriction in non-differential or jump-free χ and are then handled by D9–D10 or G5–G6 as usual.

G-Rules. The G-rules are global rules. They depend on the truth of their premisses in all states, which is ensured by the universal closure with respect to all free variables. If $x_1, \ldots, x_n$ are the free variables of Φ, then $\forall x_1 \ldots \forall x_n \, \Phi$ is its *universal closure*. Since it can be restricted to the variables bound in the DA-program α of the respective rule (see Def. 14), we denote it by $\forall^\alpha \Phi$. The G-rules are given in a form that best displays their underlying logical principles. The general pattern for applying G-rules to prove that the succedent of their conclusion holds is to prove that both their premiss and the antecedent of their conclusion hold.

G1–G2 are generalisation rules. G3 is a discrete induction schema for repetitions with inductive invariant ϕ. Similarly, G4 is a generalisation of Harel's convergence rule [22] to the hybrid case with decreasing variant φ. G3 says that ϕ holds after any number of repetitions of α, if it holds initially and remains true after each execution of α. G4 expresses that φ holds for some real number ≤ 0 after repeating α sufficiently often, if $\varphi(v)$ holds for some real number at all and decreases after every execution of α by 1 (or at least any other positive real constant).

G5 is a rule for *differential induction*, which is a continuous form of induction along differential constraints. The induction rules G3 and G5 (or G4 and G6 respectively) differ in the way the invariant is sustained. G3 uses the inductive nature of repetition. G5, instead, uses continuity of evolution and the differential equation for a continuous induction step with the *differential invariant* F: If F holds initially (antecedent of conclusion) and its total differential F' satisfies the same relations when taking into account the differential constraints (premiss), then F itself is sustained differentially (succedent of conclusion). Formula F' abbreviates $D(F)$ with z' replaced by 0 for all variables z that are unchanged by the DA-constraint, i.e., that are distinct from $\{x_1, \ldots, x_n\}$, because these are assumed constant in the semantics. By α-renaming, the y_i do not occur in F. Rule G6 is a *differential variant* rule where the variant F is attained differentially (with some minimal progress ε), rather than sustained as in G5. Differential induction, the requirement of the differential equations for G6 to be Lipschitz-continuous, and the notations $F' \geq \varepsilon$ and $\sim F$ will be illustrated in more detail in Sections 4.5–4.6 after side deductions for quantifiers have been explained in Section 4.4. Finally, G-rules can be combined with generalisation (G1–G2) to strengthen postconditions as needed.

Definition 16 (Provability). *A formula ψ is* provable *from a set Φ of formulas, denoted by $\Phi \vdash_{\text{DAL}} \psi$ iff there is a finite set $\Phi_0 \subseteq \Phi$ for which the sequent $\Phi_0 \vdash \psi$ is derivable. Derivability is inductively defined so that a sequent $\Phi \vdash \Psi$ is* derivable *iff there is a proof rule of the* DAL *calculus (Def. 15) with conclusion $\Phi \vdash \Psi$ such that all premisses of the rule are derivable.*

4.4 Deduction Modulo by Side Deduction

The F-rules constitute a purely modular interface to mathematical reasoning. They can use any theory that admits quantifier elimination and has a decidable ground theory, e.g., the theory of first-order real arithmetic, which is equivalent to the theory of real-closed fields [8]. Unlike in deduction modulo approaches of Dowek et al. [13] and Tinelli [46], the information given to the background prover is not restricted to ground formulas [46] or atomic formulas [13], and the effect of modalities has to be taken into account.

Definition 17 (Quantifier elimination). *A first-order theory admits* quantifier elimination *if to each formula ϕ, a quantifier-free formula $\mathrm{QE}(\phi)$ can be effectively associated that is equivalent (i.e., $\phi \leftrightarrow \mathrm{QE}(\phi)$ is valid) and has no other free variables. The operation* QE *is further assumed to evaluate ground formulas (i.e., without variables), yielding a decision procedure for this theory.*

Integrating quantifier elimination to deal with statements about real quantities is quite challenging in the presence of modalities that influence the value of variables and terms. Real quantifier elimination cannot be applied to formulas with mixed quantifiers and modalities like $\exists x\,[x' = -x; x := 2x]x \leq 5$. To find out which first-order constraints are actually imposed on x by this DAL formula, we have to take into account how x evolves from $\exists x$ to $x \leq 5$ along the hybrid system dynamics. Hence, our calculus first unveils the first-order constraints on x before applying QE. To achieve this in a concise and simple way, we use side deductions that we have introduced in previous work [31].

The effect of a side deduction is as follows. First, the DAL calculus discovers all relevant first-order constraints from modal formulas using a side deduction. Secondly, these constraints are reimported into the main proof and equivalently reduced using QE and the main proof continues. For instance, an application of F3 to a sequent $\Gamma \vdash \Delta, \exists x\,\phi$ starts a side deduction (marked $(\star)$ in Fig. 4) with the unquantified kernel $\Gamma \vdash \Delta, \phi$ as a goal at the bottom. This side deduction is carried out in the DAL calculus until x no longer occurs within modal formulas of the remaining open branches $\Gamma_i \vdash \Delta_i$ of $(\star)$. Once all occurrences of x are in first-order formulas, the resulting sub-goals $\Gamma_i \vdash \Delta_i$ of $(\star)$ are copied back to the main proof and QE is applied to eliminate x altogether (which determines the resulting sub-goal of rule F3 on the upper left side of Fig. 4). The remaining modal formulas not containing x can be considered as atoms for this purpose, as they do not impose constraints on x. Formally, this can be proven using a simple extension of the classical coincidence lemma: The truth-value of a formula only depends on the value of variables that actually occur in it, see [33] for details. When several quantifiers are nested, side deductions will be nested in a cascade, as they can again spawn further side deductions. According to the applicability conditions of F-rules, inner nested side deductions need to be completed by QE before outer deductions continue. For instance, further side deductions started within $(\star)$ of Fig. 4 will be completed before $(\star)$ continues and the quantifier elimination result of $(\star)$ is returned to the main F3 application.

Example 1 (Aircraft progress). To illustrate how our calculus combines arithmetic with dynamic reasoning using side deductions, we look at an aircraft example. Using the notation from Section 3, the following DAL formula expresses a simple progress property about aircraft: The aircraft at x can finally fly beyond any point $p \in \mathbb{R}^2$ by adjusting its speed vector d appropriately, using only speed vectors $d \in \mathbb{R}^2$ of bounded speed $\|d\| \leq b$, i.e., $\|d\|^2 \leq b^2 \equiv d_1^2 + d_2^2 \leq b^2$:

$$\forall p\,\exists d\,(\|d\|^2 \leq b^2 \wedge \langle \mathcal{F}(0) \rangle (x_1 \geq p_1 \wedge x_2 \geq p_2)) \ . \tag{3}$$

There, point p is constant during the evolution, i.e., $p_1' = p_2' = 0$ and $b' = 0$. The DAL proof in Fig. 5 proves this property using nested side deductions for nested quantifiers and differential variant induction G6. Applying F1 in the main branch yields a side deduction for $\forall p$, which, in turn, yields another side deduction by applying F3 for the nested quantifier $\exists d$. These nested side deductions

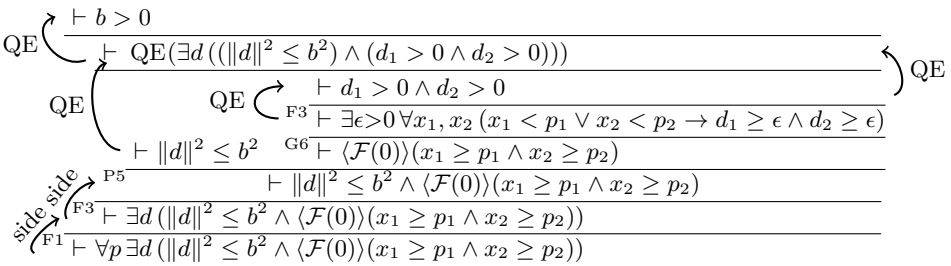


Fig. 5: Nested side deductions and differential variants for progress property

in Fig. 5 are inlined and indicated by indenting the side deductions, with arrows pointing to the start of the respective inner side deduction and back to the continuation of the outer deduction (marked with QE as in Fig. 4). The two branches for the side deduction for F3 recombine conjunctively and, after quantifiers are re-added, quantifier elimination yields $b > 0$, which reveals the parameter constraint on the speed bound b. Consequently, property (3) holds true and the proof closes for all non-zero speed bounds. The right branch of this F3 side deduction uses differential variant induction G6, as will be illustrated in Section 4.6. There, the quantifiers for x_1, x_2 result from the universal closure $\forall^\alpha$ in G6. The subsequent innermost F3 side deduction can be abbreviated by directly applying QE, because the affected formula already is first-order.

Like the other aircraft examples in this paper, formula (3) is provable in our theorem prover [36] within a few seconds, despite the complicated underlying aircraft dynamics.

4.5 Differential Induction with Differential Invariants

The purpose of G5 and G6 is to prove properties about continuous evolutions by differential induction using differential invariants or differential variants, respectively. They directly work with the differential constraints instead of complicated (possibly undecidable) arithmetic of their solutions. Unlike approaches using solutions [17, 30–33], differential induction can even be used to verify systems with nondeterministic quantified input, which would otherwise cause quantified higher-order functions for the time-dependent input of the solutions. Further, unlike in discrete induction, these differential induction rules exploit continuity of evolution and knowledge of the differential constraints for a continuous induction step. We demonstrate the capabilities and the necessity of the requirements of differential induction rules in a series of examples and counterexamples.

Rule G5 uses differential induction to prove that F is a *differential invariant*, i.e., F is closed under total differentiation (Def. 13) relative to the differential constraints. For this, the premiss of G5 shows that the total differential F'—i.e., $D(F)$ with z' replaced by 0 for unchanged variables z—holds within invariant region χ, when substituting the differential equations into F'. Now, if F holds initially (antecedent of conclusion), then F itself is sustained (succedent of conclusion). Intuitively, the premiss expresses that, within χ, the total derivative F' along the

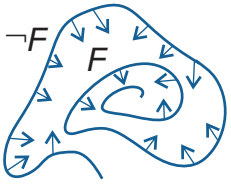


Fig. 6: Differential invariants

Journal of Logic and Computation, 20(1), pp. 309-352, 2010.

doi:10.1093/logcom/exn070

differential constraints is pointing inwards or transversal to F but never outwards to $\neg F$, see Fig. 6. At this point, it is important to note that, even though meta-proofs about DAL involve analytic reasoning, proofs within the DAL calculus are fully algebraic, including the handling of differential constraints by G5. Further observe that the premiss of G5 is a well-formed DAL formula, because all differential symbols are replaced by non-differential terms when forming $F'^{\theta_1 \ldots \theta_n}_{x'_1 \ldots x'_n}$.

Example 2 (Linear versus angular speed). Consider the following simple proof, which shows that the speed v of an aircraft at x is maintained, even when it changes its angular velocity ω nondeterministically during the flight (as in mode *free* of Fig. 2).

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash \mathrm{QE}(\forall x_1, x_2\, \forall d_1, d_2\, \forall \omega\, (2d_1(-\omega d_2) + 2d_2 \omega d_1 = 0))}\,\mathrm{QE}^{*}}{\vdash \forall x_1, x_2\, \forall d_1, d_2\, \forall \omega\, (2d_1(-\omega d_2) + 2d_2 \omega d_1 = 0)}\,\mathrm{F1}}{d_1^2 + d_2^2 = v^2 \vdash [\exists \omega\, \mathcal{F}(\omega)]\, d_1^2 + d_2^2 = v^2}\,\mathrm{G5}}{\vdash d_1^2 + d_2^2 = v^2 \to [\exists \omega\, \mathcal{F}(\omega)]\, d_1^2 + d_2^2 = v^2}\,\mathrm{P7}}{\vdash \forall v\, (d_1^2 + d_2^2 = v^2 \to [\exists \omega\, \mathcal{F}(\omega)]\, d_1^2 + d_2^2 = v^2)}\,\mathrm{F1}\ \mathrm{side}$$

The total derivative is $F' \equiv D(d_1^2 + d_2^2 = v^2) \equiv 2d_1 d'_1 + 2d_2 d'_2 = 2vv'$. Substituting the differential equations yields $F'^{-\omega d_2\ \omega d_1\ 0}_{d'_1\ d'_2\ v'} \equiv 2d_1(-\omega d_2) + 2d_2 \omega d_1 = 0$, which is valid and closes by quantifier elimination. This example shows the difference of differential continuous evolution (of d_1, d_2) and nondeterministic continuous evolution (of ω). The DA-constraint specifies how the d_i evolve along differential equations, hence d'_i is substituted in F'. For ω, instead, the DA-constraint is nondeterministic ($\exists \omega$) and does not specify how ω changes precisely. In particular, there is no equation for ω' that could be used for substition. Yet such an equation is not even needed for forming the premiss of G5, because, after α-renaming, ω cannot occur in F here, since the scope of $\exists \omega$ ends with the DA-constraint and does not extend to postcondition F. In the proof, the quantifiers for x_i and d_i result from the universal closure $\forall^\alpha$ in G5. The quantifier for ω is introduced by G6 and ensures that *all* possible evolutions of ω are taken into account as there is no specific equation for ω'. Finally note that in such cases without existential variables, side deductions can be inlined, see [33] for formal details.

Counterexample 3. For soundness of differential induction, it is crucial that Def. 13 defines $D(F \vee G)$ conjunctively as $D(F) \wedge D(G)$ instead of $D(F) \vee D(G)$. From an initial state ν which satisfies $\nu \models F$, hence $\nu \models F \vee G$, the formula $F \vee G$ only is sustained differentially if F itself is a differential invariant, not if G is. For instance, $x_1 \geq 0 \vee d_1^2 + d_2^2 = v^2$ is no differential invariant of $\exists \omega\, \mathcal{F}(\omega)$, because $x_1 \geq 0$ can be invalidated by appropriate curved flights, see formula (3). In practice, splitting differential induction proofs over disjunctions can be useful.

Counterexample 4 (Restricting differential invariance). It may be tempting to suspect that, in G5, the differential invariant F only needs to be differentially inductive at the states where F actually holds true. The differential induction needs to hold in a neighbourhood, though, such that adding F (or the border of F) to the assumptions in the premiss of G5 would be unsound! Consider the following counterexample where region $x^2 \leq 0$ is actually left immediately when

Journal of Logic and Computation, 20(1), pp. 309-352, 2010.

doi:10.1093/logcom/exn070

following $x' = 1$, which also demonstrates unsoundness of other approaches [39], including recent work by Gulwani and Tiwari [20]:

$$\dfrac{\dfrac{*\ \text{(unsound)}}{\vdash \forall x\,(x^2 \leq 0 \rightarrow 2x \leq 0)}}{x^2 \leq 0 \vdash [x' = 1]x^2 \leq 0}$$

If, however, F describes an open set (e.g., F only involves strict inequalities), then G5 is sound even when adding F to the assumptions of the premiss; see Appendix A for a proof. Likewise F can be added to the assumptions of the premiss when strengthening F' to strict inequalities, see Appendix A. If polynomial solutions exist, they can be used as differential invariants. Furthermore, differential strengthening (D15) can be an extraordinarily successful proof technique for successively enriching invariant regions by derived invariants until F itself becomes differentially inductive, as we illustrate in Section 5.

Counterexample 5 (Negative equations). It is crucial for soundness of differential induction that F is not allowed to contain negative equations. In the following counterexample, variable x can reach $x = 0$ without its derivative every being 0.

$$\dfrac{\dfrac{*\ \text{(unsound)}}{\vdash \forall x\,(1 \neq 0)}}{x \neq 0 \vdash [x' = 1]x \neq 0}$$

If, instead, both $x < 0$ and $x > 0$ are differential invariants of a system (e.g., of $x' = x$), then $x \neq 0$ can be proven indirectly by encoding it as $x < 0 \vee x > 0$.

A useful special case of D13 is the following derived weakening rule:

Lemma 6 (Differential weakening). *The following is a derived rule (where χ is non-differential):*

$$\text{(D13')}\quad \dfrac{\vdash \forall^{\alpha} \forall y_1 \ldots \forall y_k\,(\chi \rightarrow \phi)}{\vdash [\exists y_1 \ldots \exists y_k\,(x_1' = \theta_1 \wedge \cdots \wedge x_n' = \theta_n \wedge \chi)]\phi}$$

Proof. D13' clearly is sound, because χ is true along all state flows of the DA-constraint and ϕ is a consequence of χ by premiss. It can be derived as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdash \mathrm{QE}(\forall y_1 \ldots \forall y_k \forall d_1 \ldots \forall d_n\,(\chi_{x_1}^{d_1 \ldots d_n} \rightarrow \phi_{x_1}^{d_1 \ldots d_n}))}{\vdash \chi_{x_1}^{d_1 \ldots d_n} \rightarrow \phi_{x_1}^{d_1 \ldots d_n}}\,\mathrm{QE}}{\vdash [x_1 := d_1 \wedge \cdots \wedge x_n := d_n \wedge \chi_{x_1}^{d_1 \ldots d_n}]\phi}\,\mathrm{D10}}{\vdash \forall y_1 \ldots \forall y_k \forall d_1 \ldots \forall d_n\,([x_1 := d_1 \wedge \cdots \wedge x_n := d_n \wedge \chi_{x_1}^{d_1 \ldots d_n}]\phi)}\,\mathrm{F1}}{\vdash [\exists y_1 \ldots \exists y_k \exists d_1 \ldots \exists d_n\,(x_1 := d_1 \wedge \cdots \wedge x_n := d_n \wedge \chi_{x_1}^{d_1 \ldots d_n})]\phi}\,\mathrm{D6}}{\vdash [\exists y_1 \ldots \exists y_k \exists d_1 \ldots \exists d_n\,(x_1' = d_1 \wedge \cdots \wedge x_n' = d_n \wedge \chi)]\phi}\,\mathrm{D13}}{\vdash [\exists y_1 \ldots \exists y_k\,(x_1' = \theta_1 \wedge \cdots \wedge x_n' = \theta_n \wedge \chi)]\phi}\,\mathrm{D13}$$

The second application of D13 uses that fully nondeterministic continuous state change is equivalent to fully nondeterministic discrete state change, as they generate the same transitions. Finally, $\chi \rightarrow \phi$ can be obtained by α-renaming. □

Differential invariants enjoy structural closure properties. They are closed under conjunction (because of the conjunctive definition in Def. 13) and closed under differentiation.

Journal of Logic and Computation, 20(1), pp. 309-352, 2010.

doi:10.1093/logcom/exn070

Lemma 7. *Differential invariants are closed under differentiation: The total derivative of a differential invariant is an invariant of the same DA-constraint.*

Proof. Let F be a differential invariant, i.e., satisfy G5 for some DA-constraint of the form $\exists y\,(x' = \theta \wedge \chi)$, using vectorial notation for x and y. Hence, the premiss of G5 is provable: $\forall x\,\forall y\,(\chi \to {F'}^{\theta}_{x'})$ where the quantifier for x results from the universal closure $\forall^{\alpha}$. We have to show that the derivative ${F'}^{\theta}_{x'}$ is invariant and extend the proof to a proof of $[\exists y\,(x' = \theta \wedge \chi)]{F'}^{\theta}_{x'}$ by weakening (Lemma 6):

$$\cfrac{\cfrac{*}{\vdash \mathrm{QE}(\forall x\,\forall y\,(\chi \to {F'}^{\theta}_{x'}))}\,\text{F1}}{\vdash [\exists y\,(x' = \theta \wedge \chi)]{F'}^{\theta}_{x'}}\,\text{D13'}$$

□

4.6 Differential Induction with Differential Variants

Unlike the differential induction rule G5 for differential invariants, rule G6 uses differential induction to prove that F is a *differential variant*, which is attained differentially as an attractor region, rather than sustained differentially as in G5. The essential difference to G5 thus is the progress condition $F' \geq \varepsilon$ in the premiss, saying that the total differential of F along the DA-constraint is positive and at least some $\varepsilon > 0$. There, $F' \geq \varepsilon$ is a mnemonic notation for replacing all occurrences of inequalities $a \geq b$ in F' by $a \geq b + \varepsilon$ and $a > b$ by $a > b + \varepsilon$ (accordingly for $\leq, >, <$). Intuitively, the premiss expresses that, whereever χ holds but F does not yet hold, the total derivative is pointing towards F, see Fig. 7. Especially $F' \geq \varepsilon$ guarantees a minimum progress rate of ε towards F along the dynamics. To further ensure that the continuous evolution towards F remains within χ, the antedent of the conclusion shows that χ holds *until* F is attained, which can again be proven using G5. In this context, $\sim F$ is a short hand notation for *weak negation*, i.e., the operation that behaves like $\neg$, except that $\sim(a \geq b) \equiv b \geq a$ and $\sim(a > b) \equiv a \leq b$. Unlike negation, weak negation retains the border of F, which is required in G6 as χ needs to continue to hold (including the border of F) until F is reached. Especially, for G6, invariant χ is not required to hold after F has been reached successfully. The operations $F' \geq \varepsilon$ and $\sim F$ are defined accordingly for other inequalities (in G6, we do not permit F to contain equalities, see Counterexample 7 below). Again we demonstrate differential induction and the necessity of its prerequisites in a series of examples.

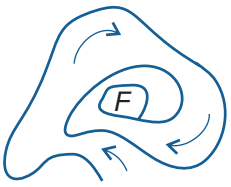


Fig. 7: Differential variants

Example 6. As an example, we turn back to Fig. 5. In the rightmost side deduction, G6 is used to prove that $F \equiv x_1 \geq p_1 \wedge x_2 \geq p_2$ is finally reached. There, the total derivative is $F' \equiv x_1' \geq 0 \wedge x_2' \geq 0$, which yields $d_1 \geq 0 \wedge d_2 \geq 0$ when substituting the equations of $\mathcal{F}(\omega)$, because $x_1' = d_1, x_2' = d_2, p_1' = p_2' = 0$. Thus $(F' \geq \varepsilon)^{\substack{d_1\ d_2\ -\omega d_2\ \omega d_1\ 0\ 0\\ x_1'\ x_2'\ d_1'\ d_2'\ p_1'\ p_2'}}$ is identical to $(F' \geq \varepsilon)^{\substack{d_1\ d_2\ 0\ 0\\ x_1'\ x_2'\ p_1'\ p_2'}}$, which gives $d_1 \geq \varepsilon \wedge d_2 \geq \varepsilon$. Similarly, the proof for formula (3) can be generalised to differential inequalities, again assuming $d_1' = d_2' = p_1' = p_2' = 0$ and $b' = 0$:

$$\forall p\,\exists d\,(\|d\|^2 \leq b^2 \wedge \langle x_1' \geq d_1 \wedge x_2' \geq d_2\rangle(x_1 \geq p_1 \wedge x_2 \geq p_2))\ .$$

Using Lemma 4, the differential inequalities, which express lower bounds on the evolution of x_1 and x_2, can be reduced to differential equations with quantified disturbance $u \in \mathbb{R}^2$:

$$\forall p\, \exists d\, .. \langle \exists u\, (x_1' = d_1 + u_1 \wedge x_2' = d_2 + u_2 \wedge u_1 \geq 0 \wedge u_2 \geq 0)\rangle (x_1 \geq p_1 \wedge x_2 \geq p_2).$$

The proof for this DAL formula is identical to Fig. 5, except that G6 yields $\forall x\, \forall u\, ((x_1 < p_1 \vee x_2 < p_2) \wedge u_1 \geq 0 \wedge u_2 \geq 0 \rightarrow d_1 + u_1 \geq \varepsilon \wedge d_2 + u_2 \geq \varepsilon)$.

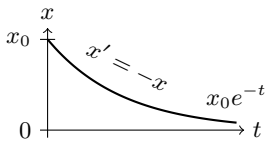
Counterexample 7 (Equational differential variants). Rule G6 is not applicable for equations like $x = y$. Even though $x = y$ can be encoded as $F \equiv x \leq y \wedge x \geq y$, the corresponding $F' \geq \varepsilon \equiv x' + \varepsilon \leq y' \wedge x' \geq y' + \varepsilon$ is equivalent to false for $\varepsilon > 0$. Indeed, assuming $a' = b' = 0$, the validity of a formula like $\langle x' = a \wedge y' = b\rangle x = y$ depends on the relationship of the initial values of x and y and the constants a and b: It is true, iff $(x - y)(a - b) < 0$ or $x = y$ holds initially.

More generally, differential variants cannot (directly) verify conjunctive equations like in $\langle x' = a \wedge y' = b\rangle (x = 0 \wedge y = 0)$ because differential variants guarantee *that* a target region F will be reached, not when precisely. In particular, $x = 0$ and $y = 0$ would not necessarily be reached simultaneously. In fact, for $a, b \neq 0$, the above reachability property is only valid iff $bx = ay \wedge ax < 0$, initially.

Counterexample 8 (Minimal progress requirement). Unlike in discrete domains, strictly monotonic sequences can converge in $\mathbb{R}$. Thus, the premiss $F' \geq \varepsilon$ for an $\varepsilon > 0$ of G6 cannot be weakened to $F' > 0$ as the counterexample in Fig. 8a shows, in which x converges monotonically to 0 along the dynamics shown in Fig. 8b. Moreover, this example demonstrates that, in the presence of convergent dynamics, a property like $x \geq 0$ can be invariant, even though it is not differentially invariant, see Fig. 8c.

$$\dfrac{\dfrac{*\ \text{(unsound)}}{\vdash \forall x\, (x > 0 \rightarrow -x < 0)}}{\vdash \langle x' = -x\rangle x \leq 0}$$

8a: Counterexample



8b: Convergent descent

$$\dfrac{\dfrac{}{x \geq 0 \vdash [x' = -x]x \geq 0}\,\text{G5}}{\dfrac{false}{\vdash \forall x\, (-x \geq 0)}\,\text{F1}}$$

8c: Non-inductive

Fig. 8: Monotonically decreasing convergent counterexample

Counterexample 9 (Lipschitz-continuity requirement). As the counterexample in Fig. 9a shows, Lipschitz-continuity (or at least the existence of a solution of sufficient duration) is, in fact, a necessary prerequisite for G6. For $x = y = 0$ initially, the solution of the differential equations in Fig. 9a is $x(t) = t$ and $y(t) = \tan t$. In explosive examples like the corresponding dynamics in Fig. 9b, where solution y grows unbounded in finite time, the duration of existence of solutions is limited so that the target region $x \geq 6$ is physically unreachable. More precisely, the dynamics is not well-posed beyond the explosive point of unbounded growth at the singularity $\frac{\pi}{2}$ and is non-physical beyond that singularity. Note that the continuous dynamics of Fig. 9 is only locally Lipschitz-continuous and disobeys

Journal of Logic and Computation, 20(1), pp. 309-352, 2010.

doi:10.1093/logcom/exn070

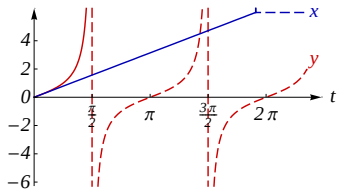
divergence of time (Section 2.2). The condition of Lipschitz-continuity is directly expressible as a formula for G6:

$$\exists L \forall y_1 \ldots \forall y_k \forall x_1 \ldots \forall x_n \, \forall \tilde{y}_1 \ldots \forall \tilde{y}_k \forall \tilde{x}_1 \ldots \forall \tilde{x}_n$$
$$(\theta_1 - \tilde{\theta}_1)^2 + \cdots + (\theta_n - \tilde{\theta}_n)^2 \leq L^2((x_1 - \tilde{x}_1)^2 + \cdots + (x_n - \tilde{x}_n)^2)$$

where $\tilde{\theta}_i$ denotes the result of substituting all x_j in θ_i by the corresponding $\tilde{x}_j$ and the y_j by $\tilde{y}_j$. Observe that, besides Lipschitz-continuity, any other condition can be used that ensures the existence of a solution of sufficient duration for G6.

$$\dfrac{\dfrac{*\ \text{(unsound)}}{\vdash \exists \varepsilon{>}0 \, \forall x \forall y \, (x < 6 \rightarrow 1 \geq \varepsilon)}}{\vdash \langle x' = 1 \wedge y' = 1 + y^2 \rangle x \geq 6}$$

9a: Counterexample



9b: Explosive dynamics

Fig. 9: Unbounded dynamics with limited duration of solutions

4.7 Soundness

In this section we prove that verification with the DAL calculus always produces correct results about DA-programs, i.e., the DAL calculus is sound.

Theorem 1 (Soundness). *The* DAL *calculus is* sound*, i.e., every* DAL *formula that can be derived in the* DAL *calculus is valid (true in all states).*

Proof. The calculus is sound if each rule instance is sound. The rules of the DAL calculus are even *locally sound*, i.e., their conclusion is true at ν if all its premisses are true in ν. Local soundness implies soundness. The local soundness proofs of D1–D4 and the propositional rules are as usual. Similarly, G3 and G4 are local versions of induction schemes, and the proof is as usual [33, 22], likewise for G1–G2. The local soundness of D9–D10 is a generalisation of the proofs for update rules [5] to first-order DJ-constraints. The proofs for D5–D8 are simple. Finally, our previous results [5, 33] can be lifted to show that locally sound rules are closed under addition of Γ, Δ context and of conjunctive DJ-constraints in Def. 16. For soundness, however, conjunctive DJ-constraints are crucial here [5, 33] as these are deterministic.

F3 Rule F3 is locally sound: Let ν be a state in which the premiss is true, i.e.,

$$\nu \models \mathrm{QE}(\exists x \bigwedge_i (\Gamma_i \vdash \Delta_i)) \ .$$

We have to show that the conclusion is true in this state. Using that quantifier elimination yields an equivalence, we see that ν also satisfies $\exists x \bigwedge_i (\Gamma_i \vdash \Delta_i)$

prior to the quantifier elimination. Hence, for some state ν_x that agrees with ν except for the value of x we obtain:

$$\nu_x \models \bigwedge_i (\Gamma_i \vdash \Delta_i) \ .$$

As side deduction ($\star$) in Fig. 4 is inductively shown to be locally sound, we can conclude that $\nu_x \models (\Gamma \vdash \Delta, \phi)$. Therefore, $\nu \models \exists x\, (\Gamma \vdash \Delta, \phi)$. Now the conjecture can be obtained using standard reasoning with quantifiers and the absence of x in Γ, Δ by rewriting the conclusion with local equivalences:

$$\exists x\, (\Gamma \vdash \Delta, \phi) \ \equiv \exists x\, (\neg\Gamma \vee \Delta \vee \phi) \ \equiv \neg\Gamma \vee \Delta \vee \exists x\, \phi \ \equiv \Gamma \vdash \Delta, \exists x\, \phi \quad (4)$$

The soundness proof for F1 is similar since (4) holds for any quantifier. The proofs of F4 and F2 can be derived using duality of quantifiers.

D12 By Lemma 3, there is an equivalent disjunctive normal form $\mathcal{D}_1 \vee \cdots \vee \mathcal{D}_n$ of $\mathcal{D}$. Thus, it only remains to show that $\rho(\mathcal{D}) \subseteq \rho((\mathcal{D}_1 \cup \ldots \cup \mathcal{D}_n)^*)$ as the converse inclusion is obvious. Let φ be a state flow for a transition $(\nu, \omega) \in \rho(\mathcal{D})$. We assume that φ is non-Zeno according to Def. 12. Thus, there is a finite number, m, of switches between the $\mathcal{D}_i$, say $\mathcal{D}_{i_1}, \mathcal{D}_{i_2}, \ldots, \mathcal{D}_{i_m}$. Then, the transition (ν, ω) belonging to φ can be simulated piecewise by m repetitions of $\mathcal{D}_1 \cup \ldots \cup \mathcal{D}_n$, where each piece selects the respective part $\mathcal{D}_{i_j}$. The proof for D11 is accordingly.

D13 Local soundness of rules D13 and D14 are immediate consequences of Lemma 3 and the respective semantics of modalities.

D15 Rule D15 can be proven locally sound using that the left premiss implies that every flow φ that satisfies $\mathcal{D}$ also satisfies χ *all along* the flow. Thus, $\varphi \models \mathcal{D}$ implies $\varphi \models \mathcal{D} \wedge \chi$ so that the right premiss entails the conclusion.

G5 Let ν satisfy the premiss and the antecedent of the conclusion as, otherwise, there is nothing to show. By Lemma 3, we can assume F to be in disjunctive normal form and consider any disjunct G of F that is true at ν. In order to show that F is sustained 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 accordingly). Finally, using vectorial notation, we write $x' = \theta$ for the differential equation system and $\exists y$ for the chain of quantifiers. Now let $\varphi : [0, r] \to \mathrm{State}(\Sigma)$ be any state flow with $\varphi \models \exists y\, (x' = \theta \wedge \chi)$ beginning in $\varphi(0) = \nu$. In particular, $\varphi \models \exists y\, \chi$, which, by antecedent, implies $\nu \models F$, i.e., $c \geq 0$ holds at ν. We assume duration $r > 0$, because the other case is immediate ($\nu \models c \geq 0$ already holds). We show that $c \geq 0$ holds all along the flow φ, i.e., $\varphi \models c \geq 0$.
Suppose there was a $\zeta \in [0, r]$ where $\varphi(\zeta) \models c < 0$, which will lead to a contradiction. Then the function $h : [0, r] \to \mathbb{R}$ defined as $h(t) = val(\varphi(t), c)$ satisfies $h(0) \geq 0 > h(\zeta)$, because $\nu \models c \geq 0$ by antecedent. Clearly, φ is of the order of $D(c)$, because: φ is of order 1 for all variables in vector x, and trivially of order ∞ for variables that do not change during the DA-constraint. Further, by α-renaming, $D(c)$ cannot contain the quantified variables y, hence, φ is not required to be of any order in y. The value of c is defined all along φ, because we have assumed χ to guard against zeros of denominators. Thus, by Lemma 1, h is continuous on $[0, r]$ and differentiable at every $\xi \in (0, r)$. The mean value theorem implies that 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 that $\frac{\mathrm{d}h(t)}{\mathrm{d}t}(\xi) < 0$. Now Lemma 1 implies that $\frac{\mathrm{d}h(t)}{\mathrm{d}t}(\xi) = val(\bar{\varphi}(\xi), D(c)) < 0$. The latter equals[1] $val(\bar{\varphi}(\xi)_y^u, D(c)_{x'}^\theta)$ by Lemma 2, because $\varphi \models \exists y\,(x' = \theta \wedge \chi)$ so that $\bar{\varphi}(\xi)_y^u \models x' = \theta \wedge \chi$ for some $u \in \mathbb{R}$ and because y' does not occur and $y \notin c$. This, however, is a contradiction, because the premiss implies that $\varphi \models \forall y\,(\chi \to D(c)_{x'}^\theta \geq 0)$ as $\forall^\alpha$ comprises all variables that change during the flow φ along $x' = \theta$, i.e., the vector x. In particular, as $\bar{\varphi}(\xi)_y^u \models \chi$ holds, we have $\bar{\varphi}(\xi)_y^u \models D(c)_{x'}^\theta \geq 0$.

G6 First, we consider the quantifier free case, again using vectorial notation. Let ν be any state satisfying the premiss and the antecedent of the conclusion. Since ν satisfies the premiss and, after α-renaming, ε is a fresh variable, we can assume ν itself to satisfy $\nu \models \forall^\alpha(\neg F \wedge \chi \to (F' \geq \varepsilon)_{x'}^\theta)$. For G6, we required $x' = \theta$ to be Lipschitz-continuous so that the global Picard-Lindelöf theorem [48, Theorem 10.VII] ensures the existence of a global solution of arbitrary duration $r \geq 0$, which is all we need here. Let φ be a state flow corresponding to a solution of the differential equation $x' = \theta$ starting in ν of some duration $r \geq 0$. If there is a point in time ζ at which $\varphi(\zeta) \models F$, then by antecedent, until (and including, because $\sim F$ contains the closure of $\neg F$) the first such point, χ holds true during φ. Hence, the restriction of φ to $[0, \zeta]$ is a state flow witnessing $\nu \models \langle x' = \theta \wedge \chi\rangle F$. If, otherwise, there is no such point, then we show that extending φ by choosing a larger r will inevitably make F true. We thus have $\varphi \models \neg F \wedge \chi$ and, by premiss, $\varphi \models {F'}_{x'}^\theta \geq \varepsilon$, because $\forall^\alpha$ comprises the variables x that change during φ. By Def. 13, ${F'}_{x'}^\theta \geq \varepsilon$ is a conjunction. Consider one of its conjuncts, say ${c'}_{x'}^\theta \geq \varepsilon$ belonging to a literal $c \geq 0$ of F (the other cases are accordingly). Again, φ is of the order of $D(c)$ and the value of c is defined along φ, because $\varphi \models \chi$ and χ is assumed to guard against zeros. Hence, by mean-value theorem, Lemma 1, and Lemma 2, we conclude for each $\zeta \in [0, r]$ that

$$val(\varphi(\zeta), c) - val(\varphi(0), c) = val(\bar{\varphi}(\xi), {c'}_{x'}^\theta)(\zeta - 0) \geq \zeta val(\varphi(0), \varepsilon)$$

for some $\xi \in (0, \zeta)$. Now as $val(\varphi(0), \varepsilon) > 0$ we have for all $\zeta > -\frac{val(\varphi(0),c)}{val(\varphi(0),\varepsilon)}$ that $\varphi(\zeta) \models c \geq 0$ and $\varphi(r) \models c \geq 0$, even $\varphi(r) \models c > 0$. By extending r sufficiently large, we have that all literals $c \geq 0$ of one conjunct of F are true, which concludes the proof, because, until F finally holds, $\varphi \models \chi$ is implied by the antecedent as shown earlier.

In the presence of quantifiers ($\exists y$ with vectorial notation), rule G6 implies a slightly stronger statement, because y is quantified universally in the premiss (and antecedent): F can be reached for *all* choices of y that respect χ (rather than just for one). By antecedent, there is a $u \in \mathbb{R}$ such that $\nu_y^u \models \chi$. Hence, ν_y^u satisfies the assumptions of the above quantifier-free case. Thus, $\nu_y^u \models \langle x' = \theta \wedge \chi\rangle F$, which entails that $\nu \models \langle \exists y\,(x' = \theta \wedge \chi)\rangle F$ using u constantly as the value for the quantified variable y during the evolution. □

As a consequence of a corresponding result in [31], the DAL calculus is not effectively axiomatisable (yet even pure reachability is already undecidable for hybrid systems [23]). For relative completeness results for differential dynamic logics, we refer to previous work [33], where the proof carries over to DAL.

[1] For $u \in \mathbb{R}$ let $\bar{\varphi}(\xi)_y^u$ denote the (augmented) state that agrees with $\bar{\varphi}(\xi)$ except that the value of y is u.

4.8 Deductive Strength of Differential Induction

We analyse the deductive power of differential induction with respect to classes of formulas that are allowed as differential invariants. For purely equational differential invariants, the deductive power is not affected by allowing or disallowing propositional operators in differential invariants:

Proposition 1. *The deductive power of differential induction with atomic equations is identical to the deductive power of differential induction with propositional combinations of polynomial equations: Formulas are provable with propositional combinations of equations as differential invariants iff they are provable with only atomic equations as differential invariants.*

Proof. We show that every differential invariant that is a propositional combination ϕ of polynomial equations is expressible as a single atomic polynomial equation (the converse inclusion is obvious). We assume ϕ to be in negation normal form and reduce ϕ inductively using the following transformations:

- If ϕ is of the form $p_1 = p_2 \vee q_1 = q_2$, then ϕ is equivalent to the single equation $(p_1 - p_2)(q_1 - q_2) = 0$. Further $\phi' \equiv p_1' = p_2' \wedge q_1' = q_2'$ directly implies $((p_1 - p_2)(q_1 - q_2))' = 0 \equiv (p_1' - p_2')(q_1 - q_2) + (p_1 - p_2)(q_1' - q_2') = 0$.
- If ϕ is of the form $p_1 = p_2 \wedge q_1 = q_2$, then ϕ is equivalent to the single equation $(p_1 - p_2)^2 + (q_1 - q_2)^2 = 0$. Further $\phi' \equiv p_1' = p_2' \wedge q_1' = q_2'$ implies $2(p_1 - p_2)(p_1' - p_2') + 2(q_1 - q_2)(q_1' - q_2') = 0$.
- If ϕ is of the form $\neg(p_1 = p_2)$, then ϕ does not qualify as a differential invariant, because it contains a negative equality, which are disallowed for G5 according to Fig. 3. □

Observe, however, that the required polynomial degree of atomic equations is larger than for propositional combinations, which can have computational disadvantages for quantifier elimination.

For general differential invariants, where inequalities are allowed, the situation is different: We show that, in general, the deductive power of differential induction depends on which class of formulas is allowed as differential invariants! Some DAL formulas cannot by proven by a differential induction step with only atomic formula but no propositional operators as differential invariant, while they are provable immediately using unrestricted differential invariants.

Theorem 2. *The deductive power of differential induction with arbitrary formulas exceeds the deductive power of differential induction with atomic formulas: All* DAL *formulas that are provable using atomic differential invariants are provable using general differential invariants, but not vice versa!*

Proof. The inclusion is obvious. Conversely, we have to show that there are DAL formulas that are provable with general differential invariants but not with atomic differential invariants. Consider the following example, which is provable using rule G5', i.e., the variant of G5 for open sets (Appendix A), with the non-atomic formula $x > 0 \wedge y > 0$ as differential invariant:

$$\text{G5'}\,\frac{\text{F1}\,\dfrac{*}{\vdash \forall x\,\forall y\,(x > 0 \wedge y > 0 \rightarrow xy > 0 \wedge xy > 0)}}{x > 0 \wedge y > 0 \vdash [x' = xy \wedge y' = xy](x > 0 \wedge y > 0)}$$

First, we show that this formula is not provable by a differential induction step with only atomic formulas as differential invariants. Suppose there was a single polynomial $p(x,y)$ in variables x,y such that $p(x,y)>0$ is a differential invariant proving the above formula, which will lead to a contradiction. The conditions for differential invariants (G5 or G5') imply that the following formulas have to be valid:

1. $x>0\land y>0\rightarrow p(x,y)>0$, as differential invariants have to hold in the prestate according to the antecedent of G5 (or G5').
2. $p(x,y)>0\rightarrow x>0\land y>0$, as the differential invariant has to imply the postcondition (when using G1 to show that the differential invariant implies the postcondition).

In particular, $x>0\land y>0\leftrightarrow p(x,y)>0$ is valid. Thus, p enjoys the property:

$$p(x,y)\geq 0 \text{ for } x\geq 0, y\geq 0, \text{ and, otherwise, } p(x,y)\leq 0\ . \tag{5}$$

Assume p has minimal total degree with property (5). Now, $p(x,0)$ is a univariate polynomial in x with zeros at all $x>0$, thus $p(x,0)=0$ is the zero polynomial, hence y divides $p(x,y)$. Accordingly, $p(0,y)=0$ for all y, hence x divides $p(x,y)$. Thus, xy divides p. But by comparing the signs, we see that polynomial $\frac{-p(-x,-y)}{xy}$ also satisfies property (5) with a smaller total degree than p, which is a contradiction.

Similarly, there is no polynomial p such that $x>0\land y>0\leftrightarrow p(x,y)=0$, because only the zero polynomial is zero on the full quadrant $(0,\infty)^2$. Finally, $x>0\land y>0\leftrightarrow p(x,y)\geq 0$ is impossible for continuity reasons that imply $p(0,0)=0$, which is a contradiction. More generally, the same argument holds for any other sign condition that is supposed to characterise one quadrant of $\mathbb{R}^2$ uniquely.

Observe that, so far, the argument does not depend on the actual dynamics and is, thus, still valid in the presence of arbitrary differential weakening (D13).

Next, to see that the above example cannot even be proven indirectly after differential strengthening (D15), we use that, inductively, the strengthening χ itself needs to be a differential invariant: Ultimately, the left sub-goal of D15 can only be shown using differential induction. The above example, however, is built such that, as $x'=xy$ is the differential equation, $xy>0$ is required for $x>0$ to be a differential invariant (which thus also requires $y>0$). Vice versa, due to $y'=xy$, formula $xy>0$ is a prerequisite for the differential invariance of $y>0$ (which thus also needs $x>0$). Yet, for differential invariance of $xy>0$, we have to prove $xy>0\rightarrow(y+x)xy>0$ for G5', because $(xy)'^{xy\ xy}_{x'\ y'}$ gives $(x'y+yx')^{xy\ xy}_{x'\ y'}$, i.e., $xyy+yxy$. But $xy>0\rightarrow(y+x)xy>0$ is, again, equivalent to $x\geq 0\lor y\geq 0$, and thus to $\neg(-x>0\land -y>0)$, which cannot be proven by atomic differential induction (or differential weakening) according to the first part of this proof. Thus, the required atomic differential invariants have circular dependencies for differential strengthenings by $x>0$, $y>0$, and $xy>0$, respectively, which cannot be resolved in any proof tree without simultaneous differential induction using non-atomic differential invariants, because differential strengthenings have to be ordered totally along each proof branch. □

As a special case, this result implies that differential induction in DAL is deductively stronger than approaches using barrier certificates [39, 40], criticality functions [10], or polynomial invariant equations [43, 41]. On top of that, the

DAL calculus adds differential strengthening and weakening techniques, which add further deductive power. The roundabout maneuver that we verify in the next section is a practical example where differential induction with mixed non-atomic formulas and successive differential strengthening turns out to be decisive.

5 Verifying Tangential Roundabout Maneuvers in Air Traffic Control

In this section we verify that the tangential roundabout maneuver for collision avoidance in air traffic control that we presented in Section 3 is collision-free, i.e., directs aircraft on flight paths with global minimal distance $p > 0$, and determine a corresponding parameter constraint on the *entry* procedure. Using differential induction and differential strengthening, the flight maneuver can be verified despite the complicated hybrid flight dynamics of aircraft.

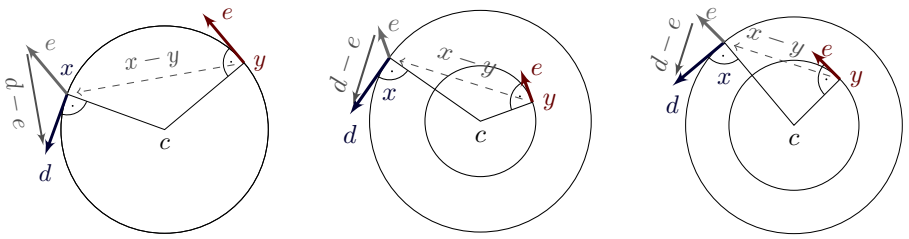
Characterisation of Safe Roundabout Dynamics. Property ϕ in Fig. 2 defines *safe states* as those with separation $\|x-y\| \geq p$. This does *not*, however, characterise the states with *safe dynamics*: Several states that satisfy ϕ will not remain safe when following curved roundabout flight maneuvers, see Fig. 1c for a counterexample violating ϕ after some time. In particular, the angular velocity ω and initial speed vectors d and e must fit to the relative positioning of the aircraft x and y for the aircraft dynamics to remain safe. In order to find out the required parametric constraints for safety of the roundabout maneuver, we analyse the DAL formula ψ in the DAL calculus and identify a corresponding parameter constraint $\mathcal{T}$. For notational convenience, we inline side deductions and slightly simplify universal closure notation $\forall^\alpha$ by taking free variables as universally quantified, because the following DAL proof needs no existential variables.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\phi \vdash \forall x,y,d,e\,(\phi \to \phi)}\,*}{\phi \vdash [\mathit{free}]\phi}\,\text{F1}, \text{D13}' \qquad \dfrac{}{\phi \vdash [\mathit{entry}](\phi \wedge \mathcal{T})}\,\ldots \qquad \dfrac{}{\phi \wedge \mathcal{T} \vdash [\mathcal{F}(\omega) \wedge \mathcal{G}(\omega)]\phi}\,\ldots}{\phi \vdash [\mathit{entry}; \mathcal{F}(\omega) \wedge \mathcal{G}(\omega)]\phi}\,\text{G1}}{\phi \vdash [\mathit{free}][\mathit{entry}; \mathcal{F}(\omega) \wedge \mathcal{G}(\omega)]\phi}\,\text{G1}}{\phi \vdash [\mathit{trm}]\phi}\,\text{D2}}{\vdash \forall^\alpha(\phi \to [\mathit{trm}]\phi)}\,\text{F1,P7}}{\phi \vdash [\mathit{trm}^*]\phi}\,\text{G3}}{\vdash \phi \to [\mathit{trm}^*]\phi}\,\text{P7}$$

The left branch closes, because postcondition ϕ is the invariant region in free flight such that its DA-constraint can be weakened by Lemma 6. In the other branches, $\mathcal{T}$ is the parameter constraint that *entry* needs to establish in addition to ϕ (middle branch) for the roundabout dynamics to be safe (right branch). Hence condition $\mathcal{T}$ mediates among the middle and right branch. Using successive quantifier elimination, we derive the following constraint $\mathcal{T}$ as a prerequisite for ϕ to be differentially inductive. It is the decisive constraint that characterises configurations with safely controllable dynamics in curved roundabout maneuvers (using vectorial notation and orthogonal complements $d^\perp$ from Section 2):

$$\begin{aligned} \mathcal{T} &\equiv d - e = \omega(x-y)^\perp \quad \big(\text{or, equivalently } (d-e)^\perp = -\omega(x-y)\big) \qquad (6)\\ &\equiv d_1 - e_1 = -\omega(x_2 - y_2) \wedge d_2 - e_2 = \omega(x_1 - y_1)\ . \end{aligned}$$

This formula expresses that the relative speed vector $d-e$ is orthogonal to the relative position $x-y$ and compatible with the angular velocity ω and tangential orientation of d and e. Figure 10a illustrates the symmetric case with identical linear speed $\|d\| = \|e\|$, Fig. 10b–10c show asymmetric cases with distinct linear speeds $\|d\| \neq \|e\|$, which is possible as well. Condition $\mathcal{T}$ gives the decisive handle



10a: Symmetric case 10b: Asymmetric case 10c: Close flight

Fig. 10: Tangential construction for characteristics $\mathcal{T}$ of roundabout dynamics

for an inductive characterisation of safe tangential roundabout configurations: For the right branch of the above proof, we need to show that the tangential configuration $\mathcal{T}$ is sufficient for ϕ to be sustained during curved evasive actions. In the following, we prove that the relative speed vector configuration $\mathcal{T}$ is itself differentially inductive (left branch) and use differential strengthening with D15 to augment the dynamics with $\mathcal{T}$ as a derived invariant for proving that the actual safety property ϕ is sustained (right branch), again by differential induction:

$$\dfrac{\dfrac{\dfrac{*}{\vdash \forall^\alpha(\mathcal{T}'_{\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)})}\,\text{F1}}{\phi, \mathcal{T} \vdash [\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)]\mathcal{T}}\,\text{G5} \qquad \dfrac{\dfrac{*}{\vdash \forall^\alpha(\mathcal{T} \to \phi'_{\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)})}\,\text{F1}}{\phi \vdash [\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)\wedge\mathcal{T}]\phi}\,\text{G5}}{\dfrac{\phi, \mathcal{T} \vdash [\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)]\phi}{\phi\wedge\mathcal{T} \vdash [\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)]\phi}\,\text{P6}}\,\text{D15}$$

Observe that differential strengthening by D15 is crucial for the proof, because neither ϕ nor $\mathcal{T}\wedge\phi$ are differentially inductive for $\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)$! Instead, the tangential configuration $\mathcal{T}$ itself is differentially inductive relative to $\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)$ (left branch) and strong enough to make ϕ differentially inductive relative to the augmented DA-constraint $\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)\wedge\mathcal{T}$ (right branch). For readability, we use a slightly weaker rule for differential induction, with ϕ rather than $[\mathcal{T}]\phi$ in the antecedent of the conclusion. This variant can be derived easily using a cut and will again be called G5. The differential induction G5 on the left and right branch close using quantifier elimination in F1 or the following algebraic equational reasoning, respectively ($\mathcal{T}'_{\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)}$ is a short notation for substituting the differential equations from $\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)$ into $D(\mathcal{T})$, see Lemma 2):

$$\begin{aligned}
\mathcal{T}'_{\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)} &\equiv (d_1'-e_1' = -\omega(x_2'-y_2') \wedge d_2'-e_2' = \omega(x_1'-y_1'))_{\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)} \\
&\equiv -\omega d_2+\omega e_2 = -\omega(d_2-e_2) \wedge \omega d_1-\omega e_1 = \omega(d_1-e_1) \quad \equiv \mathit{true} \\
\phi'_{\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)} &\equiv (2(x_1-y_1)(x_1'-y_1')+2(x_2-y_2)(x_2'-y_2') \geq 0)_{\mathcal{F}(\omega)\wedge\mathcal{G}(\omega)} \\
&\equiv 2(x_1-y_1)(d_1-e_1)+2(x_2-y_2)(d_2-e_2) \geq 0 \\
(\text{using } \mathcal{T}) &\equiv 2(x_1-y_1)(-\omega(x_2-y_2))+2(x_2-y_2)\omega(x_1-y_1) = 0 \geq 0 \ \equiv \mathit{true}.
\end{aligned}$$

Altogether, we have shown that *every* tangential roundabout evasion maneuver respecting $\mathcal{T}$ is safe. Further, the middle branch of the above proof reveals the parameter constraint imposed on *entry* for safe roundabouts, which concludes the proof of the following result.

Theorem 3 (Safety of tangential roundabout maneuver). *For every choice of the tangential entry procedure that satisfies $\phi \to [entry](\phi \wedge \mathcal{T})$, the tangential roundabout flight maneuver in Fig. 2 safely avoids collisions, i.e., it directs aircraft on flight paths with minimal horizontal aircraft separation at least $p > 0$.*

This result can be proven in our theorem prover [36] in 3s including user interactions for D13 and D15. Its proof does not need G1, which we only used here to shorten the proof presentation. Theorem 3 expresses unbounded-time safety for fully parametric tangential roundabouts with arbitrary choices for the free parameters. The proof of Theorem 3 generalises to roundabouts entered by more than two participants when ϕ and $\mathcal{T}$ are augmented accordingly. For instance, using an automatic proof procedure based on the DAL calculus, our theorem prover can prove mutual collision avoidance for 5 aircraft fully automatically [35]. Likewise, G5 and D15 can be used to prove that external separation to all other sufficiently far points is maintained during the roundabout maneuver, in particular, the maneuver only needs bounded space:

Proposition 2 (External separation of roundabout maneuvers). *Separation of aircraft x to all external points $u \in \mathbb{R}^2$ of distance beyond the roundabout diameter $2r$ is maintained:*

$$r \geq 0 \wedge (r\omega)^2 = \|d\|^2 \to \forall u\,(\|x-u\|^2 > (2r+p)^2 \to [\mathcal{F}(\omega)](\|x-u\|^2 > p^2)) \ .$$

Tangential Entry Procedure. As a simple choice for the tangential initiation procedure *entry* satisfying property $\mathcal{T}$, consider the following operation which chooses an arbitrary angular velocity ω, an arbitrary centre $c \in \mathbb{R}^2$ for the roundabout maneuver, and adjusts d and e tangentially:

$$entry \ \equiv \ \exists u\, \omega := u;\ \exists c\,(d := \omega(x-c)^\perp \wedge e := \omega(y-c)^\perp) \ . \tag{7}$$

This formula expresses that the speed vectors d and e of both aircraft at x and y, respectively, are tangentially and of the same angular velocity ω relative to the intended centre c of the roundabout, with the same orientation (Fig. 10). For this choice, the assumption of Theorem 3 can be proven after D10 substitutes the corresponding terms for d and e in $\mathcal{T}$, using F1 (or linearity of $d^\perp$):

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\phi \vdash \phi}{\scriptstyle *}\,\text{P9} \qquad \dfrac{}{\phi \vdash \omega(x-c)^\perp - \omega(y-c)^\perp = \omega(x-y)^\perp}{\scriptstyle *}}{\phi \vdash \phi \wedge \omega(x-c)^\perp - \omega(y-c)^\perp = \omega(x-y)^\perp}\,\text{P5}}{\phi \vdash [d := \omega(x-c)^\perp \wedge e := \omega(y-c)^\perp](\phi \wedge \mathcal{T})}\,\text{D10}}{\phi \vdash \forall\omega\,\forall c\,[d := \omega(x-c)^\perp \wedge e := \omega(y-c)^\perp](\phi \wedge \mathcal{T})}\,\text{F1,F1}}{\phi \vdash [entry](\phi \wedge \mathcal{T})}\,\text{D2,D6,D6}$$

It can also be shown that $\exists c\,(d = \omega(x-c)^\perp \wedge e = \omega(y-c)^\perp)$ is equivalent to $\mathcal{T}$ for nonzero ω. With choice (7), the tangential roundabout maneuver in Fig. 2 is safe and has been significantly simplified and generalised in comparison to [34].

Discussion. Our tangential roundabout maneuver leaves open how and when precisely the collision avoidance maneuver is initiated or when to leave it. For instance, (7) does not restrict c and ω but accepts any choice including choices optimising secondary objectives like fuel consumption. Furthermore, as specified in Fig. 2 and proven in this section, the roundabout maneuver can be left safely with arbitrary free flight by repeating the loop at any time: The roundabout maneuver will be initiated again during free flight when necessary. As a special case, this open policy includes free flight enabling the aircraft to leave the roundabout in their original direction. While the simple choice (7) is possibly discontinuous in d and e, it is comparably easy to see that there are fully curved entry and exit procedures that remain safe when the entry procedure is initiated with sufficient distance by using the separation limit of Proposition 2. Developing a corresponding entry procedure is, however, beyond the scope of this paper. Our proof shows that the tangential roundabout maneuver is safe for every such entry procedure. In particular, the control parameters c and ω of (7) can also be chosen such that the resulting speed vectors d and e are in a bounded range meeting external speed requirements of the aircraft:

$$\forall v\,(\phi \to \langle \mathit{entry} \rangle (\phi \wedge \mathcal{T} \wedge \|d\|^2 = \|e\|^2 = v^2)) \ .$$

6 Conclusions and Future Work

We have introduced a first-order dynamic logic for differential-algebraic programs with interacting first-order discrete jump constraints and first-order differential-algebraic constraints. For this differential-algebraic logic, DAL, we have presented a calculus for verifying hybrid systems given as differential-algebraic programs.

In differential-algebraic programs, both internal choices and disturbances during continuous evolutions and nondeterminism in discrete operations can be described uniformly by quantifiers. Most importantly, we have introduced first-order differential induction with differential invariants and differential variants for proving correctness statements with first-order differential-algebraic constraints purely algebraically using the differential constraints themselves instead of their solutions. In combination with successive differential strengthening for refining the system dynamics by auxiliary differential invariants, we obtain a powerful verification calculus for systems with challenging dynamics. We compare the deductive strength for classes of differential invariants and show that the deductive power of general differential induction exceeds the deductive power of atomic differential invariants.

We have demonstrated that our calculus can be used successfully for verifying fully parametric roundabout maneuvers in air traffic control. To the best of our knowledge, this is the first formal proof for unbounded safety of hybrid aircraft dynamics in curved collision avoidance maneuvers for air traffic control. Moreover, we argue that our fully formal proof about aircraft gives more confidence in flight maneuvers than informal approaches that do not consider the actual hybrid flight dynamics [25, 14, 18] or results that only prevent orthogonal collisions in discretisations of the system [11, 29]. Our logic DAL is also more convenient, because hybrid systems like the tangential roundabout maneuver can be specified and verified uniformly within a single logic. Despite challenging flight dynamics, the DAL formulas about aircraft and roundabout maneuvers

that we presented in this paper can be proven in our theorem prover [36] within a few seconds.

While this work answers the open issues (1), (3) and (4) raised in the work of Piazza et al. [30], we are interested in extending differential-algebraic methods to address further questions about hybrid systems. In a follow-up paper [35], we investigate algorithms for constructing differential invariants automatically on the basis of our DAL calculus presented here. Interesting future work for the aircraft case study is to find a fully curved maneuver that achieves collision avoidance by joint horizontal and vertical evasive actions.

Funding. This research was partially supported by the German Research Council (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR 14 AVACS).

Acknowledgements. I want to thank the anonymous referees for their most detailed and helpful comments. I also thank Thomas Bliem for suggestions and Ernst-Rüdiger Olderog and Johannes Faber for their proofreading remarks.

A Soundness of Differential Invariance Restrictions

While Example 4 shows that differential invariant F cannot generally be assumed to hold in the premiss of G5 without loosing soundness, we present two corresponding refinements of G5 that are indeed sound.

Proposition 3. *Using the notation of G5–G6, the following variations of differential induction G5 are sound (in the first rule, F describes an open set):*

$$\text{(G5')}\quad \frac{\vdash \forall^\alpha \forall y_1 \ldots \forall y_k \, (F \wedge \chi \to F'^{\theta_1 \ldots \theta_n}_{x'_1 \ldots x'_n})}{[\exists y_1 \ldots \exists y_k \, \chi]F \vdash [\exists y_1 \ldots \exists y_k \, (x'_1 = \theta_1 \wedge \cdots \wedge x'_n = \theta_n \wedge \chi)]F}$$

$$\text{(G5")}\quad \frac{\vdash \forall^\alpha \forall y_1 \ldots \forall y_k \, (F \wedge \chi \to (F' > 0)^{\theta_1 \ldots \theta_n}_{x'_1 \ldots x'_n})}{[\exists y_1 \ldots \exists y_k \, \chi]F \vdash [\exists y_1 \ldots \exists y_k \, (x'_1 = \theta_1 \wedge \cdots \wedge x'_n = \theta_n \wedge \chi)]F}$$

Proof. The proof that the first rule is sound is similar to the proof for G5 in Theorem 1, except that assuming $\varphi(\zeta) \models \neg F$ only yields $h(0) \geq 0 \geq h(\zeta)$, which does not lead to a contradiction. However, by using that F is open, the distance to the border of F is positive in the initial state $\varphi(0)$, which yields the inequality $h(0) > 0 \geq h(\zeta)$, and the contradiction arises accordingly.

The soundness of the second rule needs more adaptation. Repeating the argument for G5, we can assume F to be of the form $c \geq 0$. Suppose there was a $\iota \in [0, r]$ where $\varphi(\iota) \models c < 0$, which will lead to a contradiction. Let $\zeta \in [0, r]$ be the infimum of these ι, hence, $\varphi(\zeta) \models c = 0$ by continuity. Then the function $h : [0, r] \to \mathbb{R}$ defined as $h(t) = val(\varphi(t), c)$ satisfies $h(0) \geq 0 \geq h(\zeta)$, because $\nu \models c \geq 0$ by antecedent. By repeating the argument with Lemma 1 like in the proof for G5, h is continuous on $[0, r]$ and differentiable at every $\xi \in (0, r)$ with derivative $\frac{\mathsf{d}h(t)}{\mathsf{d}t}(\xi) = val(\bar{\varphi}(\xi), D(c))$, which in turn equals $val(\bar{\varphi}(\xi), D(c)^\theta_{x'})$, because $\varphi \models x' = \theta$. Now, the mean value theorem implies that there is a $\xi \in (0, \zeta)$ such that $\frac{\mathsf{d}h(t)}{\mathsf{d}t}(\xi) \cdot (\zeta - 0) = h(\zeta) - h(0) \leq 0$. In particular, as $\zeta \geq 0$, we can conclude that $\frac{\mathsf{d}h(t)}{\mathsf{d}t}(\xi) = val(\bar{\varphi}(\xi), D(c)^\theta_{x'}) \leq 0$. This, however, contradicts that the premiss implies $\bar{\varphi}(\xi) \models D(c)^\theta_{x'} > 0$, as the flow satisfies $\varphi \models \chi$ and $\varphi(\xi) \models c \geq 0$, because $\zeta > \xi$ is the infimum of the counterexamples ι with $\varphi(\iota) \models c < 0$. □

References

1. Alur, R., Dill, D.L.: A theory of timed automata. Theor. Comput. Sci. **126**(2) (1994) 183–235
2. Alur, R., Pappas, G.J., eds.: Hybrid Systems: Computation and Control, 7th International Workshop, HSCC 2004, Philadelphia, PA, USA, March 25-27, 2004, Proceedings. In Alur, R., Pappas, G.J., eds.: HSCC. Volume 2993 of LNCS., Springer (2004)
3. Anai, H., Weispfenning, V.: Reach set computations using real quantifier elimination. In Benedetto, M.D.D., Sangiovanni-Vincentelli, A.L., eds.: HSCC. Volume 2034 of LNCS., Springer (2001) 63–76
4. Asarin, E., Dang, T., Girard, A.: Reachability analysis of nonlinear systems using conservative approximation. In Maler, O., Pnueli, A., eds.: HSCC. Volume 2623 of LNCS., Springer (2003) 20–35
5. Beckert, B., Platzer, A.: Dynamic logic with non-rigid functions: A basis for object-oriented program verification. In Furbach, U., Shankar, N., eds.: IJCAR. Volume 4130 of LNCS., Springer (2006) 266–280
6. Branicky, M.S.: General hybrid dynamical systems: Modeling, analysis, and control. In Alur, R., Henzinger, T.A., Sontag, E.D., eds.: Hybrid Systems. Volume 1066 of LNCS., Springer (1995) 186–200
7. Clarke, E.M., Fehnker, A., Han, Z., Krogh, B.H., Ouaknine, J., Stursberg, O., Theobald, M.: Abstraction and counterexample-guided refinement in model checking of hybrid systems. Int. J. Found. Comput. Sci. **14**(4) (2003) 583–604
8. Collins, G.E., Hong, H.: Partial cylindrical algebraic decomposition for quantifier elimination. J. Symb. Comput. **12**(3) (1991) 299–328
9. Collins, P., Lygeros, J.: Computability of finite-time reachable sets for hybrid systems. In: CDC-ECC'05, IEEE (Dec 2005) 4688– 4693
10. Damm, W., Mikschl, A., Oehlerking, J., Olderog, E.R., Pang, J., Platzer, A., Segelken, M., Wirtz, B.: Automating verification of cooperation, control, and design in traffic applications. In Jones, C.B., Liu, Z., Woodcock, J., eds.: Formal Methods and Hybrid Real-Time Systems. Volume 4700 of LNCS., Springer (2007) 115–169
11. Damm, W., Pinto, G., Ratschan, S.: Guaranteed termination in the verification of LTL properties of non-linear robust discrete time hybrid systems. In Peled, D., Tsay, Y.K., eds.: ATVA. Volume 3707 of LNCS., Springer (2005) 99–113
12. Davoren, J.M., Nerode, A.: Logics for hybrid systems. IEEE **88**(7) (July 2000) 985–1010
13. Dowek, G., Hardin, T., Kirchner, C.: Theorem proving modulo. J. Autom. Reasoning **31**(1) (2003) 33–72
14. Dowek, G., Muñoz, C., Carreño, V.A.: Provably safe coordinated strategy for distributed conflict resolution. In: Proceedings of the AIAA Guidance Navigation, and Control Conference and Exhibit 2005, AIAA-2005-6047. (2005)
15. Fitting, M.: First-Order Logic and Automated Theorem Proving. 2nd edn. Springer, New York (1996)
16. Fitting, M., Mendelsohn, R.L.: First-Order Modal Logic. Kluwer, Norwell, MA, USA (1999)
17. Fränzle, M.: Analysis of hybrid systems: An ounce of realism can save an infinity of states. In Flum, J., Rodríguez-Artalejo, M., eds.: CSL. Volume 1683 of LNCS., Springer (1999) 126–140
18. Galdino, A.L., Muñoz, C., Ayala-Rincón, M.: Formal verification of an optimal air traffic conflict resolution and recovery algorithm. In Leivant, D., de Queiroz, R., eds.: WoLLIC. Volume 4576 of LNCS., Springer (2007) 177–188
19. Gear, C.W.: Differential-algebraic equations index transformations. SIAM J. Sci. Stat. Comput. **9**(1) (1988) 39–47
20. Gulwani, S., Tiwari, A.: Constraint-based approach for analysis of hybrid systems. [21] 190–203

21. Gupta, A., Malik, S., eds.: Computer Aided Verification, CAV 2008, Princeton, NJ, USA, Proceedings. In Gupta, A., Malik, S., eds.: CAV. Volume 5123 of LNCS., Springer (2008)
22. Harel, D., Kozen, D., Tiuryn, J.: Dynamic logic. MIT Press, Cambridge (2000)
23. Henzinger, T.A.: The theory of hybrid automata. In: LICS, Los Alamitos, IEEE Computer Society (1996) 278–292
24. Henzinger, T.A., Nicollin, X., Sifakis, J., Yovine, S.: Symbolic model checking for real-time systems. In: LICS, IEEE Computer Society (1992) 394–406
25. Hwang, I., Kim, J., Tomlin, C.: Protocol-based conflict resolution for air traffic control. Air Traffic Control Quarterly **15**(1) (2007)
26. Johansson, K.H., Sastry, S., Zhang, J., Lygeros, J.: Zeno hybrid systems. International Journal of Robust & Nonlinear Control **11** (2001) 435–451
27. Kolchin, E.R.: Differential Algebra and Algebraic Groups. Academic Press, New York (1972)
28. Livadas, C., Lygeros, J., Lynch, N.A.: High-level modeling and analysis of TCAS. Proc. IEEE - Special Issue on Hybrid Systems: Theory & Applications **88**(7) (2000) 926–947
29. Massink, M., Francesco, N.D.: Modelling free flight with collision avoidance. In: ICECCS, Los Alamitos, IEEE Computer Society (2001) 270–280
30. Piazza, C., Antoniotti, M., Mysore, V., Policriti, A., Winkler, F., Mishra, B.: Algorithmic algebraic model checking I: Challenges from systems biology. In Etessami, K., Rajamani, S.K., eds.: CAV. Volume 3576 of LNCS., Springer (2005) 5–19
31. Platzer, A.: Differential dynamic logic for verifying parametric hybrid systems. In Olivetti, N., ed.: TABLEAUX. Volume 4548 of LNCS., Springer (2007) 216–232
32. Platzer, A.: A temporal dynamic logic for verifying hybrid system invariants. In Artëmov, S.N., Nerode, A., eds.: LFCS. Volume 4514 of LNCS., Springer (2007) 457–471
33. Platzer, A.: Differential dynamic logic for hybrid systems. J. Autom. Reasoning **41**(2) (2008) 143–189
34. Platzer, A., Clarke, E.M.: The image computation problem in hybrid systems model checking. In Bemporad, A., Bicchi, A., Buttazzo, G., eds.: HSCC. Volume 4416 of LNCS., Springer (2007) 473–486
35. Platzer, A., Clarke, E.M.: Computing differential invariants of hybrid systems as fixedpoints. [21] 176–189
36. Platzer, A., Quesel, J.D.: KeYmaera: A hybrid theorem prover for hybrid systems. In Armando, A., Baumgartner, P., Dowek, G., eds.: IJCAR. Volume 5195 of LNCS., Springer (2008) 171–178
37. Platzer, A., Quesel, J.D.: Logical verification and systematic parametric analysis in train control. In Egerstedt, M., Mishra, B., eds.: HSCC. Volume 4981 of LNCS., Springer (2008) 646–649
38. Pour-El, M.B., Richards, I.: A computable ordinary differential equation which possesses no computable solution. Annals of Mathematical Logic **17** (1979) 61–90
39. Prajna, S., Jadbabaie, A.: Safety verification of hybrid systems using barrier certificates. [2] 477–492
40. Prajna, S., Jadbabaie, A., Pappas, G.J.: A framework for worst-case and stochastic safety verification using barrier certificates. IEEE T. Automat. Contr. **52**(8) (2007) 1415–1429
41. Rodríguez-Carbonell, E., Tiwari, A.: Generating polynomial invariants for hybrid systems. In Morari, M., Thiele, L., eds.: HSCC. Volume 3414 of LNCS., Springer (2005) 590–605
42. Rönkkö, M., Ravn, A.P., Sere, K.: Hybrid action systems. Theor. Comput. Sci. **290**(1) (2003) 937–973
43. Sankaranarayanan, S., Sipma, H., Manna, Z.: Constructing invariants for hybrid systems. [2] 539–554
44. Schobbens, P.Y., Raskin, J.F., Henzinger, T.A.: Axioms for real-time logics. Theor. Comput. Sci. **274**(1-2) (2002) 151–182

45. Sibirsky, K.S.: Introduction to Topological Dynamics. Noordhoff, Leyden (1975)
46. Tinelli, C.: Cooperation of background reasoners in theory reasoning by residue sharing. J. Autom. Reasoning **30**(1) (2003) 1–31
47. Tomlin, C., Pappas, G.J., Sastry, S.: Conflict resolution for air traffic management: a study in multi-agent hybrid systems. IEEE T. Automat. Contr. **43**(4) (1998) 509–521
48. Walter, W.: Ordinary Differential Equations. Springer (1998)
49. Zhou, C., Ravn, A.P., Hansen, M.R.: An extended duration calculus for hybrid real-time systems. In Grossman, R.L., Nerode, A., Ravn, A.P., Rischel, H., eds.: Hybrid Systems. Volume 736 of LNCS., Springer (1992) 36–59