

1: Operators of Differential Dynamic Logic (dL)

dL	KeYmaera X	Operator	Meaning
$e = d$	<code>e=d</code>	equals	true if values of terms e and d are equal
$e \geq d$	<code>e>=d</code>	greater-or-equal	true if value of e greater-or-equal to value of d
$p(e_1, \dots, e_k)$	<code>p(e1, ..., ek)</code>	predicate	true if p holds for the value of (e_1, \dots, e_k)
$\neg P$	<code>!P</code>	not / negation	true if P is false
$P \wedge Q$	<code>P & Q</code>	and / conjunction	true if both P and Q are true
$P \vee Q$	<code>P Q</code>	or / disjunction	true if P is true or if Q is true
$P \rightarrow Q$	<code>P -> Q</code>	implies / implication	true if P is false or Q is true
$P \leftrightarrow Q$	<code>P <-> Q</code>	equivalent / bi-implication	true if P and Q are both true or both false
$\forall x P$	<code>\forall x P</code>	for all / universal quantifier	true if P is true for all real values of variable x
$\exists x P$	<code>\exists x P</code>	exists / existential quantifier	true if P is true for some real value of variable x
$[a]P$	<code>[a]P</code>	box / $[\cdot]$ modality	true if P is true after all runs of HP a
$\langle a \rangle P$	<code><a>P</code>	diamond / $\langle \cdot \rangle$ modality	true if P is true after at least one run of HP a

Unary operators (including $\forall x, \exists x, [a], \langle a \rangle$) bind stronger than binary operators. And ; binds stronger than \cup .

2: Statements and effects of Hybrid Programs (HPs)

HP	KeYmaera X	Operation	Effect
$x := e$	<code>x:=e;</code>	discrete assignment	assigns value of term e to variable x
$x := *$	<code>x:=*;</code>	nondeterministic assign	assigns any real value to variable x
$x' = f(x) \& Q$	<code>{x'=f(x) & Q}</code>	continuous evolution	evolve along differential equation $x' = f(x)$ within evolution domain Q for any duration
$?Q$	<code>?Q;</code>	test	check first-order formula Q at current state
$a; b$	<code>a b</code>	sequential composition	HP b starts after HP a finishes
$a \cup b$	<code>a ++ b</code>	nondeterministic choice	choice between alternatives HP a or HP b
a^*	<code>{a}*</code>	nondeterministic repetition	repeats HP a n -times for any $n \in \mathbb{N}$

```

Definitions      /* function symbols cannot change their value */
  Real A = 5;      /* real-valued maximum acceleration constant is defined as 5 */
  Real B;         /* real-valued maximum braking constant is arbitrary */
End.

```

```

ProgramVariables /* program variables may change their value over time */
  Real x;         /* real-valued position */
  Real v;         /* real-valued velocity */
  Real a;         /* current acceleration chosen by controller */
End.

```

```

Problem          /* conjecture in differential dynamic logic */
  v>=0 & A>0 & B>0 /* initial condition */
->                /* implies */
[                 /* all runs of hybrid program in [...] */
 {               /* braces {} for grouping of programs */
   {?v<=5;a:=A; ++ a:=0; ++ a:=-B;} /* nondeterministic choice of acceleration a */
   {x'=v, v'=a & v>=0}           /* differential equation system with domain */
 }* @invariant(v>=0)           /* loop repeats, @invariant contract */
] v>=0                /* safety/postcondition after hybrid program */
End.

```

3: Axioms

assignb $[:=] [x := e]p(x) \leftrightarrow p(e)$

randomb $[:*] [x := *]p(x) \leftrightarrow \forall x p(x)$

testb $[?] [?Q]P \leftrightarrow (Q \rightarrow P)$

solve $['] [x' = f(x) \& q(x)]p(x) \leftrightarrow \forall t \geq 0 ((\forall 0 \leq s \leq t q(x(s))) \rightarrow [x := x(t)]p(x))$ (if $x'(t) = f(x(t))$)

choiceb $[\cup] [a \cup b]P \leftrightarrow [a]P \wedge [b]P$

composeb $[:] [a; b]P \leftrightarrow [a][b]P$

iterateb $[*] [a^*]P \leftrightarrow P \wedge [a][a^*]P$

diamond $\langle \cdot \rangle \neg[a]\neg P \leftrightarrow \langle a \rangle P$

K $K [a](P \rightarrow Q) \rightarrow ([a]P \rightarrow [a]Q)$

I $I [a^*]P \leftrightarrow P \wedge [a^*](P \rightarrow [a]P)$

V $V p \rightarrow [a]p$

4: Differential equation sequent calculus proof rules

dW dW $\frac{\Gamma_{\text{const}}, Q \rightarrow p(x), \Delta_{\text{const}}}{\Gamma \rightarrow [x' = f(x) \& Q]p(x), \Delta}$

dI dI $\frac{\Gamma, Q \rightarrow P, \Delta \quad Q \rightarrow [x' := f(x)](P)'}{\Gamma \rightarrow [x' = f(x) \& Q]P, \Delta}$

dC dC $\frac{\Gamma \rightarrow [x' = f(x) \& Q]P, \Delta \quad \Gamma \rightarrow [x' = f(x) \& Q \wedge C]P, \Delta}{\Gamma \rightarrow [x' = f(x) \& Q]P, \Delta}$

dG dG $\frac{\Gamma \rightarrow \exists y [x' = f(x), y' = a(x)y + b(x) \& Q]P, \Delta}{\Gamma \rightarrow [x' = f(x) \& Q]P, \Delta}$

5: Propositional sequent calculus proof rules

notR $\neg R \frac{\Gamma, P \rightarrow \Delta}{\Gamma \rightarrow \neg P, \Delta}$ andR $\wedge R \frac{\Gamma \rightarrow P, \Delta \quad \Gamma \rightarrow Q, \Delta}{\Gamma \rightarrow P \wedge Q, \Delta}$ orR $\vee R \frac{\Gamma \rightarrow \Delta, P, Q}{\Gamma \rightarrow P \vee Q, \Delta}$

notL $\neg L \frac{\Gamma \rightarrow \Delta, P}{\neg P, \Gamma \rightarrow \Delta}$ andL $\wedge L \frac{\Gamma, P, Q \rightarrow \Delta}{P \wedge Q, \Gamma \rightarrow \Delta}$ orL $\vee L \frac{P, \Gamma \rightarrow \Delta \quad Q, \Gamma \rightarrow \Delta}{P \vee Q, \Gamma \rightarrow \Delta}$

implyR $\rightarrow R \frac{\Gamma, P \rightarrow \Delta, Q}{\Gamma \rightarrow P \rightarrow Q, \Delta}$ equivR $\leftrightarrow R \frac{\Gamma, P \rightarrow \Delta, Q \quad \Gamma, Q \rightarrow \Delta, P}{\Gamma \rightarrow P \leftrightarrow Q, \Delta}$

implyL $\rightarrow L \frac{\Gamma \rightarrow \Delta, P \quad Q, \Gamma \rightarrow \Delta}{P \rightarrow Q, \Gamma \rightarrow \Delta}$ equivL $\leftrightarrow L \frac{P \wedge Q, \Gamma \rightarrow \Delta \quad \neg P \wedge \neg Q, \Gamma \rightarrow \Delta}{P \leftrightarrow Q, \Gamma \rightarrow \Delta}$

close id $\frac{}{P, \Gamma \rightarrow P, \Delta}$ closeTrue $\top R \frac{}{\Gamma \rightarrow \text{true}, \Delta}$ hideR WR $\frac{\Gamma \rightarrow \Delta}{\Gamma \rightarrow P, \Delta}$ PR $\frac{\Gamma \rightarrow Q, P, \Delta}{\Gamma \rightarrow P, Q, \Delta}$

cut cut $\frac{\Gamma, C \rightarrow \Delta \quad \Gamma \rightarrow \Delta, C}{\Gamma \rightarrow \Delta}$ closeFalse $\perp L \frac{}{\text{false}, \Gamma \rightarrow \Delta}$ hideL WL $\frac{\Gamma \rightarrow \Delta}{P, \Gamma \rightarrow \Delta}$ PL $\frac{Q, P, \Gamma \rightarrow \Delta}{P, Q, \Gamma \rightarrow \Delta}$

6: Quantifier sequent calculus proof rules

allR $\forall R \frac{\Gamma \rightarrow p(y), \Delta}{\Gamma \rightarrow \forall x p(x), \Delta}$ ($y \notin \Gamma, \Delta, \forall x p(x)$) existsR $\exists R \frac{\Gamma \rightarrow p(e), \Delta}{\Gamma \rightarrow \exists x p(x), \Delta}$ (arbitrary term e)

allL $\forall L \frac{\Gamma, p(e) \rightarrow \Delta}{\Gamma, \forall x p(x) \rightarrow \Delta}$ (arbitrary term e) existsL $\exists L \frac{\Gamma, p(y) \rightarrow \Delta}{\Gamma, \exists x p(x) \rightarrow \Delta}$ ($y \notin \Gamma, \Delta, \exists x p(x)$)

7: dL Sequent calculus proof rules

loop	loop	$\frac{\Gamma \vdash J, \Delta \quad J \vdash P \quad J \vdash [a]J}{\Gamma \vdash [a^*]P, \Delta}$	CEat CER	$\frac{\Gamma \vdash C(Q), \Delta \quad Q \leftrightarrow P}{\Gamma \vdash C(P), \Delta}$
generalize	MR	$\frac{\Gamma \vdash [a]Q, \Delta \quad Q \vdash P}{\Gamma \vdash [a]P, \Delta}$	CEat CEL	$\frac{\Gamma, C(Q) \vdash \Delta \quad Q \leftrightarrow P}{\Gamma, C(P) \vdash \Delta}$
generalize	ML	$\frac{\Gamma, [a]Q \vdash \Delta \quad P \vdash Q}{\Gamma, [a]P \vdash \Delta}$	CEat CQR	$\frac{\Gamma \vdash p(k), \Delta \quad k = e}{\Gamma \vdash p(e), \Delta}$
abstract	GVR	$\frac{\Gamma_{\text{const}} \vdash P, \Delta_{\text{const}}}{\Gamma \vdash [a]P, \Delta}$	CEat CQL	$\frac{\Gamma, p(k) \vdash \Delta \quad k = e}{\Gamma, p(e) \vdash \Delta}$
discreteGhost	iG	$\frac{\Gamma \vdash [y := e]p, \Delta}{\Gamma \vdash p, \Delta} \quad (y \text{ new})$	CTR	$\frac{e = k}{\Gamma \vdash c(e) = c(k), \Delta}$
			CTL	$\frac{e = k}{\Gamma, c(e) = c(k) \vdash \Delta}$
US	US	$\frac{\Gamma \vdash \Delta}{\sigma(\Gamma) \vdash \sigma(\Delta)}$	UR	$\frac{\Gamma \frac{y}{x} \vdash \Delta \frac{y}{x}}{\Gamma \vdash \Delta}$
	BRR	$\frac{\Gamma \vdash [y := e]\varphi \frac{y}{x}, \Delta}{\Gamma \vdash [x := e]\varphi, \Delta}$	BRL	$\frac{\Gamma, [y := e]\varphi \frac{y}{x} \vdash \Delta}{\Gamma, [x := e]\varphi \vdash \Delta} \quad (y, y', x' \notin \text{FV}(\varphi))$

8: Differential equation axioms

DW	DW	$[x' = f(x) \ \& \ Q]P \leftrightarrow [x' = f(x) \ \& \ Q](Q \rightarrow P)$
DI	DI	$([x' = f(x) \ \& \ Q]P \leftrightarrow [?Q]P) \leftarrow (Q \rightarrow [x' = f(x) \ \& \ Q])(P)'$
DC	DC	$([x' = f(x) \ \& \ Q]P \leftrightarrow [x' = f(x) \ \& \ Q \ \& \ C]P) \leftarrow [x' = f(x) \ \& \ Q]C$
DE	DE	$[x' = f(x) \ \& \ Q]P \leftrightarrow [x' = f(x) \ \& \ Q][x' := f(x)]P$
DG	DG	$[x' = f(x) \ \& \ Q]P \leftrightarrow \exists y [x' = f(x), y' = a(x)y + b(x) \ \& \ Q]P$

9: First-order axioms

allInst	$\forall i (\forall x p(x)) \rightarrow p(e)$
allDist	$\forall \rightarrow \forall x (P \rightarrow Q) \rightarrow (\forall x P \rightarrow \forall x Q)$
allV	$\forall \forall p \rightarrow \forall x p$
	$\exists \neg \forall x \neg P \leftrightarrow \exists x P$

10: Derived rules

cohideR	WR	$\frac{\vdash P}{\Gamma \vdash P, \Delta}$	cutR	cutR	$\frac{\Gamma \vdash Q, \Delta \quad \Gamma \vdash Q \rightarrow P, \Delta}{\Gamma \vdash P, \Delta}$	commuteEquivR	$\leftrightarrow cR \frac{\Gamma \vdash Q \leftrightarrow P, \Delta}{\Gamma \vdash P \leftrightarrow Q, \Delta}$
cohideL	WL	$\frac{P \vdash}{P, \Gamma \vdash \Delta}$	cutL	cutL	$\frac{Q, \Gamma \vdash \Delta \quad \Gamma \vdash \Delta, P \rightarrow Q}{P, \Gamma \vdash \Delta}$	commuteEquivL	$\leftrightarrow cL \frac{Q \leftrightarrow P, \Gamma \vdash \Delta}{P \leftrightarrow Q, \Gamma \vdash \Delta}$
cohide2	WLR	$\frac{P \vdash Q}{P, \Gamma \vdash Q, \Delta}$				equivifyR	$\rightarrow 2 \leftrightarrow \frac{\Gamma \vdash P \leftrightarrow Q, \Delta}{\Gamma \vdash P \rightarrow Q, \Delta}$

11: Differential axioms

DS	DS $[x' = c() \ \& \ q(x)]p(x) \leftrightarrow \forall t \geq 0 ((\forall 0 \leq s \leq t q(x + c()s)) \rightarrow [x := x + c()t]p(x))$
Dconst	$c'(c())' = 0$
Dvar	$x'(x)' = x'$
Dplus	$+'(e + k)' = (e)' + (k)'$
Dminus	$-'(e - k)' = (e)' - (k)'$
Dtimes	$\cdot'(e \cdot k)' = (e)' \cdot k + e \cdot (k)'$
Dquotient	$/'(e/k)' = ((e)' \cdot k - e \cdot (k)')/k^2$
Dcompose	$\circ'[y := g(x)][y' := 1]((f(g(x)))' = (f(y))' \cdot (g(x))')$

12: Bellerophon tactic language operators for proof search

Bellerophon	Operation	Effect
$s ; t$	sequential composition	run t on the output of s , failing if either fail
$s t$	alternative choice	run t if applying s failed, failing if both fail
t^*	saturating repetition	repeats tactic t until nothing changes any more
t^n	fixed repetition	repeats tactic t exactly n times, failing if any of those repetitions fail
$\langle t_1, \dots, t_n \rangle$	branching	runs tactic t_i on branch i , failing if any fail or if branches $\neq n$
$\text{doall}(t)$	all branches	runs tactic t on all branches i , failing if that fails on any branch
$t(j)$	at position	applies tactic t at position j of the sequent
$t(j, \{e\})$	at position	applies tactic t to expression e , which is at position j of the sequent
1	succedent position	position of first succedent formula. Similar 2, 3, ..., 'Rlast
-1	antecedent position	position of first antecedent formula. Similar -2, -3, ..., 'Llast
-4.0.1	subposition	second child of first child of fourth antecedent formula. Similar 4.1
'R	search succedent	first applicable succedent position (where formula e is, if specified)
'L	search antecedent	first applicable antecedent position (where formula e is, if specified)

```

/* Tactic "master" suffices. Mix explicit and searchy tactic for above example. */
implyR(1) ; andL('L)* ; loop({ 'v>=0' }, 1) ; <( /* < splits separate branches */
  closeId , /* initial case: shown with close by identity */
  QE, /* postcondition: prove by real arithmetic QE */
  /* induction step: decomposes hybrid program semi-explicitly */
  composeb(1) ; solve(1.1) ; choiceb(1) ; andR(1) ; <( /* controller branches */
    composeb(1) ; testb(1) ; master, /* decompose some steps then ask master */
    choiceb(1) ; andR(1) ; doall(assignb(1) ; QE) /* doall same on all branches */
  )
)

```

- [1] André Platzer. *Logical Foundations of Cyber-Physical Systems*. Springer, Switzerland, 2018. URL: <http://www.springer.com/978-3-319-63587-3>, doi:10.1007/978-3-319-63588-0.
- [2] Nathan Fulton, Stefan Mitsch, Jan-David Quesel, Marcus Völz, and André Platzer. KeYmaera X: An axiomatic tactical theorem prover for hybrid systems. In Amy Felty and Aart Middeldorp, editors, *CADE*, volume 9195 of *LNCS*, pages 527–538, Berlin, 2015. Springer. doi:10.1007/978-3-319-21401-6_36.
- [3] André Platzer. A complete uniform substitution calculus for differential dynamic logic. *J. Autom. Reas.*, 59(2):219–265, 2017. doi:10.1007/s10817-016-9385-1.
- [4] Nathan Fulton, Stefan Mitsch, Brandon Bohrer, and André Platzer. Bellerophon: Tactical theorem proving for hybrid systems. In Mauricio Ayala-Rincón and César A. Muñoz, editors, *ITP*, volume 10499 of *LNCS*, pages 207–224. Springer, 2017. doi:10.1007/978-3-319-66107-0_14.