Abstract Lipschitz Continuity

Marco Campion
Inria & ENS Paris | Université PSL, France

Isabella Mastroeni
University of Verona, Italy

Michele Pasqua
University of Verona, Italy

Caterina Urban
Inria & ENS Paris | Université PSL, France

Abstract

We introduce *Abstract Lipschitz Continuity* (ALC), a generalization of standard Lipschitz Continuity, that ensures proportionally bounded differences in the *semantic approximations* of outputs when the *semantic approximations* of inputs differ slightly. ALC distinguishes between two complementary notions of approximation: *quantitative* differences, expressed via pre-metrics, and *qualitative* (or *semantic*) differences, captured through upper closure operators. ALC allows for reasoning about bounded changes in output properties in settings where standard Lipschitz continuity is too restrictive or inapplicable, such as in program analysis and verification, where understanding semantic properties of inputs and outputs is of key importance.

In the specific context of programs, we formally relate ALC to other well-established program properties, including (Partial) Completeness and (Abstract) program Robustness. Notably, we show that ALC is a stronger requirement than Partial Completeness, a consolidated notion modeling precision loss in program analysis.

Finally, we propose a language- and domain-agnostic deductive system, parametric on the quantitative and semantic approximations of interest, for proving the ALC of programs. The goal in designing this deductive system is to track the assumptions required for ALC to ensure a compositional proof.

2012 ACM Subject Classification Theory of computation → Program analysis; Theory of computation → Abstraction; Theory of computation → Program verification

Keywords and phrases Abstract Lipschitz Continuity, Abstract Interpretation, Partial Completeness

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

42nd Conference on Very Important Topics (CVIT 2016).
Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:20
LIPICS Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

In mathematical analysis, *Lipschitz continuity* is a strong form of uniform continuity for functions computing over metric spaces, which guarantees that changes in the output are bounded proportionally to changes in the input. It finds numerous applications in various areas of mathematics, including analysis, where it ensures uniform continuity and differentiable properties [14], and optimization, where it plays a key role in convergence guarantees for iterative algorithms [32]. In machine learning, it is used to study robustness, stability and convergence of machine learning models, particularly in adversarial settings [15, 23, 24, 39]. Lipschitz continuity is also relevant and interesting for software, notably to reason about robustness of programs that execute on uncertain inputs [8, 9, 10].

The standard definition of Lipschitz continuity requires that both the input and output spaces of a function (e.g., a program) be equipped with metrics, thereby assuming that controlled variation can be meaningfully captured within the structure imposed by these raw spaces. However, this requirement is often too rigid and fails to account for forms of Lipschitz continuity that remain practically relevant in many important applications. In particular, small changes in the raw representation may correspond to negligible or even irrelevant *semantic* differences. Thus, the lack of a controlled function variation with respect to the raw inputs does not preclude the possibility of a meaningful controlled variation: variations may still be well-behaved when viewed through the lens of *semantic properties* of the inputs and outputs (cf. Ex. 11). This is particularly relevant in many applications, including machine learning [1, 19, 25] and program analysis and verification [13, 18, 36, 37], where the focus often lies on the *semantic properties* of program inputs and outputs.

Our Contribution. To address these limitations, we introduce *Abstract Lipschitz Continuity* (ALC), which ensures that small differences in *semantic approximations* of inputs lead to proportionally bounded differences in the *semantic approximations* of outputs.

We formalize semantic (or *qualitative*) approximations as *upper closure operators*, which are also used in the abstract interpretation framework to model domain abstractions [11, 12]. Values (e.g., character strings) are approximated by considering all other values sharing the same semantic property (e.g., length), admitting an error semantically related to the data. In other words, qualitative or semantic approximations add noise in the *meaning* of what is approximated (cf. Sec. 3).

Abstract Lipschitz Continuity combines these semantic approximations with *quantitative* approximations through their distance in general *pre-metric spaces* [7], i.e., not restricted to metric spaces as the standard definition of Lipschitz continuity (cf. Sec. 3).

We relate ALC for programs to other important and well-studied properties such as (Partial) Completeness in abstract interpretation [4, 7, 20] which limits the imprecision of program analysis (cf. Sec. 4), as well as (Abstract) Robustness [19] in machine learning (cf. Sec. 6). Notably, we show that any abstract Lipschitz continuous function is 0-partial complete, meaning that it does not introduce imprecision — relative to the chosen distance function — when computations are performed over semantic approximations of inputs.

Finally, we propose a novel deductive system for verifying ALC for programs, which is parametric with respect to the chosen input and output semantic approximations and distance functions (cf. Sec. 5). As a particular instance of our general deductive system, when no input and output semantic approximation is performed (i.e., the input and output semantic approximation functions are the identify functions), we find the deductive system proposed by Chaudhuri et al. [9, 10] for proving program robustness.

	pre-	quasisemi-	pseudosemi-	semi-	quasi-	pseudo-	metric
(*non-negativity*)	✓	✓	✓	✓	✓	✓	✓
(*if-identity*)	✓	✓	✓	✓	✓	✓	✓
(*iff-identity*)	✗	✓	✗	✓	✓	✗	✓
(*symmetry*)	✗	✗	✓	✓	✗	✓	✓
(*triangle-inequality*)	✗	✗	✗	✗	✓	✓	✓
Example		$\delta_{\subseteq}$	$\delta_{pat}^{\mathsf{Int}}$	δ_{pat}		$\delta_{siz}, \delta_{\Sigma}$	δ_2
Reference		Ex. 20	Ex. 7	Ex. 7		Ex. 3, 11	Ex. 2

Figure 1 Metrics and their weakening.

2 Preliminaries

We review key preliminaries on metrics, Lipschitz continuity, and abstract interpretation.

Distances. Let $\mathbb{R}^\infty$ be the set of real numbers extended with the infinite symbol ∞, such that for all $r \in \mathbb{R}$, $r < \infty$. Let $\mathbb{R}_{\geq n}$ be the restriction of $\mathbb{R}$ to values greater or equal than $n \in \mathbb{N}$. For instance, $\mathbb{R}^\infty_{\geq 0} \stackrel{\text{def}}{=} \{r \in \mathbb{R} \mid r \geq 0\} \cup \{\infty\}$.

▶ **Definition 1** (Metric). *Given a non-empty set L, a* metric *is a binary function $\delta : L \times L \to \mathbb{R}^\infty$ with the following properties $\forall x, y, z \in L$:*

(1) $\delta(x, y) \geq 0$; *(non-negativity)*

(2) $x = y \Leftrightarrow \delta(x, y) = 0$; *(iff-identity)*

(3) $\delta(x, y) = \delta(y, x)$; *(symmetry)*

(4) $\delta(x, y) \leq \delta(x, z) + \delta(z, y)$. *(triangle-inequality)*

The pair $\langle L, \delta\rangle$ is called a metric space.

▶ **Example 2** (Euclidean Distance). Consider the set of real numbers $\mathbb{R}$. We define the distance δ_2 between two real values $x, y \in \mathbb{R}$ as the absolute value of their difference, i.e., $\delta_2(x, y) \stackrel{\text{def}}{=} |x - y|$. This is the one-dimensional Euclidean distance, well-known to be a metric.

Due to their axioms, metrics are among the strongest types of distances. As we will see in the next sections, depending on what kind of data we want to measure and its abstraction, a distance may not satisfy one or more metric axioms.

A metric that does not satisfy symmetry is called a *quasi-metric*, while a metric that does not satisfy the $\Leftarrow$ implication of (*iff-identity*) is called a *pseudo-metric*. Semi-metrics satisfy all the axioms except for the triangle inequality. The function δ is called a *pre-metric* if it only satisfies (*non-negativity*) and the $\Rightarrow$ implication of the (*iff-identity*), i.e., the (*if-identity*) axiom. All the other metric axioms are not required, making the definition of pre-metric one of the weakest possible distance function. By composing the words pseudo-, quasi- and semi- we obtain different distance flavors by simply keeping the axioms that are satisfied by all the combined words. For instance, a quasisemi-metric is a pre-metric that additionally satisfies the (*iff-identity*), while a pseudosemi-metric only satisfies (*symmetry*) other than (*if-identity*). Fig. 1 summarizes the above distance notions and their properties. The last two rows display the distance symbol and the example in which the distance is defined and used for the first time. We will occasionally use the subscript δ_{L} in cases where the set L may not be immediately clear from the context. The same convention will be adopted to orderings $\preceq$. From this point forward, whenever we say that a function δ is a distance, we assume that it satisfies, at least, the axioms of a pre-metric.

▶ **Example 3** (Size Distance)**.** Consider the powerset $\wp(L)$ of a set L. We write $size(S)$ for the number of elements in the set $S \in \wp(L)$. We define the distance $\delta_{siz} : \wp(L) \times \wp(L) \to \mathbb{R}^\infty$ between two sets $S_1, S_2 \in \wp(L)$ as the absolute value of the difference in their size, i.e., $\delta_{siz}(S_1, S_2) \stackrel{\text{def}}{=} |size(S_2) - size(S_1)|$. Note that δ_{siz} is a pseudo-metric since it does not satisfy the (*iff-identity*) axiom: two sets may have the same size yet being different.

Lipschitz Continuity. In mathematical analysis, Lipschitz continuity is a strong form of uniform continuity of functions that establishes a quantitative relationship between changes to the inputs of a function and its outputs. Specifically, it imposes that perturbations to the inputs of a function lead to at most proportional changes to its outputs. The standard definition of Lipschitz continuity assumes that both the input and output domains are metric spaces.

▶ **Definition 4** (Lipschitz Continuity)**.** *Let $\langle C, \delta_{\mathtt{C}}\rangle$ and $\langle D, \delta_{\mathtt{D}}\rangle$ be metric spaces. Let $k \in \mathbb{R}_{\geq 0}$. A function $f : C \to D$ satisfies* k-Lipschitz continuity *w.r.t. $\langle \delta_{\mathtt{C}}, \delta_{\mathtt{D}}\rangle$ if and only if:*

$$\forall x, y \in C \colon \delta_{\mathtt{D}}(f(x), f(y)) \ \leq \ k\delta_{\mathtt{C}}(x, y).$$

A function f satisfies Lipschitz continuity *w.r.t. $\langle \delta_{\mathtt{C}}, \delta_{\mathtt{D}}\rangle$ if and only if there exists $k \in \mathbb{R}_{\geq 0}$ such that f satisfies k-Lipschitz continuity w.r.t. $\langle \delta_{\mathtt{C}}, \delta_{\mathtt{D}}\rangle$.*

The Lipschitz constant k provides an upper bound on the rate of change for the output of the function f, i.e., ${\delta_{\mathtt{D}}(f(x),f(y))}/{\delta_{\mathtt{C}}(x,y)} \leq k$. Note that, k-Lipschitz continuity can be equivalently formulated as follows:

$$\forall x, y \in C \colon \forall \varepsilon' \geq 0 \colon \delta_{\mathtt{C}}(x, y) \leq \varepsilon' \Rightarrow \ \delta_{\mathtt{D}}(f(x), f(y)) \leq k\varepsilon'$$

Abstract Interpretation. Abstract interpretation [11] provides a general framework for approximating functions by interpreting them over an abstract domain A rather than their exact concrete domain C. It is particularly useful in settings where exact computations are infeasible: decidability is obtained in exchange of an unavoidable information loss. We thus say that A is an *abstraction* of C. Abstractions, originally defined using Galois insertions [11], can equivalently be expressed in terms of upper closure operators [12] (ucos or closures, for short), a formulation we adopt in this work.

▶ **Definition 5** (Upper Closure Operator)**.** *An* upper closure operator *(uco) on a partially ordered set (poset, for short) $\langle C, \preceq\rangle$ is a function $\rho \colon C \to C$ with the following properties $\forall c, c' \in C$:*

(i) $c \preceq c' \Rightarrow \ \rho(c) \preceq \rho(c')$; *(monotonicity)*

(ii) $c \preceq \rho(c)$; *(extensivity)*

(iii) $\rho(\rho(c)) = \rho(c)$. *(idempotence)*

A key property of closures is that they are uniquely determined by the set of their fixpoints $\rho(C) = \{c \in C \mid \rho(c) = c\}$. The set of all upper closure operators on C is denoted by $uco(C)$. As an example, the closure $\mathsf{Sign} \in uco(\wp(\mathbb{Z}))$ abstracts a set of integers by discarding all information except the sign of its values, except when the set contains only the value 0. The closure is defined by the set of fixpoints:

$$\mathsf{Sign}(\wp(\mathbb{Z})) \stackrel{\text{def}}{=} \{\varnothing, \{0\}, \{z \in \mathbb{Z} \mid z \leq 0\}, \{z \in \mathbb{Z} \mid z \geq 0\}, \mathbb{Z}\}$$

3 Abstract Lipschitz Continuity

Semantic and Quantitative Approximations. In many domains, approximations are a fundamental tool for simplifying reasoning while preserving essential properties. Broadly, we can distinguish between *qualitative* (or *semantic*) approximations, and *quantitative* approximations.

Qualitative approximations preserve *properties* of the approximated data. For instance, let $\mathsf{Int}\colon \wp(\mathbb{Z}) \to \wp(\mathbb{Z})$ be the function that transforms a set of integers $S \in \wp(\mathbb{Z})$ into the smallest interval $[l, u] \stackrel{\text{def}}{=} \{i \in \mathbb{Z} \mid l \leq i \leq u\}$ that contains it, namely such that $S \subseteq [l, u]$, where $l \in \mathbb{Z} \cup \{-\infty\}$, $u \in \mathbb{Z} \cup \{+\infty\}$ and $l \leq u$. So, for instance, the set of integers $\{0, 1, 4\}$ can be semantically approximated by the interval $[0, 4]$ through Int. More formally, qualitative approximations can be modeled using upper closure operators (e.g., $\mathsf{Int} \in \mathit{uco}(\wp(\mathbb{Z}))$). Given a poset $\langle C, \preceq\rangle$ and $\rho \in \mathit{uco}(C)$, an element $x \in C$ is semantically approximated by $\rho(x)$, and the set $\{y \in C \mid \rho(y) = \rho(x)\}$ represents all elements in C sharing the same semantic approximation as x. Continuing the example, the set $\{\{0,4\}, \{0,1,4\}, \{0,2,4\}, \{0,3,4\}, \{0,1,2,4\}, \{0,1,3,4\}, \{0,1,2,3,4\}\}$ contains all the sets of integers S such that $\mathsf{Int}(S) = [0, 4]$.

Quantitative approximations preserve *closeness* of the approximated data, typically measured using a distance function in a suitable topological space. More formally, given a pre-metric space $\langle C, \delta\rangle$ and a fixed constant $\varepsilon \in \mathbb{R}^{\infty}_{\geq 0}$, an element $x \in C$ is quantitatively approximated by any element in the set $\{y \in C \mid \delta(x, y) \leq \varepsilon\}$. For instance, using the size distance δ_{siz} (cf. Ex. 3), we may approximate the set $\{0, 1, 4\}$ by any set of integers whose maximum distance from it is at most $\varepsilon = 1$, e.g. by the set $\{0, 1\}$ or $\{5, 6, 8, 10\}$.

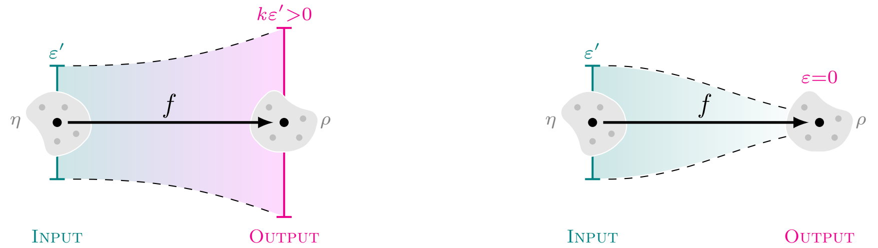
By *combining* the two forms of approximation, we obtain a general approximation that incorporates a quantitative error within a qualitative abstraction, while still keeping the two types of approximations distinct. Let $\langle C, \preceq\rangle$ be a poset and $\langle C, \delta\rangle$ be a pre-metric space, and let $\rho \in \mathit{uco}(C)$ be an abstraction. We define $\delta^{\rho}\colon C \times C \to \mathbb{R}^{\infty}_{\geq 0}$ as:

$$\delta^{\rho}(x, y) \stackrel{\text{def}}{=} \delta(\rho(x), \rho(y))$$

that is, δ^{ρ} calculates the distance between the semantic approximations of x and y with ρ. Clearly, when considering the identity function $\mathit{id} \in \mathit{uco}(C)$ as abstraction (i.e., $\forall x \in C.\ \mathit{id}(x) \stackrel{\text{def}}{=} x$), it holds that $\delta^{id}(x, y) = \delta(x, y)$ for any $x, y \in C$. Note that even if the distance δ satisfies the (*iff-identity*) axiom (thus qualifying as a quasisemi-metric), the derived distance δ^{ρ} may no longer satisfy this axiom due to the input approximation introduced by ρ. This observation also highlights why requiring metric spaces in Def. 4 would be overly restrictive when aiming to define a distance that accounts for both forms of approximation. Nevertheless, δ^{ρ} remains a pre-metric.

▶ **Proposition 6.** *Let $\langle C, \preceq\rangle$ be a poset and let $\rho \in \mathit{uco}(C)$. If $\langle C, \delta\rangle$ is a pre-metric space, then $\langle C, \delta^{\rho}\rangle$ is also a pre-metric space.*

▶ **Example 7** (Path-Length Distance)**.** Let us consider the poset $\langle\wp(\mathbb{Z}), \subseteq\rangle$ and the closure $\mathsf{Int} \in \mathit{uco}(\wp(\mathbb{Z}))$. We define the path-length distance $\delta_{pat}\colon \wp(\mathbb{Z}) \times \wp(\mathbb{Z}) \to \mathbb{N}^{\infty}$ as follows: $\delta_{pat}(S_1, S_2) \stackrel{\text{def}}{=} k$ with $k \in \mathbb{N}$ if $S_1 \subseteq S_2 \vee S_2 \subseteq S_1$ and S_2 has k more elements than S_1 or viceversa. For all other cases, the distance is ∞. So, for instance, $\delta_{pat}(\{0, 1, 4\}, \{0, 1, 4, 10\}) = \delta_{pat}(\{0, 1, 4, 10\}, \{0, 1, 4\}) = 1$ because $\{0, 1, 4, 10\}$ has one more integer than $\{0, 1, 4\}$ namely the number 10, while $\delta_{pat}(\{0, 1, 4\}, \{1, 4, 10\}) = \infty$ because both $\{0, 1, 4\} \not\subseteq \{1, 4, 10\}$ and $\{0, 1, 4\} \not\supseteq \{1, 4, 10\}$ hold. Note that δ_{pat} may differ from δ_{siz} even between comparable sets: $\delta_{pat}(\mathbb{Z}_{>0}, \mathbb{Z}_{\geq 0}) = 1 \neq \infty = \delta_{siz}(\mathbb{Z}_{>0}, \mathbb{Z}_{\geq 0})$. In fact, $\langle\wp(\mathbb{Z}), \subseteq\rangle$ can be seen as a weighted graph where each edge has weight 1 and it connects two sets S_1, S_2 such that



(a) Controlled input/output semantic approximations. **(b)** Suppression of input semantic approximation.

Figure 2 Abstract Lipschitz Continuity.

$S_1 \subset S_2 \vee S_2 \subset S_1$ and there is no other set S' such that $S_1 \subset S' \subset S_2 \vee S_2 \subset S' \subset S_1$. Then the distance $\delta_{pat}(S_1, S_2)$ corresponds to the minimum weighted path between S_1 and S_2. The pair $\langle \wp(\mathbb{Z}), \delta_{pat} \rangle$ forms a semi-metric space. It is not a metric space because δ_{pat} does not satisfy the triangle-inequality axiom: $\delta_{pat}(\{0,1,4\},\{1,4,10\}) = \infty \not\leq 2 = 1 + 1 = \delta_{pat}(\{0,1,4\},\{0,1,4,10\}) + \delta_{pat}(\{0,1,4,10\},\{1,4,10\})$.

By considering the interval abstraction $\mathsf{Int} \in uco(\wp(\mathbb{Z}))$, we can combine the two forms of approximation, namely δ_{pat} and Int, into $\delta_{pat}^{\mathsf{Int}}$: this new distance calculates the number of more elements between two interval abstractions rather than considering the original input sets. Note that $\delta_{pat}^{\mathsf{Int}}$ loses the (*iff-identity*) axiom as one interval might represent more than one set in $\wp(\mathbb{Z})$, thus $\langle \wp(\mathbb{Z}), \delta_{pat}^{\mathsf{Int}} \rangle$ forms a pseudosemi-metric space.

We can now formally define general approximations.

▶ **Definition 8** (General Approximation). *Let $\langle C, \preceq \rangle$ be a poset and $\langle C, \delta \rangle$ be a pre-metric space, and let $\rho \in \mathrm{uco}(C)$. An element $x \in C$ is semantically approximated with ρ* and *quantitatively approximated by δ, up to $\varepsilon \in \mathbb{R}_{\geq 0}^{\infty}$, by any element in the set $\{y \in C \mid \delta^{\rho}(x, y) \leq \varepsilon\}$.*

Continuing Ex. 7, the set $\{0, 1, 4\}$ can be semantically and quantitatively approximated by $\delta_{pat}^{\mathsf{Int}}$ and $\varepsilon = 1$ in any set in

$$\{S \in \wp(\mathbb{Z}) \mid \delta_{pat}^{\mathsf{Int}}(\{0,1,4\}, S) \leq 1\} = \{S \in \wp(\mathbb{Z}) \mid \mathsf{Int}(S) = [-1, 4] \vee \mathsf{Int}(S) = [0, 5]\}$$

Abstract Lipschitz Continuity. When approximations are introduced to the inputs of a function (e.g., a program), they propagate through its computations, affecting the output. Understanding how approximations evolve during computations provides insight into the behavior of the function (e.g., the program).

Abstract Lipschitz Continuity (ALC) imposes a *controlled* (linear) error propagation from a *general approximation of the inputs* to the *general approximation of the result of a function computation* (cf. Fig. 2a).

▶ **Definition 9** (Abstract Lipschitz Continuity). *Let $\langle C, \preceq_{\mathtt{C}} \rangle$ and $\langle D, \preceq_{\mathtt{D}} \rangle$ be the input and output domains (posets), respectively. Let $\langle C, \delta_{\mathtt{C}} \rangle$ and $\langle D, \delta_{\mathtt{D}} \rangle$ be pre-metric spaces. Let $\eta \in \mathrm{uco}(C)$ and $\rho \in \mathrm{uco}(D)$ be the abstractions of the input and output domains, respectively, and $k \in \mathbb{R}_{\geq 0}$. A function $f\colon C \to D$ satisfies* Abstract k-Lipschitz Continuity *(k-ALC, for short) w.r.t. $\langle \delta_{\mathtt{C}}^{\eta}, \delta_{\mathtt{D}}^{\rho} \rangle$ when:*

$$\forall x, y \in C.\ \delta_{\mathtt{D}}^{\rho}(f(x), f(y)) \leq k\delta_{\mathtt{C}}^{\eta}(x, y)$$

A function f satisfies Abstract Lipschitz Continuity *(ALC) if and only if there exists $k \in \mathbb{R}_{\geq 0}$ such that f satisfies Abstract k-Lipschitz Continuity.*

When k-ALC holds, the constant k will be called the *abstract Lipschitz constant*.

Note the difference between Def. 4 of Lipschitz Continuity, and Def. 9 of Abstract Lipschitz Continuity. The former states that the quantitative (metric) distance between two function outputs is at most k times the quantitative (metric) distance between the inputs. The latter captures that the quantitative distance between the semantic approximations (i.e., the properties) of two function outputs ($\delta^{\rho}_{\mathtt{D}}(f(x), f(y))$) is at most k times the quantitative distance between the semantic approximations of the inputs ($\delta^{\eta}_{\mathtt{C}}(x, y)$). The two definitions naturally coincide when both $\langle C, \delta_{\mathtt{C}}\rangle$ and $\langle D, \delta_{\mathtt{D}}\rangle$ are metric-spaces, and the input and output domain abstractions introduce no semantic approximation, namely when $\eta = \rho = id$. In this specific scenario, requiring Lipschitz Continuity w.r.t. $\langle \delta_{\mathtt{C}}, \delta_{\mathtt{D}}\rangle$ is equivalent to requiring ALC w.r.t. $\langle \delta^{id}_{\mathtt{C}}, \delta^{id}_{\mathtt{D}}\rangle$. This also explains why Def. 9 is a generalization of Def. 4 when the input and output domains are considered as posets.

Abstract 0-Lipschitz Continuity represents another special case in which the function computation completely suppresses the input property approximation (cf. Fig. 2b).

Similarly to the concrete definition of k-Lipschitz Continuity (cf. Def. 4), k-ALC can be equivalently reformulated as follows:

▶ **Proposition 10.** *Consider the premises of Def. 9. A function $f\colon C \to D$ satisfies k-ALC w.r.t. $\langle \delta^{\eta}_{\mathtt{C}}, \delta^{\rho}_{\mathtt{D}}\rangle$ if and only if: $\forall x, y \in C.\ \forall \varepsilon' \geq 0.\ \delta^{\eta}_{\mathtt{C}}(x, y) \leq \varepsilon' \ \Rightarrow\ \delta^{\rho}_{\mathtt{D}}(f(x), f(y)) \leq k\varepsilon'$.*

▶ **Example 11.** Let Σ be a chosen alphabet (finite set of characters) and let Σ^* be the Kleene closure of Σ, i.e., the set of all strings of finite length over Σ. We write $length(w)$ to denote the length of the string $w \in \Sigma^*$. We consider the poset $\langle \wp(\Sigma^*), \subseteq\rangle$ and the semantic property $\mathsf{Prefix} \in uco(\wp(\Sigma^*))$ which approximates a set $W \in \wp(\Sigma^*)$ of finite strings with the set of all prefixes of at least one string in W: $\mathsf{Prefix}(W) \overset{\text{def}}{=} \{w \in \Sigma^* \mid \exists w' \in \Sigma^*\colon ww' \in W\}$. We define $\delta_{\Sigma}\colon \wp(\Sigma^*) \times \wp(\Sigma^*) \to \mathbb{N}^{\infty}$ to compute the absolute difference between the number of string lengths in W_1 and W_2, namely:

$$\delta_{\Sigma}(W_1, W_2) \overset{\text{def}}{=} \delta_{siz}(\{length(w_1) \mid w_1 \in W_1\}, \{length(w_2) \mid w_2 \in W_2\})$$

where δ_{siz} is the size distance of Ex. 3, forming the pseudo-metric space $\langle \wp(\Sigma^*), \delta_{\Sigma}\rangle$. Given $|W_1| = n_1$ (e.g., $W_1 = \{a\}$, with $n_1 = 1$) and $|W_2| = n_2$ (e.g., $W_2 = \{bb, ccc\}$, with $n_2 = 2$) with, w.l.g., $n_2 \geq n_1$, then in the worst case all strings in both sets have different lengths, therefore in general $\delta_{\Sigma}(W_1, W_2) \leq n_2 - n_1$ (in the example $\delta_{\Sigma}(W_1, W_2) = 2 - 1 = 1$). The function $f\colon \wp(\Sigma^*) \to \wp(\Sigma^*)$ defined as $f(W) \overset{\text{def}}{=} \{w_1 w_2 \mid w_1, w_2 \in W\}$ concatenates all pairs of strings in W. In the example, $f(W_1) = \{aa\}$, while $f(W_2) = \{bbbb, bbccc, cccbb, cccccc\}$. We can observe that, in the worst case, in $f(W_i)$ we have $\frac{1}{2}n_i(n_i + 1)$ different lengths (the 2 factor division comes from the fact $|w_1 w_2| = |w_2 w_1|$). In the example, we do have the worst case, having $\frac{1}{2}n_2(n_2 + 1) = 3$ different lengths. Then we can show that $\delta_{\Sigma}(f(W_1), f(W_2)) \leq \frac{1}{2}(n_2 + n_1 + 1)(n_2 - n_1)$, which implies that f cannot satisfy ALC w.r.t. $\langle \delta^{id}_{\Sigma}, \delta^{id}_{\Sigma}\rangle$ since, in the worst case, the distance $\delta_{\Sigma}(f(W_1), f(W_2))$ increases the distance $\delta_{\Sigma}(W_1, W_2)$ by a factor $(\frac{1}{2}(n_2 + n_1 + 1))$ which is not constant, as Lipschitz continuity would require.

Consider now $\delta^{\mathsf{Prefix}}_{\Sigma}$, which adds all strings of smaller lengths up to the maximum length present in the set. Then, if $l_1 = \max\{length(w) \mid w \in W_1\}$ and $l_2 = \max\{length(w) \mid w \in W_2\}$, we have $\delta^{\mathsf{Prefix}}_{\Sigma}(W_1, W_2) \leq l_2 - l_1$ (supposing w.l.g., $l_2 \geq l_1$). By definition, the longest string in $f(W_i)$ has length $2l_i$, therefore, in general, we have

$$\delta^{\mathsf{Prefix}}_{\Sigma}(f(W_1), f(W_2)) \leq 2l_2 - 2l_1 \leq 2\delta^{\mathsf{Prefix}}_{\Sigma}(W_1, W_2)$$

which shows that f satisfies 2-ALC w.r.t. $\langle \delta^{\mathsf{Prefix}}_{\Sigma}, \delta^{\mathsf{Prefix}}_{\Sigma}\rangle$.

$$\mathsf{a} \in \mathsf{AExp},\ x \in \mathbb{X},\ \mathsf{b} \in \mathsf{BExp}$$
$$\mathsf{Stm} \ni \mathsf{c} ::= \ \mathbf{skip} \mid x := \mathsf{a} \mid \mathsf{b}?$$
$$\mathsf{Prog} \ni \mathsf{P} ::= \ \mathsf{c} \mid \mathsf{P} \mathbin{;} \mathsf{P} \mid \mathsf{P} \oplus \mathsf{P} \mid \mathsf{P}^*$$

(a) Language syntax.

$$[\![\mathsf{P}_1 \mathbin{;} \mathsf{P}_2]\!] c \stackrel{\text{def}}{=} [\![\mathsf{P}_2]\!] \circ [\![\mathsf{P}_1]\!] c$$
$$[\![\mathsf{P}_1 \oplus \mathsf{P}_2]\!] c \stackrel{\text{def}}{=} [\![\mathsf{P}_1]\!] c \vee [\![\mathsf{P}_2]\!] c$$
$$[\![\mathsf{P}^*]\!] c \stackrel{\text{def}}{=} \bigvee \{[\![\mathsf{P}]\!]^n c \mid n \in \mathbb{N}\}$$

(b) Semantics of programs.

Figure 3 Syntax and semantics of Prog.

4 Abstract Lipschitz Continuity for Programs

Up to this point, the ALC notion has been defined for generic functions. In this section, we focus on two specific aspects: (1) the application of ALC to programs, in particular, to functions representing (monotone) semantics of programs; and (2) a comparison between ALC and the notion of (Partial) Completeness in Abstract Interpretation, a well-established property used to characterize precision loss in program analysis.

Programs. In the following, we will consider programs written in the language Prog of regular commands [3, 33], which is general enough to cover deterministic imperative languages [3] as well as other programming paradigms that include, e.g., non-deterministic and probabilistic computations. The syntax of the language is given in Fig. 3a, where $\oplus$ denotes non-deterministic choice and * is the Kleene closure. We completed the grammar in [3] with an explicit grammar for basic commands in Stm (**skip**, variable assignments, Boolean tests), and we assume a standard grammar for arithmetic expressions in AExp and Boolean expressions in BExp. Variables x range from a denumerable set $\mathbb{X}$ while values v range from a denumerable set $\mathbb{V}$ (e.g., integer or natural numbers).

We assume to have a concrete monotone semantics $[\![\mathsf{c}]\!] \colon C \to C$ for basic commands $\mathsf{c} \in \mathsf{Stm}$ on the complete lattice $\langle C, \preceq, \vee, \wedge, \top, \bot \rangle$, where $\preceq$ is the partial order, $\vee$ is the least upper bound (lub), $\wedge$ the greatest lower bound (glb), $\top$ is the supremum of C and $\bot$ is the infimum of C. Then, the *concrete semantics* $[\![\cdot]\!] \colon \mathsf{Prog} \to C \to C$ for programs is inductively defined on program syntax as in Fig 3b. It is easy to note that, for any program $\mathsf{P} \in \mathsf{Prog}$, $[\![\mathsf{P}]\!]$ is monotone by construction. Given a program $\mathsf{P} \in \mathsf{Prog}$, the semantics $[\![\mathsf{P}]\!]$ is said to be additive when it preserves arbitrary joins, i.e., $\forall S \subseteq C.\ \bigvee\{[\![\mathsf{P}]\!] c \mid c \in S\} = [\![\mathsf{P}]\!](\bigvee S)$.

▶ **Example 12** (Collecting semantics)**.** As an example, consider the complete lattice $\langle \wp(\mathbb{M}), \subseteq, \cup, \cap, \mathbb{M}, \varnothing \rangle$ of program memories, where a program memory $\mathfrak{m} \in \mathbb{M}$ is a function mapping variables to values, namely $\mathfrak{m} \colon \mathbb{X} \to \mathbb{V}$. We can define a collecting big-step semantics $[\![\mathsf{P}]\!] \colon \wp(\mathbb{M}) \to \wp(\mathbb{M})$ for a program $\mathsf{P} \in \mathsf{Prog}$ as the standard predicate transformer semantics on sets of program memories $\wp(\mathbb{M})$. Assume a big-step evaluation semantics $\Downarrow_{\mathsf{a}}$ for arithmetic expressions and $\Downarrow_{\mathsf{b}}$ for Boolean expressions. Given a set of program memories $S \in \wp(\mathbb{M})$, the semantics of basic commands is defined as:

$$[\![\mathbf{skip}]\!] S \stackrel{\text{def}}{=} S$$
$$[\![x := \mathsf{a}]\!] S \stackrel{\text{def}}{=} \{\mathfrak{m}[x \hookleftarrow v] \mid \mathfrak{m} \in S \wedge \mathfrak{m} \Downarrow_{\mathsf{a}} v\}$$
$$[\![\mathsf{b}?]\!] S \stackrel{\text{def}}{=} \{\mathfrak{m} \in S \mid \mathfrak{m} \Downarrow_{\mathsf{b}} \mathtt{tt}\}$$

The collecting semantics for basic commands is monotone by construction on the powerset lattice of memories, and so $[\![\mathsf{P}]\!]$ is also monotone for any program $\mathsf{P} \in \mathsf{Prog}$. Moreover, it is easy to note that $[\![\mathsf{P}]\!] \colon \wp(\mathbb{M}) \to \wp(\mathbb{M})$ satisfies additivity as well.

We are now ready to instantiate Def. 9, originally stated for generic functions, to the specific case of monotone semantic functions by employing the program semantics of interest together with a chosen distance. Let $\langle C, \preceq\rangle$ be a poset and $\langle C, \delta\rangle$ a pre-metric space over the same domain. Let $\eta, \rho \in uco(C)$ be the input and output abstractions, respectively, and $k \in \mathbb{R}_{\geq 0}$. Consider a monotone program semantics $[\![\cdot]\!] \colon \mathsf{Prog} \to C \to C$. We say that *the semantics* $[\![\mathsf{P}]\!]$ *of a program* $\mathsf{P} \in \mathsf{Prog}$ satisfies Abstract k-Lipschitz Continuity w.r.t. $\langle \delta^\eta, \delta^\rho\rangle$:

$$\forall c_1, c_2 \in C.\ \delta^\rho([\![\mathsf{P}]\!]c_1, [\![\mathsf{P}]\!]c_2) \leq k\delta^\eta(c_1, c_2)$$

(Partial) Completeness. Given a monotone function $f \colon C \to D$ over posets $\langle C, \preceq_{\mathtt{C}}\rangle$ and $\langle D, \preceq_{\mathtt{D}}\rangle$ (such as the collecting big-step semantics $[\![\mathsf{P}]\!] \colon \wp(\mathbb{M}) \to \wp(\mathbb{M})$ defined above over $\langle \wp(\mathbb{M}), \subseteq, \cup, \cap, \mathbb{M}, \varnothing\rangle$), the abstractions $\eta \in uco(C)$ and $\rho \in uco(D)$ can be used to approximate computations, thus defining an abstract version $f^\natural \colon \eta(C) \to \rho(D)$ of f. An abstract function $f^\natural \colon \eta(C) \to \rho(D)$ is *sound* when $\rho \circ f \preceq_{\mathtt{D}} f^\natural \circ \eta$ [11]. A sound by construction approximation is $\bar{f} \stackrel{\text{def}}{=} \rho \circ f \circ \eta$, called the *best correct approximation* (bca) [12] of f. Any $f^\natural$ soundly approximating f is, in fact, equal or less precise than the bca, formally: $\rho \circ f \preceq_{\mathtt{D}} \bar{f} \preceq_{\mathtt{D}} f^\natural \circ \eta$ [11]. In the following, we will often shorten the composition of functions such as $\rho \circ f \circ \eta$, by $\rho f \eta$.

A sound abstract computation $f^\natural \colon \eta(C) \to \rho(D)$ performs a precise approximation of a (concrete) monotone function $f \colon C \to D$ whenever $\rho f = f^\natural \eta$. It has been proved that for a precise abstract approximation to exist, the bca $\bar{f} \stackrel{\text{def}}{=} \rho \circ f \circ \eta$ must also be precise [12, 20]. In particular, if $f^\natural$ is a precise abstract approximation of f then $f^\natural = \bar{f}$. *Completeness* [12, 20] in abstract interpretation is a desirable property that ensures the existence of a precise abstract approximation of a (concrete) monotone function f. Proving the completeness of f means proving the bca $\bar{f}$ is precise. Formally,

▶ **Definition 13** (Completeness [12, 20]). *Let* $\langle C, \preceq_{\mathtt{C}}\rangle$ *and* $\langle D, \preceq_{\mathtt{D}}\rangle$ *be posets, and let* $\eta \in uco(C)$ *and* $\rho \in uco(D)$ *be the input and output abstractions, respectively. A monotone function* $f \colon C \to D$ *satisfies* Completeness *w.r.t.* $\langle \eta, \rho\rangle$ *if and only if* $\forall x \in C \colon \rho f(x) = \rho f \eta(x)$.

In practice, Completeness is rarely satisfied. For this reason, Campion et al. [4, 6, 7] introduced a weaker notion of completeness, called *Partial Completeness*, by the use of pre-metrics compatible with the ordering of the underlying poset.

▶ **Definition 14** (Order-Compatible Distance [7]). *Let* $\langle L, \preceq\rangle$ *be a poset. A distance* $\delta : L \times L \to \mathbb{R}^\infty$ *is said to be* compatible with the ordering $\preceq$ *or, in short,* $\preceq$-compatible, *if and only if it also satisfies the following property* $\forall x, y, z \in L$:

$$x \preceq y \preceq z \ \Rightarrow\ \delta(x, y) \leq \delta(x, z) \wedge \delta(y, z) \leq \delta(x, z). \qquad \textit{(chains-order)}$$

A poset $\langle L, \preceq\rangle$ *equipped with a* $\preceq$*-compatible distance* δ *is called a* distance compatible space *and is denoted by the triple* $\langle L, \preceq, \delta\rangle$.

The purpose of the (*chains-order*) axiom is to give a meaning to distances between comparable elements. Notably, let $f_1^\natural$ and $f_2^\natural$ be sound abstract approximations of a concrete monotone function $f \colon C \to D$, i.e., $\rho \circ f \preceq_{\mathtt{D}} f_1^\natural \circ \eta$ and $\rho \circ f \preceq_{\mathtt{D}} f_2^\natural \circ \eta$. If $f_1^\natural$ is more precise than $f_2^\natural$, i.e., $f_1^\natural \preceq_{\mathtt{D}} f_2^\natural$, we expect a decrease in the imprecision (distance) with respect to the concrete computation when using $f_1^\natural$ rather than $f_2^\natural$, i.e., $\forall x \in D \colon \delta(\rho f(x), f_1^\natural \eta(x)) \leq \delta(\rho f(x), f_2^\natural \eta(x))$.

▶ **Example 15.** The poset $\langle \mathbb{R}, \leq\rangle$ equipped with the Euclidean distance δ_2 from Ex. 2 is a metric compatible space. The poset $\langle \wp(L), \subseteq\rangle$ and the size distance δ_{siz} from Ex. 3 form a pseudometric compatible space. In Ex. 7, $\langle \wp(\mathbb{Z}), \subseteq, \delta_{pat}^{\mathsf{Int}}\rangle$ is a pseudosemi-metric compatible space.

Def. 14 is general enough to be instantiated with other definitions of distances used in the literature of abstract interpretation (see, e.g., [4, 26, 27, 35, 38]).

We can now recall the definition of ε-Partial Completeness.

▶ **Definition 16** (ε-Partial Completeness [4, 7]). *Let $\langle C, \preceq_{\mathtt{C}}\rangle$ be a poset and $\langle D, \preceq_{\mathtt{D}}, \delta_{\mathtt{D}}\rangle$ be a pre-metric compatible space, let $\eta \in uco(C)$ and $\rho \in uco(D)$ be the input and output abstractions, respectively. Let $\varepsilon \in \mathbb{R}^{\infty}_{\geq 0}$. A monotone function $f : C \to D$ satisfies* ε-Partial Completeness *w.r.t. $\langle \eta, \delta^{\rho}_{\mathtt{D}}\rangle$ if and only if $\forall x \in C\colon \delta^{\rho}_{\mathtt{D}}(f(x), f\eta(x)) \leq \varepsilon$.*

The equality requirement of Def. 13 is relaxed by admitting a bounded imprecision, i.e. a bounded distance, between $\rho f(x)$ and the bca $\rho f \eta(x)$ for all $x \in C$, which must not exceed ε. The imprecision to be measured and bounded is encoded in the pre-metric $\preceq_{\mathtt{D}}$-compatible $\delta_{\mathtt{D}}$ defined over the output domain D.

▶ **Example 17.** Let $\langle \wp(\mathbb{Z}), \subseteq, \delta_{siz}\rangle$ be an instance of the pseudo-metric compatible space from Ex. 3. Consider the program $\mathsf{M}\colon\ x := x \bmod 2$ and its collecting semantics $[\![\mathsf{M}]\!] : \wp(\mathbb{Z}) \to \wp(\mathbb{Z})$. Let $\rho = \eta = \mathsf{Int}$ where $\mathsf{Int} \in uco(\wp(\mathbb{Z}))$ is the interval abstraction defined in Sec. 3. Then $[\![\mathsf{M}]\!]$ does not satisfy Completeness w.r.t. $\langle \mathsf{Int}, \mathsf{Int}\rangle$ because for the input $\{2, 4\}$ we get:

$$\mathsf{Int}([\![\mathsf{M}]\!]\{2, 4\}) = [0, 0] \subset [0, 1] = \mathsf{Int}([\![\mathsf{M}]\!]\{2, 3, 4\}) = \mathsf{Int}([\![\mathsf{M}]\!]\mathsf{Int}(\{2, 4\}))$$

However, if we allow an imprecision quantified by $\varepsilon = 1$, we get:

$$\delta^{\mathsf{Int}}_{siz}([\![\mathsf{M}]\!]\{2, 4\}, [\![\mathsf{M}]\!](\mathsf{Int}(\{2, 4\}))) = \delta_{siz}([0, 0], [0, 1]) \leq 1$$

In particular, it is easy to note that $\delta^{\mathsf{Int}}_{siz}([\![\mathsf{M}]\!]S, [\![\mathsf{M}]\!](\mathsf{Int}(S))) \leq 1$, for all sets $S \in \wp(\mathbb{Z})$, which implies that $[\![\mathsf{M}]\!]$ is 1-Partial Complete with respect to $\langle \mathsf{Int}, \delta^{\mathsf{Int}}_{siz}\rangle$.

It is worth noting that, if a function f is proved to satisfy Completeness for abstractions $\langle \eta, \rho\rangle$, then f is also 0-Partial Complete for $\langle \eta, \delta^{\rho}\rangle$ with respect to any pre-metric order-compatible δ (thanks to the (*if-identity*) axiom). However, the converse does not hold if the (*iff-identity*) axiom is not satisfied by δ, e.g., when δ is a pseudo-metric.

Abstract Lipschitz Continuity and Partial Completeness. It turns out that ALC (cf. Def 9) is a much stronger requirement than Partial Completeness (cf. Def. 16) for a program (semantics, or a monotone function). Indeed, satisfying ALC is sufficient to also satisfy 0-Partial Completeness:

▶ **Theorem 18.** *Let $\langle C, \preceq_{\mathtt{C}}, \delta_{\mathtt{C}}\rangle$ and $\langle D, \preceq_{\mathtt{D}}, \delta_{\mathtt{D}}\rangle$ be pre-metric compatible spaces, let $\eta \in uco(C)$, $\rho \in uco(D)$ be abstractions, and let $k \in \mathbb{R}_{\geq 0}$. Consider a monotone function $f : C \to D$. Then, if f satisfies k-ALC w.r.t. $\langle \delta^{\eta}_{\mathtt{C}}, \delta^{\rho}_{\mathtt{D}}\rangle$, it also satisfies 0-Partial Completeness w.r.t. $\langle \eta, \delta^{\rho}_{\mathtt{D}}\rangle$, namely:*

$$[\forall x, y \in C.\ \delta^{\rho}_{\mathtt{D}}(f(x), f(y)) \leq\ k\delta^{\eta}_{\mathtt{C}}(x, y)]\ \Rightarrow\ [\forall x \in C.\ \delta^{\rho}_{\mathtt{D}}(f(x), f\eta(x)) \leq 0]$$

Proofs of the above result, as well as of the corollary below, are provided in Appendix A. Knowing that a monotone function f is k-ALC for $\langle \delta^{\eta}_{\mathtt{C}}, \delta^{\rho}_{\mathtt{D}}\rangle$ leads to conclude that the bca $\rho f \eta$ is 0-partial complete for the same abstractions. Specifically, $\rho f \eta$ will produce *no imprecision* according to $\delta_{\mathtt{D}}$, when used to approximate f.

When $\delta_{\mathtt{D}}$ is a quasisemi-metric, then the above result implies that $\rho f \eta$ is a complete approximation of f thanks to the (*iff-identity*) axiom.

▶ **Corollary 19.** *If $\langle D, \preceq_{\mathtt{D}}, \delta_{\mathtt{D}}\rangle$ is a quasisemi-metric compatible space then k-ALC w.r.t. $\langle \delta^{\eta}_{\mathtt{C}}, \delta^{\rho}_{\mathtt{D}}\rangle$ implies Completeness w.r.t. $\langle \eta, \rho\rangle$.*

▶ **Example 20.** Let R be the following program:

$$(x > 0? ; x := x - 1) \oplus (x \leq 0? ; x := x + 1)$$

which increments all non-negative values by 1 and decrements all non-positive values by 1. Let us consider the program R^*, which is the Kleene closure of R, and its collecting semantics $[\![\mathsf{R}^*]\!] : \wp(\mathbb{Z}) \to \wp(\mathbb{Z})$. Let $\langle \wp(\mathbb{Z}), \subseteq, \delta_\subseteq \rangle$ be a quasisemi-metric compatible space where, for any two sets $S_1, S_2 \in \wp(\mathbb{Z})$, $\delta_\subseteq(S_1, S_2) \stackrel{\text{def}}{=} \delta_{siz}(S_1, S_2)$ (cf. Ex. 3) if $S_1 \subseteq S_2$, ∞ otherwise. Compared to δ_{siz}, the distance $\delta_\subseteq$ looses the (*symmetry*) and the (*triangle-inequality*) axioms but gains the (*iff-identity*) axiom. Let us also consider again the interval closure $\mathsf{Int} \in uco(\wp(\mathbb{Z}))$. The collecting semantics $[\![\mathsf{R}^*]\!]$ satisfies 1-ALC w.r.t. $\langle \delta_\subseteq^{\mathsf{Int}}, \delta_\subseteq^{\mathsf{Int}} \rangle$. Indeed, $[\![\mathsf{R}^*]\!]$ is monotone by definition, thus preserving the inclusion relation, and either reduces the distance of input intervals or leaves them unchanged. For instance:

$$\delta_\subseteq^{\mathsf{Int}}([\![\mathsf{R}^*]\!]([2,6]), [\![\mathsf{R}^*]\!]([0,7])) = \delta_\subseteq^{\mathsf{Int}}([0,6],[0,7]) = 1 \leq 3 = \delta_\subseteq^{\mathsf{Int}}([2,6],[0,7])$$
$$\delta_\subseteq^{\mathsf{Int}}([\![\mathsf{R}^*]\!]([-5,-2]), [\![\mathsf{R}^*]\!]([-7,0])) = \delta_\subseteq^{\mathsf{Int}}([-5,1],[-7,1]) = 2 \leq 5 = \delta_\subseteq^{\mathsf{Int}}([-5,-2],[-7,0])$$
$$\delta_\subseteq^{\mathsf{Int}}([\![\mathsf{R}^*]\!]([-2,3]), [\![\mathsf{R}^*]\!]([-5,3])) = \delta_\subseteq^{\mathsf{Int}}([-2,3],[-5,3]) = 3 \leq 3 = \delta_\subseteq^{\mathsf{Int}}([-2,3],[-5,3])$$

By Thm. 18, the semantics $[\![\mathsf{R}^*]\!]$ also satisfies 0-Partial Completeness w.r.t. $\langle \mathsf{Int}, \delta_\subseteq^{\mathsf{Int}} \rangle$, i.e., $\delta_\subseteq^{\mathsf{Int}}([\![\mathsf{R}^*]\!]S, [\![\mathsf{R}^*]\!]\mathsf{Int}(S)) \leq 0$, for any $S \in \wp(\mathbb{Z})$. Moreover, since $\delta_\subseteq$ is a quasisemi-metric we can also conclude that $[\![\mathsf{R}^*]\!]$ is complete w.r.t. $\langle \mathsf{Int}, \mathsf{Int} \rangle$, namely, its bca $\mathsf{Int} \circ [\![\mathsf{R}^*]\!] \circ \mathsf{Int}$ does not add any imprecision when approximating $[\![\mathsf{R}^*]\!]$. It is easy to note that 1-ALC w.r.t. $\langle \delta_\subseteq^{\mathsf{Int}}, \delta_{pat}^{\mathsf{Int}} \rangle$ also holds for $[\![\mathsf{R}^*]\!]$.

Another way to interpret Cor. 19 (and analogously Thm. 18) is as follows: if a monotone function f does not admit a precise bca $\bar{f}$ over $\langle \eta, \rho \rangle$, then f cannot be ALC for $\langle \delta_\mathsf{C}^\eta, \delta_\mathsf{D}^\rho \rangle$, where δ_C and δ_D are any quasisemi-metric order-compatible distances. This is because Partial Completeness only compares the output results (of ρf and $\rho f \eta$) on the same chain of the poset $\langle D, \preceq_\mathsf{D} \rangle$, a consequence of the soundness condition $\rho f \preceq_\mathsf{D} \rho f \eta$.

▶ **Example 21.** Consider the pseudo-metric order-compatible space $\langle \wp(\mathbb{Z}), \subseteq, \delta_{siz} \rangle$ and the interval closure $\mathsf{Int} \in uco(\wp(\mathbb{Z}))$. The semantics $[\![\mathsf{R}]\!]$ from Ex. 20 does not satisfy 0-Partial Completeness w.r.t. $\langle \mathsf{Int}, \delta_{siz}^{\mathsf{Int}} \rangle$: given $X = \{-1, 1\}$, we have $\delta_{siz}^{\mathsf{Int}}([\![\mathsf{R}]\!]X, [\![R]\!]\mathsf{Int}(X)) = \delta_{siz}^{\mathsf{Int}}([0,0],[0,1]) = 1 \neq 0$. Thus, $[\![\mathsf{R}]\!]$ cannot satisfy ALC for $\langle \delta_{siz}^{\mathsf{Int}}, \delta_{siz}^{\mathsf{Int}} \rangle$. In fact, it is easy to note that $[\![\mathsf{R}]\!]$ satisfies 1-Partial Completeness for all inputs.

In Section 6 we also relate ALC to other important program properties in the literature.

5 Proving Abstract Lipschitz Continuity for Programs

Deductive systems for the verification of Completeness [16], Partial Completeness [4] and (concrete) Lipschitz continuity [9, 10] properties of programs have already been formalized in the literature. In this section we introduce a novel deductive system, inductively defined on the program's syntax, that is able to soundly prove the new ALC notion of an additive program semantics w.r.t. the input and output abstractions $\langle \eta, \rho \rangle$ and a given pre-metric δ. Our objective in designing this deductive system has been to track the assumptions of ALC needed for having a compositional proof. Soundness here means that when the semantics of a program P is typed as k-ALC w.r.t. $\langle \delta^\eta, \delta^\rho \rangle$ by the deductive system, then $[\![\mathsf{P}]\!] \colon C \to C$ is certainly k-ALC for $\langle \delta^\eta, \delta^\rho \rangle$. Conversely, the deductive system is not complete, namely, not all abstract Lipschitz continuous program semantics proofs can be derived through the

$$\frac{[\![\mathsf{c}]\!] \in k\text{-}Lip\langle \delta^{\eta}, \delta^{\rho}\rangle}{k \vdash [\delta^{\eta}]\, \mathsf{c}\, (\delta^{\rho})}\ (\textbf{base})$$

$$\frac{k' \vdash [\delta^{\eta'}]\, \mathsf{P}\, (\delta^{\rho'}) \quad k' \leq k \quad \eta' \in t\text{-}Lip\langle \delta^{\eta}, \delta^{\eta'}\rangle \quad \rho \in s\text{-}Lip\langle \delta^{\rho'}, \delta^{\rho}\rangle}{stk \vdash [\delta^{\eta}]\, \mathsf{P}\, (\delta^{\rho})}\ (\textbf{weaken})$$

$$\frac{k_1 \vdash [\delta^{\eta}]\, \mathsf{P}_1\, (\delta^{\rho}) \quad k_2 \vdash [\delta^{\eta}]\, \mathsf{P}_2\, (\delta^{\rho}) \quad \eta \in t\text{-}Lip\langle \delta^{\rho}, \delta^{\eta}\rangle}{k_1 t k_2 \vdash [\delta^{\eta}]\, \mathsf{P}_1; \mathsf{P}_2\, (\delta^{\rho})}\ (\textbf{seq})$$

$$\frac{k_1 \vdash [\delta^{\eta}]\, \mathsf{P}_1\, (\delta^{\rho}) \quad k_2 \vdash [\delta^{\eta}]\, \mathsf{P}_2\, (\delta^{\rho}) \quad \rho \in t\text{-}Lip\langle \delta^{id}, \delta^{\rho}\rangle \quad \oplus\text{-Bound}(\langle \delta^{\eta}, \delta^{\rho}\rangle, \mathfrak{b})}{t\mathfrak{b}(k_1, k_2) \vdash [\delta^{\eta}]\, \mathsf{P}_1 \oplus \mathsf{P}_2\, (\delta^{\rho})}\ (\textbf{join})$$

$$\frac{k \vdash [\delta^{\eta}]\, \mathsf{P}\, (\delta^{\rho}) \quad \eta \in t\text{-}Lip\langle \delta^{\rho}, \delta^{\eta}\rangle \quad *\text{-Bound}(\mathsf{P}^*, m)}{(tk)^m \vdash [\delta^{\eta}]\, \mathsf{P}^*\, (\delta^{\rho})}\ (\textbf{star})$$

Figure 4 A deductive system for proving ALC for Prog.

deductive system. This means that we are performing an under-approximation of the set of all abstract Lipschitz continuous program semantics.

We first introduce the following set

$$k\text{-}Lip\langle \delta^{\eta}, \delta^{\rho}\rangle \stackrel{\text{def}}{=} \{f \in C \to C \mid f \text{ is Abstract } k\text{-Lipschitz Continuous w.r.t. } \langle \delta^{\eta}, \delta^{\rho}\rangle\}$$

of all abstract k-Lipschitz continuous functions on the complete lattice $\langle C, \preceq\rangle$ for $\langle \delta^{\eta}, \delta^{\rho}\rangle$. The following lemma outlines some basic properties of this class.

▶ **Lemma 22.** *The following hold for all functions $f \in C \to C$, closures $\eta, \rho \in uco(C)$, pre-metric δ and $k \in \mathbb{R}_{\geq 0}$:*

(i) $k \geq 1 \ \Rightarrow\ \rho \in k\text{-}Lip\langle \delta^{\rho}, \delta^{\rho}\rangle$

(ii) f *is k-Lipschitz continuous w.r.t.* $\langle \delta, \delta\rangle \ \Leftrightarrow\ f \in k\text{-}Lip\langle \delta^{id}, \delta^{id}\rangle \wedge \delta$ *metric*

(iii) $\rho \in k\text{-}Lip\langle \delta^{id}, \delta^{id}\rangle \ \Leftrightarrow\ \rho \in k\text{-}Lip\langle \delta^{id}, \delta^{\rho}\rangle$

(i) states that, when considering the same input-output abstractions (i.e. $\eta = \rho$), then the abstraction function is k-ALC for any $k \geq 1$. Moreover, for the statement (ii), when both input-output abstractions are the identity function id and the distance δ is a metric, then the class $k\text{-}Lip\langle \delta^{id}, \delta^{id}\rangle$ corresponds precisely to the set of all (concrete) k-Lipschitz continuous functions (cf. Def. 4). Finally, (iii) shows that, when a closure ρ satisfies ALC w.r.t. $\langle \delta^{id}, \delta^{id}\rangle$, then ρ also satisfies k-ALC for $\langle \delta^{id}, \delta^{\rho}\rangle$. This is due to the idempotence property of closure operators.

From now on, we fix an additive program semantics of interest $[\![\cdot]\!]\colon \mathsf{Prog} \to C \to C$ as well as the complete lattice $\langle C, \preceq, \vee, \wedge, \top, \bot\rangle$, and we will also use the statement "P is k-ALC w.r.t. $\langle \delta^{\eta}, \delta^{\rho}\rangle$" to indicate that the semantics $[\![\mathsf{P}]\!]$ is abstract k-Lipschitz continuous w.r.t. $\langle \delta^{\eta}, \delta^{\rho}\rangle$, i.e. $[\![\mathsf{P}]\!] \in k\text{-}Lip\langle \delta^{\eta}, \delta^{\rho}\rangle$.

The deductive rules are provided in Fig. 4. The judgments take the form:

$$k \vdash [\delta^{\eta}]\, \mathsf{P}\, (\delta^{\rho})$$

We will later show that deriving a judgment $k \vdash [\delta^{\eta}]\, \mathsf{P}\, (\delta^{\rho})$ through the deductive rules in Fig. 4, implies that $[\![\mathsf{P}]\!] \in k\text{-}Lip\langle \delta^{\eta}, \delta^{\rho}\rangle$. Let us examine each rule and provide an intuitive, informal explanation for better understanding.

The rule (**base**) allows to derive the triple $k \vdash [\delta^\eta]\,\mathsf{c}\,(\delta^\rho)$ for all basic transfer functions $\mathsf{c} \in \mathsf{Stm}$ (i.e., for **skip**, assignments and Boolean guards) by assuming that we have a proof of k-ALC of them, encoded by the predicate $[\![\mathsf{c}]\!] \in k\text{-}Lip\langle\delta^\eta, \delta^\rho\rangle$.

The rule (**weaken**) allows to weaken both the abstract Lipschitz constant and the abstractions considered. In particular, when we are able to derive the k'-ALC for program P w.r.t. $\langle\delta^{\eta'}, \delta^{\rho'}\rangle$, then we can always deduce a higher abstract Lipschitz constant $k \geq k'$ without changing the validity of the triple. For the input abstraction η', we can consider a new input abstraction η whenever η' is proved to be t-ALC w.r.t. $\langle\delta^\eta, \delta^{\eta'}\rangle$ with η as input abstraction. This weakening comes at the cost of multiplying the already deduced constant k' with the new constant t. This could happen, for instance, when η is in fact widening the distance $\delta^{\eta'}(c_1, c_2)$ between any two elements $c_1, c_2 \in C$, by a constant factor of t, namely by $t\delta^\eta(c_1, c_2)$. Conversely, we can weaken the output abstraction ρ' by a new abstraction ρ whenever ρ is proved to be ALC for $\langle\delta^{\rho'}, \delta^\rho\rangle$ namely with ρ' as input abstraction. Here ρ could represent a narrow output abstraction in terms of distance δ between elements in C with respect to ρ', namely having distance $\delta^\rho(c_1, c_2) \leq s\delta^{\rho'}(c_1, c_2)$ and thus introducing a new abstract Lipschitz constant s. Note that the rule (**weaken**) allows also for selecting which weakening we want to apply. For instance, if we want to weaken the abstract Lipschitz constant k' only, then we can set $\eta' \in 1\text{-}Lip\langle\delta^{\eta'}, \delta^{\eta'}\rangle$ and $\rho' \in 1\text{-}Lip\langle\delta^{\rho'}, \delta^{\rho'}\rangle$ in the premises as they always hold (cf. statement (i) of Lem. 22) without modifying any abstraction.

Composition of programs is treated by the rule (**seq**). Although it is well known that composing two (concrete) Lipschitz continuous functions f_1 and f_2 with Lipschitz constants k_1 and k_2, respectively, gives as result a new k_1k_2-Lipschitz continuous function $f_2 \circ f_1$, this in general does not always hold for ALC as abstractions come into play. However, when we have a derivation for P_1 and P_2 with abstract Lipschitz constants k_1 and k_2, respectively, and we are able to prove that the input abstraction η is t-ALC w.r.t. $\langle\delta^\rho, \delta^\eta\rangle$, then this is a sufficient condition for deriving the k_2tk_1-ALC of the composition $\mathsf{P}_1; \mathsf{P}_2$. Requiring $\eta \in t\text{-}Lip\langle\delta^\rho, \delta^\eta\rangle$ corresponds to require $\delta^\eta(c_1, c_2) \leq t\delta^\rho(c_1, c_2)$, namely that we have a linear relation between their distances: when $t \geq 1$ then ρ is widening the distance, while when $0 < t < 1$ then ρ is narrowing their distances, both cases with a constant factor of t. Note that, when the input and output abstractions coincide, i.e. $\eta = \rho$, then $\rho \in 1\text{-}Lip\langle\delta^\rho, \delta^\rho\rangle$ holds trivially (cf. statement (i) of Lem. 22). As a consequence, the ALC property is closed under composition, analogously to the standard Lipschitz continuity property.

The rule (**join**) involves the join operator. Similarly for the composition, the join of two ALC functions is not necessarily ALC. The problem here stems in the fact that the resulting abstract Lipschitz constant bound could not be determined by knowing only the abstract Lipschitz constants of both P_1 and P_2. This is because the distance between the execution of $\mathsf{P}_1 \oplus \mathsf{P}_2$ and the join of the two post-conditions, relies on the underlying structure of the input and output abstractions considered. Our solution, inspired by [4, 8], consists of introducing a new predicate $\oplus$-Bound$(\langle\delta^\eta, \delta^\rho\rangle, \mathfrak{b})$ parameterized by a binary function $\mathfrak{b} : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ producing a new abstract Lipschitz constant.

▶ **Definition 23** **($\oplus$-Bound).** *Consider a binary function $\mathfrak{b} : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$. The predicate $\oplus$-Bound$(\langle\delta^\eta, \delta^\rho\rangle, \mathfrak{b})$ holds when the function $\mathfrak{b}$ satisfies the following condition for any $\mathsf{P}_1, \mathsf{P}_2 \in \mathsf{Prog}$:*

$$[\![\mathsf{P}_1]\!] \in k_1\text{-}Lip\langle\delta^\eta, \delta^\rho\rangle \text{ and } [\![\mathsf{P}_2]\!] \in k_2\text{-}Lip\langle\delta^\eta, \delta^\rho\rangle \;\Rightarrow\; \rho[\![\mathsf{P}_1]\!] \oplus \rho[\![\mathsf{P}_2]\!] \in \mathfrak{b}(k_1, k_2)\text{-}Lip\langle\delta^\eta, \delta^\rho\rangle$$

▶ **Example 24.** Consider the pseudo-metric space $\langle\wp(\mathbb{Z}), \subseteq, \delta_{siz}\rangle$ and the collecting semantics $[\![\cdot]\!]$. Let the input and output abstractions be $\rho = \eta = \mathsf{Int}$. If we define $+(k_1, k_2) \stackrel{\text{def}}{=} k_1 + k_2$ for any $k_1, k_2 \in \mathbb{R}_{\geq 0}$, then the predicate $\oplus$-Bound$(\langle\delta^{\mathsf{Int}}_{siz}, \delta^{\mathsf{Int}}_{siz}\rangle, +)$ holds. In other words, having an ALC proof for both P_1 and P_2, with abstract Lipschitz constants k_1, k_2, respectively, gives as result:

$$
\begin{aligned}
\delta_{siz}((\mathsf{Int}[\![\mathsf{P}_1]\!] \oplus \mathsf{Int}[\![\mathsf{P}_2]\!])c_1, (\mathsf{Int}[\![\mathsf{P}_1]\!] \oplus \mathsf{Int}[\![\mathsf{P}_2]\!])c_2) &\le k_1\delta_{siz}(\mathsf{Int}(c_1), \mathsf{Int}(c_2)) \\
&\quad + k_2\delta_{siz}(\mathsf{Int}(c_1), \mathsf{Int}(c_2)) \\
&= (k_1 + k_2)\delta_{siz}(\mathsf{Int}(c_1), \mathsf{Int}(c_2))
\end{aligned}
$$

This is because, when considering δ_{siz} as distance and Int as input and output abstractions, the size of the join of two intervals can be over-approximated by the sum of the number of the elements inside the two intervals. A similar reasoning holds for the quasisemi-metric space $\langle \wp(\mathbb{Z}), \subseteq, \delta_{\subseteq} \rangle$ defined in Ex. 20.

The premise of the rule (**join**) asks for the validity of the following predicates: assume that we have an ALC derivation $k_1 \vdash [\delta^\eta]\,\mathsf{P}_1\,(\delta^\rho)$ for P_1, and $k_2 \vdash [\delta^\eta]\,\mathsf{P}_2\,(\delta^\rho)$ for P_2; if ρ is t-ALC w.r.t. $\langle \delta^{id}, \delta^\rho \rangle$, and the predicate $\oplus$-Bound$(\langle \delta^\eta, \delta^\rho \rangle, \mathfrak{b})$ holds, then we can soundly conclude that the join $\mathsf{P}_1 \oplus \mathsf{P}_2$ is ALC with abstract Lipschitz constant $t\mathfrak{b}(k_1, k_2)$.

Finally, the rule (**star**) deals with loop iterations. It requires that the program P is k-ALC for $\langle \delta^\eta, \delta^\rho \rangle$ and that the input abstraction η is t-ALC for $\langle \delta^\rho, \delta^\eta \rangle$, similar to the (**seq**) rule. In addition, (**star**) requires the assertion $*$-Bound(P^*, m) stating that the loop P^* reaches a least fixpoint in m or less iterations, where m is a constant. This condition can be established either via an auxiliary checker, e.g. an SMT solver, or by manual annotation. Under these premises, we can soundly apply (**star**) in the same way we apply (**seq**), and obtain an abstract Lipschitz constant k^m for the iterations multiplied by the constant t^m generated by applying m-times the abstraction, thus concluding with the $(tk)^m$-ALC of P^*.

The following theorem shows that our proposed deductive system is sound, namely, if $k \vdash [\delta^\eta]\,\mathsf{P}\,(\delta^\rho)$ can be derived by applying the rules of Fig. 4, then $[\![\mathsf{P}]\!]$ satisfies k-ALC w.r.t. $\langle \delta^\eta, \delta^\rho \rangle$. The proof can be found in Appendix B.

▶ **Theorem 25** (Soundness). *Let $\mathsf{P} \in \mathsf{Prog}$, δ be a pre-metric and $\eta, \rho \in uco(C)$ be the input and output abstractions, respectively. Then:*

$$k \vdash [\delta^\eta]\,\mathsf{P}\,(\delta^\rho) \;\Rightarrow\; [\![\mathsf{P}]\!] \in k\text{-}Lip\langle \delta^\eta, \delta^\rho \rangle$$

▶ **Example 26.** Consider the following program ReLU:

$$(x < 0?\,;\, x := 0) \oplus (x \ge 0?\,;\, \textbf{skip})$$

implementing the ReLU rectifier function in artificial neural networks [31], that filters the input below 0. Consider the quasisemi-metric space $\langle \wp(\mathbb{Z}), \subseteq, \delta_{\subseteq} \rangle$ and the input and output abstraction $\eta = \rho = \mathsf{Int}$. We want to prove that the collecting semantics $[\![\mathsf{ReLU}]\!] : \wp(\mathbb{Z}) \to \wp(\mathbb{Z})$ satisfies 1-ALC for $\langle \delta_{\subseteq}^{\mathsf{Int}}, \delta_{\subseteq}^{\mathsf{Int}} \rangle$. Let us start by analyzing the base commands on the left of $\oplus$. Because the Boolean guard $x < 0?$ is either preserving or removing values from the input, by the rule (**base**), we can derive $1 \vdash [\delta_{\subseteq}^{\mathsf{Int}}]\, x < 0?\, (\delta_{\subseteq}^{\mathsf{Int}})$. The command $x := 0$ is neutralizing any distance between input sets since $\delta_{\subseteq}^{\mathsf{Int}}([\![x := 0]\!]S_1, [\![x := 0]\!]S_2) = 0$ for any $S_1, S_2 \in \wp(\mathbb{Z})$. So we can derive $0 \vdash [\delta_{\subseteq}^{\mathsf{Int}}]\, x := 0\, (\delta_{\subseteq}^{\mathsf{Int}})$ by the rule (**base**). Since $\mathsf{Int} \in 1\text{-}Lip\langle \delta_{\subseteq}^{\mathsf{Int}}, \delta_{\subseteq}^{\mathsf{Int}} \rangle$ follows from Lem. 22, we can infer $0 \vdash [\delta_{\subseteq}^{\mathsf{Int}}]\, x < 0?\,;\, x := 0\, (\delta_{\subseteq}^{\mathsf{Int}})$ by the rule (**seq**). For the base commands on the right of $\oplus$, we get $1 \vdash [\delta_{\subseteq}^{\mathsf{Int}}]\, x \ge 0?\, (\delta_{\subseteq}^{\mathsf{Int}})$ with rule (**base**). The **skip** command does not modify the distance of the input sets, so $1 \vdash [\delta_{\subseteq}^{\mathsf{Int}}]\, \textbf{skip}\, (\delta_{\subseteq}^{\mathsf{Int}})$ can be derived by (**base**). Since $\mathsf{Int} \in 1\text{-}Lip\langle \delta_{\subseteq}^{\mathsf{Int}}, \delta_{\subseteq}^{\mathsf{Int}} \rangle$ holds, the rule (**seq**) derives $1 \vdash [\delta_{\subseteq}^{\mathsf{Int}}]\, x \ge 0?\,;\, \textbf{skip}\, (\delta_{\subseteq}^{\mathsf{Int}})$. Now for the $\oplus$ operation, we consider $\mathfrak{b}$ as the sum operation $+$ as shown in Ex. 24, thus guaranteeing a sound upper bound for the abstract

Lipschitz constants on the program join. By Lem. 22, the condition $\mathsf{Int} \in 1\text{-}Lip\langle \delta_{\subseteq}^{id}, \delta_{\subseteq}^{\mathsf{Int}} \rangle$ is equivalent to requiring $\delta_{\subseteq}(\mathsf{Int}(S_1), \mathsf{Int}(S_2)) \leq \delta_{\subseteq}(S_1, S_2)$ for all $S_1, S_2 \in \wp(\mathbb{Z})$, which is clearly satisfied by Int. Therefore, by $\mathsf{Int} \in 1\text{-}Lip\langle \delta_{\subseteq}^{id}, \delta_{\subseteq}^{\mathsf{Int}} \rangle$, $\oplus\text{-}Bound(\langle \delta_{\subseteq}^{\mathsf{Int}}, \delta_{\subseteq}^{\mathsf{Int}} \rangle, +)$, $+(0,1) = 1$ and the two derivations on the left and right parts of $\oplus$, we can conclude by the rule (**join**): $1 \vdash [\delta_{\subseteq}^{\mathsf{Int}}]\, \mathsf{ReLU}\, (\delta_{\subseteq}^{\mathsf{Int}})$. By Thm. 25, this implies that $[\![\mathsf{ReLU}]\!]$ satisfies 1-ALC for $\langle \delta_{\subseteq}^{\mathsf{Int}}, \delta_{\subseteq}^{\mathsf{Int}} \rangle$.

Although the proof system of Fig. 4 is sound, it is not complete: there might exist programs that satisfy k-ALC for which the system fails to derive a proof, or for which it only establishes the property with a larger abstract Lipschitz constant $k' \geq k$.

▶ **Example 27.** Consider the program R of Ex. 20 together with the quasisemi-metric space $\langle \wp(\mathbb{Z}), \subseteq, \delta_{\subseteq} \rangle$ and the collecting semantics $[\![\mathsf{R}]\!] : \wp(\mathbb{Z}) \to \wp(\mathbb{Z})$. By following similar reasoning done in Ex. 26, we can derive $1 \vdash [\delta_{\subseteq}^{\mathsf{Int}}]\, x > 0?\,;\, x := x - 1\, (\delta_{\subseteq}^{\mathsf{Int}})$ and $1 \vdash [\delta_{\subseteq}^{\mathsf{Int}}]\, x \leq 0?\,;\, x := x + 1\, (\delta_{\subseteq}^{\mathsf{Int}})$. The rule (**join**) then concludes with $2 \vdash [\delta_{\subseteq}^{\mathsf{Int}}]\, \mathsf{R}\, (\delta_{\subseteq}^{\mathsf{Int}})$ because $+(1,1) = 2$, thus stating that $[\![\mathsf{R}]\!]$ is 2-ALC w.r.t. $\langle \delta_{\subseteq}^{\mathsf{Int}}, \delta_{\subseteq}^{\mathsf{Int}} \rangle$. Although the conclusion is correct, it is not precise since $[\![\mathsf{R}]\!] \in 1\text{-}Lip\langle \delta_{\subseteq}^{\mathsf{Int}}, \delta_{\subseteq}^{\mathsf{Int}} \rangle$. The imprecision here arises from the bound function $\mathfrak{b} = +$, which overestimates the number of elements produced by the join of two intervals.

For the program R^*, however, the deductive system cannot prove ALC for any constant k. This is due to the fact that rule (**star**) cannot be applied when there is no constant bound m on the number of iterations of R^*, unless the input is restricted to a fixed bound.

As a direct consequence of Thm. 25, if we instantiate the abstractions with $\eta = \rho = id$ and δ is a metric, then the deductive rules of Fig. 4 derive judgments for the standard Lipschitz continuity of programs (cf. Def. 4 with $[\![\mathsf{P}]\!]$ as f). This is because all the predicates on abstractions, such as $\eta \in t\text{-}Lip\langle \delta^{\rho}, \delta^{\eta} \rangle$ for (**seq**) and (**star**), and $\rho \in t\text{-}Lip\langle \delta^{id}, \delta^{\rho} \rangle$ for (**join**), are trivially true (cf. Lem. 22).

▶ **Corollary 28.** *Let $\mathsf{P} \in \mathsf{Prog}$ and δ be a metric. Then:*

$$k \vdash [\delta^{id}]\, \mathsf{P}\, (\delta^{id}) \;\Rightarrow\; [\![\mathsf{P}]\!] \textit{ is } k\textit{-Lipschitz continuous w.r.t. } \langle \delta, \delta \rangle$$

▶ **Example 29.** Consider the metric space $\langle \mathbb{M}, \leq, \delta_2 \rangle$ where $\leq$ is assumed to be component-wise and δ_2 is the Euclidean distance (cf. Ex 2). Assume that $\mathsf{P} \in \mathsf{Prog}$ is an always terminating program and let $[\![\mathsf{P}]\!] \colon \mathbb{M} \to \mathbb{M}$ represents the standard denotational semantics mapping a program state to the resulting program state after execution of P. If we instantiate the deductive system of Fig. 4 with the abstraction $\eta = \rho = id$, the semantics $[\![\mathsf{P}]\!] \colon \mathbb{M} \to \mathbb{M}$ and the metric δ_2, then the inductive rules correspond to those proposed by Chaudhuri et al. in [9]. This shows that the deductive system presented by Chaudhuri et al. in [9] is in fact an instance of $k \vdash [\delta^{\eta}]\, \mathsf{P}\, (\delta^{\rho})$.

6 Related Work

Abstract Lipschitz continuity finds some instances in the literature. Fo instance, 0-ALC corresponds to require: $\forall x, y \in C.\ \delta_{\mathsf{D}}^{\rho}(f(x), f(y)) \leq 0$. When δ_{D} satisfies the (*iff-identity*) axiom, the notion collapses to Abstract Robustness [19] with different models of perturbation (qualitative, quantitative, or combined). In addition, when $\eta = \rho = id$ the notion collapses to the standard program Robustness notion [19].

As we have already discussed in Sec. 4, Partial completeness, whose underlying idea was to replace indistinguishability (of abstract computations) with similarity (measured by a pre-metric distance), has a strong relation with ALC. The same idea in the literature led to another notion that can be seen as an instantiation of our approach, which is Approximate

Non-Interference [34]. This notion, originally introduced in a probabilistic process algebra, requires the *observable* behaviors of two agents under a similarity threshold ε, instead of being identical (as required by standard Non-Interference [21]). Then, we can see Approximate Non-Interference as an instance of ALC, where the observation of the output is the abstraction, and a measured distance between these observables must be under a finite threshold, which is the finite distance between the input processes.

As discussed in Sec. 3, the standard mathematical notion of Lipschitz continuity is an instance of ALC. In particular, when f is the standard input/output denotational program semantics $[\![\cdot]\!]\colon \mathsf{Prog} \to \mathbb{M} \to \mathbb{M}$ and the distance considered is the standard Euclidean metric, then ALC corresponds to the Lipschitz continuity of programs as formalized by Chaudhuri et al. in [9, 10] (referred to as *program Robustness*). We have also shown in Ex. 29 that the proof system in Fig. 4 is a strict generalization of the one in proposed in [9]. This is because the triple $k \vdash [\delta^\eta]\,\mathsf{P}\,(\delta^\rho)$ enables reasoning about property perturbations, encoded with input and output abstractions, over weaker distances (pre-metric spaces) of *any* additive program semantics. It is also worth noting that, in contrast to [9], our proposed deductive system tracks the necessary assumptions for the base cases $[\![\mathsf{c}]\!]$ required to apply the inductive rules, whereas in [9] the authors also provide an analysis for the base cases.

7 Conclusion

Abstract Lipschitz continuity is a generalization of the classical mathematical notion of Lipschitz continuity. It is parameterized by input and output pre-metric spaces, as well as by input and output domain abstractions, which are formalized as upper closure operators. This generalized framework enables the formalization of properties of the form: "*Perturbations in the input properties induce proportionally bounded (linear) changes in the output properties*". We also formally proved its relation with the Partial Completeness property in abstract interpretation, by isolating the constraint under which the two notions, apparently unrelated, have a strong relation. Finally, we developed a deductive system for proving the ALC property of additive semantics of programs.

The proposed ALC notion is a *global* property, in the sense that it is universally quantified over all inputs. As a future work, we plan to formalize its *local* version, namely requiring ALC over a strict subset of the input domain, and study its relation with other local properties in the context of abstract interpretation [2, 3, 5]. Dropping the universal quantification may invalidate the correlation already established between the global counterparts. Also, reasoning about local properties may be more challenging, as the proposed deductive system requires nontrivial modifications to be used for proving ALC on a subset of executions.

We formalized abstractions as ucos, which have been proven to be equivalent to Galois insertions [12], namely admitting a surjective best abstraction function. In the future, we would like to consider weaker abstraction notions able to formalize properties that do not necessarily admit a best abstraction function, such as the domain of convex polyhedra [22]. In this direction, the notion of weak closures defined in [28] could be considered.

Finally, in [28] the authors showed that, under certain assumptions, there is a correspondence between the Completeness property in abstract interpretation and the Abstract Non-Interference (ANI) property in language based security [17, 18]. While ANI does not directly model continuity properties of functions, the connection established in [28], together with Cor. 19, suggests a potential relation between ANI and ALC, which we plan to investigate as future work. The same could also apply to other quantitative program properties, like Quantitative Input Data Usage [29, 30].

References

1 Kumail Alhamoud, Hasan Abed Al Kader Hammoud, Motasem Alfarra, and Bernard Ghanem. Generalizability of adversarial robustness under distribution shifts. *Trans. Mach. Learn. Res.*, 2023, 2023. URL: https://openreview.net/forum?id=XNFo3dQiCJ.

2 Roberto Bruni, Roberto Giacobazzi, Roberta Gori, and Francesco Ranzato. A logic for locally complete abstract interpretations. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pages 1–13. IEEE, 2021. doi:10.1109/LICS52264.2021.9470608.

3 Roberto Bruni, Roberto Giacobazzi, Roberta Gori, and Francesco Ranzato. A correctness and incorrectness program logic. *J. ACM*, 70(2):15:1–15:45, 2023. doi:10.1145/3582267.

4 Marco Campion, Mila Dalla Preda, and Roberto Giacobazzi. Partial (in)completeness in abstract interpretation: limiting the imprecision in program analysis. *Proc. ACM Program. Lang.*, 6(POPL):1–31, 2022. doi:10.1145/3498721.

5 Marco Campion, Mila Dalla Preda, Roberto Giacobazzi, and Caterina Urban. Monotonicity and the precision of program analysis. *Proc. ACM Program. Lang.*, 8(POPL):1629–1662, 2024. doi:10.1145/3632897.

6 Marco Campion, Mila Dalla Preda, and Roberto Giacobazzi. On the properties of partial completeness in abstract interpretation. In Ugo Dal Lago and Daniele Gorla, editors, *Proceedings of the 23rd Italian Conference on Theoretical Computer Science, ICTCS 2022, Rome, Italy, September 7-9, 2022*, volume 3284 of *CEUR Workshop Proceedings*, pages 79–85. CEUR-WS.org, 2022. URL: https://ceur-ws.org/Vol-3284/8665.pdf.

7 Marco Campion, Caterina Urban, Mila Dalla Preda, and Roberto Giacobazzi. A formal framework to measure the incompleteness of abstract interpretations. In Manuel V. Hermenegildo and José F. Morales, editors, *Static Analysis - 30th International Symposium, SAS 2023, Cascais, Portugal, October 22-24, 2023, Proceedings*, volume 14284 of *Lecture Notes in Computer Science*, pages 114–138. Springer, 2023. doi:10.1007/978-3-031-44245-2_7.

8 Swarat Chaudhuri, Sumit Gulwani, and Roberto Lublinerman. Continuity Analysis of Programs. In *Proceedings of the 37th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '10, page 57–70, New York, NY, USA, 2010. Association for Computing Machinery. doi:10.1145/1706299.1706308.

9 Swarat Chaudhuri, Sumit Gulwani, and Roberto Lublinerman. Continuity and Robustness of Programs. 55(8):107–115, 2012. doi:10.1145/2240236.2240262.

10 Swarat Chaudhuri, Sumit Gulwani, Roberto Lublinerman, and Sara NavidPour. Proving Programs Robust. In Tibor Gyimóthy and Andreas Zeller, editors, *SIGSOFT/FSE'11 19th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE-19) and ESEC'11: 13th European Software Engineering Conference (ESEC-13), Szeged, Hungary, September 5-9, 2011*, pages 102–112. ACM, 2011. doi:10.1145/2025113.2025131.

11 Patrick Cousot and Radhia Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In Robert M. Graham, Michael A. Harrison, and Ravi Sethi, editors, *Conference Record of the Fourth ACM Symposium on Principles of Programming Languages, Los Angeles, California, USA, January 1977*, pages 238–252. ACM, 1977. doi:10.1145/512950.512973.

12 Patrick Cousot and Radhia Cousot. Systematic design of program analysis frameworks. In Alfred V. Aho, Stephen N. Zilles, and Barry K. Rosen, editors, *Conference Record of the Sixth Annual ACM Symposium on Principles of Programming Languages, San Antonio, Texas, USA, January 1979*, pages 269–282. ACM Press, 1979. doi:10.1145/567752.567778.

13 Patrick Cousot and Radhia Cousot. An abstract interpretation-based framework for software watermarking. In *Proceedings of the 31st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '04, pages 173–185, New York, NY, USA, 2004. Association for Computing Machinery. doi:10.1145/964001.964016.

14 Lawrence C Evans. *Partial differential equations*, volume 19. American Mathematical Society, 2022.

15 Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George J. Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 11423–11434, 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/95e1533eb1b20a97777749fb94fdb944-Abstract.html.

16 Roberto Giacobazzi, Francesco Logozzo, and Francesco Ranzato. Analyzing program analyses. In Sriram K. Rajamani and David Walker, editors, *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2015, Mumbai, India, January 15-17, 2015*, pages 261–273. ACM, 2015. doi:10.1145/2676726.2676987.

17 Roberto Giacobazzi and Isabella Mastroeni. Abstract non-interference: parameterizing non-interference by abstract interpretation. In Neil D. Jones and Xavier Leroy, editors, *Proceedings of the 31st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2004, Venice, Italy, January 14-16, 2004*, pages 186–197. ACM, 2004. doi:10.1145/964001.964017.

18 Roberto Giacobazzi and Isabella Mastroeni. Abstract non-interference: A unifying framework for weakening information-flow. *ACM Trans. Priv. Secur.*, 21(2):9:1–9:31, 2018. doi:10.1145/3175660.

19 Roberto Giacobazzi, Isabella Mastroeni, and Elia Perantoni. Adversities in abstract interpretation - accommodating robustness by abstract interpretation. *ACM Trans. Program. Lang. Syst.*, 46(2):5, 2024. doi:10.1145/3649309.

20 Roberto Giacobazzi, Francesco Ranzato, and Francesca Scozzari. Making abstract interpretations complete. *J. ACM*, 47(2):361–416, 2000. doi:10.1145/333979.333989.

21 Joseph A. Goguen and José Meseguer. Security policies and security models. In *1982 IEEE Symposium on Security and Privacy, Oakland, CA, USA, April 26-28, 1982*, pages 11–20. IEEE Computer Society, 1982. doi:10.1109/SP.1982.10014.

22 Branko Grünbaum, Victor Klee, Micha A Perles, and Geoffrey Colin Shephard. *Convex polytopes*, volume 16. Springer, 1967.

23 Yujia Huang, Huan Zhang, Yuanyuan Shi, J. Zico Kolter, and Anima Anandkumar. Training certifiably robust neural networks with efficient local lipschitz bounds. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 22745–22757, 2021. URL: https://proceedings.neurips.cc/paper/2021/hash/c055dcc749c2632fd4dd806301f05ba6-Abstract.html.

24 Ziwei Ji and Matus Telgarsky. Directional convergence and alignment in deep learning. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL: https://proceedings.neurips.cc/paper/2020/hash/c76e4b2fa54f8506719a5c0dc14c2eb9-Abstract.html.

25 Lin Li, Yifei Wang, Chawin Sitawarin, and Michael W. Spratling. Oodrobustbench: a benchmark and large-scale analysis of adversarial robustness under distribution shift. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL: https://openreview.net/forum?id=kAFevjEYsz.

26 Dennis Liew, Tiago Cogumbreiro, and Julien Lange. Sound and partially-complete static analysis of data-races in GPU programs. *Proc. ACM Program. Lang.*, 8(OOPSLA2):2434–2461, 2024. doi:10.1145/3689797.

27 Francesco Logozzo. Towards a quantitative estimation of abstract interpretations. In *Workshop on Quantitative Analysis of Software*. Microsoft, June 2009. URL: https://www.microsoft.com/en-us/research/publication/towards-a-quantitative-estimation-of-abstract-interpretations/.

28 Isabella Mastroeni and Michele Pasqua. Domain precision in galois connection-less abstract interpretation. In Manuel V. Hermenegildo and José F. Morales, editors, *Static Analysis - 30th International Symposium, SAS 2023, Cascais, Portugal, October 22-24, 2023, Proceedings*, volume 14284 of *Lecture Notes in Computer Science*, pages 434–459. Springer, 2023. doi:10.1007/978-3-031-44245-2_19.

29 Denis Mazzucato, Marco Campion, and Caterina Urban. Quantitative input usage static analysis. In Nathaniel Benz, Divya Gopinath, and Nija Shi, editors, *NASA Formal Methods - 16th International Symposium, NFM 2024, Moffett Field, CA, USA, June 4-6, 2024, Proceedings*, volume 14627 of *Lecture Notes in Computer Science*, pages 79–98. Springer, 2024. doi:10.1007/978-3-031-60698-4_5.

30 Denis Mazzucato, Marco Campion, and Caterina Urban. Quantitative static timing analysis. In Roberto Giacobazzi and Alessandra Gorla, editors, *Static Analysis - 31st International Symposium, SAS 2024, Pasadena, CA, USA, October 20-22, 2024, Proceedings*, volume 14995 of *Lecture Notes in Computer Science*, pages 268–299. Springer, 2024. doi:10.1007/978-3-031-74776-2_11.

31 Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 807–814. Omnipress, 2010. URL: https://icml.cc/Conferences/2010/papers/432.pdf.

32 Yurii Nesterov. *Lectures on Convex Optimization*. Springer Publishing Company, Incorporated, 2nd edition, 2018. doi:10.1007/978-3-319-91578-4.

33 Peter W. O'Hearn. Incorrectness logic. *Proc. ACM Program. Lang.*, 4(POPL):10:1–10:32, 2020. doi:10.1145/3371078.

34 Alessandra Di Pierro, Chris Hankin, and Herbert Wiklicky. Approximate non-interference. *J. Comput. Secur.*, 12(1):37–82, 2004. doi:10.3233/JCS-2004-12103.

35 Alessandra Di Pierro and Herbert Wiklicky. Measuring the precision of abstract interpretations. In Kung-Kiu Lau, editor, *Logic Based Program Synthesis and Transformation, 10th International Workshop, LOPSTR 2000 London, UK, July 24-28, 2000, Selected Papers*, volume 2042 of *Lecture Notes in Computer Science*, pages 147–164. Springer, 2000. doi:10.1007/3-540-45142-0_9.

36 Xavier Rival and Kwangkeun Yi. *Introduction to static analysis: an abstract interpretation perspective*. Mit Press, 2020.

37 Daniel Schoepe and Andrei Sabelfeld. Understanding and enforcing opacity. In *2015 IEEE 28th Computer Security Foundations Symposium*, pages 539–553, 2015. doi:10.1109/CSF.2015.41.

38 Pascal Sotin. Quantifying the Precision of Numerical Abstract Domains. Research report, February 2010. URL: https://inria.hal.science/inria-00457324.

39 Bohang Zhang, Du Jiang, Di He, and Liwei Wang. Rethinking lipschitz neural networks and certified robustness: A boolean function perspective. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL: http://papers.nips.cc/paper_files/paper/2022/hash/7b04ec5f2b89d7f601382c422dfe07af-Abstract-Conference.html.

A Proofs for Section 4 (Abstract Lipschitz Continuity for Programs)

▶ **Theorem 18.** *Let $\langle C, \preceq_{\mathtt{C}}, \delta_{\mathtt{C}}\rangle$ and $\langle D, \preceq_{\mathtt{D}}, \delta_{\mathtt{D}}\rangle$ be pre-metric compatible spaces, let $\eta \in uco(C)$, $\rho \in uco(D)$ be abstractions, and let $k \in \mathbb{R}_{\geq 0}$. Consider a monotone function $f : C \to D$. Then, if f satisfies k-ALC w.r.t. $\langle \delta^{\eta}_{\mathtt{C}}, \delta^{\rho}_{\mathtt{D}}\rangle$, it also satisfies 0-Partial Completeness w.r.t. $\langle \eta, \delta^{\rho}_{\mathtt{D}}\rangle$, namely:*

$$[\forall x, y \in C.\ \delta^{\rho}_{\mathtt{D}}(f(x), f(y)) \leq \ k\delta^{\eta}_{\mathtt{C}}(x, y)] \ \Rightarrow\ [\forall x \in C.\ \delta^{\rho}_{\mathtt{D}}(f(x), f\eta(x)) \leq 0]$$

Proof. Let us assume Abstract k-Lipschitz Continuity w.r.t. $\langle \delta^{\eta}_{\mathtt{C}}, \delta^{\rho}_{\mathtt{D}}\rangle$, namely $\forall x, y \in C.\ \delta^{\rho}_{\mathtt{D}}(f(x), f(y)) \leq \ k\delta^{\eta}_{\mathtt{C}}(x, y)$. We have to prove 0-Partial Completeness w.r.t. $\langle \eta, \delta^{\rho}_{\mathtt{D}}\rangle$, namely $\forall x \in C.\ \delta^{\rho}_{\mathtt{D}}(f(x), f\eta(x)) \leq 0$. Let $y = \eta(x)$. Since Abstract k-Lipschitz Continuity holds, we have $\forall x \in C.\ \delta^{\rho}_{\mathtt{D}}(f(x), f(\eta(x))) \leq \ k\delta^{\eta}_{\mathtt{C}}(x, \eta(x))$ and, by idempotence of η, we have $\delta^{\eta}_{\mathtt{C}}(x, \eta(x)) = 0$. Hence, 0-Partial Completeness w.r.t. $\langle \eta, \delta^{\rho}_{\mathtt{D}}\rangle$ holds. ◀ ◀

▶ **Corollary 19.** *If $\langle D, \preceq_{\mathtt{D}}, \delta_{\mathtt{D}}\rangle$ is a quasisemi-metric compatible space then k-ALC w.r.t. $\langle \delta^{\eta}_{\mathtt{C}}, \delta^{\rho}_{\mathtt{D}}\rangle$ implies Completeness w.r.t. $\langle \eta, \rho\rangle$.*

Proof. Continuing the proof of Thm. 18, we reached 0-Partial Completeness w.r.t. $\langle \eta, \delta^{\rho}_{\mathtt{D}}\rangle$ because, by fixing $y = \eta(x)$ and by the idempotence of η, we get $\forall x \in C.\ \delta^{\rho}_{\mathtt{D}}(f(x), f\eta(x)) \leq 0$. Then, since $\delta_{\mathtt{D}}$ is a quasisemi-metric, it satisfies the *(iff-identity)* axiom (together with the (*non-negativity*)), and so $\delta^{\rho}_{\mathtt{D}}(f(x), f\eta(x)) \leq 0$ corresponds to $\delta^{\rho}_{\mathtt{D}}(f(x), f\eta(x)) = 0$ which implies $\forall x \in C.\ \rho f(x) = \rho f\eta(x)$. ◀

B Proofs for Section 5 (Proving Abstract Lipschitz Continuity for Programs)

▶ **Theorem 25** (Soundness). *Let $\mathsf{P} \in \mathsf{Prog}$, δ be a pre-metric and $\eta, \rho \in uco(C)$ be the input and output abstractions, respectively. Then:*

$$k \vdash [\delta^{\eta}]\, \mathsf{P}\, (\delta^{\rho}) \ \Rightarrow\ [\![\mathsf{P}]\!] \in k\text{-}Lip\langle \delta^{\eta}, \delta^{\rho}\rangle$$

Proof. (**base**): immediate by the definition of $k \vdash [\delta^{\eta}]\, \mathsf{c}\, (\delta^{\rho})$ and the assumption $[\![\mathsf{c}]\!] \in k\text{-}Lip\langle \delta^{\eta}, \delta^{\rho}\rangle$.

(**weaken**): The proof for the weakening of k is immediate. Let us show the proof for weakening the input abstraction η'. Assume $k \vdash [\delta^{\eta'}]\, \mathsf{P}\, (\delta^{\rho})$ and $\eta' \in t\text{-}Lip\langle \delta^{\eta}, \delta^{\eta'}\rangle$. The second assumption can be written as $\forall c_1, c_2 \in C$: $\delta^{\eta'}(\eta'(c_1), \eta'(c_2)) \leq t\delta^{\eta}(c_1, c_2)$ which, by the idempotence property of η', corresponds to $\delta^{\eta'}(c_1, c_2) \leq t\delta^{\eta}(c_1, c_2)$. We get the following derivations $\forall c_1, c_2 \in C$:

$$\begin{aligned}
\delta^{\rho}([\![\mathsf{P}]\!]c_1, [\![\mathsf{P}]\!]c_2) \ &\leq\ [\text{by } k \vdash [\delta^{\eta'}]\, \mathsf{P}\, (\delta^{\rho})] \\
k\delta^{\eta'}(c_1, c_2) \ &\leq\ [\text{by } \eta' \in t\text{-}Lip\langle \delta^{\eta}, \delta^{\eta'}\rangle] \\
tk\delta^{\eta}(c_1, c_2) \ &\Leftrightarrow\ [\text{by judgment definition}] \\
tk \vdash [\delta^{\eta}]\, \mathsf{P}\, (\delta^{\rho})
\end{aligned}$$

The proof for weakening the output abstraction ρ is similar and therefore omitted.

(**seq**): Assume we have a derivation $k_1 \vdash [\delta^{\eta}]\, \mathsf{P}_1\, (\delta^{\rho})$ for program P_1, a derivation $k_2 \vdash [\delta^{eta}]\, \mathsf{P}_2\, (\delta^{\rho})$ for program P_2, and $\eta \in t\text{-}Lip\langle \delta^{\rho}, \delta^{\eta}\rangle$. By η idempotent, the last assumption can be written as: $\forall c_1, c_2 \in C.\ \delta^{\eta}(c_1, c_2) \leq t\delta^{\rho}(c_1, c_2)$. Then we get the following derivations $\forall c_1, c_2 \in C$:

$$\delta^{\rho}([\![\mathsf{P}_1; \mathsf{P}_2]\!]c_1, [\![\mathsf{P}_1; \mathsf{P}_2]\!]c_2) \ =\ [\text{by definition of } [\![\mathsf{P}_1; \mathsf{P}_2]\!] \text{ and } (\textit{if-identity}) \text{ of } \delta^{\rho}]$$

$$
\begin{aligned}
\delta^\rho([\![\mathsf{P}_2]\!][\![\mathsf{P}_1]\!]c_1, [\![\mathsf{P}_2]\!][\![\mathsf{P}_1]\!]c_2) \;&\leq\; [\text{by } k_2 \vdash [\delta^\eta]\, \mathsf{P}_2\, (\delta^\rho)] \\
k_2\delta^\eta([\![\mathsf{P}_1]\!]c_1, [\![\mathsf{P}_1]\!]c_2) \;&\leq\; [\text{by } \eta \in t\text{-}Lip\langle \delta^\rho, \delta^\eta\rangle] \\
tk_2\delta^\rho([\![\mathsf{P}_1]\!]c_1, [\![\mathsf{P}_1]\!]c_2) \;&\leq\; [\text{by } k_1 \vdash [\delta^\eta]\, \mathsf{P}_1\, (\delta^\rho)] \\
k_1\, tk_2\delta^\eta(c_1, c_2) \;&\Leftrightarrow\; [\text{by judgment definition}] \\
k_1\, tk_2 \vdash [\delta^\eta]\, \mathsf{P}_1; \mathsf{P}_2\, (\delta^\rho)&
\end{aligned}
$$

(**join**): Assume we have a derivation $k_1 \vdash [\delta^\eta]\, \mathsf{P}_1\, (\delta^\rho)$ for program P_1, a derivation $k_2 \vdash [\delta^\eta]\, \mathsf{P}_2\, (\delta^\rho)$ for program P_2, $\rho \in \delta\text{-}Lip\langle t, id\rangle\rho$, and the predicate $\oplus$-Bound$(\langle \eta, \rho\rangle, \mathfrak{b})$ holds for bound function $\mathfrak{b}$. By Lem. 22, the assumption $\rho \in t\text{-}Lip\langle \delta^{id}, \delta^\rho\rangle$ can be written as: $\forall c_1, c_2 \in C.\ \delta^\rho(c_1, c_2) \leq t\delta(c_1, c_2)$. Then we get the following derivations $\forall c_1, c_2 \in C$:

$$
\begin{aligned}
\delta^\rho([\![\mathsf{P}_1 \oplus \mathsf{P}_2]\!]c_1, [\![\mathsf{P}_1 \oplus \mathsf{P}_2]\!]c_2) \;&=\; [\text{by definition of } [\![\mathsf{P}_1 \oplus \mathsf{P}_2]\!] \text{ and } (\textit{if-identity}) \text{ of } \delta^\rho] \\
\delta^\rho([\![\mathsf{P}_1]\!]c_1 \vee [\![\mathsf{P}_2]\!]c_1, [\![\mathsf{P}_1]\!]c_2 \vee [\![\mathsf{P}_2]\!]c_2) \;&=\; [\text{by } \rho(\rho(c_1) \vee \rho(c_2)) = \rho(c_1 \vee c_2) \text{ and } (\textit{if-identity}) \text{ of } \delta^\rho] \\
\delta^\rho(\rho[\![\mathsf{P}_1]\!]c_1 \vee \rho[\![\mathsf{P}_2]\!]c_1, \rho[\![\mathsf{P}_1]\!]c_2 \vee \rho[\![\mathsf{P}_2]\!]c_2) \;&\leq\; [\text{by } \rho \in t\text{-}Lip\langle \delta^{id}, \delta^\rho\rangle] \\
t\delta(\rho[\![\mathsf{P}_1]\!]c_1 \vee \rho[\![\mathsf{P}_2]\!]c_1, \rho[\![\mathsf{P}_1]\!]c_2 \vee \rho[\![\mathsf{P}_2]\!]c_2) \;&\leq\; [\text{by } k_1 \vdash [\delta^\eta]\, \mathsf{P}_1\, (\delta^\rho),\ k_2 \vdash [\delta^\eta]\, \mathsf{P}_2\, (\delta^\rho),\ \oplus\text{-Bound}(\langle \eta, \rho\rangle, \mathfrak{b})] \\
\mathfrak{b}(k_1, k_2)t\delta^\eta(c_1, c_2) \;&\Leftrightarrow\; [\text{by judgment definition}] \\
\mathfrak{b}(k_1, k_2)t \vdash [\delta^\eta]\, \mathsf{P}_1 \oplus \mathsf{P}_2\, (\delta^\rho)&
\end{aligned}
$$

(**star**): Assume we have a derivation $k \vdash [\delta^\eta]\, \mathsf{P}\, (\delta^\rho)$ for program P, $\eta \in t\text{-}Lip\langle \delta^\rho, \delta^\eta\rangle$ and a bound m on the number of iterations by the predicate $*$-Bound(P^*, m). We obtain the following inequalities:

$$
\begin{aligned}
\delta^\rho([\![\mathsf{P}^*]\!]c_1, [\![\mathsf{P}^*]\!]c_2) \;&=\; [\text{by } *\text{-Bound}(\mathsf{P}^*, m),\ [\![\mathsf{P}]\!] \text{ additive and } (\textit{if-identity}) \text{ of } \delta^\rho] \\
\delta^\rho([\![\mathsf{P}]\!]^m c_1, [\![\mathsf{P}]\!]^m c_2) \;&=\; [\text{by definition of } [\![\mathsf{P}; \mathsf{P}]\!] \text{ and } (\textit{if-identity}) \text{ of } \delta^\rho] \\
\delta^\rho([\![\mathsf{P}]\!][\![\mathsf{P}]\!]^{m-1} c_1, [\![\mathsf{P}]\!][\![\mathsf{P}]\!]^{m-1} c_2) \;&\leq\; [\text{by } k \vdash [\delta^\eta]\, \mathsf{P}\, (\delta^\rho)] \\
k\delta^\eta([\![\mathsf{P}]\!]^{m-1} c_1, [\![\mathsf{P}]\!]^{m-1} c_2) \;&\leq\; [\text{by } \eta \in t\text{-}Lip\langle \delta^\rho, \delta^\eta\rangle] \\
tk\delta^\rho([\![\mathsf{P}]\!]^{m-1} c_1, [\![\mathsf{P}]\!]^{m-1} c_2) \;&\leq\; [\text{by applying } m-1 \text{ composition steps}] \\
(tk)^m \delta^\eta(c_1, c_2) \;&\Leftrightarrow\; [\text{by judgment definition}] \\
(tk)^m \vdash [\delta^\eta]\, \mathsf{P}^*\, (\delta^\rho)&
\end{aligned}
$$

◀ ◀