Refining and Compressing Abstract Domains

Roberto Giacobazzi* Francesco Ranzato**

*Dipartimento di Informatica, Università di Pisa
Corso Italia 40, 56125 Pisa, Italy
giaco@di.unipi.it

**Dipartimento di Matematica Pura ed Applicata, Università di Padova
Via Belzoni 7, 35131 Padova, Italy
franz@math.unipd.it

Abstract. In the context of Cousot and Cousot's abstract interpretation theory, we present a general framework to define, study and handle operators modifying abstract domains. In particular, we introduce the notions of operators of refinement and compression of abstract domains: A refinement enhances the precision of an abstract domain; a compression operator (compressor) can exist relatively to a given refinement, and it simplifies as much as possible a domain of input for that refinement. The adequateness of our framework is shown by the fact that most of the existing operators on abstract domains fall in it. A precise relationship of adjunction between refinements and compressors is also given, justifying why compressors can be understood as inverses of refinements.

1 Introduction

It is well known that abstract domains play a fundamental rôle in abstract interpretation [5, 6], since the precision of an abstract interpretation-based program analysis strongly depends on the expressive power of the chosen abstract domain. Much work has been therefore devoted to define systematic operators for enhancing the precision of representation of abstract domains. Relevant examples are Cousot and Cousot's reduced product, disjunctive completion and reduced cardinal power [6], Nielson's tensor product [18], Giacobazzi and Ranzato's dependencies and dual-Moore-set completion [13], the open product and pattern completion of Cortesi et al. [4], to cite the most known ones. The basic idea is that richer abstract domains can be obtained by combining simpler ones or by lifting them by adding new information. These operators on abstract domains provide high level facilities to tune the analysis in accuracy and cost, and some of them have been included as tools for abstract domain design aid in modern systems for program analysis, like for instance in System Z [22] and in PLAI [1].

We carry on this idea of operators enhancing the precision of abstract domains and we present in Sect. 3 a general and precise framework to handle these operators, which encompasses and improves the ideas sketched in [9]. The central notion is that of *abstract domain refinement*, that intuitively is any operator performing an action of refinement on abstract domains, with respect to their standard ordering relation of precision. There exists a strong link between refinements and *closure operators*, and many lattice-theoretic properties of closures are inherited by refinements. We introduce a generic pattern of definition for domain refinements, which allows to recover most of the important refinements

listed above. Moreover, as an instance of this scheme, we present a new *refinement of completeness*. Roughly speaking, an abstract domain D is *complete* for a semantic function f defined on the concrete domain when no loss of precision is introduced by approximating f in the best possible way (i.e. by considering its best correct approximation, cf. [5, 6]) with respect to D. Thus, for a domain D, our refinement of completeness provides the most abstract domain which is more precise than D and complete for a given continuous concrete semantic function.

Recently, also operators of simplification of abstract domains have been defined and studied, like the operations of complementation in [3] and least disjunctive basis in [14]. As well as refinements, we show in Sect. 4 that these operators can be expressed in a formal and precise way in our framework. Actually, these operators are instances of our notion of *operator of compression* (or *compressor*). Roughly speaking, for a given abstract domain refinement $\mathfrak{R}$, its relative compressor simplifies a domain D of input for $\mathfrak{R}$, by returning the domain (if this exists) which contains the least amount of information required as input by $\mathfrak{R}$ to reach the same enhancement obtainable from D. This is somehow similar to the operation of compression on files – hence our terminology. In more precise terms, if $\mathfrak{R}$ is a unary refinement and D is an abstract domain, then an abstract domain A is the *optimal basis* of D for $\mathfrak{R}$, if A is the most abstract solution to the equation $\mathfrak{R}(X) = \mathfrak{R}(D)$. Obviously, if an optimal basis exists then it is necessarily unique. We say that $\mathfrak{R}$ is *invertible* on a given class of abstract domains if there exists the optimal basis of any domain D in the class. In this case, the compressor $\mathfrak{R}^-$ relative to $\mathfrak{R}$ (also called the *inverse* of $\mathfrak{R}$) provides the optimal basis $\mathfrak{R}^-(D)$ of D for $\mathfrak{R}$. The problem of inverting a refinement is often hard to solve in a satisfactory way, and, in general, not all domain refinements admit a corresponding compressor defined for a significant class of abstract domains. We show that complementation and least disjunctive basis give rise, respectively, to the compressors relative to reduced product and disjunctive completion refinements, and we give a generic scheme for defining invertible refinements. Moreover, we show that invertible refinements provide solutions to the problem of decomposing abstract domains into simpler factors. If $\mathfrak{R}$ is an n-ary refinement and $D = \mathfrak{R}(D_1, \ldots, D_n)$, then the tuple $\langle D_1, \ldots, D_n \rangle$ can be considered as a decomposition of D relative to $\mathfrak{R}$. We then present a general iterative method which starting from any decomposition relative to an invertible refinement provides minimal decompositions, i.e. decompositions involving the most abstract factors.

It is important to note that our notion of inversion of a refinement does not correspond to the more customary inversion in the sense of adjunctions – on the contrary, we observe that, in general, this is not possible. However, we show in Sect. 5 that this asymmetry can be overcome by considering a modified ordering relation between abstract domains, that is induced in a natural way by the refinement itself. We prove that for this lifted order on abstract domains, an invertible refinement and its compressor do constitute an adjunction. This provides a firm mathematical relationship between refinements and compressors, and gives a more precise justification to the use of the term "inverse".

2 Preliminaries

The structure $\langle uco(C), \sqsubseteq, \sqcup, \sqcap, \lambda x.\top, \lambda x.x\rangle$ denotes the complete lattice of all *upper closure operators* (shortly closures) on a complete lattice $\langle C, \leq, \vee, \wedge, \top, \bot\rangle$, where $\rho \sqsubseteq \eta$ iff $\forall x \in C.\ \rho(x) \leq \eta(x)$. The complete lattice of all *lower closure operators* on C is denoted by $lco(C)$ and is dual-isomorphic to $uco(C)$. Recall that each closure operator $\rho \in uco(C)$ is uniquely determined by the set of its fixpoints, which is its image, i.e. $\rho(C) = \{x \in C \mid \rho(x) = x\}$, that $\rho \sqsubseteq \eta$ iff $\eta(C) \subseteq \rho(C)$, and that a subset $X \subseteq C$ is the set of fixpoints of a closure iff $X = \{\wedge Y \mid Y \subseteq X\}$ (note that $\top \in X$). $\langle \rho(C), \leq\rangle$ is a complete meet subsemilattice of C but, in general, it is not a complete sublattice of C.

In the standard Cousot and Cousot abstract interpretation theory, abstract domains can be equivalently specified either by Galois connections or by closure operators [6]. In the first case, concrete and abstract domains are related by a pair of adjoint functions. This provides a way to relate domains containing objects having different representation. In the second case instead, an abstract domain is specified as (the set of fixpoints of) an upper closure on the concrete domain. Thus, the closure operator approach is particularly convenient when reasoning about properties of abstract domains independently from the representation of their objects, as in our case. Hence, we will identify $uco(C)$ with the complete lattice of all possible abstract domains of the concrete domain (i.e. any complete lattice) C. The ordering on $uco(C)$ corresponds precisely to the standard order used in abstract interpretation to compare abstract domains with regard to their precision: D_1 is *more precise* than D_2 iff $D_1 \sqsubseteq D_2$ in $uco(C)$ ($\sqsubset$ denotes strict ordering). The lub and glb on $uco(C)$ have therefore the following meaning as operators on domains. Suppose $\{D_i\}_{i\in I} \subseteq uco(C)$: (i) $\sqcup_{i\in I} D_i$ is the most concrete among the domains which are abstractions of all the D_i's, i.e. it is their least common abstraction; (ii) $\sqcap_{i\in I} D_i$ is (isomorphic to) the well-known reduced product of all the D_i's, and, equivalently, it is the most abstract among the domains (abstracting C) which are more concrete than every D_i. Whenever C is a meet-continuous complete lattice (i.e., for any chain $Y \subseteq C$ and $x \in C$: $x \wedge (\vee Y) = \vee_{y\in Y}(x \wedge y)$), $uco(C)$ enjoys the lattice-theoretic property of *pseudocomplementedness* (cf. [12]). This property allowed to define the operation of *complementation* of abstract domains (cf. [3]), namely an operation which, starting from any two domains $C \sqsubseteq D$, where C is meet-continuous, gives as result the most abstract domain $C \sim D$, such that $(C \sim D) \sqcap D = C$.

3 Abstract Domain Refinements

Intuitively, an abstract domain refinement is an operator that, for any tuple $\langle D_i\rangle_{1\leq i\leq n}$ of domains of input (ranging on a given domain of definition), provides as output a domain more precise than each D_i. It is also very reasonable to expect that such an operator is monotone. These observations naturally lead to the definition below. In the following, a generic tuple of objects is denoted by $\mathbf{O}$, $\pi_i(\mathbf{O})$ denotes its i-th component, and $\mathbf{O}[X/i]$ denotes the tuple obtained from $\mathbf{O}$ by replacing $\pi_i(\mathbf{O})$ with X. Also, C is a complete lattice acting as the concrete domain and $\mathbf{U} \subseteq uco(C)^n$, $n \geq 1$, is a given tuple of sets of domains abstracting C (for simplicity, we only consider refinements of finite arity

– actually those having a practical meaning – although a generalization would be straightforward). When $n = 1$ we denote $\mathbf{U}$ as the set $U \subseteq uco(C)$. We extend on tuples the glb of $uco(C)$: For any tuple of domains $\mathbf{D}$, $\sqcap\mathbf{D} = \sqcap_{1\leq i\leq n}\pi_i(\mathbf{D})$.

Definition 3.1 A map $\mathfrak{R} : \mathbf{U} \to uco(C)$ is a (*n-ary abstract domain*) *refinement* if: (i) $\mathfrak{R}$ is monotone; (ii) $\mathfrak{R}$ is reductive: $\forall \mathbf{D} \in \mathbf{U}.\ \mathfrak{R}(\mathbf{D}) \sqsubseteq \sqcap\mathbf{D}$. □

The kernel of definition of any refinement $\mathfrak{R} : \mathbf{U} \to uco(C)$ is given by $\mathbb{K}_{\mathbf{U}} = \sqcap_{1\leq i\leq n}\pi_i(\mathbf{U})$. Often, refinements are defined on any tuple of abstract domains, i.e., $\mathfrak{R} : uco(C)^n \to uco(C)$, as in the case of reduced product and disjunctive completion, later considered. We will call them *full* refinements, in order to distinguish them from generic ones as allowed by Definition 3.1. Any n-ary refinement $\mathfrak{R} : \mathbf{U} \to uco(C)$ induces a family of refinements of lower arity obtained by fixing some of the domains of input. For instance, by fixing $n - 1$ domains, we get the unary refinements $\lambda X.\mathfrak{R}(\mathbf{D}[X/i]) : \pi_i(\mathbf{U}) \to uco(C)$. Also, $\mathfrak{R}$ induces the canonical unary *self-refinement* $\mathfrak{R}_{\mathbb{1}} : \mathbb{K}_{\mathbf{U}} \to uco(C)$ defined as $\mathfrak{R}_{\mathbb{1}}(D) = \mathfrak{R}(D, ..., D)$. Conversely, any n-uple $\mathbf{R} = \langle\mathfrak{R}_i\rangle_{1\leq i\leq n}$ of unary refinements $\mathfrak{R}_i : U_i \to uco(C)$ induces an n-ary refinement $\mathfrak{R}_{\mathbf{R}} : U_1\times...\times U_n \to uco(C)$ defined as $\mathfrak{R}_{\mathbf{R}}(\mathbf{D}) = \sqcap_{1\leq i\leq n}\mathfrak{R}_i(\pi_i(\mathbf{D}))$, and called *attribute independent*.

It is important to remark that Definition 3.1 lacks of any requirement of idempotence. For instance, for a unary refinement $\mathfrak{R} : U \to uco(C)$ may well happen that a refined domain $\mathfrak{R}(D) \in U$ can still be object of further refinement, i.e. $\mathfrak{R}(\mathfrak{R}(D)) \sqsubset \mathfrak{R}(D)$. Due to lack of space, in the paper we will only consider examples of idempotent refinements, although a relevant example of nonidempotent refinement can be given by the dependencies between abstract domains of [13]. However, it is worth noting that, by monotonicity, any refinement can be lifted to an idempotent one as the limit of a possibly transfinite Kleene fixpoint iteration sequence. It is therefore reasonable requiring idempotence for refinements, i.e. that a refinement upgrades abstract domains all at once.

Definition 3.2 An n-ary refinement $\mathfrak{R} : \mathbf{U} \to uco(C)$ is *idempotent* if for any $i \in [1, n]$ and $\mathbf{D} \in \mathbf{U}$ such that $\mathfrak{R}(\mathbf{D}) \in \mathbb{K}_{\mathbf{U}}$, $\mathfrak{R}(\mathbf{D}) = \mathfrak{R}(\mathbf{D}[\mathfrak{R}(\mathbf{D})/i])$. □

Proposition 3.3 *For any $\mathfrak{R} : \mathbf{U} \to uco(C)$, the following are equivalent:*
(a) *$\mathfrak{R}$ is idempotent;*
(b) *For any $\mathbf{D} \in \mathbf{U}$ such that $\mathfrak{R}(\mathbf{D}) \in \mathbb{K}_{\mathbf{U}}$, $\mathfrak{R}(\mathbf{D}) = \mathfrak{R}_{\mathbb{1}}(\mathfrak{R}(\mathbf{D}))$.*
If $\mathbb{K}_{\mathbf{U}}$ is a (finitely) meet subsemilattice of $uco(C)$ then (a) *is equivalent to:*
(c) *$\mathfrak{R}_{\mathbb{1}}$ is idempotent and for any $\mathbf{D} \in \mathbf{U}$ such that $\mathfrak{R}(\mathbf{D}) \in \mathbb{K}_{\mathbf{U}}$, $\mathfrak{R}(\mathbf{D}) = \mathfrak{R}_{\mathbb{1}}(\sqcap\mathbf{D})$.*

The following example yields a generic and useful pattern of definition for full idempotent refinements.

Example 3.4 Consider any property P of abstract domains, i.e. a subset of the lattice of abstract interpretations $P \subseteq uco(C)$. For any fixed $n \in \mathbb{N}$, define the operator $\mathfrak{R}_P : uco(C)^n \to uco(C)$ as $\mathfrak{R}_P = \lambda\mathbf{D}.\sqcup\{A \in uco(C) \mid A \in P,\ A \sqsubseteq \sqcap\mathbf{D}\}$. Thus, $\mathfrak{R}_P(\mathbf{D})$ is the least common abstraction of all domains that satisfy P and are more concrete (viz. precise) than every $\pi_i(\mathbf{D})$ for $i \in [1, n]$. It is immediate to observe that $\mathfrak{R}_P$ is monotone and reductive. Also, it is easily seen that $\mathfrak{R}_P$ satisfies the condition (c) of Proposition 3.3. Thus, $\mathfrak{R}_P$ always defines a full idempotent refinement. However, in general, $\mathfrak{R}_P(\mathbf{D})$ may not satisfy P. On the other hand, the following characterization holds.

Proposition 3.5 $\forall \mathbf{D}.\ \Re_P(\mathbf{D}) \in P \Leftrightarrow P \in lco(uco(C)) \Rightarrow \Re_P = \lambda \mathbf{D}.P(\sqcap \mathbf{D})$.

Thus, for a property P which is a lower closure, $\Re_P(\mathbf{D})$ is the most abstract domain which satisfies P and is more concrete than every $\pi_i(\mathbf{D})$, or, equivalently, $\Re_P(\mathbf{D})$ is the least extension of $\sqcap \mathbf{D}$ that satisfies P. It is also worth noting that $\Re_P(\mathbf{D})$ is the greatest fixpoint of the equation $X = P(X) \sqcap (\sqcap \mathbf{D})$ in $uco(C)$. □

Note that any unary idempotent refinement $\Re : U \to uco(C)$ such that $\Re(U) \subseteq U$ (we say in this case that $\Re$ is *well-defined* on U) actually is a *lower closure operator* on the poset $\langle U, \sqsubseteq \rangle$, with the order inherited from $uco(C)$, i.e. $\Re \in lco(U)$. In particular, any unary full idempotent refinement $\Re$ is a lower closure on $uco(C)$, i.e. $\Re \in lco(uco(C))$, a case already considered in [9]. Also, for any n-ary full idempotent refinement $\Re : uco(C)^n \to uco(C)$, we have that any unary refinement $\lambda X.\Re(\mathbf{D}[X/i])$ $(i \in [1, n])$ induced by $\Re$ is a lower closure operator on $uco(C)$, as well as the self-refinement $\Re_{\mathbb{1}}$. It would be straightforward, although notationally tedious, to generalize this latter observation to generic n-ary (possibly nonfull) idempotent refinements that satisfy a suitably generalized condition of well-definedness. These observations are fairly important, since unary idempotent refinements inherit all the lattice-theoretic properties of lower closures (see [23] for a few of them). For instance, whenever the domain of definition $\langle U, \sqsubseteq \rangle$ is a complete lattice, we get that these refinements well-defined on U form a complete lattice $\langle lco(U), \sqsubseteq \rangle$ (by a slight abuse of notation, we always use the ordering symbol $\sqsubseteq$ for any kind of closures), where $\Re_1 \sqsubseteq \Re_2$ iff for any $A \in U$, $\Re_1(A) \sqsubseteq \Re_2(A)$ iff the set of abstract domains refined by $\Re_1$ is contained in the set of those refined by $\Re_2$. Thus, analogously to the case of abstract domains, the complete ordering $\sqsubseteq$ between idempotent refinements can be interpreted as a relation of precision among refinement operators, where $\Re_1$ is more precise than $\Re_2$ iff $\Re_1 \sqsubseteq \Re_2$. Moreover, any unary idempotent refinement well-defined on a complete subsemilattice U of $uco(C)$ enjoys the following properties of *compositionality w.r.t. the reduced product and least common abstraction.*

Proposition 3.6 *If U is a complete meet (join) subsemilattice of $uco(C)$, $\Re : U \to U$ is an idempotent refinement, and $\{D_i\}_{i \in I} \subseteq \wp(U)$, then $\Re(\sqcap_{i \in I} D_i) = \Re(\sqcap_{i \in I} \Re(D_i))$ $(\Re(\sqcup_{i \in I} \Re(D_i)) = \sqcup_{i \in I} \Re(D_i))$.*

Reduced Product Refinement. The simplest and probably most familiar example of abstract domain refinement is the reduced product [6], which is the glb in the lattice of abstractions. For simplicity, we consider it as a binary refinement. For any fixed concrete domain C (i.e., any complete lattice), reduced product is obviously an idempotent full refinement $\Re_\sqcap : uco(C) \times uco(C) \to uco(C)$. Thus, the unary refinement induced by $\Re_\sqcap$, i.e. $\lambda X.\ A \sqcap X$ is a lower closure, and for it the properties discussed above hold. It is worth noting that $\Re_\sqcap$ is the simplest instance of the family of refinements defined in Example 3.4, since $\Re_\sqcap$ is $\Re_P$ for the trivial property $P = uco(C)$. Also, $\Re_\sqcap$ is the attribute independent combination of the trivial identity refinements. Reduced product has been successfully applied as a domain refinement in program analysis e.g. in [1, 16, 21].

Disjunctive Completion Refinement. The disjunctive completion [6] enhances an abstract domain so that its disjunction operation (i.e. lub) becomes

precise (as that of the concrete domain). Abstract domains with a precise disjunction (also called disjunctive abstract domains) correspond to additive closure operators. Disjunctive completion can be given as an instance of the general scheme of Example 3.4, where the property P is given by additivity: $P = uco^a(C)$, the subset of $uco(C)$ of additive closures. Hence, the disjunctive completion $\mathfrak{R}_\vee : uco(C) \to uco(C)$ is defined as $\mathfrak{R}_\vee(D) = \sqcup\{A \in uco^a(C) \mid A \sqsubseteq D\}$. Thus, $\mathfrak{R}_\vee$ is an idempotent full refinement. It is easy to observe that $uco^a(C)$ defines a lower closure on $uco(C)$. Then, by Proposition 3.5, $\mathfrak{R}_\vee(D)$ is the most abstract disjunctive domain that is more concrete than D. The disjunctive completion refinement has been applied in program analysis e.g. in [8, 15, 10].

Negative Completion Refinement. Assume the concrete domain C be a complete Boolean algebra. It is easy to verify that if $\rho \in uco^a(C)$ then $\neg\rho = \{\neg x \in C \mid x \in \rho\} \in uco^a(C)$. The negative completion refinement is then defined on disjunctive abstract domains, $\mathfrak{R}_\neg : uco^a(C) \to uco(C)$, as follows: $\mathfrak{R}_\neg(A) = A \sqcap \neg A$. Thus, $\mathfrak{R}_\neg$ lifts a given disjunctive abstract domain A to the reduced product of A with its negative abstract domain, namely to the most abstract domain containing both A and $\neg A$. It is now simple to check that $\mathfrak{R}_\neg : uco^a(C) \to uco(C)$ is an idempotent refinement. It is worth noting that, in general, $\mathfrak{R}_\neg(A)$ may not be disjunctive (i.e., $\mathfrak{R}_\neg$ is not well-defined on $uco^a(C)$).

The Refinement of Completeness. Abstract interpretation is intended to create *sound* approximations of the concrete semantics of programs. If the program semantics is specified as the least fixpoint of a monotone semantic operation $f : C \to C$ on a complete lattice C, then, in the closure operator approach, the soundness criterion for an abstract domain given by $\rho \in uco(C)$ and for an abstract monotone semantic operation $f^\sharp : \rho(C) \to \rho(C)$, is $\forall c \in C.\ \rho(f(c)) \leq f^\sharp(\rho(c))$. This ensures the global soundness of the abstract semantics, i.e. $\rho(lfp(f)) \leq lfp(f^\sharp)$ (cf. [5]). *Completeness* is the dual relation $\forall c \in C.\ f^\sharp(\rho(c)) \leq \rho(f(c))$. Because soundness is always required in abstract interpretation, in the following we abuse terminology and say that $f^\sharp$ is complete for f if $\rho \circ f = f^\sharp \circ \rho$. In this case $\rho(lfp(f)) = lfp(f^\sharp)$. Completeness in abstract interpretation is a quite rare ideal situation, where for a given abstract domain no loss of precision is introduced by abstract semantic operations. Completeness is especially recurrent between (concrete) semantics of programming languages (cf. [2, 7, 11]). Issues of completeness and related notions have also been studied in [17, 19, 20]. Completeness can be made a property of abstract domains, by making this notion independent on the choice for $f^\sharp$. Recall that the best correct approximation of f w.r.t. ρ is given by $\rho \circ f : \rho(C) \to \rho(C)$. Thus, we consider completeness of the best correct approximation: $\rho \in uco(C)$ is *complete* for f if $\rho \circ f = \rho \circ f \circ \rho$. For example, let us consider the canonical 4-point abstract domain $Sign = \{\emptyset, \mathbb{Z}_{<0}, \mathbb{Z}_{>0}, \mathbb{Z}\}$, which is an obvious abstraction of $\langle \wp(\mathbb{Z}), \subseteq \rangle$. It is simple to show that $Sign$ is complete for the monotone operation of integer multiplication $\lambda X.n \cdot X : \wp(\mathbb{Z}) \to \wp(\mathbb{Z})$ (where $n \in \mathbb{Z}$ and $n \cdot X = \{n \cdot m \mid m \in X\}$). On the other hand, $\rho = \{\mathbb{Z}_{>0}, \mathbb{Z}\}$ (with $Sign \sqsubseteq \rho$) is not complete for $\lambda X.\ n \cdot X$ with $n < 0$: In fact, e.g., $\rho(n \cdot \{-3\}) = \mathbb{Z}_{>0}$, but, because $\rho(\{-3\}) = \mathbb{Z}$, $\rho(n \cdot \rho(\{-3\})) = \rho(\mathbb{Z}) = \mathbb{Z}$. The property of completeness for a semantic func-

tion f is therefore given by $\Gamma(f) = \{\rho \in uco(C) \mid \rho \circ f = \rho \circ f \circ \rho\}$. Following the scheme of Example 3.4, we can define an idempotent full refinement of completeness $\Re_{\Gamma(f)} : uco(C) \to uco(C)$ as $\Re_{\Gamma(f)}(\rho) = \sqcup\{\eta \in uco(C) \mid \eta \in \Gamma(f),\ \eta \sqsubseteq \rho\}$.

Theorem 3.7 *If f is continuous then $\Gamma(f) \in lco(uco(C))$.*

Thus, by Proposition 3.5, we have that for a continuous f, $\Re_{\Gamma(f)}(D)$ actually is the (unique) most abstract domain which includes D and is complete for f. For instance, it is possible to check that for $n < 0$, $\Re_{\Gamma(\lambda X.n \cdot X)}(\{\mathbb{Z}_{>0}, \mathbb{Z}\}) = Sign$.

4 Abstract Domain Compressors

We have introduced the notion of abstract domain refinement as a formalization (and generalization) of many existing operators devoted to enhance the expressiveness of abstract domains. However, no operator performing a dual action of simplification on abstract domains has been proposed up till now. We now formalize the idea of a simplifying operator that gives as input to a fixed refinement the simplest domains (i.e. most abstract) which can be object of that refinement. Let $\Re : \mathbf{U} \to uco(C)$ be a (possibly nonidempotent) refinement. Define $\Re_k^- : \mathbf{U} \to uco(C)$, $k \in [1, n]$, as $\Re_k^- = \lambda \mathbf{D}.\sqcup\{A \in \pi_k(\mathbf{U}) \mid \Re(\mathbf{D}[A/k]) = \Re(\mathbf{D})\}$. For $\mathbf{D} \in \mathbf{U}$, $\Re_k^-(\mathbf{D})$ is the least common abstraction of all domains in $\pi_k(\mathbf{U})$ that, when substituted to $\pi_k(\mathbf{D})$ as k-th input for $\Re$, do not change the output.

Definition 4.1 $\Re_k^-(\mathbf{D})$ is the *k-th optimal basis* of $\mathbf{D} \in \mathbf{U}$ for $\Re$ if $\Re_k^-(\mathbf{D}) \in \pi_k(\mathbf{U})$ and $\Re(\mathbf{D}) = \Re(\mathbf{D}[\Re_k^-(\mathbf{D})/k])$. The refinement $\Re$ is *k-invertible* (or *admits the k-th inverse*) on $V \subseteq \pi_k(\mathbf{U})$ if for all $\mathbf{D} \in \mathbf{U}[V/k]$, $\Re_k^-(\mathbf{D})$ is the k-th optimal basis of $\mathbf{D}$ for $\Re$. When $\Re$ is k-invertible, the map $\Re_k^- : \mathbf{U}[V/k] \to \pi_k(\mathbf{U})$ is called the *k-th compressor* for $\Re$. □

Note that if the domain of definition $V \subseteq \pi_k(\mathbf{U})$ of the k-th compressor $\Re_k^-$ is a complete join subsemilattice of $uco(C)$, then the condition $\Re_k^-(\mathbf{D}) \in \pi_k(\mathbf{U})$ in the above definition can be omitted. For $K \subseteq [1, n]$, we say that $\Re$ is K-invertible on a $|K|$-tuple $\mathbf{V}$, where $\forall i \in K.\ \pi_i(\mathbf{V}) \subseteq \pi_i(\mathbf{U})$, if it is k-invertible on $\pi_k(\mathbf{V})$, for any $k \in K$. In particular, $\Re$ is *fully invertible* on $\mathbf{V} \subseteq \mathbf{U}$ if it is $[1, n]$-invertible on $\mathbf{V}$. For the simpler case of a unary refinement $\Re : U \to uco(C)$, we have that $\Re^- : U \to uco(C)$ is defined as $\Re^-(D) = \sqcup\{A \in U \mid \Re(A) = \Re(D)\}$, and $\Re$ is invertible on $V \subseteq U$ iff for any $D \in V$, $\Re(\Re^-(D)) = \Re(D)$. It is simple to observe that the above definition of k-invertibility can be formulated by using the unary refinements induced by a (n-ary) refinement. More precisely, if $\Re : \mathbf{U} \to uco(C)$ is a refinement, then we have already seen that for any $k \in [1, n]$ and $\pi_i(\mathbf{D}) \in \pi_i(\mathbf{U})$ $(i \neq k)$, $\lambda X.\Re(\mathbf{D}[X/k]) : \pi_k(\mathbf{U}) \to uco(C)$ is a unary refinement. It is then easily seen that $\Re$ is k-invertible in $V \subseteq \pi_k(\mathbf{U})$ iff $\lambda X.\Re(\mathbf{D}[X/k])$ is (1-)invertible on $V \subseteq \pi_k(\mathbf{U})$. In this case, for the compressor $(\lambda X.\Re(\mathbf{D}[X/k]))^- : V \to \pi_k(\mathbf{U})$ and the k-th compressor $\Re_k^-$ of $\Re$, the following mutual equality result holds: $\forall D \in V.\ (\lambda X.\Re(\mathbf{D}[X/k]))^-(D) = \Re_k^-(\mathbf{D})$.

Not all domain refinements are invertible in a satisfactory way. An example is provided by the negative completion refinement $\Re_\neg$ of Sect. 3. In fact, as observed in [9], the optimal basis of the domain *Sign* (in Sect. 3) for $\Re_\neg$ does not exist.

Since *Sign* enjoys all most important lattice-theoretic properties, this means that $\mathfrak{R}_{\neg}$ is not invertible on any really significant class of abstract domains.

As the following result says, compressors relative to idempotent refinements are extensive and idempotent.

Proposition 4.2 *If* $\mathfrak{R} : \mathbf{U} \to uco(C)$ *is idempotent and* k*-invertible in* V*, then the compressor* $\mathfrak{R}_k^- : \mathbf{U}[V/k] \to \pi_k(\mathbf{U})$ *is extensive (i.e.* $\pi_k(\mathbf{D}) \sqsubseteq \mathfrak{R}_k^-(\mathbf{D})$*) and idempotent (i.e.* $\mathfrak{R}_k^-(\mathbf{D}) \in V \Rightarrow \mathfrak{R}_k^-(\mathbf{D}[\mathfrak{R}_k^-(\mathbf{D})/k]) = \mathfrak{R}_k^-(\mathbf{D})$*).*

In general, compressors are neither monotone nor antimonotone: [14] proves that the least disjunctive basis operator is neither monotone nor antimonotone, and later we will show that the least disjunctive basis is the compressor relative to the disjunctive completion refinement. On the other hand, as expected, a compressor applied to a refined domain performs no further simplification.

Proposition 4.3 *If* $\mathfrak{R} : \mathbf{U} \to uco(C)$ *is idempotent and* k*-invertible in* V *then for any* $\mathbf{D} \in \mathbf{U}[V/k]$ *such that* $\mathfrak{R}(\mathbf{D}) \in V$*,* $\mathfrak{R}_k^-(\mathbf{D}[\mathfrak{R}(\mathbf{D})/k]) = \mathfrak{R}_k^-(\mathbf{D})$*.*

An n-ary refinement $\mathfrak{R} : \mathbf{U} \to uco(C)$ is *commutative* if for any permutation τ of $\{1, \ldots, n\}$, $\mathfrak{R}(\pi_{\tau(1)}(\mathbf{D}), \ldots, \pi_{\tau(n)}(\mathbf{D})) = \mathfrak{R}(\mathbf{D})$ holds. For instance, the reduced product refinement $\mathfrak{R}_{\sqcap}$ is obviously commutative as well as any attribute independent refinement. For commutative refinements, the following result holds (this result admits a straightforward, although notationally tedious, generalization for generic K-commutativity and invertibility).

Proposition 4.4 *If* $\mathfrak{R} : \mathbf{U} \to uco(C)$ *is a (possibly nonidempotent) commutative refinement and* $k \in [1, n]$ *then,* $\mathfrak{R}$ *is fully invertible on* $\mathbf{V} \subseteq \mathbf{U}$ *iff* $\mathfrak{R}$ *is* k*-invertible on* $\pi_k(\mathbf{V})$ *iff for all* $\mathbf{D} \in \mathbf{V}$*,* $\lambda X.\mathfrak{R}(\mathbf{D}[X/k]) : \pi_k(\mathbf{U}) \to uco(C)$ *is (1-)invertible on* $\pi_k(\mathbf{V})$*.*

Not all refinements are commutative. Examples of noncommutative refinements are reduced power [6], dependencies [13], and tensor product [18]. Due to lack of space, we do not formalize these operators as refinements. In the following, we show how the results in [3, 12, 14] on complementation and least disjunctive basis of abstract domains, actually permit to define the compressors relative to reduced product and disjunctive completion respectively. These results also suggest a generalization towards a general pattern of invertible refinements.

The Inverse of Reduced Product. Since $\mathfrak{R}_{\sqcap}$ is commutative, by Proposition 4.4, $\mathfrak{R}_{\sqcap}$ is fully invertible on some $V \times V \subseteq uco(C)^2$ iff for any $D \in V$, $\lambda X.(D \sqcap X)$ is invertible on V. As recalled in Sect. 2, for any meet-continuous complete lattice C and $D \in uco(C)$, one can define the complement abstract domain $C \sim D$. Moreover, it is immediate to note that, for any complete lattice C, if $D_1, D_2 \in uco(C)$ satisfy the ascending chain condition (to be ACC, for short; DCC is dual) then $D_1 \sqcap D_2$ is ACC as well, and hence meet-continuous. These observations directly imply that we can invert the reduced product on the ACC abstractions of any concrete domain C (i.e. a plain complete lattice). Let us define $ACC(C) = \{D \in uco(C) \mid D \text{ is ACC}\}$, for any complete lattice C.

Theorem 4.5 *If* $D \in ACC(C)$ *then* $\lambda X.D \sqcap X$ *is invertible on* $ACC(C)$*, and the corresponding compressor* $(\lambda X.D \sqcap X)^- : ACC(C) \to uco(C)$ *is defined as* $(\lambda X.D \sqcap X)^-(E) = (D \sqcap E) \sim D$*.*

Thus, $\Re_\sqcap$ is fully invertible on $ACC(C) \times ACC(C)$. For instance, if $D_1, D_2 \in ACC(C)$, we have that the first compressor is $(\Re_\sqcap)_1^-(D_1, D_2) = (D_1 \sqcap D_2) \sim D_2$.

The Inverse of Disjunctive Completion. Giacobazzi and Ranzato defined and studied in [14] the operator of least disjunctive basis on abstract domains, that corresponds exactly to the compressor for the disjunctive completion refinement. Hence, the results in [14] can be reformulated as follows.

Theorem 4.6

(i) *If C is co-algebraic completely distributive then $\Re_\vee$ is invertible on all $uco(C)$.*
(ii) *If C is distributive then $\Re_\vee$ is invertible on $\{A \in uco(C) \mid A \text{ is finite}\}$.*

Compressing Lower and Upper Refinements. Define an *upper (lower) improvement* on C as any map $\mathcal{I} : \wp(C) \to \wp(C)$ such that $\forall S \in \wp(C). \forall s \in S. \forall s' \in \mathcal{I}(S).\ s \leq s'$ $(s' \leq s)$. Glb and lub are obvious examples of lower and upper improvements. We prove that upper and lower improvements induce invertible refinements in a natural way. This provides a general pattern for defining new invertible refinements. For an upper (lower) improvement $\mathcal{I}$ on C, define the corresponding *upper (lower) set-refinement* $\Re^{\mathcal{I}} : \wp(C) \to \wp(C)$ as: $\Re^{\mathcal{I}}(X) = X \cup (\cup_{S \subseteq X} \mathcal{I}(S))$. It turns out that $\Re^{\mathcal{I}}$ is a lower closure on $\langle \wp(C), \supseteq \rangle$. However, in general, for a closure $\rho \in uco(C)$, $\Re^{\mathcal{I}}(\rho)$ may not be in $uco(C)$. But, when a unary full idempotent refinement $\Re \in lco(uco(C))$ is the restriction on $uco(C)$ of an upper (lower) set-refinement, i.e. there exists an upper (lower) improvement $\mathcal{I}$ on C such that $\Re = \Re^{\mathcal{I}}_{\restriction uco(C)}$ (in this case, we call $\Re$ an *upper (lower) refinement*), the following general theorem of inversion for $\Re$ holds.

Theorem 4.7 *If C is a complete lattice satisfying the DCC (ACC), then any upper (lower) refinement $\Re \in lco(uco(C))$ is invertible on all $uco(C)$.*

For instance, if C is distributive and $\mathcal{I} = \vee$, we get for free the inversion of disjunctive completion of Theorem 4.6 (ii). By Proposition 4.4, the attribute independent refinement induced by a family of upper or lower refinements is invertible under suitable hypotheses derived by Theorem 4.7. By this last observation, it would be possible (but we omit the details) to derive as a consequence of Theorem 4.7 the result of inversion for the reduced product of Theorem 4.5.

Minimal $\Re$-decompositions. For a given refinement $\Re : \mathbf{U} \to uco(C)$ of arity $n > 1$, we say that $\mathbf{D} \in \mathbf{U}$ is a *$\Re$-decomposition* of $D \in uco(C)$, if $D = \Re(\mathbf{D})$. If $\mathbf{D}, \mathbf{E} \in \mathbf{U}$ are two $\Re$-decompositions of D then $\mathbf{D}$ is *better* than $\mathbf{E}$ if $\mathbf{E} \sqsubseteq \mathbf{D}$ componentwise.[1] The intended meaning is that $\mathbf{D}$ is better than $\mathbf{E}$ because it is a less costly decomposition (in particular, $\sum_{i=1}^n |\pi_i(\mathbf{D})| \leq \sum_{i=1}^n |\pi_i(\mathbf{E})|$). Obviously, this relation induces a partial ordering between $\Re$-decompositions of D, but, in general, optimal (i.e. least) $\Re$-decompositions for this order do not exist. For instance, $\langle D, \{\top\} \rangle$ and $\langle \{\top\}, D \rangle$ are uncomparable minimal $\Re_\sqcap$-decompositions of D. It is easy to see that, if $\Re$ is idempotent and fully invertible on $\mathbf{V} \subseteq \mathbf{U}$ and $\mathbf{D}$ is a $\Re$-decomposition of D, then for any $k \in [1, n]$ the tuple $\mathbf{D}\big[\Re_k^-(\mathbf{D})/k\big]$ $(*)$ is still a $\Re$-decomposition of D which is better than $\mathbf{D}$, and

[1] For commutative refinements, both this definition and the successive development would identify decompositions up to permutation – however, we omit the details.

that $\mathbf{D}$ is a minimal $\mathfrak{R}$-decomposition of D iff $\forall k \in [1,n].\ \pi_k(\mathbf{D}) = \mathfrak{R}_k^-(\mathbf{D})$. Thus, each $\mathfrak{R}$-decomposition can be improved by iterating the above step $(*)$ as shown in the following nondeterministic function $\mathfrak{R}$-min, where *choose* selects an arbitrary element from its input set.

```
fun ℜ-min (D:array[1,n] of domains)
J := {1,...,n};
repeat
   k := choose(J);
   J := J \ {k};
   D := D[ℜ_k^-(D)/k]
until J = ∅
output D
```

Theorem 4.8 *Let $\mathfrak{R}$ be an idempotent and fully invertible refinement on $\mathbf{V}$. If, for any $k \in [1,n]$, $\mathfrak{R}_k^-$ is anti-monotone then for any $\mathbf{D} \in \mathbf{V}$, $\mathfrak{R}$-min$(\mathbf{D})$ is a minimal $\mathfrak{R}$-decomposition of $\mathfrak{R}(\mathbf{D})$.*

Note that, for a $\mathfrak{R}$-decomposition $\mathbf{D}$ of D, we can get at most $n!$ different minimal $\mathfrak{R}$-decompositions of D. For instance, if $\mathfrak{R}(D_1, D_2) = D$ then $\langle \mathfrak{R}_1^-(D_1, \mathfrak{R}_2^-(D_1, D_2)), \mathfrak{R}_2^-(D_1, D_2)\rangle$ is a minimal $\mathfrak{R}$-decomposition of D. Theorem 4.8 generalizes the results of [3, Sect. 4], since the compressor relative to reduced product is anti-monotone (cf. [3]).

5 A Relation of Adjunction between Refinements and Compressors

Assume that $\mathfrak{R}$ is an idempotent n-ary refinement $\mathfrak{R} : \mathbf{U} \to \mathit{uco}(C)$ that is k-invertible on $V \subseteq \pi_k(\mathbf{U}) \subseteq \mathit{uco}(C)$, for some $k \in [1,n]$. We saw in Sect. 4 that any (k-th) unary refinement $\lambda X.\mathfrak{R}(\mathbf{D}[X/k]) : \pi_k(\mathbf{U}) \to \mathit{uco}(C)$ induced by $\mathfrak{R}$ is invertible in V, and the corresponding compressor (of type $V \to \pi_k(\mathbf{U})$) is defined as $(\lambda X.\mathfrak{R}(\mathbf{D}[X/k]))^- = \lambda X \in V.\mathfrak{R}_k^-(\mathbf{D}[X/k])$. In general, the refinement $\lambda X.\mathfrak{R}(\mathbf{D}[X/k])$ and the relative compressor $\lambda X.\mathfrak{R}_k^-(\mathbf{D}[X/k])$ do not constitute an adjunction on the poset of domains $\langle V, \sqsubseteq\rangle$ of invertibility, i.e. for all $A \in \pi_k(\mathbf{U})$ and $B \in V$, $\mathfrak{R}(\mathbf{D}[A/k]) \sqsubseteq B \Leftrightarrow A \sqsubseteq \mathfrak{R}_k^-(\mathbf{D}[B/k]))$ may not hold. This is due to the fact that compressors, in general, are not monotone, as observed after Proposition 4.2. Since, by Proposition 4.2, compressors are idempotent and extensive, this also implies that compressors $\lambda X.\mathfrak{R}_k^-(\mathbf{D}[X/k])$, well-defined on V, are not upper closures on $\langle V, \sqsubseteq\rangle$, as instead we would expect by viewing compressors as inverses of refinements.

We solve this asymmetry between abstract domain refinements and compressors by modifying the standard ordering $\sqsubseteq$ of precision between domains, so as to keep into account the rôle of $\mathfrak{R}$. We maintain the above scenario and also suppose that the refinement $\lambda X.\mathfrak{R}(\mathbf{D}[X/k])$ is well-defined in $\pi_k(\mathbf{U})$, namely for any $D \in \pi_k(\mathbf{U})$, $\mathfrak{R}(\mathbf{D}[D/k]) \in \pi_k(\mathbf{U})$, and that $\langle \pi_k(\mathbf{U}), \sqsubseteq\rangle$ is a complete sublattice of $\langle \mathit{uco}(C), \sqsubseteq\rangle$. These hypotheses imply that $\lambda X.\mathfrak{R}(\mathbf{D}[X/k])$ is a lower closure on the complete lattice $\langle \pi_k(\mathbf{U}), \sqsubseteq\rangle$. Then, we define the following relation $\sqsubseteq^{\mathfrak{R}}$ (that actually depends also on the fixed arguments $\pi_i(\mathbf{D})$, $i \neq k$) on $\pi_k(\mathbf{U})$:

$$A \sqsubseteq^{\mathfrak{R}} B \text{ iff } \mathfrak{R}(\mathbf{D}[A/k]) \sqsubseteq \mathfrak{R}(\mathbf{D}[B/k]) \ \&\ (\mathfrak{R}(\mathbf{D}[B/k]) \sqsubseteq \mathfrak{R}(\mathbf{D}[A/k]) \Rightarrow A \sqsubseteq B).$$

Theorem 5.1 *$\langle \pi_k(\mathbf{U}), \sqsubseteq^{\mathfrak{R}}\rangle$ is a complete lattice.*

Note that $A \sqsubseteq B \Rightarrow A \sqsubseteq^{\mathfrak{R}} B$. Thus, we call $\sqsubseteq^{\mathfrak{R}}$ the *lifting of* $\sqsubseteq$ *via* $\mathfrak{R}$. This lifted complete partial order reflects precisely the relative precision of domains with respect to the refinement $\lambda X.\mathfrak{R}(\mathbf{D}[X/k])$: A is more precise than B in the lifted order if the refinement of A is more precise than the refinement of B in the

standard sense and, when they are the same (i.e. $(\Re(\mathbf{D}[B/k]) = \Re(\mathbf{D}[A/k]))$, then A contains more information. For this ordering $\sqsubseteq^{\Re}$, we get back a relation of adjunction between the invertible refinement and its compressor.

Theorem 5.2 $\forall A \in \pi_k(\mathbf{U}), B \in V.\ \Re(\mathbf{D}[A/k]) \sqsubseteq^{\Re} B \Leftrightarrow A \sqsubseteq^{\Re} \Re_k^-(\mathbf{D}[B/k]))$.

As a consequence, $\lambda X.\ \Re_k^-(\mathbf{D}[X/k])$ is an upper closure operator on $\langle V, \sqsubseteq^{\Re}\rangle$ (provided it is well-defined on V). For example, we get an adjunction between reduced product and complementation w.r.t. the lifted order. For any complete lattice C and $D \in uco(C)$, the lifted order on $uco(C)$ is defined as follows: For all $A, B \in uco(C)$, $A \sqsubseteq^{\sqcap} B$ iff $D \sqcap A \sqsubseteq D \sqcap B\ \&\ (D \sqcap B \sqsubseteq D \sqcap A \Rightarrow A \sqsubseteq B)$. Hence, the adjunction between refinement (reduced product) and compressor (complementation) is the following: For any $A \in uco(C)$ and $B \in ACC(C)$, $D \sqcap A \sqsubseteq^{\sqcap} B \Leftrightarrow A \sqsubseteq^{\sqcap} (D \sqcap B) \sim D$.

Acknowledgments. We are grateful to Francesca Scozzari for her contribution to Theorem 3.7 and to one anonymous referee for many helpful suggestions.

References

1. M. Codish, A. Mulkers, M. Bruynooghe, M. García de la Banda, and M. Hermenegildo. Improving abstract interpretations by combining domains. *ACM TOPLAS*, 17(1):28–44, 1995.
2. M. Comini and G. Levi. An algebraic theory of observables. In *Proc. ILPS'94*, pp. 172–186, 1994.
3. A. Cortesi, G. Filé, R. Giacobazzi, C. Palamidessi, and F. Ranzato. Complementation in abstract interpretation. *ACM TOPLAS*, 19(1):7–47, 1997.
4. A. Cortesi, B. Le Charlier, and P. Van Hentenryck. Combinations of abstract domains for logic programming. In *Proc. POPL'94*, pp. 227–239, 1994.
5. P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proc. POPL'77*, pp. 238-252, 1977.
6. P. Cousot and R. Cousot. Systematic design of program analysis frameworks. In *Proc. POPL'79*, pp. 269–282, 1979.
7. P. Cousot and R. Cousot. Inductive definitions, semantics and abstract interpretation. In *Proc. POPL'92*, pp. 83–94, 1992.
8. P. Cousot and R. Cousot. Higher-order abstract interpretation (and application to comportment analysis generalizing strictness, termination, projection and PER analysis of functional languages). In *Proc. IEEE ICCL'94*, pp. 95–112, 1994.
9. G. Filé, R. Giacobazzi, and F. Ranzato. A unifying view of abstract domain design. *ACM Comput. Surv.*, 28(2):333–336, 1996.
10. G. Filé and F. Ranzato. Improving abstract interpretations by systematic lifting to the powerset. In *Proc. ILPS'94*, pp. 655–669, 1994.
11. R. Giacobazzi. "Optimal" collecting semantics for analysis in a hierarchy of logic program semantics. In *Proc. STACS'96*, LNCS 1046, pp. 503-514, 1996.
12. R. Giacobazzi, C. Palamidessi, and F. Ranzato. Weak relative pseudo-complements of closure operators. *Algebra Universalis*, 36(3):405-412, 1996.
13. R. Giacobazzi and F. Ranzato. Functional dependencies and Moore-set completions of abstract interpretations and semantics. In *Proc. ILPS'95*, pp. 321–335, 1995.
14. R. Giacobazzi and F. Ranzato. Optimal domains for disjunctive abstract interpretation. To appear in *Sci. Comput. Program.* Preliminary version in LNCS 1058, pp. 141–155, 1996.
15. T.P. Jensen. Disjunctive strictness analysis. In *Proc. LICS'92*, pp. 174–185. 1992.
16. K. Muthukumar and M. Hermenegildo. Combined determination of sharing and freeness of program variables through abstract interpretation. In *Proc. ICLP'91*, pp. 49–63, 1991.
17. A. Mycroft. Completeness and predicate-based abstract interpretation. In *Proc. PEPM'93*.
18. F. Nielson. Tensor products generalize the relational data flow analysis method. In *Proc. 4th Hungarian Comput. Sci. Conf.*, pp. 211–225, 1985.
19. U. Reddy and S. Kamin. On the power of abstract interpretation. In *Proc. IEEE ICCL'92*, 1992.
20. R.C. Sekar, P. Mishra, and I.V. Ramakrishnan. On the power and limitation of strictness analysis. To appear in *J. ACM*. Preliminary version in *Proc. POPL'91*, pp. 37–48, 1991.
21. R. Sundararajan and J. Conery. An abstract interpretation scheme for groundness, freeness, and sharing analysis of logic programs. In *Proc. FST&TCS'92*, LNCS 652, pp. 203–216, 1992.
22. K. Yi and W.L. Harrison. Automatic generation and management of interprocedural program analyses. In *Proc. POPL'93*, pp. 246–259, 1993.
23. M. Ward. The closure operators of a lattice. *Ann. Math.*, 43(2):191–196, 1942.