A Temporal Differential Dynamic Logic Formal Embedding

Lauren M. White[1], Laura Titolo[2],
J. Tanner Slagel[1], and César Muñoz[1]

[1]NASA Langley Research Center
Hampton, VA, USA

[2] AMA Inc.
Hampton, VA, USA

email: lauren.m.white@nasa.gov

January 2024 at Certified Programs and Proofs (CPP) 2024

Overview

- **$\mathbf{dTL^2}$:** Differential Temporal Dynamic Logic [1]
- **PVS:** Interactive theorem prover [2]

Result: Embedding of $\mathbf{dTL^2}$ in PVS

- Extend the functionality of the embedding of $\mathbf{dL}$ in **PVS**, **Plaidypvs** [3] by allowing for temporal reasoning
- Fully operational in **PVS**

Plaidypvs: **P**roperly **A**ssured **I**mplementation of Differential **Dy**namic Logic for Hybrid **P**rogram **V**erification and **S**pecification

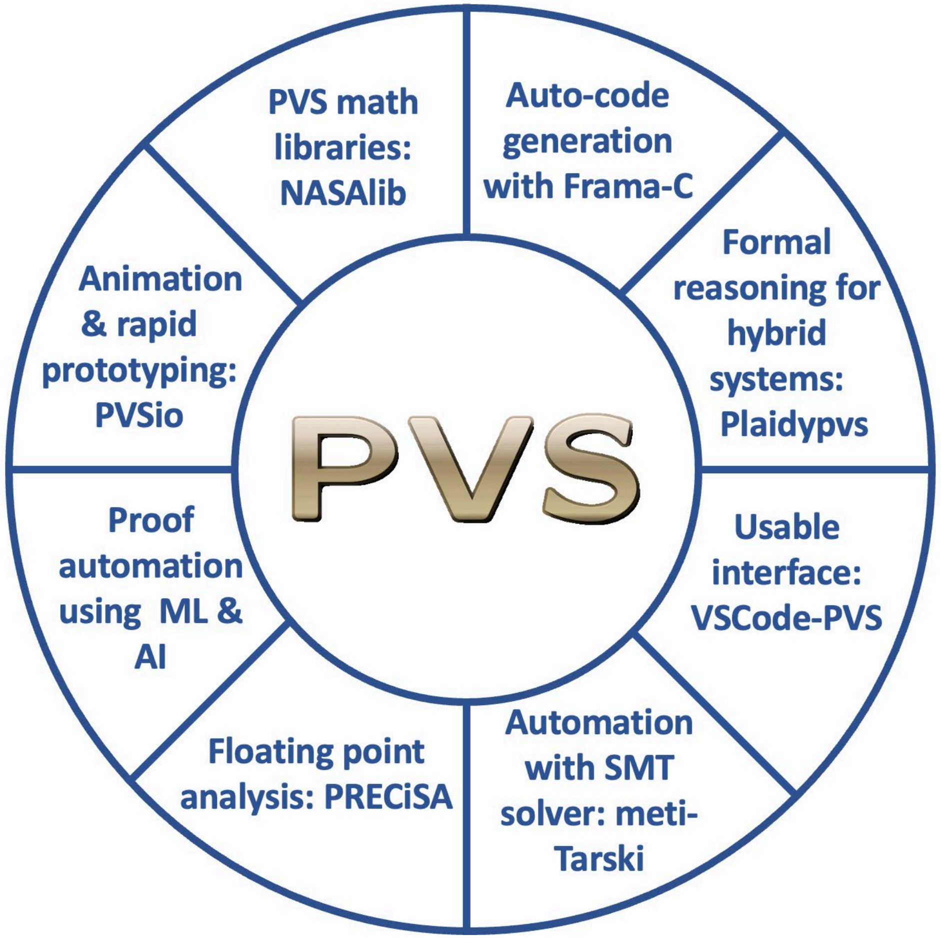
[1] Jeannin & Platzer. 2014. "dTL^2: differential temporal dynamic logic with nested temporalities for hybrid systems." Springer International.

[2] PVS website, SRI International: https://pvs.csl.sri.com

[3] The embedding of dL in PVS, NASA's PVS library: https://github.com/nasa/pvslib/dL

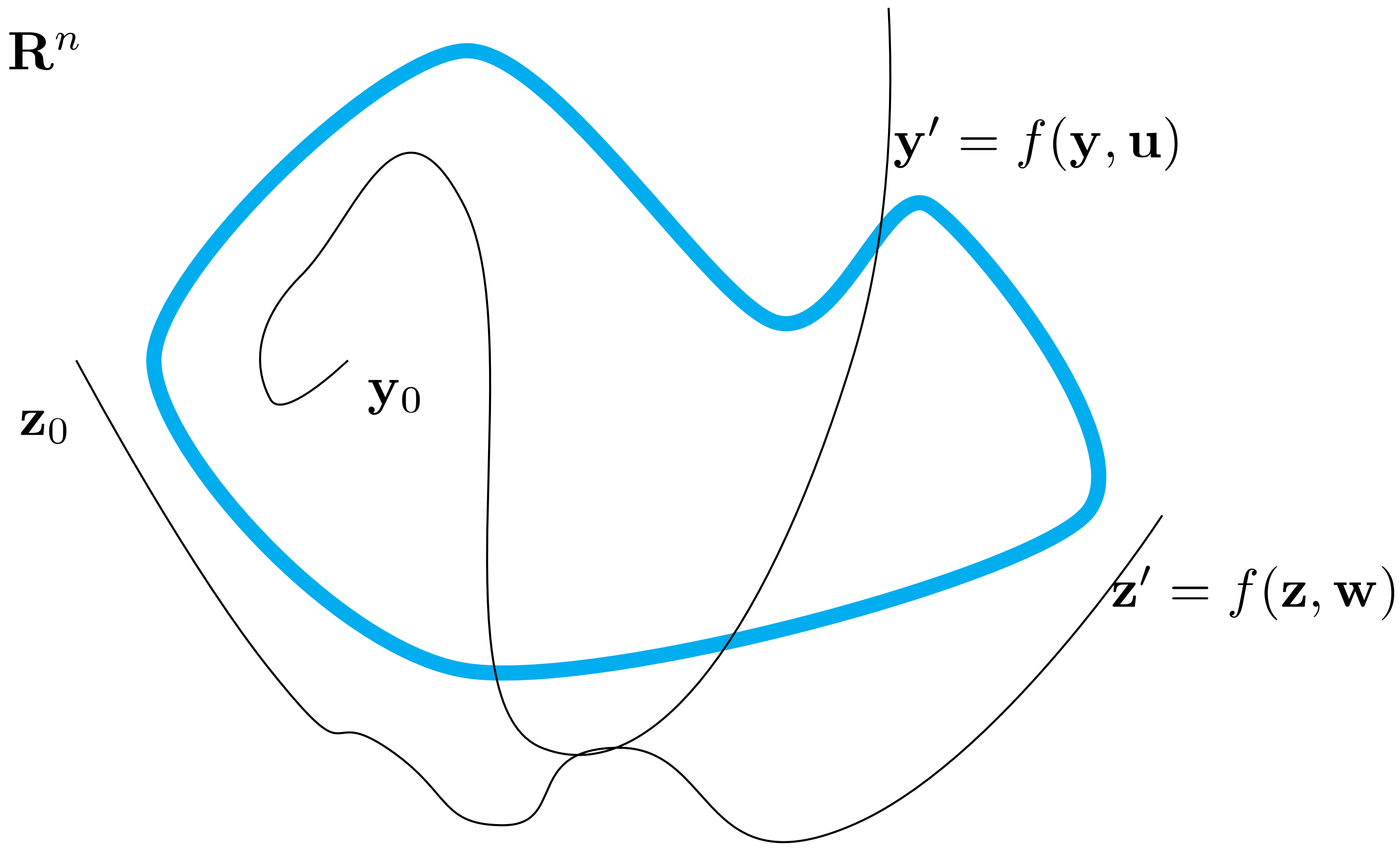
PVS

- "Prototype Verification System" developed by SRI International
- Interactive theorem prover
  - Higher order logic
  - Completely typed, dependent types
- Automation
  - Customizable tactics and strategies
- PVSio animation and rapid prototyping
- NASA PVS library [4]
  - 58 libraries with over 35,000 lemmas
- Visual studio code extension [5]



[4] NASAlib, maintained by NASA Langley Formal Methods Group: https://github.com/nasa/pvslib
[5] VSCode-PVS, Paolo Masci: https://github.com/nasa/vscode-pvs

Plaidypvs and Hybrid Systems



- **Hybrid system:** dynamical system that exhibits
  - Continuous behavior
  - Discrete behavior

Plaidypvs:

- **Formal specification** of hybrid systems
- **Formal reasoning** of hybrid systems

Hybrid Programs

Hybrid programs allow **formal specification** of hybrid systems:

- Discrete jump set:

  $(x_1 \coloneqq \theta_1, \dots, x_n \coloneqq \theta_n)$
- Differential equations:

$\{x_1' \coloneqq \theta_1, \dots, x_n' \coloneqq \theta_n \;\&\; \chi\}$

  - $\{x_i\}_{i=1}^{n}$ variables
  - $\{\theta_i\}_{i=1}^{n}$ assignments (ex. – functions of past variable assignments)
  - χ first-order formula that describes the domain

For hybrid programs Hp_1, Hp_2, first-order formula χ:

- Union

  $(Hp_1 \cup Hp_2)$
- Sequence

  $(Hp_1; Hp_2)$
- Repeat

  $(Hp_1)^*$
- Test

  $(?\,\chi)$

Hybrid Programs (continued)

Example: $$\alpha \coloneqq \big((?\,(v < V_0); \{v' = M \,\&(v \le V_0 + \tau)\}) \\ \cup\, (?\,(v \ge V_0); \{v' = -M \,\&(v \ge V_0 - \tau)\})\big)^*$$

The hybrid program α models a simple cruise controller.

When the velocity is less than V_0, set the acceleration to M and when the velocity is greater than or equal to V_0, set the acceleration to $-M$. The velocity is allowed to be between $V_0 - \tau$ and $V_0 + \tau$.

Semantics of $\mathbf{dL}$ vs. $\mathbf{dTL^2}$

In $\mathbf{dL}$:

- Points in space defined as Environment types

$$\mathcal{E} : [\mathbb{N} \to \mathbb{R}]$$

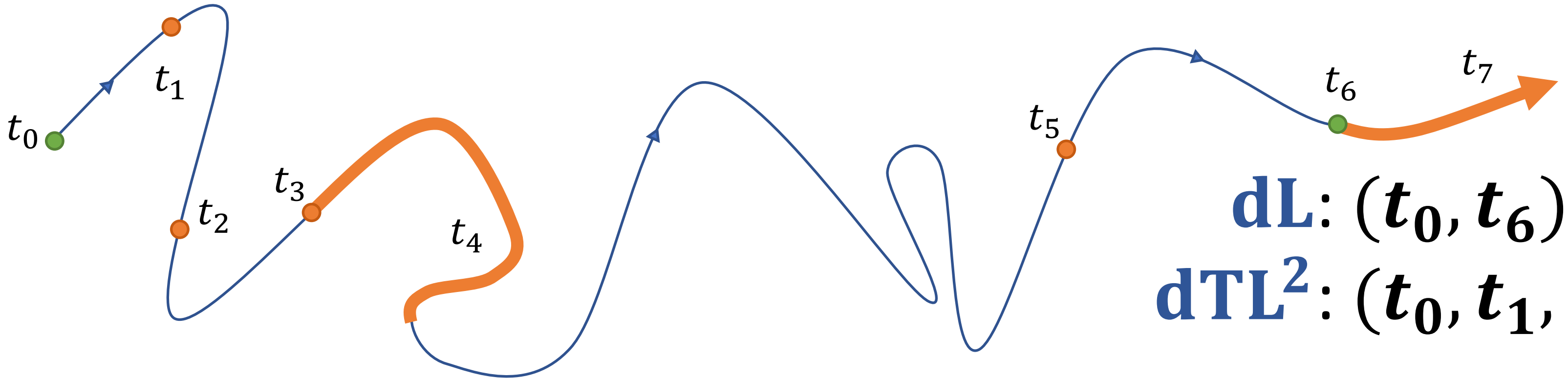
- Semantically relates input/output states of an HP.

In $\mathbf{dTL^2}$:

- Traces defined as lists with elements:

1. Environment type
2. $f : [D \to \mathcal{E}]$ where D is finite or infinite
3. Error

- Elements of the trace are semantically related in chronological order.



$\mathbf{dL}$: $(\boldsymbol{t_0}, \boldsymbol{t_6})$

$\mathbf{dTL^2}$: $(\boldsymbol{t_0}, \boldsymbol{t_1}, \dots, \boldsymbol{t_6}, \boldsymbol{t_7})$

$\mathbf{dL}$ and the $\mathbf{dTL^2}$ extension

$\mathbf{dL}$ allows **formal reasoning** of the input/output (io) of hybrid programs:

- For hybrid program Hp and state predicate P
  - Allruns, Someruns for io
    $[Hp]_{st}\mathrm{P}, \quad \langle Hp\rangle_{st}\mathrm{P}$

$\mathbf{dTL^2}$ extends $\mathbf{dL}$ by also allowing for **formal reasoning** of traces:

- For a trace predicate ϕ
  - Allruns, Someruns for traces
    $[Hp]_{tr}\phi, \quad \langle Hp\rangle_{tr}\phi$
  - Globally, Eventually
    $\Box\phi, \quad \Diamond\phi$

$$t \vDash \Box\phi \coloneqq \text{For all suffixes } s \text{ of } t,\ s \vDash \phi$$

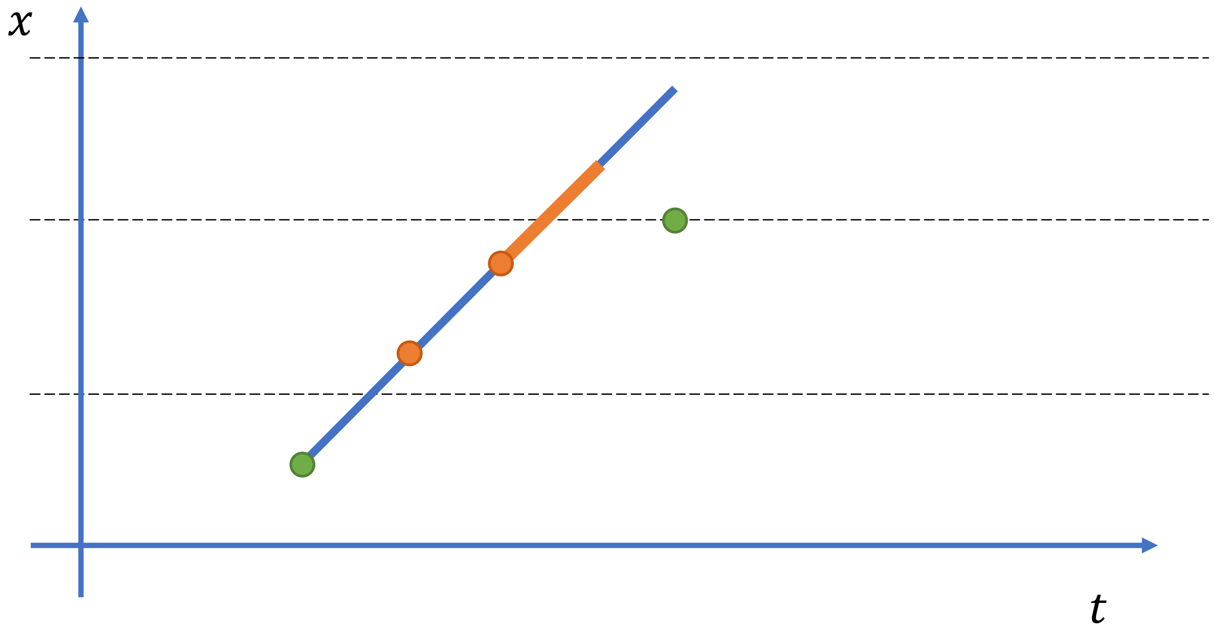
$$t \vDash \Diamond\phi \coloneqq \text{Exists a suffix } s \text{ of } t,\ s \vDash \phi$$

Solving Problems in $\mathbf{dL}$ and $\mathbf{dTL}^2$

Example:

$$\alpha \coloneqq \{x' = 6\}; (x \coloneqq \lfloor x \rfloor)$$

TRUE: $[\alpha]_{st}(x \in \mathbb{Z})$ and $\langle\alpha\rangle_{tr}\Diamond(x \notin \mathbb{Z})$



Elimination of Error Traces

Example:

$$\alpha \coloneqq x \coloneqq 0; (?\, x < 10; x \coloneqq x + 1\,)^*; ?\, x \geq 10$$

Semantics with errors

$$\begin{aligned}[\![\alpha]\!] = \{\, & (x = 0) \cdot ERROR, \ldots, \\ & (x = 0) \cdot (x = 1) \cdots (x = 9) \cdot ERROR, \\ & (x = 0) \cdot (x = 1) \cdots (x = 9) \cdot (x = 10)\}\end{aligned}$$

Semantics without errors

$$[\![\alpha]\!] = \{(x = 0) \cdot (x = 1) \cdots (x = 9) \cdot (x = 10)\}$$

After the elimination of error traces…

$$[\alpha]_{tr} \Diamond (x = 10)$$

$\mathbf{dTL^2}$: Differential Temporal Dynamic Logic – Rule Schema

Assignment:

$$\frac{\phi \vee [x := \theta]_{st}(\psi \vee \phi)}{[x := \theta]_{tr}\psi \sqcup \Diamond\phi} \; ([:=]\sqcup)$$

Differential Equation:

$$\frac{\neg\chi \vee (\phi \wedge [\bar{x} = \bar{\theta}\&\chi]_{st}(\psi \wedge \phi))}{[\bar{x} = \bar{\theta}\&\chi]_{tr}\psi \sqcap \Box\phi} \; (['\,]\sqcap)$$

Repetition:

$$\frac{[\alpha]_{tr}(([\beta]_{tr}(\phi \sqcup \Diamond\psi)) \sqcup \Diamond\psi)}{[\alpha;\beta]_{tr}(\phi \sqcup \Diamond\psi)} \; ([;]\sqcup)$$

Test:

$$\frac{\neg\chi \vee \phi}{[?\chi]_{tr}\phi \blacktriangleleft \Diamond\Box\psi} \; ([?] \blacktriangleleft \Diamond)$$

..and more!

Hybrid Programs in PVS

Elements in a Trace list

typecheck-file | evaluate-in-pvsio | view-as-markdown

```
TraceState : DATATYPE
BEGIN
  IMPORTING hp_def,
            hp_expr

  STATE(state : Environment) : state?
  INF_DIFF(s0:Environment, inf_behavior:{ib:[(hp(0)) -> Environment] | ib(0)=s0}) : inf_diff?
  STATE_DIFF(D:{D:(dd?) | EXISTS(b:posreal): D = closed_interval(0,b)}
            , s0:Environment
            , behavior:{b:[(D) -> Environment]|b(0)=s0}) : state_diff?
  ERROR : error?

END TraceState
```

Trace semantics of **ASSIGN**

```
LET envi = car(trace),
    envo = nth(trace,1),
    l = assigns(hp) IN
state?(envi) AND
state?(envo) AND
  ((FORALL (i:below(length(l))) :
   LET (vari,rexpr) = nth(l,i) IN
     state(envo)(vari) = rexpr(state(envi)) AND
     FORALL (varj:(not_in_map(l))) :
     state(envo)(varj) = state(envi)(varj))
```

Well founded Traces

```
wf_trace?(trace:list[TraceState]) : bool =
  cons?(trace) AND (NOT error?(car(trace))) AND
  FORALL (i:below(length(trace)-1)) :
    state?(nth(trace,i)) OR state_diff?(nth(trace,i))
```

Formal Verification of Soundness of $\mathbf{dTL^2}$

Sequential Composition:

```
%  [;]⊓
% [A]([B](P ⊓ □Q) ⊓ □Q)
% ------------------------
% [A;B](P ⊓ □Q)
```

prove | discharge-tccs | status-proofchain | show-prooflite

```
dltl_SEQcap: LEMMA
    ALLRUNS_tr(A, normDLGLOBALLY(ALLRUNS_tr(B, normDLGLOBALLY(P,Q)), Q)) |-
    ALLRUNS_tr(SEQ(A,B), normDLGLOBALLY(P,Q))
```

Test:

```
%  [?]⊓
% ¬P ∨ (Q ∧ R)
% ========================
% [?P](Q ⊓ □R)
```

prove | discharge-tccs | status-proofchain | show-prooflite

```
dltl_TESTcap_eq: LEMMA
DLOR(DLNOT(P),DLAND(Q,R)) = ALLRUNS_tr(TEST(P),normDLGLOBALLY(Q,R))
```

84 rules/axioms of $\mathbf{dTL^2}$ in PVS

Example

Specification (.pvs)

```
floor_re(re:RealExpr)(env:Environment): real = floor(re(env))
integer_pred_re?(re:RealExpr)(env:Environment): bool = integer_pred(re(env))

prove | discharge-tccs | status-proofchain | show-prooflite
eventually_not_int: LEMMA
SOMERUNS_tr(SEQ(DIFF((:(x,cnst(6)):)),ASSIGN((:(x,floor_re(val(x))):))),
            DLEVENTUALLY(State_Trace(NOT integer_pred_re?(val(x)))))

prove | discharge-tccs | status-proofchain | show-prooflite
end_state_int: LEMMA
ALLRUNS(SEQ(DIFF((:(x,cnst(6)):)),ASSIGN((:(x,floor_re(val(x))):))),
                       integer_pred_re?(val(x)))
```

Interactive theorem prover

```
PVS PROOF EXPLO...
  eventually_not_int (unfinished)
    • (rewrite "norm_DLEVENTUALLY")
    ★ (rewrite "dltl_SEQcup_dual" :dir rl)
    ★ (rewrite "dltl_ASSIGNcup_dual" :...
    ★ (dl-assert)
    ★ (rewrite "dltl_DIFFcup_dual_e...
      ☆ (1)
      ☆ (2)
```

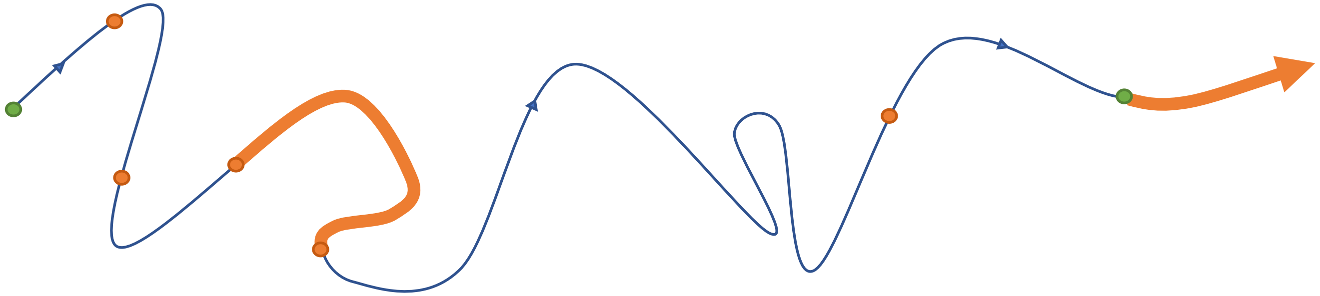
```
eventually_not_int.1 :

  |-------
{1}   ((: :) |-
       (: SOMERUNS_st(ASSIGN((: (x, floor_re(val(x))) :)),
                      DLNOT(integer_pred_re?(val(x)))),
          SOMERUNS_st(DIFF((: (x, cnst(6)) :), DLBOOL(TRUE)),
                      DLOR(DLOR(DLNOT(integer_pred_re?(val(x))),
                                SOMERUNS_st(ASSIGN
                                            ((: (x, floor_re(val(x))) :)),
                                            DLNOT
                                            (integer_pred_re?(val(x))))),
                           DLNOT(integer_pred_re?(val(x))))) :))

>> |
```

Summary

- **$\mathbf{dTL}^2$:** Differential Temporal Dynamic Logic for hybrid programs
- **PVS:** Interactive theorem prover

Result: Embedding of $\mathbf{dTL}^2$ in PVS

- Formal verification of $\mathbf{dTL}^2$
- Fully operational embedding in **PVS**



Thank you for listening!