Logical optimality of groundness analysis

Francesca Scozzari

Laboratoire d'Informatique (LIX), École Polytechnique, F-91128 Palaiseau Cedex, France

Dipartimento di Informatica, Università di Pisa, Corso Italia 40, 56125 Pisa, Italy

E-mail: scozzari@di.unipi.it

Abstract

In the context of the abstract interpretation theory, we study the relations among various abstract domains for groundness analysis of logic programs. We reconstruct the well-known domain $\mathcal{P}os$ as a logical domain in a fully automatic way and we prove that it is the best abstract domain which can be set up from the property of groundness by applying logic operators only. We propose a new notion of optimality which precisely captures the relation between $\mathcal{P}os$ and its natural concrete domain. This notion enables us to discriminate between the various abstract domains for groundness analysis from a computational point of view and to compare their relative precision. Finally, we propose a new domain for groundness analysis which has the advantage of being independent from the specific program and we show its optimality.

Key words: Abstract interpretation, abstract domain, static analysis, logic programming, groundness, Heyting completion, intuitionistic logic.

1 Introduction

In the logic programming field, groundness is probably the most popular and also the most studied instance of static analysis. The aim of groundness analysis is to detect whether a variable is definitely ground, i.e., it is bound to a term which contains no variable. This allows optimizing compilers to significantly speed up unification – the prime computational step of any logic language. For example, consider the simple logic program P:

```
p(X) :- q(X,a).

q(Y,Z) :- Y=Z.
```

where a is a constant of the language and capitol letters denote variables. Computed answer solutions for the queries $p(X)$ and $q(Y, Z)$ are, respectively, the substitutions $\{ X \backslash a \}$ and $\{ Y \backslash Z \}$. Therefore, the result of groundness analysis is that every solution for the predicate p will be definitely ground.

In the *abstract interpretation* framework [7,8], much work has been devoted to develop and study abstract domains for groundness analysis of logic programs. The goal of abstract interpretation is to compute an *abstract semantics* which is obtained by replacing concrete values with abstract objects, i.e., elements of the abstract domain. In the last ten years, many domains have been proposed, from the simple domain $\mathcal{G}$ by Jones and Søndergaard [21], to more complex ones, like $\mathcal{D}ef$ [2] and $\mathcal{P}os$ [23,4,2]. The common basic idea is to use substitutions as concrete values and logical formulas as abstract objects. Basic groundness properties are expressed by variables and more structured formulas express relations among groundness of different variables. Intuitively, the abstract formula consisting of a single variable X denotes the set of all the substitutions which ground the program variable X. For instance, the abstract domain $\mathcal{G}$ ([21]) considers as abstract objects conjunctions of variables (plus a top object denoted by *true*), where the abstract value $X \wedge Y$ denotes that both variables X and Y are ground. In this domain, concrete values $\{ X \backslash a \}$ and $\{ Y \backslash Z \}$ are approximated, respectively, by X and *true*, the latter to denote no groundness information. In this context, the abstract semantics of the program P showed above returns *true* for the query $q(Y, Z)$, getting rid of the relation between the two variables introduced by the clause `q(Y,Z) :- Y=Z`. As a consequence, by propagating this information to the predicate p through the clause `p(X) :- q(X,a).`, the abstract semantics returns the answer *true* also for $p(X)$. Hence, this simple domain fails to detect that any answer for the predicate p will be definitely ground. A more precise abstract domain can be constructed by considering relations between groundness of different variables. If $Y \Leftrightarrow Z$ is an abstract object which stands for all substitutions where, for all possible instances, Y is ground if and only if Z is ground, the abstract semantics returns for the query $q(Y, Z)$ the answer $Y \Leftrightarrow Z$ and for $p(X)$ the more precise and expected answer X, which precisely says that any answer for the predicate p will be ground.

Different abstract domains lead to different precision levels of the abstract semantics. This is because in the abstract interpretation framework, the (optimal) abstract semantics can always be automatically derived from the concrete semantics. Roughly, such an abstract semantics is obtaining by miming the behavior of the concrete semantics on the abstract objects. Therefore, the choice of the abstract domain is the main point in the development of the analysis.

1.1 Motivations

$\mathcal{P}os$ is certainly a well studied domain for groundness analysis of logic programs [21,23,4,2]. It is also the most widely used, since it is able to characterize both pure groundness, i.e., if a variable is instantiated to a ground term, and groundness relations between different variables, i.e., whether the groundness of a variable depends on the groundness of other variables. In standard literature, the domain $\mathcal{P}os$ is built in 3 steps: First consider the set of formulas built from a finite set Var and connectives $\wedge, \vee$ and $\Rightarrow$; then consider the quotient with respect to classic logical equivalence, finally select only the positive formulas (which are true when all variables are set to true). The domain so obtained is then related to the concrete one (sets of substitutions closed by instantiation) through a suitable concretization function, explicitly proving that it induces a Galois insertion. This method of constructing domains suffers from many drawbacks. The most important one is that the domain has to be "invented" in some way, through a procedure which is usually not formally related to the property we analyze. After defining the abstract domain, it has to be explicitly proved that the domain is an abstraction of the concrete one and, more importantly, that it is actually useful for the analysis.

Our idea is that both the construction method and the logic formulas of the abstract domains should directly reflect the concrete domain and the property to be analyzed. As logic programs compute substitutions and groundness is a property closed by instantiation, the most natural choice for the concrete domain contains as elements sets of substitutions closed by instantiation, denoted by $\wp^{\downarrow}(Sub)$. We show that the propositional formulas used in the definition of $\mathcal{P}os$ do not follow this approach, i.e., they are not explicitly related to the concrete domain. In order to understand this apparent asymmetry between $\mathcal{P}os$ and its concrete domain, we just need to make the following observations.

- $\mathcal{P}os$ is defined as a Boolean algebra with connectives $\wedge, \vee$ and $\Rightarrow$. Since $\wp^{\downarrow}(Sub)$ is not Boolean, $\mathcal{P}os$ can not inherit its algebraic properties from the concrete domain. Therefore, the construction process to set up the abstract domain $\mathcal{P}os$ turns out to be unrelated to the concrete domain $\wp^{\downarrow}(Sub)$.
- Even the basic operations of $\mathcal{P}os$, $\wedge, \vee$ and $\Rightarrow$, except for conjunction, do not derive from corresponding operations in the concrete domain. When we try to compare the operations on $\mathcal{P}os$ to the corresponding operations on $\wp^{\downarrow}(Sub)$, we find out that only the conjunction ($\wedge$) on $\mathcal{P}os$ come from the meet on $\wp^{\downarrow}(Sub)$, which is set intersection. In fact, the disjunction ($\vee$) of $\mathcal{P}os$ does not correspond to the concrete join (set union), as proved in [11]. Moreover, in the concrete domain, we can not have a notion of (classic) implication, since it is not a Boolean algebra. So there is no way to inherit the implication $\Rightarrow$ from a corresponding operation of $\wp^{\downarrow}(Sub)$.

1.2 The intuition behind our reconstruction of $\mathcal{P}os$

The main problem in the standard definition of $\mathcal{P}os$ is that its natural concrete domain is not a Boolean algebra. In particular, it does not allow us to define an operation of classic implication. But $\wp^{\downarrow}(Sub)$ is a rich enough algebra to enable us to define a notion of *intuitionistic implication* (or *relative pseudocomplement*). The intuitionistic implication is a generalization of the classic one which leads to a weaker notion of implication. Intuitively, given two formulas a and b, the intuitionistic implication $a \to b$ is defined as the most abstract formula c such that $a \wedge c$ derives b. Moving from the logic setting to the algebraic one, it is worth noting that not all concrete domains admit a notion of intuitionistic implication between every pair of objects. In other words, given two concrete object a and b, such a most abstract object $a \to b$ may fail to exist. When intuitionistic implications do exist for every pair of concrete objects, we say that the concrete domain is a model of *intuitionistic logic*. We briefly recall the main features of intuitionistic logic (see [20,27,3]).

- It is the weakest logic where the *modus ponens* law still holds, that is given two objects a and b it holds that $a \wedge (a \to b)$ derives b, i.e., $a \wedge (a \to b) \leq b$
- The negation $\neg a$ is defined as $a \to false$, whose algebraic counterpart is the notion of pseudocomplement.
- Implication and disjunction become independent operations. For instance, the classic tautology $a \to b = \neg a \vee b$ does not hold anymore.
- As a consequence, also the tautology $a \vee \neg a = true$ does not hold.

In this paper, we show a different and automatic way of building the abstract domain $\mathcal{P}os$, which directly comes from the definition of groundness and the concrete domain properties. The simplest domain defining the property of groundness is the most abstract domain containing Var, which is $\mathcal{G} = \wp(Var)$ by Jones and Søndergaard [21], where each variable v denotes the set of substitutions which ground v. It is well known that $\wp^{\downarrow}(Sub)$ is a model of intuitionistic logic. Thus we can look at the operations $\cap$ and $\cup$ of $\wp^{\downarrow}(Sub)$ as logical connectives which precisely correspond to the $\wedge$ and $\vee$ of intuitionistic logic and, more importantly, we can define a notion of intuitionistic implication $\to$ for every pair of concrete objects. Let us denote by $A \xrightarrow{\curlywedge} B$ the space of all intuitionistic implications between two abstract domains A and B. In the first part of the paper, we show that $\mathcal{P}os$ is exactly the most abstract domain which contains all the (double) intuitionistic implications between elements in $\mathcal{G}$, i.e., $\mathcal{P}os = (\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) \xrightarrow{\curlywedge} \mathcal{G}$. This result generalizes a similar one for the abstract domain $\mathcal{D}ef$ in [17] which proves that $\mathcal{D}ef$ contains all the intuitionistic implications between elements in $\mathcal{G}$, i.e., $\mathcal{D}ef = \mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}$.

Our construction allows us to study in detail the relations between all these domains and to compare their expressive power and precision in a very simple

way. We derive many useful properties, such as a normal form for its elements and a new result of optimality which precisely discriminates between the two domains $\mathcal{D}ef$ and $\mathcal{P}os$. This allows us to answer to some open questions such as "why $\mathcal{P}os$ is considered optimal" from a computational point of view. Our formalization of $\mathcal{P}os$ enjoys the following properties.

- $\mathcal{P}os$ is built by using only the definition of groundness (the domain $\mathcal{G}$) and the properties of the concrete domain. We need not use quotient, Boolean algebra, nor positive formulas. The construction of the abstract domains $\mathcal{G}, \mathcal{D}ef$ and $\mathcal{P}os$ now follows a unique, precise logic: The algebraic structure of the concrete domain. Moreover, we no longer need to prove that these domains are indeed abstractions of the concrete one, since it holds by construction. Also the relations among all the domains are now automatically derivable.
- The operations which characterize $\mathcal{P}os$ are now exactly the same of the concrete one: $\wedge$ and $\rightarrow$ in $\mathcal{P}os$ precisely correspond to $\cap$ and $\rightarrow$ in $\wp^{\downarrow}(Sub)$ and no other operation is considered in the construction process.
- We immediately obtain a theorem of representation for $\mathcal{P}os$ which states that every element in $\mathcal{P}os$ is an implication between two elements in $\mathcal{D}ef$ or, by using the characterization of $\mathcal{D}ef$, that every element in $\mathcal{P}os$ is an implication of implications between elements in $\mathcal{G}$, independently from the cardinality of Var.
- We answer to some open questions on $\mathcal{P}os$. The join on $\mathcal{P}os$ differs from the join on $\wp^{\downarrow}(Sub)$ but, in some cases, they coincide.

  $$Why\ \mathcal{P}os \text{ contains exactly those concrete joins?}$$

  Our representation theorem states that the abstract join on $\mathcal{P}os$ coincides with the concrete one (set union) if and only if it can be written as an intuitionistic implication.
- Since $\mathcal{P}os$ is now a subalgebra of $\wp^{\downarrow}(Sub)$ with respect to the operations $\wedge$ and $\rightarrow$, all the properties of $\mathcal{P}os$ are directly derivable from the properties of the concrete domain. In particular, since $\wp^{\downarrow}(Sub)$ is a model of intuitionistic propositional logic, it follows that also $\mathcal{P}os$ is. Therefore we gain from the logic an axiomatization for $\mathcal{P}os$.

In the second part of the paper, we use our formalization of $\mathcal{P}os$ in order to understand why $\mathcal{P}os$ can be considered a "good" domain. To this end, we try to refine $\mathcal{P}os$. We wonder which is the domain which contains the implications between formulas in $\mathcal{P}os$ and find out an important closure property: $\mathcal{P}os \xrightarrow{\wedge} \mathcal{P}os = \mathcal{P}os$. This implies that $\mathcal{P}os$ can not be further refined with respect to intuitionistic implication. Therefore $\mathcal{P}os$ is the most abstract domain which contains $\mathcal{G}$ and is closed with respect to concrete $\wedge$ and $\rightarrow$. We then consider the disjunctive completion of $\mathcal{P}os$, denoted by $\curlyvee(\mathcal{P}os)$, which includes all unions of formulas in $\mathcal{P}os$. Filé and Ranzato proved in [11] that $\mathcal{P}os$ is strictly contained in $\curlyvee(\mathcal{P}os)$. We try to refine $\curlyvee(\mathcal{P}os)$ by allowing implications between disjunctive formulas and we prove that it can not be

further refined, i.e., $\curlyvee(\mathcal{P}os) \xrightarrow{\curlywedge} \curlyvee(\mathcal{P}os) = \curlyvee(\mathcal{P}os)$.

In the last part, we propose a new, program independent domain for groundness analysis, which is able to describe groundness relations for infinite sets of variables. Finally, we show that the new domain still enjoys the closure property of $\mathcal{P}os$.

1.3 Related work

Much work has been devoted to the study of abstract domains for groundness analysis. The first ones concentrated on the definition of groundness and basic properties [21,9], while the last ones studied different characterizations of various abstract domains. [23] proposed propositional formulas to represent groundness relations. Many authors followed this approach [5,2] and contributed to develop and study the domains $\mathcal{D}ef$ and $\mathcal{P}os$, while others focused on the abstract operations or slightly different characterizations [24,22].

All these authors share the same approach: They construct abstract domains independently from the property to be analyzed and then prove some properties of the abstraction. Their attention is entirely focused on the representation of formulas in the abstract domains. This forces to always work up to isomorphism.

Our idea is to concentrate exclusively on the property of groundness. We reconstruct the domain $\mathcal{P}os$ in a systematic way and show that it is possible to avoid *ad hoc* characterizations in the construction process of domains. A first example in this direction is shown in [14], where the abstract domain $\mathcal{D}ef$ has been reconstructed as the space of monotone functions between elements in $\mathcal{G}$. Our work is based on the Heyting completion refinement operator [17]. The first use of this operator has been presented in [17], where Heyting completion is used to construct the abstract domain $\mathcal{D}ef$, by exploiting the characterization given in [14]. An attempt of building the domain $\mathcal{P}os$ starting from the disjunctive completion of $\mathcal{G}$ is shown in [17]. Even in this case, the construction does not directly come from the basic property of groundness, but from a domain more complex than $\mathcal{G}$. Moreover, no property of optimality has been taken into account in this construction.

2 Preliminaries

Throughout the paper we assume familiarity with lattice theory [3,19], abstract interpretation [6,7,25] and logic programming [1].

2.1 Notation and basic notions

Let A, B and C be sets. $A \backslash B$ denotes the set-theoretic difference, $A \subset B$ the proper inclusion and, if $X \subseteq A$, $\overline{X}$ is the set-theoretic complement of X. $X \subseteq_f A$ denotes that X is a finite subset of A. The powerset of A is denoted by $\wp(A)$ and the set of all finite subsets of A by $\wp_f(A)$. If A is a poset, we usually denote the corresponding partial order by $\leq_A$ and, for $I \subseteq A$, $\downarrow I = \{x \in A \mid \exists y \in I.\ x \leq_A y\}$ is the downward closure of I. $\wp^{\downarrow}(A)$ denotes the set of downward closed elements of A, where $I \subseteq A$ is downward closed if $I = \downarrow I$. A complete lattice A with partial ordering $\leq_A$, least upper bound $\vee_A$ (join), greatest lower bound $\wedge_A$ (meet), greatest element $\top_A$ and least element $\bot_A$, is denoted by $\langle A, \leq_A, \vee_A, \wedge_A, \top_A, \bot_A \rangle$. $\wp(A)$ and $\wp^{\downarrow}(A)$ are complete lattice with respect to set-theoretic inclusion, where the join is set union and the meet is set intersection. We write $f : A \longmapsto B$ to mean that f is a total function from A to B. In the following we sometimes use Church's lambda notation for functions, so that a function f will be denoted by $\lambda x.f(x)$. If $C \subseteq A$ then $f(C) = \{f(x) \mid x \in C\}$. By $g \circ f$ we denote the composition $\lambda x.g(f(x))$. Let $\langle A, \leq_A, \vee_A, \wedge_A, \top_A, \bot_A \rangle$ and $\langle B, \leq_B, \vee_B, \wedge_B, \top_B, \bot_B \rangle$ be complete lattices. A function $f : A \longmapsto B$ is *additive* if for any $C \subseteq A$, $f(\vee_A C) = \vee_B f(C)$.

2.2 Abstract interpretation and Galois connections

The standard Cousot and Cousot theory of abstract interpretation is based on the notion of *Galois connection* [6]. If C and A are posets and $\alpha : C \longmapsto A$, $\gamma : A \longmapsto C$ are monotone functions, such that $\forall c \in C.\ c \leq_C \gamma(\alpha(c))$ and $\forall a \in A.\ \alpha(\gamma(a)) \leq_A a$, then we call the quadruple $\langle C, \alpha, A, \gamma \rangle$ a *Galois connection* between C and A. If in addition $\forall a \in A.\ \alpha(\gamma(a)) = a$, then $\langle C, \alpha, A, \gamma \rangle$ is a *Galois insertion* of A in C. In the setting of abstract interpretation, C and A are called, respectively, *concrete* and *abstract domain*, and they are assumed to be complete lattices. Any Galois connection $\langle C, \alpha, A, \gamma \rangle$ can be lifted to a Galois insertion by identifying in an equivalence class those objects in A having the same image (meaning) in C.

Let L be a complete lattice $\langle L, \leq_L, \vee_L, \wedge_L, \top_L, \bot_L \rangle$ playing the role of the concrete domain. An (*upper*) *closure operator* on L is a function $\alpha : L \longmapsto L$ monotone, idempotent and extensive (i.e., $\forall x \in L.\ x \leq_L \alpha(x)$) [25]. Each closure operator α is uniquely determined by the set of its fixpoints, which is its image $\alpha(L)$. $\alpha(L)$ is a complete lattice with respect to $\leq_L$, but, in general, it is not a complete sublattice of L, since the join in $\alpha(L)$ might be different from $\vee_L$. $\alpha(L)$ is a complete sublattice of L iff α is additive. $X \subseteq L$ is the set of fixpoints of a closure operator on L iff X is a *Moore family* of L, i.e., $\top_L \in X$ and X is completely meet-closed (i.e., for any non-empty $Y \subseteq X$,

$\wedge_L Y \in X$). For any $X \subseteq L$, we denote by $\curlywedge(X)$ the *Moore closure* of X, i.e., the least subset of L containing X, which is a Moore family of L. We denote by $\langle Moore(L), \sqsubseteq, \sqcap, \sqcup, \{\top\}, L\rangle$ the complete lattice of all Moore families of L. The ordering $\sqsubseteq$ is the inverse of set inclusion ($\supseteq$), $\sqcap$ is the most abstract Moore family which contains set union, and $\sqcup$ is simply set intersection. The equivalence between Galois insertions, closure operators and Moore families is well known [3]. However, closure operators and Moore families are often more practical and concise than Galois insertions to reason about abstract domains, being independent from representation choices for domain objects [7]. Any Galois insertion $\langle L, \alpha, A, \gamma\rangle$ is uniquely determined (up to isomorphism) by the closure operator $\gamma \circ \alpha$, and, conversely, any closure operator uniquely determines a Galois insertion (up to isomorphism). The complete lattice of all abstract domains (identified up to isomorphism) on L is therefore isomorphic to $Moore(L)$. The order relation on $Moore(L)$ corresponds to the ordering used to compare abstract domains with regard to their precision: If A and B are abstractions of L, then A is *more concrete* than B iff $A \sqsubseteq B$ as Moore families. In the following, given a Moore family X, we denote by α_X the corresponding closure operator.

2.3 Logic programming

Let $\mathcal{V}$ be a denumerable [1] set of variables. We fix a first-order language $\mathcal{L}$, with variables ranging in $\mathcal{V}$. *Term* is the set of terms of $\mathcal{L}$. For any syntactic object s, $vars(s)$ denotes the set of its variables. A term t is ground if $vars(t) = \emptyset$.

The set of idempotent *substitutions* [26], i.e., finite mappings from $\mathcal{V}$ to terms in $\mathcal{L}$, is denoted by Sub. If θ and σ are substitutions, $dom(\theta)$ denotes the set $\{\, v \in \mathcal{V} \mid \theta(v) \neq v. \,\}$, which is always finite by definition, and $\sigma \circ \theta$ denotes the substitution $\lambda v.\theta(\sigma(v))$. Objects in Sub are partially ordered by instantiation: $a \leq b$ iff $\exists \theta \in Sub.\ a = b \circ \theta$. If we add to Sub an extra element τ as least element, we obtain a complete lattice $\langle Sub^\tau, \vee, \wedge, \epsilon, \tau\rangle$ where $\vee$ is most specific anti-instance, $\wedge$ is usual unification and ϵ is the empty substitution. Note that unification $\wedge$ in Sub is a partial operation and it becomes total when considering the complete lattice Sub^τ. Substitutions are lifted to terms in the usual way.

2.4 Intuitionistic logic and Heyting algebras

Let L be a complete lattice and $a, b \in L$. The *pseudo-complement* (or *intuitionistic implication*) of a *relatively* to b, if it exists, is the unique element

[1] A set A is denumerable if $|A| = \omega$.

$a \to b \in L$ such that for any $x \in L$: $a \wedge_L x \leq_L b$ iff $x \leq_L a \to b$. Relative pseudo-complements, when they exist, are uniquely given by $a \to b = \bigvee_L \{ c \mid a \wedge_L c \leq_L b \}$. A complete lattice L is a *complete Heyting algebra* (cHa) if it is *relatively pseudo-complemented*, that is $a \to b$ exists for every $a, b \in L$. In particular, any cHa L is *distributive*, i.e., for all $a, b, c \in L$ it holds that $(a \vee_L b) \wedge_L c = (a \wedge_L c) \vee_L (b \wedge_L c)$. From a logical point of view, an Heyting algebra L is a set equipped with three operations (connectives) $\wedge, \vee, \to$ and constants $\bot, \top$ which satisfy the following equations for every $a, b, c \in L$:

$$a \to a = \top \qquad a \wedge \bot = \bot \tag{1}$$
$$(a \to b) \wedge b = b \qquad a \wedge (a \to b) = a \wedge b \tag{2}$$
$$a \to (b \wedge c) = (a \to b) \wedge (a \to c) \qquad (a \vee b) \to c = (a \to c) \wedge (b \to c) \tag{3}$$

Meet, join and relative pseudo-complement of Heyting algebras precisely correspond to conjunction, disjunction and intuitionistic implication of *intuitionistic logic* (see Birkhoff [3]). In the following, in order to keep distinct classic and intuitionistic implication, we always denote by $\Rightarrow$ the classic one and by $\to$ the intuitionistic one.

An example of cHa is the complete lattice $\langle \wp^{\downarrow}(Sub), \leq, \vee, \wedge, Sub, \emptyset \rangle$. The ordering $\leq$ is set inclusion and the logical operations $\vee$ and $\wedge$ correspond to $\cup$ and $\cap$ operations (set union and set intersection). Given $a, b \in \wp^{\downarrow}(Sub)$, the intuitionistic implication $a \to b = \bigvee \{ c \in \wp^{\downarrow}(Sub) \mid a \wedge c \leq b \}$ is also given by ([3])

$$a \to b = \{ \theta \in Sub \mid \forall \delta \leq \theta \ \delta \in a \ \Rightarrow \delta \in b \}.$$

Example 1 *Let $x, y, w \in \mathcal{V}$ be variables, $X = \{ \theta \in Sub \mid vars(\theta(x)) = \emptyset \}$ and $Y = \{ \theta \in Sub \mid vars(\theta(y)) = \emptyset \}$.*

$$\begin{aligned} X \to Y &= \{ \theta \in Sub \mid \forall \delta \leq \theta \ \delta \in X \ \Rightarrow \delta \in Y \} \\ &= \{ \theta \in Sub \mid \forall \delta \leq \theta \ vars(\delta(x)) = \emptyset \ \Rightarrow vars(\delta(y)) = \emptyset \}. \end{aligned}$$

Given a binary functor f, the substitution $\{ x \backslash f(y, w) \}$ belongs to $X \to Y$, while $\{ y \backslash f(x, w) \}$ does not.

3 Domains refinements

Domain refinements have been introduced in [10] in order to formalize enhancing operators on abstract domains. The idea is to describe the class of operators which, given an abstract domain A, return an abstract domain more concrete than A. More formally, a mapping $F : Moore(L) \to Moore(L)$ is a domain refinement if it is monotone and reductive (i.e., $F(A) \sqsubseteq A$ for any

$A \in Moore(L)$). In this section, we briefly recall the definitions of some refinement operators which will be used in the following.

3.1 Reduced product

The *reduced product* [7] of abstract domains corresponds to the meet operation ($\sqcap$) of closure operators. Given two abstract domains X and Y, the reduced product of X and Y, denoted by $X \sqcap Y$, is the most abstract domain which includes both X and Y, i.e., $X \sqcap Y = \curlywedge(X \cup Y)$.

3.2 Disjunctive completion

The *disjunctive completion* [7] of an abstract domain A is the most abstract domain which includes A and is a complete (join-)sublattice of L, i.e., abstract joins coincide with concrete ones. The disjunctive completion of A is defined as the most abstract domain which is an additive closure and includes A (cf. [7,15]):

$$\curlyvee(A) = \sqcup\{\, X \in Moore(L) \mid X \sqsubseteq A \text{ and } \alpha_X \text{ is additive} \,\}.$$

Proposition 2 *If L is a cHa and A is finite, $\curlyvee(A) = \{\, \vee_L X \mid X \subseteq A \,\}$.*

PROOF. L is distributive, since it is a cHa. Being a finite abstraction of L, $\{\, \vee_L X \mid X \subseteq A \,\}$ is also distributive (both abstract meet and abstract join coincide with the concrete ones). This assures us that $\{\, \vee_L X \mid X \subseteq A \,\}$ is a Moore family. □

3.3 Heyting completion

Heyting completion [17] is a refinement operator which has been recently introduced to logically interpret the Cousot and Cousot's *reduced cardinal power* refinement [7]. In this section we recall the definition and some basic results on Heyting completion refinement from [17,18]. The aim of this refinement is to enhance domains to represent relational information. The idea is to enrich an abstract domain by adding all the relative pseudo-complements (intuitionistic implications) built from every pair of elements in the given domain. From a logical point of view, the new domain is the collection of intuitionistic formulas built from the connectives $\wedge$ and $\rightarrow$, without nested implications.

For the sake of simplicity, we recall the definition only in case the concrete domain L is a cHa. Given two abstract domains A and B, the Heyting completion of A wrt B, denoted by $A \xrightarrow{\curlywedge} B$, is captured by the most abstract Moore family containing all the relative pseudo-complements (without nesting):

$$A \xrightarrow{\curlywedge} B = \bigwedge(\{\, a \to b \in L \mid a \in A, b \in B \,\}).$$

Heyting completion refinement is argumentwise monotone and reductive on the second argument, i.e., $B \xrightarrow{\curlywedge} A \sqsubseteq A$.

We recall some basic algebraic properties of Heyting completion with respect to other domain operations. In particular, we consider reduced product ($\sqcap$) and disjunctive completion ($\curlyvee$) of abstract domains. In the following of the paper, $A, B, C \in \mathit{Moore}(L)$.

Proposition 3 ([18]) *Let L be a cHa.*

(1) $A \xrightarrow{\curlywedge} (B \sqcap C) = (A \xrightarrow{\curlywedge} B) \sqcap (A \xrightarrow{\curlywedge} C)$
(2) $(A \sqcap B) \xrightarrow{\curlywedge} C = A \xrightarrow{\curlywedge} (B \xrightarrow{\curlywedge} C)$
(3) $\curlyvee(A) \xrightarrow{\curlywedge} B = A \xrightarrow{\curlywedge} B$ *if A is finite.*

The main feature of Heyting completion is the reduction of the approximation error introduced in the abstract computations by using the meet operation. The next proposition shows a key property of abstract domain obtained by Heyting completion in terms of the achieved precision of the meet operation.

Proposition 4 ([18]) *Let L be a cHa. $B \sqsubseteq A \xrightarrow{\curlywedge} A$ if and only if for any $x, y \in L$ $\alpha_A(\alpha_A(x) \wedge y) = \alpha_A(\alpha_A(x) \wedge \alpha_B(y))$.*

The previous result formalizes the intuition that, given two concrete objects $a, b \in L$, if a belongs to the abstract domain A, i.e., $\alpha_A(a) = a$, than using $\alpha_B(b)$ insteads of b does not affect the precision of the abstract meet, when the result is observed in A, since $\alpha_A(a \wedge b) = \alpha_A(a \wedge \alpha_B(b))$. The idea is that, whenever at least one object belongs to A, than we can approximate the other one in any domain more concrete than $A \xrightarrow{\curlywedge} A$ without introducing approximation errors in computing abstract meets.

4 Properties of the concrete domain

Since logic programs compute substitutions and groundness is a property closed by instantiation, the most natural choice for concrete domain is $\wp^{\downarrow}(\mathit{Sub})$. Most of the works about domains for groundness analysis and, in general, for

properties closed by instantiation, use $\wp^{\downarrow}(Sub)$ [2] as concrete domain. A formal explanation of this choice, with respect to the more general $\wp(Sub)$, can be found in [13], where it is proved that $\wp^{\downarrow}(Sub)$ is "concrete enough" for performing groundness analysis. In other words, groundness can be observed on a semantics computed on $\wp^{\downarrow}(Sub)$ without loosing precision with respect to a more concrete semantics on $\wp(Sub)$.

We show a list of properties of this lattice which will be often used throughout the paper. It is well known that $\wp^{\downarrow}(Sub)$ is a cHa, where meet and join are precisely set intersection and set union. Moreover, $\wp^{\downarrow}(Sub)$ is completely distributive, that is (possibly infinite) joins distribute over (possibly infinite) meets. These properties are very useful in order to precisely characterize the intuitionistic implication on $\wp^{\downarrow}(Sub)$. Given $a, b \in \wp^{\downarrow}(Sub)$, the general definition of implication defines $a \to b$ as $\bigvee \{ c \in \wp^{\downarrow}(Sub) \mid a \wedge c \leq b \}$. Since $\wp^{\downarrow}(Sub)$ is completely distributive, it can be simplified in [3]

$$a \to b = \{ \theta \in Sub \mid \forall \delta \leq \theta \ \delta \in a \ \Rightarrow \delta \in b \}.$$

Let us first recall two basic properties of implication which allows us to decompose complex implications. Let $a \in \wp^{\downarrow}(Sub)$ and $\{ b_i \}_{i \in I} \subseteq \wp^{\downarrow}(Sub)$. Then the following two equalities hold ([3]).

$$a \to \wedge_{i \in I} b_i = \wedge_{i \in I} (a \to b_i)$$

$$(\vee_{i \in I} a_i) \to b = \wedge_{i \in I} (a_i \to b)$$

Moreover, by exploiting the above characterization of intuitionistic implication in $\wp^{\downarrow}(Sub)$ we can state the following result on the decomposition of implications.

Theorem 5 *Let $\sigma \in Sub$ and $b \in \wp^{\downarrow}(Sub)$.*

$$(\downarrow \sigma) \to b = \{ \theta \in Sub \mid \theta \wedge \sigma \in b \}$$ [3].

PROOF. Let $\theta \in \downarrow \sigma \to b$. By definition of implication, $\forall \delta \leq \theta$ it holds $\delta \leq \sigma \ \Rightarrow \delta \in b$. If θ and σ do not unify, there is nothing to prove. Otherwise, it holds $\theta \wedge \sigma \leq \theta$ and $\theta \wedge \sigma \leq \sigma$, therefore $\theta \wedge \sigma \in b$ since $\theta \in \downarrow \sigma \to b$.

Let θ such that $\theta \wedge \sigma \in b$ and let $\delta \leq \theta$. If $\delta \leq \sigma$ then $\delta \leq \theta \wedge \sigma$. Since $\theta \wedge \sigma \in b$ and b is downward closed, also $\delta \in b$. $\Box$

[2] More complex domains are used in [5] (a combination of $\wp^{\downarrow}(Sub)$ with sets of variables), and in [22] (the so-called *ex-equations*).

[3] Note that unification $\wedge$ in Sub is a partial operation. We shall abuse the notation and denote by $\theta \wedge \sigma \in b$ the statment *if $\theta \wedge \sigma$ is defined then $\theta \wedge \sigma \in b$*.

5 Groundness: $\mathcal{D}ef$ and $\mathcal{P}os$

Many domains have been proposed in order to study groundness of logic programs. If Var is the (finite) set of variables of interest, the simplest one is $\mathcal{G} = \wp(Var)$, due to Jones and Søndergaard [21], where each $V \subseteq Var$ denotes the set of substitutions which ground every variable in V. The domain $\mathcal{G}$ can be equivalently defined as a domain of formulas by considering the set of formulas built from $Var \cup \{ \top \}$ by using the connective $\wedge$, quotiented with respect to (classic) logical equivalence (KL). Since each abstract domain must contain $\top$, in the following, for simplicity of notation, we shall identify Var with $Var \cup \{ \top \}$. If we denote by $Form(A, \circ)$ the set of formulas built from A by using the connective $\circ$, then

$$\mathcal{G} = Form(Var, \wedge)_{/KL}$$

where a formula $v_1 \wedge \ldots \wedge v_n$ represents the set $\{ v_1, \ldots, v_n \}$ and $\top$ stands for the empty set. This domain is certainly the most intuitive one and represents exactly the property we want to analyze, since it is built from the definition of groundness itself. Unfortunately, $\mathcal{G}$ is not very useful for groundness analysis, since it fails in capturing the groundness relations between different variables.

Other domains, based on (classic) propositional logic, have been proposed in order to enrich $\mathcal{G}$. The domain $\mathcal{D}ef$ [2] is built by considering all the classic propositional formulas whose models are closed under intersection (the so-called *definite* formulas). $\mathcal{D}ef$ has been also characterized as the set of formulas which are conjunctions of definite clauses, always quotiented with respect to (classic) logical equivalence.

$$\mathcal{D}ef = \{ \bigwedge_{i \in I} (\bigwedge_{j \in J} x_i^j \Rightarrow y^j) \mid \forall i \in I, j \in J \; x_i^j, y^j \in Var \}_{/KL}.$$

The most widely used domain for groundness analysis is the domain $\mathcal{P}os$ [23,5,2]. $\mathcal{P}os$ is able to characterize both pure groundness, i.e., whether a variable is instantiated to ground terms during the program execution, and the relations between the groundness of different program variables, providing in this sense a clear example of relational analysis. $\mathcal{P}os$ is the set of (classic) propositional formulas built from Var, by using the connectives $\wedge, \vee, \Rightarrow$. $\mathcal{P}os$ can be defined in different ways [5,2]:

$$\mathcal{P}os = Form(Var, \wedge, \vee, \Rightarrow)_{/KL} = Form(Var, \wedge, \Rightarrow)_{/KL}.$$

We can relate the three domains to the concrete domain $\wp^{\downarrow}(Sub)$ by using the same concretization function. The interpretation of the connectives is the classical one: $\phi \leq \psi$ if and only if ψ is a logical consequence of ϕ. We say that $I \subseteq Var$ is a model for ϕ, denoted by $I \models \phi$, if ϕ is true in the interpretation which assigns true to all the variables in I and $false$ to the

other variables [2]. For the sake of simplicity, we write θ `grounds` ϕ to denote $\{ x \in Var \mid vars(\theta(x)) = \emptyset \} \models \phi$, for $\theta \in Sub$. The concretization function is given by ([5]):

$$\gamma_{Sub}(\phi) = \{ \theta \in Sub \mid \forall \sigma \leq \theta \ \sigma \ \texttt{grounds} \ \phi \}.$$

As usual, we shall abuse the notation and call $\mathcal{G}$, $\mathcal{D}ef$ and $\mathcal{P}os$ the corresponding isomorphic images, subsets of $\wp^{\downarrow}(Sub)$: $\gamma_{Sub}(\mathcal{G})$, $\gamma_{Sub}(\mathcal{D}ef)$ and $\gamma_{Sub}(\mathcal{P}os)$.

6 Implicational groundness analysis

The first step toward the automatic construction of domains is to isolate the property to analyze and to look for the simplest domain which describe the property. In the case of groundness analysis this is clear: the basic groundness property says if a variable is ground or not. If we fix a finite set Var of variables of interest, big enough to analyze the program, the most abstract domain which describes the groundness of all variables in Var is by definition $\curlywedge(Var)$. It contains $\top$ and all possible conjunctions of variables in Var. This domain is exactly the domain $\mathcal{G} = \wp(Var)$ proposed by Jones and Søndergaard [21]. Therefore, we shall use $\mathcal{G}$ as the basis to reconstruct the other domains.

On the side of domain reconstruction for groundness analysis, a first example has been shown by Giacobazzi and Ranzato in [14]. They proved that the abstract domain $\mathcal{D}ef$ is the reduced cardinal power of $\mathcal{G}$, i.e., $\mathcal{D}ef$ can be seen as the space of monotone functions between elements in $\mathcal{G}$. The first result about implicational domains was proved in[17]. The authors, exploiting the previous characterization of $\mathcal{D}ef$, proved that $\mathcal{D}ef$ is precisely the Heyting completion of $\mathcal{G}$, i.e.,

$$\mathcal{D}ef = \mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}. \tag{4}$$

In this section we show that also $\mathcal{P}os$ can be obtained by using the Heyting completion operator, starting from the domain $\mathcal{G}$. In particular, we prove that $\mathcal{P}os = (\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) \xrightarrow{\curlywedge} \mathcal{G}$. Therefore by Eq. (4), $\mathcal{P}os = \mathcal{D}ef \xrightarrow{\curlywedge} \mathcal{G}$. In order to prove it, we need to better understand the relations among the operations of $\mathcal{P}os$ and the operations of $\wp^{\downarrow}(Sub)$. Since $\mathcal{P}os$ is a Moore family of $\wp^{\downarrow}(Sub)$, the meet operation ($\wedge$) on $\mathcal{P}os$ must be the restriction of the meet operation on $\wp^{\downarrow}(Sub)$, which is set intersection. It follows that for every family of formulas $\{ \phi_i \}_{i \in I} \subseteq \mathcal{P}os$ it clearly holds

$$\bigwedge_{i \in I} \gamma_{Sub}(\phi_i) = \gamma_{Sub}(\bigwedge_{i \in I} \phi_i). \tag{5}$$

As proved in [11], the join operation ($\vee$) of $\mathcal{P}os$ is not the restriction of the concrete one on $\wp^{\downarrow}(Sub)$, which is set union. Nevertheless, if we consider variables

only, the join operation on $\mathcal{P}os$ corresponds to the concrete join on $\wp^{\downarrow}(Sub)$.

Lemma 6 *Let* $\{ y_j \}_{j \in J} \subseteq Var$.

$$\bigvee_{j \in J} \gamma_{Sub}(y_j) = \gamma_{Sub}(\bigvee_{j \in J} y_j).$$

PROOF.
$$\begin{aligned}
\gamma_{Sub}(\textstyle\bigvee_{j \in J} y_j) &= \{ \theta \in Sub \mid \forall \sigma \leq \theta.\ \sigma \,\mathtt{grounds}\, \textstyle\bigvee_{j \in J} y_j \} \\
&= \{ \theta \in Sub \mid \theta \,\mathtt{grounds}\, \textstyle\bigvee_{j \in J} y_j \} \\
&= \{ \theta \in Sub \mid \exists j \in J.\ \theta \,\mathtt{grounds}\, y_j \} \\
&= \textstyle\bigvee_{j \in J} \{ \theta \in Sub \mid \theta \,\mathtt{grounds}\, y_j \} \\
&= \textstyle\bigvee_{j \in J} \{ \theta \in Sub \mid \forall \sigma \leq \theta.\ \sigma \,\mathtt{grounds}\, y_j \} \\
&= \textstyle\bigvee_{j \in J} \gamma_{Sub}(y_j). \qquad \Box
\end{aligned}$$

A similar result holds also for the implication. In general, the classic implication $\Rightarrow$ does not coincide with the intuitionistic one. If we consider implications between a conjunction of variables and a disjunction of variables only, then they are isomorphic.

Lemma 7 *Let* $X = \bigwedge_{i \in I} x_i$ *and* $Y = \bigvee_{j \in J} y_j$ *with* $x_i, y_j \in Var$.

$$\gamma_{Sub}(X \Rightarrow Y) = \gamma_{Sub}(X) \rightarrow \gamma_{Sub}(Y).$$

PROOF.
$$\begin{aligned}
&\gamma_{Sub}(X \Rightarrow Y) \\
&= \{ \theta \in Sub \mid \forall \sigma \leq \theta.\ \sigma \,\mathtt{grounds}\, X \Rightarrow Y \} \\
&= \{ \theta \in Sub \mid \forall \sigma \leq \theta.\ \sigma \,\mathtt{grounds}\, X \Rightarrow \sigma \,\mathtt{grounds}\, Y \} \\
&= \{ \theta \in Sub \mid \forall \sigma \leq \theta.\ (\forall i \in I\ \sigma \,\mathtt{grounds}\, x_i) \Rightarrow (\exists j \in J\ \sigma \,\mathtt{grounds}\, y_j) \} \\
&= \{ \theta \in Sub \mid \forall \sigma \leq \theta.\ (\forall i \in I\ \sigma \in \gamma_{Sub}(x_i)) \Rightarrow (\exists j \in J\ \sigma \in \gamma_{Sub}(y_j)) \} \\
&= \{ \theta \in Sub \mid \forall \sigma \leq \theta.\ \sigma \in \gamma_{Sub}(X) \Rightarrow \sigma \in \gamma_{Sub}(Y) \} \\
&= \gamma_{Sub}(X) \rightarrow \gamma_{Sub}(Y).
\end{aligned}$$

The last step holds because of the explicit representation for the element $a \rightarrow b$, i.e., $a \rightarrow b = \{ \theta \in Sub \mid \forall \sigma \leq \theta\ \sigma \in a \Rightarrow \sigma \in b \}$. $\Box$

The idea is to find a normal form for elements in $\mathcal{P}os$ where disjunction is allowed between variables only, and (classic) implication is allowed between a conjunction and a disjunction of variables only. It is well known that every (classic) formula can be put in conjunctive normal form, i.e., it can be written as conjunction of disjunctions. Moreover, every disjunction is a clause. Therefore we can write it as a unique implication, by putting all the negative variables on the left and the positive variables on the right.

Lemma 8 *Each formula $\phi \in \mathcal{P}os$ is equivalent to a formula*

$$\bigwedge_{k\in K}(\bigwedge_{i\in I} x_{i,k} \Rightarrow \bigvee_{j\in J} y_{j,k})$$

in $\mathcal{P}os$ for suitable $x_{i,k}, y_{j,k} \in Var$.

PROOF. Since $\mathcal{P}os$ is a Boolean algebra, every formula can be put in disjunctive normal form (as conjunction of disjunctions) where every disjunction contains both variables and negated variables, i.e., formulas of kind $x \Rightarrow \bot$. Therefore we have only to prove that every disjunction can be transformed into our normal form. By exploiting the identity: $(\bigvee_{j\in J} y_j) \vee \bigvee_{i\in I}(x_i \Rightarrow \bot) = (\bigwedge_{i\in I} x_i) \Rightarrow (\bigvee_{j\in J} y_j)$ we obtain the result. $\Box$

This result helps us to characterize the isomorphic image $\gamma_{Sub}(\mathcal{P}os)$ of $\mathcal{P}os$ in the concrete domain $\wp^{\downarrow}(Sub)$. The idea is to obtain the concretization of each abstract formula by simply replacing the abstract operations with the concrete ones. In some sense, this correspondence may seem counter-intuitive, since the concrete domain $\wp^{\downarrow}(Sub)$ is not a Boolean algebra, as the next example shows.

Example 9 *In a Boolean algebra $(a \wedge b) \Rightarrow c = (a \Rightarrow c) \vee (b \Rightarrow c)$ holds. Let $x, y, z \in Var$, $X = \gamma_{Sub}(x)$, $Y = \gamma_{Sub}(y)$ and $Z = \gamma_{Sub}(z)$. Consider in $\wp^{\downarrow}(Sub)$ the formula*

$$(X \wedge Y) \rightarrow Z = \{\, \theta \in Sub \mid \forall \delta \leq \theta \;\; \delta \in X \wedge Y \;\; \Rightarrow \delta \in Z \,\}.$$

We show that $(X \rightarrow Z) \vee (Y \rightarrow Z)$ is strictly contained in $(X \wedge Y) \rightarrow Z$. In fact, the substitution $\{\, z \backslash f(x,y) \,\}$ belongs to $(X \wedge Y) \rightarrow Z$ but not to $(X \rightarrow Z)$ nor to $(Y \rightarrow Z)$.

The idea is to transform each formula in $\mathcal{P}os$ in normal form and then to exploit the isomorphism given by Lemmata 6 and 7, as shown in the next example.

Example 10 *Let $x \vee (x \Leftrightarrow y)$ be a formula in $\mathcal{P}os$. We look for its concretization $\gamma_{Sub}(x \vee (x \Leftrightarrow y))$. First we transform $x \vee (x \Leftrightarrow y)$ in normal form, which*

always exists by Lemma 8.

$$\begin{aligned} & x \vee (x \Leftrightarrow y) = x \vee ((x \Rightarrow y) \wedge (y \Rightarrow x)) = \\ = & (x \vee (x \Rightarrow y)) \wedge (x \vee (y \Rightarrow x)) = (x \vee \neg x \vee y) \wedge (x \vee \neg y \vee x) = \\ = & x \vee \neg y \vee x = y \Rightarrow x \end{aligned}$$

By using Lemma 7, we transform $y \Rightarrow x$ *and obtain* $\gamma_{Sub}(y \Rightarrow x) = \gamma_{Sub}(y) \rightarrow \gamma_{Sub}(x)$*, which is a formula in* $\wp^{\downarrow}(Sub)$*.*

Therefore, we have a constructive method to transform formulas in $\mathcal{P}os$ into formulas in $\wp^{\downarrow}(Sub)$ by simply replacing variables with their concretizations and abstract operations with the corresponding concrete ones. Hence, for a generic element in $\mathcal{P}os$ $\phi = \bigwedge_{k\in K}(\bigwedge_{i\in I} x_{i,k} \Rightarrow \bigvee_{j\in J} y_{j,k})$, the concretization $\gamma_{Sub}(\phi)$ of ϕ is precisely $\bigwedge_{k\in K}(\bigwedge_{i\in I} \gamma_{Sub}(x_{i,k}) \rightarrow \bigvee_{j\in J} \gamma_{Sub}(y_{j,k}))$, as proved by the following theorem.

Theorem 11

$$\gamma_{Sub}(\mathcal{P}os) = \{ \bigwedge_{k\in K} (\bigwedge_{i\in I} X_{i,k} \rightarrow \bigvee_{j\in J} Y_{j,k}) \mid X_{i,k}, Y_{j,k} \in \gamma_{Sub}(Var) \}\,[4].$$

PROOF. By Lemma 8, we know that

$$\mathcal{P}os = \{ \bigwedge_{k\in K} (\bigwedge_{i\in I} x_{i,k} \Rightarrow \bigvee_{j\in J} y_{j,k}) \mid x_{i,k}, y_{j,k} \in Var \}.$$

$$\begin{aligned} & \gamma_{Sub}(\textstyle\bigwedge_{k\in K}(\bigwedge_{i\in I} x_{i,k} \Rightarrow \bigvee_{j\in J} y_{j,k})) && \text{[by Eq. (5)]} \\ = & \textstyle\bigwedge_{k\in K} \gamma_{Sub}(\bigwedge_{i\in I} x_{i,k} \Rightarrow \bigvee_{j\in J} y_{j,k}) && \text{[by Lemma 7]} \\ = & \textstyle\bigwedge_{k\in K}(\gamma_{Sub}(\bigwedge_{i\in I} x_{i,k}) \rightarrow \gamma_{Sub}(\bigvee_{j\in J} y_{j,k})) && \text{[by Eq. (5) and Lemma 6]} \\ = & \textstyle\bigwedge_{k\in K}(\bigwedge_{i\in I} \gamma_{Sub}(x_{i,k}) \rightarrow \bigvee_{j\in J} \gamma_{Sub}(y_{j,k})). \end{aligned}$$

Hence, we have that

$$\begin{aligned} \gamma_{Sub}(\mathcal{P}os) &= \{ \textstyle\bigwedge_{k\in K}(\bigwedge_{i\in I} \gamma_{Sub}(x_{i,k}) \rightarrow \bigvee_{j\in J} \gamma_{Sub}(y_{j,k})) \mid x_{i,k}, y_{j,k} \in Var \} = \\ &= \{ \textstyle\bigwedge_{k\in K}(\bigwedge_{i\in I} X_{i,k} \rightarrow \bigvee_{j\in J} Y_{j,k}) \mid X_{i,k}, Y_{j,k} \in \gamma_{Sub}(Var) \}. \qquad \square \end{aligned}$$

Since $\mathcal{P}os$ is isomorphic to $\gamma_{Sub}(\mathcal{P}os)$ and Theorem 11 allows us to describe concrete objects in $\gamma_{Sub}(\mathcal{P}os)$ as formulas, from now on, we identify $\mathcal{P}os$ with its concretization $\gamma_{Sub}(\mathcal{P}os)$.

[4] Note that $X_{i,k}, Y_{j,k}$ denote sets of substitutions.

Our aim is to describe the domain $\mathcal{P}os$ by using the Heyting completion refinement only. Therefore, we look for a normal form for concrete objects which involves meet and intuitionistic implication only and it does not contain any disjunction. Note that, in the concrete formulas of Theorem 11, disjunctions are allowed between variables only. The last step is then to find a representation in $\wp^{\downarrow}(Sub)$ for unions of variables in terms of intuitionistic implications. First of all, we prove that every concrete double implication can be rearranged as a disjunction and a (single) implication.

Lemma 12 *Let* $X = \bigwedge_{i\in I} x_i$, $Y = \bigvee_{j\in J} y_j$ *and* $Z = \bigvee_{k\in K} z_k$, *where* $x_i, y_j, z_k \in \gamma_{Sub}(Var)$.

$$(X \to Y) \to Z = (X \vee Z) \wedge (Y \to Z).$$

PROOF. Since $\wp^{\downarrow}(Sub)$ is a cHa, the following inclusions trivially hold.

- $(X \vee Z) \wedge (Y \to Z) \subseteq (X \to Y) \to Z$
- $(X \to Y) \to Z \subseteq Y \to Z$.

We have to prove that $(X \to Y) \to Z \subseteq X \vee Z$. Let $\theta \in (X \to Y) \to Z$ and assume, for contradiction, that $\theta \notin X$ and $\theta \notin Z$. Therefore $\theta \notin Y$ (otherwise, from $\theta \in Y \to Z$ it would follow $\theta \in Z$).

Let $V = \bigcup_{i\in I} vars(\theta(x_i)) \cup \bigcup_{j\in J} vars(\theta(y_j))$, where, by abusing the notation, we identify a concrete object $x \in \gamma_{Sub}(Var)$ with the corresponding variable in Var. Let n be a new variable, not in V. Consider the substitution δ so defined: $\forall u \in Var$

$$\delta(u) = \begin{cases} n & \text{if } u \in V \backslash dom(\theta) \\ u & \text{otherwise.} \end{cases}$$

Note that $dom(\delta) \cap dom(\theta) = \emptyset$. Therefore the substitution $\theta \circ \delta$ is well defined.

- $\forall u \in Term$ $\delta(u)$ is ground iff u is ground, since δ transforms variables in variables without affecting the groundness.
- $\forall v \in V \backslash dom(\theta)$, $vars(\delta(v)) = \{ n \}$. By definition of δ.
- $\forall i \in I$, it holds:
  - if $vars(\theta(x_i)) = \emptyset$ then $vars(\theta \circ \delta(x_i)) = \emptyset$
  - if $vars(\theta(x_i)) \neq \emptyset$ then $vars(\theta \circ \delta(x_i)) = \{ n \}$
  - since $\theta \notin \bigwedge_{i\in I} x_i$, there exists $i \in I$ such that $vars(\theta(x_i)) \neq \emptyset$, and thus $vars(\theta \circ \delta(x_i)) = \{ n \}$.

  Therefore, $\bigcup_{i\in I} vars(\theta \circ \delta(x_i)) = \{ n \}$.
- since $\theta \notin \bigvee_{j\in J} y_j$, $\forall j \in J$ $vars(\theta(y_j)) \neq \emptyset$ and thus $vars(\theta \circ \delta(y_j)) = \{ n \}$.

We prove now that $\theta \circ \delta \in \bigwedge_{i\in I} x_i \to \bigvee_{j\in J} y_j$.

$$
\begin{array}{ll}
 & \theta \circ \delta \in \bigwedge_{i\in I} x_i \to \bigvee_{j\in J} y_j \\
\Leftrightarrow \forall \sigma \leq \theta \circ \delta & \sigma \in \bigwedge_{i\in I} x_i \Rightarrow \sigma \in \bigvee_{j\in J} y_j \\
\Leftrightarrow \forall \sigma \leq \theta \circ \delta & \theta \circ \delta \circ \sigma \in \bigwedge_{i\in I} x_i \Rightarrow \theta \circ \delta \circ \sigma \in \bigvee_{j\in J} y_j \qquad [\theta \circ \delta \circ \sigma = \sigma] \\
\Leftrightarrow \forall \sigma \leq \theta \circ \delta & \forall i \in I\ \theta \circ \delta \circ \sigma \in x_i \Rightarrow \exists j \in J\ \theta \circ \delta \circ \sigma \in y_j \\
\Leftrightarrow \forall \sigma \leq \theta \circ \delta & \forall i \in I\ vars(\theta \circ \delta \circ \sigma(x_i)) = \emptyset \Rightarrow \\
 & \exists j \in J\ vars(\theta \circ \delta \circ \sigma(y_j)) = \emptyset \\
\Leftrightarrow \forall \sigma \leq \theta \circ \delta & (\forall i \in I\ \forall v \in vars(\theta \circ \delta(x_i))\ vars(\sigma(v)) = \emptyset) \Rightarrow \\
 & \exists j \in J\ \forall v \in vars(\theta \circ \delta(y_j))\ vars(\sigma(v)) = \emptyset \\
\Leftrightarrow \forall \sigma \leq \theta \circ \delta & (\forall v \in \cup_{i\in I} vars(\theta \circ \delta(x_i))\ vars(\sigma(v)) = \emptyset) \Rightarrow \\
 & \exists j \in J\ \forall v \in vars(\theta \circ \delta(y_j))\ vars(\sigma(v)) = \emptyset \\
\Leftrightarrow \forall \sigma \leq \theta \circ \delta & \forall v \in \{\, n \,\} vars(\sigma(v)) = \emptyset \Rightarrow \\
 & \exists j \in J\ \forall v \in \{\, n \,\}\ vars(\sigma(v)) = \emptyset \qquad [vars(\theta \circ \delta(y_j)) = \{\, n \,\}] \\
\Leftrightarrow \forall \sigma \leq \theta \circ \delta & vars(\sigma(n)) = \emptyset \Rightarrow vars(\sigma(n)) = \emptyset
\end{array}
$$

Therefore $\theta \circ \delta \in \bigwedge_{i\in I} x_i \to \bigvee_{j\in J} y_j$. Moreover $\theta \circ \delta \notin Z$ since $\theta \notin Z$ by hypothesis. Thus $\theta \circ \delta \notin (X \to Y) \to Z$, and therefore $\theta \notin (X \to Y) \to Z$, which is a contradiction. It follows that either $\theta \in X$ or $\theta \in Z$, i.e., $\theta \in X \vee Z$. $\Box$

The previous lemma can be generalized to deal with conjunctions of implications. Such a result helps us to decompose complex formulas with disjunctions and implications into double implications and therefore to derive a transformation from formulas in $\mathcal{P}os$ into formulas in $\wp^{\downarrow}(Sub)$ which allows $\wedge$ and $\to$ only. In particular, note that implication is used with at most two levels of nesting. This suggests to construct $\mathcal{P}os$ by applying twice the Heyting completion operator.

Lemma 13 *Let $x_{i,k}, y_{j,k}, z_k \in \gamma_{Sub}(Var)$, $X_k = \bigwedge_{i\in I} x_{i,k}$, $Y_k = \bigvee_{j\in J} y_{j,k}$ and $Z = \bigvee_{k=1..n} z_k$. It holds:*

$$
(\bigwedge_{k=1}^{n} (X_k \to Y_k)) \to Z =
$$
$$
(\bigvee_{k=1}^{n} X_k \vee Z) \wedge ((\bigwedge_{k=1}^{n} Y_k) \to Z) \wedge \bigwedge_{G \subset \{\, 1..n \,\}} ((\bigwedge_{k\in G} Y_k) \to (\bigvee_{k\in \overline{G}} X_k \vee Z))
$$

where $\overline{G} = \{\, 1..n \,\} \backslash G$.

PROOF. The proof is by induction on n. For $n = 1$ the thesis boils down to prove $(X \to Y) \to Z = (X \vee Z) \wedge (Y \to Z)$, which is the result in Lemma 12. Let assume it for $n-1$ and prove for n.

$$(\bigwedge_{k=1}^{n}(X_k \to Y_k)) \to Z = (X_n \to Y_n) \to (\bigwedge_{k=1}^{n-1}(X_k \to Y_k)) \to Z \text{ [by Ind. Hypothesis]}$$

$$=(X_n \to Y_n) \to \{(\bigvee_{k=1}^{n-1} X_k \vee Z) \wedge ((\bigwedge_{k=1}^{n-1} Y_k) \to Z) \wedge \bigwedge_{G \subset \{\,1..n-1\,\}} ((\bigwedge_{k\in G} Y_k) \to (\bigvee_{k\in \overline{G}} X_k \vee Z))\}$$

$$=((X_n \to Y_n) \to (\bigvee_{k=1}^{n-1} X_k \vee Z)) \wedge ((X_n \to Y_n) \to ((\bigwedge_{k=1}^{n-1} Y_k) \to Z)) \wedge ((X_n \to Y_n) \to \bigwedge_{G \subset \{\,1..n-1\,\}} ((\bigwedge_{k\in G} Y_k) \to (\bigvee_{k\in \overline{G}} X_k \vee Z))) \quad \text{[by Lemma 12]}$$

$$=((X_n \vee \bigvee_{k=1}^{n-1} X_k \vee Z) \wedge (Y_n \to (\bigvee_{k=1}^{n-1} X_k \vee Z))) \wedge ((\bigwedge_{k=1}^{n-1} Y_k) \to ((X_n \vee Z) \wedge (Y_n \to Z))) \wedge \bigwedge_{G \subset \{\,1..n-1\,\}} ((X_n \to Y_n) \to ((\bigwedge_{k\in G} Y_k) \to (\bigvee_{k\in \overline{G}} X_k \vee Z)))$$

$$=(\bigvee_{k=1}^{n} X_k \vee Z) \wedge (Y_n \to (\bigvee_{k=1}^{n-1} X_k \vee Z)) \wedge ((\bigwedge_{k=1}^{n-1} Y_k) \to (X_n \vee Z)) \wedge (\bigwedge_{k=1}^{n-1} Y_k \to (Y_n \to Z)) \wedge \bigwedge_{G \subset \{\,1..n-1\,\}} ((\bigwedge_{k\in G} Y_k) \to ((X_n \to Y_n) \to (\bigvee_{k\in \overline{G}} X_k \vee Z)))$$

$$=(\bigvee_{k=1}^{n} X_k \vee Z) \wedge (Y_n \to (\bigvee_{k=1}^{n-1} X_k \vee Z)) \wedge ((\bigwedge_{k=1}^{n-1} Y_k) \to (X_n \vee Z)) \wedge ((\bigwedge_{k=1}^{n} Y_k) \to Z) \wedge \bigwedge_{G \subset \{\,1..n-1\,\}} ((\bigwedge_{k\in G} Y_k) \to ((X_n \vee \bigvee_{k\in \overline{G}} X_k \vee Z) \wedge (Y_n \to (\bigvee_{k\in \overline{G}} X_k \vee Z))))$$

$$=(\bigvee_{k=1}^{n} X_k \vee Z) \wedge (Y_n \to (\bigvee_{k=1}^{n-1} X_k \vee Z)) \wedge (\bigwedge_{k=1}^{n-1} Y_k \to (X_n \vee Z)) \wedge ((\bigwedge_{k=1}^{n} Y_k) \to Z) \wedge \bigwedge_{G \subset \{\,1..n-1\,\}} ((\bigwedge_{k\in G} Y_k \to (X_n \vee \bigvee_{k\in \overline{G}} X_k \vee Z)) \wedge (\bigwedge_{k\in G} Y_k \to (Y_n \to (\bigvee_{k\in \overline{G}} X_k \vee Z))))$$

$$=(\bigvee_{k=1}^{n} X_k \vee Z) \wedge (Y_n \to (\bigvee_{k=1}^{n-1} X_k \vee Z)) \wedge ((\bigwedge_{k=1}^{n-1} Y_k) \to (X_n \vee Z)) \wedge ((\bigwedge_{k=1}^{n} Y_k) \to Z) \wedge \bigwedge_{G \subset \{\,1..n-1\,\}} (((\bigwedge_{k\in G} Y_k) \to (\bigvee_{k\in \overline{G}\cup\{\,n\,\}} X_k \vee Z)) \wedge ((\bigwedge_{k\in G\cup\{\,n\,\}} Y_k) \to (\bigvee_{k\in \overline{G}} X_k \vee Z)))$$

$$=(\bigvee_{k=1}^{n} X_k \vee Z) \wedge ((\bigwedge_{k=1}^{n} Y_k) \to Z) \wedge \bigwedge_{G \subset \{\,1..n\,\}} ((\bigwedge_{k\in G} Y_k) \to (\bigvee_{k\in \overline{G}} X_k \vee Z)). \qquad \square$$

We first exploit this result to prove that $\mathcal{P}os$ is closed for the Heyting completion operator.

Theorem 14 *$\mathcal{P}os = \mathcal{P}os \xrightarrow{\curlywedge} \mathcal{P}os$.*

PROOF. Since the Heyting completion operator is reductive on the second argument, it holds that $\mathcal{P}os \xrightarrow{\curlywedge} \mathcal{P}os \sqsubseteq \mathcal{P}os$. For the other direction, by Theorem 11, every element in $\mathcal{P}os$ is of the form $\bigwedge_{k\in K}(X_k \to Y_k)$ where $X_k = \bigwedge_{i\in I} x_{i,k}$ and $Y_k = \bigvee_{j\in J} y_{j,k}$. By construction, any element in $\mathcal{P}os \xrightarrow{\curlywedge} \mathcal{P}os$ is of the form

$$\bigwedge_{m\in M} ((\bigwedge_{k\in K}(X_k^m \to Y_k^m)) \to (\bigwedge_{a\in A}(W_a^m \to Z_a^m)))$$

where $W_a^m = \bigwedge_{b\in B} w_{b,a}^m$ and $Z_a^m = \bigvee_{c\in C} y_{c,a}^m$. $\mathcal{P}os$ being a Moore family, it suffices to prove that for each $m \in M$, every object $(\bigwedge_{k\in K}(X_k^m \to Y_k^m)) \to (\bigwedge_{a\in A}(W_a^m \to Z_a^m))$ is in $\mathcal{P}os$. Moreover:

$$\begin{aligned}
&(\bigwedge_{k\in K}(X_k^m \to Y_k^m)) \to (\bigwedge_{a\in A}(W_a^m \to Z_a^m)) \\
&= \bigwedge_{a\in A}((\bigwedge_{k\in K}(X_k^m \to Y_k^m)) \to (W_a^m \to Z_a^m)) \\
&= \bigwedge_{a\in A}(\bigwedge_{k\in K}(X_k^m \to Y_k^m) \wedge W_a^m) \to Z_a^m.
\end{aligned}$$

Again, it suffices to prove that for each $a \in A$, every element of the form $(\bigwedge_{k\in K}(X_k^m \to Y_k^m) \wedge W_a^m) \to Z_a^m$ is in $\mathcal{P}os$. Moreover by writing $W_a^m = \bigwedge_{b\in B} w_{b,a}^m = \bigwedge_{b\in B}(\top \to w_{b,a}^m)$, we only need to prove that every formula of the form $\bigwedge_{k\in K}(X_k \to Y_k) \to Z$ is in $\mathcal{P}os$.

By Lemma 13, it holds:

$$\begin{aligned}
&\bigwedge_{k\in K}(X_k \to Y_k) \to Z = \\
&\qquad = (\bigvee_{k\in K} X_k \vee Z) \wedge ((\bigwedge_{k\in K} Y_k) \to Z) \wedge \bigwedge_{G\subset K} ((\bigwedge_{k\in G} Y_k) \to (\bigvee_{k\in \overline{G}} X_k \vee Z)).
\end{aligned}$$

Note that, by distributivity of $\wp^{\downarrow}(Sub)$ and Theorem 11, all formulas $\bigvee_{k\in K} X_k \vee Z$, $(\bigwedge_{k\in K} Y_k) \to Z$ and $(\bigwedge_{k\in G} Y_k) \to (\bigvee_{k\in \overline{G}} X_k \vee Z)$ belong to $\mathcal{P}os$. Therefore, the conjunction also belongs to $\mathcal{P}os$. $\square$

We are now in the position to prove the main theorem of this section, that is $\mathcal{P}os$ is the double Heyting completion of $\mathcal{G}$.

Theorem 15 *$\mathcal{P}os = (\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) \xrightarrow{\curlywedge} \mathcal{G}$.*

PROOF. By Theorem 14, $\mathcal{P}os = (\mathcal{P}os \xrightarrow{\curlywedge} \mathcal{P}os) \xrightarrow{\curlywedge} \mathcal{P}os \sqsubseteq (\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) \xrightarrow{\curlywedge} \mathcal{G}$.

For the other direction, by Theorem 11, every element in $\mathcal{P}os$ is a conjunction of formulas of the form $\bigwedge_{i\in I} x_i \to \bigvee_{j\in J} y_j$, where $x_i, y_j \in \gamma_{Sub}(Var)$. By Lemma 13, it holds:

$$
\begin{aligned}
& (\textstyle\bigwedge_{j\in J}(y_j \to \bot)) \to \bot \\
= {} & (\textstyle\bigwedge_{j\in J}(y_j \to \bigwedge_{v\in Var} v)) \to \bigwedge_{u\in Var} u) \\
= {} & \textstyle\bigwedge_{u\in Var}(\bigwedge_{j\in J} \bigwedge_{v\in Var}(y_j \to v)) \to u) \qquad \text{[by Lemma 13]} \\
= {} & \textstyle\bigwedge_{u\in Var}((\bigvee_{j\in J} y_j \vee u) \wedge (\bigwedge_{v\in Var} v \to \bigwedge_{u\in Var} u) \wedge \\
& \textstyle\bigwedge(\bigwedge_{v\in Var} v \to (\bigvee_{j\in J} y_j \vee u))) \\
= {} & \textstyle\bigwedge_{u\in Var}((\bigvee_{j\in J} y_j \vee u) \wedge (\bot \to \bot) \wedge \bigwedge(\bot \to (\bigvee_{j\in J} y_j \vee u))) \\
= {} & \textstyle\bigwedge_{u\in Var}(\bigvee_{j\in J} y_j \vee u) \\
= {} & \textstyle\bigvee_{j\in J} y_j \vee \bigwedge_{u\in Var} u \\
= {} & \textstyle\bigvee_{j\in J} y_j \vee \bot = \bigvee_{j\in J} y_j.
\end{aligned}
$$

It follows that

$$
\begin{aligned}
& \textstyle\bigwedge_{i\in I} x_i \to \bigvee_{j\in J} y_j \\
= {} & \textstyle\bigwedge_{i\in I} x_i \to ((\bigwedge_{j\in J}(y_j \to \bot)) \to \bot) \\
= {} & \textstyle(\bigwedge_{i\in I} x_i \wedge \bigwedge_{j\in J}(y_j \to \bot)) \to \bot.
\end{aligned}
$$

By definition of Heyting completion, $\bigwedge_{i\in I} x_i \wedge \bigwedge_{j\in J}(y_j \to \bot) \in \mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}$ and therefore $(\bigwedge_{i\in I} x_i \wedge (\bigwedge_{j\in J}(y_j \to \bot))) \to \bot \in (\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) \xrightarrow{\curlywedge} \mathcal{G}$. Thus $\mathcal{P}os \subseteq (\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) \xrightarrow{\curlywedge} \mathcal{G}$, which concludes the proof. □

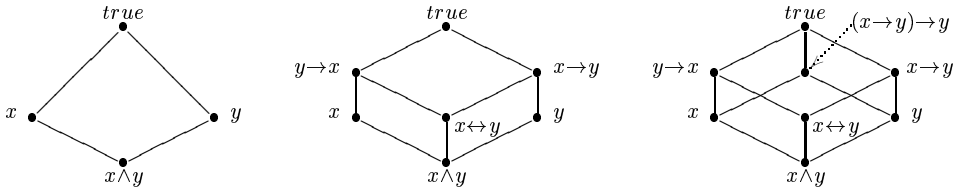
By Eq. (4), we can immediately derive the relations between the domains $\mathcal{P}os$ and $\mathcal{D}ef$.

Corollary 16 $\mathcal{P}os = \mathcal{D}ef \xrightarrow{\curlywedge} \mathcal{G} = \mathcal{D}ef \xrightarrow{\curlywedge} \mathcal{D}ef$.

PROOF. By Eq. (4), we obtain $\mathcal{P}os = \mathcal{D}ef \xrightarrow{\curlywedge} \mathcal{G}$. For the other equation:

$$
\begin{aligned}
& \mathcal{D}ef \xrightarrow{\curlywedge} \mathcal{D}ef && \text{[by Eq. (4)]} \\
= {} & (\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) \xrightarrow{\curlywedge} (\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) && \text{[by (2) in Prop. 3]} \\
= {} & ((\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) \sqcap \mathcal{G}) \xrightarrow{\curlywedge} \mathcal{G} && \text{[by monotonicity]} \\
= {} & (\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) \xrightarrow{\curlywedge} \mathcal{G}. && \square
\end{aligned}
$$

For $Var = \{ x, y \}$, the domains $\mathcal{G}$, $\mathcal{D}ef$ and $\mathcal{P}os$ are depicted below.



$\mathcal{G}$ $\qquad$ $\mathcal{G} \xrightarrow{\wedge} \mathcal{G}$ $\qquad$ $(\mathcal{G} \xrightarrow{\wedge} \mathcal{G}) \xrightarrow{\wedge} \mathcal{G}$

The first important consequence of Theorem 15 is that the domain $\mathcal{P}os$ is constructed by using the domain $\mathcal{G}$ and the logical properties of the concrete domain only. We do not need to "invent" the domain, to prove that it is actually an abstraction of $\wp^{\downarrow}(Sub)$, nor to prove that it refines $\mathcal{G}$, since all these properties hold by construction. In our framework, $\mathcal{P}os$ arises as the natural refinement of $\mathcal{G}$ and $\mathcal{D}ef$. Moreover, we get a normal form for elements in $\gamma_{Sub}(\mathcal{P}os)$. This result allows us to see $\mathcal{P}os$ into $\wp^{\downarrow}(Sub)$ and to directly deal with $\gamma_{Sub}(\mathcal{P}os)$. In fact, when we use a formula in $\mathcal{P}os$, actually we use an equivalence class with respect to (classic) logic equivalence (KL). In our formalization, we do not need to use equivalence classes any longer. Moreover, the normal form of concrete formulas is indeed very natural and preserves the intuitive meaning of abstract formulas.

7 Optimality of groundness analysis

7.1 Which optimality for $\mathcal{P}os$?

Usually, in abstract domains, some points are used to represent the result of the analysis, while other ones are used during the computation only. In $\mathcal{P}os$, the objects which represent the final result of groundness analysis are those in $\mathcal{G}$ only, since the other ones do not provide basic groundness information. From Eq.'s (1), (2), and (3), we note that only axioms (1) and (2) produce a result which belongs to $\mathcal{G}$ (i.e., formulas with conjunctions only). This suggests that an abstract domain, to be optimal, should contain all and only those formulas which have an implicational form, since implications only can be reduced to formulas in $\mathcal{G}$. Therefore, to obtain an optimal domain, we have to add in the abstract domain all and only the implications. This concept is precisely captured by the notion of implicational domain equation.

An abstract domain X is *closed for A with respect to Heyting completion* if it

is the most abstract solution of the implicational domain equation

$$X = A \sqcap (X \xrightarrow{\curlywedge} X)$$

where A is a given abstract domain. The solution, which always exists, is the most abstract domain X which is more concrete than A and is closed for Heyting completion. Moreover, it turns out to be the most refined domain, built from A, which we can obtain by using the Heyting completion refinement.

7.2 $\mathcal{P}os$ is closed for $\mathcal{G}$

The question which naturally arises is: Which is the most abstract solution to the equation

$$X = \mathcal{G} \sqcap (X \xrightarrow{\curlywedge} X) \tag{6}$$

or, in other words, which is the most abstract domain which is more precise than $\mathcal{G}$ and closed with respect to Heyting completion? In the previous section we have shown that $\mathcal{P}os = (\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) \xrightarrow{\curlywedge} \mathcal{G}$ and $\mathcal{P}os \xrightarrow{\curlywedge} \mathcal{P}os = \mathcal{P}os$. It easily follows that $\mathcal{P}os \xrightarrow{\curlywedge} \mathcal{G} = \mathcal{P}os$, and therefore $\mathcal{P}os$ is the most abstract solution of Eq. (6), as shown by the next theorem.

Theorem 17 *$\mathcal{P}os$ is the most abstract solution to the equation $X = \mathcal{G} \sqcap (X \xrightarrow{\curlywedge} X)$.*

PROOF. By Theorem 15, $\mathcal{P}os = (\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) \xrightarrow{\curlywedge} \mathcal{G}$ and therefore, by (2) in Prop. 3, we have that $\mathcal{P}os \xrightarrow{\curlywedge} \mathcal{P}os = \mathcal{P}os \xrightarrow{\curlywedge} \mathcal{G}$. Moreover, by Theorem 14, $\mathcal{P}os = \mathcal{P}os \xrightarrow{\curlywedge} \mathcal{P}os$, and thus $\mathcal{P}os \xrightarrow{\curlywedge} \mathcal{G} = \mathcal{P}os$. Note that, by Theorem 15, $\mathcal{P}os$ is clearly the most abstract domain with this property, and therefore the most abstract solution of Eq. (6). □

The theorem states that we can not further refine $\mathcal{P}os$ by using the Heyting completion refinement and therefore it yields a representation result for elements in $\mathcal{P}os$. It precisely states that an element in $\wp^{\downarrow}(Sub)$ belongs to $\mathcal{P}os$ if and only if it can be written by using (concrete) meets and implications only, starting from objects in $\mathcal{G}$. Hence, it completely characterizes the concrete image of $\mathcal{P}os$, not only the abstract objects. It is worth noting that, differently from previous characterizations of $\mathcal{P}os$ ([2,5]), all these results hold on the concrete domain. In addition, our construction provides a characterization of the abstract disjunction: a disjunctive formula belongs to $\mathcal{P}os$ if and only if it can be written by using $\wedge$ and $\rightarrow$ only. It would be impossible to state a similar result with classic propositional formulas, since it is well known that each formula can be written by means of $\wedge$ and $\Leftrightarrow$ only.

This result allows us to answer the question "why $\mathcal{P}os$ is considered optimal without being disjunctive", also from an intuitive point of view. We know that a disjunctive formula belongs to $\mathcal{P}os$ if and only if it can be put in implicational form. From a logic point of view, the elements useful for the analysis are implications only (as they can be reduced by *modus ponens*). Since a good domain should contain all and only those joins which are indeed useful, we would only include those joins that can be reduced, which, in turn, are exactly the joins that $\mathcal{P}os$ contains. This explains the result in [12] that $\mathcal{P}os$ is complete with respect to $\curlyvee(\mathcal{P}os)$, i.e., the domains $\mathcal{P}os$ and $\curlyvee(\mathcal{P}os)$ have the same precision with respect to groundness analysis.

7.3 On disjunctive completion of $\mathcal{P}os$

We proved that we can not further refine $\mathcal{P}os$ by Heyting completion and [12] showed that it is superfluous to use $\curlyvee(\mathcal{P}os)$ instead of $\mathcal{P}os$. We may try to refine $\mathcal{P}os$ by disjunctive completion and then to refine it by Heyting completion. We now show that, even combining disjunctive completion and Heyting completion, we still obtain the domain $\curlyvee(\mathcal{P}os)$, when starting from $\mathcal{G}$. Formally, we look for the solution of the equation

$$X = \curlyvee(\mathcal{P}os) \sqcap (X \stackrel{\curlywedge}{\longrightarrow} X).$$

We show that also $\curlyvee(\mathcal{P}os)$ is closed with respect to Heyting completion. Before proving this result, we need a technical lemma to precisely describe the objects in $\mathcal{P}os$.

Lemma 18 *Let $x_i, y_j \in \gamma_{Sub}(Var)$, for $i \in I$ and $j \in J$.*

$$\theta \in \bigwedge_{i\in I} x_i \rightarrow \bigvee_{j\in J} y_j \Leftrightarrow \exists j \in J \; vars(\theta(y_j)) \subseteq \cup_{i\in I} vars(\theta(x_i)).$$

PROOF. First note that, by definition, it holds:

$$\begin{aligned}
&\theta \in \textstyle\bigwedge_{i\in I} x_i \rightarrow \bigvee_{j\in J} y_j \\
\Leftrightarrow\; &\forall \delta \leq \theta \; \delta \in \textstyle\bigwedge_{i\in I} x_i \Rightarrow \delta \in \bigvee_{j\in J} y_j \\
\Leftrightarrow\; &\forall \delta \leq \theta \;\; \cup_{i\in I} vars(\delta(x_i)) = \emptyset \Rightarrow \exists j \in J \; vars(\delta(y_j)) = \emptyset.
\end{aligned}$$

For contradiction, assume that $\forall j \in J$ it holds $vars(\theta(y_j)) \not\subseteq \cup_{i\in I} vars(\theta(x_i))$. It follows that:

$$\begin{aligned}
&\forall j \in J \; vars(\theta(y_j)) \setminus \cup_{i\in I} vars(\theta(x_i)) \neq \emptyset \\
\Rightarrow\; &\forall j \in J \; \exists w_j \in vars(\theta(y_j)) \setminus \cup_{i\in I} vars(\theta(x_i)).
\end{aligned}$$

Let $\delta_1 = \theta \circ \{\, u\backslash a \mid u \in \cup_{i\in I} vars(\theta(x_i)) \,\}$ where a is a constant of the language. Then, $\forall i \in I$ it holds $\delta_1 \in x_i$. In fact:

$$\begin{aligned}\delta_1(x_i) &= \theta \circ \{\, u\backslash a \mid u \in \cup_{i\in I} vars(\theta(x_i)) \,\}(x_i)\\ &= \{\, u\backslash a \mid u \in \cup_{i\in I} vars(\theta(x_i)) \,\}(\theta(x_i))\end{aligned}$$

Therefore, $vars(\delta_1(x_i)) = \emptyset$ and $\delta_1 \in \bigwedge_{i\in I} x_i$. But $\forall j \in J$ $\delta_1 \notin y_j$ since $vars(\delta_1(y_j)) = vars(\{\, u\backslash a \mid u \in \cup_{i\in I} vars(\theta(x_i)) \,\}(\theta(y_j)))$ and there always exists $w_j \in vars(\theta(y_j))\backslash \cup_{i\in I} vars(\theta(x_i))$.

The other direction follows from the fact that if $vars(\theta(y_j)) \subseteq \cup_{i\in I} vars(\theta(x_i))$ then $\forall \delta \leq \theta$ $vars(\delta(y_j)) \subseteq \cup_{i\in I} vars(\delta(x_i))$. $\Box$

Theorem 19 $\curlyvee(\mathcal{P}os) = \curlyvee(\mathcal{P}os) \xrightarrow{\curlywedge} \curlyvee(\mathcal{P}os)$.

PROOF. By (3) in Prop. 3, $\curlyvee(\mathcal{P}os) \xrightarrow{\curlywedge} \curlyvee(\mathcal{P}os) = \mathcal{P}os \xrightarrow{\curlywedge} \curlyvee(\mathcal{P}os)$. Moreover, by Proposition 2, each object in $\curlyvee(\mathcal{P}os)$ is a disjunction of elements in $\mathcal{P}os$. Therefore, it suffices to prove that for any $x_i, y_j, w_m^k, z_n^k \in \gamma_{Sub}(Var)$ it holds that

$$\begin{aligned}(\bigwedge_{i\in I} x_i \to \bigvee_{j\in J} y_j) \to \bigvee_{k\in K} (\bigwedge_{m\in M} w_m^k \to \bigvee_{n\in N} z_n^k) = \\ \bigvee_{k\in K} ((\bigwedge_{i\in I} x_i \to \bigvee_{j\in J} y_j) \to (\bigwedge_{m\in M} w_m^k \to \bigvee_{n\in N} z_n^k)).\end{aligned}$$

By Lemma 18 we have that

$$\begin{aligned}&\theta \in (\textstyle\bigwedge_{i\in I} x_i \to \bigvee_{j\in J} y_j) \to \bigvee_{k\in K} (\bigwedge_{m\in M} w_m^k \to \bigvee_{n\in N} z_n^k)\\ &\Leftrightarrow \forall \delta \leq \theta\ (\exists j \in J\ vars(\delta(y_j)) \subseteq \cup_{i\in I} vars(\delta(x_i))) \Rightarrow\\ &\qquad (\exists k \in K\ \exists n \in N\ vars(\delta(z_n^k)) \subseteq \cup_{m\in M} vars(\delta(w_m^k)))\\ &\Leftrightarrow (\exists j \in J\ vars(\theta(y_j)) \subseteq \cup_{i\in I} vars(\theta(x_i))) \Rightarrow\\ &\qquad (\exists k \in K \exists n \in N\ vars(\theta(z_n^k)) \subseteq \cup_{m\in M} vars(\theta(w_m^k)))\\ &\Leftrightarrow \exists k \in K\ ((\exists j \in J\ vars(\theta(y_j)) \subseteq \cup_{i\in I} vars(\theta(x_i))) \Rightarrow\\ &\qquad (\exists n \in N\ vars(\theta(z_n^k)) \subseteq \cup_{m\in M} vars(\theta(w_m^k))))\\ &\Leftrightarrow \exists k \in K\ (\forall \delta \leq \theta\ (\exists j \in J\ vars(\delta(y_j)) \subseteq \cup_{i\in I} vars(\delta(x_i))) \Rightarrow\\ &\qquad (\exists n \in N\ vars(\delta(z_n^k)) \subseteq \cup_{m\in M} vars(\delta(w_m^k))))\\ &\Leftrightarrow \theta \in \textstyle\bigvee_{k\in K} ((\bigwedge_{i\in I} x_i \to \bigvee_{j\in J} y_j) \to (\bigwedge_{m\in M} w_m^k \to \bigvee_{n\in N} z_n^k))\end{aligned}$$

which concludes the proof. $\Box$

Therefore, being disjunctive, $\curlyvee(\mathcal{P}os)$ is also the greatest solution of the following equation:

$$X = \mathcal{G} \sqcap (X \xrightarrow{\wedge} X) \sqcap \curlyvee(X).$$

Intuitively, it means that $\curlyvee(\mathcal{P}os)$ is the most concrete domain we can obtain by disjunctive completion and Heyting completion refinements starting from $\mathcal{G}$ and, in view of the result of completeness of $\mathcal{P}os$ with respect to $\curlyvee(\mathcal{P}os)$, it formally confirms that $\mathcal{P}os$ is definitely the best domain for groundness analysis.

8 Optimality in the literature

The result that $\mathcal{P}os$ is closed with respect to Heyting completion helps us to better understand the properties of the abstract unification of $\mathcal{P}os$ with respect to $\mathcal{D}ef$'s. In the classic literature of abstract interpretation, many notions of optimality have been proposed in order to compare abstract domains (cf. [7,5,16]). In any abstract domain, abstract meet is always the best correct approximation of the concrete one by construction. So we do not need to distinguish between abstract and concrete meet. Moreover, since unification is exactly meet operation ([26]), it follows that abstract unification is always the best correct approximation of the concrete one, by simply choosing as abstract unification the meet operation (Corollary 6.6 in [22]). Therefore, for both domains $\mathcal{P}os$ and $\mathcal{D}ef$, abstract unification is the best correct approximation of concrete unification, i.e., for each $A, B \in \wp^{\downarrow}(Sub)$ it holds:

$$\alpha_{\mathcal{P}os}(A) \wedge \alpha_{\mathcal{P}os}(B) = \alpha_{\mathcal{P}os}(\alpha_{\mathcal{P}os}(A) \wedge \alpha_{\mathcal{P}os}(B))$$
$$\alpha_{\mathcal{D}ef}(A) \wedge \alpha_{\mathcal{D}ef}(B) = \alpha_{\mathcal{D}ef}(\alpha_{\mathcal{D}ef}(A) \wedge \alpha_{\mathcal{D}ef}(B)).$$

On the contrary, $\mathcal{P}os$ and $\mathcal{D}ef$ are not complete for unification, i.e., it does not hold neither $\alpha_{\mathcal{P}os}(A) \wedge \alpha_{\mathcal{P}os}(B) = \alpha_{\mathcal{P}os}(A \wedge B)$ nor $\alpha_{\mathcal{D}ef}(A) \wedge \alpha_{\mathcal{D}ef}(B) = \alpha_{\mathcal{D}ef}(A \wedge B)$ [5] , as shown in the next example.

Example 20 *Let* $\theta = \{ X \backslash f(Y, a) \}$ *and* $\sigma = \{ X \backslash f(a, Y) \}$. *It holds that*

$$\alpha_{\mathcal{P}os}(\downarrow \theta \wedge \downarrow \sigma) = \alpha_{\mathcal{D}ef}(\downarrow \theta \wedge \downarrow \sigma) = X \wedge Y$$

$$\alpha_{\mathcal{P}os}(\downarrow \theta) \wedge \alpha_{\mathcal{P}os}(\downarrow \sigma) = \alpha_{\mathcal{D}ef}(\downarrow \theta) \wedge \alpha_{\mathcal{D}ef}(\downarrow \sigma) = X \leftrightarrow Y.$$

We propose a new notion of optimality which is able to discriminate between $\mathcal{P}os$ and $\mathcal{D}ef$, that is a characterization which is stronger than best correct

[5] A similar (negative) result for $\mathcal{P}os \cup \{ \emptyset \}$ already appears in [5]

approximation and weaker than completeness. The idea is to consider unifications between an abstract and a concrete object. We shall prove that $\mathcal{P}os$ satisfies the equality:

$$\alpha_{\mathcal{P}os}(\alpha_{\mathcal{P}os}(A) \wedge B) = \alpha_{\mathcal{P}os}(A) \wedge \alpha_{\mathcal{P}os}(B)$$

while $\mathcal{D}ef$ does not.

Since $\mathcal{D}ef = \mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}$, by Proposition 4, $\mathcal{D}ef$ enjoys the following property:

$$\alpha_{\mathcal{G}}(\alpha_{\mathcal{G}}(A) \wedge B) = \alpha_{\mathcal{G}}(\alpha_{\mathcal{G}}(A) \wedge \alpha_{\mathcal{D}ef}(B)).$$

Analogously, since $\mathcal{P}os = \mathcal{D}ef \xrightarrow{\curlywedge} \mathcal{D}ef$, it holds that:

$$\alpha_{\mathcal{D}ef}(\alpha_{\mathcal{D}ef}(A) \wedge B) = \alpha_{\mathcal{D}ef}(\alpha_{\mathcal{D}ef}(A) \wedge \alpha_{\mathcal{P}os}(B)).$$

In particular, they are the most abstract domains which satisfy the above equalities. In addition, since $\mathcal{P}os$ is closed for Heyting completion, by Proposition 4, the following equality holds.

$$\alpha_{\mathcal{P}os}(\alpha_{\mathcal{P}os}(A) \wedge B) = \alpha_{\mathcal{P}os}(A) \wedge \alpha_{\mathcal{P}os}(B).$$

Note that the domain $\mathcal{D}ef$ does not satisfy this property, as proved in the next example.

Example 21 *Let* $\theta = \{ X \backslash a \}$, $\sigma = \{ Y \backslash a \}$ *and* $\eta = \{ X \backslash Y \}$. *We have that*

$$\alpha_{\mathcal{D}ef}((\downarrow \theta \vee \downarrow \sigma) \wedge (\downarrow \eta)) = \alpha_{\mathcal{D}ef}(X \wedge Y) = X \wedge Y$$

$$\alpha_{\mathcal{D}ef}(\alpha_{\mathcal{D}ef}(\downarrow \theta \vee \downarrow \sigma) \wedge (\downarrow \eta)) = \alpha_{\mathcal{D}ef}(Sub \wedge (\downarrow \eta)) = \alpha_{\mathcal{D}ef}(\downarrow \eta) = X \Leftrightarrow Y.$$

It is worth noting that $\mathcal{P}os$ is the most abstract domain which contains $\mathcal{G}$ and satisfies the above property. In this way we obtain a precise characterization of $\mathcal{P}os$, which depends on the domain $\mathcal{G}$ only. The next theorem shows this result.

Corollary 22 *$\mathcal{P}os$ is the most abstract domain which contains $\mathcal{G}$ and enjoys the following property:* $\forall A, B \in \wp^{\downarrow}(Sub)$

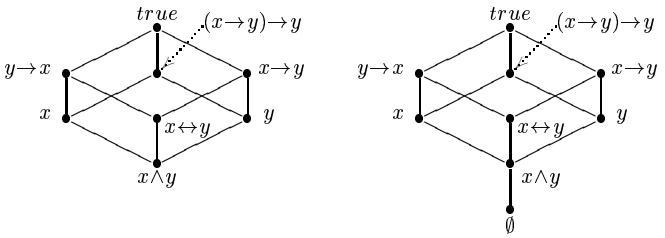
$$\alpha_{\mathcal{P}os}(\alpha_{\mathcal{P}os}(A) \wedge B) = \alpha_{\mathcal{P}os}(A) \wedge \alpha_{\mathcal{P}os}(B)$$

PROOF. By Theorem 14 and Proposition 4, it follows that $\alpha_{\mathcal{P}os}(\alpha_{\mathcal{P}os}(A) \wedge B) = \alpha_{\mathcal{P}os}(A) \wedge \alpha_{\mathcal{P}os}(B)$. Moreover, by Theorem 17, $\mathcal{P}os$ is the most abstract domain which contains $\mathcal{G}$ and is closed with respect to Heyting completion. Therefore, it is the most abstract domain which contains $\mathcal{G}$ and satisfies the above equality. □

Recall that the meet operation is precisely unification. Therefore, $\mathcal{P}os$ represents the most abstract domain where abstract unification is complete, providing that at least one of the arguments belongs to $\mathcal{P}os$. The importance of this characterization is that it allows us to discriminate between the precision of abstract unification in the domains $\mathcal{P}os$ and $\mathcal{D}ef$. In fact, for both domains, abstract unification is the best correct approximation of the concrete unification and, in addition, neither $\mathcal{P}os$ nor $\mathcal{D}ef$ are complete for unification. On the contrary, the previous equality precisely captures the threshold between the two domains. Moreover, this is not only a result of differentiation, but a complete characterization of $\mathcal{P}os$ since it turns out that $\mathcal{P}os$ is the most abstract domain which satisfies this equality, which could be taken as an alternative definition of $\mathcal{P}os$.

9 Reachability analysis

Some authors (cf. [5,24,2]) consider a slightly different definition of the domain $\mathcal{P}os$, by including also the empty set of substitutions as least element. Given a domain A, we denote by $A_\perp$ the domain $A \cup \{ \emptyset \}$, which is always a Moore family. For $Var = \{ x, y \}$, the domain $\mathcal{P}os$ and $\mathcal{P}os_\perp$ are depicted below.



$\mathcal{P}os$ $\qquad\qquad$ $\mathcal{P}os_\perp$

The domain $\mathcal{P}os_\perp$ is used to perform reachability analysis. When a fixpoint semantics is computed, the starting value is usually the least element of the domain. In case the semantics of a program is empty, the concrete semantics is $\emptyset$, while the corresponding abstract element in $\mathcal{P}os$ is the conjunction of all variables of interest. The abstract domain can not distinguish between programs with empty semantics and programs whose answer solutions are all ground. In order to make this distinction clear and, therefore, to add an extra precision to the domain, $\mathcal{P}os_\perp$ is used. When the abstract semantics is $\emptyset$, we conclude that no predicate has a solution. In some sense, no solution is "reached" during the computation.

We show that our reconstruction of $\mathcal{P}os$ applies to $\mathcal{P}os_\perp$ as well, by considering as starting domain the domain $\mathcal{G}_\perp = \mathcal{G} \cup \{ \emptyset \}$. This domain can be seen as the reduced product of $\mathcal{G}$ and the two-points domain $\{ \top, \emptyset \}$, which is the basic domain for reachability analysis. Note that $\mathcal{G}_\perp$ is the most abstract domain

which encodes groundness and reachability analysis. The next theorem shows that $\mathcal{P}os_\perp$ can be obtained by $\mathcal{G}_\perp$ with three steps of the Heyting completion operator and that $\mathcal{P}os_\perp$ is still closed.

Theorem 23 *The following equations hold.*

- $\mathcal{G}_\perp = \curlywedge(Var \cup \{\,\emptyset\,\})$
- $\mathcal{D}ef_\perp = \mathcal{G}_\perp \xrightarrow{\curlywedge} \mathcal{G}_\perp$
- $\mathcal{P}os_\perp = (\mathcal{G}_\perp \xrightarrow{\curlywedge} \mathcal{G}_\perp) \xrightarrow{\curlywedge} \mathcal{G}_\perp$
- $\mathcal{P}os_\perp = \mathcal{P}os_\perp \xrightarrow{\curlywedge} \mathcal{P}os_\perp$

PROOF. $\mathcal{G}_\perp = \curlywedge(Var \cup \{\,\emptyset\,\})$ trivially holds by definition of $\mathcal{G}_\perp$.

Recall that the least element $\perp$ of $\mathcal{P}os$ is $\bigwedge Var$. We first show that $\perp \rightarrow \emptyset = \emptyset$. The proof is for contradiction. Suppose there exists a substitution $\theta \in \perp \rightarrow \emptyset$. By definition, $\forall \delta \leq \theta\ \delta \in \perp\ \Rightarrow \delta \in \emptyset$, that is $\forall \delta \leq \theta\ \delta \notin \perp$. But given any substitution θ, there always exists an instance which grounds all the variables in Var, since Var is finite. Therefore $\perp \rightarrow \emptyset = \emptyset$. It follows that, for each $a \in \mathcal{P}os$, $a \rightarrow \emptyset \leq \perp \rightarrow \emptyset = \emptyset$, that is $a \rightarrow \emptyset = \emptyset$. Moreover for each $a \in \mathcal{P}os$, $\emptyset \rightarrow a = \top$. It follows that

- $\mathcal{G}_\perp \xrightarrow{\curlywedge} \mathcal{G}_\perp \subseteq \mathcal{D}ef_\perp$
- $(\mathcal{G}_\perp \xrightarrow{\curlywedge} \mathcal{G}_\perp) \xrightarrow{\curlywedge} \mathcal{G}_\perp \subseteq \mathcal{P}os_\perp$
- $\mathcal{P}os_\perp \xrightarrow{\curlywedge} \mathcal{P}os_\perp \subseteq \mathcal{P}os_\perp$

We prove the other inclusions by monotonicity.

- $\mathcal{P}os_\perp \subseteq \mathcal{P}os_\perp \xrightarrow{\curlywedge} \mathcal{P}os_\perp$ trivially holds.
- $\mathcal{D}ef_\perp = \mathcal{D}ef \cup \{\,\emptyset\,\} = (\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) \cup \{\,\emptyset\,\} \subseteq (\mathcal{G}_\perp \xrightarrow{\curlywedge} \mathcal{G}_\perp) \cup \{\,\emptyset\,\} = \mathcal{G}_\perp \xrightarrow{\curlywedge} \mathcal{G}_\perp$
- $\mathcal{P}os_\perp = \mathcal{P}os \cup \{\,\emptyset\,\} = ((\mathcal{G} \xrightarrow{\curlywedge} \mathcal{G}) \xrightarrow{\curlywedge} \mathcal{G}) \cup \{\,\emptyset\,\} \subseteq ((\mathcal{G}_\perp \xrightarrow{\curlywedge} \mathcal{G}_\perp) \xrightarrow{\curlywedge} \mathcal{G}_\perp) \cup \{\,\emptyset\,\} = (\mathcal{G}_\perp \xrightarrow{\curlywedge} \mathcal{G}_\perp) \xrightarrow{\curlywedge} \mathcal{G}_\perp$. $\square$

As a consequence, an analogous result of Corollary 22 holds for the domain $\mathcal{P}os_\perp$, as shown in the next corollary.

Corollary 24 *$\mathcal{P}os_\perp$ is the most abstract domain which contains $\mathcal{G}_\perp$ and enjoys the following property: $\forall A, B \in \wp^{\downarrow}(Sub)$*

$$\alpha_{\mathcal{P}os_\perp}(\alpha_{\mathcal{P}os_\perp}(A) \wedge B) = \alpha_{\mathcal{P}os_\perp}(A) \wedge \alpha_{\mathcal{P}os_\perp}(B).$$

PROOF. The proof is analogous to that one for Corollary 22. $\square$

10 Program independent domains

In the previous sections, we have fixed a finite set of variables $V \subset_f \mathcal{V}$, where $\mathcal{V}$ is denumerable, and shown how to automatically build the domains $\mathcal{G}_\perp^V$, $\mathcal{D}ef_\perp^V$ and $\mathcal{P}os_\perp^V$. This naturally induces an ordering on the abstract domains for groundness analysis built from different sets of variables. A question which naturally arises is if it is possible to construct a $\mathcal{P}os$-like abstract domain which is independent from the fixed variables. We show that the answer is positive and this domain is exactly the reduced product of all domains $\mathcal{P}os_\perp^V$ for each finite set V. Since all operators used in the construction process are monotone, the ordering is simply the set theoretical one on underlying sets of variables.

Proposition 25 *Let $V \subseteq W \subset_f \mathcal{V}$.*

- $\mathcal{G}_\perp^W \sqsubseteq \mathcal{G}_\perp^V$
- $\mathcal{D}ef_\perp^W \sqsubseteq \mathcal{D}ef_\perp^V$
- $\mathcal{P}os_\perp^W \sqsubseteq \mathcal{P}os_\perp^V$.

PROOF. Straightforwardly from the fact that all operators used in the definitions are monotone. □

Moreover, for the basic case of the domain $\mathcal{G}$, the following property holds.

Proposition 26 *Let $V, W \subset_f \mathcal{V}$. $\mathcal{G}_\perp^V \sqcap \mathcal{G}_\perp^W = \mathcal{G}_\perp^{V \cup W}$.*

PROOF.
$$\begin{aligned} \mathcal{G}_\perp^V \sqcap \mathcal{G}_\perp^W &= \curlywedge(V \cup \{\,\emptyset\,\}) \sqcap \curlywedge(W \cup \{\,\emptyset\,\}) \\ &= \curlywedge(V \cup W \cup \{\,\emptyset\,\}) \\ &= \mathcal{G}_\perp^{V \cup W}. \end{aligned}$$ □

The idea is to exploit the ordering on these domains to construct program independent domains, which are suitable for each program of any dimension and are still based on the same concrete domain $\wp^\downarrow(Sub)$. In this section we show that this is always possible and that the new domains still enjoy the same properties of the corresponding finite domains.

The basic idea is to consider the reduced product of all domains for any finite set of variables. Clearly, such domains are able to deal with any variable in $\mathcal{V}$.

We define the domain $\mathcal{G}^\omega$, $\mathcal{D}ef^\omega$ and $\mathcal{P}os^\omega$ as follows:

$$\begin{aligned}\mathcal{G}^\omega &= \sqcap_{V \subset_f \mathcal{V}} \mathcal{G}^V \\ \mathcal{D}ef^\omega &= \sqcap_{V \subset_f \mathcal{V}} \mathcal{D}ef^V \\ \mathcal{P}os^\omega &= \sqcap_{V \subset_f \mathcal{V}} \mathcal{P}os^V .\end{aligned}$$

We show that all the relations proved in the finite case still hold. We start, as before, from the basic domain $\mathcal{G}^\omega$ and reconstruct the other two as Heyting completion and double Heyting completion. As before, we let a variable v denote the set of substitutions which ground v. First note that, all domains contain the element $\bot$ obtained as the infinite conjunction of all variables in $\mathcal{V}$.

Proposition 27 $\bigwedge \mathcal{V} = \emptyset$.

PROOF. By definition, we have that $\bigwedge \mathcal{V} = \{\, \theta \in Sub \mid \forall v \in \mathcal{V}\ \theta \in v \,\} = \{\, \theta \in Sub \mid \forall v \in \mathcal{V}\ vars(\theta(v)) = \emptyset \,\}$. Again by definition, there exists no substitution which grounds an infinite set of variables. Since $|\mathcal{V}|$ is infinite, it implies that $\bigwedge \mathcal{V} = \emptyset$. □

Corollary 28 *The following properties hold.*

- $\mathcal{G}^\omega = \sqcap_{V \subset_f \mathcal{V}} \mathcal{G}^V_\bot$
- $\mathcal{D}ef^\omega = \sqcap_{V \subset_f \mathcal{V}} \mathcal{D}ef^V_\bot$
- $\mathcal{P}os^\omega = \sqcap_{V \subset_f \mathcal{V}} \mathcal{P}os^V_\bot$

PROOF. By definition, all domains contain all variables $v \in \mathcal{V}$ and are closed by possibly infinite meets. Therefore they contain $\bigwedge \mathcal{V}$. □

Before proving the properties and relations among the new domains, we show that they do not contain any infinite conjunction different from $\bot$, that is any possibly infinite conjunction either reduces to a finite one or is equivalent to $\bot$. This allows us to keep on using a finitary propositional logic instead of an infinitary one. We first prove that $\mathcal{G}^\omega$ can be built by using only finite formulas and the empty set.

Proposition 29 $\mathcal{G}^\omega = \{\, \bigwedge V \mid V \subset_f \mathcal{V} \,\} \cup \{\, \emptyset \,\}$.

PROOF. One direction is trivial, that is $\mathcal{G}^\omega \supseteq \{\, \bigwedge V \mid V \subset_f \mathcal{V} \,\} \cup \{\, \emptyset \,\}$. Let $x \in \mathcal{G}^\omega$. Then there exists a family of variables $v_i \in \mathcal{V}$ for $i \in I$ such that $x = \bigwedge_{i \in I} v_i$. If $|\{\, v_i \,\}_{i \in I}|$ is finite, then there exists $W \subset_f \mathcal{V}$ such that $\bigwedge_{i \in I} v_i =$

$\bigwedge_{v_i \in W} v_i$ and the thesis trivially follows. Assume $|\{ v_i \}_{i \in I}|$ to be infinite. Then $\bigwedge_{i \in I} v_i = \{ \theta \in Sub \mid \forall i \in I \ \theta \in v_i \} = \{ \theta \in Sub \mid \forall i \in I \ vars(\theta(v_i)) = \emptyset \}$. As $|\{ v_i \}_{i \in I}|$ is infinite, by definition of substitution, there exists no substitution which grounds an infinite set of variables. Therefore $x = \emptyset$. $\square$

This result allows us to establish an isomorphism between the domain $\mathcal{G}^\omega$ and $\wp_f(\mathcal{V})$, analogously to the finite case. As suggested by Proposition 29, in order to obtain the isomorphism, we only need to add to $\wp_f(\mathcal{V})$ a least element $\bot$, by extending the ordering in the obvious way.

Corollary 30 *$\mathcal{G}^\omega$ is isomorphic to $\wp_f(\mathcal{V}) \cup \{ \bot \}$.*

PROOF. Straightforward from Proposition 29 by simply representing subsets as conjunctions and mapping the empty set of substitutions to $\bot$. $\square$

It is worth noting that $\mathcal{G}^\omega$ is not isomorphic to $\wp(\mathcal{V})$, since the latter one contains distinct infinite sets of variables, all of which should be reduced to the empty set of substitutions. Therefore we obtain a Galois connection with the concrete domain, and not a Galois insertion. As a consequence, both domains $\mathcal{G}^\omega$, $\mathcal{D}ef^\omega$ and $\mathcal{P}os^\omega$ can be reconstructed from the corresponding finite ones without introducing infinite conjunctions nor infinite implications.

Proposition 31 *The following properties hold.*

(1) $\mathcal{G}^\omega = \cup_{V \subset_f \mathcal{V}} \mathcal{G}^V_\bot$
(2) $\mathcal{D}ef^\omega = \cup_{V \subset_f \mathcal{V}} \mathcal{D}ef^V_\bot$
(3) $\mathcal{P}os^\omega = \cup_{V \subset_f \mathcal{V}} \mathcal{P}os^V_\bot$.

PROOF. [(Sketch)]

(1) $\mathcal{G}^\omega = \cup_{V \subset_f \mathcal{V}} \mathcal{G}^V_\bot$ is straightforward by Proposition 29.
(2) Let $a_i \in \cup_{V \subset_f \mathcal{V}} \mathcal{D}ef^{\mathcal{V}}_\bot$ for $i \in I$. We prove that $\bigwedge_{i \in I} a_i \in \cup_{V \subset_f \mathcal{V}} \mathcal{D}ef^{\mathcal{V}}_\bot$. If $\bigwedge_{i \in I} a_i$ is equivalent to either a finite conjunction of implications or $\top$ we have nothing to prove. Otherwise, we claim that $\bigwedge_{i \in I} a_i = \bot$. In fact, by definition, a substitution can involve only a finite set of variables. For contradiction, if there exists a substitution $\theta \in \bigwedge_{i \in I} a_i$, it would follows that either θ binds a variables to a term which contains infinite variables, i.e., an infinite term, or θ grounds an infinite number of variables. This is because $\bigwedge_{i \in I} a_i$ is not equivalent to a finite formula. But θ would not be a substitution, by definition. Therefore $\mathcal{D}ef^\omega = \cup_{V \subset_f \mathcal{V}} \mathcal{D}ef^V_\bot$ holds.
(3) Analogous to the previous point. $\square$

The last characterization says that when constructing $\mathcal{G}^\omega$ starting from $\mathcal{G}^V_\perp$, we can simply take the union of all such domains and this is actually a Moore family, i.e., we do not need to add infinite conjunctions in order to obtain an abstract domain. This proves that when dealing with $\mathcal{G}^\omega$, $\mathcal{D}ef^\omega$ and $\mathcal{P}os^\omega$, we do not have to worry about infinite formulas and we can use a finitary logic to describe abstract objects. We now reconstruct the relations between the new domains. We first need a technical Lemma which allows us to combine such domains.

Lemma 32 *Let $A_i, B \in Moore(\wp^\downarrow(Sub))$ for $i \in I$. If $\bigsqcap_{i\in I} A_i = \bigcup_{i\in I} A_i$ then $(\bigsqcap_{i\in I} A_i) \xrightarrow{\curlywedge} B = \bigsqcap_{i\in I}(A_i \xrightarrow{\curlywedge} B)$.*

PROOF. By monotonicity of Heyting completion, it holds $\bigsqcap_{i\in I}(A_i \xrightarrow{\curlywedge} B) \subseteq \bigsqcap_{i\in I}((\bigsqcap_{i\in I} A_i) \xrightarrow{\curlywedge} B) = (\bigsqcap_{i\in I} A_i) \xrightarrow{\curlywedge} B$.

For the other inclusion, we show that for all $a \in \bigsqcap_{i\in I} A_i$ and $b \in B$ it holds $a \to b \in \bigsqcap_{i\in I}(A_i \xrightarrow{\curlywedge} B)$. By hypothesis, $\bigsqcap_{i\in I} A_i = \bigcup_{i\in I} A_i$. Therefore, there exists $j \in I$ such that $a \in A_j$. Thus $a \to b \in A_j \xrightarrow{\curlywedge} B$. By monotonicity $a \to b \in \bigsqcap_{i\in I}(A_i \xrightarrow{\curlywedge} B)$. $\square$

Theorem 33 *The following properties hold.*

(1) $\mathcal{G}^\omega = \curlywedge(\mathcal{V})$
(2) $\mathcal{D}ef^\omega = \mathcal{G}^\omega \xrightarrow{\curlywedge} \mathcal{G}^\omega$
(3) $\mathcal{P}os^\omega = (\mathcal{G}^\omega \xrightarrow{\curlywedge} \mathcal{G}^\omega) \xrightarrow{\curlywedge} \mathcal{G}^\omega$
(4) $\mathcal{P}os^\omega = \mathcal{P}os^\omega \xrightarrow{\curlywedge} \mathcal{P}os^\omega$

PROOF.

(1) $\mathcal{G}^\omega = \sqcap_{V\subset_f\mathcal{V}} \mathcal{G}^V_\perp$
$= \sqcap_{V\subset_f\mathcal{V}} \curlywedge(V \cup \{\emptyset\})$
$= \curlywedge(\cup_{V\subset_f\mathcal{V}} V \cup \{\emptyset\})$
$= \curlywedge(\mathcal{V} \cup \{\emptyset\})$ [by Proposition 27]
$= \curlywedge(\mathcal{V})$.

(2) $\mathcal{D}ef^\omega = \sqcap_{V\subset_f\mathcal{V}} \mathcal{D}ef^V$
$= \sqcap_{V\subset_f\mathcal{V}} (\mathcal{G}^V \xrightarrow{\curlywedge} \mathcal{G}^V)$
$\subseteq \sqcap_{V\subset_f\mathcal{V}} (\mathcal{G}^\omega \xrightarrow{\curlywedge} \mathcal{G}^\omega)$
$= \mathcal{G}^\omega \xrightarrow{\curlywedge} \mathcal{G}^\omega$.

By monotonicity of Heyting completion, it holds:

$$\begin{aligned}
\mathcal{G}^\omega \xrightarrow{\curlywedge} \mathcal{G}^\omega &= (\sqcap_{V \subset_f \mathcal{V}} \mathcal{G}^V) \xrightarrow{\curlywedge} (\sqcap_{W \subset_f \mathcal{V}} \mathcal{G}^W) \\
&= \sqcap_{W \subset_f \mathcal{V}} ((\sqcap_{V \subset_f \mathcal{V}} \mathcal{G}^V) \xrightarrow{\curlywedge} \mathcal{G}^W) \\
&= \sqcap_{W \subset_f \mathcal{V}} \sqcap_{V \subset_f \mathcal{V}} (\mathcal{G}^V \xrightarrow{\curlywedge} \mathcal{G}^W) \\
&\subseteq \sqcap_{W \subset_f \mathcal{V}} \sqcap_{V \subset_f \mathcal{V}} ((\mathcal{G}^V \sqcap \mathcal{G}^W) \xrightarrow{\curlywedge} (\mathcal{G}^W \sqcap \mathcal{G}^V)) \\
&\subseteq \sqcap_{W \subset_f \mathcal{V}} \sqcap_{V \subset_f \mathcal{V}} (\mathcal{G}^{V \cup W} \xrightarrow{\curlywedge} \mathcal{G}^{V \cup W}) \\
&\subseteq \sqcap_{W \subset_f \mathcal{V}} \sqcap_{V \subset_f \mathcal{V}} (\mathcal{D}ef^{V \cup W}) \\
&= \sqcap_{V \subset_f \mathcal{V}} (\mathcal{D}ef^V).
\end{aligned}$$

(3) analogous to Point (2) above.

(4) by monotonicity of Heyting completion, $\mathcal{P}os^\omega \subseteq \mathcal{P}os^\omega \xrightarrow{\curlywedge} \mathcal{P}os^\omega$ holds. For the other inclusion:

$$\begin{aligned}
\mathcal{P}os^\omega \xrightarrow{\curlywedge} \mathcal{P}os^\omega &= (\sqcap_{V \subset_f \mathcal{V}} \mathcal{P}os^V) \xrightarrow{\curlywedge} (\sqcap_{W \subset_f \mathcal{V}} \mathcal{P}os^W) \\
&= \sqcap_{W \subset_f \mathcal{V}} ((\sqcap_{V \subset_f \mathcal{V}} \mathcal{P}os^V) \xrightarrow{\curlywedge} \mathcal{P}os^W) \\
&= \sqcap_{W \subset_f \mathcal{V}} \sqcap_{V \subset_f \mathcal{V}} (\mathcal{P}os^V \xrightarrow{\curlywedge} \mathcal{P}os^W) \\
&\subseteq \sqcap_{W \subset_f \mathcal{V}} \sqcap_{V \subset_f \mathcal{V}} (\mathcal{P}os^{V \cup W} \xrightarrow{\curlywedge} \mathcal{P}os^{V \cup W}) \\
&= \sqcap_{W \subset_f \mathcal{V}} \sqcap_{V \subset_f \mathcal{V}} (\mathcal{P}os^{V \cup W}) \\
&= \sqcap_{V \subset_f \mathcal{V}} (\mathcal{P}os^V). \qquad \square
\end{aligned}$$

Our approach in constructing a unique, program independent domain differs from [5] and [22] mainly in the choice of the concrete domain. In [5], the authors consider a different concrete domain: $(\wp(Sub) \times \wp_f(\mathcal{V})) \cup \{ \top_c, \bot_c \}$. Our results show that, even keeping the simple concrete domain $\wp^\downarrow(Sub)$, we can construct program independent domains. Moreover, our approach allows us to compare different abstract domains for different sets of variables, while they are incomparable in the other approach. In [22], a still different concrete domain is used, i.e., sets of ex-equations. Intuitively, ex-equations are existentially quantified conjunctions of basic term equations. The whole SLD resolution theory is then lifted to ex-equations, by extending the usual definitions of unification and semantics to this domain. It is known from [26] that the lattice of idempotent substitutions is isomorphic to the lattice of term equations. By definition, ex-equations are more expressive, and therefore more concrete, than simple equations. As a consequence, sets of ex-equations are more concrete than $\wp(Sub)$, and thus than $\wp^\downarrow(Sub)$. But, as shown by our construction, $\wp^\downarrow(Sub)$ is concrete enough to formalize program independent groundness analysis.

11 Conclusions

In this paper we have reconstructed the domain $\mathcal{P}os$ in a completely automatic way, by using the domain $\mathcal{G}$ and the properties of the concrete domain only. With our formalization, we reobtain the standard properties of $\mathcal{P}os$ simply by construction. We show that we can get rid of positive formulas and equivalence classes in the definition of $\mathcal{P}os$ and use directly the operations which naturally arise from the concrete domain. Moreover, we show a result of optimality by proving that $\mathcal{P}os$ contains exactly all and only the elements really useful to the analysis, from a computational point of view. We find an alternative equational definition of $\mathcal{P}os$ in terms of the properties of the abstract unification and show that this characterization precisely captures the threshold between the domains $\mathcal{P}os$ and $\mathcal{D}ef$. This result provides a constructive proof that $\mathcal{P}os$ is a well-built domain from both the formal and the logical point of view. Finally, we propose a new, program independent domain for groundness analysis, which still enjoys all the properties of the standard domain $\mathcal{P}os$.

In our opinion, the main feature of the construction we present, is that it can easily be applied to other kind of analyses of logic programs, since it depends only on the properties of the concrete domain. We trust that many other analyses can be naturally formalized in this way, in particular analyses of properties closed under instantiation, such as type analysis, where the concrete domain is always $\wp^{\downarrow}(Sub)$.

References

[1] K. R. Apt. Introduction to logic programming. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, pages 495–574. Elsevier, Amsterdam and The MIT Press, Cambridge, 1990.

[2] T. Armstrong, K. Marriott, P. Schachte, and H. Søndergaard. Boolean functions for dependency analysis: algebraic properties and efficient representation. In B. Le Charlier, editor, *Proc. of the 1st Int'l Static Analysis Symposium (SAS '94)*, volume 864 of *LNCS*, pages 266–280. Springer-Verlag, 1994.

[3] G. Birkhoff. Lattice theory. In *AMS Colloquium Publication, third ed.* AMS Press, 1967.

[4] A. Cortesi, G. Filè, and W. Winsborough. *Prop* revisited: Propositional formula as abstract domain for groundness analysis. In *Proc. Sixth IEEE Symp. on Logic In Computer Science*, pages 322–327. IEEE Computer Society Press, 1991.

[5] A. Cortesi, G. Filè, and W. Winsborough. Optimal groundness analysis using propositional logic. *Journal of Logic Programming*, 27(2):137–167, 1996.

[6] P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the 4th ACM Symposium on Principles of Programming Languages (POPL '77)*, pages 238–252. ACM Press, 1977.

[7] P. Cousot and R. Cousot. Systematic design of program analysis frameworks. In *Conference Record of the 6th ACM Symposium on Principles of Programming Languages (POPL '79)*, pages 269–282. ACM Press, 1979.

[8] P. Cousot and R. Cousot. Abstract interpretation and applications to logic programs. *Journal of Logic Programming*, 13(2 & 3):103–179, 1992.

[9] S.K. Debray. Static inference of modes and data dependencies in logic programs. *ACM TOPLAS*, 11:418–450, 1989.

[10] G. Filé, R. Giacobazzi, and F. Ranzato. A unifying view on abstract domain design. *ACM Computing Surveys*, 28(2):333–336, 1996.

[11] G. Filé and F. Ranzato. Improving abstract interpretations by systematic lifting to the powerset. In M. Bruynooghe, editor, *Proc. of the 1994 Int'l Logic Programming Symposium (ILPS '94)*, pages 655–669. The MIT Press, 1994.

[12] G. Filé and F. Ranzato. The powerset operator on abstract interpretations. *Theoretical Computer Science*, 1998. To appear. Preliminary version in [11].

[13] R. Giacobazzi. "Optimal" collecting semantics for analysis in a hierarchy of logic program semantics. In C. Puech and R. Reischuk, editors, *Proc. of the 13th Int'l Symposium on Theoretical Aspects of Computer Science (STACS '96)*, volume 1046 of *LNCS*, pages 503–514. Springer-Verlag, 1992.

[14] R. Giacobazzi and F. Ranzato. Functional dependencies and Moore-set completions of abstract interpretations and semantics. In J. Lloyd, editor, *Proc. 1995 Int'l Symposium on Logic Programming (ILPS'95)*, pages 321–335. The MIT Press, 1995.

[15] R. Giacobazzi and F. Ranzato. Compositional optimization of disjunctive abstract interpretations. In H.R. Nielson, editor, *Proc. of the 1996 European Symposium on Programming*, volume 1058 of *LNCS*, pages 141–155. Springer-Verlag, Berlin, 1996.

[16] R. Giacobazzi and F. Ranzato. Completeness in abstract interpretation: A domain perspective. In M. Johnson, editor, *Proc. of the 6th International Conference on Algebraic Methodology and Software Technology (AMAST'97)*, volume 1349 of *LNCS*, pages 231–245. Springer-Verlag, Berlin, 1997.

[17] R. Giacobazzi and F. Scozzari. Intuitionistic implication in abstract interpretation. In H. Glaser, P. Hartel, and H. Kuchen, editors, *Proc. of Ninth International Symposium on Programming Languages, Implementations, Logics and Programs PLILP'97*, volume 1292 of *LNCS*, pages 175–189. Springer-Verlag, 1997.

[18] R. Giacobazzi and F. Scozzari. A logical model for relational abstract domains. *ACM Transactions on Programming Languages and Systems*, 20(5):1067–1109, 1998.

[19] G. Gierz, K.H. Hofmann, K. Keimel, J.D. Lawson, M. Mislove, , and D.S. Scott. *A Compendium of Continuous Lattices*. Springer-Verlag, 1980.

[20] J.-Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types*. Cambridge University Press, 1989.

[21] N. D. Jones and H. Søndergaard. A Semantics-based Framework for the Abstract Interpretation of Prolog. In S. Abramsky and C. Hankin, editors, *Abstract Interpretation of Declarative Languages*, pages 123–142. Ellis Horwood Ltd, 1987.

[22] K. Marriot, H. Søndergaard, and N. D. Jones. Denotational abstract interpretation of logic programs. *ACM TOPLAS*, 16(3):607–648, 1994.

[23] K. Marriott and H. Søndergaard. Abstract interpretation of logic programs: the denotational approach. In A. Bossi, editor, *Proc. GULP '90*, pages 399–425. Padova, 1990.

[24] K. Marriott and H. Søndergaard. Precise and efficient groundness analysis for logic programs. *ACM Letters on Programming Languages and Systems*, 2(1-4):181–196, 1993.

[25] J. Morgado. Some results on the closure operators of partially ordered sets. *PortugaliæMathematica*, 19(2):101–139, 1960.

[26] C. Palamidessi. Algebraic properties of idempotent substitutions. In M. S. Paterson, editor, *Proc. of the 17th International Colloquium on Automata, Languages and Programming*, volume 443 of *LNCS*, pages 386–399. Springer-Verlag, 1990.

[27] D. van Dalen. *Logic and Structure*. 3rd ed. Springer-Verlag, 1994.