A Complete Uniform Substitution Calculus for Differential Dynamic Logic

André Platzer*

Abstract

This article introduces a relatively complete proof calculus for differential dynamic logic \( (dL) \) that is entirely based on uniform substitution, a proof rule that substitutes a formula for a predicate symbol everywhere. Uniform substitutions make it possible to rely on concrete axioms rather than axiom schemata, substantially simplifying implementations. Instead of subtle schema variables and soundness-critical side conditions on the occurrence patterns of logical variables to restrict their infinitely many schema instances to sound ones, the resulting calculus adopts only a finite number of ordinary \( dL \) formulas as axioms, which uniform substitutions instantiate soundly. The static semantics of differential dynamic logic and the soundness-critical restrictions it imposes on proof steps is captured exclusively in uniform substitutions and variable renamings as opposed to being spread in delicate ways across the prover implementation. In addition to sound uniform substitutions, this article introduces differential forms for differential dynamic logic that make it possible to internalize differential invariants, differential substitutions, and derivations as first-class axioms to reason about differential equations. The axiomatization is proved to be sound and relatively complete.

Keywords: Differential dynamic logic, Uniform substitution, Axioms, Differentials, Static semantics, Axiomatization

1 Introduction

Differential dynamic logic \((dL)\) \([5,7]\) is a logic for proving correctness properties of hybrid systems. It has a sound and complete proof calculus relative to differential equations \([5,7]\) and a sound and complete proof calculus relative to discrete systems \([7]\). Both sequent calculi \([5]\) and Hilbert-type axiomatizations \([7]\) have been presented for \( dL \) but only the former has been implemented. The implementation of \( dL \)'s sequent calculus in KeYmaera \([12]\) makes it straightforward for users to prove properties of hybrid systems, because it provides proof rules performing natural decompositions for each operator. The downside is that the implementation of the rule schemata and their different and subtle side conditions on occurrence constraints and relations of reading

*Computer Science Department, Carnegie Mellon University, Pittsburgh, USA aplatzer@cs.cmu.edu
An extended abstract has appeared at CADE 2015 \([10]\) with its proofs listed in \([11]\).
A. Platzer

A Complete Uniform Substitution Calculus for Differential Dynamic Logic

and writing of variables as well as rule applications in context is quite nontrivial and inflexible in KeYmaera.

The goal of this article is to identify how to, instead, make it straightforward to implement the axioms and proof rules of differential dynamic logic in a parsimonious way by writing down a finite list of axioms (concrete formulas, not axiom schemata that represent an infinite list of axioms subject to sophisticated soundness-critical schema variable matching and side condition checking implementations). The resulting calculus require multiple axioms to be combined with one another to obtain the effect that a user would want for proving a hybrid system conjecture. This article argues that this is still a net win for hybrid systems, because a substantially simpler prover core is easier to implement correctly, and the need to combine multiple axioms to obtain user-level proof steps can be achieved equally well by appropriate tactics, which are not soundness-critical. Furthermore, multiple axioms, each with a more elementary effect, yield a more modular and flexible reasoning technique.

To achieve this goal, this article follows observations for differential game logic [9] that highlight the significance and elegance of uniform substitutions, a classical proof rule for first-order logic [2, §35,40]. Uniform substitutions uniformly instantiate predicate and function symbols with formulas and terms, respectively, as functions of their arguments. In the presence of the nontrivial binding structure that nondeterminism and differential equations of hybrid programs induce for the dynamic modalities of differential dynamic logic, flexible but sound uniform substitutions become more complex for $dL$, but can still be read off elegantly from its static semantics. In fact, $dL$'s static semantics is solely captured in the implementation of uniform substitution (and variable renaming), thereby leading to a completely modular proof calculus.

This approach is dual to other successful ways of solving the intricacies and subtleties of substitutions [1,3] by imposing occurrence side conditions on axiom schemata and proof rules, which is what uniform substitutions can get rid of. The uniform substitution framework shares many goals with other logical frameworks [4], including leading to smaller soundness-critical cores, more flexibility when augmenting reasoning techniques, and reducing the gap between a logic and its theorem prover. Logical frameworks shine when renaming and substitution of the object language are in line with those of the meta-language. Uniform substitutions are, arguably, better adopted for yielding a parsimonious approach for languages with the intricate binding of imperative and especially hybrid system dynamics in which, e.g., the same occurrence of a variable can be both free and bound. Or in which side conditions on axiom schemata include that one term is a solution of a symbolic initial value problem for a differential equation described by another term [5, 7]. Uniform substitutions lead to an exceedingly parsimonious core. They reduce axioms to one concrete formula as opposed to an algorithm that accepts infinitely many formulas under certain side conditions. They also reduce proof rules to a pair of concrete formulas as opposed to programs that transform formulas. Derived axioms, derived rules, rule application mechanisms, lemmas, definitions, and parametric invariant search are all definable from these primitives.

This article introduces a static and dynamic semantics for differential-form $dL$, proves coincidence lemmas (Section 2) and uniform substitution lemmas, culminating in a soundness proof for uniform substitutions (Section 3), which lead to a parsimoniously implementable axiomatization (Section 4). The calculus exploits the new differential forms that this article adds to $dL$ for
internalizing differential invariants [6], differential cuts [6 8], differential ghosts [8], differential substitutions, total differentials and Lie-derivations [6 8] as first-class citizens in \(dL\), culminating in entirely modular axioms for differential equations and a superbly modular soundness proof (Section 5). This approach is to be contrasted with earlier less modular approaches for differential invariants that were based on complex built-in rules that mixed multiple reasoning aspects into one special rule [6 8]. The relationship to related work from previous presentations of differential dynamic logic [5 7] continues to apply except that \(dL\) now internalizes differential equation reasoning axiomatically via differential forms. The logic is proved to be sound and relatively complete (Section 5). This gives a highly modular approach where the static semantics, uniform substitutions, and axioms are all defined separately and do not interact.

2 Differential-Form Differential Dynamic Logic

This section presents differential-form differential dynamic logic, which adds differential forms to differential dynamic logic [5 7] in order to internalize reasoning about differential equations and differentials themselves as first-class citizens into the logic.

2.1 Syntax

Formulas and hybrid programs (HPs) of \(dL\) are defined by simultaneous induction based on the following definition of terms. Similar simultaneous inductions are used throughout the proofs for \(dL\). The set of all variables is \(\mathcal{V}\). For any subst \(V \subseteq \mathcal{V}\) is \(V_\prime \overset{\text{def}}{=} \{x' : x \in V\}\) the set of differential symbols \(x'\) for the variables in \(V\). Function symbols are written \(f, g, h\), predicate symbols \(p, q, r\), and variables \(x, y, z \in \mathcal{V}\) with corresponding differential symbols \(x', y', z' \in \mathcal{V}'\). Program constants are \(a, b, c\).

Definition 1 (Terms). Terms are defined by this grammar (with \(\theta, \eta, \theta_1, \ldots, \theta_k\) as terms, \(x \in \mathcal{V}\) as variable, \(x' \in \mathcal{V}'\) differential symbol, and \(f\) function symbol):

\[
\theta, \eta ::= x | x' | f(\theta_1, \ldots, \theta_k) | \theta + \eta | \theta \cdot \eta | (\theta)' \]

Number literals such as 0,1 are allowed as function symbols without arguments that are always interpreted as the numbers they denote. Beyond differential symbols \(x'\), differential-form \(dL\) allows differentials \((\theta)'\) of terms \(\theta\) as terms for the purpose of axiomatically internalizing reasoning about differential equations.

Definition 2 (Hybrid program). Hybrid programs (HPs) are defined by the following grammar (with \(\alpha, \beta\) as HPs, program constant \(a\), variable \(x\), term \(\theta\) possibly containing \(x\), and formula \(\psi\) of first-order logic of real arithmetic):

\[
\alpha, \beta ::= a | x:=\theta | x' := \theta | ?\psi | x' = \theta & \psi | \alpha \cup \beta | \alpha; \beta | \alpha^* \]
Assignments $x := \theta$ of $\theta$ to variable $x$, tests $?\psi$ of the formula $\psi$ in the current state, differential equations $x' = \theta \& \psi$ restricted to the evolution domain constraint $\psi$, nondeterministic choices $\alpha \cup \beta$, sequential compositions $\alpha; \beta$, and nondeterministic repetition $\alpha^*$ are as usual in $\mathbf{dL}$ [5, 7]. The assignment $x := \theta$ instantaneously changes the value of $x$ to that of $\theta$. The test $?\psi$ checks whether $\psi$ is true in the current state and discards the program execution otherwise. The continuous evolution $x' = \theta \& \psi$ will follow the differential equation $x' = \theta$ for any amount of time, nondeterministically, without leaving the evolution domain constraint $\psi$. The nondeterministic choice $\alpha \cup \beta$ either follows subprogram $\alpha$ or $\beta$, nondeterministically. The sequential composition $\alpha; \beta$ first follows $\alpha$ and then, upon completion of $\alpha$, follows $\beta$. The nondeterministic repetition $\alpha^*$ repeats $\alpha$ any number of times, nondeterministically.

The effect of the differential assignment $x' := \theta$ to differential symbol $x'$ is similar to the effect of the assignment $x := \theta$ to variable $x$, except that it changes the value of the differential symbol $x'$ around instead of the value of $x$. It is not to be confused with the differential equation $x' = \theta$, which will follow said differential equation continuously for an arbitrary amount of time. The differential assignment $x' := \theta$, instead, only assigns the value of $\theta$ to the differential symbol $x'$ discretely once at an instant of time. Program constants $a$ are uninterpreted, i.e., their behavior depends on the interpretation in the same way that the values of function symbols $f$ and predicate symbols $p$ depend on their interpretation.

**Definition 3** ($\mathbf{dL}$ formula). The formulas of (differential-form) differential dynamic logic ($\mathbf{dL}$) are defined by the grammar (with $\mathbf{dL}$ formulas $\phi, \psi$, terms $\theta, \eta, \theta_1, \ldots, \theta_k$, predicate symbol $p$, quantifier symbol $C$, variable $x$, HP $\alpha$):

$$\phi, \psi ::= \theta \geq \eta \mid p(\theta_1, \ldots, \theta_k) \mid C(\phi) \mid \neg \phi \mid \phi \& \psi \mid \forall x \phi \mid \exists x \phi \mid [\alpha]\phi \mid \langle \alpha \rangle \phi$$

Operators $>, \leq, <, \lor, \rightarrow, \leftrightarrow$ are definable, e.g., $\phi \rightarrow \psi$ as $\neg(\phi \land \neg \psi)$. Likewise $[\alpha]\phi$ is equivalent to $\neg(\langle \alpha \rangle \neg \phi)$ and $\forall x \phi$ equivalent to $\neg \exists x \neg \phi$. The modal formula $[\alpha]\phi$ expresses that $\phi$ holds after all runs of $\alpha$, while the dual $\langle \alpha \rangle \phi$ expresses that there is a run of $\alpha$ after which $\phi$ holds. Quantifier symbols $C$ (with formula $\phi$ as argument), i.e., higher-order predicate symbols that bind all variables of $\phi$, are unnecessary but are included since they internalize contextual congruence reasoning efficiently with uniform substitutions.

**Example 1** (Simple car). The $\mathbf{dL}$ formula

$$v \geq 0 \land b > 0 \rightarrow [(a := -b \cup a := 5); x' = v, v' = a \& v \geq 0]^* v \geq 0$$

expresses that a car starting with nonnegative velocity $v \geq 0$ and braking power $b > 0$ will always have nonnegative velocity when following a HP that repeatedly provides a nondeterministic control choice between putting the acceleration $a$ to braking $(a := -b)$ or to a positive constant $(a := 5)$ before following the differential equation system $x' = v, v' = a \& v \geq 0$ restricted to the evolution domain constraint $v \geq 0$ for any amount of time. The formula in (1) is always true because the car never moves backward. But similar questions quickly become challenging, e.g., about safe distances to other cars or for models with more detailed physical dynamics.
2.2 Dynamic Semantics

A state is a mapping from variables \( \mathcal{V} \) and differential symbols \( \mathcal{V}' \) to \( \mathbb{R} \). The set of states is denoted \( \mathcal{S} \). Let \( \mathcal{V}'_r \) denote the state that agrees with state \( \mathcal{V} \) except for the value of variable \( x \), which is changed to \( r \in \mathbb{R} \), and accordingly for \( \mathcal{V}'_r \). The interpretation of a function symbol \( f \) with arity \( n \) (i.e., \( n \) arguments) is a (smooth) function \( I(f) : \mathbb{R}^n \to \mathbb{R} \) of \( n \) arguments.

**Definition 4** (Semantics of terms). For each interpretation \( I \), the semantics of a term \( \theta \) in a state \( \mathcal{V} \in \mathcal{S} \) is its value \( [\theta]I\mathcal{V} \) in \( \mathbb{R} \). It is defined inductively as follows

1. \([x]I\mathcal{V} = \mathcal{V}(x)\) for variable \( x \in \mathcal{V} \)
2. \([x']I\mathcal{V} = \mathcal{V}(x')\) for differential symbol \( x' \in \mathcal{V}' \)
3. \([f(\theta_1,\ldots,\theta_k)]I\mathcal{V} = I(f)([\theta_1]I\mathcal{V},\ldots,[\theta_k]I\mathcal{V})\) for function symbol \( f \)
4. \([\theta + \eta]I\mathcal{V} = [\theta]I\mathcal{V} + [\eta]I\mathcal{V}\)
5. \([\theta \cdot \eta]I\mathcal{V} = [\theta]I\mathcal{V} \cdot [\eta]I\mathcal{V}\)
6. \([((\theta)')I\mathcal{V}] = \sum_{x \in \mathcal{V}} \mathcal{V}(x') \frac{d[\theta]I\mathcal{V}_x}{dx} = \sum_{x \in \mathcal{V}} \mathcal{V}(x') \frac{d[\theta]I\mathcal{V}_x}{dx} \]

Time-derivatives are undefined in an isolated state \( \mathcal{V} \). The clou is that differentials can still be given a local semantics: \( [(\theta)'I\mathcal{V}] \) is the sum of all (analytic) spatial partial derivatives at \( \mathcal{V} \) of the value of \( \theta \) by all variables \( x \) (or rather their values \( X \)) multiplied by the corresponding tangent described by the value \( \mathcal{V}(x') \) of differential symbol \( x' \). That sum over all variables \( x \in \mathcal{V} \) has finite support, because \( \theta \) only mentions finitely many variables \( x \) and the partial derivative by variables \( x \) that do not occur in \( \theta \) is 0. The spatial derivatives exist since \( [\theta]I\mathcal{V} \) is a composition of smooth functions, so smooth. Thus, the semantics of \( [(\theta)'I\mathcal{V}] \) is the differential\(^1\) of (the value of) \( \theta \), hence a differential one-form giving a real value for each tangent vector (i.e., point of a vector field) described by the values \( \mathcal{V}(x') \). The values \( \mathcal{V}(x') \) of the differential symbols \( x' \) select the direction in which \( x \) changes, locally. They are the local shadow of \( \frac{dx}{dt} \) if only that time-derivative even existed, which it does not in an isolated state \( \mathcal{V} \). Along the flow of (the vector field corresponding to a) differential equation, though, the value of the differential \( (\theta)'I\mathcal{V} \) coincides with the analytic time-derivative of \( \theta \) (Lemma 13).

The interpretation of predicate symbol \( p \) with arity \( n \) is an \( n \)-ary relation \( I(p) \subseteq \mathbb{R}^n \). The interpretation of quantifier symbol \( C \) is a functional \( I(C) \) mapping subsets \( M \subseteq \mathcal{S} \) of states where its argument is true to subsets \( I(C)(M) \subseteq \mathcal{S} \) of states where \( C \) applied to that argument is then true.

**Definition 5** (d\(\mathcal{L} \)) semantics). The semantics of a d\(\mathcal{L} \) formula \( \phi \), for each interpretation \( I \) with a corresponding set of states \( \mathcal{S} \), is the subset \([\phi]I \subseteq \mathcal{S} \) of states in which \( \phi \) is true. It is defined inductively as follows

1. \([\theta \geq \eta]I = \{ \mathcal{V} \in \mathcal{S} : [\theta]I\mathcal{V} \geq [\eta]I\mathcal{V} \} \)
2. \([p(\theta_1,\ldots,\theta_k)]I = \{ \mathcal{V} \in \mathcal{S} : ([\theta_1]I\mathcal{V},\ldots,[\theta_k]I\mathcal{V}) \in I(p) \} \)
3. \([C(\theta)]I = I(C)([\theta]I) \) for quantifier symbol \( C \)
4. \([\neg \phi]I = ([\phi]I)^{C} = \mathcal{S} \setminus [\phi]I \)

\(^1\)A slight abuse of notation rewrites the differential as \( [(\theta)'I\mathcal{V}] = d[\theta]I\mathcal{V} = \sum_{i=1}^{n} \frac{d[\theta]I\mathcal{V}_x}{dx} dx_i \) when \( x_1,\ldots,x_n \) are the variables in \( \theta \) and their differentials \( dx_i \) form the basis of the cotangent space, which, when evaluated at a point \( \mathcal{V} \) whose values \( \mathcal{V}(x') \) determine the tangent vector alias vector field, coincides with Def.4
5. \([\phi \land \psi]I = [\phi]I \cap [\psi]I\)
6. \([\exists x \phi]I = \{v \in \mathcal{S} : \forall v' \in [\phi]I \text{ for some } r \in \mathbb{R}\}\)
7. \([\{x\} \phi]I = [\phi]I \circ [\phi]I = \{v : \omega \in [\phi]I \text{ for some } \omega \text{ such that } (v, \omega) \in [\phi]I\}\)
8. \([\{x\} \phi]I = [\neg \{x\} \neg \phi]I = \{v : \omega \in [\phi]I \text{ for all } \omega \text{ such that } (v, \omega) \in [\phi]I\}\)

A d\(\mathcal{L}\) formula \(\phi\) is valid in \(I\), written \(I \models \phi\), iff \([\phi]I = \mathcal{S}\), i.e. \(v \in [\phi]I\) for all states \(v\). Formula \(\phi\) is valid, written \(\models \phi\), iff \(I \models \phi\) for all interpretations \(I\).

The interpretation of a program constant \(a\) is a state-transition relation \(I(a) \subseteq \mathcal{S} \times \mathcal{S}\), where \((v, \omega) \in I(a)\) iff HP \(a\) can run from initial state \(v\) to final state \(\omega\).

**Definition 6** (Transition semantics of HPs). For each interpretation \(I\), each HP \(\alpha\) is interpreted semantically as a binary transition relation \([\alpha]I \subseteq \mathcal{S} \times \mathcal{S}\) on states, defined inductively by

1. \([a]I = I(a)\) for program constants \(a\)
2. \([x := \theta]I = \{(v, v') : r = [\theta]Iv\} = \{(v, \omega) : \omega = v \text{ except } [x]I\omega = [\theta]Iv\}\)
3. \([x' := \theta]I = \{(v, v') : r = [\theta]Iv\} = \{(v, \omega) : \omega = v \text{ except } [x']I\omega = [\theta]Iv\}\)
4. \([\gamma v]I = \{(v, v) : v \in [\gamma]I\}\)
5. \([x' = \theta \land \psi]I = \{(v, \omega) : I, \varphi \models x' = \theta \land \psi\}I\) for all \(0 \leq \zeta \leq r\), for some function \(\varphi : [0, r] \to \mathcal{S}\) of some duration \(r\) for which all \(\varphi(\zeta)(x') = \frac{d\varphi(t)(x)}{dt}(\zeta)\) exist and \(v = \varphi(0)\) on \(\{x'\}\) and \(\omega = \varphi(r)\); i.e., \(\varphi\) solves the differential equation and satisfies \(\psi\) at all times. In case \(r = 0\), the only condition is that \(v = \omega\) on \(\{x'\}\) and \(\omega(x') = [\theta]I\omega\) and \(\omega \in [\psi]I\).
6. \([\alpha \cup \beta]I = [\alpha]I \cup [\beta]I\)
7. \([\alpha; \beta]I = [\alpha]I \circ [\beta]I = \{(v, \omega) : (v, \mu) \in [\alpha]I, (\mu, \omega) \in [\beta]I\}\)
8. \([\alpha^n]I = ([\alpha]I)^n = \bigcup_{n \in \mathbb{N}} [\alpha^n]I\) with \(\alpha^{n+1} \equiv \alpha^n\); \(\alpha^0 \equiv ?true\)

where \(\rho^*\) denotes the reflexive transitive closure of relation \(\rho\).

The equality in \([\alpha^x]I\) follows from the Scott-continuity of HPs \([9, \text{Lemma } 3.7]\). The initial values \(v(x')\) of differential symbols \(x'\) do not influence the behavior of \((v, \omega) \in [x' = \theta \land \psi]I\), because they may not be compatible with the time-derivatives for the differential equation, e.g. in \(x' := 1; x' = 2\), with a \(x'\) mismatch. The final values \(\omega(x')\) after \(x' = \theta \land \psi\) will coincide with the derivatives at the final state, though.

Functions and predicates are interpreted by interpretation \(I\) and are only influenced indirectly by \(v\) through the values of their arguments. So \(p(e) \to [x := x + 1]p(e)\) is valid if \(x\) is not in \(e\) since the change in \(x\) does not change whether \(p(e)\) is true (Lemma\(\mathcal{L}_2\)). By contrast \(p(x) \to [x := x + 1]p(x)\) is invalid, since it is false when \(I(p) = \{d : d \leq 5\}\) and \(v(x) = 4.5\). If the semantics of \(p\) were to depend on the state \(v\) instead of just \(I\), then there would be no discernible relationship between the truth-values of \(p\) in different states, so not even \(p \to [x := x + 1]p\) would be valid.

### 2.3 Static Semantics

The dynamic semantics gives meaning to d\(\mathcal{L}\) formulas and HPs but is on its own quite inaccessible for reasoning purposes. The static semantics of d\(\mathcal{L}\) and HPs defines some computable aspects of
their behavior that can be read off directly from their syntactic structure without running their programs or evaluating their dynamical effects. The most important aspects of the static semantics concern free or bound occurrences of variables (which are closely related to the notions of scope and definitions/uses in compilers). Bound variables \(x\) are those that are bound by \(\forall x\) or \(\exists x\), but also those that are bound by modalities such as \([x := 5y]\) or \((x' = 1)\) or \([x := 1 \cup x' = 1]\) or \([x := 1 \cup \text{true}]\). In either case, the scope of the bound variable \(x\) is limited to the postcondition of the quantifier or of the modality.

Free variables are those that may be read, so the semantics of a formula depends only on the value of its free variables (Lemma 4). Bound variables may be written to, so their value may change with its quantifier or during the execution of a HP (Lemma 1). The notions of free and bound variables are defined by simultaneous induction in the subsequent definitions: free variables for terms \(\text{FV}(\theta)\), formulas \(\text{FV}(\phi)\), and HPs \(\text{FV}(\alpha)\), as well as bound variables for formulas \(\text{BV}(\phi)\) and for HPs \(\text{BV}(\alpha)\). For HPs, there will be a need to distinguish must-bound variables (MBV) that are bound/written to on all executions of \(\alpha\) from (may-)bound variables (BV) which are bound on some (not necessarily all) execution paths of \(\alpha\). For example, \([x := 1 \cup (x := 0; y := x + 1)]\) has bound variables \(\{x, y\}\) but must-bound variables only \(\{x\}\), because \(y\) is not written to in the first choice. This complication does not happen for quantifiers or strictly nested languages like \(\lambda\)-calculi.

**Definition 7** (Bound variable). The set \(\text{BV}(\phi) \subseteq \mathcal{V} \cup \mathcal{V}'\) of bound variables of dL formula \(\phi\) is defined inductively as

\[
\begin{align*}
\text{BV}(\theta \geq \eta) &= \text{BV}(p(\theta_1, \ldots, \theta_k)) = \emptyset \\
\text{BV}(C(\phi)) &= \mathcal{V} \cup \mathcal{V}' \\
\text{BV}(\neg \phi) &= \text{BV}(\phi) \\
\text{BV}(\phi \wedge \psi) &= \text{BV}(\phi) \cup \text{BV}(\psi) \\
\text{BV}(\forall x \phi) &= \text{BV}(\exists x \phi) = \{x\} \cup \text{BV}(\phi) \\
\text{BV}([\alpha] \phi) &= \text{BV}(\langle \alpha \rangle \phi) = \text{BV}(\alpha) \cup \text{BV}(\phi)
\end{align*}
\]

**Definition 8** (Free variable). The set \(\text{FV}(\theta) \subseteq \mathcal{V} \cup \mathcal{V}'\) of free variables of term \(\theta\), i.e. those that occur in \(\theta\), is defined inductively as

\[
\begin{align*}
\text{FV}(x) &= \{x\} \\
\text{FV}(x') &= \{x'\} \\
\text{FV}(f(\theta_1, \ldots, \theta_k)) &= \text{FV}(\theta_1) \cup \cdots \cup \text{FV}(\theta_k) \\
\text{FV}(\theta + \eta) &= \text{FV}(\theta \cdot \eta) = \text{FV}(\theta) \cup \text{FV}(\eta) \\
\text{FV}((\theta)) &= \text{FV}(\theta) \cup \text{FV}(\theta')
\end{align*}
\]

The set \(\text{FV}(\phi)\) of free variables of dL formula \(\phi\), i.e. all those that occur in \(\phi\) outside the scope of quantifiers or modalities binding it, is defined inductively as

\[
\text{FV}(\theta \geq \eta) = \text{FV}(\theta) \cup \text{FV}(\eta)
\]
programs without program constants, then MBV

\( \text{Definition 9} \) (Bound variable). The set \( \text{BV}(\alpha) \subseteq \mathcal{V} \cup \mathcal{V}' \) of \textit{bound variables} of HP \( \alpha \), i.e. all those that may potentially be written to, is defined inductively:

\[
\begin{align*}
\text{BV}(a) &= \mathcal{V} \cup \mathcal{V}' \\
\text{BV}(x := \theta) &= \{x\} \\
\text{BV}(x' := \theta) &= \{x'\} \\
\text{BV}(?y) &= \emptyset \\
\text{BV}(x' = \theta \land \psi) &= \{x, x'\} \\
\text{BV}(\alpha \cup \beta) &= \text{BV}(\alpha) \cup \text{BV}(\beta) \\
\text{BV}(\alpha^*) &= \text{BV}(\alpha)
\end{align*}
\]

\( \text{Definition 10} \) (Must-bound variable). The set \( \text{MBV}(\alpha) \subseteq \text{BV}(\alpha) \subseteq \mathcal{V} \cup \mathcal{V}' \) of \textit{must-bound variables} of HP \( \alpha \), i.e. all those that must be written to on all paths of \( \alpha \), is defined inductively as

\[
\begin{align*}
\text{MBV}(a) &= \emptyset \\
\text{MBV}(\alpha) &= \text{BV}(\alpha) \\
\text{MBV}(\alpha \cup \beta) &= \text{MBV}(\alpha) \cap \text{MBV}(\beta) \\
\text{MBV}(\alpha; \beta) &= \text{MBV}(\alpha) \cup \text{MBV}(\beta) \\
\text{MBV}(\alpha^*) &= \emptyset
\end{align*}
\]

Obviously, \( \text{MBV}(\alpha) \subseteq \text{BV}(\alpha) \). If \( \alpha \) is only built by sequential compositions from atomic programs without program constants, then \( \text{MBV}(\alpha) = \text{BV}(\alpha) \).
Definition 11 (Free variable). The set $\text{FV}(\alpha) \subseteq \mathcal{V} \cup \mathcal{V}'$ of free variables of HP $\alpha$, i.e. all those that may potentially be read, is defined inductively as

$$
\begin{align*}
\text{FV}(a) &= \mathcal{V}' \cup \mathcal{V}'' \\
\text{FV}(x := \theta) &= \text{FV}(x' := \theta) = \text{FV}(\theta) \\
\text{FV}(?\psi) &= \text{FV}(\psi) \\
\text{FV}(x' = \theta \land \psi) &= \{x\} \cup \text{FV}(\theta) \cup \text{FV}(\psi) \\
\text{FV}(\alpha \cup \beta) &= \text{FV}(\alpha) \cup \text{FV}(\beta) \\
\text{FV}(\alpha ; \beta) &= \text{FV}(\alpha) \cup (\text{FV}(\beta) \setminus \text{MBV}(\alpha)) \\
\text{FV}(\alpha^*) &= \text{FV}(\alpha)
\end{align*}
$$

The variables of HP $\alpha$, whether free or bound, are $V(\alpha) = \text{FV}(\alpha) \cup \text{BV}(\alpha)$. The simpler definition $\text{FV}(\alpha \cup \beta) = \text{FV}(\alpha) \cup \text{FV}(\beta)$ would be correct, but the results would be less precise then. Unlike $x$, the left-hand side $x'$ of differential equations is not added to the free variables of $\text{FV}(x' = \theta \land \psi)$, because its behavior does not depend on the initial value of differential symbols $x'$, only the initial value of $x$. Free and bound variables are the set of all variables $\mathcal{V}'$ and differential symbols $\mathcal{V}''$ for program constants $a$, because their effect depends on the interpretation $I$, so may read and write all $\text{FV}(a) = \text{BV}(a) = \mathcal{V}' \cup \mathcal{V}''$ but not on all paths $\text{MBV}(a) = \emptyset$. Subsequent results about free and bound variables are, thus, vacuously true when program constants occur.

The static semantics defines which variables are readable or writable. There may not be any run of $\alpha$ in which a variable is read or written to. If $x \not\in \text{FV}(\alpha)$, though, then $\alpha$ cannot read the value of $x$. If $x \not\in \text{BV}(\alpha)$, it cannot write to $x$. Def. 11 is parsimonious. For example, $x$ is not a free variable of the following program

$$(x := 1 \cup x := 2); z := x + y$$

because $x$ is never actually read, since $x$ must have been defined on every execution path of the first part before being read by the second part. No execution of the above program, thus, depends on the initial value of $x$, which is why it is not a free variable. This would have been different for the less precise definition

$$
\text{FV}(\alpha ; \beta) = \text{FV}(\alpha) \cup \text{FV}(\beta)
$$

There is a limit to the precision with which any static analysis can determine which variables are really read or written [13]. The static semantics in Def. 11 will, e.g., consider $x$ a free variable of the following program even though no execution could read it, because they fail test $?\text{false}$ when running the branch reading $x$:

$$
z := 0; (?\text{false}; z := z + x)^*
$$

The signature, i.e. set of function, predicate, quantifier symbols, and program constants in $\phi$ is denoted by $\Sigma(\phi)$ (accordingly for terms and programs). It is defined like $\text{FV}(\phi)$ except that all occurrences are free since $d\mathcal{Z}$ has no higher-order binding constructs. The dynamic semantics interprets variables in $\mathcal{V} \cup \mathcal{V}'$ by the state $\nu$. The symbols in $\Sigma(\phi)$ are interpreted by the interpretation $I$. 

---

A. Platzer  
A Complete Uniform Substitution Calculus for Differential Dynamic Logic
2.4 Correctness of Static Semantics

The static semantics of $\mathcal{L}_d$ is correct, i.e. soundly captures how the dynamic semantics reads and writes variables. The following result reflects that HPs have bounded effect: for a variable $x$ to be modified during a run of $\alpha$, $x$ needs the be a bound variable in HP $\alpha$, i.e. $x \in \text{BV}(\alpha)$, so that $\alpha$ can write to $x$. The converse is not true, because $\alpha$ may bind a variable $x$, e.g. by having an assignment to $x$, that never actually changes the value of $x$, such as $x := x$ or because the assignment can never be executed. The following program, for example, binds $x$ but will never change the value of $x$ because there is no way of satisfying the test $\text{false}: (?\text{false}; x := 42) \cup z := x + 1$.

**Lemma 1 (Bound effect).** If $(v, \omega) \in [\alpha] I$, then $v = \omega$ on $\text{BV}(\alpha)^C$.

**Proof.** The proof is by a straightforward structural induction on $\alpha$.

- Since $\text{BV}(a) = \mathcal{V} \cup \mathcal{V}'$, the statement is vacuously true for program constant $a$, because $\text{BV}(a)^C = \emptyset$.
- $(v, \omega) \in [x := \theta] I = \{(v, \omega) : \omega = v$ except that $[x] I \omega = [\theta] I v\}$ implies that $v = \omega$ except for $\{x\} = \text{BV}(x := \theta)$.
- $(v, \omega) \in [x' := \theta] I = \{(v, \omega) : v = \omega$ except that $[x'] I \omega = [\theta] I v\}$ implies that $v = \omega$ except for $\{x'\} = \text{BV}(x' := \theta)$.
- $(v, \omega) \in [\psi] I = \{(v, \psi) : \psi \in [\psi] I$ i.e. $\nu \in [\psi] I$ fits to $\text{BV}(\psi) = \emptyset\}$
- $(v, \omega) \in [\alpha \cup \beta] I = [\alpha] I \cup [\beta] I$ implies $(v, \omega) \in [\alpha] I$ or $(v, \omega) \in [\beta] I$, which, by induction hypothesis, implies $v = \omega$ on $\text{BV}(\alpha)^C$ or $v = \omega$ on $\text{BV}(\beta)^C$, respectively. Either case implies $v = \omega$ on $(\text{BV}(\alpha) \cup \text{BV}(\beta))^C = \text{BV}(\alpha \cup \beta)^C$.
- $(v, \omega) \in [\alpha; \beta] I = [\alpha] I \circ [\beta] I$, i.e. there is a $\mu$ such that $(v, \mu) \in [\alpha] I$ as well as $(\mu, \omega) \in [\beta] I$. So, by induction hypothesis, $v = \mu$ on $\text{BV}(\alpha)^C$ and $\mu = \omega$ on $\text{BV}(\beta)^C$. By transitivity, $v = \omega$ on $(\text{BV}(\alpha) \cup \text{BV}(\beta))^C = \text{BV}(\alpha; \beta)^C$.
- $(v, \omega) \in [\alpha^*] I = \bigcup_{n \in \mathbb{N}} [\alpha^n] I$, then there is an $n \in \mathbb{N}$ and a sequence $v_0 = v, v_1, \ldots, v_n = \omega$ such that $(v_i, v_{i+1}) \in [\alpha]$ for all $i < n$. By $n$ uses of the induction hypothesis, $v_i = v_{i+1}$ on $\text{BV}(\alpha)^C$ for all $i < n$. Thus, $v = v_0 = v_n = \omega$ on $\text{BV}(\alpha)^C = \text{BV}(\alpha^*)^C$.

Similarly, only $\text{BV}(\phi)$ change their value during the evaluation of formulas.

The value of a term only depends on the values of its free variables. When evaluating a term $\theta$ in two states $v, \tilde{v}$ that differ widely but agree on the free variables $\text{FV}(\theta)$ of $\theta$, the values of $\theta$ in both states coincide. Accordingly, the value of a term will agree for different interpretations $I, J$ that agree on the symbols $\Sigma(\theta)$ that occur in $\theta$.

**Lemma 2 (Coincidence).** If $v = \tilde{v}$ on $\text{FV}(\theta)$ and $I = J$ on $\Sigma(\theta)$, then $[\theta] I v = [\theta] J \tilde{v}$.

**Proof.** The proof is by structural induction on $\theta$. IH is short for induction hypothesis.

- $[x] I v = v(x) = \tilde{v}(x) = [x] J \tilde{v}$ for variable $x$ since $v = \tilde{v}$ on $\text{FV}(x) = \{x\}$.
Corollary 3. The semantics of differentials simplifies to a sum over the free variables:

\[
[(\theta')I]v = \sum_{x \in FV(\theta) \cap \mathcal{X}} v(x') \frac{\partial [\theta]I}{\partial x}(v) = \sum_{x \in FV(\theta) \cap \mathcal{X}} v(x') \frac{\partial [\theta]Iv_x^X}{\partial x}
\]

By a more subtle argument, the values of $d\mathcal{L}$ formulas also only depend on the values of their free variables. When evaluating $d\mathcal{L}$ formula $\phi$ in two states $v$, $\tilde{v}$ that differ but agree on the free variables $FV(\phi)$ of $\phi$, the (truth) values of $\phi$ in both states coincide. Lemma 4 and 5 are proved by simultaneous induction, reflecting the simultaneously inductive definitions of formulas and programs.

Lemma 4 (Coincidence). If $v = \tilde{v}$ on $FV(\phi)$ and $I = J$ on $\Sigma(\phi)$, then $v \in [(\phi)I] \iff \tilde{v} \in [(\phi)J]$.

Proof. The proof is by structural induction on $\phi$. To simplify the proof, doubly negated existential quantifiers are considered structurally smaller than universal quantifiers and doubly negated diamond modalities smaller than box modalities.

1. $v \in \{p(\theta_1, \ldots, \theta_k)\}I$ iff $[(\theta_1)Iv_1, \ldots, (\theta_k)Iv_k] \in I(p)$ iff $\{[(\theta_1)Jv_1, \ldots, (\theta_k)Jv_k] \in J(p)\}$ iff $\tilde{v} \in \{p(\theta_1, \ldots, \theta_k)\}J$ by Lemma 2 since $FV(\theta_i) \subseteq FV(p(\theta_1, \ldots, \theta_k))$ and $I$ and $J$ were assumed to agree on the function symbol $p$ that occurs in the formula.

2. $v \in [\theta \geq \eta]I$ iff $[(\theta)Iv] \geq [\eta]Iv$ iff $[(\theta)Jv] \geq [\eta]Jv$ iff $\tilde{v} \in [\theta \geq \eta]J$ by Lemma 2 since $FV(\theta) \cup FV(\eta) \subseteq FV(\theta) \geq \eta$ and the interpretation of $\geq$ is fixed.

3. $v \in [C(\phi)]I = J(C)((\phi)J)$ iff, by IH, $\tilde{v} \in [C(\phi)J] = J(C)((\phi)J)$ since $v = \tilde{v}$ on $FV(C(\phi)) = \mathcal{X}' \cup \mathcal{X}''$, so $v = \tilde{v}$, and $I = J$ on $\Sigma(C(\phi)) = \{C\} \cup \Sigma(\phi)$, so $I(C) = J(C)$ and, by induction hypothesis, $[(\phi)I] = [(\phi)J]$. 

4. \( \nu \in [\neg \phi]I \) iff \( \nu \not\in [\phi]^J \) iff, by IH, \( \nu \not\in [\phi]^J \) iff \( \nu \in [\neg \phi]J \) by induction hypothesis as \( \text{FV}(\neg \phi) = \text{FV}(\phi) \).

5. \( \nu \in [\phi \land \psi]I \) iff \( \nu \in [\phi]I \cap [\psi]I \) iff, by IH, \( \nu \in [\phi]I \cap [\psi]I \) iff \( \nu \in [\phi \land \psi]J \) by induction hypothesis using \( \text{FV}(\phi \land \psi) = \text{FV}(\phi) \cup \text{FV}(\psi) \).

6. \( \nu \in [\exists x \phi]I \) iff \( \nu^r \in [\phi]I \) for some \( r \in \mathbb{R} \) iff \( \nu^r \in [\phi]I \) for some \( r \in \mathbb{R} \) iff \( \nu^r \in [\exists x \phi]J \) for the same \( r \) by induction hypothesis using that \( \nu^r = \nu^r \) on \( \text{FV}(\phi) \subseteq \{x\} \cup \text{FV}(\exists x \phi) \).

7. The case \( \forall x \phi \) follows from the equivalence \( \forall x \phi \equiv \neg \exists x \neg \phi \) using \( \text{FV}(\neg \exists x \neg \phi) = \text{FV}(\forall x \phi) \).

8. \( \nu \in [(\alpha)\phi]I \) iff there is a \( \omega \) such that \( (\nu, \omega) \in [(\alpha)]I \) and \( \omega \in [\phi]I \). Since \( \nu = \tilde{\nu} \) on \( \text{FV}(\langle \alpha \rangle \phi) \supseteq \text{FV}(\alpha) \) and \( (\nu, \omega) \in [(\alpha)]I \), Lemma[5] implies with \( I = J \) on \( \Sigma(\alpha) \subseteq \Sigma(\langle \alpha \rangle \phi) \) that there is an \( \bar{\omega} \) such that \( (\tilde{\nu}, \bar{\omega}) \in [(\alpha)]J \) and \( \omega = \bar{\omega} \) on \( \text{FV}(\langle \alpha \rangle \phi) \cup \text{MBV}(\alpha) = \text{FV}(\alpha) \cup (\text{FV}(\phi) \setminus \text{MBV}(\alpha)) \cup \text{MBV}(\alpha) = \text{FV}(\alpha) \cup \text{FV}(\phi) \cup \text{MBV}(\alpha) \supseteq \text{FV}(\phi) \).

9. \( \nu \in [(\alpha)\phi]I = [\neg (\langle \alpha \rangle \neg \phi)]I \) iff \( \nu \not\in [(\alpha)\neg \phi]^J \) iff \( \nu \not\in [(\alpha)\neg \phi]^J \) iff \( \nu \in [(\alpha)\phi]J \) by induction hypothesis using \( \text{FV}(\langle \alpha \rangle \neg \phi) = \text{FV}(\langle \alpha \rangle \phi) \).

\( \square \)

In a sense, the runs of an HP \( \alpha \) also only depend on the values of its free variables, because its behavior cannot depend on the values of variables that it never reads. That is, if \( \nu = \tilde{\nu} \) on \( \text{FV}(\alpha) \) and \( (\nu, \omega) \in [(\alpha)]I \), then there is an \( \bar{\omega} \) such that \( (\tilde{\nu}, \bar{\omega}) \in [(\alpha)]J \) and \( \omega \) and \( \bar{\omega} \) agree in some sense. There is a subtlety, though. The resulting states \( \omega \) and \( \bar{\omega} \) will only continue to agree on \( \text{FV}(\alpha) \) and the variables that are bound on the particular path that \( \alpha \) took for the transition \( (\nu, \omega) \in [(\alpha)]I \). On variables \( z \) that are neither free (so the initial states \( \nu \) and \( \tilde{\nu} \) have not been assumed to coincide) nor bound on the particular path that \( \alpha \) took, \( \omega \) and \( \bar{\omega} \) may continue to disagree, because \( z \) has not been written to.

**Example 2.** Let \( (\nu, \omega) \in [(\alpha)]I \). It is not enough to assume \( \nu = \tilde{\nu} \) only on \( \text{FV}(\alpha) \) in order to guarantee \( \omega = \bar{\omega} \) on \( \text{V}(\alpha) \) for some \( \bar{\omega} \) such that \( (\tilde{\nu}, \bar{\omega}) \in [(\alpha)]J \), because

\[
\alpha \overset{\text{def}}{=} x := 1 \cup y := 2
\]

will force the final states to agree only on either \( x \) or on \( y \), whichever one was assigned to during the respective run of \( \alpha \), not on both \( \text{BV}(\alpha) = \{x, y\} \), even though any initial states \( \nu, \tilde{\nu} \) agree on \( \text{FV}(\alpha) = \emptyset \). Note that this can only happen because \( \text{MBV}(\alpha) = \emptyset \neq \text{BV}(\alpha) = \{x, y\} \).
Yet, the respective resulting states $\omega$ and $\tilde{\omega}$ do agree on the variables that are bound on all paths of $\alpha$, rather than just somewhere in $\alpha$. That is on the must-bound variables of $\alpha$. If initial states agree on (at least) all free variables $\text{FV}(\alpha)$ that HP $\alpha$ may read, then the final states agree on those as well as on all variables that $\alpha$ must write, i.e. on $\text{MBV}(\alpha)$.

**Lemma 5** (Coincidence). If $v = \tilde{v}$ on $V \supseteq \text{FV}(\alpha)$ and $I = J$ on $\Sigma(\alpha)$ and $(v, \omega) \in [\alpha]I$, then there is an $\tilde{\omega}$ such that $(\tilde{v}, \tilde{\omega}) \in [\alpha]I$ and $\omega = \tilde{\omega}$ on $V \cup \text{MBV}(\alpha)$.

\[
\begin{array}{ccc}
\text{on } BV(\alpha) & \\ \hline
\nu & \alpha & \omega \\
\hline
\nu & \alpha & \tilde{\omega} \\
\hline
\end{array}
\]

\[
\begin{array}{ccc}
V \supseteq \text{FV}(\alpha) & \\ \hline
\nu & \alpha & V \cup \text{MBV}(\alpha) \\
\hline
\nu & \alpha & V \cup \text{MBV}(\alpha) \\
\hline
\end{array}
\]

**Proof.** The proof is by induction on the structural complexity of $\alpha$, where $\alpha^*$ is considered to be structurally more complex than HPs of any length but with less nested repetitions, which induces a well-founded order on HPs. For atomic programs $\alpha$, for which $\text{BV}(\alpha) = \text{MBV}(\alpha)$, it is enough to conclude agreement on $V(\alpha) \overset{\text{def}}{=} \text{FV}(\alpha) \cup \text{BV}(\alpha) = \text{FV}(\alpha) \cup \text{MBV}(\alpha)$, because any variable in $V \setminus V(\alpha)$ is in $\text{BV}(\alpha)^c$, which remains unchanged by $\alpha$ according to Lemma 1.

1. Since $\text{FV}(\alpha) = \gamma' \cup \gamma''$ so $v = \tilde{v}$, the statement is vacuously true for program constant $a$.

2. $(v, \omega) \in [x := \theta]I = \{(v, \omega) : \omega = v \text{ except that } [x]I\omega = [\theta]Iv\}$ then there is a transition $(\tilde{v}, \tilde{\omega}) \in [x := \theta]J$ and $\tilde{\omega}(x) = [x]J\tilde{\omega} = [\theta]J\tilde{v} = [\theta]Iv = [x]I\omega = v(x)$ by Lemma 4, since $v = \tilde{v}$ on $\text{FV}(x := \theta) = \text{FV}(\theta)$ and $I = J$ on $\Sigma(\theta)$. So, $\omega = \tilde{\omega}$ on $\text{BV}(x := \theta) = \{x\}$. Also, $v = \omega$ on $\text{BV}(x := \theta)^c$ and $\tilde{v} = \tilde{\omega}$ on $\text{BV}(x := \theta)^c$ by Lemma 4. Since $v = \tilde{v}$ on $\text{FV}(x := \theta)$, these imply $\omega = \tilde{\omega}$ on $\text{FV}(x := \theta) \setminus \text{BV}(x := \theta)$. Since $\omega = \tilde{\omega}$ on $\text{BV}(x := \theta)$ had been shown already, this implies $\omega = \tilde{\omega}$ on $V(x := \theta)$.

3. $(v, \omega) \in [x' := \theta]I = \{(v, v') : r = [\theta]Iv\}$. As $[\theta]Iv = [\theta]J\tilde{v}$ by Lemma 2, since $\text{FV}(\theta) \subseteq \text{FV}(x' := \theta)$, this implies $(\tilde{v}, \tilde{\omega}) \in [x' := \theta]J = \{(\tilde{v}, \tilde{\omega}) : r = [\theta]J\tilde{v}\}$. By construction $\omega = \tilde{\omega}$ on $\text{BV}(x' := \theta) = \{x'\}$ and they continue to agree on $\text{FV}(x' := \theta) \setminus \text{BV}(x' := \theta)$.

4. $(v, \omega) \in [?\psi]I = \{(v, v) : v \in [?\psi]I \text{ i.e. } v \in [?\psi]I\}$ then $\omega = v$ by Def. 6. Since, $v \in [?\psi]I$ and $v = \tilde{v}$ on $\text{FV}(?\psi)$ and $I = J$ on $\Sigma(\psi) = \Sigma(?\psi)$, Lemma 4 implies that $\tilde{v} \in [?\psi]J$, so $(\tilde{v}, \tilde{\omega}) \in [?\psi]J$. So $v = \tilde{v}$ on $\text{FV}(?\psi) = \text{FV}(?\psi)$ since $\text{BV}(?\psi) = \emptyset$.

5. $(v, \omega) \in [x' = \theta & ?\psi]I$ implies that there is an $\tilde{\omega}$ reached from $\tilde{v}$ by following the differential equation for the same amount it took to reach $\omega$ from $v$. The solution will be the same, because $I = J$ on $\Sigma(x' = \theta & ?\psi)$ and $v = \tilde{v}$ on $\text{FV}(x' = \theta & ?\psi)$, which, using Lemma 4, contains all the variables whose values the differential equation solution depends on. Thus, both solutions agree on all variables that evolve during the continuous evolution, i.e. $\text{BV}(x' = \theta & ?\psi)$. Thus, $(\tilde{v}, \tilde{\omega}) \in [x' = \theta & ?\psi]J$ and $\omega = \tilde{\omega}$ on $V(x' = \theta & ?\psi)$.

6. $(v, \omega) \in [\alpha \cup \beta]I = [\alpha]I \cup [\beta]I$ implies $(v, \omega) \in [\alpha]I$ or $(v, \omega) \in [\beta]I$, which since $V \supseteq \text{FV}(\alpha) \cup \text{FV}(\beta) \supseteq \text{FV}(\alpha)$ and $V \supseteq \text{FV}(\alpha \cup \beta) \supseteq \text{FV}(\beta)$ implies, by induction hypothesis, that
there is an \( \tilde{\omega} \) such that \(( \tilde{\nu}, \tilde{\omega} ) \in [\alpha]_I \) and \( \omega = \tilde{\omega} \) on \( V \cup \text{MBV}(\alpha) \) or that there is an \( \tilde{\omega} \) such that \(( \tilde{\nu}, \tilde{\omega} ) \in [\beta]_I \) and \( \omega = \tilde{\omega} \) on \( V \cup \text{MBV}(\beta) \), respectively. In either case, there is a \( \tilde{\omega} \) such that \(( \tilde{\nu}, \tilde{\omega} ) \in [\alpha \cup \beta]_I \) and \( \omega = \tilde{\omega} \) on \( V \cup \text{MBV}(\alpha \cup \beta) \), because \([\alpha]_I \subseteq [\alpha \cup \beta]_I \) and \([\beta]_I \subseteq [\alpha \cup \beta]_I \) and \( \text{MBV}(\alpha \cup \beta) = \text{MBV}(\alpha) \cap \text{MBV}(\beta) \).

7. \((\nu, \omega) \in [\alpha; \beta]_I = [\alpha]_I \circ [\beta]_I \), i.e. there is a state \( \mu \) such that \((\nu, \mu) \in [\alpha]_I \) and \((\mu, \omega) \in [\beta]_I \).

Since \( V \supseteq \text{FV}(\alpha; \beta) \supseteq \text{FV}(\alpha) \), by induction hypothesis, there is a \( \tilde{\mu} \) such that \((\tilde{\nu}, \tilde{\mu}) \in [\alpha]_I \) and \( \mu = \tilde{\mu} \) on \( V \cup \text{MBV}(\alpha) \). Since \( V \supseteq \text{FV}(\alpha; \beta) \), so \( V \cup \text{MBV}(\alpha) \supseteq \text{FV}(\alpha; \beta) \cup \text{MBV}(\alpha) = \text{FV}(\alpha) \cup (\text{FV}(\beta) \setminus \text{MBV}(\alpha)) \cup \text{MBV}(\alpha) = \text{FV}(\alpha) \cup \text{FV}(\beta) \cup \text{MBV}(\alpha) \supseteq \text{FV}(\beta) \) by Def.\,1, and since \((\mu, \omega) \in [\beta]_I \), the induction hypothesis implies that there is an \( \tilde{\omega} \) such that \((\tilde{\mu}, \tilde{\omega}) \in [\beta]_I \) and \( \omega = \tilde{\omega} \) on \((V \cup \text{MBV}(\alpha)) \cup \text{MBV}(\beta) = V \cup \text{MBV}(\alpha; \beta) \).

8. \((\nu, \omega) \in [\alpha^n]_I = \bigcup_{n \in \mathbb{N}} [\alpha^n]_I \) iff there is an \( n \in \mathbb{N} \) such that \((\nu, \omega) \in [\alpha^n]_I \). The case \( n = 0 \) follows from the assumption \( \nu = \tilde{\nu} \) on \( V \supseteq \text{FV}(\alpha) \), since \( \omega = \tilde{\omega} \) holds in that case and \( \text{MBV}(\alpha^0) = \emptyset \). The case \( n > 0 \) proceeds as follows. Since \( \text{FV}(\alpha^n) = \text{FV}(\alpha^n) = \text{FV}(\alpha) \), the induction hypothesis applied to the structurally simpler HP \( \alpha^n \) with less loops implies that there is an \( \tilde{\omega} \) such that \((\tilde{\nu}, \tilde{\omega}) \in [\alpha^n]_I \) and \( \omega = \tilde{\omega} \) on \((V \cup \text{MBV}(\alpha^n)) \supseteq V = V \cup \text{MBV}(\alpha^n) \), since \( \text{MBV}(\alpha^n) = \emptyset \). Since \([\alpha^n]_I \subseteq [\alpha]_I \) by Def.
3, this concludes the proof.

When assuming \( \nu \) and \( \tilde{\nu} \) to agree on all \( V(\alpha) \), whether free or bound, \( \omega \) and \( \tilde{\omega} \) will continue to agree on \( V(\alpha) \). Using hybrid computation sequences, the agreement in Lemma\,5 can also be shown to continue to hold for \( \omega = \tilde{\omega} \) on \( V \cup W \), where \( W \) is the set of must-bound variables on the hybrid computation sequence that \( \alpha \) actually took for the transition \((\nu, \omega) \in [\alpha]_I \), which could be larger than \( \text{MBV}(\alpha) \).

**Corollary 6** (Coincidence). If \( \nu = \tilde{\nu} \) on \( V(\alpha) \) and \( I = I \) on \( \Sigma(\alpha) \) and \((\nu, \omega) \in [\alpha]_I \), then there is an \( \tilde{\omega} \) such that \((\tilde{\nu}, \tilde{\omega}) \in [\alpha]_I \) and \( \omega = \tilde{\omega} \) on \( V(\alpha) \). The same continues to hold if \( \nu = \tilde{\nu} \) only on \( V(\alpha) \setminus \text{MBV}(\alpha) \).

**Proof.** By Lemma\,5 with \( V = V(\alpha) \supseteq \text{FV}(\alpha) \) or \( V = V(\alpha) \setminus \text{MBV}(\alpha) \), respectively. \( \Box \)
This concludes the static semantics of $\mathcal{DL}$, which characterizes syntactically what kind of state change formulas and HPs may cause ($\text{BV}(\phi), \text{BV}(\alpha)$) and what part of the state their values and behavior depends on ($\text{FV}(\phi), \text{FV}(\alpha)$). Together, these provide uniform substitutions with all they need to know to determine what changes during substitutions will go unnoticed (variables are not free so changes to their value has no affect) and what state-change an expression may cause itself (variables that are not bound cannot change their value during that expression). The interaction of those free and bound variables is what uniform substitutions need to determine whether a particular uniform substitution preserves truth in a proof.

3 Uniform Substitutions

The uniform substitution rule $[\text{US}]$ from first-order logic [2, §35,40] substitutes all occurrences of predicate $p(\cdot)$ by a formula $\psi(\cdot)$, i.e. it replaces all occurrences of $p(\theta)$, for any (vectorial) term $\theta$, by the corresponding $\psi(\theta)$ simultaneously:

$$(\text{US}_1) \quad \frac{\phi}{\psi} \quad \frac{\phi}{p(\cdot)} \quad (\text{US}) \quad \frac{\phi}{\psi(\cdot)} \quad \frac{\phi}{\sigma(\phi)}$$

Soundness of rule $[\text{US}]$ requires all relevant substitutions of $\psi(\theta)$ for $p(\theta)$ to be admissible and requires that no $p(\theta)$ occurs in the scope of a quantifier or modality binding a variable of $\psi(\theta)$ other than the occurrences in $\theta$; see [2, §35,40].

This section develops rule $[\text{US}]$ as a more general and constructive definition of $[\text{US}]$. The $\mathcal{DL}$ calculus uses uniform substitutions that affect terms, formulas, and programs. A uniform substitution $\sigma$ is a mapping from expressions of the form $f(\cdot)$ to terms $\sigma f(\cdot)$, from $p(\cdot)$ to formulas $\sigma p(\cdot)$, from $C(\cdot)$ to formulas $\sigma C(\cdot)$, and from program constants $a$ to HPs $\sigma a$. Vectorial extensions are accordingly for uniform substitutions of other arities $k \geq 0$. Here $\cdot$ is a reserved function symbol of arity zero and _ a reserved quantifier symbol of arity zero, which are used to mark the positions where the respective argument, e.g., $\theta$ to $p(\cdot)$, will end up in the replacement $\sigma p(\cdot)$.

Example 3. The uniform substitution $\sigma = \{ f \mapsto x + 1, p(\cdot) \mapsto (\cdot \neq x) \}$ will substitute all occurrences of function symbol $f$ (which has arity 0 here) by $x + 1$ and simultaneously substitutes all occurrences of predicate symbol $p(\theta)$ with any argument $\theta$ by the corresponding $(\theta \neq x)$. Whether that uniform substitution is sound depends on admissibility of $\sigma$ for the formula $\phi$ in $[\text{US}]$. It is admissible (and thus sound) for

$$\begin{align*}
\text{US} & \quad [y := f]p(2f) \leftrightarrow [y := f]p(2y) \\
\text{US} & \quad [y := x + 1][2(x + 1) \neq x \leftrightarrow [y := x + 1]2y \neq x] \\
\sigma & = \{ f \mapsto x + 1, p(\cdot) \mapsto (\cdot \neq x) \}
\end{align*}$$

but is not admissible (and, in fact, would be unsound) for:

$$\begin{align*}
\text{clash} & \quad [x := f]p(x) \leftrightarrow p(f) \\
\text{US} & \quad [x := x + 1]x \neq x \leftrightarrow x + 1 \neq x \\
\sigma & = \{ f \mapsto x + 1, p(\cdot) \mapsto (\cdot \neq x) \}
\end{align*}$$
Figure 1 defines the result $\sigma(\phi)$ of applying to a $d\mathcal{L}$ formula $\phi$ the uniform substitution $\sigma$ that uniformly replaces all occurrences of function $f$ by a (instantiated) term and all occurrences of predicate $p$ or quantifier $C$ symbols by a (instantiated) formula as well as of program constant $a$ by a program. The notation $\sigma f(\cdot)$ denotes the replacement for $f(\cdot)$ according to $\sigma$, i.e. the value $\sigma f(\cdot)$ of function $\sigma$ at $f(\cdot)$. By contrast, $\sigma(\phi)$ denotes the result of applying $\sigma$ to $\phi$ according to Fig. 1 (likewise for $\sigma(\theta)$ and $\sigma(\alpha)$). The notation $f \in \sigma$ signifies that $\sigma$ replaces $f$, i.e. $\sigma f(\cdot) \neq f(\cdot)$. Finally, $\sigma$ is a total function when augmented with $\sigma g(\cdot) = g(\cdot)$ for all $g \not\in \sigma$. Accordingly for predicate symbols, quantifier symbols, and program constants. The cases for $g \not\in \sigma$, $p \not\in \sigma$, $C \not\in \sigma$ and $b \not\in \sigma$ then follow from the other cases but are listed explicitly for clarity. Observe that arguments are put in for the placeholders $\cdot$ and $\_\_$ recursively by uniform substitutions $\{\cdot \mapsto \sigma(\theta)\}$ in Fig. 1 which is defined since $\cdot$ is a function symbol and $\_\_$ a quantifier symbol, both of arity 0.

**Definition 12** (Admissible uniform substitution). The uniform substitution $\sigma$ is $U$-admissible for $\phi$ (or $\theta$ or $\alpha$, respectively) with respect to the set $U \subseteq \mathcal{V} \cup \mathcal{V}'$ iff $\mathrm{FV}(\sigma|_{\Sigma(\phi)}) \cap U = \emptyset$, where $\sigma|_{\Sigma(\phi)}$ is the restriction of $\sigma$ that only replaces symbols that occur in $\phi$ and $\mathrm{FV}(\sigma) = \bigcup_{f \in \sigma} \mathrm{FV}(\sigma f(\cdot)) \cup \bigcup_{p \in \sigma} \mathrm{FV}(\sigma p(\cdot))$ are the free variables that $\sigma$ introduces. The uniform substitution $\sigma$ is admissible for $\phi$ (or $\theta$ or $\alpha$, respectively) iff all admissibility conditions during its application according to Fig. 1 hold, which check that the bound variables $U$ of each operator are not free in the substitution on its arguments, i.e. $\sigma$ is $U$-admissible. Otherwise the substitution clashes so its result $\sigma(\phi)$ ($\sigma(\theta)$ or $\sigma(\alpha)$) is not defined.

The proof rule $\text{US}$ is only applicable if $\sigma$ is admissible for $\phi$. In all subsequent results, all applications of uniform substitutions are required to be defined (no clash).

### 3.1 Correctness of Uniform Substitutions

Let $I^R_\nu$ denote the interpretation that agrees with interpretation $I$ except for the interpretation of predicate symbol $p$, which is changed to $R \subseteq \mathbb{R}$. Accordingly for predicate symbols of other arities, for function symbols $f$, and quantifier symbols $C$. The semantic counterpart of the syntactic operation of uniform substitutions are adjoint interpretations. When admissible, the value of an expression in its adjoint interpretation agrees with the value of its uniform substitute in the original interpretation.

**Corollary 7** (Substitution adjoints). The adjoint interpretation $\sigma^*_\nu I$ to substitution $\sigma$ for $I$, $\nu$ is the interpretation that agrees with $I$ except that for each function symbol $f \in \sigma$, predicate symbol $p \in \sigma$, quantifier symbol $C \in \sigma$, and program constant $a \in \sigma$:

- $\sigma^*_\nu I(f) : \mathbb{R} \rightarrow \mathbb{R}; d \mapsto [\sigma f(\cdot)] I^d \nu$
- $\sigma^*_\nu I(p) = \{d \in \mathbb{R} : \nu \in [\sigma p(\cdot)] I^d \}$
- $\sigma^*_\nu I(C) : \mathcal{V}(\mathbb{R}) \rightarrow \mathcal{V}(\mathbb{R}); R \mapsto [\sigma C(\_\_)] I^R$
- $\sigma^*_\nu I(a) = [\sigma a] I$

If $\nu = \omega$ on $\mathrm{FV}(\sigma)$, then $\sigma^*_\nu I = \sigma^*_\omega I$. If $\sigma$ is $U$-admissible for $\phi$ (or $\theta$ or $\alpha$, respectively) and $\nu = \omega$ on $U^c$, then

$$[\theta]\sigma^*_\nu I = [\theta]\sigma^*_\omega I \text{ i.e. } [\theta]\sigma^*_\nu I\mu = [\theta]\sigma^*_\omega I\mu \text{ for all states } \mu$$
\[\begin{align*}
\sigma(x) &= x & \text{for variable } x \in \mathcal{V} \\
\sigma(x') &= x' & \text{for differential symbol } x' \in \mathcal{V}' \\
\sigma(f(\theta)) &= (\sigma(f))(\sigma(\theta)) \overset{\text{def}}{=} \{ \cdot \mapsto \sigma(\theta) \}(\sigma f(\cdot)) & \text{for function symbol } f \in \sigma \\
\sigma(g(\theta)) &= g(\sigma(\theta)) & \text{for function symbol } g \not\in \sigma \\
\sigma(\theta + \eta) &= \sigma(\theta) + \sigma(\eta) \\
\sigma(\theta \cdot \eta) &= \sigma(\theta) \cdot \sigma(\eta) \\
\sigma((\theta)^\prime) &= (\sigma(\theta))^\prime & \text{if } \sigma \mathcal{V} \cup \mathcal{V}'\text{-admissible for } \theta \\
\sigma(\theta \geq \eta) &\equiv \sigma(\theta) \geq \sigma(\eta) \\
\sigma(p(\theta)) &\equiv (\sigma(p))(\sigma(\theta)) \overset{\text{def}}{=} \{ \cdot \mapsto \sigma(\theta) \}(\sigma p(\cdot)) & \text{for predicate symbol } p \in \sigma \\
\sigma(q(\theta)) &\equiv q(\sigma(\theta)) & \text{for predicate symbol } q \not\in \sigma \\
\sigma(C \phi) &\equiv \sigma(C)(\sigma(\phi)) \overset{\text{def}}{=} \{ \cdot \mapsto \sigma(\phi) \}(\sigma C(\cdot)) & \text{if } \sigma \mathcal{V}' \cup \mathcal{V}'\text{-admissible for } \phi, C \in \sigma \\
\sigma(C \phi) &\equiv C(\sigma(\phi)) & \text{if } \sigma \mathcal{V}' \cup \mathcal{V}'\text{-admissible for } \phi, C \not\in \sigma \\
\sigma(\neg \phi) &\equiv \neg \sigma(\phi) \\
\sigma(\phi \land \psi) &\equiv \sigma(\phi) \land \sigma(\psi) \\
\sigma(\forall x \phi) &\equiv \forall x \sigma(\phi) & \text{if } \sigma \{x\}\text{-admissible for } \phi \\
\sigma(\exists x \phi) &\equiv \exists x \sigma(\phi) & \text{if } \sigma \{x\}\text{-admissible for } \phi \\
\sigma(\lbrack \alpha \rbrack \phi) &\equiv \lbrack \sigma(\alpha) \rbrack \sigma(\phi) & \text{if } \sigma \text{ BV}(\sigma(\alpha))\text{-admissible for } \phi \\
\sigma(\langle \alpha \rangle \phi) &\equiv \langle \sigma(\alpha) \rangle \sigma(\phi) & \text{if } \sigma \text{ BV}(\sigma(\alpha))\text{-admissible for } \phi \\
\sigma(\alpha) &\equiv \sigma a & \text{for program constant } a \in \sigma \\
\sigma(b) &\equiv b & \text{for program constant } b \not\in \sigma \\
\sigma(x := \theta) &\equiv x := \sigma(\theta) \\
\sigma(x' := \theta) &\equiv x' := \sigma(\theta) \\
\sigma(x' = \theta \land \psi) &\equiv x' = \sigma(\theta) \land \sigma(\psi) & \text{if } \sigma \{x, x'\}\text{-admissible for } \theta, \psi \\
\sigma(? \psi) &\equiv ? \sigma(\psi) \\
\sigma(\alpha \cup \beta) &\equiv \sigma(\alpha) \cup \sigma(\beta) \\
\sigma(\alpha; \beta) &\equiv \sigma(\alpha); \sigma(\beta) & \text{if } \sigma \text{ BV}(\sigma(\alpha))\text{-admissible for } \beta \\
\sigma(\alpha^* ) &\equiv (\sigma(\alpha))^* & \text{if } \sigma \text{ BV}(\sigma(\alpha))\text{-admissible for } \alpha
\end{align*}\]

Figure 1: Recursive application of uniform substitution \(\sigma\)
\[
[\phi]_{\sigma^v} I = [\phi]_{\sigma^v}^o I
\]
\[
[\alpha]_{\sigma^v} I = [\alpha]_{\sigma^v}^o I
\]

**Proof.** For well-definedness of \(\sigma^v I\), note that \(\sigma^v I(f)\) is a smooth function since \(\sigma f(\cdot)\) has smooth values. First, \(\sigma^v I(a) = [\sigma a] I = \sigma^o I(a)\) holds because the adjoint to \(\sigma\) for \(I, v\) in the case of programs is independent of \(v\) (the program has access to its respective initial state at runtime). Likewise \(\sigma^v I(C) = \sigma^o I(C)\) for quantifier symbols. By Lemma 2, \([\sigma f(\cdot)] I^d \nu = [\sigma f(\cdot)] I^d \phi\) when \(\nu = \omega\) on \(\text{FV}(\sigma f(\cdot)) \subseteq \text{FV}(\sigma)\). Also \(\nu \in [\sigma p(\cdot)] I^d \nu\) iff \(\omega \in [\sigma p(\cdot)] I^d \nu\) by Lemma 4 when \(\nu = \omega\) on \(\text{FV}(\sigma p(\cdot)) \subseteq \text{FV}(\sigma)\). Thus, \(\sigma^v I = \sigma^o I\) when \(\nu = \omega\) on \(\text{FV}(\sigma)\).

If \(\sigma\) is \(U\)-admissible for \(\phi\) (or \(\theta\) or \(\alpha\)), then \(\text{FV}(\sigma f(\cdot)) \cap U = \emptyset\), i.e. \(\text{FV}(\sigma f(\cdot)) \subseteq U^c\) for every function symbol \(f \in \Sigma(\phi)\) (or \(\theta\) or \(\alpha\)) and likewise for predicate symbols \(p \in \Sigma(\phi)\). Since \(\nu = \omega\) on \(U^c\) was assumed, so \(\sigma^o I = \sigma^v I\) on the function and predicate symbols in \(\Sigma(\phi)\) (or \(\theta\) or \(\alpha\)). Finally \(\sigma^o I = \sigma^v I\) on \(\Sigma(\phi)\) (or \(\Sigma(\theta)\) or \(\Sigma(\alpha)\), respectively) implies that \(\sigma \in [\phi]_{\sigma^v} I\) iff \(\nu \in [\phi]_{\sigma^o} I\) by Lemma 4 and that \([\theta]_{\sigma^v} I = [\theta]_{\sigma^o} I\) by Lemma 2 and that \([\alpha]_{\sigma^o} I = [\alpha]_{\sigma^v} I\) by Lemma 5 respectively.

Substituting equals for equals is sound by the compositional semantics of \(dL^z\). The more general uniform substitutions are still sound, because interpretations of uniform substitutes correspond to interpretations of their adjoints. The semantic modification of adjoint interpretations has the same effect as the syntactic uniform substitution, proved by simultaneous induction. Recall that all substitutions in the following are assumed to be defined (not clash), otherwise the lemmas make no claim.

**Lemma 8** (Uniform substitution). The uniform substitution \(\sigma\) and its adjoint interpretation \(\sigma^v I, \nu\) to \(\sigma\) for \(I, \nu\) have the same semantics for terms \(\theta\):

\[
[\sigma(\theta)] I \nu = [\theta]_{\sigma^v} I \nu
\]

**Proof.** The proof is by structural induction on \(\theta\) and the structure of \(\sigma\).

- \([\sigma(x)] I \nu = [x] I \nu = [x]_{\sigma^v} I \nu\) since \(x \notin \sigma\) for variable \(x \in \mathcal{V}\).
- \([\sigma(x')] I \nu = [x'] I \nu = [x']_{\sigma^v} I \nu\) as \(x' \notin \sigma\) for differential symbol \(x' \in \mathcal{V}'\).
- \([\sigma(f(\theta))] I \nu = [\sigma(f(\sigma(\theta)))) I \nu = [\sigma(f(\sigma(\theta)))) I \nu \quad \text{by induction hypothesis twice, once for } \sigma(\theta) \text{ and once for } \sigma(\theta)\) for \(\sigma(\theta)\) on the smaller \(\theta\) and once for \(\sigma(\theta)\) on the possibly bigger term \(\sigma f(\cdot)\) but the structurally simpler uniform substitution \(\{ \cdot \mapsto \sigma(\theta) \} \ldots\) that is a substitution on the symbol \(\cdot\) of arity zero, not a substitution of functions with arguments. For well-foundedness of the induction note that the \(\cdot\) substitution only happens for function symbols \(f\) with at least one argument \(\theta\) (for \(\sigma \in \sigma\)) so not for \(\cdot\) itself.
- \([\sigma(g(\theta))] I \nu = [g(\sigma(\theta))] I \nu = I(g)([\sigma(\sigma(\theta))] I \nu) \quad \text{by induction hypothesis and since } I(g) = \sigma^v I(g)\) as the interpretation of \(g\) does not change in \(\sigma^v I\) for \(g \notin \sigma\).
- \([\sigma(\theta + \eta)] I \nu = [\sigma(\theta) + \sigma(\eta)] I \nu = [\sigma(\theta)] I \nu + [\sigma(\eta)] I \nu = [\theta + \eta]_{\sigma^v} I \nu\) by induction hypothesis.
Lemma 9 (Uniform substitution). The uniform substitution $\sigma$ and its adjoint interpretation $\sigma^* I, v$ to $\sigma$ for $I, v$ have the same semantics for formulas $\phi$:

$$v \in [\sigma(\phi)] I \iff v \in [\phi] \sigma^* I$$

Proof. The proof is by structural induction on $\phi$ and the structure on $\sigma$.

- $v \in [\sigma(\theta \cdot \eta)] I$ iff $v \in [\sigma(\theta)] I \cdot [\sigma(\eta)] I$ by induction hypothesis.
- $[\sigma((\theta'))] I = [\sigma(\theta')] I = \sum_x v(x') \frac{\partial [\sigma(\theta)] I^x}{\partial x} = \sum_x v(x') \frac{\partial [\sigma^* I^x \sigma^*]}{\partial x}$ by induction hypothesis, provided $\sigma$ is $\mathcal{V} \cup \mathcal{V}'$-admissible for $\theta$, i.e. does not introduce any variables or differential symbols, so that Corollary 7 implies $\sigma^* I = \sigma^* \sigma^* I$ for all $v, \omega$ (that agree on $(\mathcal{V} \cup \mathcal{V}')^c = \emptyset$, which imposes no condition on $v, \omega$).
The proof is by structural induction on Lemma 10 (Uniform substitution). The uniform substitution $\sigma$ and its adjoint interpretation $\sigma^+_\alpha I, \nu$ to $\sigma$ for $I, \nu$ have the same semantics for programs $\alpha$:

$$(v, \omega) \in [\sigma(\alpha)]I \iff (v, \omega) \in [\alpha]\sigma^+_\alpha I$$

Proof. The proof is by structural induction on $\alpha$.

- $(v, \omega) \in [\sigma(a)]I = [\sigma a]I = \sigma^+_\alpha I(a) = [a]\sigma^+_\alpha I$ for program constant $a \in \sigma$ (the proof is accordingly for $a \notin \sigma$).
- $(v, \omega) \in [\sigma(x := \theta)]I = [\sigma x := \theta]I$ iff $\nu = v^\sigma(\theta) I^\nu = v^\theta \sigma^+_\alpha I^\nu$ by Lemma 9 which is, thus, equivalent to $(v, \omega) \in [x := \theta]\sigma^+_\alpha I$.
- $(v, \omega) \in [\sigma(x' := \theta)]I = [\sigma x' := \theta]I$ iff $\omega = v^\sigma(\theta) I^\omega = v^\theta \sigma^+_\alpha I^\omega$ by Lemma 9 which is, thus, equivalent to $(v, \omega) \in [x' := \theta]\sigma^+_\alpha I$.
- $(v, \omega) \in [\sigma(? \psi)]I = [?? \psi]I$ iff $\omega = v$ and $v \in [\sigma(\psi)]I$, iff, by Lemma 9 $\omega = v$ and $v \in [\psi]\sigma^+_\alpha I$, which is equivalent to $(v, \omega) \in [?? \psi]\sigma^+_\alpha I$.
- $(v, \omega) \in [\sigma(x' = \theta \& \psi)]I = [\sigma x' = \theta \& \psi]I$ (provided $\sigma \{x, x'\}$-admissible for $\theta, \psi$) iff $\exists \theta : [0, T] \rightarrow \mathcal{P}$ with $\omega = v, \phi(T) = \omega$ and for all $t \geq 0$: $\theta(t) = [\sigma(\theta)]I \phi(t) = [\theta]\sigma^\phi_{\phi(t)} I \phi(t)$ by Lemma 8 as well as $\phi(t) \in [\sigma(\psi)]I$, which, by Lemma 9 is equivalent to $\theta(t) \in [\psi]\sigma^\phi_{\phi(t)} I$. Also $(v, \omega) \in [x' = \theta \& \psi]\sigma^+_\alpha I$ iff $\exists \theta : [0, T] \rightarrow \mathcal{P}$ with $\omega = v, \phi(T) = \omega$ and for all $t \geq 0$: $\theta(t) = [\theta]\sigma^\phi_{\phi(t)} I \phi(t)$ and $\phi(t) \in [\psi]\sigma^+_\alpha I$. Finally, $[\theta]\sigma^\phi_{\phi(t)} I = [\theta]\sigma^\phi_{\phi(t)} I = [\psi]\sigma^+_\alpha I$ by Corollary 7 since $\sigma \{x, x'\}$-admissible for $\theta, \psi$ and $\phi = \phi(t)$ on $\text{BV}(x' = \theta \& \psi)^{C} = \{x, x'\}$ by Lemma 10.
- $(v, \omega) \in [\sigma(\alpha \cup \beta)]I = [\sigma(\alpha \cup \sigma(\beta))]I = [\sigma(\alpha)]I \cup [\sigma(\beta)]I$, which, by induction hypothesis, is equivalent to $(v, \omega) \in [\alpha]\sigma^+_\alpha I \cup [\beta]\sigma^+_\alpha I$, which is equivalent to $(v, \omega) \in [\alpha]\sigma^+_\alpha I \cup [\beta]\sigma^+_\alpha I = [\alpha \cup \beta]\sigma^+_\alpha I$. 

\[\square\]
• \((v, \omega) \in [\sigma(\alpha; \beta)]I = [\sigma(\alpha); \sigma(\beta)]I = [\sigma(\alpha)]I \circ [\sigma(\beta)]I\) (provided \(\sigma\) is BV\((\sigma(\alpha))\)-admissible for \(\beta\)) iff there is a \(\mu\) such that \((v, \mu) \in [\sigma(\alpha)]I\) and \((\mu, \omega) \in [\sigma(\beta)]I\), which, by induction hypothesis, is equivalent to \((v, \mu) \in [\alpha]\sigma^v_\beta I\) and \((\mu, \omega) \in [\beta]\sigma^\mu_\beta I\). Yet, \([\beta]\sigma^\mu_\beta I = [\beta]\sigma^v_\beta I\) by Corollary\([7]\), because \(\sigma\) is BV\((\sigma(\alpha))\)-admissible for \(\beta\) and \(v = \omega\) on BV\((\sigma(\alpha))\)^{\ominus} by Lemma\([1]\) since \((v, \mu) \in [\sigma(\alpha)]I\). Finally, \((v, \mu) \in [\alpha]\sigma^v_\beta I\) and \((\mu, \omega) \in [\beta]\sigma^\mu_\beta I\) for some \(\mu\) is equivalent to \((v, \omega) \in [\alpha; \beta]\sigma^v_\beta I\).

• \((v, \omega) \in [\sigma(\alpha^*; \beta)]I = [(\sigma(\alpha))^*]I = [(\sigma(\alpha))]I^* = \bigcup_{n \in \mathbb{N}}([\sigma(\alpha)]I)^n\) (provided \(\sigma\) is BV\((\sigma(\alpha))\)-admissible for \(\alpha\)) iff there are \(n \in \mathbb{N}\) and \(v_0 = v, v_1, \ldots, v_n = \omega\) such that \((v_i, v_{i+1}) \in [\sigma(\alpha)]I\) for all \(i < n\). By \(n\) uses of the induction hypothesis, this is equivalent to \((v_i, v_{i+1}) \in [\alpha]\sigma^v_\beta I\) for all \(i < n\), which is equivalent to \((v_i, v_{i+1}) \in [\alpha]\sigma^\mu_\beta I\) by Corollary\([7]\), since \(\sigma\) is BV\((\sigma(\alpha))\)-admissible for \(\alpha\) and \(v_{i+1} = v_i\) on BV\((\sigma(\alpha))\)^{\ominus} by Lemma\([1]\) as \((v_i, v_{i+1}) \in [\sigma(\alpha)]I\) for all \(i < n\). Thus, this is equivalent to \((v, \omega) \in [\alpha^*]\sigma^v_\beta I = ([\alpha]\sigma^\mu_\beta I)^*\).

\[\square\]

### 3.2 Soundness

The uniform substitution lemmas are the key insights for the soundness of proof rule \([\text{US}]\) which is only applicable if its uniform substitution is defined.

**Theorem 11** (Soundness of uniform substitution). The rule \([\text{US}]\) is sound and so is its special case \([\text{US}_1]\). That is, if their premise is valid, then so is their conclusion.

\[\frac{\phi}{\sigma(\phi)}\text{ \([\text{US}]\)}\]

**Proof.** Let the premise \(\phi\) of \([\text{US}]\) be valid, i.e. \(v \in [\phi]I\) for all interpretations and states \(I, v\). To show that the conclusion is valid, consider any interpretation and state \(I, v\) and show \(v \in [\sigma(\phi)]I\). By Lemma\([9]\), \(v \in [\sigma(\phi)]I\) iff \(v \in [\phi]\sigma^v_\beta I\). The latter holds, because \(v \in [\phi]I\) for all \(I, v\), including for \(\sigma^v_\beta I, v\), by premise. The rule \([\text{US}_1]\) is the special case of \([\text{US}]\) where \(\sigma\) only substitutes predicate symbol \(p\).

Uniform substitutions can also be used to soundly instantiate locally sound proof rules just like proof rule \([\text{USR}]\) soundly instantiates axioms or other valid formulas (Theorem\([1]\)). The use of the proof rule from Theorem\([12]\) in a proof will be marked \([\text{USR}]\).

**Theorem 12** (Soundness of uniform substitution of rules). All uniform substitution instances (with \(FV(\sigma) = \emptyset\)) of locally sound inferences are locally sound, i.e. for any interpretation \(I\), the conclusion is valid in \(I\) if all its premises are valid in \(I\).

\[
\frac{\phi_1 \ldots \phi_n}{\psi}\text{ locally sound implies }\frac{\sigma(\phi_1) \ldots \sigma(\phi_n)}{\sigma(\psi)}\text{ locally sound}
\]

**Proof.** If

\[
\frac{\phi_1 \ldots \phi_n}{\psi}\text{ locally sound }\]

(2)
\( \langle \phi \rangle \quad \langle a \rangle p(\bar{x}) \leftrightarrow \neg [a] \neg p(\bar{x}) \)

\[ [:=] \quad \nu[x := f]p(x) \leftrightarrow p(f) \]

\[ [?] \quad \nu[?q]p \leftrightarrow (q \rightarrow p) \]

\[ \cup \quad \nu[a \cup b]p(\bar{x}) \leftrightarrow [a]p(\bar{x}) \land [b]p(\bar{x}) \]

\[ [:] \quad \nu[a;b]p(\bar{x}) \leftrightarrow [a][b]p(\bar{x}) \]

\[ [*] \quad \nu[a^*]p(\bar{x}) \leftrightarrow p(\bar{x}) \land [a]^*p(\bar{x}) \]

\[ K \quad \nu[a](p(\bar{x}) \rightarrow q(\bar{x})) \rightarrow ([a]p(\bar{x}) \rightarrow [a]q(\bar{x})) \]

\[ I \quad \nu[a^*](p(\bar{x}) \rightarrow [a]p(\bar{x})) \rightarrow (p(\bar{x}) \rightarrow [a^*]p(\bar{x})) \]

\[ V \quad \nu p \rightarrow [a]p \]

\[
\frac{\sigma(\phi_1) \ldots \sigma(\phi_n)}{\sigma(\psi)}
\]

where \( \sigma \) is any uniform substitution (for which the above results are defined, i.e. no clash) with \( \text{FV}(\sigma) = \emptyset \). To show this, consider any \( I \) in which all premises of (3) are valid, i.e. \( I \models \sigma(\phi_j) \) for all \( j \). That is, \( \nu \in [\sigma(\phi_j)]I \) for all \( \nu \) and all \( j \). By Lemma 9, \( \nu \in [\sigma(\phi_j)]I \) is equivalent to \( \nu \in [\sigma(\phi_j)]\sigma^*vI \), which, thus, also holds for all \( \nu \) and all \( j \). By Corollary 7, \( [\sigma(\phi_j)]\sigma^*vI = [\sigma(\phi_j)]\sigma_\omega I \) for any \( \omega \), since \( \text{FV}(\sigma) = \emptyset \).

Consequently, all premises of (2) are valid in \( \sigma^*vI \), i.e. \( \sigma^*vI \models \psi \) by local soundness of (2). That is, \( \nu \in [\psi]\sigma^*vI = [\psi]\sigma^*vI \) by Corollary 7 for all \( \nu \). By Lemma 9, \( \nu \in [\psi]\sigma^*vI \) is equivalent to \( \nu \in [\sigma(\psi)]I \), which continues to hold for all \( \nu \). Thus, \( I \models \sigma(\psi) \), i.e. the conclusion of (3) is valid in \( I \), hence (3) locally sound. Consequently, all uniform substitution instances (3) of locally sound inferences (2) with \( \text{FV}(\sigma) = \emptyset \) are locally sound.

\[\Box\]

## 4 Differential Dynamic Logic Axioms

Proof rules and axioms for a Hilbert-type axiomatization of \( \mathcal{dL} \) from prior work [7] are shown in Fig. 2 except that, thanks to proof rule US axioms and proof rules now comprise the finite list of concrete \( \mathcal{dL} \) formulas in Fig. 2 as opposed to an infinite collection of axioms from a finite list of axiom schemata along with schema variables, side conditions, and implicit instantiation rules. Soundness of the axioms follows from soundness of corresponding axiom schemata [7], but is easier to prove standalone, because it is a finite list of formulas without the need to prove soundness for all their instances. Soundness of axioms, thus, reduces to validity of one formula as
opposed to validity of all formulas that can be generated by the instantiation mechanism complying with the respective side conditions for that axiom schema.

The proof rules in Fig.2 are axiomatic rules, i.e. pairs of concrete dL formulas instantiated by [USR]. Soundness of axiomatic rules reduces to proving that their concrete conclusion formula is a consequence of their premise formula. Further, \( \bar{x} \) is the vector of all relevant variables, which is finite-dimensional, or, in practice, considered as a built-in vectorial term. Proofs in the uniform substitution dL calculus use US (and variable renaming such as \( \forall x p(x) \to \forall y p(y) \)) to instantiate the axioms from Fig.2 to the required form. [CT][CQ][CE] are congruence rules, which are included for efficiency to use axioms in any context, even if not needed for completeness.

**Remark 1.** The use of variable vector \( \bar{x} \) is not essential but simplifies concepts. An equivalent axiomatization is obtained when considering \( p(\bar{x}) \) to be a quantifier symbol of arity 0 in the axiomatization, or as \( C(true) \) with a quantifier symbol of arity 1. Neither replacements of quantifier symbols nor (vectorial) placeholders \( \cdot \) for the substitutions \( \{p(\cdot) \mapsto \psi\} \) that are used for \( p(\bar{x}) \) cause any free variables in the substitution. The mnemonic notation \( \sigma = \{p(\bar{x}) \mapsto \phi\} \) adopted for such uniform substitutions reminds that the variables \( \bar{x} \) are not free in \( \sigma \) even if they occur in the replacement \( \phi \).

Sound axioms are just valid formulas so true in all states. For example, in any state where \([a][b]p(\bar{x})\) is true, \([a;b]p(\bar{x})\) is true, too, by equivalence axiom [3]. Using that axiom to replace one by the other is a truth-preserving transformation, i.e. in any state in which one is true, the other is true, too. Sound rules are validity-preserving, i.e. the conclusion is valid if the premises are valid, which is weaker than truth-preserving transformations. For proof search, all dL axioms are written such that their assumption (for implications) or right-hand side (for equivalences) is structurally simpler.

**Real Quantifiers.** Besides (decidable) real arithmetic (whose use is denoted [R]), complete axioms for first-order logic can be adopted to express universal instantiation [\( \forall \)](if \( p \) is true of all \( x \) it is also true of constant function symbol \( f \)), distributivity [\( \forall \to \)] and vacuous quantification [\( \forall \psi \)] (predicate \( p \) of arity zero does not depend on \( x \)).

\[
\begin{align*}
(\forall i) & \quad (\forall x p(x)) \to p(f) \\
(\forall \to) & \quad \forall x (p(x) \to q(x)) \to (\forall x p(x) \to \forall x q(x)) \\
(\forall \psi) & \quad p \to \forall x p
\end{align*}
\]

**The Significance of Clashes.** This section illustrates how soundness-critical it is for [USR] to produce substitution clashes by showing unsound naïve instantiation attempts that [USR] prevents successfully. All such instantiations have to be disallowed in a corresponding axiom schema calculus. US clashes for substitutions that introduce a free variable into a bound context. Example [3] showed that even an occurrence of \( p(x) \) in a context where \( x \) is bound does not permit mentioning \( x \) in the replacement except in the \( \cdot \) places. US can directly handle even nontrivial binding structures,
though, e.g. from $[\cdot :=] $ with the substitution $\sigma = \{ f \mapsto x^2, p(\cdot) \mapsto [(z := x+z) ; z := x+yz] y \geq \cdot \}$:

\[ [x := f] p(x) \leftrightarrow p(f) \]

\[ [x := x^2] [(z := x+z) ; z := x+yz] y \geq x \leftrightarrow [(z := x^2+z) ; z := x^2+yz] y \geq x^2 \]

It is soundness-critical that US clashes when substituting a formula with a free dependence on a variable bound by the replacement of $p$ in $\mathbb{V}$ with a formula that mentions the bound variable $x$:

\[ p \rightarrow \forall x p \quad x \geq 0 \rightarrow \forall x (x \geq 0) \quad \{ p \mapsto x \geq 0 \} \]

It is soundness-critical that US clashes when substituting $p$ in vacuous program axiom $\mathbb{V}$ with a formula with a free occurrence of a variable bound by the replacement of $a$:

\[ p \rightarrow [a]p \quad x \geq 0 \rightarrow [x' = -1]x \geq 0 \quad \{ a \mapsto x' = -1, p \mapsto x \geq 0 \} \]

Additional free variables are acceptable in replacements for $p$ as long as they are not bound in the particular context into which they will be substituted:

\[ p \rightarrow [a]p \quad y \geq 0 \rightarrow [x' = -1]y \geq 0 \quad \{ a \mapsto x' = -1, p \mapsto y \geq 0 \} \]

Complex formulas are acceptable as replacements for $p$ if their free variables are not bound in the context, e.g., using $\sigma = \{ a \mapsto x' := 5x, p \mapsto [x' = x^2 - 2x + 2] x \geq 1 \}$:

\[ p \rightarrow [a]p \]

\[ \mathbb{U} \quad [x' = x^2 - 2x + 2]x \geq 1 \rightarrow [x' := 5x] [x' = x^2 - 2x + 2]x \geq 1 \]

But it is soundness-critical that US clashes when substituting a formula with a free dependence on $x'$ for $p$ into a context where $x'$ will be bound after the substitution:

\[ p \rightarrow [a]p \quad (x-1)' \geq 0 \rightarrow [x' := 5x] (x-1)' \geq 0 \quad \{ a \mapsto x' := 5x, p \mapsto (x-1)' \geq 0 \} \]

Gödel's generalization rule $G$ uses $p(\bar{x})$ instead of $p$ from $\mathbb{V}$ so its USR instance allows all variables $\bar{x}$ to occur in the replacement without causing a clash:

\[ G_{\text{USR}} \quad (x-1)' \geq 0 \rightarrow (x-1)' \geq 0 \quad \{ a \mapsto x' = -1, p(\bar{x}) \mapsto (x-1)' \geq 0 \} \]

Intuitively, the argument $\bar{x}$ in this uniform substitution instance of $G$ was not introduced as part of the substitution but merely put in for the placeholder $\cdot$ instead. Let $\bar{x} = (x,y), \{ a \mapsto x := x+1, b \mapsto x := 0; y' = -2, p(\bar{x}) \mapsto x \geq y \}, US$ derives:

\[ US \quad [a \cup b] p(\bar{x}) \leftrightarrow [a] p(\bar{x}) \wedge [b] p(\bar{x}) \]

\[ [x := x + 1 \cup (x := 0; y' = -2)] x \geq y \leftrightarrow [x := x + 1] x \geq 0 \wedge [x := 0; y' = -2] x \geq y \]
With \( \vec{x} = (x, y) \) and \( \{ a \mapsto x := x + 1 \cup y := 0, b \mapsto y' = -1, p(\vec{x}) \mapsto x \geq y \} \), \textbf{US} derives:

\[
\text{US} \vdash [a; b] p(\vec{x}) \leftrightarrow [a][b] p(\vec{x}) \quad \text{(4)}
\]

\[
(x := x + 1 \cup y := 0); y' = -1|x \geq y \leftrightarrow [x := x + 1 \cup y := 0][y' = -1|x \geq y
\]

It is soundness-critical that \textbf{US} clashes when substituting free variables into quantifier symbols, because their semantics and their substitutes may query the argument at any state, e.g., with \( \sigma = \{ p \mapsto x > 0, q \mapsto x > 1, C(\cdot) \mapsto [x' = 1]_{\cdot} \} \):

\[
(p \leftrightarrow q) \rightarrow (C(p) \leftrightarrow C(q))
\]

Not all axioms fit to the uniform substitution framework, though. The Barcan axiom was used in a completeness proof for the Hilbert-type calculus for differential dynamic logic [7] (but not in the completeness proof for its sequent calculus [5]):

\[
(\forall x [\alpha] p(x) \rightarrow [\alpha] \forall x p(x) \quad (x \notin \alpha)
\]

\textbf{B} is unsound without the restriction \( x \notin \alpha \), though, so that the following would be an unsound axiom:

\[
\forall x [a] p(x) \rightarrow [a] \forall x p(x)
\]

because \( x \notin a \) cannot be enforced for program constants, since their effect might very well depend on the value of \( x \) or since they might write to \( x \). In (4), \( x \) cannot be written by \( a \) without violating soundness:

\[
\forall x [a] p(x) \rightarrow [a] \forall x p(x)
\]

\[
\forall x [x := 0] x \geq 0 \rightarrow [x := 0] \forall x (x \geq 0)
\]

\{ \( a \mapsto x := 0, p(\cdot) \mapsto \cdot \geq 0 \) \}

nor can \( x \) be read by \( a \) in (4) without violating soundness:

\[
\forall x [a] p(x) \rightarrow [a] \forall x p(x)
\]

\[
\forall x [?(y = x^2)] y = x^2 \rightarrow [?(y = x^2)] xy = x^2
\]

\{ \( a \mapsto ?(y = x^2), p(\cdot) \mapsto y = -2 \) \}

Thus, the completeness proof for differential dynamic logic from prior work [7] does not carry over. A more general completeness result for differential game logic [9] implies, however, that \textbf{B} is unnecessary for completeness.

## 5 Differential Equations and Differential Axioms

Section 4 leverages the first-order features of d\(\mathcal{L} \) and proof rule \textbf{US} to obtain a finite list of axioms without side-conditions. They lack axioms for differential equations, though. Classical calculi for d\(\mathcal{L} \) have axioms for replacing differential equations with a quantifier for time \( t \geq 0 \) and an assignment for their solutions \( \vec{x}(t) \) [5, 7]:

\[
([\phi]) \quad [x' = \theta] \phi \leftrightarrow \forall t \geq 0[x := \vec{x}(t)] \phi \quad (\vec{x}'(t) = \theta)
\]
Figure 3 shows axioms for proving properties of differential equations (DW–DS) as well as axioms for invariants, cuts, effects, and ghosts.

**Differentials: Invariants, Cuts, Effects, and Ghosts**

The axioms can prove properties of more general “unsolvable” differential equations. They can also obtain a logically internalized version of differential invariants and related proof rules for differentials.

This section leverages US and the new differential forms in $\mathcal{DL}$ to obtain a logically internalized version of differential invariants and related proof rules for differential equations $\mathcal{G}$ as axioms (without schema variables and without side-conditions). These axioms can prove properties of more general “unsolvable” differential equations. They can also prove all properties of differential equations that can be proved with solutions $\mathcal{G}$ while guaranteeing correctness of the solution as part of the proof.

### 5.1 Differentials: Invariants, Cuts, Effects, and Ghosts

Figure 3 shows axioms for proving properties of differential equations (DW–DS) as well as axioms for differential substitutions ($[\cdot := \cdot]$), and differential axioms for differentials ($\cdot'$). Axiom $\checkmark'$ identifying $(x)' = x'$ for variables $x \in \mathcal{V}'$ and axiom $\checkmark'$ for functions $f$ and number literals of arity 0 are used implicitly to save space. Some axioms use reverse implications $\phi \leftarrow \psi$ instead of $\psi \rightarrow \phi$ for emphasis.

**Differential weakening** axiom $\mathcal{DW}$ internalizes that differential equations can never leave their evolution domain $q(x)$. After all evolutions along $\phi' = f(x) & q(x)$ will the evolution domain con-
A. Platzer  
A Complete Uniform Substitution Calculus for Differential Dynamic Logic

straint \( q(x) \) hold, because it holds all along. [\text{DW}] implies\(^2\)

\[
[x' = f(x) \& q(x)] p(x) \iff [x' = f(x) \& q(x)] (q(x) \rightarrow p(x))
\]

also called [\text{DW}], which allows to export the evolution domain constraint to the postcondition. Its (right) assumption is best proved by [\text{G}] yielding premise \( q(x) \rightarrow p(x) \). The differential cut axiom [\text{DC}] is a cut for differential equations. It internalizes that differential equations always staying in \( r(x) \) always stay in \( p(x) \) iff \( p(x) \) always holds after the differential equation that is restricted to the smaller evolution domain & \( q(x) \land r(x) \). [\text{DC}] is a differential variant of modal modus ponens [\text{K}].

Differential effect axiom [\text{DE}] internalizes that the effect on differential symbols along a differential equation is a differential assignment assigning the right-hand side \( f(x) \) to the left-hand side \( x' \). The differential assignment \( x' := f(x) \) in [\text{DE}] mimics instantaneously the (continuous) effect that the differential equation \( x' = f(x) \& q(x) \) has on \( x' \), thereby selecting the appropriate vector field for subsequent differentials. Axiom [\text{DI}] internalizes differential invariants [\cite{Platzer2017}], i.e. that a differential equation stays in \( p(x) \) if it starts in \( p(x) \) and if its differential \( (p(x))' \) always holds after the differential equation \( x' = f(x) \& q(x) \). The differential equation also vacuously stays in \( p(x) \) if it starts outside \( q(x) \), since it is stuck then. The (right) assumption of [\text{DI}] is best proved by [\text{DE}] to select the appropriate vector field \( x' = f(x) \) for the differential \( (p(x))' \) and a subsequent [\text{G}] to make the evolution domain constraint \( q(x) \) available as an assumption. For simplicity, this article focuses on atomic postconditions for which \( (\theta \geq \eta)' \equiv (\theta > \eta)' \equiv (\theta)' \geq (\eta)' \) and \( (\theta = \eta)' \equiv (\theta \neq \eta)' \equiv (\eta)' = (\eta)' \), etc. Axiom [\text{DG}] internalizes differential ghosts [\cite{Platzer2017}], i.e. that additional differential equations can be added if their solution exists long enough, which can enable new invariants that are not otherwise provable [\cite{Platzer2017}]. Axiom [\text{DS}] solves constant differential equations, and more complex differential equations with the help of [\text{DGDC}].

Vectorial generalizations to systems of differential equations are possible for the axioms in Fig. [\ref{fig:axioms}].

Axiom \([\eta :=] \) is the differential assignment analogue of \( \equiv := \). The differential axioms for differentials \( +' y', c' y', \&' y', y' \) axiomatize differentials of polynomials. They are related to corresponding rules for time-derivatives, except that those would be ill-defined in a local state so it is crucial to work with differentials that have a local semantics in individual states. Note that uniform substitutions correctly maintain that \( y \) does not occur in replacements for \( a(x), b(x) \) for [\text{DGDC}] and \( x \) does not occur in replacements for \( f \) in [\text{DS}], which are both soundness-critical, but that occurrences of \( x \) in replacements for \( f \) are acceptable in \([\eta :=] \).

5.2 Example Proofs

This section illustrates how the uniform substitution calculus for \( \mathcal{DL} \) can be used to achieve a number of different reasoning techniques from the same proof primitives. While the same flexibility enables these different techniques also for proofs of hybrid systems, the following examples focus on differential equations to illustrate how the axioms in Fig. [\ref{fig:axioms}] are meant to be used.

\(^2\)\([x' = f(x) \& q(x)] (q(x) \rightarrow p(x)) \rightarrow [x' = f(x) \& q(x)] p(x) \) derives by [\text{K}] from [\text{DW}] The converse implication \([x' = f(x) \& q(x)] p(x) \rightarrow [x' = f(x) \& q(x)] (q(x) \rightarrow p(x)) \) derives by [\text{K}] since [\text{G}] derives \([x' = f(x) \& q(x)] (p(x) \rightarrow (q(x) \rightarrow p(x))) \).
Example 4 (Contextual equivalence proof). The following proof proves a property of a differential equation using differential invariants without having to solve that differential equation. One use of rule \texttt{US} is shown explicitly, other uses of \texttt{US} are similar to obtain and use the other axiom instances. \texttt{CE} is used together with \texttt{MP}.

Previous calculi \cite{6, 8} collapse this proof into a single proof step but with complicated built-in operator implementations that silently perform the same reasoning yet in an intransparent way. The approach presented here combines separate axioms to achieve the same effect in a modular way, because they have individual responsibilities internalizing separate logical reasoning principles in \textit{differential-form} \(d\mathbb{R}\). Tactics combining the axioms as indicated make the axiomatic way equally convenient. Clever cuts or \texttt{MP} uses enable proofs in which the main argument remains as fast \cite{6, 8} while the additional premises subsequently check soundness.

Example 5 (Flat proof). Rules \texttt{CQ, CE} simplify the proof in Example 4 substantially but are not necessary because a proof without contextual equivalence is possible:
Example 6 (Parametric proof). The proofs in Example 4 and 5 use (implicit) cuts with equivalences that predict the outcome of the right premise, which is conceptually simple but inconvenient for proof search. More constructively, a direct proof can use a free function symbol \( j(x,x') \) to obtain a straightforward parametric proof, instead:

\[
\begin{align*}
\text{CQ} & : (x \times x')' = j(x,x') \\
\text{CE} & : [x' := x^3] j(x,x') \geq 0 \\
\text{DE} & : [x' := x^3] (x \times x \geq 1)' \\
\text{DI} & : x \times x \geq 1 \rightarrow [x' := x^3] x \times x \geq 1
\end{align*}
\]

After conducting this proof with two open premises, the free function symbol \( j(x,x') \) can be instantiated as needed by a uniform substitution (USR from Theorem 12):

\[
\begin{align*}
\text{USR} & : x^3 \times x + x \times x^3 \geq 0 \\
\text{DI} & : x \times x \geq 1 \rightarrow [x' := x^3] x \times x \geq 1
\end{align*}
\]

The same technique is helpful for invariant search, in which case a free predicate symbol \( p(\bar{x}) \) is used and instantiated by rule [USR] lazily once the conditions it needs to fulfill to close the proof become clear.

Example 7 (Forward computation proof). The proof in Example 6 involves less search than the proofs of the same formula in Example 4 and 5. But it still ultimately requires foresight to identify the appropriate instantiation of \( j(x,x') \) for which the proof closes. For (differential) invariant search, such proof search is essentially unavoidable \( \Box \) even if the technique in Example 6 maximally postpones the search.

When applied from left to right, the differential axioms \( C^f \times x + x = f(x) \) however, perform a deterministic computation that always simplifies the terms by pushing differential operators inside along a well-founded order. For example, all proof search in the right branch of the last proof of Example 6 can be replaced by a deterministic forward computation proof starting from reflexivity and drawing on axiom instances as needed in a forward proof:

\[
\begin{align*}
\text{MP} & : (x \times x)' = (x \times x)' \\
\text{CT} & : (x \times x)' = (x)' \times x + x \times (x)' \\
\text{DI} & : (x \times x)' = x' \times x + x \times x'
\end{align*}
\]

Efficient proof search combines this forward computation proof technique with the backward proof approach from Example 6. Even the remaining positions where axioms still match can be pre-computed as a simple function of the axiom that has been applied, e.g., from its fixed pattern of occurrences of differential operators.
Example 8 (Axiomatic differential equation solver). Axiomatic equivalence proofs for solving differential equations involve $\text{DG}$ for introducing a time variable, $\text{DC}$ to cut the solutions in, $\text{DW}$ to export the solution to the postcondition, inverse $\text{DC}$ to remove the evolution domain constraints again, inverse $\text{DG}$ to remove the original differential equations, and finally $\text{DS}$ to solve the differential equation for time:

\[
* \\
\phi \rightarrow \forall s \geq 0 (x_0 + \frac{a}{2} s^2 + v_0 s \geq 0) \\
\rightarrow \phi \rightarrow \forall s \geq 0 [t := 0 + 1s] x_0 + \frac{a}{2} t^2 + v_0 t \geq 0 \\
\rightarrow \phi \rightarrow [t' = 1] x_0 + \frac{a}{2} t^2 + v_0 t \geq 0 \\
\rightarrow \phi \rightarrow [v' = a, t' = 1] x_0 + \frac{a}{2} t^2 + v_0 t \geq 0 \\
\rightarrow \phi \rightarrow [x' = v, v' = a, t' = 1] x_0 + \frac{a}{2} t^2 + v_0 t \geq 0 \\
\rightarrow \phi \rightarrow [x' = v, v' = a, t' = 1 \& v = v_0 + at] x_0 + \frac{a}{2} t^2 + v_0 t \geq 0 \\
\rightarrow \phi \rightarrow [x' = v, v' = a, t' = 1 \& v = v_0 + at \& x = x_0 + \frac{a}{2} t^2 + v_0 t \rightarrow x \geq 0] \\
\rightarrow \phi \rightarrow [x' = v, v' = a, t' = 1 \& v = v_0 + at \& x = x_0 + \frac{a}{2} t^2 + v_0 t, x \geq 0] \\
\rightarrow \phi \rightarrow [x' = v, v' = a, t' = 1 \& v = v_0 + at, x \geq 0] \\
\rightarrow \phi \rightarrow [x' = v, v' = a] x \geq 0 \\
\rightarrow \phi \rightarrow [x' = v, v' = a] x \geq 0
\]

where $\phi$ is $a \geq 0 \& v = v_0 \geq 0 \& x = x_0 \geq 0$. The existential quantifier for $t$ is instantiated by 0, leading to $[t := 0]$ (suppressed in the proof for readability reasons). The 4 uses of $\text{DC}$ lead to 2 additional premises proving that $v = v_0 + at$ and then $x = x_0 + \frac{a}{2} t^2 + v_0 t$ are differential invariants (using $\text{DI}\text{DE}\text{DW}$). Shortcuts using only $\text{DW}$ instead are possible. But the elaborate proof above generalizes to $\langle \rangle$ because it is an equivalence proof. The additional premises for $\text{DC}$ with $v = v_0 + at$ prove as follows:

\[
* \\
\rightarrow \phi \rightarrow \forall s \geq 0 (x_0 + \frac{a}{2} s^2 + v_0 s \geq 0) \\
\rightarrow \phi \rightarrow [t' = 0 + 1s] x_0 + \frac{a}{2} t^2 + v_0 t \geq 0 \\
\rightarrow \phi \rightarrow [v' = a, t' = 1] x_0 + \frac{a}{2} t^2 + v_0 t \geq 0 \\
\rightarrow \phi \rightarrow [x' = v, v' = a, t' = 1] x_0 + \frac{a}{2} t^2 + v_0 t \geq 0 \\
\rightarrow \phi \rightarrow [x' = v, v' = a, t' = 1 \& v = v_0 + at] x_0 + \frac{a}{2} t^2 + v_0 t \geq 0 \\
\rightarrow \phi \rightarrow [x' = v, v' = a, t' = 1 \& v = v_0 + at \& x = x_0 + \frac{a}{2} t^2 + v_0 t \rightarrow x \geq 0] \\
\rightarrow \phi \rightarrow [x' = v, v' = a, t' = 1 \& v = v_0 + at \& x = x_0 + \frac{a}{2} t^2 + v_0 t, x \geq 0] \\
\rightarrow \phi \rightarrow [x' = v, v' = a, t' = 1 \& v = v_0 + at, x \geq 0] \\
\rightarrow \phi \rightarrow [x' = v, v' = a] x \geq 0 \\
\rightarrow \phi \rightarrow [x' = v, v' = a] x \geq 0
\]
After that, the additional premises for $\mathbf{DC}$ with $x = x_0 + \frac{a}{2}t^2 + v_0t$ prove as follows:

5.3 Differential Substitution Lemmas

The key insight for the soundness of $\mathbf{DI}$ axiom is that the analytic time-derivative of the value of a term $\eta$ along a differential equation $x' = \theta \land \psi$ agrees with the values of its differential $(\eta)'$ along the particular vector field of that differential equation.

**Lemma 13** (Differential). If $I, \varphi \models x' = \theta \land \psi$ holds for some flow $\varphi: [0, r] \rightarrow \mathcal{S}$ of any duration $r > 0$, then for all $0 \leq \zeta \leq r$:

$$[(\eta)']I\varphi(\zeta) = \frac{d[\eta]I\varphi(t)}{dt}(\zeta)$$

**Proof.** By chain rule [15, §3.10]:

$$\frac{d[\eta]I\varphi(t)}{dt}(\zeta) = ([\eta]I \circ \varphi)'(\zeta) = (\nabla[\eta]I)(\varphi(\zeta)) \cdot \varphi'(\zeta) = \sum_x \frac{\partial[\eta]I}{\partial x}(\varphi(\zeta)) \partial_x(\varphi'(\zeta))$$

where $(\nabla[\eta]I)(\varphi(\zeta))$, the spatial gradient $\nabla[\eta]I$ at $\varphi(\zeta)$, is the vector of $\frac{\partial[\eta]I}{\partial x}(\varphi(\zeta)) = \frac{\partial[\eta]I\varphi(\zeta)}{\partial x}$.

Chain rule and Def. 4 and Def. 6 thus, imply:

$$[(\eta)']I\varphi(\zeta) = \sum_x \varphi(\zeta)(x') \frac{\partial[\eta]I\varphi(t)}{\partial X}(\zeta) = \sum_x \frac{\partial[\eta]I\varphi(\zeta)}{\partial X} \frac{\partial\varphi(t)}{\partial x}(x) \cdot \frac{d[\eta]I\varphi(t)}{dt}(\zeta)$$

The key insight for the soundness of differential effects axiom $\mathbf{DE}$ is that differential assignments mimicking the differential equation are vacuous along that differential equation. The differential substitution resulting from a subsequent use of $[\cdot := \cdot]$ axiom is crucial to relay the values of the time-derivatives of the state variables $x$ along a differential equation by way of their corresponding differential symbol $x'$. In combination, this makes it possible to soundly substitute the right-hand side of a differential equation for its left-hand side in a proof.

**Lemma 14** (Differential assignment). If $I, \varphi \models x' = \theta \land \psi$ for a flow $\varphi: [0, r] \rightarrow \mathcal{S}$ of any duration $r \geq 0$, then

$I, \varphi \models \phi \leftrightarrow [x' := \theta]\phi$
Lemma 15 (Derivations). The following equations of differentials are valid:

\[
(f)' = 0 \quad \text{for arity 0 functions/numbers } f \tag{5}
\]

\[
(x)' = x' \quad \text{for variables } x \in \mathcal{V} \tag{6}
\]

\[
(\theta + \eta)' = (\theta)' + (\eta)' \tag{7}
\]

\[
(\theta \cdot \eta)' = (\theta)' \cdot \eta + \theta \cdot (\eta)' \tag{8}
\]

\[
[y := \theta][y' := 1][(f(\theta))' = (f(y))' \cdot (\theta)'] \quad \text{for } y, y' \not\in \theta \tag{9}
\]

Proof. The proof shows each equation separately. The first parts consider any constant function (i.e., arity 0) or number literal \( f \) for (5) and align the differential \((x)'\) of a term that happens to be a variable \( x \in \mathcal{V} \) with its corresponding differential symbol \( x' \in \mathcal{V}' \) for (6). The other cases exploit linearity for (7) and Leibniz properties of partial derivatives for (8). Case (9) exploits the chain rule and assignments and differential assignments for the fresh \( y, y' \) to mimic partial derivatives. Equation (9) generalizes to functions \( f \) of arity \( n > 1 \), in which case \( \cdot \) is the (definable) Euclidean scalar product.

\[
[(f)']Iv = \sum_x v(x') \frac{\partial [f]Iv^X_x}{\partial X} = \sum_x v(x') \frac{\partial f}{\partial X} = 0 \tag{5}
\]

\[
[(x)']Iv = \sum_y v(y') \frac{\partial [x]Iv^X_y}{\partial X} = v(x') \frac{\partial x}{\partial X} = v(x') = [x']Iv \tag{6}
\]

\[
[(\theta + \eta)']Iv = \sum_x v(x') \frac{\partial [\theta + \eta]Iv^X_x}{\partial X} = \sum_x v(x') \frac{\partial \theta}{\partial X} + \frac{\partial \eta}{\partial X} = \sum_x v(x') \frac{\partial \theta}{\partial X} + \sum_x v(x') \frac{\partial \eta}{\partial X} = [\theta]'Iv + [\eta]'Iv \tag{7}
\]

\[
[(\theta \cdot \eta)']Iv = \sum_x v(x') \frac{\partial [\theta \cdot \eta]Iv^X_x}{\partial X} = \sum_x v(x') \frac{\partial \theta}{\partial X} \cdot \eta + \theta \cdot \frac{\partial \eta}{\partial X} = \sum_x v(x') \frac{\partial \theta}{\partial X} \cdot \eta + \theta \cdot \frac{\partial \eta}{\partial X} = [\theta]'Iv \cdot [\eta]'Iv \tag{8}
\]
The uniform substitution calculus for differential-form \(\mathcal{L}^x\) calculus is **sound**, i.e. all formulas that it proves from valid premises are valid. The soundness argument is entirely modular. The
concrete $dL$ axioms in Fig. 2, 3 are valid and the axiomatic proof rules (i.e. pairs of formulas) in Fig. 2 are locally sound (Theorem 16), which implies soundness. The uniform substitution rule is sound so only concludes valid formulas from valid premises (Theorem 11), which implies that $dL$ axioms (and other provable $dL$ formulas) can only be instantiated soundly by rule US. Uniform substitution instances of locally sound inferences are locally sound (Theorem 12), which implies that $dL$ axiomatic proof rules in Fig. 2 can only be instantiated soundly by uniform substitutions.

**Theorem 16 (Soundness).** The uniform substitution calculus for $dL$ is sound, that is, every formula that is provable by the $dL$ axioms and proof rules is valid, i.e. true in all states of all interpretations. The axioms in Fig. 2 and 3 are valid formulas and the axiomatic proof rules in Fig. 2 are locally sound.

**Proof.** The axioms (and most proof rules) in Fig. 2 are special instances of corresponding axiom schemata and proof rules for differential dynamic logic [7] and, thus, sound. All proof rules in Fig. 2 (but not US itself) are even locally sound, which implies soundness, i.e. that their conclusions are valid (in all $I$) if their premises are. Note that rules $\forall$MP can be augmented soundly to use $p(\dot{x})$ instead of $p(x)$ or $p$, respectively, such that the $FV(\sigma) = \emptyset$ requirement of Theorem 12 will be met during US instances of all axiomatic proof rules.

**DW Soundness of DW** uses that differential equations can never leave their evolution domain by Def. 6. To show $\nu \in [[x' = f(x) & q(x)]q(x)]I$, consider any $\phi$ of any duration $r \geq 0$ solving $I, \phi \models x' = f(x) \land q(x)$. Then $I, \phi \models q(x)$ hence $\phi(r) \in [[q(x)]I$.

**DC Soundness of DC** is a stronger version of soundness for the differential cut rule [6]. DC is a differential version of the modal modus ponens K. The core is that if $r(x)$ always holds after the differential equation and $p(x)$ always holds after the differential equation $x' = f(x) \land q(x)$, then $p(x)$ always holds after the differential equation $x' = f(x) \land q(x)$.

Let $\nu \in [[x' = f(x) \land q(x)]r(x)]I$. Since all restrictions of solutions are this, this is equivalent to $I, \phi \models r(x)$ for all $\phi$ of any duration solving $I, \phi \models x' = f(x) \land q(x)$ and starting in $\phi(0) = \nu$ on $\{x'\}^C$. Consequently, for all $\phi$ starting in $\phi(0) = \nu$ on $\{x'\}^C$, $I, \phi \models x' = f(x) \land q(x)$ is equivalent to $I, \phi \models x' = f(x) \land q(x) \land r(x)$. Hence, $\nu \in [[x' = f(x) \land q(x) \land r(x)]p(x)]I$ is equivalent to $\nu \in [[x' = f(x) \& q(x)]p(x)]I$.

**DE Soundness of DE** is genuine to differential-form $dL$ leveraging Lemma 14. Consider any state $\nu$. Then $\nu \in [[x' = f(x) \land q(x)]p(x, x')]I$ iff $\phi(r) \in [[p(x, x')]I$ for all solutions $\phi : [0, r] \rightarrow \mathcal{S}$ of $I, \phi \models x' = f(x) \land q(x)$ of any duration $r$ starting in $\phi(0) = \nu$ on $\{x'\}^C$. That is equivalent to: for all $\phi$, if $I, \phi \models x' = f(x) \land q(x)$ then $I, \phi \models p(x, x')$. By Lemma 14, $I, \phi \models p(x, x')$ iff $I, \phi \models [x' := f(x)]p(x, x')$, so that is equivalent to $\phi(r) \in [[x' := f(x)]p(x, x')]I$ for all solutions $\phi : [0, r] \rightarrow \mathcal{S}$ of $I, \phi \models x' = f(x) \land q(x)$ of any duration $r$ starting in $\phi(0) = \nu$ on $\{x'\}^C$, which is, consequently, equivalent to $\nu \in [[x' = f(x) \land q(x)]x' := f(x)]p(x, x')]I$.

**DI Soundness of DI** has some relation to the soundness proof for differential invariants [6], yet is generalized to leverage differentials. The proof is only shown for $p(x) \equiv g(x) \geq 0$, in which case $(p(x))' \equiv ((g(x))' \geq 0)$, because the variation for other formulas then is as in previous

---

3 The uniform substitution proof calculus improves modularity and gives stronger equivalence formulations of axioms, though.
Consider a state $v$ in which $v \in \llbracket q(x) \rightarrow (p(x) \land [x' = f(x) \& q(x)](p(x))') \rrbracket I$. If $v \not\in \llbracket q(x) \rrbracket'$, there is nothing to show, because there is no solution of $x' = f(x) \& q(x)$ for any duration, so the consequence holds vacuously. Otherwise, $v \in \llbracket q(x) \rrbracket I$, which implies, by assumption, that $v \in \llbracket p(x) \land [x' = f(x) \& q(x)](p(x))') \rrbracket I$. To show that $v \in \llbracket [x' = f(x) \& q(x)](p(x)) \rrbracket I$ consider any solution $\varphi$ of any duration $r \geq 0$. The case $r = 0$ follows from $v \in \llbracket p(x) \rrbracket I$ by Lemma 4, since $\text{FV}(p(x)) = \{x\}$ is disjoint from $\{x'\}$, which, unlike $x$, is changed by evolutions of any duration, including 0. That leaves the case $r > 0$.

Let $\varphi$ be a solution of $x' = f(x) \& q(x)$ according to Def. 6, so $v = \varphi(0)$ on $\{x'\}^C$ and $I, \varphi \models x' = f(x) \& q(x)$. By $v \in \llbracket [x' = f(x) \& q(x)](p(x))' \rrbracket I$, this implies $I, \varphi \models (p(x))'$. Since $r > 0$, Lemma 13 implies $0 \leq \llbracket (g(x))' \rrbracket I\varphi(\zeta) = \frac{dI\varphi(t)}{dt}(\zeta)$ for all $\zeta$. Together with $\varphi(0) \in \llbracket p(x) \rrbracket I$ (by Lemma 4 and $\text{FV}(p(x)) \cap \{x'\} = \emptyset$ from $v \in \llbracket p(x) \rrbracket I$, which is $\varphi(0) \in \llbracket g(x) \geq 0 \rrbracket I$, this implies $\varphi(\zeta) \in \llbracket g(x) \geq 0 \rrbracket I$ for all $\zeta$, including $r$, by the mean-value theorem since the value $\llbracket g(x) \rrbracket I\varphi(t)$ of term $g(x)$ along $\varphi$ is continuous in $t$ on $[0, r]$ and differentiable on $(0, r)$ as compositions of the, by Def. 4 smooth, evaluation function and the differentiable solution $\varphi(t)$ of a differential equation.

Let $v \in \llbracket \exists y \left[ x' = f(x), y' = a(x)y + b(x) \& q(x) \right](p(x)) \rrbracket I$, that is, $v \in \llbracket [x' = f(x), y' = a(x)y + b(x) \& q(x)](p(x)) \rrbracket I$ for some value $d \in \mathbb{R}$. In order to show that $v \in \llbracket [x' = f(x) \& q(x)](p(x))' \rrbracket I$, consider any $\varphi : [0, r] \rightarrow \mathcal{S}$ such that $I, \varphi \models x' = f(x) \& q(x)$ and $\varphi(0) = v$ on $\{x'\}^C$. By modifying the values of $y, y'$ along $\varphi$, this function can be augmented to a solution $\bar{\varphi} : [0, r] \rightarrow \mathcal{S}$ such that $I, \bar{\varphi} \models x' = f(x) \& y' = a(x)y + b(x) \& q(x)$ and $\bar{\varphi}(0)(y) = d$ as shown below. The assumption then implies $\bar{\varphi}(r) \in \llbracket p(x) \rrbracket I$, which, by Lemma 4, is equivalent to $\varphi(r) \in \llbracket p(x) \rrbracket I$ since $y, y' \not\in \text{FV}(p(x))$ and $\varphi(r) = \bar{\varphi}(r)$ on $\{y, y'\}^C$, which implies $v \in \llbracket [x' = f(x) \& q(x)](p(x)) \rrbracket I$, since $\varphi$ was arbitrary.

The construction of the modification $\bar{\varphi}$ of $\varphi$ on $\{y, y'\}$ proceeds as follows. By Picard-Lindelöf [16 §10.VII], there is a solution $y : [0, r] \rightarrow \mathbb{R}$ of the initial-value problem

$$y(0) = d$$

$$y'(t) = F(t, y(t)) \overset{\text{def}}{=} y(t)[a(x)]I\varphi(t) + [b(x)]I\varphi(t)$$

because $F(t, y)$ is continuous on $[0, r] \times \mathbb{R}$ (since $[a(x)]I\varphi(t)$ and $[b(x)]I\varphi(t)$ are continuous in $t$ as compositions of the, by Def. 4 smooth, evaluation function and the continuous solution $\varphi(t)$ of a differential equation) and because $F(t, y)$ satisfies the Lipschitz condition

$$\|F(t, y) - F(t, z)\| = \|y - z\|[a(x)]I\varphi(t) \leq \|y - z\| \max_{t \in [0, r]} \|a(x)\|I\varphi(t)$$

where the maximum exists, because it is a maximum of a continuous function on the compact set $[0, r]$. The modification $\bar{\varphi}$ agrees with $\varphi$ on $\{y, y'\}^C$. On $\{y, y'\}$, the modification $\bar{\varphi}$ is defined as $\bar{\varphi}(t)(y) = y(t)$ and $\bar{\varphi}(t)(y') = F(t, y(t))$, respectively, for the solution $y(t)$ of (10), in particular $\bar{\varphi}(t)(y')$ agrees with the time-derivative $y'(t)$ of the value $\bar{\varphi}(t)(y) = y(t)$ of $y$ along $\bar{\varphi}$. By construction, $\bar{\varphi}(0)(y) = d$ and $I, \bar{\varphi} \models x' = f(x) \& y' = a(x)y + b(x) \& q(x)$, because $\varphi(t) = \bar{\varphi}(t)$ on $\{y, y'\}^C$ so that $x' = f(x) \& q(x)$ continues to hold along $\bar{\varphi}$ by Lemma 2 because $y, y' \not\in \text{FV}(x' = f(x) \& q(x))$, and because $y' = a(x)y + b(x)$ holds along $\bar{\varphi}$ by (10).
Conversely, let \( v \in \llbracket x' = f(x) \& q(x) \rrbracket p(x) \rrbracket I \). This direction shows a stronger version of \( v \in \llbracket \forall y \, x' = f(x), y' = a(x) \& b(x) \& q(x) \rrbracket p(x) \rrbracket I \) by showing for all terms \( \eta \) that \( v \in \llbracket \forall y \, x' = f(x), y' = \eta \& q(x) \rrbracket p(x) \rrbracket I \). Consider any \( d \in \mathbb{R} \) and term \( \eta \) and show \( v^d \in \llbracket x' = f(x), y' = \eta \& q(x) \rrbracket p(x) \rrbracket I \). Consider any \( \varphi : [0, r] \to \mathcal{S} \) such that \( I, \varphi \models x' = f(x) \land y' = \eta \land q(x) \) with \( \varphi(0) = v^d \) on \( \{x', y'\}^C \). Then the restriction \( \varphi|_{\{x', y'\}^C} \) of \( \varphi \) to \( \{x', y'\}^C \) with \( \varphi|_{\{x', y'\}^C}(t) = v^d \) on \( \{x', y'\} \) for all \( t \in [0, r] \) still solves \( I, \varphi|_{\{x', y'\}^C} \models x' = f(x) \land q(x) \) by Lemma 2 since \( \varphi|_{\{x', y'\}^C} = \varphi \) on \( \{x', y'\}^C \) and \( y, y' \notin \text{FV}(x' = f(x) \land q(x)) \). It also satisfies \( \varphi|_{\{x', y'\}^C}(0) = v^d \) on \( \{x'\}^C \), because \( \varphi(0) = v^d \) on \( \{x', y'\}^C \) yet \( \varphi|_{\{x', y'\}^C}(y') = v^d(y') \). Thus, by assumption, \( \varphi|_{\{x', y'\}^C}(r) \in \llbracket p(x) \rrbracket I \), which implies \( \varphi(r) \in \llbracket p(x) \rrbracket I \) by Lemma 4 because \( \varphi = \varphi|_{\{x', y'\}^C} \) on \( \{y', y'\}^C \) and \( y, y' \notin \text{FV}(p(x)) \).

**DS** Soundness of the solution axiom \( \text{DS} \) follows from existence and uniqueness of global solutions of constant differential equations. Consider any state \( v \). There is a unique 16\,§10.VII global solution \( \varphi : [0, \infty) \to \mathcal{S} \) defined as \( \varphi(\xi)(x) \equiv [x + ft]\llbracket v^\xi \rrbracket \) and \( \varphi(\xi)(x') \equiv \frac{df(\xi)(x)}{d\xi}(\xi) = f(0) \) and \( \varphi(\xi)(v) \equiv v \) on \( \{x', v\}^C \). This solution satisfies \( \varphi(0) = v(x) \) on \( \{x'\}^C \) and \( I, \varphi \models x' = f \), i.e. \( \varphi(\xi) \in \llbracket x' = f \rrbracket I \) for all \( 0 \leq \xi \leq r \). All solutions of \( x' = f \) from initial state \( v \) are restrictions of \( \varphi \) to subintervals of \( [0, \infty) \). The (unique) state \( \omega \) that satisfies \( (v^\xi, \omega) \in \llbracket x := x + ft \rrbracket I \) agrees with \( \omega = \varphi(\xi) \) on \( \{x'\}^C \), so that, by \( x' \notin \text{FV}(p(x)) \), Lemma 4 implies that \( \omega \in \llbracket p(x) \rrbracket I \) iff \( \varphi(\xi) \in \llbracket p(x) \rrbracket I \).

First consider axiom \( [x' = f]p(x) \leftrightarrow \forall t \geq 0 [x := x + ft]p(x) \) for the special case \( q(x) \equiv \text{true} \). If \( v \in \llbracket x' = f \rrbracket p(x) \rrbracket I \), then \( \varphi(\xi) \in \llbracket p(x) \rrbracket I \) for all \( \xi \geq 0 \), because the restriction of \( \varphi \) to \( [0, \xi] \) solves \( x' = f \) from \( v \), thus \( \omega \in \llbracket p(x) \rrbracket I \), which implies \( v^\xi \in \llbracket [x := x + ft]p(x) \rrbracket I \), so \( v \in \llbracket \forall t \geq 0 [x := x + ft]p(x) \rrbracket I \) as \( \xi \geq 0 \) was arbitrary. Conversely, \( v \in \llbracket \forall t \geq 0 [x := x + ft]p(x) \rrbracket I \) implies \( v^\xi \in \llbracket [x := x + ft]p(x) \rrbracket I \) for all \( \xi \geq 0 \), i.e. \( \omega \in \llbracket p(x) \rrbracket I \) when \( (v^\xi, \omega) \in \llbracket x := x + ft \rrbracket I \). Lemma 4 again implies \( \varphi(\xi) \in \llbracket p(x) \rrbracket I \) for all \( \xi \geq 0 \), so \( v \in \llbracket [x' = f]p(x) \rrbracket I \), since all solutions are restrictions of \( \varphi \).

**Soundness of DS** follows using that all solutions \( \varphi : [0, r] \to \mathcal{S} \) of \( x' = f(x) \& q(x) \) satisfy \( \varphi(\xi) \in \llbracket q(x) \rrbracket I \) for all \( 0 \leq \xi \leq r \), which, using Lemma 4 as above, is equivalent to \( v \in \llbracket \forall 0 \leq s \leq r [q(x + fs)] \rrbracket I \) when \( v(r) = r \).

**:\=** Soundness of \( [\cdot] := \) follows from the semantics of differential assignments (Def. 6) and compositionality. In detail: \( x' := f \) changes the value of symbol \( x' \) to the value of \( f \). The predicate \( p \) has the same value for arguments \( x' \) and \( f \) if those arguments have same value.

**>+ \&<** Soundness of the derivation axioms \( + \& < \) follows from Lemma 15 since they are special instances of (7) and (8) and (6), respectively. For axiom \( \& \) observe that \( y, y' \notin g(x) \).

Let the premise \( p(\bar{x}) \) be valid in some \( I \), i.e. \( I \models p(\bar{x}) \), i.e. \( \omega \in \llbracket p(\bar{x}) \rrbracket I \) for all \( \omega \). Then, the conclusion \( [a]p(\bar{x}) \) is valid in the same \( I \), i.e. \( v \in \llbracket [a]p(\bar{x}) \rrbracket I \) for all \( v \), because \( \omega \in \llbracket p(\bar{x}) \rrbracket I \) for all \( \omega \), so also for all \( \omega \) with \( (\nu, \omega) \in [a]I \). Thus, \( \mathcal{G} \) is locally sound.

Let the premise \( p(x) \) be valid in some \( I \), i.e. \( I \models p(x) \), i.e. \( \omega \in \llbracket p(x) \rrbracket I \) for all \( \omega \). Then, the conclusion \( \forall x p(x) \) is valid in the same \( I \), i.e. \( v \in \llbracket \forall x p(x) \rrbracket I \) for all \( v \), i.e. \( v^d \in \llbracket p(x) \rrbracket I \) for all \( d \in \mathbb{R} \), because \( \omega \in \llbracket p(x) \rrbracket I \) for all \( \omega \), so in particular for all \( \omega = v^d \) for any real \( d \in \mathbb{R} \). Thus, \( \forall \) is locally sound.
Let the premise \( f(\bar{x}) = g(\bar{x}) \) be valid in some \( I \), i.e. \( I \models f(\bar{x}) = g(\bar{x}) \), which is \( \nu \in [f(\bar{x}) = g(\bar{x})]I \) for all \( \nu \), i.e. \( [f(\bar{x})]IV = [g(\bar{x})]IV \) for all \( \nu \). Consequently, \( [f(\bar{x})]IV \in I(p) \) iff \( [g(\bar{x})]IV \in I(p) \). So, \( I \models f(\bar{x}) \leftrightarrow g(\bar{x}) \). Thus, \( \text{CQ} \) is locally sound.

Let the premise \( p(\bar{x}) \leftrightarrow q(\bar{x}) \) be valid in some \( I \), i.e. \( I \models p(\bar{x}) \leftrightarrow q(\bar{x}) \), which is \( \nu \in [p(\bar{x}) \leftrightarrow q(\bar{x})]I \) for all \( \nu \). Consequently, \( [p(\bar{x})]I = [q(\bar{x})]I \). Thus, \( [C(p(\bar{x}))]I = I(C)(([p(\bar{x})]I = [q(\bar{x})]I) = [C(q(\bar{x}))]I \). This implies \( I \models C(p(\bar{x})) \leftrightarrow C(q(\bar{x})) \), hence the conclusion is valid in \( I \). Thus, \( \text{CE} \) is locally sound.

Rule \( \text{CT} \) is a (locally sound) derived rule and only included for reference. \( \text{CT} \) is derivable from \( \text{CQ} \) using \( \text{p}(\cdot) \overset{\text{def}}{=} (c(\cdot) = c(g(\bar{x}))) \) and reflexivity of =.

Modus ponens \( \text{MP} \) is locally sound with respect to the interpretation \( I \) and the state \( \nu \), which implies local soundness. If \( \nu \in [p \rightarrow q]I \) and \( \nu \in [p]I \) then \( \nu \in [q]I \).

Uniform substitution is sound by Theorem 11 just not locally sound.

Observe that uniform substitutions are not limited to merely instantiating \( \mathcal{L}' \) axioms and axiomatic proof rules. Rule \( \text{US} \) can be used to instantiate any \( \mathcal{L}' \) formula soundly (Theorem 11), which, in particular, gives a simple mechanism for derived axioms and lemmas, which are just \( \mathcal{L}' \) formulas that have a proof. Uniform substitutions can instantiate any locally sound proof as well (Theorem 12), which, in particular, gives a simple mechanism for derived axiomatic rules, definitions, and invariant search with lazy instantiation of invariants. These are just proofs from the \( \mathcal{L}' \) rules and axioms in Fig. 2 and 3 whose premises and conclusions are uniformly substituted to instantiate the requisite function or predicate symbols (recall Example 6).

\textbf{Example 9.} Crucially note that rule \( \text{US} \) itself is only sound but not locally sound, so it cannot have been used on any open premises at any point during a proof that is to be instantiated by proof rule \( \text{USR} \) from Theorem 12. The following sound proof has an open premise on which \( \text{US} \) has been used at some point during the proof:

\[
\begin{align*}
\text{MP} & \quad 1 = 0 
\quad 1 \rightarrow [x' = 2]x < 5 \\
\text{US} & \quad 1 = 0 
\quad [x' = 2]x < 5
\end{align*}
\]

This use of \( \text{US} \) makes the above proof sound but not locally sound, which prevents rule \( \text{USR} \) in Theorem 12 from (incorrectly) concluding a uniform substitution instance under \( \sigma = \{ f(\cdot) \mapsto 0 \} \) of this inference:

\[
\begin{align*}
\text{clash} & \quad 0 = 0 
\quad [x' = 2]x < 5
\end{align*}
\]

Indeed, rule \( \text{US} \) assumes that its premise (here \( f(x) = 0 \)) is valid, so valid in all interpretations \( I \), but the latter (clashing) substitution instance only proves one particular choice for \( f \) to satisfy \( f(x) = 0 \). Contrast this with a use of rule \( \text{US} \) in a premise that closes, which proves that this premise is valid, so valid in all interpretations \( I \) with any interpretations for its program constant and function and predicate symbols, which, thus, continues to make the proof locally sound and Theorem 12 applicable.
5.5 Completeness

By Theorem 16, the \( \mathcal{DL} \) calculus is sound, so every \( \mathcal{DL} \) formula that is provable using the \( \mathcal{DL} \) axioms and proof rules is valid, i.e., true in all states of all interpretations. The more intriguing converse has an answer, too. Namely whether the \( \mathcal{DL} \) calculus is complete, i.e., can prove all \( \mathcal{DL} \) formulas that are valid. Previous calculi for \( \mathcal{DL} \) were proved to be complete relative to differential equations [5, 7] and also proved complete relative to discrete dynamics [7]. A generalization of the Hilbert calculus to hybrid games was even proved complete schematically [9]. The uniform substitution calculus for differential-form \( \mathcal{DL} \) is, to a large extent, a specialization of previous calculi tailored to significantly simplify soundness arguments. Yet, completeness does not transfer when restricting proof calculi. In fact, one key question is whether the restrictions imposed upon proofs for soundness purposes by the simple technique of uniform substitutions does also preserve completeness.

The first challenge is to prove that uniform substitutions are flexible enough to prove all required instances of the \( \mathcal{DL} \) axioms and axiomatic proof rules. For simplicity, consider \( p(\bar{x}) \) to be a quantifier symbol of arity 0.

**Lemma 17** (Surjective axioms). Let \( \varphi \) be a \( \mathcal{DL} \) formula that is built only from quantifier symbols of arity 0 and program constants but no function or predicate symbols. Then \( \varphi \) is surjective, i.e., \( \text{US} \) can instantiate \( \varphi \) to any of its axiom schema instances which are those formulas that are obtained by uniformly replacing program constants \( a \) by any hybrid programs and quantifier symbols \( C() \) by formulas. Axiomatic rules consisting of surjective \( \mathcal{DL} \) formulas are surjective, i.e., \( \text{USR} \) can instantiate them to any of their proof rule schema instances.

**Proof.** Let \( \varphi \) be the desired instance of the axiom schema belonging to \( \varphi \), that is, let \( \varphi \) be obtained from \( \varphi \) by uniformly replacing each quantifier symbol \( C() \) by some formula, naively but consistently (same replacement for \( C() \) in all places) and accordingly for program constants \( a \). The proof is by structural induction on \( \varphi \) to show that there is a uniform substitution \( \sigma \) with \( \text{FV}(\sigma) = /0 \) such that \( \sigma(\varphi) = \tilde{\varphi} \).

The proof for formulas is by simultaneous induction with programs:

1. Consider quantifier symbol \( C() \) of arity 0 and let \( \tilde{\varphi} \) be the desired instance. Define \( \sigma = \{ C() \mapsto \tilde{\varphi} \} \), which has \( \text{FV}(\sigma) = /0 \), because it only substitutes quantifier symbols. Then \( \sigma(C()) \equiv \sigma C() \equiv \tilde{\varphi} \). The substitution is admissible for all arguments, since there are none.

2. Consider \( \phi \land \psi \) and let \( \tilde{\phi} \land \tilde{\psi} \) be the desired instance (which has to have this shape). By induction hypothesis, there are uniform substitutions \( \sigma, \tau \) with \( \text{FV}(\sigma) = \text{FV}(\tau) = /0 \) such that \( \sigma(\phi) = \tilde{\phi} \) and \( \sigma(\psi) = \tilde{\psi} \). Then the union \( \sigma \cup \tau \) of uniform substitutions \( \sigma \) and \( \tau \) is defined, because for all symbols \( a \) of any syntactic category: if \( a \in \sigma \) and \( a \in \tau \), then \( \sigma a = \tau a \) since all replacements are uniform, so the same replacement is used everywhere in \( \phi \land \psi \) for the same symbol \( a \). Consequently, \( (\sigma \cup \tau)(\phi) = \sigma(\phi) = \tilde{\phi} \) and \( (\sigma \cup \tau)(\psi) = \tau(\psi) = \tilde{\psi} \), because all symbols that are replaced are replaced uniformly everywhere so either do not occur in \( \phi \) or are already handled by \( \sigma \) (and likewise either do not occur in \( \psi \) or are already handled by \( \tau \)). Finally, \( \text{FV}(\sigma \cup \tau) = \text{FV}(\sigma) \cup \text{FV}(\tau) = /0 \).
3. Case 2 generalizes to a general uniform replacement argument: the induction hypothesis and uniform replacement assumptions imply for each subexpression $\theta \circ \eta$ of $\phi$ with any operator $\circ$ that the corresponding desired instance has to have the same shape $\tilde{\theta} \circ \tilde{\eta}$ and that there are uniform substitutions $\sigma, \tau$ with $\text{FV}(\sigma) = \text{FV}(\tau) = \emptyset$ such that their union $\sigma \cup \tau$ is defined and $(\sigma \cup \tau)(\theta \circ \eta) = \sigma(\theta) \circ \tau(\eta) = \tilde{\theta} \circ \tilde{\eta}$ and $\text{FV}(\sigma \cup \tau) = \text{FV}(\sigma) \cup \text{FV}(\tau) = \emptyset$.

This shows the cases $\phi \lor \psi, \phi \rightarrow \psi, \phi \leftrightarrow \psi$ and also $\neg \phi$.

4. Consider $\forall x \phi$ with desired instance $\forall x \tilde{\phi}$, which has to have this shape. By induction hypothesis, there is a uniform substitution $\sigma$ with $\text{FV}(\sigma) = \emptyset$ such that $\sigma(\phi) = \tilde{\phi}$. Thus, $\sigma(\forall x \phi) = \forall x \sigma(\phi) = \forall x \tilde{\phi}$, which is $\{x\}$-admissible because $\text{FV}(\sigma) = \emptyset$.

5. The case $\exists x \phi$ is accordingly.

6. Consider $[\alpha] \phi$ with desired instance $[\tilde{\alpha}] \tilde{\phi}$. By induction hypothesis and the uniform replacement argument, there are uniform substitutions $\sigma, \tau$ such that $(\sigma \cup \tau)([\alpha] \phi) = [\sigma(\alpha)] \tau(\phi) = [\tilde{\alpha}] \tilde{\phi}$ which is admissible, because $\sigma \cup \tau$ is $\text{BV}((\sigma \cup \tau)(\alpha))$-admissible for $[\alpha] \phi$ since $\text{FV}(\sigma \cup \tau) = \emptyset$.

7. The case $(\alpha) \phi$ is accordingly.

The proof for hybrid programs is by simultaneous induction with formulas, where most cases are in analogy to the previous cases, except:

1. Consider program constant $a$ with desired instance $\tilde{a}$. Then $\sigma = \{a \mapsto \tilde{a}\}$ has $\text{FV}(\sigma) = \emptyset$ and satisfies $\sigma(a) = \sigma a = \tilde{a}$.

2. Consider the case $x' = \theta \& \psi$ with desired instance $x' = \tilde{\theta} \& \tilde{\psi}$, which has to have this shape.

By induction hypothesis and the uniform replacement argument, there are uniform substitutions $\sigma, \tau$ such that $(\sigma \cup \tau)(x' = \theta \& \psi) \equiv x' = \sigma(\theta) \& \tau(\psi) \equiv x' = \tilde{\theta} \& \tilde{\psi}$, where admissibility again follows from $\text{FV}(\sigma \cup \tau) = \emptyset$.

3. Consider the case $\alpha^*$ with desired instance $(\tilde{\alpha})^*$, which has to have this shape. By induction hypothesis, there is a uniform substitution $\sigma$ such that $\sigma(\alpha) \equiv \tilde{\alpha}$ and $\text{FV}(\sigma) = \emptyset$. Then $\sigma(\alpha^*) \equiv (\sigma(\alpha))^* \equiv (\tilde{\alpha})^*$, which is $\text{BV}(\sigma(\alpha))$-admissible since $\text{FV}(\sigma) = \emptyset$.

4. The case $\alpha; \beta$ is similar and case $\alpha \cup \beta$ follows directly from the uniform replacement argument.

The corresponding result for axiomatic rules built from surjective $\mathcal{L}$ formulas follows since surjective $\mathcal{L}'$ formulas can be instantiated by rule [US] to any instance, which, thus, continues to hold for the premises and conclusions in rule [USR].

Lemma [17] easily generalizes to the case of quantifier symbols with arguments that have no function or predicate symbols, since those are always $\forall' \cup \forall'$-admissible. Generalizations to function and predicate symbol instances are possible with adequate care. Axiom [17] is also surjective.
because it does not have any bound variables, so admissibility of all its instances is obvious. Similarly for rules \( \text{MP} \) and, with a twist, even rule \( \forall \) see proof of Theorem \[16\] Axioms \( \forall \alpha \rightarrow \forall \beta \) can be augmented for surjectivity in similar ways.

This concludes the basis from which the schematic completeness proof relative to any differentially expressive logic \[4\] which was first conducted for differential game logic \[9\], can be adapted and augmented with proofs of uniform substitution instantiability to obtain a (schematic) relative completeness proof for differential-form \( dL \). Both the first-order logic of differential equations \[5\] and discrete dynamic logic \[7\] are differentially expressive for \( dL \).

**Theorem 18** (Relative completeness). The \( dL \) calculus is a sound and complete axiomatization of hybrid systems relative to any differentially expressive logic \( L \), i.e. every valid \( dL \) formula is provable in the \( dL \) calculus from \( L \) tautologies.

**Proof.** This proof refines the completeness proof for the axiom schemata of differential game logic \[9\] with explicit base proofs of \( \text{US} \) and \( \text{USR} \) instantiability. Write \( \models_L \phi \) to indicate that \( dL \) formula \( \phi \) can be derived in the \( dL \) proof calculus from valid \( L \) formulas. Soundness follows from Theorem \[16\] so it remains to prove completeness. For every valid \( dL \) formula \( \phi \) it has to be proved that \( \phi \) can be derived from valid \( L \) tautologies within the \( dL \) calculus: from \( \models_L \phi \) prove \( \models \phi \). The proof proceeds as follows: By propositional recombinasion, inductively identify fragments of \( \phi \) that correspond to \( \phi_1 \rightarrow \langle \alpha \rangle \phi_2 \) or \( \phi_1 \rightarrow [\alpha] \phi_2 \) logically. Find structurally simpler formulas from which these properties can be derived taking care that the resulting formulas are simpler than the original one in a well-founded order. Finally, prove that the original \( dL \) formula can be re-derived from the subproofs in the \( dL \) calculus.

The first insight is that, with the rules \( \text{MP} \) and \( \forall \) and (by Lemma \[17\] all) instances of \( \forall \alpha \rightarrow \forall \beta \) and real arithmetic, the \( dL \) calculus contains a complete axiomatization of first-order logic. Thus, all first-order logic tautologies can be used without further notice in the remainder of the proof. Furthermore, by Lemma \[17\] all instances of \( \langle \cdot \rangle \cup \{ \ast \} \cup \{ \ast \} \cup \{ \ast \} \) can be proved by \( \text{US} \) in the \( dL \) calculus.

By appropriate propositional derivations, assume \( \phi \) to be given in conjunctive normal form. Assume that negations are pushed inside over modalities using the dualities \( \neg [\alpha] \phi \equiv \langle \alpha \rangle \neg \phi \) and \( \neg \langle \alpha \rangle \phi \equiv [\alpha] \neg \phi \) that are provable by axiom \( \{ \} \) and that negations are pushed inside over quantifiers using provable first-order equivalences \( \forall x \phi \equiv \exists x \neg \phi \) and \( \neg \exists x \phi \equiv \forall x \neg \phi \). The remainder of the proof follows an induction on a well-founded partial order \( \prec \) from previous work \[9\] induced on \( dL \) formulas by the lexicographic ordering of the overall structural complexity of the hybrid programs in the formula and the structural complexity of the formula itself, with the logic \( L \) placed at the bottom of the partial order \( \prec \). The base logic \( L \) is considered of lowest complexity by reactivity, because \( \models L \) immediately implies \( \models \) for all formulas \( F \) of \( L \). In the following, \( IH \) is short for induction hypothesis. The proof follows the syntactic structure of \( dL \) formulas.

0. If \( \phi \) has no hybrid programs, then \( \phi \) is a first-order formula; hence provable by assumption (even decidable by rule \( \forall \) [14] if in first-order real arithmetic, i.e. no uninterpreted symbols occur).

\[4\]A logic \( L \) is differentially expressive (for \( dL \) ) if every \( dL \) formula \( \phi \) has an equivalent \( \phi^L \) in \( L \) and all equivalences of the form \( \langle x' = \theta \rangle G \leftrightarrow (\langle x' = \theta \rangle G)^\phi \) are provable in its calculus.
1. \( \phi \) is of the form \( \neg \phi_1 \); then \( \phi_1 \) is first-order and quantifier-free, as negations are assumed to be pushed inside, so Case 0 applies.

2. \( \phi \) is of the form \( \phi_1 \land \phi_2 \), then \( \models L \phi_1 \) and \( \models L \phi_2 \) by IH, which combine propositionally to a proof for \( \models L \phi_1 \land \phi_2 \) using MP twice with the propositional tautology \( \phi_1 \to (\phi_2 \to \phi_1 \land \phi_2) \).

3. The case where \( \phi \) is of the form \( \exists x \phi_2 \), \( \forall x \phi_2 \), \( \langle \alpha \rangle \phi_2 \) or \( [\alpha] \phi_2 \) is included in Case 4 with \( \phi_1 \equiv \text{false} \).

4. \( \phi \) is a disjunction and—without loss of generality—has one of the following forms (otherwise use provable associativity and commutativity to reorder):

\[
\begin{align*}
\phi_1 \lor \langle \alpha \rangle \phi_2 \\
\phi_1 \lor [\alpha] \phi_2 \\
\phi_1 \lor \exists x \phi_2 \\
\phi_1 \lor \forall x \phi_2.
\end{align*}
\]

Let \( \phi_1 \lor \langle \alpha \rangle \phi_2 \) be a unified notation for those cases. Then, \( \phi_2 \prec \phi \), since \( \phi_2 \) has less modalities or quantifiers. Likewise, \( \phi_1 \prec \phi \) because \( \langle \alpha \rangle \phi_2 \) contributes one modality or quantifier to \( \phi \) that is not part of \( \phi_1 \). When abbreviating the simpler formulas \( \neg \phi_1 \) by \( F \) and \( \phi_2 \) by \( G \), the validity \( \models \phi \) yields \( \models \neg F \lor \langle \alpha \rangle G \), so \( \models F \to \langle \alpha \rangle G \), from which the remainder of the proof inductively derives

\[ \models L F \to \langle \alpha \rangle G. \]  

The proof of (11) is by induction on the syntactic structure of \( \langle \alpha \rangle \).

(a) If \( \langle \alpha \rangle \) is the operator \( \forall x \) then \( \models F \to \forall x G \), where \( x \) can be assumed not to occur in \( F \) by a bound variable renaming. Hence, \( \models F \to G \). Since \( G \prec \forall x G \), because it has less quantifiers, also \( (F \to G) \prec (F \to \forall x G) \), hence \( \models F \to G \) is derivable by IH. Then, \( \models L F \to \forall x G \) derives with Lemma 17 by generalization rule \( \forall \), since \( x \) does not occur in \( F \):

\[
\begin{align*}
F & \to G \\
\forall \forall (F \to G) & \to \forall x F \to \forall x G \\
\forall \forall F & \to \forall x G
\end{align*}
\]

The instantiations succeed by the remark after Lemma 17 using for \( \forall \forall \) that \( x \notin \forall (F) \). The formula \( F \to \forall x G \) is even decidable if in first-order real arithmetic [14]. The remainder of the proof concludes \( (F \to \psi) \prec (F \to \phi) \) from \( \psi \prec \phi \) without further notice. The operator \( \forall \forall \) can be obtained correspondingly by uniform renaming.

(b) If \( \langle \alpha \rangle \) is the operator \( \exists x \) then \( \models F \to \exists x G \). If \( F \) and \( G \) are \( L \) formulas, then, since \( L \) is closed under first-order connectives, so is the valid formula \( F \to \exists x G \), which is, then, provable by IH and even decidable if in first-order real arithmetic [14].
Otherwise, $F, G$ correspond to $L$ formulas by expressiveness of $L$, which implies the existence of an $L$ formula $G'$ such that $\models G' \leftrightarrow G$. Since $L$ is closed under first-order connectives, the valid formula $F \rightarrow \exists x (G')$ is provable by IH, because $(F \rightarrow \exists x (G')) \land (F \rightarrow \exists x (G))$ since $G' \in L$ while $G \not\in L$. Now, $\models G' \iff G$ implies $\models G' \rightarrow G$, which is derivable by IH, because $(G' \rightarrow G) \land \phi$ since $G'$ is in $L$. From $\vdash L G' \rightarrow G$, the derivable dual of axiom $\forall \rightarrow G \land (\forall x (p(x) \rightarrow q(x))) \rightarrow (\exists x p(x) \rightarrow \exists x q(x))$, derives $\vdash L \exists x (G') \rightarrow \exists x G$, which combines with $\vdash L F \rightarrow \exists x (G')$ essentially by rule $\text{MP}$ to $\vdash L F \rightarrow \exists x G$.

---

$$G' \rightarrow G$$

$$\forall x (G' \rightarrow G) \rightarrow \exists x (G') \rightarrow \exists x G$$

---

The instantiations succeed by Lemma 17 and the subsequent remark.

(c) $\models F \rightarrow \langle x' = \theta \rangle G$ implies $\models F \rightarrow (\langle x' = \theta \rangle G)'$, which is derivable by IH, because $(F \rightarrow (\langle x' = \theta \rangle G)) \land \phi$ since $(\langle x' = \theta \rangle G)'$ is in $L$. Since $L$ is differentially expressive, $\vdash L \langle x' = \theta \rangle G \iff (\langle x' = \theta \rangle G)'$ is provable, and, consequently, $\vdash L F \rightarrow \langle x' = \theta \rangle G$ derives from $\vdash L F \rightarrow (\langle x' = \theta \rangle G)'$ by $\text{MP}$.

(d) $\models F \rightarrow [x' = \theta] G$ implies $\models F \rightarrow [x' = \theta] \neg G$. Thus, $\models F \rightarrow [x' = \theta] \neg (\langle x' = \theta \rangle G)'$, which is derivable by IH, because $(F \rightarrow [x' = \theta] \neg (\langle x' = \theta \rangle G)) \land \phi$ as $(\langle x' = \theta \rangle G)'$ is in $L$. Since $L$ is differentially expressive, $\vdash L [x' = \theta] G \iff (\langle x' = \theta \rangle G)'$ is provable, so $\vdash L F \rightarrow [x' = \theta] \neg G$ derives from $\vdash L F \rightarrow [x' = \theta] \neg (\langle x' = \theta \rangle G)'$ by propositional congruence. Axiom $[\cdot]$ thus, derives $\vdash L F \rightarrow [x' = \theta] G$ with Lemma 17.

(e) $\models F \rightarrow [x' = \theta \& \psi] G$, then this formula has an equivalent [9, Lemma 3.4] without evolution domains which can be used as a definitorial abbreviation to conclude this case. Similarly for $\models F \rightarrow \langle x' = \theta \& \psi \rangle G$.

(f) The cases where $\alpha$ is of the form $x := \theta$, $\psi$, $\beta \cup \gamma$, or $\beta; \gamma$ are consequences of the soundness of the equivalence axioms $\text{:=}$ plus the duals obtained via the duality axiom $\text{[\cdot]}$. Whenever their respective left-hand side is valid, their right-hand side is valid and of smaller complexity (the programs get simpler), and hence derivable by IH. Thus, $F \rightarrow \langle \alpha \rangle G$ derives by applying the respective axiom. This proof focuses on the $\langle\rangle$ cases, because $\langle\rangle$ cases derive by axiom $\text{:=}$ with Lemma 17 from the $\langle\rangle$ equivalences.

(g) $\models F \rightarrow [y := \theta] G$ then this formula can be proved, using a fresh variable $x \not\in V(\theta) \cup V(G)$, with the following proof by bound variable renaming (rule $\text{BR}$), in which $G^z_\gamma$ is the result of uniformly renaming $y$ to $x$ in $G$

$$\vdash L F \rightarrow \forall x (x = \theta \rightarrow G^z_\gamma)$$

using an equational form of the assignment axiom $\text{:=}$

$$\vdash L [x := f] p(\bar{x}) \leftrightarrow \forall x (x = f \rightarrow p(\bar{x}))$$
The above proof only used equivalence transformations, so its premise is valid iff its conclusion is, which it is by assumption. The assumption, thus, implies \( \vdash F \rightarrow \forall x (x = \theta \rightarrow G^x_\theta) \). Since \( (F \rightarrow \forall x (x = \theta \rightarrow G^x_\theta)) \land (F \rightarrow [y := \theta]G) \), because there are less hybrid programs, \( \vdash_L F \rightarrow \forall x (x = \theta \rightarrow G^x_\theta) \) by IH. The above proof, thus, derives \( \vdash_L F \rightarrow [y := \theta]G \).

The equational assignment axiom \( [\_ := \_] \) can either be adopted as an axiom in place of \( [\_ := \_] \). Or it can be derived from axiom \( [\_ := \_] \) with the uniform substitution \( \sigma = \{ q(\cdot) \mapsto p(\cdot,X) \} \) when splitting the variables \( \bar{x} \) into the variable \( x \) and the other variables \( X \) such that \( x \notin X \):

\[
\begin{align*}
\text{FOL} & \quad q(f) \leftrightarrow \forall x (x = f \rightarrow q(x)) \\
\text{:=} & \quad [x := f]q(x) \leftrightarrow \forall x (x = f \rightarrow q(x)) \\
\text{US} & \quad [x := f]p(x,X) \leftrightarrow \forall x (x = f \rightarrow p(x,X)) \\
& \quad [x := f]p(\bar{x}) \leftrightarrow \forall x (x = f \rightarrow p(\bar{x}))
\end{align*}
\]

It only remains to be shown that \( [\_ := \_] \) can be instantiated as indicated in the above proof. This follows from Lemma 17 with the additional observation that the required uniform substitution \( \\{ f \mapsto \theta \} \) of function symbol \( f \) of arity 0 without argument \( \bar{x} \) will not cause a clash during [US] because the only bound variable \( x \) in \( [\_ := \_] \) is not free in the substitution since \( x \notin \text{V}(\theta) \).

Other proofs involving stuttering and renaming are possible. Direct proofs of \( F \rightarrow [y := \theta]G \) by axiom \( [\_ := \_] \) are possible if the substitution is admissible.

(h) \( \vdash F \rightarrow [?\psi]G \) implies \( \vdash F \rightarrow (\psi \rightarrow G) \). Since \( (\psi \rightarrow G) \land [?\psi]G \), because it has less modalities, \( \vdash_L F \rightarrow (\psi \rightarrow G) \) is derivable by IH. Hence, with the remark after Lemma 17, axiom \( [?] \) instantiates to \( [?\psi]G \leftrightarrow (\psi \rightarrow G) \), so derives \( \vdash_L F \rightarrow [?\psi]G \) by propositional congruence, which is used without further notice subsequently.

(i) \( \vdash F \rightarrow [\beta \cup \gamma]G \) implies \( \vdash F \rightarrow [\beta]G \land [\gamma]G \). Since \( [\beta]G \land [\gamma]G \land [\beta \cup \gamma]G \), because, even if the propositional and modal structure increased, the structural complexity of both hybrid programs \( \beta \) and \( \gamma \) is smaller than that of \( \beta \cup \gamma \) (formula \( G \) did not change), \( \vdash_L F \rightarrow [\beta]G \land [\gamma]G \) is derivable by IH. Hence, with Lemma 17, axiom \( \cup \) instantiates to \( [\beta \cup \gamma]G \leftrightarrow [\beta]G \land [\gamma]G \), so derives \( \vdash_L F \rightarrow [\beta \cup \gamma]G \) by propositional congruence.

(j) \( \vdash F \rightarrow [\beta; \gamma]G \), which implies \( \vdash F \rightarrow [\beta][\gamma]G \). Since \( [\beta][\gamma]G \land [\beta; \gamma]G \), because, even if the number of modalities increased, the overall structural complexity of the hybrid programs decreased because there are less sequential compositions, \( \vdash_L F \rightarrow [\beta][\gamma]G \) is derivable by IH. Hence, with Lemma 17, \( \vdash_L F \rightarrow [\beta; \gamma]G \) derives by axiom \( [;] \) by propositional congruence.

(k) \( \vdash F \rightarrow [\beta^*]G \) can be derived by induction as follows. Formula \( [\beta^*]G \), which expresses that all numbers of repetitions of \( \beta^* \) satisfy \( G \), is an inductive invariant of \( \beta^* \), because \( [\beta^*]G \rightarrow [\beta][\beta^*]G \) is valid, even provable by \( [\_] \). Thus, its equivalent \( L \) encoding is also an inductive invariant:

\[ \varphi \equiv ([\beta^*]G)^\land. \]
Then $F \rightarrow \varphi$ and $\varphi \rightarrow G$ are valid (zero repetitions are possible), so derivable by IH, since $(F \rightarrow \varphi) \prec \varphi$ and $(\varphi \rightarrow G) \prec \varphi$ hold, because $\varphi$ is in $L$. As above, $\varphi \rightarrow [\beta]\varphi$ is valid, and thus derivable by IH, since $\beta$ has less loops than $\beta^*$. By monotonicity rule $\text{M}$ (from $p(\bar{x}) \rightarrow q(\bar{x})$ conclude $[a]p(\bar{x}) \rightarrow [a]q(\bar{x})$), which derives from $\text{GK}$ by Lemma $17$ with a classical argument, as well as rule $\text{ind}$ (from $p(\bar{x}) \rightarrow [a]p(\bar{x})$ conclude $p(\bar{x}) \rightarrow [a^*]p(\bar{x})$), which derives from $\text{JG}$ by Lemma $17$, the respective rules can be instantiated by Lemma $17$ and the resulting derivations combine by $\text{MP}$:

\[
\frac{\varphi \rightarrow [\beta]\varphi}{\varphi \rightarrow [\beta^*]\varphi} \quad \frac{\varphi \rightarrow G}{\varphi \rightarrow [\beta^*]G}
\]

(1) $\vdash F \rightarrow (\beta^*)G$. Let $x$ be the vector of free variables $\text{FV}((\beta^*)G)$. Since $(\beta^*)G$ is a least pre-fixpoint $[\beta^*]$, for all $\mathcal{L}$ formulas $\psi$ with $\text{FV}(\psi) \subseteq \text{FV}((\beta^*)G)$:

$$\vdash \forall x(G \vee (\beta)\psi \rightarrow \psi) \rightarrow ((\beta^*)G \rightarrow \psi)$$

In particular, this holds for a fresh predicate symbol $p$ with arguments $x$:

$$\vdash \forall x(G \vee (\beta)p(x) \rightarrow p(x)) \rightarrow ((\beta^*)G \rightarrow p(x))$$

Using $\vdash F \rightarrow (\beta^*)G$, this implies

$$\vdash \forall x(G \vee (\beta)p(x) \rightarrow p(x)) \rightarrow (F \rightarrow p(x))$$

As $(\forall x(G \vee (\beta)p(x) \rightarrow p(x)) \rightarrow (F \rightarrow p(x))) \prec \varphi$, because, even if the formula complexity increased, the structural complexity of the hybrid programs decreased, because $\varphi$ has one more loop, this fact is derivable by IH:

$$\vdash_L \forall x(G \vee (\beta)p(x) \rightarrow p(x)) \rightarrow (F \rightarrow p(x))$$

The uniform substitution $\sigma = \{p(x) \mapsto (\beta^*)G\}$ is admissible since $\text{FV}(\sigma) = \emptyset$ as $(\beta^*)G$ has free variables $x$. Since, furthermore, $p \notin \Sigma(F) \cup \Sigma(G) \cup \Sigma(\beta)$, $\text{US}$ derives:

$$\forall x(G \vee (\beta)p(x) \rightarrow p(x)) \rightarrow (F \rightarrow p(x))$$

The dual $(a^*)p(\bar{x}) \leftrightarrow p(\bar{x}) \cdot (a^*)p(\bar{x})$ resulting from axiom $[\gamma]$ with axiom $[\varphi]$ by Lemma $17$ continues this derivation by Lemma $17$:

\[
\frac{\forall x(G \vee (\beta)(\beta^*)G \rightarrow (\beta^*)G) \rightarrow (F \rightarrow (\beta^*)G)}{\forall x(G \vee (\beta)(\beta^*)G \rightarrow (\beta^*)G)}
\]

Observe that rule $[\gamma]$ (and $\text{MP}$) instantiates as needed with $\text{USR}$ by Lemma $17$. 

44
This concludes the derivation of \( (11) \), because all operators \( \{ \alpha \} \) for the form \( (11) \) have been considered. From \( (11) \), which is \( \vdash_L \neg \phi_1 \rightarrow \{ \alpha \} \phi_2 \) after resolving abbreviations, \( \vdash_L \phi_1 \lor \{ \alpha \} \phi_2 \) derives propositionally.

This completes the proof of completeness (Theorem 18), because all syntactical forms of \( dL \) formulas have been covered.

With the notable but expected exceptions of loops and differential equations, where (differential) invariant search for parametric predicates \( j(x, x') \) as in Example 6 is in order, the proof of Theorem 18 confirms that unification of axioms on the left and a chase as in Example 7 gives a sufficient proof strategy. This shows relative completeness for the axiomatization considering \( p(\bar{x}) \) as a quantifier symbol. Completeness for the case where \( p(\bar{x}) \) is considered as a predicate symbol with vectorial argument \( \bar{x} \) follows by proving that each instance of axioms and axiomatic proof rules can be obtained. That proof uses a proof rule that allows stuttering to turn \( \phi \) into its equivalent \( [x:=x]\phi \) anywhere, which is needed for uniform substitutions \( \{ p(\cdot) \mapsto [x:=\cdot]\psi \} \) when \( \psi \) has bound occurrences of \( x \) that are not must-bound.

6 Conclusions

With differential forms for local reasoning about differential equations, uniform substitutions lead to a simple and modular proof calculus for differential dynamic logic that is entirely based on axioms and axiomatic rules, instead of soundness-critical schema variables with side-conditions in axiom schemata. The US calculus is straightforward to implement and enables flexible reasoning with axioms by contextual equivalence. Efficiency can be regained by tactics that combine multiple axioms and rebalance the proof to obtain short proof search branches. Contextual equivalence rewriting for implications would be possible directly when adding monotone quantifier symbols \( C \) whose substitution instances limit placeholder \( \_ \) to positive polarity.

Acknowledgment

This material is based upon work supported by the National Science Foundation under NSF CAREER Award CNS-1054246.

The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution or government. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of any sponsoring institution or government.

References

[1] Alonzo Church. A formulation of the simple theory of types. J. Symb. Log., 5(2):56–68, 1940.
[2] Alonzo Church. *Introduction to Mathematical Logic, Volume I*. Princeton University Press, Princeton, NJ, 1956.

[3] Leon Henkin. Banishing the rule of substitution for functional variables. *J. Symb. Log.*, 18(3):pp. 201–208, 1953.

[4] Frank Pfenning. Logical frameworks. In John Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning (in 2 volumes)*, pages 1063–1147. Elsevier and MIT Press, 2001.

[5] André Platzer. Differential dynamic logic for hybrid systems. *J. Autom. Reas.*, 41(2):143–189, 2008. doi:10.1007/s10817-008-9103-8.

[6] André Platzer. Differential-algebraic dynamic logic for differential-algebraic programs. *J. Log. Comput.*, 20(1):309–352, 2010. doi:10.1093/logcom/exn070.

[7] André Platzer. The complete proof theory of hybrid systems. In *LICS*, pages 541–550. IEEE, 2012. doi:10.1109/LICS.2012.64.

[8] André Platzer. The structure of differential invariants and differential cut elimination. *Log. Meth. Comput. Sci.*, 8(4):1–38, 2012. doi:10.2168/LMCS-8(4:16)2012.

[9] André Platzer. Differential game logic. *ACM Trans. Comput. Log.*, 17(1):1:1–1:51, 2015. doi:10.1145/2817824.

[10] André Platzer. A uniform substitution calculus for differential dynamic logic. In Amy Felty and Aart Middeldorp, editors, *CADE*, volume 9195 of *LNCS*, pages 467–481. Springer, 2015. doi:10.1007/978-3-319-21401-6_32.

[11] André Platzer. A uniform substitution calculus for differential dynamic logic. *CoRR*, abs/1503.01981, 2015. arXiv:1503.01981.

[12] André Platzer and Jan-David Quesel. KeYmaera: A hybrid theorem prover for hybrid systems. In Alessandro Armando, Peter Baumgartner, and Gilles Dowek, editors, *IJCAR*, volume 5195 of *LNCS*, pages 171–178. Springer, 2008. doi:10.1007/978-3-540-71070-7_15.

[13] H. Gordon Rice. Classes of recursively enumerable sets and their decision problems. *Trans. AMS*, 89:25–59, 1953.

[14] Alfred Tarski. *A Decision Method for Elementary Algebra and Geometry*. University of California Press, Berkeley, 2nd edition, 1951.

[15] Wolfgang Walter. *Analysis 2*. Springer, 4 edition, 1995.

[16] Wolfgang Walter. *Ordinary Differential Equations*. Springer, 1998.