Abstract Interpretation with Infinitesimals

Towards Scalability in *Nonstandard Static Analysis*

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

[1] University of Tokyo, Japan
[2] Rice University, USA

Abstract. Towards the goal of correctness and reliability of hybrid systems, we continue the *nonstandard static analysis* program 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

Hybrid systems exhibit both discrete *jump* and continuous *flow* dynamics. Quality assurance of such systems are of paramount importance due to the current ubiquity of *cyber-physical systems (CPS)* like cars, airplanes, and many others. For the formal verification approach to hybrid systems, the challenges are: 1) to incorporate flow-dynamics; and 2) to do so at the lowest possible cost, so that the discrete framework smoothly transfers to hybrid situations. A large body of existing work uses *differential equations* explicitly in the syntax; see the discussion of related work below.

In [17], instead, an alternative approach of *nonstandard static analysis*—combining *static analysis* and *nonstandard analysis*—is proposed. Its basic idea is to introduce a constant `dt` for an *infinitesimal* (i.e. infinitely small) value, and *turn flow into jump*. With `dt`, the continuous operation of integration can be represented by a while-loop, to which existing discrete techniques such as Hoare-style program logics readily apply. For a rigorous mathematical development we employed *nonstandard analysis (NSA)* beautifully formalized by Robinson [16].

Concretely, in [17] we took the common combination of a WHILE-language and a Hoare logic (e.g. in the textbook [19]); and added a constant `dt` to obtain a modeling and verification framework for hybrid systems. Its components are called WHILE$^{\mathtt{dt}}$ and HOARE$^{\mathtt{dt}}$. The soundness of HOARE$^{\mathtt{dt}}$ is proved against denotational semantics defined in the language of NSA. Subsequently in the *nonstandard static analysis* program: in [13] we presented a prototype automatic theorem prover for HOARE$^{\mathtt{dt}}$; and in [18] we applied the same idea to stream processing systems, realizing a verification framework for *signal processing* as in Simulink.

Underlying these technical developments is the idea of what we call *sectionwise execution*, illustrated by the following example.

Example 1.1 Let c_{elapse} be the program on the right. The value of `dt` is infinitesimal; therefore the `while` loop will not terminate within finitely many steps. Nevertheless it is somehow intuitive to expect that after an "execution" of this program, the value of t should be infinitesimally close to 1.

$$t := 0\ ;\quad \texttt{while}\ t \leq 1\ \texttt{do}\quad t := t + \texttt{dt}$$

One possible way of thinking is to imagine *sectionwise execution*. For each natural number i we consider the *i-th section* of the program c_{elapse}, denoted by $c_{\mathsf{elapse}}|_i$ and shown on the right. Concretely, $c_{\mathsf{elapse}}|_i$ is obtained by replacing the infinitesimal `dt` in c_{elapse} with $\frac{1}{i+1}$. Informally $c_{\mathsf{elapse}}|_i$ is the "i-th approximation" of the original c_{elapse}.

$$t := 0\ ;\quad \texttt{while}\ t \leq 1\ \texttt{do}\quad t := t + \tfrac{1}{i+1}$$

A section $c_{\mathsf{elapse}}|_i$ does terminate within finite steps and yields $1 + \frac{1}{i+1}$ as the value of t. Now we collect the outcomes of sectionwise executions and obtain a sequence

$$(\,1+1,\ 1+\tfrac{1}{2},\ 1+\tfrac{1}{3},\ \ldots,\ 1+\tfrac{1}{i},\ \ldots\,) \tag{1}$$

which is thought of as a progressive approximation of the actual outcome of the original program c_{elapse}. Indeed, in the language of NSA, the sequence (1) represents a *hyperreal number* r that is infinitesimally close to 1.

We note that a program in WHILE$^{\mathtt{dt}}$ is *not* executable in general: the program c_{elapse} executes infinitely many iterations. It is however an advantage of *static* approaches to verification, that programs need not be executed to prove their correctness. Instead, well-defined mathematical semantics suffices and supports deductive reasoning. This is what we do, with the denotational semantics of WHILE$^{\mathtt{dt}}$ exemplified in Example 1.1.

Our Contribution In the previous work [13,17,18] *invariant discovery* has been a big obstacle in scalability of the proposed verification techniques—as is usual in deductive verification. The current work, as a first step towards scalability of our approach, extends *abstract interpretation* [4] with infinitesimals. The abstract interpretation methodology is known for its ample applicability (it is employed in symbolic model checking as well as many deductive techniques) and scalability (the static analyzer Astrée [6] has been successfully used e.g. for Airbus's flight control system).

Our theoretical contribution includes: the theory of *hyper abstract interpretation* where well-known abstract domains, like the ones by intervals and convex polyhedra, are "$*$-transformed" to the abstract domains for hyperreals; their soundness in over-approximating semantics of WHILE$^{\mathtt{dt}}$ programs; and introduction of the notion of *uniform* widening operators. With the latter, inductive approximation is guaranteed to terminate within finitely many steps—even after extension to the nonstandard setting. We show that many known widening operators, if not all, are indeed uniform.

These theoretical results form a basis of our prototype implementation, that successfully analyzes: *water-level monitor*, a common example of piecewise-linear hybrid dynamics; and also *thermostat* that is beyond piecewise-linear.

Organization In §2 we review preliminaries on: nonstandard analysis; the modeling language WHILE$^{\mathtt{dt}}$ from [17]; and abstract interpretation. In §3 we extend the theory of

abstract interpretation with infinitesimals and build the theory of nonstandard abstract interpretation. An example of water-level monitor is analyzed in §4, using convex polyhedra and the widening ∇_M [10, 12]. There we present our prototype implementation and the experiment results, too.

2 Preliminaries

2.1 Nonstandard Analysis

Here we list a minimal set of necessary definitions and results in nonstandard analysis (NSA) [16]. Some details that will be needed in our proofs are deferred to Appendix A; further details of NSA are found e.g. in [8, 14].

Hyperreals We fix an *index set* $I = \mathbb{N}$, and an *ultrafilter* $\mathcal{F} \subseteq \mathcal{P}(I)$ that extends the cofinite filter $\mathcal{F}_{\mathrm{c}} := \{S \subseteq I \mid I \setminus S \text{ is finite}\}$. Its properties to be noted: 1) for any $S \subseteq I$, exactly one of S and $I \setminus S$ belongs to $\mathcal{F}$; 2) if S is *cofinite* (i.e. $I \setminus S$ is finite), then S belongs to $\mathcal{F}$.

Definition 2.1 (hyperreal $r \in {}^*\mathbb{R}$) We define the set ${}^*\mathbb{R}$ of *hyperreal numbers* (or *hyperreals*) by ${}^*\mathbb{R} := \mathbb{R}^I/{\sim_{\mathcal{F}}}$. It is therefore the set of infinite sequences on $\mathbb{R}$ modulo the following equivalence $\sim_{\mathcal{F}}$: we have $(a_0, a_1, \dots) \sim_{\mathcal{F}} (a'_0, a'_1, \dots)$ if

$$\{i \in I \mid a_i = a'_i\} \in \mathcal{F} \ , \quad \text{for which we say } \text{“}d_i = d'_i \text{ for almost every } i.\text{”} \tag{2}$$

A *hypernatural* $n \in {}^*\mathbb{N}$ is defined similarly.

It follows that: two sequences $(a_i)_i$ and $(a'_i)_i$ that coincide except for finitely many indices i represent the same hyperreal. The predicates besides $=$ (such as $<$) are defined in the same way. A notable consequence is the existence of an infinitesimal number: a hyperreal $\omega^{-1} := [(1, \frac{1}{2}, \frac{1}{3}, \dots)]$ is positive ($0 < \omega^{-1}$) but is smaller than any (standard) positive real r, since the latter is identified with the hyperreal $[(r, r, \dots)]$.

A hyperreal r is *limited* if it is not infinite, i.e. if there is a standard positive real $K \in \mathbb{R}$ such that $-K < r < K$. It is well-known (see [8, 14]) that a limited hyperreal r has a unique standard real that is infinitely close to r. This standard real is called the *shadow* of r and denoted by $\mathrm{sh}(r)$. The notion of shadow is a generalization of that of limit: if $(a_i)_i$ converges then $\mathrm{sh}\big([(a_0, a_1, \dots)]\big) = \lim_{i\to\infty} a_i$. See e.g. [8, 14].

Superstructure What we need from the logical machinery of NSA goes beyond the elementary fragment we just presented. The advanced fragment employs a set theory-like first-order language $\mathscr{L}_X$ and a so-called *superstructure* as a model.

A superstructure is a “universe,” constructed step by step from a certain base set X (whose typical examples are $\mathbb{R}$ and ${}^*\mathbb{R}$). We assume $\mathbb{N} \subseteq X$.

Definition 2.2 (superstructure) A *superstructure* $V(X)$ over X is defined by $V(X) := \bigcup_{n\in\mathbb{N}} V_n(X)$, where $V_0(X) := X$ and $V_{n+1}(X) := V_n(X) \cup \mathcal{P}(V_n(X))$.

(Ordered) pairs (a, b) and tuples $(a_1, \ldots, a_m)$ are defined in $V(X)$ as is usually done in set theory, e.g. $(a, b) := \{\{a\}, \{a, b\}\}$. The set $V(X)$ is closed under many set formation operations. For example the function space $a \to b$ is thought of as a collection of special binary relations (i.e. $a \to b \subseteq \mathcal{P}(a \times b)$), hence is in $V(X)$.

The First-Order Language $\mathscr{L}_X$ We use the following first-order language $\mathscr{L}_X$, defined for each choice of the base set like $\mathbb{R}$ and ${}^*\mathbb{R}$.

Definition 2.3 (the language $\mathscr{L}_X$) *Terms* in $\mathscr{L}_X$ consist of: variables $x, y, x_1, x_2, \ldots$; and a constant a for each entity $a \in V(X)$.

Formulas in $\mathscr{L}_X$ are constructed as follows.

- The predicate symbols are $=$ and $\in$; both are binary. The *atomic formulas* are of the form $s = t$ or $s \in t$ (where s and t are terms).
- Any Boolean combination of formulas is a formula. We use the symbols $\wedge, \vee, \neg$ and $\Rightarrow$.
- Given a formula A, a variable x and a term s, the expressions $\forall x \in s.\, A$ and $\exists x \in s.\, A$ are formulas.

Note that quantifiers always come with a bound s. The language $\mathscr{L}_X$ depends on the choice of X (it determines the set of constants). We shall also use the following syntax sugars in $\mathscr{L}_X$, as is common in NSA.

(s, t) pair	$(s_1, \ldots, s_m)$ tuple
$s \times t$ direct product	
$s \subseteq t$ inclusion, short for $\forall x \in s.\, x \in t$	
$s(t)$ function application; short for x such that $(t, x) \in s$	
$s \circ t$ function composition, $(s \circ t)(x) = s(t(x))$	
$s \leq t$ inequality in $\mathbb{N}$; short for $(s, t) \in \leq$ where $\leq \subseteq \mathbb{N}^2$	

Remark 2.4 The first-order language $\mathscr{L}_X$ resides in a different level from the languages like WHILE$^{\mathtt{dt}}$. $\mathscr{L}_X$ is used to formalize the semantics of those object-level languages and to prove their meta-properties. $\mathscr{L}_X$ is a *meta-level language* in this sense.

Definition 2.5 (semantics of $\mathscr{L}_X$) We interpret $\mathscr{L}_X$ in the superstructure $V(X)$ in the obvious way. Let A be a closed formula; we say A is *valid* if A is true in $V(X)$.

The $*$-Transform and the Transfer Principle The so-called *ultrapower construction* yields a canonical map

$$ {}^*(\_) \;:\; V(X) \longrightarrow V({}^*X) \;, \qquad a \longmapsto {}^*a \tag{3} $$

that is called the **-transform*. It is a map from the universe $V(X)$ of standard entities to $V({}^*X)$ of nonstandard entities. The details of its construction are in Appendix A.

The map ${}^*(\_)$ becomes a *monomorphism*, a notion in NSA. Most notably it satisfies the *transfer principle* (Lem. 2.7).

Definition 2.6 (*-transform of formulas) Let A be a formula in $\mathscr{L}_X$. The **-transform* of A, denoted by ${}^*A$, is a formula in $\mathscr{L}_{{}^*X}$ obtained by replacing each constant a occurring in A with the constant ${}^*a$ that designates the element ${}^*a \in V({}^*X)$.

Lemma 2.7 (the transfer principle) *For any closed formula A in $\mathscr{L}_X$, A is valid (in $V(X)$) if and only if ${}^*A$ is valid (in $V({}^*X)$).* □

2.2 The Modeling Language WHILE$^{\mathtt{dt}}$

WHILE$^{\mathtt{dt}}$, a modeling language for hybrid systems based on NSA, is introduced in [17]. It is an augmentation of a usual imperative language (such as **IMP** in [19]) with a constant `dt` that expresses an infinitesimal number.

Definition 2.8 Let $\mathbf{Var}$ be the set of variables. The syntax of WHILE$^{\mathtt{dt}}$ is as follows:

$$\begin{aligned}
\mathbf{AExp} \ni a &::= x \mid r \mid a_1 \ \mathtt{aop}\ a_2 \mid \mathtt{dt} \\
&\qquad \text{where } x \in \mathbf{Var}, r \in \mathbb{R} \text{ and } \mathtt{aop} \in \{+, -, \cdot, /\} \\
\mathbf{BExp} \ni b &::= \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2 \\
\mathbf{Cmd} \ni c &::= \mathtt{skip} \mid x := a \mid c_1; c_2 \mid \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \mid \mathtt{while}\ b\ \mathtt{do}\ c.
\end{aligned}$$

An expression $a \in \mathbf{AExp}$ is an *arithmetic expression*, $b \in \mathbf{BExp}$ is a *Boolean expression* and $c \in \mathbf{Cmd}$ is a *command*.

In the usual, standard abstract interpretation (without `dt`), a command c is assigned its *collecting semantics* $\mathcal{P}(\mathbf{Var} \to \mathbb{R}) \to \mathcal{P}(\mathbf{Var} \to \mathbb{R})$ (see e.g. [4]). This is semantics by reachable sets of memory states, as the concrete semantics. Presence of `dt` in the syntax of WHILE$^{\mathtt{dt}}$ calls for an infinitesimal number in the picture. Therefore we replace the set $\mathbb{R}$ by ${}^*\mathbb{R}$; for WHILE$^{\mathtt{dt}}$ commands, their collecting semantics is of the type $\mathcal{P}(\mathbf{Var} \to {}^*\mathbb{R}) \to \mathcal{P}(\mathbf{Var} \to {}^*\mathbb{R})$.

Definition 2.9 *Collecting semantics* for WHILE$^{\mathtt{dt}}$, in Table 1, has the following types.

$$\begin{array}{lcl}
[\![a]\!] : & (\mathbf{Var} \to {}^*\mathbb{R}) \to {}^*\mathbb{R} & \text{for } a \in \mathbf{AExp} \\
[\![b]\!] : & (\mathbf{Var} \to {}^*\mathbb{R}) \to \mathbb{B} & \text{for } b \in \mathbf{BExp} \\
[\![c]\!] : & \mathcal{P}(\mathbf{Var} \to {}^*\mathbb{R}) \to \mathcal{P}(\mathbf{Var} \to {}^*\mathbb{R}) & \text{for } c \in \mathbf{Cmd}
\end{array}$$

In [17] and in §1, the semantics of a while loop is defined using the idea of sectionwise execution, instead of as a least fixed point. This is not suited for our current purpose of employing abstract interpretation—the latter is after all for computing least fixed points. The collecting semantics in Def. 2.9 (Table 1) does use least fixed points; it is based on the alternative WHILE$^{\mathtt{dt}}$ semantics introduced in [15] (it will also appear in the full version of [13, 17]). The equivalence of the two semantics is established in [15].

In the rest of the paper we restrict the set of variables $\mathbf{Var}$ to be finite. This assumption—a realistic one when we focus on the program to be analyzed—makes our NSA framework much simpler. Therefore $\mathcal{P}(\mathbf{Var} \to X)$ is equal to $\mathcal{P}(X^n)$ ($X = \mathbb{R}$ or ${}^*\mathbb{R}$) for some $n \in \mathbb{N}$; we prefer the latter notation in what follows.

2.3 Abstract Interpretation

Abstract interpretation [7] is a well-established technique in static analysis. We make a brief review of its basic theory; it is mostly for the purpose of fixing notations.

$$\llbracket x \rrbracket \boldsymbol{\sigma} := \boldsymbol{\sigma}(x) \text{ for each } x \in \mathbf{Var} \qquad \llbracket \mathtt{true} \rrbracket \boldsymbol{\sigma} := \mathrm{tt}$$
$$\llbracket r \rrbracket \boldsymbol{\sigma} := r \text{ for each } r \in \mathbb{R} \qquad \llbracket \mathtt{false} \rrbracket \boldsymbol{\sigma} := \mathrm{ff}$$
$$\llbracket a_1 \ \mathtt{aop}\ a_2 \rrbracket \boldsymbol{\sigma} := \llbracket a_1 \rrbracket \ \mathtt{aop}\ \llbracket a_2 \rrbracket \qquad \llbracket b_1 \wedge b_2 \rrbracket \boldsymbol{\sigma} := \llbracket b_1 \rrbracket \wedge \llbracket b_2 \rrbracket$$
$$\llbracket \mathtt{dt} \rrbracket \boldsymbol{\sigma} := [(1, \tfrac{1}{2}, \tfrac{1}{3}, \cdots)] \qquad \llbracket \neg b \rrbracket \boldsymbol{\sigma} := \neg(\llbracket b \rrbracket \boldsymbol{\sigma})$$

$$\llbracket \mathtt{skip} \rrbracket \mathbf{S} := \mathbf{S}$$
$$\llbracket x := a \rrbracket \mathbf{S} := \{ \boldsymbol{\sigma}[\llbracket a \rrbracket \boldsymbol{\sigma} / x] \mid \boldsymbol{\sigma} \in \mathbf{S} \}$$
$$\llbracket c_1 ; c_2 \rrbracket \mathbf{S} := \llbracket c_2 \rrbracket (\llbracket c_1 \rrbracket \mathbf{S})$$
$$\llbracket \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \rrbracket \mathbf{S} := \begin{array}{l} \{ \llbracket c_1 \rrbracket \boldsymbol{\sigma} \mid \boldsymbol{\sigma} \in \mathbf{S},\ \llbracket b \rrbracket \boldsymbol{\sigma} = \mathrm{tt} \} \\ \cup \{ \llbracket c_2 \rrbracket \boldsymbol{\sigma} \mid \boldsymbol{\sigma} \in \mathbf{S},\ \llbracket b \rrbracket \boldsymbol{\sigma} = \mathrm{ff} \} \end{array}$$
$$\llbracket \mathtt{while}\ b\ \mathtt{do}\ c \rrbracket \mathbf{S} := \mathrm{lfp}({}^{*}\Phi(\llbracket b \rrbracket)(\llbracket c \rrbracket))$$
$$\text{where } \Phi : (\mathbf{St} \to \mathbb{B} \cup \{\bot\}) \to (\mathcal{P}(\mathbf{Var} \to \mathbb{R}) \to \mathcal{P}(\mathbf{Var} \to \mathbb{R})) \to$$
$$((\mathcal{P}(\mathbf{Var} \to \mathbb{R}) \to \mathcal{P}(\mathbf{Var} \to \mathbb{R})) \to (\mathcal{P}(\mathbf{Var} \to \mathbb{R}) \to \mathcal{P}(\mathbf{Var} \to \mathbb{R})))$$
$$\text{is defined by } \Phi(f)(g) = \lambda\psi.\ \lambda S.\ \{\psi(g(\sigma)) \mid \sigma \in S,\ f(\sigma) = \mathrm{tt}\} \cup \{\sigma \mid \sigma \in S,\ f(\sigma) = \mathrm{ff}\}.$$

Table 1. WHILE$^{\mathtt{dt}}$ collecting semantics

Definition 2.10 (Galois connection) Let $(L, \sqsubseteq)$ and $(\overline{L}, \overline{\sqsubseteq})$ be posets, and $\alpha : L \to \overline{L}$ and $\gamma : \overline{L} \to L$ be functions. A tuple $\big((L, \sqsubseteq), (\overline{L}, \overline{\sqsubseteq}), \alpha, \gamma\big)$ is said to be a *Galois connection* if: for each $x \in L$ and $\overline{y} \in \overline{L}$, we have $\alpha x \ \overline{\sqsubseteq}\ \overline{y} \Leftrightarrow x \sqsubseteq \gamma \overline{y}$. This fact is denoted by $L \underset{\gamma}{\overset{\alpha}{\rightleftharpoons}} \overline{L}$; and we call L a *concrete domain*, $\overline{L}$ an *abstract domain*, α an *abstraction function* and γ a *concretization function*.

Proposition 2.11 *A Galois connection* $(L, \sqsubseteq) \underset{\gamma}{\overset{\alpha}{\rightleftharpoons}} (\overline{L}, \overline{\sqsubseteq})$ *extends to monotone endo-functions. Concretely, it yields a Galois connection* $(L \underset{\text{mono.}}{\longrightarrow} L) \underset{\vec{\gamma}}{\overset{\vec{\alpha}}{\rightleftharpoons}} (\overline{L} \underset{\text{mono.}}{\longrightarrow} \overline{L})$ *where* $L \underset{\text{mono.}}{\longrightarrow} L$ *and* $\overline{L} \underset{\text{mono.}}{\longrightarrow} \overline{L}$ *are the posets of monotone functions ordered by the pointwise extension of* $\sqsubseteq$ *and* $\overline{\sqsubseteq}$*. The functions* $\vec{\gamma}$ *and* $\vec{\alpha}$ *here are defined by:* $\vec{\gamma}(f) = \gamma \circ f \circ \alpha$ *and* $\vec{\alpha}(f) = \alpha \circ f \circ \gamma$*, respectively.* □

A Galois connection allows us to over-approximate, in the abstract domain $\overline{L}$, least-fixed points in the concrete domain L. Let $L \underset{\gamma}{\overset{\alpha}{\rightleftharpoons}} \overline{L}$ be a Galois connection such that L is additionally a cpo; $F \colon L \to L$ be a continuous function on L; and $\bot \in L$ be an element such that $\bot \sqsubseteq F(\bot)$. The *least fixed point relative to* $\bot$, denoted by $\mathrm{lfp}_{\bot} F$, is the least among the fixed points of F above $\bot$; by the cpo structure it is given by $\bigsqcup_{n \in \mathbb{N}} F^n \bot$.

Proposition 2.12 *In the above setting, assume further that:* $\overline{F} : \overline{L} \to \overline{L}$ *be a monotone function such that* $F \sqsubseteq \vec{\gamma}(\overline{F})$*; and that* $\overline{x} \in \overline{L}$ *is a prefixed point of* $\overline{F}$ *(i.e.* $\overline{F}(\overline{x}) \ \overline{\sqsubseteq}\ \overline{x}$*) such that* $\alpha(\bot) \ \overline{\sqsubseteq}\ \overline{x}$*.*

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

In a typical situation where we wish to analyze imperative programs, we use $\mathcal{P}(\mathbb{R}^n)$—the set of subsets of memory states—as a concrete domain L. The programs' interpretation as functions F on L is defined in a straightforward manner; but *computing* such interpretation is nontrivial, principally due to loops. In abstract interpretation one aims to over-approximate the interpretation of a loop—i.e. $\mathrm{lfp}_{\perp} F$ in L, where F is the interpretation of the loop's body—exploiting Prop. 2.12. Towards that goal we have to: 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 we will be using 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).

Definition 2.13 (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

$$y_0 \;=\; x_0 \quad \text{and} \qquad y_{i+1} \;=\; y_i \nabla x_{i+1} \;\; \text{for each } i \in \mathbb{N}$$

  is ultimately stationary.

A widening operator on a fixed abstract domain $\overline{L}$ is not at all unique. In this paper we will discuss two widening operators previously introduced for the $\mathsf{Intv}_{\mathbb{R}}$, and three for $\mathbb{CP}_n$.

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

Proposition 2.14 (convergence of iteration sequences) *Let $(L, \sqsubseteq)$ be a poset; $F : L \to L$ be a monotone function; $\perp \in L$ be such that $\perp \sqsubseteq F(\perp)$; $\nabla : L \times L \to L$ be a widening operator; and $\langle X_i \rangle_{i \in \mathbb{N}} \in L^{\mathbb{N}}$ be the infinite sequence defined by*

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

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 $\perp \sqsubseteq \bigsqcup_{i \in \mathbb{N}} X_n$. □

Example I: Interval Domains The *interval domain* $\mathsf{Intv}_{\mathbb{Z}}$ over integers is introduced in [4]; we will be using its variant $\mathsf{Intv}_{\mathbb{R}}$ over reals.

Definition 2.15 (interval domains Intv) The *domains of intervals of integers*, and *of reals*, are defined by

$$\begin{aligned}\mathsf{Intv}_{\mathbb{Z}} &:= \{\bot\} \cup \{\, [l,u] \mid l \in \mathbb{Z} \cup \{-\infty\}, u \in \mathbb{Z} \cup \{\infty\}, l \le u \,\}\ ,\\ \mathsf{Intv}_{\mathbb{R}} &:= \{\bot\} \cup \{\, [l,u] \mid l \in \mathbb{R} \cup \{-\infty\}, u \in \mathbb{R} \cup \{\infty\}, l \le u \,\}\ .\end{aligned}$$

For both of these, the order $\sqsubseteq$ is the inclusion order of intervals, with $\bot$ being the least.

Each of the above has a Galois connection with $L = \mathcal{P}(\mathbb{R})$. For example, the abstraction function $\alpha : \mathcal{P}(\mathbb{R}) \to \mathsf{Intv}_{\mathbb{R}}$ maps $X \in \mathcal{P}(\mathbb{R})$ to the interval $[\min(X), \max(X)]$; and the concretization function $\gamma : \mathsf{Intv}_{\mathbb{R}} \to \mathcal{P}(\mathbb{R})$ maps $[l,u]$ to $\{r \in \mathbb{R} \mid l \le r \le u\}$ and $\bot$ to $\emptyset$. It is straightforward to extend the last Galois connection $\mathcal{P}(\mathbb{R}) \rightleftharpoons \mathsf{Intv}_{\mathbb{R}}$ to $\mathcal{P}(\mathbb{R}^n) \rightleftharpoons (\mathsf{Intv}_{\mathbb{R}})^n$: this is done in a pointwise manner; precisely speaking the latter arises as a composition of two Galois connections via $(\mathcal{P}(\mathbb{R}))^n$.

It is straightforward to interpret a (loop-free) imperative program c on the interval domain, as a monotone function $[\![c]\!]_{\mathsf{Intv}} \colon (\mathsf{Intv}_{\mathbb{R}})^n \to (\mathsf{Intv}_{\mathbb{R}})^n$. We have the following fact that is well-known in the community; we leave the precise syntax for a program c implicit (one can use a syntax like the one of WHILE$^{\mathsf{dt}}$ presented later).

Lemma 2.16 *Let c be a loop-free imperative program. We have $[\![c]\!] \sqsubseteq \vec{\gamma}([\![c]\!]_{\mathsf{Intv}})$, where $\vec{\gamma}$ is the function in Prop. 2.11 applied to $\mathcal{P}(\mathbb{R}^n) \rightleftharpoons (\mathsf{Intv}_{\mathbb{R}})^n$.* □

We go on to describe two widening operators for $\mathsf{Intv}_{\mathbb{R}}$ that appear in the literature.

Definition 2.17 (widening operator ∇_{Intv} [4]) Let $\mathsf{Intv} = \mathsf{Intv}_{\mathbb{Z}}$ or $\mathsf{Intv}_{\mathbb{R}}$. The following function $\nabla_{\mathsf{Intv}} : \mathsf{Intv} \times \mathsf{Intv} \to \mathsf{Intv}$ is a widening operator on Intv.

$$[l_1,u_1]\nabla_{\mathsf{Intv}}[l_2,u_2] = \big[\,\text{if } l_1 \le l_2 \text{ then } l_1 \text{ else } -\infty,\ \text{if } u_1 \ge l_2 \text{ then } u_1 \text{ else } \infty\,\big]$$

Definition 2.18 (widening operator $\nabla_{\mathsf{Intv}_{\mathbb{Z},c}}$) Let $c \in \mathbb{Z}$ be fixed. The following function $\nabla_{\mathsf{Intv}_{\mathbb{Z},c}} : \mathsf{Intv}_{\mathbb{Z}} \times \mathsf{Intv}_{\mathbb{Z}} \to \mathsf{Intv}_{\mathbb{Z}}$ is a widening operator on $\mathsf{Intv}_{\mathbb{Z}}$.

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

Example II: the Domain of Convex Polyhedra The *domain of convex polyhedra*, introduced in [7], is one of the most commonly used relational abstract domains.

Definition 2.19 (domain of convex polyhedra $\mathbb{CP}_n$) An n-dimensional *convex polyhedron* is the intersection of finitely many (closed) affine half-spaces. We denote the set of convex polyhedra in $\mathbb{R}^n$ by $\mathbb{CP}_n$. Its poset structure $\sqsubseteq$ is given by the inclusion order.

The domain of convex polyhedra is usually defined over integers and that forms a Galois connection with $\mathcal{P}(\mathbb{Z}^n)$. However, convex polyhedra over reals (or rationals)—although they have also been used for abstract interpretation [11, 12]—do not form a Galois connection in which $\mathcal{P}(\mathbb{R}^n)$ is a concrete domain. Consider a disk; there is no smallest convex polyhedron that contains it.

We use the following axiomatic "quick fix" that, if theoretically cumbersome, suffices for our goal of program verification.

Definition 2.20 (approximable subset $\mathbb{R}^n$) A subset $S \subseteq \mathbb{R}^n$ of the n-dimensional Euclidean space is *approximable* if there exists the smallest polyhedron that contains it. The set of all approximable subsets $S \subseteq \mathbb{R}^n$ is denoted by Appr_n.

Lemma 2.21 *1. We have a Galois connection $\mathsf{Appr}_n \rightleftharpoons \mathbb{CP}_n$: its concretization carries a convex polyhedron P to P considered as a subset of $\mathbb{R}^n$; and its abstraction is defined by the definition of Appr_n.*

2. Let c be a loop-free imperative program, in which n variables occur freely. Its interpretation $[\![c]\!] \colon \mathcal{P}(\mathbb{R}^n) \to \mathcal{P}(\mathbb{R}^n)$—defined in the obvious manner—restricts to a function $[\![c]\!] \colon \mathsf{Appr}_n \to \mathsf{Appr}_n$.

3. Let c be the same as above. Its interpretation $[\![c]\!]_{\mathbb{CP}} \colon \mathbb{CP}_n \to \mathbb{CP}_n$ can be defined, much like in [7], using constraint systems and generator systems as presentations of convex polyhedra. This satisfies $[\![c]\!] \sqsubseteq \vec{\gamma}([\![c]\!]_{\mathbb{CP}})$. □

What corresponds to Prop. 2.12 needs some care to formulate.

Proposition 2.22 *Let $F \colon \mathsf{Appr}_n \to \mathsf{Appr}_n$ be a continuous function; $\bot \in \mathsf{Appr}_n$ be such that $\bot \sqsubseteq F(\bot)$; $\overline{F} \colon \mathbb{CP}_n \to \mathbb{CP}_n$ be a monotone function such that $F \sqsubseteq \vec{\gamma}(\overline{F})$; and $\overline{x} \in \mathbb{CP}_n$ be such that $\overline{F}(\overline{x}) \mathrel{\overline{\sqsubseteq}} \overline{x}$ and $\alpha(\bot) \mathrel{\overline{\sqsubseteq}} \overline{x}$.*

Then the least fixed point $\mathrm{lfp}_{\bot} F$ relative to $\bot$—it need not lie in Appr_n but exists in $\mathcal{P}(\mathbb{R}^n)$—is over-approximated by $\overline{x}$, that is, $\mathrm{lfp}_{\bot} F \sqsubseteq \gamma(\overline{x})$. □

The above results let us work with the Galois connection $\mathsf{Appr}_n \rightleftharpoons \mathbb{CP}_n$ for static analysis.

We go on to discuss three widening operators for $\mathbb{CP}_n$ that appear in the literature. There we will need to explicitly manipulate presentations of a convex polyhedron by a *constraint system* and a *generator system*. We only present the definition of the former; further details are found in [7].

Definition 2.23 A *constraint system* C is a system $\{\, f_1(\mathbf{x}) \le c_1,\ \ldots,\ f_m(\mathbf{x}) \le c_m \,\}$ of ($\mathbb{R}$-coefficient) linear inequalities. The convex polyhedron $\{\mathbf{x} \in \mathbb{R}^n \mid f_i(\mathbf{x}) \le c_i,\ \forall i\}$ induced by C is denoted by $\mathsf{con}(C)$.

We will be studying three widening operators on $\mathbb{CP}_n$ in the extended nonstandard setting. They are namely: the *standard widening operator* ∇_S [9];[3] the *widening operator ∇_M up to M* [10, 12]; and the *precise widening operator* ∇_N [2]. We briefly describe the former two; the definition of the last is omitted for the lack of space.

Definition 2.24 (standard widening ∇_S) Let $P_1, P_2 \in \mathbb{CP}_n$; and C_1 and C_2 be constraints system that induce P_1 and P_2, respectively. *The standard widening operator* $\nabla_S : \mathbb{CP}_n \times \mathbb{CP}_n \to \mathbb{CP}_n$ is defined by

$$P_1 \nabla_S P_2 := \begin{cases} P_2 & \text{if } P_1 = \emptyset \\ \mathsf{con}\left(\begin{array}{l}\{\varphi \in C_1 \mid C_2 \text{ implies } \varphi, \text{ i.e. } \varphi \text{ is everywhere true in } P_2\} \\ \cup\{\psi \in C_2 \mid \exists \varphi \in C_1.\ P_1 = \mathsf{con}(C_1[\psi/\varphi])\}\end{array}\right) & \text{otherwise.} \end{cases}$$

[3] The name "standard" is confusing with the distinction between *standard* and *nonstandard* entities in NSA. The use of "standard" in the former sense is scarce in this paper.

Intuitively $P_1 \nabla_S P_2$ is represented by the set of those linear constraints of P_1 which are satisfied by every point of P_2. The second argument of $\cup$ in the second case is there to deal with singularities and not essential.

The following second widening operator ∇_M refines ∇_S. This is what we use in our implementation. Here M is a parameter.

Definition 2.25 (widening up to M, ∇_M) Let $P_1, P_2 \in \mathbb{CP}_n$, and M be a (given) finite set of linear inequalities. The *widening operator up to* M is defined by

$$P_1 \nabla_M P_2 \;:=\; \big(\mathcal{P}_1 \nabla_S P_2\big) \;\cap\; \mathsf{con}\big(\{\varphi \in M \mid P_i \subseteq \mathsf{con}(\{\varphi\}) \text{ for } i = 1, 2\}\big) \;.$$

The parameter M is usually taken to be the set of linear inequalities that occur in the program under analysis.

3 Abstract Interpretation Augmented with Infinitesimals

Building on the theoretical foundations we have just described, we now present an abstract interpretation framework for the analysis of WHILE$^{\mathtt{dt}}$ programs—and hence the hybrid dynamics modeled thereby. We introduce two *abstract hyperdomains* over ${}^*\mathbb{R}$ (by intervals and by convex polyhedra) as the transfer of the corresponding (standard, over $\mathbb{R}$) abstract domains. We then interpret WHILE$^{\mathtt{dt}}$ programs in them, and transfer the five widening operators mentioned in §2.3 to the nonstandard setting. We classify them into *uniform* ones—for which termination is guaranteed even in the nonstandard setting—and nonuniform ones.

3.1 Abstract Domains over Hyperreals

Definition 3.1 (interval hyperdomain ${}^*\mathsf{Intv}_R$) The *interval hyperdomain* over ${}^*\mathbb{R}$ is defined by

$${}^*\mathsf{Intv}_R \;:=\; \{\bot\} \cup \big\{\, [l, u] \;\mid\; l \in {}^*\mathbb{R} \cup \{-\infty\},\, u \in {}^*\mathbb{R} \cup \{\infty\},\, l \le u \,\big\}.$$

It comes with an obvious Galois connection $\mathcal{P}({}^*\mathbb{R}) \rightleftharpoons {}^*\mathsf{Intv}_R$.

The notation ${}^*\mathsf{Intv}_R$ here might seem strange: it is a variant of $\mathsf{Intv}_{\mathbb{R}}$ and is hence more like $\mathsf{Intv}_{{}^*R}$. The following characterization, however, justifies our notation. In fact the characterization is what allows us to transfer all the properties of $\mathsf{Intv}_{\mathbb{R}}$ to ${}^*\mathsf{Intv}_{\mathbb{R}}$. We present its proof, since it is a prototype of most of the proofs that follow.

Proposition 3.2 *The interval hyperdomain ${}^*\mathsf{Intv}_{\mathbb{R}}$ coincides with ${}^*(\mathsf{Intv}_{\mathbb{R}})$, the $*$-transform (see (3)) of the (standard) interval domain $\mathsf{Intv}_{\mathbb{R}}$.*

Proof. The following $\mathscr{L}_{\mathbb{R}}$-sentence is valid in $\mathbb{R}$, by the definition of the set $\mathsf{Intv}_{\mathbb{R}}$.

$$\forall x \in \mathbb{R}.\, \big(\, x \in \mathsf{Intv}_{\mathbb{R}} \Leftrightarrow x \in \big(\{\bot\} \cup \{[l, u] \mid l \in \mathbb{R} \cup \{-\infty\}, u \in \mathbb{R} \cup \{\infty\}, l \le u\}\big) \,\big).$$

Therefore by Lem. 2.7, its $*$-transform (a $\mathscr{L}_{{}^*\mathbb{R}}$-sentence)

$$\forall x \in {}^*\mathbb{R}.\, \big(\, x \in {}^*(\mathsf{Intv}_{\mathbb{R}}) \Leftrightarrow x \in \big(\{\bot\} \cup \{[l, u] \mid l \in {}^*\mathbb{R} \cup \{-\infty\}, u \in {}^*\mathbb{R} \cup \{\infty\}, l \le u\}\big) \,\big).$$

is valid in ${}^*\mathbb{R}$ (where we used Lem. A.2.4). This is the claim itself. □

We extend convex polyhedra to the current nonstandard setting, too.

Definition 3.3 (convex polyhedra over ${}^*\mathbb{R}$) A *convex polyhedron* on $({}^*\mathbb{R})^n$ is an intersection of finite number of affine half-spaces on $({}^*\mathbb{R})^n$, that is, the set of points $\mathbf{x} \in ({}^*\mathbb{R})^n$ that satisfy a certain finite set of linear inequalities. The set of all convex polyhedra on $({}^*\mathbb{R})^n$ is denoted by $\mathbb{CP}_n^{{}^*\mathbb{R}}$.

It comes with an obvious Galois connection $\mathsf{Appr}_{{}^*\mathbb{R},n} \rightleftharpoons \mathbb{CP}_n^{{}^*\mathbb{R}}$, where $\mathsf{Appr}_{{}^*\mathbb{R},n} \subseteq \mathcal{P}(({}^*\mathbb{R})^n)$ is defined in the same way as in Def. 2.20.

Proposition 3.4 *The set $\mathbb{CP}_n^{{}^*\mathbb{R}}$ of all convex polyhedra over ${}^*\mathbb{R}^n$ is a (proper) subset of ${}^*\mathbb{CP}_n$, the $*$-transform of the (standard) domain of convex polyhedra over $\mathbb{R}^n$.* □

What lies in the difference between the two sets $\mathbb{CP}_n^{{}^*\mathbb{R}} \subsetneq {}^*\mathbb{CP}_n$ is, for example, a disk as a subset of $\mathbb{R}^2$ (hence of ${}^*\mathbb{R}^2$). In ${}^*\mathbb{CP}_2$ one can use a constraint system C whose number of inequalities is a hypernatural number $m \in {}^*\mathbb{N}$; using e.g. $m = \omega = [(0, 1, 2, \dots)]$ allows us to approximate a disk with progressive precision.

In the following development of nonstandard abstract interpretation, we will use ${}^*\mathbb{CP}_n$ as an abstract domain since it allows transfer of properties in the standard world (over $\mathbb{R}$). We note, however, that our over-approximation of the interpretation $[\![c]\!]$ of a loop-free WHILE$^{\mathtt{dt}}$ program c is always given in $\mathbb{CP}_n^{{}^*\mathbb{R}}$, i.e. with finitely many linear inequalities.

3.2 Theory of Hyper Abstract Interpretation

We have seen that the "hyperdomains" ${}^*\mathsf{Intv}_{\mathbb{R}}$ and ${}^*\mathbb{CP}_n$ are in Galois connections to the concrete domains $\mathcal{P}({}^*\mathbb{R})$ and $\mathsf{Appr}_{{}^*R,n}$, respectively; hence to them all the abstract interpretation infrastructure in §2.3 apply. This, however, does not suffice for our purpose of static analysis of WHILE$^{\mathtt{dt}}$ programs. The reason is simply that the interpretation $[\![c]\!]$ of a WHILE$^{\mathtt{dt}}$ program c may fail to be continuous. A simple counterexample is given by $c \equiv (\mathtt{x:=x+dt})$.

Fortunately it turns out that we can rely on the $*$-*transform* (§2.1) of the theory in §2.3, where it suffices to impose a weaker condition of $*$-*continuity*—instead of the (standard) continuity—on the function $[\![c]\!]$. This theoretical framework of *hyper abstract interpretation*, which we shall describe here, is an extension of the *transferred domain theory* studied in [3, 18]. Part of the latter is found also in Appendix B.

Definition 3.5 (hyper-Galois connection) A *hyper-Galois connection*, denoted by ${}^*L \underset{{}^*\gamma}{\overset{{}^*\alpha}{\rightleftharpoons}} {}^*\overline{L}$, is a quintuple $\left({}^*L, {}^*\overline{L}, {}^*\alpha, {}^*\gamma\right)$ of: the $*$-transform of a poset L; that of a poset $\overline{L}$; the $*$-transform ${}^*\alpha\colon {}^*L \to {}^*\overline{L}$ of a function $\alpha\colon L \to \overline{L}$; and the $*$-transform ${}^*\gamma\colon {}^*\overline{L} \to {}^*L$ of γ. We require that the data $(L, \overline{L}, \alpha, \gamma)$ forms a Galois connection (Def. 2.10).

The above ${}^*\alpha$ is an internal function (i.e. ${}^*\alpha \in {}^*(L \to \overline{L})$); see Appendix A for details.

The notion of $*$-*continuous function* $f'\colon {}^*L \to {}^*L$ is defined analogously, namely that f' is the $*$-transform of some continuous function $f\colon L \to L$. See Appendix B.

Here is the counterpart of Prop. 2.12. As announced, it only requires F's $*$-continuity.

Theorem 3.6 *Let $(L, \sqsubseteq)$ be a cpo, $(\overline{L}, \overline{\sqsubseteq})$ be a poset such that $L \underset{\gamma}{\overset{\alpha}{\rightleftharpoons}} \overline{L}$, and consider the induced hyper-Galois connection ${}^*L \underset{{}^*\gamma}{\overset{{}^*\alpha}{\rightleftharpoons}} {}^*\overline{L}$. 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. 3.6 in over-approximating loop semantics in $\textsc{While}^{\mathtt{dt}}$. Some care is needed here. By $*$-transforming standard Galois connections $\mathcal{P}(\mathbb{R}^n) \rightleftharpoons (\mathsf{Intv}_{\mathbb{R}})^n$ and $\mathsf{Appr}_n \rightleftharpoons \mathbb{CP}_n$ we obtain hyper-Galois connections Def. 3.5 to which Thm. 3.6 applies. The resulting concrete hyperdomains, however, are ${}^*(\mathcal{P}(\mathbb{R}^n))$ and ${}^*\mathsf{Appr}_n$; these are not precisely what we expect, namely $\mathcal{P}({}^*(\mathbb{R})^n)$ and $\mathsf{Appr}_{{}^*\mathbb{R},n} \subseteq \mathcal{P}({}^*(\mathbb{R})^n)$. The former two are proper subsets of the latter two, respectively; more specifically the former consist of all the *internal* entities (Def. A.3) that reside in the latter.

These gaps are not a problem, by the following result.

Proposition 3.7 *1. Let c be a $\textsc{While}^{\mathtt{dt}}$ command (that can contain loops). The subset ${}^*(\mathcal{P}(\mathbb{R}^n))$ of $\mathcal{P}({}^*(\mathbb{R})^n)$ is closed under the collecting semantics $[\![c]\!]$, that is, $[\![c]\!]\mathbf{S}$ is internal if $\mathbf{S}$ is internal.*

2. Let c be a loop-free $\textsc{While}^{\mathtt{dt}}$ command (that can contain loops). The subset ${}^*\mathsf{Appr}_n$ of $\mathcal{P}({}^*(\mathbb{R})^n)$ is closed under $[\![c]\!]$. □

Our goal is over-approximation of the semantics of iteration of a loop-free $\textsc{While}^{\mathtt{dt}}$ program c, relying on Thm. 3.6. 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 $[\![\_]\!]_{\mathsf{Intv}}$ and $[\![\_]\!]_{\mathbb{CP}}$ (in standard abstract interpretation, §2.3) can be used in such $\overline{F}$.

Proposition 3.8 *Let $L \underset{\gamma}{\overset{\alpha}{\rightleftharpoons}} \overline{L}$ 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. ${}^*F \, {}^*\sqsubseteq (\vec{{}^*\gamma})({}^*\overline{F})$.* □

We summarize what we observed so far, on nonstandard abstract interpretation.

Corollary 3.9 (soundness of ${}^*\mathsf{Intv}_{\mathbb{R}}$ and ${}^*\mathbb{CP}_n$ for $\textsc{While}^{\mathtt{dt}}$) *1. Let c be a loop-free $\textsc{While}^{\mathtt{dt}}$ command; $\bot \in \mathcal{P}({}^*(\mathbb{R})^n)$ be an internal element (i.e. an element of ${}^*(\mathcal{P}(\mathbb{R}^n))$); and $\overline{x} \in {}^*\mathsf{Intv}_{\mathbb{R}}$ be such that $({}^*[\![c]\!]_{\mathsf{Intv}})(\overline{x}) \, {}^*\sqsubseteq \overline{x}$ and $\bot \, {}^*\sqsubseteq \overline{x}$. Then we have an over-approximation $\mathrm{lfp}_{\bot} [\![c]\!] \, {}^*\sqsubseteq \overline{x}$.*

2. Let c be a loop-free $\textsc{While}^{\mathtt{dt}}$ command; $\bot \in \mathsf{Appr}_{{}^*\mathbb{R},n}$ be an internal element (i.e. an element of ${}^*(\mathcal{P}(\mathbb{R}^n))$); and $\overline{x} \in {}^*\mathbb{CP}_n$ be such that $({}^*[\![c]\!]_{\mathbb{CP}})(\overline{x}) \, {}^*\sqsubseteq \overline{x}$ and $\bot \, {}^*\sqsubseteq \overline{x}$. Then we have an over-approximation $\mathrm{lfp}_{\bot} [\![c]\!] \, {}^*\sqsubseteq \overline{x}$. □

3.3 Hyperwidening and Uniform Widening Operators

Towards our goal of using Thm. 3.6, the last remaining step is to find a prefixed point $\overline{x}$, i.e. $\overline{F}(\overline{x})\ {}^*\!\sqsubseteq\ \overline{x}$. This is where widening operators are standardly used; see §2.3.

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 3.10 *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*

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

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. 3.10 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[4] and on which we never actually operate—and therefore their infinite/infinitesimal nature has been not a problem. In contrast, computation of 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. 3.10) is of no use.

As a remedy we introduce *uniformity* of (standard) widening operators. It strengthen the original termination condition (Def. 2.13) 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 3.11 (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* $i \in \mathbb{N}$ such that: for any ascending chain $\langle x_k\rangle \in L^{\mathbb{N}}$ starting from x_0, there exists $j \leq i$ 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. 3.10; its proof relies on instantiating the uniform bound i in a suitable $\mathscr{L}_{\mathbb{R}}$-formula with a Skolem constant, before transfer.

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

Theorem 3.12 *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. 3.12 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 some of the commonly-known widening operators

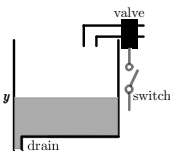
Theorem 3.13 *Uniformity of the five widening operators in §2.3 is listed in the table below. Among them three are uniform; two are not.*

interval domain Intv		*convex polyhedra* $\mathbb{CP}_n$		
∇_{Intv} *(Def. 2.17)*	$\nabla_{\mathsf{Intv}_{\mathbb{Z},c}}$ *(Def. 2.18)*	∇_S *(Def. 2.24)*	∇_M *(Def. 2.25)*	∇_N *([2])*
✓	×	✓	✓	×

□

4 Example, Implementation and Experiments

Water-level monitor is a commonly used example of hybrid systems [1]. See the figure on the right. There is a water tank with a constant drain (2 cm per second). When the water level y gets lower than 5 cm the switch is turned on, which eventually opens up the valve but only after a time lag of two seconds. While the valve is open, the water level y rises by 1 cm per second. Once y reaches 10 cm the switch is turned off, which will shut the valve but again after a time lag of two seconds.



Its WHILE$^{\mathsf{dt}}$ model is found in Fig. 1, on the left. Here 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 ($1 \leq x \leq 12$). As is customary in abstract interpretation we turn the program into a control flow graph, presented in Appendix C. We classify the other 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, the soundness of whose use is guaranteed in Cor. 3.9.

In fixed-point computation we use ∇_M, the widening operator up to M (Def. 2.25). The set M of linear constraints is taken from the program: $M = \{x \leq 5, x \geq 5, x \leq$

```
(*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
      }
};;
```

Fig. 1. Example WHILE$^{\mathtt{dt}}$ code

$10, x \geq 10, l \leq 2, l \geq 2\}$. As is common in abstract interpretation (see e.g. [5]), we add delays of widening by an *extrapolation threshold*.

The iteration sequence we obtain is depicted in Appendix C, together with other details. In the end we successfully obtain a prefixed point $\overline{x}$ and that implies $1 - \mathtt{dt} \leq x \leq 12 + \mathtt{dt}$ is an invariant. This is what we wanted modulo infinitesimal gaps.

Implementation We implemented a prototype tool for analysis of WHILE$^{\mathtt{dt}}$ programs. The tool currently supports: $\mathbb{CP}_n$ as an abstract domain; and ∇_M (Def. 2.25) as a widening operator. Its input is given by: 1) a WHILE$^{\mathtt{dt}}$ program; 2) an initial state; 3) the set M of linear constraints used in ∇_M; and 4) an extrapolation threshold as well as the timing of widening delays (the last can also be given interactively). Its output is a convex polyhedron that over-approximates the set of reachable memory states.

Our tool consists principally of the following two components: 1) an OCaml frontend for parsing and forming an iteration sequence; and 2) a Mathematica backend that executes operations on convex polyhedra. The two components are interconnected by a C++ program, via MathLink.

Experiments We analyzed two WHILE$^{\mathtt{dt}}$ programs—water-level monitor and thermostat (Fig 1)—with our prototype. They are common hybrid system examples; see [1] and also §4. The experiments were on Apple MacBook Pro with 2.6 GHz Dual-core Intel Core i5 CPU and 8 GB memory.

Water-Level Monitor This is a piecewise-linear dynamics and a typical example used in hybrid automata literature. Our tool automates the analysis presented in §4; the execution time was 34.051 sec.

Thermostat This example exhibits dynamics that is beyond piecewise-linear; and is hence in principle beyond the reach of hybrid-automaton model checker, unless suitable approximation is employed. Our approach of nonstandard abstraction successfully analyzes this example; without explicit piecewise-linear approximation; we believe this result witnesses a potential of our approach. Our tool executes in 6.692 sec. and outputs an approximation from which we obtain an invariant $18 - 54 * \mathtt{dt} \leq x \leq 22 + 24 * \mathtt{dt}$.

The tool is admittedly of an experimental nature and leaves a lot of room for improvement. For example, automatic extraction of the set M from the program will be straightforward. Supporting abstract domains and widening operators other than $\mathbb{CP}_n$ and ∇_M is imminent future work.

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. 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.
6. Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, David Monniaux, and Xavier Rival. The astreé analyzer. In *Programming Languages and Systems, 14th European Symposium on Programming,ESOP 2005, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2005, Edinburgh, UK, April 4-8, 2005, Proceedings*, pages 21–30, 2005.
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. *Dtermination automatique de relations linaires vrifies par les variables d'un programme*. Thse de 3e cycle, Universit Scientifique et Mdicale 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-Eric Proy, and Pascal Raymond. Verification of linear hybrid systems by means of convex approximations. In *SAS*, pages 223–237, 1994.
12. 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.
13. 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.
14. A.E. Hurd and P.A. Loeb. *An Introduction to Nonstandard Real Analysis*. Pure and Applied Mathematics. Elsevier Science, 1985.
15. Kengo Kido. *An Alternative Denotational Semantics for an Imperative Language with Infinitesimals*. Bachelor's thesis, The University of Tokyo: Japan, 2013.
16. A. Robinson. *Non-standard Analysis*. Studies in logic and the foundations of mathematics. North-Holland Pub. Co., 1966.
17. 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.

18. 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.
19. Glynn Winskel. *The Formal Semantics of Programming Languages: An Introduction*. MIT Press, Cambridge, MA, USA, 1993.

A Further on NSA in Superstructure

The definitions and results listed below are all well-established and commonly used in NSA. We follow [14, Chap. II], in which more details can be found.

Remark A.1 (choice of the index set I) In §2.1 we used the set $\mathbb{N}$ of natural numbers as the index set I. It is common in NSA, however, to use I that is bigger than $\mathbb{N}$, and an ultrafilter $\mathcal{F} \subseteq \mathcal{P}(I)$ over I. The merit of doing so is that the resulting monomorphism ${}^*(\_)$ (see below) can be chosen to be an *enlargement*; see [14, Chap. II]. In what follows, however, we favor concreteness and keep using $I = \mathbb{N}$ as the index set.

The transfer principle is a powerful result and we rely on it in the subsequent developments. Here are the first examples of its use; they are proved by transferring a suitable formula A.

Lemma A.2 *1. For $a \in V(X) \setminus X$ we obtain an injective map*

$$ {}^*(\_) \ : \ a \longrightarrow {}^*a \ , \quad (b \in a) \longmapsto ({}^*b \in {}^*a) \tag{4} $$

as a restriction of ${}^*(\_)$ in (3).

2. *If a is a finite set, the map (4) is an isomorphism $a \stackrel{\cong}{\Rightarrow} {}^*a$.*
3. *Let $a \to b$ be the set of functions from a to b. We have ${}^*(a \to b) \subseteq {}^*a \to {}^*b$.*
4. *${}^*(a_1 \times \cdots \times a_m) = {}^*a_1 \times \cdots \times {}^*a_m$; and ${}^*(a_1 \cup \cdots \cup a_m) = {}^*a_1 \cup \cdots \cup {}^*a_m$.*
5. *For a binary relation $r \subseteq a \times a$, we have ${}^*r \subseteq {}^*a \times {}^*a$. Moreover, r is an order if and only if ${}^*r$ is an order.* □

Internal Sets The distinction between *internal* and *external* entities is central in NSA. In this paper however it is much of formality, since all the entities we use are internal. Here we present only the relevant definitions, leaving their intuitions to [14, §II.6]. In Appendix B, especially Rem. B.7, we will see that being internal is crucial for transfer.

Definition A.3 (internal entity) An element $b \in V({}^*X)$ is *internal* with respect to ${}^*(\_) : V(X) \to V({}^*X)$ if there is $a \in V(X)$ such that $b \in {}^*a$. It is *external* if it is not internal.

Lemma A.4 *A function $f : {}^*a \to {}^*b$ is internal if and only if $f \in {}^*(a \to b)$.* □

The Ultrapower Construction We collect some necessary facts about the ultrapower construction of the monomorphism ${}^*(\_)$ in (3). Its details are beyond our scope; they are found in [14, §II.4].

The map ${}^*(\_)$ in fact factorizes into the following three steps.

$$ \begin{array}{ccc} V(X) & \xrightarrow{\quad {}^*(\_) \quad} & V({}^*X) \\ {\scriptstyle \overline{(\_)}}\downarrow & & \uparrow{\scriptstyle M} \\ \bigcup_{n\in\mathbb{N}} \big(V_n(X) \setminus V_{n-1}(X)\big)^I & \xrightarrow[\ [\_]\ ]{} & \prod^0_{\mathcal{F}} V(X) \end{array} \tag{5} $$

The first factor $\overline{(\_)}$ maps $a \in V(X)$ to the constant function $\overline{a}$ such that $\overline{a}(i) = a$ for each $i \in I$; recall that we have chosen $I = \mathbb{N}$ (Rem. A.1). The second $[\_]$ takes a quotient modulo the ultrafilter $\mathcal{F}$; finally the third factor M is the so-called *Mostowski collapse*.

For an intuition let us exhibit these maps in the simple setting of §2.1. The first factor $\overline{(\_)}$ corresponds to forming constant streams: $a \mapsto \overline{a} = (a, a, \dots)$. The second $[\_]$ is quotienting modulo $\sim_{\mathcal{F}}$ of (2). The third map M does nothing—it is a book-keeping function that is only needed in the extended setting of superstructures.

The next result [14, Thm. 4.5] is about "starting from the lower-left corner" in (5). It follows from the definition of M and is a crucial step in the proof of the transfer principle (Lem. 2.7). It serves as an important lemma, too, later for the semantics of WHILE$^{\mathsf{dt}}$.

Lemma A.5 (Łoś' theorem) *Let A be a formula in $\mathscr{L}_X$ with its free variables contained in $\{x_1, \dots, x_m\}$; and $a_1, \dots, a_m \in \bigcup_{n\in\mathbb{N}} \big(V_n(X) \setminus V_{n-1}(X)\big)^I$. Then*

$$\begin{aligned} &{}^*A\big[\, M[a_1]/x_1, \dots, M[a_m]/x_m \,\big] \text{ is valid} \\ \iff\quad &\big\{ i \in I \mid A[a_1(i)/x_1, \dots, a_m(i)/x_m] \text{ is valid} \big\} \in \mathcal{F} \ . \end{aligned}$$

As a special case, let $S \in V(X)$, then

$$M[a] \in {}^*S \quad\iff\quad a(i) \in S \text{ for almost every } i. \qquad \square$$

Corollary A.6 *Let $a, b \in V(X)$; and for each $i \in I$, $f_i \in (a \to b)$ and $x_i \in a$. Then $M[(f_i)_{i\in I}]$ is an internal function ${}^*a \to {}^*b$; and $M[(x_i)_{i\in I}] \in {}^*a$. Moreover,*

$$M[(f_i(x_i))_{i\in I}] = \Big(M[(f_i)_{i\in I}]\Big)\Big(M[(x_i)_{i\in I}]\Big) \ . \qquad \square$$

B Appendix: Domain Theory, Transferred

The semantics of WHILE$^{\mathsf{dt}}$ is most naturally introduced by solving recursive equations on the flat domain over hyperreals. Here we present necessary theoretical foundations—they are like in [3, §2.2] and [18]—identifying the set ${}^*\mathbb{R} \cup \{\bot\}$ as a *hyperdomain* and *-transferring domain theory.

The current section is an adaptation is what appeared in the appendix of [18]; and the definitions and results are similar to those in [3, §2.2], where what we call a hyperdomain is called an *internal domain*, and a *-continuous function is called an *internal continuous function*. The way we formulate these notions is however a bit different: we favor more explicit use of *-transforms, since this aids deductive verification via the transfer principle.

Definition B.1 In what follows we employ the theory of NSA presented in Appendix A. As the base set of a superstructure $V(X)$ (Def. 2.2), we take $X = \mathbb{R} \cup \mathbb{B} \cup \mathbf{Var}$.

Definition B.2 (hyperdomain) A *hyperdomain* is the pair of *-transforms $({}^*D, {}^*\sqsubseteq)$ of a cpo $(D, \sqsubseteq)$.

Example B.3 The set $\mathcal{P}(\mathbf{Var} \to \mathbb{R})$ is a complete lattice with respect to the inclusion order $\subseteq$, therefore is a cpo. Its $*$-transfer $\big({}^*\big(\mathcal{P}(\mathbf{Var} \to \mathbb{R})\big),\ {}^*\subseteq\big)$ constitutes a hyperdomain.

We note that the set ${}^*\big(\mathcal{P}(\mathbf{Var} \to \mathbb{R})\big)$ coincides with the set of internal subsets of the space $\{f : {}^*\mathbf{Var} \to {}^*\mathbb{R} \mid f \text{ is an internal function}\}$. Moreover, under the assumption that $\mathbf{Var}$ is a finite set (e.g. the set of variables occurring in a program c), we can see that the last set $\{f : {}^*\mathbf{Var} \to {}^*\mathbb{R} \mid f \text{ is an internal function}\}$ coincides with the function space $\mathbf{Var} \to {}^*\mathbb{R}$. For this we use Lem. A.2.4.

Note that ${}^*\sqsubseteq$ is an order in ${}^*D$ (Lem. A.2.5). Hyperdomain is the notion on which we wish to establish a suitable fixed point property.[5] Towards that goal, we first formulate the definitions of cpo and continuous function as $\mathscr{L}_X$-formulas, so that they can be transferred.

$$
\begin{aligned}
&\mathsf{BinRel}_{a,r} :\equiv\ r \subseteq a \times a \qquad \mathsf{Refl}_{a,r} :\equiv\ \forall x \in a.\, (x,x) \in r\\
&\mathsf{Trans}_{a,r} :\equiv\ \forall x,y,z \in a.\, \big((x,y) \in r \wedge (y,z) \in r \Rightarrow (x,z) \in r \big)\\
&\mathsf{AntiSym}_{a,r} :\equiv\ \forall x,y \in a.\, \big((x,y) \in r \wedge (y,x) \in r \Rightarrow x = y \big)\\
&\mathsf{Poset}_{a,r} :\equiv\ \mathsf{BinRel}_{a,r} \wedge \mathsf{Refl}_{a,r} \wedge \mathsf{Trans}_{a,r} \wedge \mathsf{AntiSym}_{a,r}\\
&\mathsf{HasBot}_{a,r} :\equiv\ \exists x \in a. \forall y \in a.\, (x,y) \in r\\
&\mathsf{AscCn}_{a,r}(s) :\equiv\ \forall x, x' \in \mathbb{N}.\, (x \le x' \Rightarrow (s(x), s(x')) \in r)\\
&\mathsf{UpBd}_{a,r}(b,s) :\equiv\ \forall x \in \mathbb{N}.\, ((s(x), b) \in r)\\
&\mathsf{Sup}_{a,r}(p,s) :\equiv\ \mathsf{UpBd}_{a,r}(p,s) \wedge \forall b \in a.\, (\mathsf{UpBd}_{a,r}(b,s) \Rightarrow (p,b) \in r)
\end{aligned}
$$

Recall that the inclusion $\mathbb{N} \subseteq X$ is assumed (Def. 2.2). These formulas are used in:

$$
\begin{aligned}
\mathsf{CPO}_{a,r} :\equiv\ & \mathsf{Poset}_{a,r} \wedge \mathsf{HasBot}_{a,r} \wedge\\
& \forall s \in (\mathbb{N} \to a).\, \big(\mathsf{AscCn}_{a,r}(s) \Rightarrow \exists p \in a.\, \mathsf{Sup}_{a,r}(p,s) \big)\ ,\\
\mathsf{Conti}_{a_1,r_1,a_2,r_2}(f) :\equiv\ & \forall s \in (\mathbb{N} \to a_1). \forall p \in a_1.\\
& \big(\mathsf{AscCn}_{a_1,r_1}(s) \wedge \mathsf{Sup}_{a_1,r_1}(p,s) \Rightarrow \mathsf{Sup}_{a_2,r_2}(f(p), f \circ s) \big)\ .
\end{aligned}
\tag{6}
$$

Definition B.4 (*-continuous function) Let $({}^*D_1, {}^*\sqsubseteq_1)$ and $({}^*D_2, {}^*\sqsubseteq_2)$ be hyperdomains. A function $f : {}^*D_1 \to {}^*D_2$ is **-continuous* if it is internal and satisfies the *-transform of the formula $\mathsf{Conti}_{D_1,\sqsubseteq_1,D_2,\sqsubseteq_2}$. That is to be precise: ${}^*(\mathsf{Conti}_{D_1,\sqsubseteq_1,D_2,\sqsubseteq_2})(f)$ is valid.[6] The set of *-continuous functions from ${}^*D_1$ to ${}^*D_2$ is denoted by ${}^*D_1 \to_{*\mathrm{ct}} {}^*D_2$.

Lemma B.5 $({}^*D_1 \to_{*\mathrm{ct}} {}^*D_2) = {}^*(D_1 \to_{\mathrm{ct}} D_2)$. *Here* $\to_{\mathrm{ct}}$ *denotes the set of continuous functions.*

---

[5] We believe an even more general setting is possible, by defining a hyperdomain to be an internal set $D' \in V({}^*X)$ that satisfies a suitable formula like $\mathsf{CPO}_{a,r}$ in (6). Here we do not need such generality.

[6] We note that the condition is different from (somewhat informal) "${}^*\mathsf{Conti}_{{}^*D_1,{}^*\sqsubseteq_1,{}^*D_2,{}^*\sqsubseteq_2}(f)$ is valid." In the former a chain s ranges over internal functions $s \in {}^*(\mathbb{N} \to D_1)$, while in the latter s can also be an external function ${}^*\mathbb{N} \to {}^*D_1$.

Proof. Assume $f \in {}^*(D_1 \to_{\mathrm{ct}} D_2)$. The following closed formula is valid in $V(X)$:

$$\forall f' \in (D_1 \to D_2).\, \big(f' \in (D_1 \to_{\mathrm{ct}} D_2) \Leftrightarrow \mathsf{Conti}(f')\big) \ ,$$

where Conti is short for $\mathsf{Conti}_{D_1,\sqsubseteq_1,D_2,\sqsubseteq_2}$. By transfer we have

$$\forall f' \in {}^*(D_1 \to D_2).\, \big(f' \in {}^*(D_1 \to_{\mathrm{ct}} D_2) \Leftrightarrow {}^*\mathsf{Conti}(f')\big) \tag{7}$$

valid in $V({}^*X)$. Thus f satisfies ${}^*\mathsf{Conti}(f')$. Obviously f is internal; therefore $f \in ({}^*D_1 \to_{{}^*\mathrm{ct}} {}^*D_2)$.

Conversely, assume $f \in ({}^*D_1 \to_{{}^*\mathrm{ct}} {}^*D_2)$. By the definition of *-continuity, f is internal, hence by Lem. A.4 we have $f \in {}^*(D_1 \to D_2)$. Moreover, using the definition of *-continuity and (7), we have $f \in {}^*(D_1 \to_{\mathrm{ct}} D_2)$. □

Lemma B.6 *Let $({}^*D, {}^*\sqsubseteq)$ be a hyperdomain. Then a *-continuous function $f : {}^*D \to {}^*D$ has a least fixed point. Moreover, the function ${}^*\mu$ that maps f to its least fixed point $({}^*\mu)(f)$ is *-continuous.*

Proof. By the usual construction in a cpo, we obtain the map

$$\mu \ : \ (D \to_{\mathrm{ct}} D) \to_{\mathrm{ct}} D \ , \quad f \mapsto \textstyle\bigsqcup_{n\in\mathbb{N}} f^n(\bot) \ .$$

Continuity of μ is easy and standard. As its *-transform we obtain a function ${}^*\mu : ({}^*D \to_{{}^*\mathrm{ct}} {}^*D) \to_{{}^*\mathrm{ct}} {}^*D$, where we used Lem. B.5 and A.2. The fact that ${}^*\mu$ returns least fixed points is shown by the transfer of the following $\mathscr{L}_X$-formula.

$$\forall f \in (D \to_{\mathrm{ct}} D).\, \big(\ f(\mu(f)) = \mu(f) \ \wedge \forall x \in D.\, (f(x) = x \ \Rightarrow \mu(f) \sqsubseteq x)\ \big) \qquad \square$$

Remark B.7 It is crucial in the previous lemma that $f : {}^*D \to {}^*D$ is an internal function. Specifically: recall that a formula A must be closed in order to be transferred (Lem. 2.7); and that in $\mathscr{L}_X$ only bounded quantifiers ($\forall x \in s$ with some bound s) are allowed. For internal f we find such a bound by $f \in {}^*(D \to D)$; for external f this is not possible.

C Appendix: Details of the Water-Level Monitor Example

As preparation, we represent the WHILE$^{\mathtt{dt}}$ program shown in §4 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.

In this section, 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 ("A0-B0"). 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 ("D2"). 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

Fig. 2. Control flow graph of water-level monitor

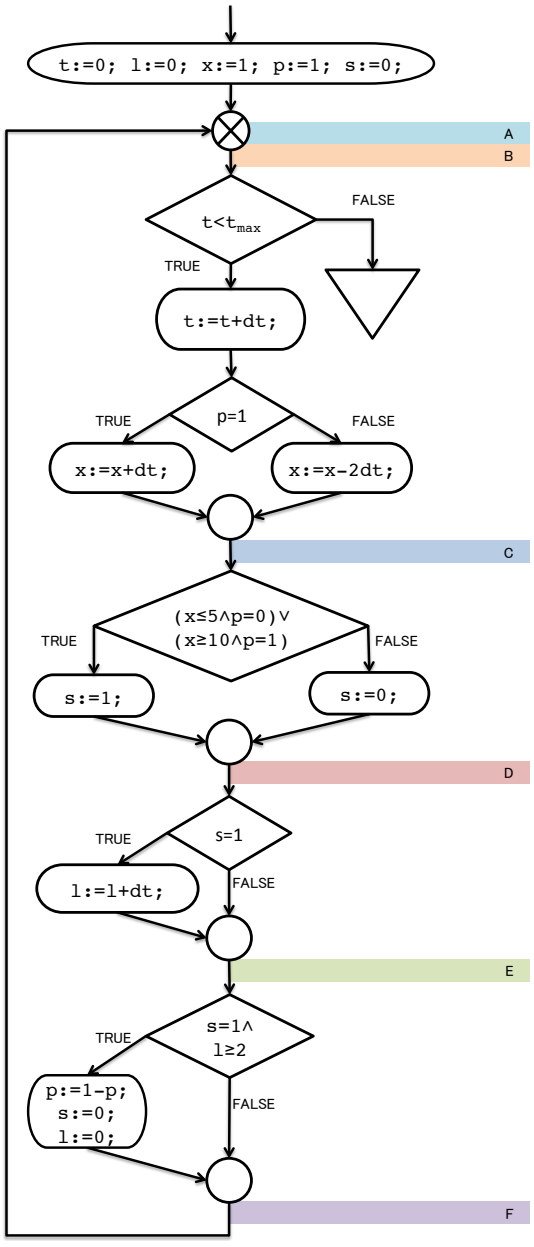


Fig. 3. p=1 and s=0

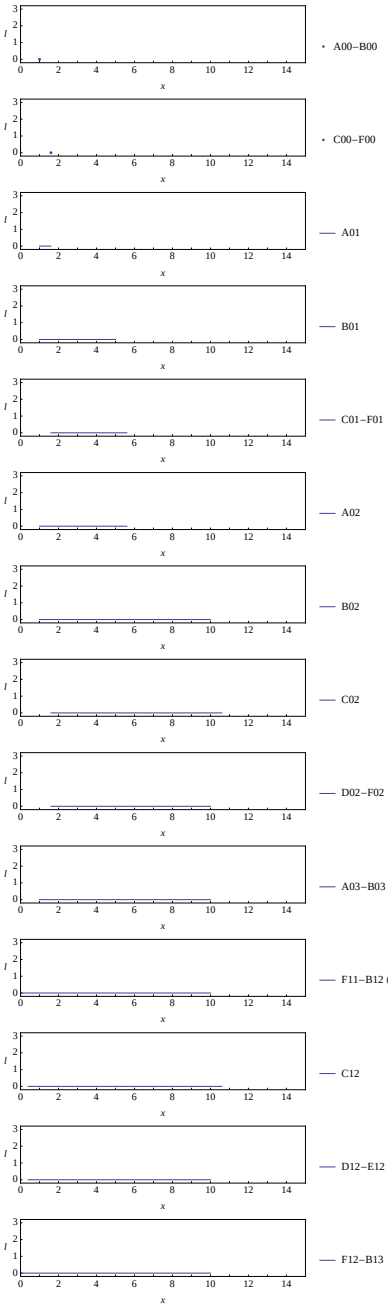


Fig. 4. p=1 and s=1

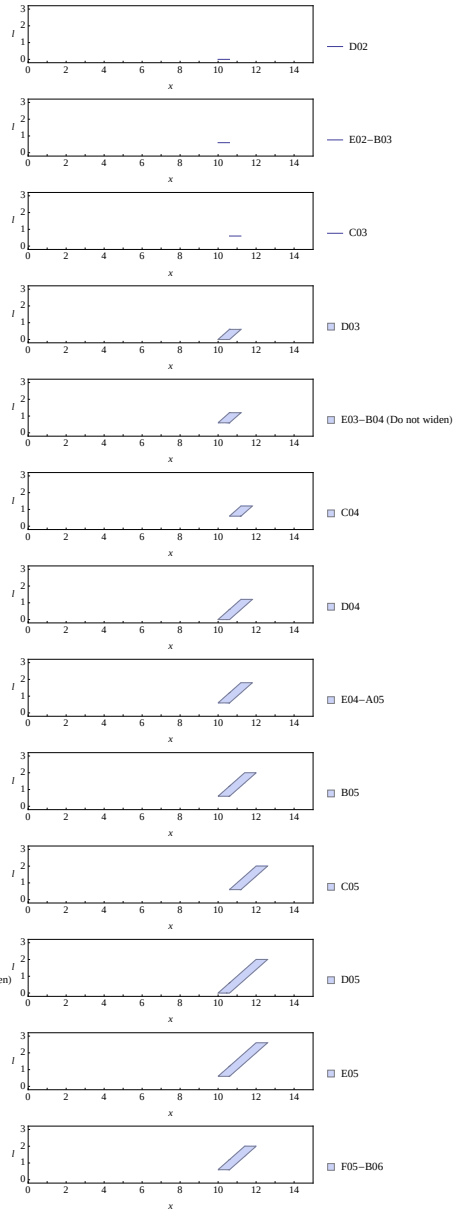


Fig. 5. p=0 and s=0

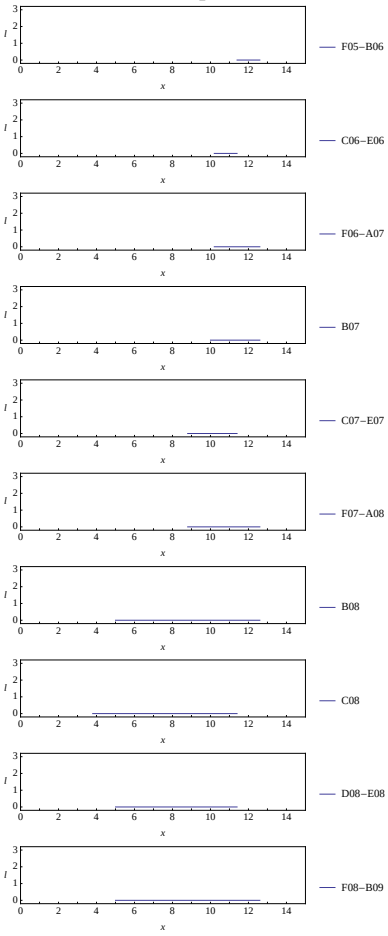
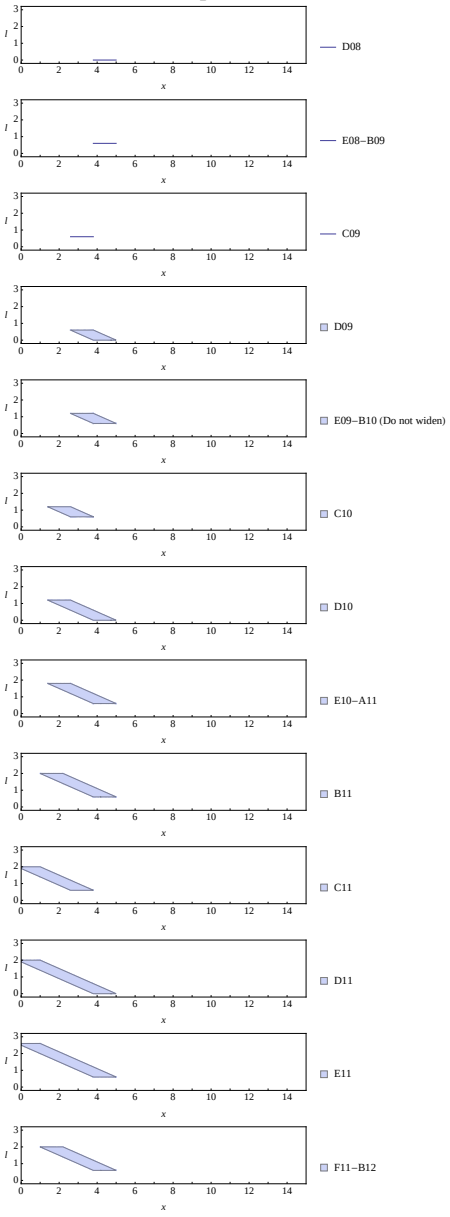


Fig. 6. p=0 and s=1



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.

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 $A_0 = \mathsf{con}\{p = 1, s = 0, x = 1, l = 0\}$. Then we compute the states as follows:

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

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 $1 - \mathtt{dt} \leq x \leq 12 + \mathtt{dt}$.

D Appendix: Omitted Proofs

In the proofs we use the following $\mathscr{L}_{\mathbb{R}}$-formulas.

Definition D.1 We define the following $\mathscr{L}_{\mathbb{R}}$-formulas:

$$
\begin{aligned}
&\mathsf{Refl}_{L,\sqsubseteq} :\equiv \ \forall l \in L.\ (l, l) \in \sqsubseteq \\
&\mathsf{Trans}_{L,\sqsubseteq} :\equiv \ \forall l, m, n \in L.\ \Big(\big((l, m) \in \sqsubseteq \land (m, n) \in \sqsubseteq\big) \Rightarrow (l, m) \in \sqsubseteq\Big) \\
&\mathsf{AntiSym}_{L,\sqsubseteq} :\equiv \ \forall l, m \in L.\ \Big(\big((l, m) \in \sqsubseteq \land (m, l) \in \sqsubseteq\big) \Rightarrow l = m\Big) \\
&\mathsf{Poset}_{L,\sqsubseteq} :\equiv \ \mathsf{Refl}_{L,\sqsubseteq} \land \mathsf{Trans}_{L,\sqsubseteq} \land \mathsf{AntiSym}_{L,\sqsubseteq} \\
&\mathsf{Galois}_{L_1,\sqsubseteq_1,L_2,\sqsubseteq_2,\alpha,\gamma} :\equiv \ \forall x \in L_1.\ \forall \overline{y} \in L_2.\ \big(\alpha(x) \sqsubseteq_2 \overline{y} \Leftrightarrow x \sqsubseteq_1 \gamma(\overline{y})\big) \\
&\mathsf{AscCn}_{L,\sqsubseteq}(s) :\equiv \ \forall n, m \in \mathbb{N}.\ \big(n \leq m \Rightarrow s(n) \sqsubseteq s(m)\big) \\
&\mathsf{Sup}_{L,\sqsubseteq}(p, s) :\equiv \ \big(\forall n \in \mathbb{N}.\ s(n) \sqsubseteq p\big) \land \forall q \in L.\Big(\big(\forall n \in \mathbb{N}.\ s(n) \sqsubseteq q\big) \Rightarrow p \sqsubseteq q\Big) \\
&\mathsf{Cpo}_{L,\sqsubseteq} :\equiv \ \mathsf{Poset}_{L,\sqsubseteq} \land \forall s \in \mathbb{N} \to L.\ \big(\mathsf{AscCn}_{L,\sqsubseteq}(s) \Rightarrow \exists p \in L.\ \mathsf{Sup}_{L,\sqsubseteq}(p, s)\big) \\
&\mathsf{Monotone}_{L_1,\sqsubseteq_1,L_2,\sqsubseteq_2}(f) :\equiv \ \forall x, y \in L_1.\ x \sqsubseteq_1 y \Rightarrow f(x) \sqsubseteq_2 f(y)
\end{aligned}
$$

$$\mathsf{Conti}_{L_1,\sqsubseteq_1,L_2,\sqsubseteq_2}(f) :\equiv \forall s \in \mathbb{N} \to L_1.\ \forall p \in L_1.$$
$$\Big(\big(\mathsf{AscCn}_{L_1,\sqsubseteq_1}(s) \wedge \mathsf{Sup}_{L_1,\sqsubseteq_1}(p,s)\big) \Rightarrow \mathsf{Sup}_{L_2,\sqsubseteq_2}\big(f(p), f \circ s\big)\Big)$$
$$\mathsf{Basis}_{L,\sqsubseteq}(\bot, f) :\equiv\ \bot \sqsubseteq f(\bot)$$
$$\mathsf{Cover}_{L,\sqsubseteq,\nabla} :\equiv\ \forall x, y \in L.\ (x \sqsubseteq x \nabla y) \wedge y \sqsubseteq x \nabla y)$$
$$\mathsf{Term}_{L,\sqsubseteq,\nabla} :\equiv\ \forall x \in \mathbb{N} \to L.\ \mathsf{AscCn}(x) \Rightarrow$$
$$\Big(\forall y \in \mathbb{N} \to L.\ \Big(\big(y(0) = x(0) \wedge \forall n \in \mathbb{N}.\ y(n+1) = y(n) \nabla x(n+1)\big)$$
$$\Rightarrow \exists k \in \mathbb{N}.\ y(k) = y(k+1)\Big)\Big)$$
$$\mathsf{Widen}_{L,\sqsubseteq,\nabla} :\equiv\ \mathsf{Cover}_{L,\sqsubseteq,\nabla} \wedge \mathsf{Term}_{L,\sqsubseteq,\nabla}$$
$$\mathsf{WidenSeq}_{L,\sqsubseteq,\nabla}(X, \bot, F) :\equiv$$
$$X(0) = \bot\ \wedge\ \forall n \in \mathbb{N}.\ X(n+1) = X(n) \nabla F(X(n)).$$

D.1 Proof of Prop. 3.4

Proof. The constraint system C for a (standard) convex polyhedron $P \in \mathbb{CP}_n$ can be expressed by a pair $(\mathbf{A}, \mathbf{b})$ of an $m \times n$-matrix $\mathbf{A}$ and an m-vector $\mathbf{b}$, where m is the number of linear inequalities in C. The same applies to a (nonstandard) convex polyhedron $P \in {}^*\mathbb{CP}_n^{{}^*\mathbb{R}}$. For each of $X \in \{\mathbb{R}, {}^*\mathbb{R}\}$, let us denote, by $\mathsf{Constr}_{X,m,n}$, the set of all convex polyhedra over X^n that can be expressed with m linear inequalities.

Then $\mathbb{CP}_n = \bigcup_{m \in \mathbb{N}} \mathsf{Constr}_{\mathbb{R},m,n}$ (with $\bigcup_{m \in \mathbb{N}}$ expressed using an existential quantifier $\exists m \in \mathbb{N}$) is a valid $\mathscr{L}_{\mathbb{R}}$-sentence by Def. 2.19. By the transfer principle (Lem. 2.7), we have a valid $\mathscr{L}_{{}^*\mathbb{R}}$-sentence ${}^*(\mathbb{CP}_n) = \bigcup_{m \in {}^*\mathbb{N}} \mathsf{Constr}_{{}^*\mathbb{R},m,n}$. It has as its subset the set ${}^*\mathbb{CP}_n^{{}^*\mathbb{R}} = \bigcup_{m \in \mathbb{N}} \mathsf{Constr}_{{}^*\mathbb{R},m,n}$ since $\mathbb{N} \subseteq {}^*\mathbb{N}$. This proves the claim. □

D.2 Proof of Thm. 3.6

Proof. Let $L, \overline{L} \in \mathbb{U}$ be sets, $\sqsubseteq \in \mathcal{P}(L \times L)$ and $\overline{\sqsubseteq} \in \mathcal{P}(\overline{L} \times \overline{L})$ be binary relations on L and $\overline{L}$ respectively, $\alpha : L \to \overline{L}$ and $\gamma : \overline{L} \to L$ be functions. Then, the following $\mathscr{L}_{\mathbb{R}}$-sentence is valid (because it is equivalent to Prop. 2.12):

$$\forall F \in L \to L.\ \forall \overline{F} \in \overline{L} \to \overline{L}.\ \forall \bot \in L.\ \forall \overline{\bot} \in \overline{L}.\ \forall \overline{x} \in \overline{L}.$$
$$\Big(\mathsf{Cpo}_{L,\sqsubseteq} \wedge \mathsf{Poset}_{\overline{L},\overline{\sqsubseteq}} \wedge \mathsf{Conti}_{L,\sqsubseteq,L,\sqsubseteq}(F) \wedge \mathsf{Monotone}_{\overline{L},\overline{\sqsubseteq},\overline{L},\overline{\sqsubseteq}}(\overline{F}) \wedge \mathsf{Galois}_{L,\sqsubseteq,\overline{L},\overline{\sqsubseteq},\alpha,\gamma}$$
$$\wedge\ F \sqsubseteq \vec{\gamma}(\overline{F}) \wedge \bot \sqsubseteq F(\bot) \wedge \bot \sqsubseteq \gamma(\overline{\bot}) \wedge \overline{\bot} \mathrel{\overline{\sqsubseteq}} \overline{x} \wedge \overline{F}(\overline{x}) \mathrel{\overline{\sqsubseteq}} \overline{x}$$
$$\Rightarrow \mathrm{lfp}_{\bot} F \sqsubseteq \gamma(\overline{x})\Big).$$

By applying Lem. 2.7 to this $\mathscr{L}_{\mathbb{R}}$-sentence, we have the following valid $\mathscr{L}_{{}^*\mathbb{R}}$-sentence:

$$
\begin{aligned}
&\forall F \in {}^*(L \to L).\ \forall \overline{F} \in {}^*(\overline{L} \to \overline{L}).\ \forall \bot \in {}^*L.\ \forall \overline{\bot} \in {}^*\overline{L}.\ \forall \overline{x} \in {}^*\overline{L}.\\
&\Big({}^*\mathsf{Cpo}_{L,\sqsubseteq} \wedge {}^*\mathsf{Poset}_{\overline{L},\overline{\sqsubseteq}} \wedge {}^*\mathsf{Conti}_{L,\sqsubseteq,L,\sqsubseteq}(F) \wedge {}^*\mathsf{Monotone}_{\overline{L},\overline{\sqsubseteq},\overline{L},\overline{\sqsubseteq}}(\overline{F}) \wedge {}^*\mathsf{Galois}_{L,\sqsubseteq,\overline{L},\overline{\sqsubseteq},\alpha,\gamma}\\
&\wedge\ F\ {}^*\sqsubseteq\ {}^*\vec{\gamma}(\overline{F}) \wedge \bot\ {}^*\sqsubseteq\ F(\bot) \wedge \bot\ {}^*\sqsubseteq\ {}^*\gamma(\overline{\bot}) \wedge \overline{\bot}\ {}^*\overline{\sqsubseteq}\ \overline{x} \wedge \overline{F}(\overline{x})\ {}^*\overline{\sqsubseteq}\ \overline{x}\\
&\Rightarrow {}^*\mathrm{lfp}_{\bot} F\ {}^*\sqsubseteq\ {}^*\gamma(\overline{x})\Big).
\end{aligned}
$$

This yields the statement of this theorem. For example, we can confirm that ${}^*\mathsf{Galois}_{\mathsf{L},\sqsubseteq,\overline{\mathsf{L}},\overline{\sqsubseteq},\alpha,\gamma}$ holds, that is, $({}^*L, {}^*\overline{L}, {}^*\alpha, {}^*\gamma)$ is a hyper-Galois connection, from the hypothesis $L \underset{\gamma}{\overset{\alpha}{\rightleftharpoons}} \overline{L}$ in the theorem statement . □

D.3 Proof of Thm. 3.10

Proof. Let $L \in \mathbb{U}$ be a set, $\sqsubseteq \in \mathcal{P}(L \times L)$ be a binary relation on L and $\nabla : L \times L \to L$ be a function. Then, the following $\mathscr{L}_{\mathbb{R}}$-sentence is valid (because it is equivalent to Prop. 2.14):

$$
\begin{aligned}
&\forall F \in L \to L.\ \forall \bot \in L.\ \forall X \in \mathbb{N} \to L.\\
&\mathsf{Poset}_{L,\sqsubseteq} \wedge \mathsf{Monotone}_{L,\sqsubseteq,L,\sqsubseteq}(F) \wedge \mathsf{Basis}_{L,\sqsubseteq}(\bot, F) \wedge \mathsf{Widen}_{L,\sqsubseteq,\nabla}\\
&\wedge \mathsf{WidenSeq}_{L,\sqsubseteq,\nabla}(X, \bot, F)\\
&\Rightarrow \exists i \in \mathbb{N}.\ \forall j \in \mathbb{N}.\ i \leq j \Rightarrow X(i) = X(j)\\
&\wedge \forall k \in \mathbb{N}.\ \Big(\big(\forall l \in \mathbb{N}.\ k \leq l \Rightarrow X(k) = X(l)\big) \Rightarrow F\big(X(k)\big) \sqsubseteq X(k)\Big).
\end{aligned}
$$

By applying Lem. 2.7 to this $\mathscr{L}_{\mathbb{R}}$-sentence, we have the following valid $\mathscr{L}_{\mathbb{R}}$-sentence:

$$
\begin{aligned}
&\forall F \in {}^*(L \to L).\ \forall \bot \in {}^*L.\ \forall X \in {}^*(\mathbb{N} \to L).\\
&{}^*\mathsf{Poset}_{L,\sqsubseteq} \wedge {}^*\mathsf{Monotone}_{L,\sqsubseteq,L,\sqsubseteq}(F) \wedge {}^*\mathsf{Basis}_{L,\sqsubseteq}(\bot, F) \wedge {}^*\mathsf{Widen}_{L,\sqsubseteq,\nabla}\\
&\wedge {}^*\mathsf{WidenSeq}_{L,\sqsubseteq,\nabla}(X, \bot, F)\\
&\Rightarrow \exists i \in {}^*\mathbb{N}.\ \forall j \in {}^*\mathbb{N}.\ i \leq j \Rightarrow X(i) = X(j)\\
&\wedge \forall k \in {}^*\mathbb{N}.\ \Big(\big(\forall l \in {}^*\mathbb{N}.\ k \leq l \Rightarrow X(k) = X(l)\big) \Rightarrow F\big(X(k)\big)\ {}^*\sqsubseteq\ X(k)\Big)
\end{aligned}
$$

This yields the statement of this theorem. Note that the well-definedness of the iteration hyper-sequence (by induction on $i \in {}^*\mathbb{N}$) is implicit in the above transfer arguments.

□

D.4 Proof of Thm. 3.12

Proof. We can characterize uniform widening operators as an $\mathscr{L}_{\mathbb{R}}$-sentence as follows (covering condition has been already expressed as an $\mathscr{L}_{\mathbb{R}}$-formula in Def. D.1):

$$\begin{aligned}
&\mathsf{UnifTerm}_{L,\sqsubseteq,\nabla} :\equiv\ \forall x_0 \in L.\ \exists i \in \mathbb{N}.\ \forall x \in \mathbb{N} \to L.\ (\mathsf{AscCn}(x) \wedge x(0) = x_0) \Rightarrow \\
&\quad \Big(\forall y \in \mathbb{N} \to L.\ \Big(\big(y(0) = x(0) \wedge \forall n \in \mathbb{N}.\ y(n+1) = y(n) \nabla x(n+1)\big) \\
&\quad \Rightarrow \exists j \in \mathbb{N}.\ \big(j \leq i \wedge y(j) = y(j+1)\big) \Big) \Big) \\
&\mathsf{UnifWiden}_{L,\sqsubseteq,\nabla} :\equiv\ \mathsf{Cover}_{L,\sqsubseteq,\nabla} \wedge \mathsf{UnifTerm}_{L,\sqsubseteq,\nabla}
\end{aligned}$$

Let $L \in \mathbb{U}$ be a set, $\sqsubseteq \in \mathcal{P}(L \times L)$ be a binary relation on L and $\nabla : L \times L \to L$ be a function. Then, we can see directly that the following $\mathscr{L}_{\mathbb{R}}$-sentence is valid:

$$\forall \bot \in L.\ \exists i \in \mathbb{N}.\ \Psi(\bot)(i), \tag{8}$$

where

$$\begin{aligned}
&\Psi(\bot)(i) = \\
&\quad \forall F \in L \to L.\ \forall X \in \mathbb{N} \to L. \\
&\quad \mathsf{Poset}_{L,\sqsubseteq} \wedge \mathsf{Monotone}_{L,\sqsubseteq,L,\sqsubseteq}(F) \wedge \mathsf{Basis}_{L,\sqsubseteq}(\bot, F) \wedge \mathsf{UnifWiden}_{L,\sqsubseteq,\nabla} \\
&\quad \wedge \mathsf{WidenSeq}_{L,\sqsubseteq,\nabla}(X, \bot, F) \\
&\quad \Rightarrow \forall j \in \mathbb{N}.\ i \leq j \Rightarrow X(i) = X(j) \\
&\quad \wedge \forall k \in \mathbb{N}.\ \Big(\big(\forall l \in \mathbb{N}.\ k \leq l \Rightarrow X(k) = X(l)\big) \Rightarrow F\big(X(k)\big) \sqsubseteq X(k) \Big).
\end{aligned}$$

Assume that $\bot \in L$ is given. Then, by the $\mathscr{L}_{\mathbb{R}}$-sentence (8), there exists $i \in \mathbb{N}$ such that $\Psi(\bot)(i)$ holds. Therefore, by transferring $\Psi(\bot)(i)$, ${}^*\Psi(\bot)(i)$ holds for such $i \in \mathbb{N}$. Note that ${}^*\Psi(\bot)(i)$ is the following $\mathscr{L}_{{}^*\mathbb{R}}$-sentence ($\bot$ and i are dealt with as constants in the following $\mathscr{L}_{{}^*\mathbb{R}}$-sentence because $\bot$ and i are defined outside the $\mathscr{L}_{{}^*\mathbb{R}}$-sentence):

$$\begin{aligned}
&\forall F \in {}^*(L \to L).\ \forall X \in {}^*(\mathbb{N} \to L). \\
&{}^*\mathsf{Poset}_{L,\sqsubseteq} \wedge {}^*\mathsf{Monotone}_{L,\sqsubseteq,L,\sqsubseteq}(F) \wedge {}^*\mathsf{Basis}_{L,\sqsubseteq}({}^*\bot, F) \wedge {}^*\mathsf{UnifWiden}_{L,\sqsubseteq,\nabla} \\
&\wedge {}^*\mathsf{WidenSeq}_{L,\sqsubseteq,\nabla}(X, {}^*\bot, F) \\
&\Rightarrow \forall j \in {}^*\mathbb{N}.\ i \leq j \Rightarrow X(i) = X(j) \\
&\wedge \forall k \in {}^*\mathbb{N}.\ \Big(\big(\forall l \in {}^*\mathbb{N}.\ k \leq l \Rightarrow X(k) = X(l)\big) \Rightarrow F\big(X(k)\big)\ {}^*\sqsubseteq\ X(k) \Big).
\end{aligned}$$

This yields Thm. 3.12. □

D.5 Proof of Thm. 3.13

We prove the uniformity and nonuniformity of five widening operators (∇_{Intv}, $\nabla_{\mathsf{Intv}_{\mathbb{Z},c}}$, ∇_S, ∇_M, ∇_N) in this order.

Proof. Let $\langle X_i \rangle_i$ be a iteration sequence defined by ∇_{Intv}, a basis $[l_0, u_0]$ and a monotone function F. By definition of ∇_{Intv} and the construction of $\langle X_i \rangle_i$, regardless of the function F, $\langle X_i \rangle_i$ consists of at most 4 elements: $[l_0, u_0]$, $[-\infty, u_0]$, $[l_0, \infty]$ and $[-\infty, \infty]$. Thus for any basis $[l_0, u_0]$ and monotone function F, we can reach a prefixed point by iterating the widening operator at most 3 times and this means the widening operator ∇_{Intv} is uniform. □

Proof. Let $d \in \mathbb{Z}$ be a constant less than c that is used in the definition of $\nabla_{\mathsf{Intv}_{\mathbb{Z},c}}$. And let $\langle x_i \rangle \in \mathsf{Intv}_{\mathbb{Z}}^{\mathbb{N}}$ be an ascending chain defined by $x_k = [d + k, d + k]$. Then, it takes $c - d$ steps for the convergence of the iteration sequence defined using $\nabla_{\mathsf{Intv}_{\mathbb{Z},c}}$. So we can choose $d \in \mathbb{Z}$ so that it takes arbitrary large number of steps. This yields that this widening operator is not uniform. □

Proof. Let $d \in \mathbb{Z}$ be a constant less than c that is used in the definition of $\nabla_{\mathsf{Intv}_{\mathbb{Z},c}}$. And let $\langle x_i \rangle \in \mathsf{Intv}_{\mathbb{Z}}^{\mathbb{N}}$ be an ascending chain defined by $x_k = [d + k, d + k]$. Then, it takes $c - d$ steps for the convergence of the iteration sequence defined using $\nabla_{\mathsf{Intv}_{\mathbb{Z},c}}$. So we can choose $d \in \mathbb{Z}$ so that it takes arbitrary large number of steps. This yields that this widening operator is not uniform. □

Proof. The constraints in M may be added in the iteration sequence, but by the definition of the standard widening ∇_S, a constraint in M will never appear once it is violated. Therefore the number of steps for an iteration sequence to converge is at most $\#(M)$ larger than the case of standard widening. □

Proof. Assume that $P_1 = \mathsf{con}\{0 \leq x \leq 1, 0 \leq y \leq 1, z = 0\} \in \mathbb{CP}_3$, $P_2 \in \mathbb{CP}_3$ includes P_1 and the linear equation "$z = 0$" is not included in C_2. Then $P_1 \curvearrowright_N P_2$ holds because $\#eq(C_1) > \#eq(C_2)$. The maximum number of steps for an iteration sequence starting from P_2 to converge is $\#C_2$. This is not limited uniformly because you can define P_2 such that $\#C_2$ is as large as you like. □