EPiC Series in Computer Science

Volume 37, 2015, Pages 11–22

Symbolic and Numerical Methods for Reachability Analysis, 1st International Workshop, SNR 2015

Abstract Interpretation with Infinitesimals*

Towards Scalability in *Nonstandard Static Analysis*
(Extended Abstract)

Kengo Kido[1,2], Swarat Chaudhuri[3], and Ichiro Hasuo[1]

[1] University of Tokyo, Japan
[2] JSPS Research Fellow
[3] Rice University, USA

Abstract

Towards the goal of correctness and reliability of hybrid systems, we continue our *nonstandard static analysis* program (with Suenaga and Sekine) where hybrid dynamics is turned into purely discrete one with explicit use of *infinitesimals*. While our previous results have focused on deductive verification by program logics, the current work aims at automation and enhanced scalability by extending *abstract interpretation*—a technique known for its ample scalability and widespread use in various verification tools—with infinitesimals. Our theoretical results include soundness and termination via *uniform* widening operators; and our prototype implementation successfully verifies some benchmark examples.

1 Introduction, or rather: Examples, Implementation and Experiments

The current abstract reports our work-in-progress, summarizing our preprint [13] (that is planned to be revised thoroughly).

Our contributions include a prototype tool that overapproximates reachable regions of hybrid systems—it is based on the nonstandard abstract interpretation framework presented in §3. We start out with describing the implementation.

We assume that a hybrid system in question is modeled in the language WHILE$^{\mathtt{dt}}$—a modeling language for hybrid systems introduced in [15]. It is an imperative language augmented with (a constant for) an infinitely small positive (*infinitesimal*) constant `dt`; this enables us to model continuous/hybrid dynamics with while loops, not with explicit ODEs. Our framework benefits from such WHILE$^{\mathtt{dt}}$ modeling in that, in order to adapt abstract interpretation to hybrid applications, there is no need to extend its theory for dealing with ODEs—in fact the theory *transfers* almost literally.

The way our tool works is the same as a (standard) abstract interpreter. Its input consists of: 1) a WHILE$^{\mathtt{dt}}$ program; 2) an initial state; and 3) an extrapolation threshold as well as the timing of widening

*Thanks are due to Kohei Suenaga and the anonymous referees for the preprint [13] (on the occasion of its submission to CAV 2015), for useful discussions and comments. The authors are supported by Grants-in-Aid for Young Scientists (A) No. 24680001, JSPS; and K.K. is supported by Grant-in-Aid for JSPS Fellows.

delays (two common parameters for abstract interpretation; the latter can be given interactively). Its output is a convex polyhedron that over-approximates the set of reachable memory states. The tool (as an abstract interpreter) currently supports: the domain of convex polyhedra as an abstract domain; and ∇_M, the "widening up to" from [10, 11] as a widening operator.

Our implementation consists principally of the following two components: 1) an OCaml frontend for parsing and forming an iteration sequence (that approximates loop semantics); and 2) a backend that executes operations on convex polyhedra, implemented in the computer algebra system (CAS) *Mathematica*. The two components are interconnected by a C++ program, via MathLink. It enables us to deal with the constant `dt` symbolically as a truly infinitesimal number.

Experiments. We analyzed two benchmark hybrid systems—water-level monitor and thermostat—expressed as the WHILE$^{\mathtt{dt}}$ programs in Fig. 1. They are common hybrid system examples; see e.g. [1]. The experiments were on Apple MacBook Pro with 2.6 GHz Dual-core Intel Core i5 CPU and 8 GB memory.

As the outcome (i.e. over-approximation of the reachable regions), we obtain: $1-\mathtt{dt} \leq x \leq 12+\mathtt{dt}$ for the water-level monitor, in 34.051 sec.; and $18 - 54 * \mathtt{dt} \leq x \leq 22 + 24 * \mathtt{dt}$ for the thermostat, in 6.692 sec. (Each execution time is the average of ten runs.) Note that these outputs essentially mean $1 \leq x \leq 12$ and $18 \leq x \leq 22$, respectively, since `dt` denotes an infinitesimal number. These reachable regions are exactly what one would expect.

```
(*Water-Level Monitor*)
l := 0; x := 1; p := 1; s := 0;
while true do {
   if p = 1 then x := x + dt
      else x := x - 2 * dt;
   if (x <= 5 && p = 0) then s := 1
      else {if (x >= 10 && p = 1)
               then s := 1
               else s := 0
      };
   if s = 1 then l := l + dt
      else skip;
   if s = 1 && l >= 2
      then {p := 1 - p; s := 0; l := 0}
      else skip
};;
```

```
(*Thermostat*)
x := 22; p := 0;
while true do {
   if p = 0 then x := x - 3 * x * dt
      else x := x + 3 * (30 - x) * dt;
   if x >= 22 then p := 0
      else {if x <= 18 then p := 1
               else skip
      }
};;
```

Figure 1: Example WHILE$^{\mathtt{dt}}$ code

Overview. In the rest of the current abstract we describe the theory behind this tool—the theoretical framework of *abstract interpretation with infinitesimals*. The principle of our approach is to use `dt` to *turn continuous dynamics into discrete ones* and then to rely on the logical/model-theoretical infrastructure of nonstandard analysis (NSA) [8, 14], most notably on the celebrated result of the *transfer principle*. Accordingly, most of the subsequent descriptions are devoted to the formalization of abstract interpretation in a first-order logic, and how it allows "transfer" by the transfer principle.

The process of transfer is mostly straightforward—except for *widening* operators, an indispensable component in the framework of abstract interpretation [4]. We therefore introduce the notion of *uniformity* of widening, and identify those which are uniform. Among them is the operator ∇_M used in our implementation.

Further details of the results reported in the current abstract are found in the preprint [13].

2 Preliminaries

2.1 Nonstandard Modeling of Hybrid Systems

WHILE$^{\mathtt{dt}}$ is a modeling language for hybrid systems introduced in [15]. It is an extension of a usual imperative language, with a constant `dt` that represents an *infinitesimal*—a positive value that is smaller than any positive real.

In WHILE$^{\mathtt{dt}}$, we model continuous flow in hybrid systems as if it were infinitely many infinitesimal jumps, dispensing with explicit use of differential equations. An example is shown on the right, where t is understood to grow from 0 to 1 in a continuous manner. The usual Hoare-style program logic is just as valid in this extension, leading to an automatic precondition generator in [12].

$$\begin{array}{l} t := 0; \\ \texttt{while}\ (t \leq 1) \\ \quad t := t + \mathtt{dt} \end{array} \quad (\dagger)$$

2.2 Nonstandard Analysis and Semantics of WHILE$^{\mathtt{dt}}$

The formal semantics of WHILE$^{\mathtt{dt}}$ is given using Robinson's *nonstandard analysis (NSA)*, a framework that supports use of infinitesimals in a mathematically rigorous manner (see e.g. [8] for details). There *hyperreals*—including (standard) reals, infinitesimals, and infinites as their multiplicative inverses—are (equivalence classes of) infinite sequences of real numbers. For example, the hyperreal $\omega^{-1} = \left[(\mathbf{1}, \frac{\mathbf{1}}{\mathbf{2}}, \frac{\mathbf{1}}{\mathbf{3}}, \dots)\right]$ is infinitesimal and is used as the denotation $[\![\mathtt{dt}]\!]$ of the constant `dt`. The set of all hyperreals are denoted by ${}^*\mathbb{R}$.

The semantics of the program $(\dagger)$ is then defined as follows. We consider the *i-th section* of the execution of the program, for each $i \in \mathbb{N}$:

$$\begin{array}{l} t := 0; \\ \texttt{while}\ (t \leq 1) \\ \quad t := t + \mathbf{1} \end{array} \quad \begin{array}{l} t := 0; \\ \texttt{while}\ (t \leq 1) \\ \quad t := t + \frac{\mathbf{1}}{\mathbf{2}} \end{array} \quad \cdots \quad \begin{array}{l} t := 0; \\ \texttt{while}\ (t \leq 1) \\ \quad t := t + \frac{\mathbf{1}}{\mathbf{i+1}} \end{array} \quad \cdots \qquad (*)$$

Each section is a `dt`-free program with obvious semantics. The values of t after the execution of each section are then bundled to obtain $\left[(1 + \mathbf{1}, 1 + \frac{\mathbf{1}}{\mathbf{2}}, 1 + \frac{\mathbf{1}}{\mathbf{3}}, \dots)\right]$, a hyperreal that is infinitely close to 1.

2.3 Transfer Principle

The *transfer principle* is one of the most important results in NSA. Roughly it states that a first-order logical formula in $\mathbb{R}$ is valid if and only if essentially the same formula in ${}^*\mathbb{R}$ (we call it **-transform* of the formula) is valid. We do not go into the details of the first-order language that allows transfer (see e.g. [8] for details). Nevertheless we note that the language is quite a rich one—like the one for set theory, with $\in$ being a binary predicate—one that is rich enough to allow for the theory of abstract interpretation to be expressed within it.

2.4 Abstract Interpretation

Abstract interpretation [7] is a well-established technique in static analysis. We present a minimal set of definitions and propositions that will be transferred to the nonstandard setting in §3, assuming the reader's familiarity with the theory. See, e.g. [4], for details.

Galois connection is a notion that underlies abstract interpretation. Once we have a Galois connection between two domains, we have the following proposition. In the proposition, the *least fixed point relative to* $\perp$, denoted by $\mathrm{lfp}_{\perp} F$, is the least among the fixed points of F above $\perp$; by L's cpo structure and F's continuity it is given by $\bigsqcup_{n\in\mathbb{N}} F^n \perp$.

Proposition 1. *Let $L \underset{\gamma}{\overset{\alpha}{\rightleftarrows}} \overline{L}$ be a Galois connection such that L is additionally a cpo; $F\colon L \to L$ be a continuous function; $\overline{F} : \overline{L} \to \overline{L}$ be a monotone function such that $F \sqsubseteq \vec{\gamma}(\overline{F})$; $\bot \in L$ be an element such that $\bot \sqsubseteq F(\bot)$. Assume that $\overline{x} \in \overline{L}$ is a prefixed point of $\overline{F}$ (i.e. $\overline{F}(\overline{x}) \mathrel{\overline{\sqsubseteq}} \overline{x}$) such that $\alpha(\bot) \mathrel{\overline{\sqsubseteq}} \overline{x}$.*

Then $\overline{x}$ over-approximates $\mathrm{lfp}_{\bot} F$, that is, $\mathrm{lfp}_{\bot} F \sqsubseteq \gamma(\overline{x})$. □

Prop. 1 enables us to adopt a typical abstract interpretation workflow. That is, given a concrete domain L and a loop that is interpreted over L, we 1) find a suitable abstract domain $\overline{L}$; 2) interpret the loop's body as a suitable function $\overline{F}$ on $\overline{L}$; and 3) find a suitable prefixed point $\overline{x}$ of $\overline{F}$.

For the first task among the above three, not a small number of options have been proposed in the literature; among them are the *interval domain* $\mathsf{Intv}_{\mathbb{R}}$ over $\mathbb{R}$, and the *domain of convex polyhedra* $\mathbb{CP}_n$, that are used with the concrete domain $\mathcal{P}(\mathbb{R}^n)$ for reachability analysis. We will be using them later.

Once an abstract domain $\overline{L}$ is fixed, very often there is standard interpretation of (the loop-free fragment of) an imperative language on $\overline{L}$; this achieves the second task. For the last task (i.e. finding a prefixed point), the following notion of *widening* is used (together with *narrowing* that we will not be using). We recall its definition, since its certain detail becomes relevant to our technical developments in §3.

Definition 2 (widening operator)**.** Let $(L, \sqsubseteq)$ be a poset. A function $\nabla : L \times L \to L$ is said to be a *widening operator* if the following two conditions hold.

- (*Covering*) For any $x, y \in L$, $x \sqsubseteq x \nabla y$ and $y \sqsubseteq x \nabla y$.
- (*Termination*) For any ascending chain $\langle x_i \rangle \in L^{\mathbb{N}}$, the chain $\langle y_i \rangle \in L^{\mathbb{N}}$ defined by $y_0 = x_0$ and $y_{i+1} = y_i \nabla x_{i+1}$ (for each $i \in \mathbb{N}$) is ultimately stationary.

The use of widening is as below: the covering condition ensures that the outcome is a prefixed point; and the procedure terminates thanks to the termination condition.

Proposition 3 (convergence of iteration sequences)**.** *Let $(L, \sqsubseteq)$ be a poset; $F : L \to L$ be a monotone function; $\bot \in L$ be such that $\bot \sqsubseteq F(\bot)$; $\nabla : L \times L \to L$ be a widening operator; and $\langle X_i \rangle_{i \in \mathbb{N}} \in L^{\mathbb{N}}$ be the infinite sequence defined by*

$$X_0 = \bot \ ; \qquad \text{and, for each } i \in \mathbb{N}, \quad X_{i+1} = \begin{cases} X_i & (\text{if } F(X_i) \sqsubseteq X_i) \\ X_i \nabla F(X_i) & (otherwise) \end{cases}$$

Then the sequence $\langle X_i \rangle_{i \in \mathbb{N}}$ is increasing and ultimately stationary; moreover its limit $\bigsqcup_{i \in \mathbb{N}} X_n$ is a prefixed point of F such that $\bot \sqsubseteq \bigsqcup_{i \in \mathbb{N}} X_n$. □

Note that a widening operator on a fixed abstract domain $\overline{L}$ is not at all unique.

3 Abstract Interpretation Augmented with Infinitesimals

We now present an abstract interpretation framework for the analysis of WHILE$^{\mathsf{dt}}$ programs—and hence the hybrid dynamics modeled thereby. Details are in [13].

3.1 Theory of Hyper Abstract Interpretation

We can easily see that the abstract domains $\mathsf{Intv}_{\mathbb{R}}$ and $\mathbb{CP}_n$ can be transferred and they are in Galois connections with $\mathcal{P}(^*\mathbb{R})$, where $^*\mathbb{R}$ is the set of all hyperreals (see §2.2). The transferred abstract domains can be seen as the interval domain and the domain of convex polyhedra over $^*\mathbb{R}$, respectively.

This, however, does not suffice for our purpose of abstract interpretation of WHILE$^{\mathtt{dt}}$ programs. Let c be the command `x:=x+dt` and the collecting semantics of the loop whose body is c (assume x is initially 0). The latter's inductive approximation does not saturate after ω steps—the collecting semantics grows from $[0,1]$ to $[0,1+\mathtt{dt}]$, from the ω-th step to the $(\omega+1)$-th. This makes the basic assumption of the framework in §2.4, namely that $\mathrm{lfp}_{\bot} F = \bigsqcup_{n\in\omega} F^n \bot$, fail.

Fortunately it turns out that we can rely on the $*$-*transform* (§2.3) of the theory in §2.4, where it suffices to impose a different condition of *hyperdomain* structure on the domain and $*$-*continuity* on the function $[\![c]\!]$—instead of the cpo structure and the (standard) continuity. This theoretical framework of *hyper abstract interpretation*, which we shall describe here, is an extension of the *transferred domain theory* studied in [3, 16]. For further details of the NSA notions used (such as that of internal entity), see e.g. [8].

Here is the counterpart of Prop. 1. As announced, it only requires L's cpo structure (which is equivalent to ${}^*L$'s hyperdomain structure) and F's $*$-continuity.

Theorem 4. *Let $(L,\sqsubseteq)$ be a cpo, $(\overline{L},\overline{\sqsubseteq})$ be a poset such that $L \underset{\gamma}{\overset{\alpha}{\rightleftarrows}} \overline{L}$, and consider its $*$-transform ${}^*L \underset{{}^*\gamma}{\overset{{}^*\alpha}{\rightleftarrows}} {}^*\overline{L}$. Let $F : {}^*L \to {}^*L$ be a $*$-continuous function; $\bot \in {}^*L$ be such that $\bot \,{}^*\!\sqsubseteq F(\bot)$, and $\overline{F} : {}^*\overline{L} \to {}^*\overline{L}$ be an internal function that is monotone with respect to ${}^*\overline{\sqsubseteq}$. Assume that $F \,{}^*\!\sqsubseteq (*\vec{\gamma})(\overline{F})$; and that $\overline{x} \in {}^*\overline{L}$ is a prefixed point of $\overline{F}$, i.e. $\overline{F}(\overline{x}) \,{}^*\overline{\sqsubseteq}\, \overline{x}$.*

Then $\overline{x}$ over-approximates $\mathrm{lfp}_{\bot} F$, that is, $\mathrm{lfp}_{\bot} F \,{}^*\!\sqsubseteq\, {}^*\gamma(\overline{x})$. □

We shall use Thm. 4 in over-approximating loop semantics in WHILE$^{\mathtt{dt}}$. Technically some care is needed here regarding the difference between ${}^*(\mathcal{P}(\mathbb{R}^n))$ and $\mathcal{P}({}^*\mathbb{R}^n)$, but we skip the details.

Our goal is over-approximation of the semantics of iteration of a loop-free WHILE$^{\mathtt{dt}}$ program c, relying on Thm. 4. Towards the goal, the next step is to find a suitable $\overline{F} : {}^*\overline{L} \to {}^*\overline{L}$ that "stepwise approximates" $F = [\![c]\!]$, the collecting semantics of c. The next result implies that the $*$-transformation of the usual semantics used in standard abstract interpretation for (loop-free) programs can be used in such $\overline{F}$.

Proposition 5. *Let $L \underset{\gamma}{\overset{\alpha}{\rightleftarrows}} \overline{L}$ be a Galois connection. Assume that $F : L \to L$ is stepwise approximated by $\overline{F} : \overline{L} \to \overline{L}$, that is, $F \sqsubseteq \vec{\gamma}(\overline{F})$. Then the (internal) function ${}^*F : {}^*L \to {}^*L$ is over-approximated by ${}^*\overline{F} : {}^*\overline{L} \to {}^*\overline{L}$, i.e. ${}^*F \,{}^*\!\sqsubseteq\, (*\vec{\gamma})(*\overline{F})$.* □

Remark 6. It is known in the community that convex polyhedra fail to form a Galois connection (in the above precise sense), and one should alternatively use a concretization-only foundation of abstract interpretation [5]. Doing so in our current nonstandard setting is not hard.

3.2 Hyperwidening and Uniform Widening Operators

Towards our goal of using Thm. 4, the last remaining step is to find a prefixed point $\overline{x}$, i.e. $\overline{F}(\overline{x}) \,{}^*\overline{\sqsubseteq}\, \overline{x}$. This is where widening operators are standardly used; see §2.4.

We can try $*$-transforming a (standard) notion—a strategy that we have used repeatedly in the current section. This yields the following result, that has a problem that is discussed shortly.

Theorem 7. *Let $(L,\sqsubseteq)$ be a poset and $\nabla : L\times L \to L$ be a widening operator on L. Let $F : {}^*L \to {}^*L$ be a monotone and internal function; and $\bot \in {}^*L$ be such that $\bot \,{}^*\!\sqsubseteq F(\bot)$. The iteration* hyper-*sequence $\langle X_{i\in{}^*\mathbb{N}}\rangle$—indexed by hypernaturals $i \in {}^*\mathbb{N}$—that is defined by*

$$X_0 = \bot, \quad X_{i+1} = \begin{cases} X_i & (\mathit{if}\ F(X_i)\ {}^*\!\sqsubseteq\ X_i) \\ X_i\, {}^*\!\nabla F(X_i) & (\mathit{otherwise}) \end{cases} \text{ for all } i \in {}^*\mathbb{N}$$

reaches its limit within some hypernatural number of steps and the limit $\bigsqcup_{i\in{}^*\mathbb{N}} X_i$ *is a prefixed point of* F *such that* $\bot\ {}^*\sqsubseteq \bigsqcup_{i\in{}^*\mathbb{N}} X_i$. □

The problem of Thm. 7 is that the *finite-step convergence* of iteration sequences for the original widening operator is now transferred to *hyperfinite-step convergence*. This is not desired. All the entities from NSA that we have used so far are constructs in denotational semantics—whose only role is to ensure soundness of verification methodologies[1] and on which we never actually operate—and therefore their infinite/infinitesimal nature has not been a problem. In contrast, the iteration hypersequence $\langle X_{i\in{}^*\mathbb{N}}\rangle$ is what we actually compute to over-approximate program semantics; and therefore its termination guarantee within $i \in {}^*\mathbb{N}$ steps (Thm. 7) is of no use.

As a remedy we introduce *uniformity* of (standard) widening operators. It strengthens the original termination condition (Def. 2) by imposing a uniform bound i for stability of arbitrary chains $\langle x_i\rangle \in L^{\mathbb{N}}$. Logically the change means replacing $\forall\exists$ by $\exists\forall$.

Definition 8 (uniform widening)**.** Let $(L, \sqsubseteq)$ be a poset. A function $\nabla : L \times L \to L$ is said to be a *uniform widening operator* if the following two conditions hold.

- (Covering) For any $x, y \in L$, $x \sqsubseteq x\nabla y$ and $y \sqsubseteq x\nabla y$.
- (*Uniform termination*) Let $x_0 \in L$. There exists a *uniform bound* $i \in \mathbb{N}$ (i depends on x_0) such that: for any ascending chain $\langle x_k\rangle \in L^{\mathbb{N}}$ starting from x_0, there exists $j \le i$ at which the chain $\langle y_k\rangle \in L^{\mathbb{N}}$, defined as follows, stabilizes (i.e. $y_j = y_{j+1}$).

$$y_0 = x_0, \qquad y_{k+1} = y_k \nabla x_{k+1} \quad \text{for all } k \in \mathbb{N}$$

It is straightforward that uniform termination implies termination. The following theorem is a "practical" improvement of Thm. 7; its proof relies on instantiating the uniform bound i in a suitable $\mathscr{L}_{\mathbb{R}}$-formula with a Skolem constant, before transfer.

Theorem 9. *Let* $(L, \sqsubseteq)$ *be a poset and* $\nabla \in L \times L \to L$ *be a uniform widening operator on* L*. Let* $F : {}^*L \to {}^*L$ *be a monotone and internal function; and* $\bot \in L$ *be such that* ${}^*\bot\ {}^*\sqsubseteq F({}^*\bot)$*. The iteration sequence* $\langle X_i\rangle_{i\in\mathbb{N}}$ *defined by*

$$X_0 = {}^*\bot, \quad X_{i+1} = \begin{cases} X_i & (if\ F(X_i)\ {}^*\sqsubseteq X_i) \\ X_i\ {}^*\nabla\ F(X_i) & (otherwise) \end{cases} \quad \text{for all } i \in \mathbb{N}$$

reaches its limit within some finite number of steps; and the limit $\bigsqcup_{i\in\mathbb{N}} X_i$ *is a prefixed point of* F *such that* ${}^*\bot\ {}^*\sqsubseteq \bigsqcup_{i\in\mathbb{N}} X_i$. □

Note that uniformity of ∇ is a *sufficient condition* for the termination of nonstandard iteration sequences (by ${}^*\nabla$); Thm. 9 does not prohibit other useful widening operators in the nonstandard setting. Furthermore, there can be a useful (nonstandard) widening operator except for the ones ${}^*\nabla$ that arise via standard ones ∇.

We investigate uniformity of the widening operators for the interval domain and the domain of convex polyhedra listed below. The definitions are omitted except for one widening operator for the interval domain.

- ∇_{Intv}, a well-known widening operator for the interval domain, introduced in [4];
- $\nabla_{\mathsf{Intv}_{\mathbb{Z},c}}$, a widening operator for the interval domain (over $\mathbb{Z}$) defined as follows:

$$[l_1, u_1]\nabla_{\mathsf{Intv}_{\mathbb{Z},c}}[l_2, u_2] := \big[\ \text{if } \min(l_1, l_2) > -c \text{ then } \min(l_1, l_2) \text{ else } -\infty\,,\\ \text{if } \max(u_1, u_2) < c \text{ then } \max(l_1, l_2) \text{ else } \infty\ \big],$$

  where $c \in \mathbb{Z}$ is fixed;

[1] Recall that WHILE$^{\mathrm{dt}}$ is a *modeling* language and we do not execute them

- ∇_S, the *standard widening* operator for $\mathbb{CP}_n$ introduced in [7, 9];
- ∇_M, the widening operator *up to* M introduced in [10, 11] for $\mathbb{CP}_n$; and
- ∇_N, the *precise widening* operator for $\mathbb{CP}_n$ introduced in [2].

Note that $\nabla_{\mathsf{Intv}_{\mathbb{Z},c}}$ is different from a common widening operator called *interval widening with thresholds*; $\nabla_{\mathsf{Intv}_{\mathbb{Z},c}}$ is there as a non-uniform example.

Theorem 10. *Uniformity of five widening operators presented above are listed in the table below. Among them three are uniform; two are not.*

interval domain Intv		*convex polyhedra* $\mathbb{CP}_n$		
∇_{Intv} *([4])*	$\nabla_{\mathsf{Intv}_{\mathbb{Z},c}}$	∇_S *([7, 9])*	∇_M *([10, 11])*	∇_N *([2])*
✓	×	✓	✓	×

□

To conclude: Thm. 4, 9 and 10 altogether ensure the soundness and termination of our algorithm implemented in the tool (§1) that relies on the abstract domain of convex polyhedra and the widening operator ∇_M.

4 Example of Analysis

Finally we showcase how our framework analyzes hybrid systems. Water-level monitor shown in Fig. 1 is employed as an example hybrid system. In this example, x is the water level, l is the counter for the time lag, p is the pump switch and s is the signal from the sensor.

The verification goal is to see that the level x stays within 1 cm and 12 cm ($1 \leq x \leq 12$). We classify the variables into continuous/numerical ones (l, x) and discrete ones (p, s). Accordingly we have:

- $({}^*\mathbb{R}^2)^4$ as a concrete domain, corresponding to two continuous values (${}^*\mathbb{R}^2$) for each of $(p, s) \in \{(0, 0), \ldots, (1, 1)\}$; and
- $({}^*\mathbb{CP}_2)^4$ as an abstract domain, where $\mathbb{CP}_2$ is the domain of convex polyhedra on $\mathbb{R}^2$.

In fixed-point computation we use ∇_M, the widening operator up to M (see, e.g., [10] for more details). The set M of linear constraints is taken from the program: $M = \{x \leq 5, x \geq 5, x \leq 10, x \geq 10, l \leq 2, l \geq 2\}$. As is common in abstract interpretation (see e.g. [6]), we add delays of widening by an *extrapolation threshold*.

As preparation, we represent the WHILE$^{\mathtt{dt}}$ program shown in Fig. 1 that expresses the behavior of a water-level monitor, as the control flow graph in Fig. 2. The iteration sequence we obtain in the analysis is presented in Fig. 3 to Fig. 6. The value of `dt` is fixed to be 0.6 in these graphs just for maintaining the appearance of them.

Then we show how we obtain the iteration sequence presented in Fig. 3 to Fig. 6. Henceforth the word "state" means an abstracted hyperstate. The graph legend shows the program point. For example, the topmost graph in Fig. 3 shows the state on $p = 1 \wedge s = 0$ from the program point A to B in Fig. 2 in the first iteration ("A00-B00"). The topmost graph in Fig. 4 shows the state on $p = 1 \wedge s = 1$ at the program point D in Fig. 2 in the third iteration ("D02"). When the graph is not given for a program point, the state at the program point is the same as that in the previous iteration. For example, the state on $p = 1 \wedge s = 0$ at E in the fifth iteration is the same as the graph whose legend is "D02-F02" in Fig. 3. The program point where we choose not to widen by extrapolation threshold is explicitly presented by adding "Do not widen" to the graph legend.

We denote the state at the program point, e.g. "A01" by A_1. A_k is $B_{k-1} \sqcup F_{k-1}$, the convex hull of B_{k-1} and F_{k-1}. B_k is A_k if we chose "do not widen", or $B_{k-1} \nabla_M A_k$ otherwise. C_k, D_k, E_k, F_k can be computed in a obvious manner.

Figure 2: Control flow graph of water-level monitor

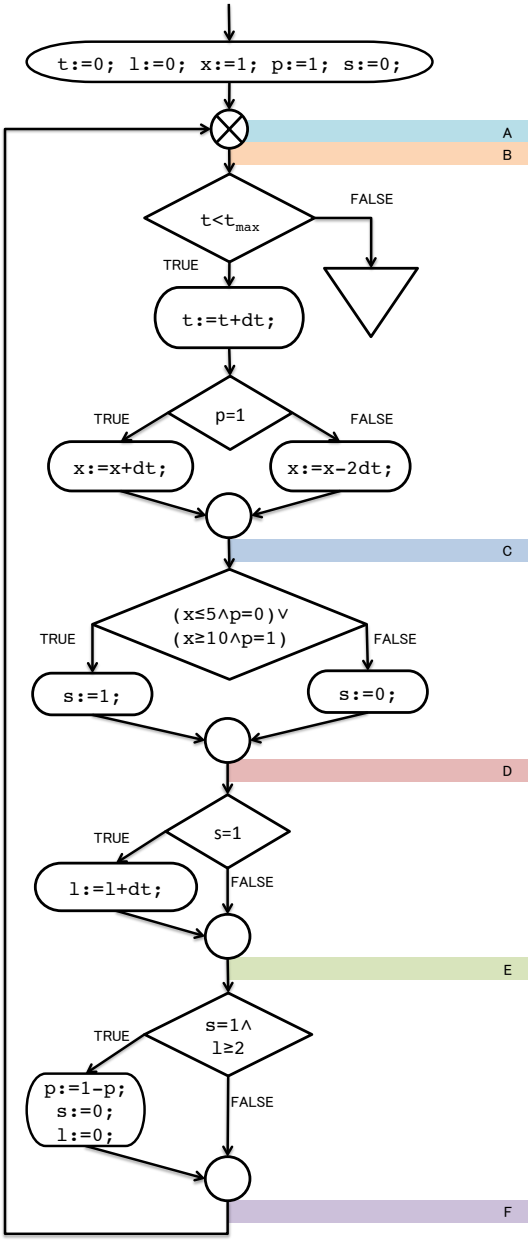


Figure 3: p=1 and s=0

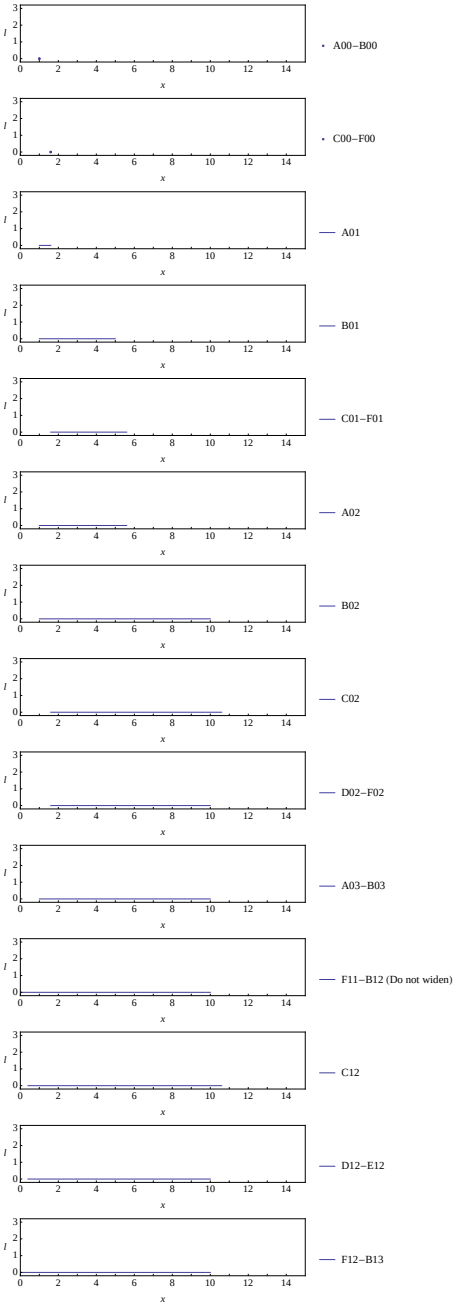


Figure 4: p=1 and s=1

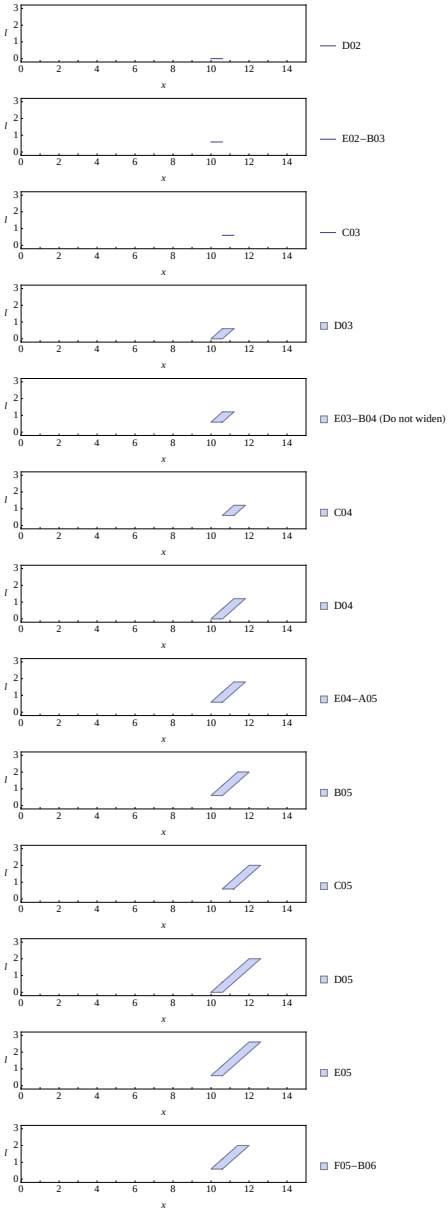


Figure 5: p=0 and s=0

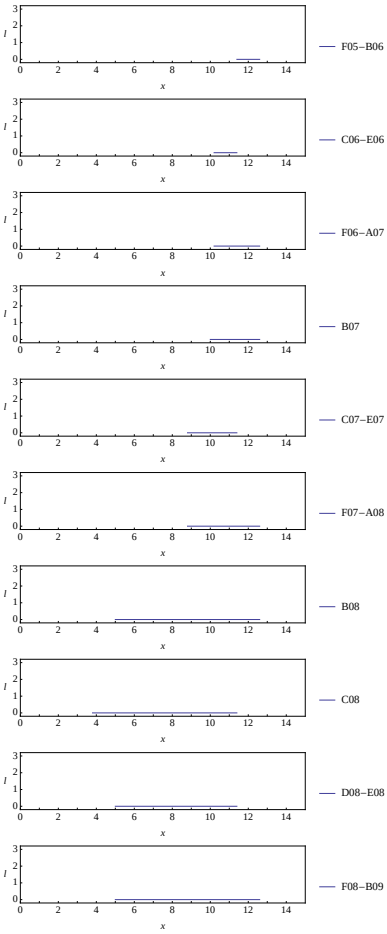
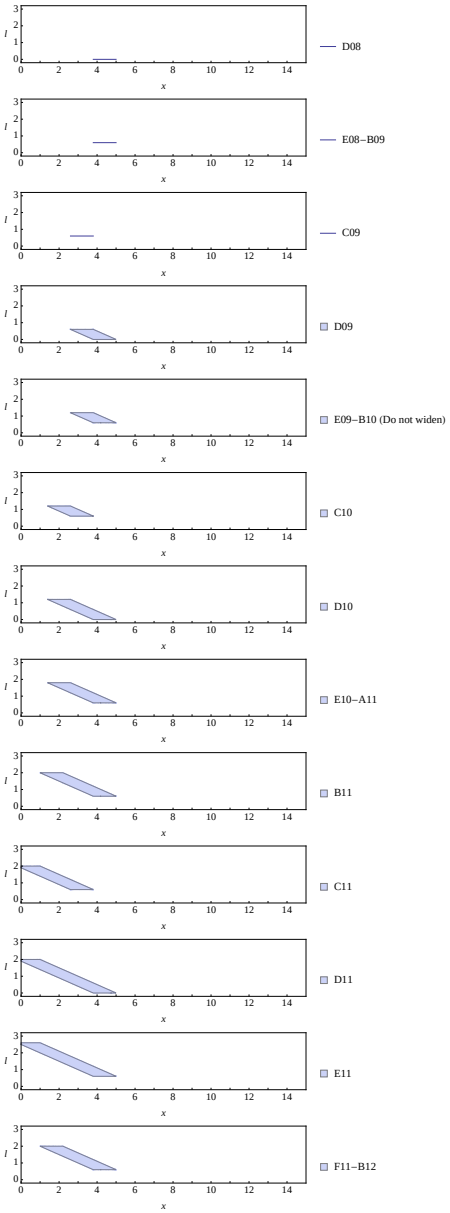


Figure 6: p=0 and s=1



When starting static analysis, we set the states at all program points to $\bot$ (the empty convex polyhedron). At first, the assignments in the entry node are evaluated and $A_0 = \mathsf{con}\{p = 1, s = 0, x = 1, l = 0\}$ where con is a function that maps a set of inequalities to its corresponding convex polyhedra. Then we compute the states as follows:

$$
\begin{aligned}
&B_0 = \bot \nabla_M A_0 = A_0 \\
&C_0 = \mathsf{con}\{p = 1, s = 0, x = 1 + \mathtt{dt}, l = 0\} \text{ by applying } x := x + \mathtt{dt} \text{ because } p = 1 \\
&D_0 = C_0 \text{ because } (x \leq 5 \wedge p = 0) \vee (x \geq 10 \wedge p = 1) \text{ is interpreted as ff at "C00"} \\
&\qquad\qquad \text{and } s \text{ is already assigned to } 0 \\
&E_0 = F_0 = D_0 \text{ because } s = 0 \\
&A_1 = B_0 \sqcup F_0 = \mathsf{con}\{p = 1, s = 0, 1 \leq x \leq 1 + \mathtt{dt}, l = 0\} \\
&B_1 = B_0 \nabla_M A_1 \\
&\quad = \mathsf{con}\{p = 1, s = 0, x = 1, l = 0\} \nabla_M \mathsf{con}\{p = 1, s = 0, 1 \leq x \leq 1 + \mathtt{dt}, l = 0\} \\
&\quad = \mathsf{con}\{p = 1, s = 0, 1 \leq x \leq 5, l = 0\}
\end{aligned}
$$

Note that $M = \{x \leq 5, x \geq 5, x \leq 10, x \geq 10, l \leq 2, l \geq 2\}$. We can iteratively compute in this way and at "B13", we reach a prefixed point and soundly approximate the least fixed point. The result means that the level of the water x always satisfies $1 - \mathtt{dt} \leq x \leq 12 + \mathtt{dt}$. This is what we wanted modulo infinitesimal gaps.

References

[1] Rajeev Alur, Costas Courcoubetis, Thomas A. Henzinger, and Pei-Hsin Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In *Hybrid Systems*, pages 209–229, 1992.

[2] Roberto Bagnara, Patricia M. Hill, Elisa Ricci, and Enea Zaffanella. Precise widening operators for convex polyhedra. *Sci. Comput. Program.*, 58(1-2):28–56, 2005.

[3] Romain Beauxis and Samuel Mimram. A non-standard semantics for Kahn networks in continuous time. In *CSL*, pages 35–50, 2011.

[4] Patrick Cousot and Radhia Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the Fourth ACM Symposium on Principles of Programming Languages, Los Angeles, California, USA, January 1977*, pages 238–252, 1977.

[5] Patrick Cousot and Radhia Cousot. Abstract interpretation frameworks. *J. Log. Comput.*, 2(4):511–547, 1992.

[6] Patrick Cousot and Radhia Cousot. Comparing the galois connection and widening/narrowing approaches to abstract interpretation. In *Programming Language Implementation and Logic Programming, 4th International Symposium, PLILP'92, Leuven, Belgium, August 26-28, 1992, Proceedings*, pages 269–295, 1992.

[7] Patrick Cousot and Nicolas Halbwachs. Automatic discovery of linear restraints among variables of a program. In *Conference Record of the Fifth Annual ACM Symposium on Principles of Programming Languages, Tucson, Arizona, USA, January 1978*, pages 84–96, 1978.

[8] R. Goldblatt. *Lectures on the Hyperreals: An Introduction to Nonstandard Analysis*. Graduate Texts in Mathematics. Springer New York, 1998.

[9] Nicolas Halbwachs. *Détermination automatique de relations linéaires vérifiées par les variables d'un programme*. Thèse de 3e cycle, Université Scientifique et Médicale de Grenoble, 1979.

[10] Nicolas Halbwachs. Delay analysis in synchronous programs. In *Computer Aided Verification, 5th International Conference, CAV '93, Elounda, Greece, June 28 - July 1, 1993, Proceedings*, pages 333–346, 1993.

[11] Nicolas Halbwachs, Yann-Erick Proy, and Patrick Roumanoff. Verification of real-time systems using linear relation analysis. *Formal Methods in System Design*, 11(2):157–185, 1997.

[12] Ichiro Hasuo and Kohei Suenaga. Exercises in nonstandard static analysis of hybrid systems. In *Computer Aided Verification - 24th International Conference, CAV 2012, Berkeley, CA, USA, July 7-13, 2012 Proceedings*, pages 462–478, 2012.

[13] Kengo Kido, Swarat Chaudhuri, and Ichiro Hasuo. Abstract interpretation with infinitesimals: Towards scalability in *Nonstandard Static Analysis*. Preprint, available at: `http://www-mmm.is.s.u-tokyo.ac.jp/~kkido`, 2015.

[14] A. Robinson. *Non-standard Analysis*. Studies in logic and the foundations of mathematics. North-Holland Pub. Co., 1966.

[15] Kohei Suenaga and Ichiro Hasuo. Programming with infinitesimals: A while-language for hybrid system modeling. In *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part II*, pages 392–403, 2011.

[16] Kohei Suenaga, Hiroyoshi Sekine, and Ichiro Hasuo. Hyperstream processing systems: nonstandard modeling of continuous-time signals. In *The 40th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '13, Rome, Italy - January 23 - 25, 2013*, pages 417–430, 2013.