Galois Connection Based Abstract Interpretations for Strictness Analysis

Patrick Cousot[1] and Radhia Cousot[2]

[1] LIENS, École Normale Supérieure
45, rue d'Ulm
75230 Paris cedex 05 (France)
cousot@dmi.ens.fr

[2] LIX, École Polytechnique
91128 Palaiseau cedex (France)
radhia@polytechnique.fr

Abstract. The abstract interpretation framework based upon the approximation of a fixpoint collecting semantics using Galois connections and widening/narrowing operators on complete lattices [CC77a, CC79b] has been considered difficult to apply to Mycroft's strictness analysis [Myc80, Myc81] for which denotational semantics was though to be more adequate (because non-termination has to be taken into account), see e.g. [AH87], page 25.

Considering a non-deterministic first-order language, we show, contrary to expectation, and using the classical Galois connection-based framework, that Mycroft strictness analysis algorithm is the abstract interpretation of a relational semantics (a big-steps operational semantics including non-termination which can be defined in $G^{\infty}SOS$ either in rule-based or fixpoint style by induction on the syntax of programs [CC92])

An improved version of Johnsson's algorithm [Joh81] is obtained by a subsequent dependence-free abstraction of Mycroft's dependence-sensitive method.

Finally, a compromise between the precision of dependence-sensitive algorithms and the efficiency of dependence-free algorithms is suggested using widening operators.

Keywords: Abstract interpretation; Relational semantics; Strictness analysis; Galois connection; Dependence-free and dependence-sensitive analysis; Widening.

Abstract interpretation [CC77a, CC79b] is a method for designing approximate semantics of programs which can be used to gather information about programs in order to provide safe answers to questions about their run-time behaviours. These semantics can be used to design manual proof methods or to specify automatic program analyzers. When the semantic analysis of programs is to be automated, the answers can only be partial or approximate (that is correct/safe/sound but incomplete) since questions such as termination for all input data are undecidable.

By considering that non-terminating and erroneous behaviors are equivalent, *call-by-need* can be replaced by *call-by-value* in functional programs either whenever the actual argument is always evaluated at least once in the function body or upon the later recursive calls (so that if the evaluation of the actual argument does not terminate or is erroneous then so does the function call) or whenever the function call does

not terminate or is erroneous (whether the actual argument is evaluated or not). Alan Mycroft's *strictness analysis* [Myc80, Myc81] is an abstract interpretation designed to recognize these situations. Observe the importance of taking non-termination $\perp$ into account: left-to-right addition is strict in its second argument since it does not terminate either when evaluation of its first argument is guaranteed not to terminate (in which case the second argument is not needed) or the evaluation of its first argument does terminate but that of the second does not. In the classical definition of strictness, errors Ω must also be identified with non-termination $\perp$. Otherwise left-to-right addition would not be strict in its second argument since $\texttt{true} + \perp = (1/0) + \perp = \Omega$.

Given a recursive function declaration $f(x, y) \equiv e$, Alan Mycroft has designed an abstract function $f^\sharp(x, y) \equiv e^\sharp$ on the abstract domain $\{0, 1\}$ such that:

- if $f^\sharp(0, b) = 0$ for $b \in \{0, 1\}$ then f is strict in x;
- if $f^\sharp(a, 0) = 0$ for $a \in \{0, 1\}$ then f is strict in y;
- if $f^\sharp(1, 1) = 0$ then f never terminates;
- if $f^\sharp(a, b) = 1$ for $a, b \in \{0, 1\}$ then the abstraction is too approximate and no conclusion can be drawn.

The abstract value $e^\sharp$ of an expression e is determined as follows:

- If the expression e is reduced to a constant k then its abstract value $e^\sharp$ is 1 which is the best possible approximation to the fact that it is an integer.
- If the expression e is reduced to a formal parameter x then its evaluation is erroneous or does not terminate if and only if the evaluation of the actual argument is erroneous or does not terminate so that its abstract value $e^\sharp$ is x which denotes the value of the abstract actual argument.
- If the expression e is a basic operation such as $e_1 + e_2$ (always needing its two arguments e_1 and e_2) and the abstract values $e_1^\sharp$ and $e_2^\sharp$ of expressions e_1 and e_2 are obtained recursively, then $(e_1 + e_2)^\sharp = (e_1^\sharp \wedge e_2^\sharp)$ where $(0 \wedge 0) = (0 \wedge 1) = (1 \wedge 0) = 0$ and $(1 \wedge 1) = 1$ since the evaluation of $e_1 + e_2$ is erroneous or does not terminate when that of e_1 or that of e_2 is erroneous or does not terminate, which we conclude from $e_1^\sharp = 0$ or $e_2^\sharp = 0$.
- The evaluation of a conditional $e \equiv (e_1 \rightarrow e_2, e_3)$ is erroneous or does not terminate when the evaluation of the condition e_1 is erroneous (for example non-boolean) or does not terminate so that $e_1^\sharp = 0$ implies $e^\sharp = 0$. If the evaluation of the condition e_1 terminates and the returned boolean value is unknown (so that $e_1^\sharp = 1$), then erroneous termination or non-termination of e is guaranteed only if those of both e_2 and e_3 are guaranteed so that $e^\sharp = 0$ only if $e_2^\sharp = 0$ and $e_3^\sharp = 0$. Therefore $e^\sharp$ can be defined as $(e_1^\sharp \wedge (e_2^\sharp \vee e_3^\sharp))$ where $(0 \vee 0) = 0$ and $(0 \vee 1) = (1 \vee 0) = (1 \vee 1) = 1$.

For the traditional addition example:

$$\texttt{f(x, y)} \equiv \texttt{((x = 0)} \rightarrow \texttt{y, (1 + f(x - 1, y)))}$$

we have:

$$f^\sharp(x, y) = (x \wedge 1) \wedge (y \vee (1 \wedge f^\sharp(x \wedge 1, y)))$$

After simplifications (such as $(x \wedge 1) = (1 \wedge x) = x$, $(x \wedge 0) = (0 \wedge x) = 0$, etc.), we get:

$$f^\sharp(x, y) = x \wedge (y \vee f^\sharp(x, y))$$

so that:

$$f^\sharp(0,0) = 0 \qquad f^\sharp(1,0) = f^\sharp(1,0) \qquad f^\sharp(0,1) = 0 \qquad f^\sharp(1,1) = 1$$

Constraint $f^\sharp(0,1) = 0$ shows that f is strict in its first parameter. Constraint $f^\sharp(1,0) = f^\sharp(1,0)$ is satisfied by $f^\sharp(1,0) = 0$ proving that f is also strict in its second parameter. It is also satisfied by $f^\sharp(1,0) = 1$ which states that the call may terminate or not, which is compatible with but less precise than $f^\sharp(1,0) = 0$ stating that the call is erroneous or does not terminate.

Mycroft's strictness analysis is dependence-sensitive[3] in that for example we might have $f^\sharp(0,0) = 0$ whereas $f^\sharp(0,1) = 1$ and $f^\sharp(1,0) = 1$ so that f is jointly strict in its two parameters without being strict in any of them. Thomas Johnsson [Joh81], followed by John Hugues [Hug88], proposed a dependence-free strictness analysis method that would not discover this property. However it is more efficient and can discover useful information when no dependence relationship between parameters is needed. For the above addition example, the equations:

$$f^{\sharp}[\mathtt{x}](x) = x \qquad f^{\sharp}[\mathtt{y}](y) = y \vee f^{\sharp}[\mathtt{y}](y)$$

are such that $f^{\sharp}[\mathtt{x}](0) = 0$ and $f^{\sharp}[\mathtt{y}](0) = 0$ proving that f is strict in its two parameters.

The purpose of abstract interpretation is to prove the safeness of such program analysis methods with respect to a semantics, or better to formally design them by approximation of the semantics of programs.

1 Principles of Abstract Interpretation

Abstract interpretation, when reduced to its basic principles introduced in [CC77a], can be understood as the approximation of a *collecting semantics*[4].

– The collecting semantics collects a class of properties of interest about a program[5]. It is assumed to be defined as the least fixpoint $\mathrm{lfp}^{\sqsubseteq}_{\perp} F$ of a monotonic operator F where the set of *concrete properties*:

$$\mathcal{F}(\sqsubseteq, \perp, \top, \sqcup, \sqcap) \text{ is a complete lattice.} \tag{1}$$

$\sqsubseteq$ is called the *computational ordering*. The definition of the operator F is by structural induction on the syntax of the program. F is assumed to be monotonic[6] (but,

[3] In order to later avoid the confusion between "relational semantics" and "relational analysis" in the sense of S.S. Muchnick and N.D. Jones [JM81], we use "dependence-sensitive" for "relational" and "dependence-free" for "independent attribute" analysis.

[4] introduced in [CC77a] under the name *static semantics*, and renamed "collecting" by F. Nielson.

[5] invariance properties in [CC77a].

[6] $S \stackrel{\mathrm{m}}{\longmapsto} T$ denotes the set of total functions $\varphi \in S \mapsto T$ on posets $S(\sqsubseteq)$ and $T(\preceq)$ which are *monotonic*, that is, $\forall x, y \in S : x \sqsubseteq y \Rightarrow \varphi x \preceq \varphi y$.

for simplicity, could be assumed to be continuous[7,8]):

$$F \in \mathcal{F} \stackrel{\mathrm{m}\sqsubseteq}{\longmapsto} \mathcal{F} \tag{2}$$

Proposition 1 Tarski's fixpoint theorem [CC79a]. *Let $\mathbb{O}$ be the class of ordinals*[9]*. Define $F^0(X) = X$, $F^{\lambda+1}(X) = F(F^\lambda(X))$ for successor ordinals $\lambda+1$ and $F^\lambda(X) = \bigsqcup_{\beta<\lambda} F^\beta(X)$ for limit ordinals λ. (1) and (2) imply that the transfinite iteration sequence $F^\lambda(\bot)$, $\lambda \in \mathbb{O}$ is well-defined, increasing for $\sqsubseteq$ and ultimately stationary $\left(\exists \epsilon \in \mathbb{O} : \forall \lambda \geq \epsilon : F^\lambda(\bot) = F^\epsilon(\bot)\right)$. Its limit:*

$$\mathrm{lfp}^{\sqsubseteq}_{\bot} F \;=\; \bigsqcup_{\lambda \in \mathbb{O}} F^\lambda(\bot) \tag{3}$$

is the least fixpoint of F for $\sqsubseteq$. In particular if $F \in \mathcal{F} \stackrel{\mathrm{c}\sqcup}{\longmapsto} \mathcal{F}$ is continuous then:

$$\mathrm{lfp}^{\sqsubseteq}_{\bot} F \;=\; \bigsqcup_{n \in \mathbb{N}} F^n(\bot) \tag{4}$$

– Since the collecting semantics is not effectively computable and sometimes not even computer representable, approximations must be considered. The notion of approximation is formalized using an *approximation relation*: $\varphi \leq \phi$ means that property ϕ safely approximates φ. To effectively compute such an approximation of $\mathrm{lfp}^{\sqsubseteq}_{\bot} F$, we replace concrete properties in $\mathcal{F}$ by more crude abstract properties in a well-chosen set $\mathcal{F}^\sharp$. Concrete properties $\varphi \in \mathcal{F}$ and abstract properties $\varphi^\sharp \in \mathcal{F}^\sharp$ should be related by a "semantic function" $\gamma \in \mathcal{F}^\sharp \mapsto \mathcal{F}$. The idea is then to mimic the computation of $\mathrm{lfp}^{\sqsubseteq}_{\bot} F = \sqcup_{\lambda \in \mathbb{O}} F^\lambda(\bot)$ within $\mathcal{F}^\sharp$ as $\sqcup^\sharp_{\lambda \in \mathbb{O}} F^{\sharp\lambda}(\bot^\sharp)$ by defining $\bot^\sharp$, $F^\sharp$ and $\sqcup^\sharp$, such that:

$$\bot \leq \gamma(\bot^\sharp) \tag{5}$$

for all $\varphi \in \mathcal{F}$ and $\phi^\sharp \in \mathcal{F}^\sharp$:

$$\varphi \leq \gamma(\phi^\sharp) \;\Rightarrow\; F(\varphi) \leq \gamma(F^\sharp(\phi^\sharp)) \tag{6}$$

and for all $\{\varphi_\beta \mid \beta \in \mathbb{O}\} \subseteq \mathcal{F}$, $\{\phi^\sharp_\beta \mid \beta \in \mathbb{O}\} \subseteq \mathcal{F}^\sharp$ and all limit ordinals λ (ω only if F is continuous):

$$\left(\forall \beta \leq \beta' < \lambda : \varphi_\beta \sqsubseteq \varphi_{\beta'} \wedge \varphi_\beta \leq \gamma(\phi^\sharp_\beta)\right) \;\Rightarrow\; \bigsqcup_{\beta<\lambda} \varphi_\beta \leq \gamma\left(\bigsqcup^\sharp_{\beta<\lambda} \phi^\sharp_\beta\right) \tag{7}$$

We get a fixpoint approximation method:

$$\mathrm{lfp}^{\sqsubseteq}_{\bot} F \leq \gamma\left(\bigsqcup^\sharp{}_{\lambda \in \mathbb{O}} F^{\sharp\lambda}\left(\bot^\sharp\right)\right) \tag{8}$$

---

[7] If $\lim \in \wp(P) \rightarrowtail P$ then a mapping $\varphi \in P \mapsto P$ is said to be a *complete* lim-*morphism*, written $\varphi \in P \stackrel{\lim}{\longmapsto} P$, (respectively lim-*upper-continuous*, lim-*lower-continuous*) if $\lim\{\varphi(x) \mid x \in X\} = \varphi(\lim\{x \mid x \in X\})$ for all $X \subseteq P$ (respectively increasing and decreasing chains $X \subseteq P$) when the limits are well-defined. For short, it is *continuous* if $\sqcup$-upper-continuous.

[8] Continuity would require the non-determinism of the primitives of the functional language considered in section 2 to be bounded.

[9] Ordinals constitute an extension into the infinite of the order properties of the naturals numbers: 0, 1, 2, ..., ω, $\omega+1$, $\omega+2$, ..., $\omega.2$, $\omega.2+1$, $\omega.2+2$, ..., ω^2, ..., ω^ω, ..., $\omega^{\omega^{\cdots}}$, ... where successor ordinals have the form $\lambda+1$, $\lambda \in \mathbb{O}$ and limit ordinals are of the form $\omega.\lambda$, $\lambda \in \mathbb{O} - \{0\}$ and ω is the least upper bound of the naturals.

Proposition 2 Fixpoint approximation. *(1), (2), (5), (6) and (7) imply (8).*

– It is often the case that the approximation relation is a partial ordering:

$$\leq \text{ is a partial order on } \mathcal{F} \tag{9}$$

Then, we can consider an abstract version $\leq^\sharp$ on $\mathcal{F}^\sharp$ of the concrete approximation ordering $\leq$ on $\mathcal{F}$:[10]

$$\leq^\sharp \text{ is a partial order on } \mathcal{F}^\sharp \tag{10}$$

The abstract approximation ordering approximates the concrete one, hence the semantic function γ is monotonic:

$$\gamma \in \mathcal{F}^\sharp \stackrel{\text{m}\leq^\sharp}{\longmapsto} \mathcal{F} \tag{11}$$

An interesting situation is when all concrete properties $\varphi \in \mathcal{F}$ have a best corresponding approximation $\alpha(\varphi) \in \mathcal{F}^\sharp$ where $\alpha \in \mathcal{F} \mapsto \mathcal{F}^\sharp$. Since $\alpha(\varphi)$ is an approximation that correctly describes φ, we have:

$$\forall \varphi \in \mathcal{F} : \varphi \leq \gamma(\alpha(\varphi)) \tag{12}$$

and since it is the most precise one, we also have:

$$\forall \varphi \in \mathcal{F} : \forall \phi^\sharp \in \mathcal{F}^\sharp : \varphi \leq \gamma(\phi^\sharp) \;\Rightarrow\; \alpha(\varphi) \leq^\sharp \phi^\sharp \tag{13}$$

(9) and (10) imply that (11), (12) and (13) are equivalent to the characteristic property of *Galois connections* written:

$$\mathcal{F}(\leq) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \mathcal{F}^\sharp(\leq^\sharp) \stackrel{\text{def}}{=} \forall \varphi \in \mathcal{F} : \forall \phi^\sharp \in \mathcal{F}^\sharp : \varphi \leq \gamma(\phi^\sharp) \;\Leftrightarrow\; \alpha(\varphi) \leq^\sharp \phi^\sharp \tag{14}$$

As suggested in [CC77c], a Galois connection can be lifted to higher-order functional spaces for the pointwise ordering:

$$\varphi \leq \phi \stackrel{\text{def}}{=} \forall x : \varphi(x) \leq \phi(x) \tag{15}$$

as follows:

Proposition 3 Functional Galois connection. *(9), (10), (14) for* $\langle \alpha_0, \gamma_0 \rangle$ *and* $\langle \alpha_1, \gamma_1 \rangle$ *and (15) for* $\leq$ *and* $\leq^\sharp$ *imply:*

$$\mathcal{F}_0 \stackrel{\text{m}\leq}{\longmapsto} \mathcal{F}_1(\leq) \underset{\lambda\Phi\bullet\alpha_1\circ\Phi\circ\gamma_0}{\overset{\lambda\Psi\bullet\gamma_1\circ\Psi\circ\alpha_0}{\leftrightarrows}} \mathcal{F}_0^\sharp \stackrel{\text{m}\leq^\sharp}{\longmapsto} \mathcal{F}_1^\sharp(\leq^\sharp) \tag{16}$$

– When no abstract property is useless, that is the abstraction function α is surjective or equivalently $\alpha \circ \gamma$ is the identity, we have a *Galois surjection* written:

$$\mathcal{F}(\leq) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \mathcal{F}^\sharp(\leq^\sharp) \stackrel{\text{def}}{=} \mathcal{F}(\leq) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \mathcal{F}^\sharp(\leq^\sharp) \wedge \forall \phi \in \mathcal{F}^\sharp : \alpha \circ \gamma(\phi) = \phi \tag{17}$$

Corollary 4 Functional Galois connection. *(9), (10), (14) for* $\langle \alpha_1, \gamma_1 \rangle$*, (17) for* $\langle \alpha_0, \gamma_0 \rangle$ *and (15) for* $\leq$ *and* $\leq^\sharp$ *imply:*

$$\mathcal{F}_0 \stackrel{\text{m}\leq}{\longmapsto} \mathcal{F}_1(\leq) \underset{\lambda\Phi\bullet\alpha_1\circ\Phi\circ\gamma_0}{\overset{\lambda\Psi\bullet\gamma_1\circ\Psi\circ\alpha_0}{\leftrightarrows}} \mathcal{F}_0^\sharp \longmapsto \mathcal{F}_1^\sharp(\leq^\sharp) \tag{18}$$

Corollary 5 Functional Galois surjection. *(9), (10), (17) for* $\langle \alpha_0, \gamma_0 \rangle$ *and* $\langle \alpha_1, \gamma_1 \rangle$ *and (15) for* $\leq$ *and* $\leq^\sharp$ *imply:*

$$\mathcal{F}_0 \stackrel{\text{m}\leq}{\longmapsto} \mathcal{F}_1(\leq) \underset{\lambda\Phi\bullet\alpha_1\circ\Phi\circ\gamma_0}{\overset{\lambda\Psi\bullet\gamma_1\circ\Psi\circ\alpha_0}{\leftrightarrows}} \mathcal{F}_0^\sharp \stackrel{\text{m}\leq^\sharp}{\longmapsto} \mathcal{F}_1^\sharp(\leq^\sharp) \tag{19}$$

[10] We can always define $\varphi^\sharp \leq^\sharp \phi^\sharp \stackrel{\text{def}}{=} \gamma(\varphi^\sharp) \leq \gamma(\phi^\sharp)$ and reason on the equivalence classes $\mathcal{F}^\sharp/\equiv^\sharp$ where $\varphi^\sharp \equiv^\sharp \phi^\sharp \stackrel{\text{def}}{=} \varphi^\sharp \leq^\sharp \phi^\sharp \wedge \phi^\sharp \leq^\sharp \varphi^\sharp$.

– Proposition 3 and corollary 5 lead to the hypothesis that approximations are preserved by the semantic function F:

$$F \in \mathcal{F} \stackrel{\mathrm{m}\leq}{\longmapsto} \mathcal{F} \tag{20}$$

Then, by defining $\bot^\sharp$, $F^\sharp$ and $\sqcup^\sharp$ such that:

$$\bot^\sharp \geq^\sharp \alpha(\bot) \tag{21}$$

$$F^\sharp \geq^\sharp \alpha \circ F \circ \gamma \tag{22}$$

and for all $\{\varphi_\beta \mid \beta \in \mathbb{O}\} \subseteq \mathcal{F}$, $\{\phi^\sharp_\beta \mid \beta \in \mathbb{O}\} \subseteq \mathcal{F}^\sharp$ and all limit ordinals λ (ω only if F is continuous):

$$\Big(\forall \beta \leq \beta' < \lambda : \varphi_\beta \sqsubseteq \varphi_{\beta'} \wedge \alpha(\varphi_\beta) \leq^\sharp \phi^\sharp_\beta\Big) \;\Rightarrow\; \alpha\Big(\bigsqcup_{\beta<\lambda} \varphi_\beta\Big) \leq^\sharp \bigsqcup^\sharp_{\beta<\lambda} \phi^\sharp_\beta \tag{23}$$

we obtain the following fixpoint approximation method:

Proposition 6 Fixpoint approximation. *(1), (2), (9), (10), (14), (15), (20), (21), (22) and (23) imply (8).*

– If the computational ordering has an abstract correspondent:

$$\mathcal{F}^\sharp(\sqsubseteq^\sharp, \bot^\sharp, \top^\sharp, \sqcup^\sharp, \sqcap^\sharp) \text{ is a complete lattice.} \tag{24}$$

such that:

$$\mathcal{F}(\sqsubseteq) \underset{\alpha^*}{\overset{\gamma}{\leftrightarrows}} \mathcal{F}^\sharp(\sqsubseteq^\sharp) \tag{25}$$

then we can characterize $\bot^\sharp$ and $\sqcup^\sharp$, as follows:

$$\bot^\sharp = \alpha(\bot) \tag{26}$$

$$\textstyle\bigsqcup^\sharp_{\lambda\in\mathbb{O}} \varphi^\sharp_\lambda = \alpha\Big(\bigsqcup_{\lambda\in\mathbb{O}} \gamma(\varphi^\sharp_\lambda)\Big) \tag{27}$$

Lemma 7. *(1), (24) and (25) imply (26) and (27).*

If the abstract computational and approximation orderings coincide then we can simplify hypothesis (23) as follows:

Lemma 8. *(1), (24), (25) and $\leq^\sharp = \sqsubseteq^\sharp$ imply (23).*

Corollary 9 Fixpoint approximation. *(1), (2), (9), (10), (14), (15), (20), (22), (24), (25) and $\leq^\sharp = \sqsubseteq^\sharp$ imply (8).*

2 Syntax and Relational Semantics of a First-order Functional Language

2.1 Syntax

In order to simplify the manipulation of tuples of parameters, we understand a tuple $(e_1, \ldots, e_i, \ldots, e_n)$ of actual arguments corresponding to formal parameters $(v_1, \ldots, v_i, \ldots, v_n)$ as a vector $\vec{e}$. This vector $\vec{e}$ is a notation for a special kind of functions from its domain $\mathrm{Dom}\,\vec{e} = \{v_1, \ldots, v_i, \ldots, v_n\}$ into the set of expressions such that $\vec{e}[v_i] = e_i$. The domain of $\vec{e}$ is left implicit in the notation $(e_1, \ldots, e_i, \ldots, e_n)$ but can be obtained from the syntactic context. The correspondence between v_i and e_i is given as usual by a positional notation so that each v_i and e_i has a rank $\mathrm{rk}(v_i) = \mathrm{rk}(e_i) = i$. We write $v \in \vec{e}$ for $v \in \mathrm{Dom}\,\vec{e}$ so that $\vec{e} = \prod_{v\in\vec{e}} \vec{e}[v]$. For uniformity, we also use a vector notation $\vec{v}$ for the tuple of formal parameters

$(v_1, \ldots, v_i, \ldots, v_n)$ with the convention that $v_i = \vec{v}[v_i]$. Finally, the same notation is used for functions, so that a program consists of function declarations $\vec{f}\vec{v} \equiv \vec{F}$, also written:

$$\prod_{f \in \vec{f}} f\vec{v} \equiv \vec{F}[f]$$

where the body $\vec{F}[f]$ of function f is an expression e depending upon the formal parameters $v \in \vec{v}$ and containing recursive calls to the functions $f \in \vec{f}$. This expression e is written according to the following syntax[11]:

e	$::= k$	constant
	$\mid v$	variable
	$\mid b\vec{e}$	basic operation
	$\mid f\vec{e}$	function call
	$\mid (e_1 \rightarrow e_2,\ e_3)$	conditional
$\vec{v}$	$::= (v_1, \ldots, v_n)$	tuple of formal parameters
$\vec{e}$	$::= (e_1, \ldots, e_n)$	tuple of actual arguments

2.2 Semantic Domains

Computed values belong to a given set Υ containing the booleans $\{\mathrm{tt}, \mathrm{ff}\}$ and other basic values such as natural numbers $\mathbb{N}$, integers $\mathbb{Z}$, reals $\mathbb{R}$, etc. We use $\Omega \notin \Upsilon$ to denote run-time errors and $\perp \notin \Upsilon$ to denote non-termination. Therefore saying that an expression has value Ω means that its evaluation terminates by the production of an error message. Saying that its value is $\perp$ means that its evaluation does not terminate. We define $\Upsilon^\Omega \stackrel{\text{def}}{=} \Upsilon \cup \{\Omega\}$, $\Upsilon_\perp \stackrel{\text{def}}{=} \Upsilon \cup \{\perp\}$ and $\Upsilon_\perp^\Omega \stackrel{\text{def}}{=} \Upsilon \cup \{\Omega, \perp\}$. Values ν taken by a variable v belong to $\mathcal{D}^\Re$ and values $\vec{\nu}$ taken by a tuple $\vec{v}$ of variables belong to $\vec{\mathcal{D}}^\Re$[12]:

$$\mathcal{D}^\Re \stackrel{\text{def}}{=} \Upsilon_\perp^\Omega \qquad \vec{\mathcal{D}}^\Re \stackrel{\text{def}}{=} \prod_{v \in \vec{v}} \mathcal{D}^\Re$$

Using Ω to denote run-time errors and $\perp$ to denote non-termination, we can always consider total functions over $\Upsilon_\perp^\Omega$ corresponding to partial functions over Υ. Contrary to a common practice in denotational semantics, which is explained and followed by [Mos90] (see page 599), we avoid representing errors by $\perp$ since we want to discriminate finite against infinite executions[13]. The *relational semantics* of a function declaration $f\vec{v} \equiv e$ is a relation $f^\Re \in \mathcal{F}^\Re$ where:

$$\mathcal{F}^\Re \stackrel{\text{def}}{=} \wp(\vec{\mathcal{D}}^\Re \times \mathcal{D}^\Re)$$

We write $f(\vec{\nu}) \rightsquigarrow \nu$ for $\langle \vec{\nu}, \nu \rangle \in f^\Re$ which means that the call of function f with actual arguments having values $\vec{\nu}$ may return the result ν. If this call may terminate

[11] Expressions e and tuples $\vec{e}$ of expressions have an attribute $e.\vec{v}$ (resp. $\vec{e}.\vec{v}$) which is the list of visible parameters and an attribute $e.\partial$ (resp. $\vec{e}.\partial$) which is the set of headings $f\vec{v}$ of the visible functions f which can occur free in the expression. Tuples of variables $\vec{v}$ (respectively tuples of expressions $\vec{e}$) have a domain attribute $\mathrm{Dom}\,\vec{v}$ (resp. $\mathrm{Dom}\,\vec{e}$) which is the set of variables occurring in the tuple (resp. the set of formal parameters corresponding to each expression in the tuple).

[12] $\vec{\mathcal{D}}^\Re$ is a shorthand for $\vec{\mathcal{D}}_{\vec{v}}^\Re = \prod_{v \in \mathrm{Dom}\,\vec{v}} \mathcal{D}^\Re$ where $\vec{v}$ is left context-dependent.

[13] More refinement of Ω might be useful. For example we could differentiate the class of run-time failures due to a type error (using "wrong" as in [Mil78] for the value 1/tt) from those due to partially defined well-typed functions (such as 1/0). Each kind of error might even be described by a different error value (corresponding to different error messages).

erroneously then $f(\vec{\nu}) \rightsquigarrow \Omega$ whereas $f(\vec{\nu}) \rightsquigarrow \bot$ indicates that nontermination is possible. The relational semantics $\vec{f}^{\Re}$ of a program $\prod_{f \in \vec{f}} f\vec{v} \equiv \vec{F}[f]$ (where the expression $\vec{F}[f]$ is the body of function f) is a tuple of functions belonging to $\vec{\mathcal{F}}^{\Re}$:

$$\vec{\mathcal{F}}^{\Re} \stackrel{\text{def}}{=} \prod_{f \in \vec{f}} \mathcal{F}^{\Re}$$

An expression e can be evaluated when the relational semantics $\vec{\varphi}$ of the functions $\vec{f}$ and the value $\vec{\nu}$ of the variables $\vec{v}$ which may occur free in e are known. We write $\vec{\varphi}, \vec{\nu} \vdash e \rightsquigarrow \nu$ to mean that evaluation of e in environment $\vec{\varphi}$, $\vec{\nu}$ may return ν. In particular this evaluation returns an error if $\nu = \Omega$ and does not terminate when $\nu = \bot$. Therefore the relational semantics $e^{\Re}$ of the expression e belongs to $\mathcal{E}^{\Re}$[14]:

$$\mathcal{E}^{\Re} \stackrel{\text{def}}{=} \vec{\mathcal{F}}^{\Re} \stackrel{\text{m}\subseteq}{\longmapsto} \wp(\vec{\mathcal{D}}^{\Re} \times \mathcal{D}^{\Re})$$

whereas the relational semantics $\vec{e}^{\Re}$ of a tuple of expressions $\vec{e}$ belongs to $\vec{\mathcal{E}}^{\Re}$[14]:

$$\vec{\mathcal{E}}^{\Re} \stackrel{\text{def}}{=} \vec{\mathcal{F}}^{\Re} \stackrel{\text{m}\subseteq}{\longmapsto} \wp(\vec{\mathcal{D}}^{\Re} \times \vec{\mathcal{D}}^{\Re})$$

2.3 Rule-based Presentation of the Relational Semantics

We now present the relational semantics of expressions, by induction on their syntax and then the semantics of programs. We use the technique of bi-inductive definitions introduced in [CC92], but resort only to the intuitive understanding of the reader. $\vec{f}^{\Re}, \vec{\nu} \vdash e \rightsquigarrow \nu$ means that evaluation of expression e may return value ν in the evaluation environment specified by $\vec{f}^{\Re}$, giving the relational semantics $\vec{f}^{\Re}[f]$ of the functions f used in e, and $\vec{\nu}$ giving the value $\vec{\nu}[v]$ of the variables v which can occur free in e.

– We assume that the value $\underline{k} \in \Upsilon$ of the constant k is given. Therefore:

$$\vec{f}^{\Re}, \vec{\nu} \vdash k \rightsquigarrow \underline{k} \; + \tag{28}$$

Such a positive axiom schema $\vec{f}^{\Re}, \vec{\nu} \vdash e \rightsquigarrow \nu$ +, marked "+", describes finite behaviors in $\vec{\mathcal{F}}^{\Re} \mapsto \wp(\vec{\mathcal{D}}^{\Re} \times \Upsilon^{\Omega})$ that is terminating or erroneous evaluations of expression e for all possible values of $\vec{f}^{\Re}$, $\vec{\nu}$ and ν.

– A formal parameter v denotes the value of the corresponding actual parameter which is stored in $\vec{\nu}[v]$:

$$\vec{f}^{\Re}, \vec{\nu} \vdash v \rightsquigarrow \vec{\nu}[v] \; + \qquad \text{if } \vec{\nu}[v] \in \Upsilon^{\Omega} \tag{29}$$

If the evaluation of the actual parameter does not terminate, which is formally represented by the fact that $\vec{\nu}[v] = \bot$, and the corresponding formal parameter v is used, then this use leads to a non-terminating evaluation:

$$\vec{f}^{\Re}, \vec{\nu} \vdash v \rightsquigarrow \bot \; - \qquad \text{if } \vec{\nu}[v] = \bot \tag{30}$$

Such a negative axiom schema, marked "-", describes infinite behaviors in $\vec{\mathcal{F}}^{\Re} \mapsto \wp(\vec{\mathcal{D}}^{\Re} \times \{\bot\})$ that is non-terminating computations for all possible values of $\vec{f}^{\Re}$, $\vec{\nu}$ and v satisfying the side condition $\vec{\nu}[v] = \bot$[15].

[14] Once again, e and $\vec{e}$ is left context-dependent so as to avoid the precise but heavy notation $\vec{\mathcal{E}}^{\Re}_{\vec{e}} = \left(\prod_{f\vec{v} \in \vec{e}.\partial} \wp\left(\left(\prod_{v \in \text{Dom } \vec{v}} \mathcal{D}^{\Re} \right) \times \mathcal{D}^{\Re} \right) \right) \stackrel{\text{m}\subseteq}{\longmapsto} \wp\left(\left(\prod_{v \in \vec{e}.\vec{v}} \mathcal{D}^{\Re} \right) \times \left(\prod_{v \in \text{Dom } \vec{e}} \mathcal{D}^{\Re} \right) \right)$.

[15] here all non-terminating computations deliver $\bot$ but in lazy languages they might also result in infinite data structures.

– The evaluation of a tuple $\vec{e}$ of actual parameters consists in evaluating each parameter $\vec{e}[v]$, $v \in \vec{e}$. This evaluation does not terminate if the evaluation of at least one component does not terminate. Else the evaluation of the tuple $\vec{e}$ of actual parameters is terminating or erroneous:

$$\frac{\forall v \in \vec{e} : \vec{f}^{\Re}, \vec{\nu} \vdash \vec{e}[v] \rightsquigarrow \vec{\nu}'[v]}{\vec{f}^{\Re}, \vec{\nu} \vdash \vec{e} \rightsquigarrow \vec{\nu}'}+ \qquad \text{if } \forall v \in \vec{e} : \vec{\nu}'[v] \in \Upsilon^{\Omega}$$

$$\frac{\forall v \in \vec{e} : \vec{f}^{\Re}, \vec{\nu} \vdash \vec{e}[v] \rightsquigarrow \vec{\nu}'[v]}{\vec{f}^{\Re}, \vec{\nu} \vdash \vec{e} \rightsquigarrow \vec{\nu}'}- \qquad \text{if } \exists v \in \vec{e} : \vec{\nu}'[v] = \bot$$

– We assume that the relational semantics of each basic operation b is specified by a total relation $b^{\Re} \in \mathbb{P}(\vec{\Upsilon}^{\Omega}_{\bot} \times \Upsilon^{\Omega}_{\bot})$ [16] (or a function $\underline{b} \in \vec{\Upsilon}^{\Omega}_{\bot} \mapsto \Upsilon^{\Omega}_{\bot}$ if the language is deterministic, in which case $b^{\Re} \stackrel{\text{def}}{=} \{\langle \vec{\nu}, \underline{b}(\vec{\nu})\rangle \mid \vec{\nu} \in \vec{\Upsilon}^{\Omega}_{\bot}\}$). We write $b(\vec{\nu}) \rightsquigarrow \nu$ for $\langle \vec{\nu}, \nu\rangle \in b^{\Re}$.

Example 1. • The integer addition is strict in its two parameters and is defined only for integers so that $\underline{\text{plus}}(\nu, \nu') \stackrel{\text{def}}{=} ((\nu = \bot \vee \nu' = \bot) \rightarrow \bot, ((\nu \notin \mathbb{Z} \vee \nu' \notin \mathbb{Z}) \rightarrow \Omega, (\nu + \nu')))$ where $+$ denotes the usual mathematical addition.

• For McCarthy's conjunction, we would have $\underline{\text{cand}}(\nu, \nu') \stackrel{\text{def}}{=} (\nu = \bot \rightarrow \bot, (\nu = \text{ff} \rightarrow \text{ff}, (\nu = \text{tt} \rightarrow ((\nu' \in \{\bot, \text{tt}, \text{ff}\} \rightarrow \nu', \Omega)), \Omega))$. Observe that the first parameter is passed by value since evaluation of the conjunction does not terminate as soon as this first parameter does not terminate whereas the second is passed by need, and used only if the first is tt.

• For the unbounded random assignment '?', which does not need its argument, we have $?(\nu) \rightsquigarrow n$ if and only if $n \in \mathbb{N}$.

• For the bounded random assignment, we have:

$$\begin{aligned}
?(\langle \nu, \nu'\rangle) &\rightsquigarrow \bot \quad \text{iff} \quad (\nu = \bot) \vee (\nu' = \bot),\\
?(\langle \nu, \nu'\rangle) &\rightsquigarrow \Omega \quad \text{iff} \quad (\nu \in \Upsilon^{\Omega} - \mathbb{Z}) \vee (\nu' \in \Upsilon^{\Omega} - \mathbb{Z}) \vee\\
&\qquad\qquad\qquad (\nu \in \mathbb{Z} \wedge \nu' \in \mathbb{Z} \wedge \nu > \nu'),\\
?(\langle \nu, \nu'\rangle) &\rightsquigarrow z \quad \text{iff} \quad (\nu \in \mathbb{Z} \wedge \nu' \in \mathbb{Z} \wedge \nu \leq z \leq \nu').
\end{aligned}$$

□

The value of $b\vec{e}$ is obtained by applying b to the value of the arguments $\vec{e}$ (which may be Ω or $\bot$ without preventing the result to be in Υ when the argument is not needed):

$$\frac{\vec{f}^{\Re}, \vec{\nu} \vdash \vec{e} \rightsquigarrow \vec{\nu}' \ \wedge \ b(\vec{\nu}') \rightsquigarrow \bot}{\vec{f}^{\Re}, \vec{\nu} \vdash b\vec{e} \rightsquigarrow \bot}-$$

$$\frac{\vec{f}^{\Re}, \vec{\nu} \vdash \vec{e} \rightsquigarrow \vec{\nu}' \ \wedge \ b(\vec{\nu}') \rightsquigarrow \nu}{\vec{f}^{\Re}, \vec{\nu} \vdash b\vec{e} \rightsquigarrow \nu}+ \qquad \text{if } \nu \in \Upsilon^{\Omega}$$

– The evaluation of the conditional $(e_1 \rightarrow e_2, e_3)$ does not terminate when the evaluation of the condition e_1 does not terminate:

$$\frac{\vec{f}^{\Re}, \vec{\nu} \vdash e_1 \rightsquigarrow \bot}{\vec{f}^{\Re}, \vec{\nu} \vdash (e_1 \rightarrow e_2, e_3) \rightsquigarrow \bot}-$$

[16] We define the set $\mathbb{P}(S \times T)$ of *total binary relations* on $S \times T$ as $\{\rho \in \wp(S \times T) \mid \forall s \in S : \exists t \in T : \langle s, t\rangle \in \rho\}$ so that $\rho \in \mathbb{P}(S \times T)$ implies that for every $s \in S$ there exists at least one $t \in T$ such that the pair $\langle s, t\rangle$ belongs to ρ.

The evaluation of the conditional $(e_1 \rightarrow e_2, e_3)$ terminates with an error when the evaluation of the condition e_1 terminates without returning a boolean:

$$\frac{\vec{f}^{\Re}, \vec{\nu} \vdash e_1 \rightsquigarrow \nu \ \wedge \ \nu \in \Upsilon^{\Omega} - \{\mathrm{tt}, \mathrm{ff}\}}{\vec{f}^{\Re}, \vec{\nu} \vdash (e_1 \rightarrow e_2, e_3) \rightsquigarrow \Omega}\ +$$

If the evaluation of the condition e_1 returns a boolean b, then the value of the conditional $(e_1 \rightarrow e_2, e_3)$ is that of e_2 if b is true and that of e_3 if b is false, including the run-time error and non-termination cases:

$$\frac{\vec{f}^{\Re}, \vec{\nu} \vdash e_1 \rightsquigarrow \mathrm{tt} \ \wedge \ \vec{f}^{\Re}, \vec{\nu} \vdash e_2 \rightsquigarrow \nu}{\vec{f}^{\Re}, \vec{\nu} \vdash (e_1 \rightarrow e_2, e_3) \rightsquigarrow \nu}\ + \qquad \text{if } \nu \in \Upsilon^{\Omega}$$

$$\frac{\vec{f}^{\Re}, \vec{\nu} \vdash e_1 \rightsquigarrow \mathrm{tt} \ \wedge \ \vec{f}^{\Re}, \vec{\nu} \vdash e_2 \rightsquigarrow \bot}{\vec{f}^{\Re}, \vec{\nu} \vdash (e_1 \rightarrow e_2, e_3) \rightsquigarrow \bot}\ -$$

$$\frac{\vec{f}^{\Re}, \vec{\nu} \vdash e_1 \rightsquigarrow \mathrm{ff} \ \wedge \ \vec{f}^{\Re}, \vec{\nu} \vdash e_3 \rightsquigarrow \nu}{\vec{f}^{\Re}, \vec{\nu} \vdash (e_1 \rightarrow e_2, e_3) \rightsquigarrow \nu}\ + \qquad \text{if } \nu \in \Upsilon^{\Omega}$$

$$\frac{\vec{f}^{\Re}, \vec{\nu} \vdash e_1 \rightsquigarrow \mathrm{ff} \ \wedge \ \vec{f}^{\Re}, \vec{\nu} \vdash e_3 \rightsquigarrow \bot}{\vec{f}^{\Re}, \vec{\nu} \vdash (e_1 \rightarrow e_2, e_3) \rightsquigarrow \bot}\ -$$

– The relational semantics of the function call $f\vec{e}$ can be defined knowing the value $\vec{\nu}$ of the free variables $\vec{v}$ which may appear within $\vec{e}$ and the relational semantics $f^{\Re} = \vec{f}^{\Re}[f]$ of the function $f \in \vec{f}$ of the program $\prod_{f \in \vec{f}} f\vec{v} \equiv \vec{F}[f]$. $f^{\Re}$ specifies a set of arguments-result pairs $f(\vec{\nu}') \rightsquigarrow \nu$. The call $f\vec{e}$ may return value ν if, given the value $\vec{\nu}'$ of the actual parameters $\vec{e}$, the function f may return that value ν, that is $f(\vec{\nu}') \rightsquigarrow \nu$. Therefore, we have:

$$\frac{\vec{f}^{\Re}, \vec{\nu} \vdash \vec{e} \rightsquigarrow \vec{\nu}' \ \wedge \ f(\vec{\nu}') \rightsquigarrow \nu}{\vec{f}^{\Re}, \vec{\nu} \vdash f\vec{e} \rightsquigarrow \nu}\ + \qquad \text{if } \nu \in \Upsilon^{\Omega} \tag{31}$$

$$\frac{\vec{f}^{\Re}, \vec{\nu} \vdash \vec{e} \rightsquigarrow \vec{\nu}' \ \wedge \ f(\vec{\nu}') \rightsquigarrow \bot}{\vec{f}^{\Re}, \vec{\nu} \vdash f\vec{e} \rightsquigarrow \bot}\ - \tag{32}$$

Observe that in an effective implementation of the above rules (31) and (32), parameters must be passed by need since a non-terminating actual parameter $\vec{e}[v]$, such that $\vec{\nu}'[v] = \bot$, will actually affects the computation only if the corresponding formal parameter v is needed in the body $\vec{F}[f]$ of function f according to rules (29) and (30). Moreover call-by-name would be inadequate since the effective parameters are evaluated only once. However this kind of implementation detail is omitted in the mathematical description of the relational semantics where the behaviours of the actual parameters, including the non-terminating ones, can be described prior to describing the behaviour of the function call.

– The relational semantics $\vec{f}^{\Re}$ of a program $\prod_{f \in \vec{f}} f\vec{v} \equiv \vec{F}[f]$ specifies that a call of function f consists in evaluating the function body $\vec{F}[f]$ with formal parameters bound to their actual values:

$$\frac{\vec{f}^{\Re}, \vec{\nu} \vdash \vec{F}[f] \rightsquigarrow \nu}{f(\vec{\nu}) \rightsquigarrow \nu}\ + \qquad \text{if } \nu \in \Upsilon^{\Omega} \tag{33}$$

$$\frac{\vec{f}^{\Re}, \vec{\nu} \vdash \vec{F}[f] \rightsquigarrow \bot}{f(\vec{\nu}) \rightsquigarrow \bot}\ - \tag{34}$$

These rules are inductive since recursive function calls may occur in function bodies. For the positive rule (33), this is an induction of the length of the computations: the computations for the recursive calls, are included in the computation of the main call, hence shorter. The basis of this induction is given by the axioms (28) and (29) which involve a single step computation. This reasoning is no longer valid for non-terminating calls, that is in the negative rule (34), since the main and recursive sub-calls all involve infinite computations. Hence the induction involved in the negative rules should be understood quite differently. One must imagine that the basis is given by all infinite computations $f(\vec{\nu}) \rightsquigarrow \bot$ and that the rule eliminates those which cannot be obtained by an evaluation of the function body $\vec{F}[f]$ where recursive calls must correspond to non-terminating calls which are not yet eliminated. Repeating this process ad infinitum, only the non-terminating behaviours will be left out.

Example 2. Let us consider $\mathtt{f(x)} \equiv (\mathtt{x} = \mathtt{0} \rightarrow \mathtt{1}, \mathtt{f(x-1)})$. Starting with the basis $f^{\Re} = \{f(n) \rightsquigarrow \bot \mid n \in \mathbb{Z}_{\bot}^{\Omega}\}$ and applying the above rules to the function body we derive that $f(\Omega) \rightsquigarrow \Omega$, $f(0) \rightsquigarrow 1$ and $f(n) \rightsquigarrow \bot$ if $n \in \mathbb{Z}_{\bot} - \{0\}$ so that, by elimination of the impossible infinite behaviour, we get $f^{\Re} = \{\langle \Omega, \Omega\rangle, \langle 0, 1\rangle\} \cup \{\langle n, \bot\rangle \mid n \in \mathbb{Z}_{\bot} - \{0\}\}$. Repeating this process, we discover that $f(1)$ calls $f(0)$ which has no infinite behaviour so that the non-termination of $f(1)$ is impossible. The second approximation of $f^{\Re}$ is now $\{\langle \Omega, \Omega\rangle\} \cup \{\langle n, 1\rangle \mid 0 \leq n \leq 1\} \cup \{\langle n, \bot\rangle \mid \mathbb{Z}_{\bot} - \{0, 1\}\}$. After i repetitions of this elimination process we get $f^{\Re} = \{\langle \Omega, \Omega\rangle\} \cup \{\langle n, 1\rangle \mid 0 \leq n < i\} \cup \{\langle n, \bot\rangle \mid \mathbb{Z}_{\bot} - \{0, \ldots, i-1\}\}$ so that passing to the limit we conclude that $f^{\Re} = \{\langle \Omega, \Omega\rangle\} \cup \{\langle n, 1\rangle \mid 0 \leq n\} \cup \{\langle n, \bot\rangle \mid n < 0 \vee n = \bot\}$. □

More generally, such bi-inductive definitions can be given a precise meaning as specifications of fixpoints of monotone operators on complete lattices [CC92] which we now illustrate for the relational semantics.

2.4 Fixpoint Presentation of the Relational Semantics

In order to capture the idea that terminating evaluations of recursive functions are obtained inductively by construction from the terminating evaluation of recursive calls while non-terminating evaluations are obtained destructively by elimination of inaccessible recursive calls, let us introduce the partial order $\sqsubseteq^{\Re}$ on $\mathcal{F}^{\Re}$ together with the corresponding least upper bound and greatest lower bound defined by:

$$\bot^{\Re} \stackrel{\text{def}}{=} \vec{\Upsilon}_{\bot}^{\Omega} \times \{\bot\} \tag{35}$$

$$\top^{\Re} \stackrel{\text{def}}{=} \vec{\Upsilon}_{\bot}^{\Omega} \times \Upsilon^{\Omega} \tag{36}$$

$$\varphi \sqsubseteq^{\Re} \varphi' \stackrel{\text{def}}{=} (\varphi \cap \top^{\Re}) \subseteq (\varphi' \cap \top^{\Re}) \ \wedge \ (\varphi \cap \bot^{\Re}) \supseteq (\varphi' \cap \bot^{\Re}) \tag{37}$$

$$\sqcup^{\Re}_{i \in \Delta} \varphi_i \stackrel{\text{def}}{=} \cup_{i \in \Delta}(\varphi_i \cap \top^{\Re}) \ \cup \ \cap_{i \in \Delta}(\varphi_i \cap \bot^{\Re}) \tag{38}$$

$$\sqcap^{\Re}_{i \in \Delta} \varphi_i \stackrel{\text{def}}{=} \cap_{i \in \Delta}(\varphi_i \cap \top^{\Re}) \ \cup \ \cup_{i \in \Delta}(\varphi_i \cap \bot^{\Re})$$[17]

$\mathcal{F}^{\Re}(\sqsubseteq^{\Re}, \bot^{\Re}, \top^{\Re}, \sqcup^{\Re}, \sqcap^{\Re})$ is a complete lattice. The pointwise extension to tuples of relations $\vec{\mathcal{F}}^{\Re}(\vec{\sqsubseteq}^{\Re}, \vec{\bot}^{\Re}, \vec{\top}^{\Re}, \vec{\sqcup}^{\Re}, \vec{\sqcap}^{\Re})$ is also a complete lattice. Therefore we can define the relational semantics $\vec{f}^{\Re}$ of the program $\prod_{f \in \vec{f}} f\vec{v} \equiv \vec{F}[f]$ (where the body $\vec{F}[f]$ of function $f \in \vec{f}$ is an expression) as the least fixpoint of a monotonic operator

[17] This is for $\Delta \neq \emptyset$. As usual, $\sqcup^{\Re}\emptyset = \bot^{\Re}$ and $\sqcap^{\Re}\emptyset = \top^{\Re}$.

$\vec{F}^{\Re}$ on this complete lattice $\vec{\mathcal{F}}^{\Re}$ (thus avoiding the difficulties resulting from the non-existence of arbitrary upper bounds in cpos). This operator $\vec{F}^{\Re} \in \vec{\mathcal{F}}^{\Re} \stackrel{\mathrm{m}\vec{\sqsubseteq}^{\Re}}{\longmapsto} \vec{\mathcal{F}}^{\Re}$ corresponds to rule schemata (33) and (34) [18]:

$$\vec{f}^{\Re} \stackrel{\text{def}}{=} \mathrm{lfp}^{\vec{\sqsubseteq}^{\Re}}_{\vec{\perp}^{\Re}} \vec{F}^{\Re} \qquad \vec{F}^{\Re} \stackrel{\text{def}}{=} \lambda\vec{\varphi}\bullet\prod_{f\,\epsilon\,\vec{f}} \vec{F}[f]^{\Re}[\![\vec{\varphi}]\!] \tag{39}$$

The operator $\vec{F}[f]^{\Re} \in \vec{\mathcal{F}}^{\Re} \stackrel{\mathrm{m}\sqsubseteq^{\Re}}{\longmapsto} \mathcal{F}^{\Re}$ is defined by induction on the syntax of the expression $\vec{F}[f]$ constituting the body of function f according to the axiom and rule schemata (28) to (34)[19]:

$$k^{\Re}[\![\vec{\varphi}]\!] \stackrel{\text{def}}{=} \{\langle\vec{\nu},\, \underline{k}\rangle \mid \vec{\nu} \in \vec{\mathcal{D}}^{\Re}\} \tag{40}$$

$$v^{\Re}[\![\vec{\varphi}]\!] \stackrel{\text{def}}{=} \{\langle\vec{\nu},\, \vec{\nu}[v]\rangle \mid \vec{\nu} \in \vec{\mathcal{D}}^{\Re}\} \tag{41}$$

$$\vec{e}^{\Re}[\![\vec{\varphi}]\!] \stackrel{\text{def}}{=} \{\langle\vec{\nu},\, \vec{\nu}'\rangle \mid \forall v\,\epsilon\,\vec{e} : \langle\vec{\nu},\, \vec{\nu}'[v]\rangle \in \vec{e}[v]^{\Re}[\![\vec{\varphi}]\!]\} \tag{42}$$

$$b\vec{e}^{\Re}[\![\vec{\varphi}]\!] \stackrel{\text{def}}{=} \vec{e}^{\Re}[\![\vec{\varphi}]\!] \circ b^{\Re} \tag{43}$$

$$\begin{aligned}(e_1 \to e_2,\, e_3)^{\Re}[\![\vec{\varphi}]\!] \stackrel{\text{def}}{=}\ & \{\langle\vec{\nu},\, \perp\rangle \mid \langle\vec{\nu},\, \perp\rangle \in e_1^{\Re}[\![\vec{\varphi}]\!]\} \\ \cup\ & \{\langle\vec{\nu},\, \Omega\rangle \mid \exists\nu \in \Upsilon^{\Omega} - \{\mathrm{tt},\, \mathrm{ff}\} : \langle\vec{\nu},\, \nu\rangle \in e_1^{\Re}[\![\vec{\varphi}]\!]\} \\ \cup\ & \{\langle\vec{\nu},\, \nu\rangle \mid \langle\vec{\nu},\, \mathrm{tt}\rangle \in e_1^{\Re}[\![\vec{\varphi}]\!] \wedge \langle\vec{\nu},\, \nu\rangle \in e_2^{\Re}[\![\vec{\varphi}]\!]\} \\ \cup\ & \{\langle\vec{\nu},\, \nu\rangle \mid \langle\vec{\nu},\, \mathrm{ff}\rangle \in e_1^{\Re}[\![\vec{\varphi}]\!] \wedge \langle\vec{\nu},\, \nu\rangle \in e_3^{\Re}[\![\vec{\varphi}]\!]\}\end{aligned} \tag{44}$$

$$f\vec{e}^{\Re}[\![\vec{\varphi}]\!] \stackrel{\text{def}}{=} \vec{e}^{\Re}[\![\vec{\varphi}]\!] \circ \vec{\varphi}[f] \tag{45}$$

Example 3. For the program:

$$\texttt{f(x)} \equiv \texttt{(x = 0} \to \texttt{0, (x < 0} \to \texttt{f(?(())), f(x - 1)))}$$

where `{()}` is the unit type and `?(())` returns any nonnegative integer (so that the nondeterminism is unbounded) the equation is:

$$\begin{aligned}\vec{F}^{\Re}[f][\![\vec{\varphi}]\!] = {} & \{\langle\perp,\, \perp\rangle, \langle\Omega,\, \Omega\rangle, \langle 0,\, 0\rangle\} \cup \{\langle x,\, y\rangle \mid x < 0 \wedge \exists n \geq 0 : \langle n,\, y\rangle \in \vec{\varphi}[f]\} \\ & \cup \{\langle x,\, y\rangle \mid x > 0 \wedge \langle x - 1,\, y\rangle \in \vec{\varphi}[f]\}\end{aligned}$$

The transfinite iterates $\varphi^{\lambda} = \vec{F}^{\Re}[f]^{\lambda}(\perp^{\Re})$ are:

$$\begin{aligned}\varphi^{0} &= \perp^{\Re} = \mathbb{Z}^{\Omega}_{\perp} \times \{\perp\} \\ \varphi^{1} &= \{\langle\perp,\, \perp\rangle, \langle\Omega,\, \Omega\rangle, \langle 0,\, 0\rangle\} \cup \{\langle x,\, \perp\rangle \mid x \neq 0\} \\ \varphi^{2} &= \{\langle\perp,\, \perp\rangle, \langle\Omega,\, \Omega\rangle\} \cup \{\langle x,\, 0\rangle \mid x < 2\} \cup \{\langle x,\, \perp\rangle \mid x < 0 \vee x \geq 2\} \\ &\cdots \\ \varphi^{n} &= \{\langle\perp,\, \perp\rangle, \langle\Omega,\, \Omega\rangle\} \cup \{\langle x,\, 0\rangle \mid x < n\} \cup \{\langle x,\, \perp\rangle \mid x < 0 \vee x \geq n\} \\ &\cdots \\ \varphi^{\omega} &= \{\langle\perp,\, \perp\rangle, \langle\Omega,\, \Omega\rangle\} \cup \{\langle x,\, 0\rangle \mid x \in \mathbb{Z}\} \cup \{\langle x,\, \perp\rangle \mid x < 0\} \\ \varphi^{\omega+1} &= \{\langle\perp,\, \perp\rangle, \langle\Omega,\, \Omega\rangle\} \cup \{\langle x,\, 0\rangle \mid x \in \mathbb{Z}\} \\ \varphi^{\omega+2} &= \varphi^{\omega+1}\end{aligned}$$

proving that the program returns 0 for all integer parameters. □

[18] $\mathrm{lfp}^{\leq}_{x}\varphi$ is the least fixpoint of φ greater than or equal to x for partial ordering $\leq$.

[19] The *composition* $\rho \circ \rho'$ of two binary relations $\rho \in \wp(S \times T)$ and $\rho' \in \wp(T \times U)$ is the relation $\{\langle s,\, u\rangle \in S \times U \mid \exists t \in T : \langle s,\, t\rangle \in \rho \wedge \langle t,\, u\rangle \in \rho'\}$.

We observe that the semantics is well-defined:

Lemma 10.

$$\begin{array}{c} \vec{\varphi} \mathrel{\vec{\subseteq}} \vec{\varphi}' \Rightarrow e[\![\vec{\varphi}]\!] \subseteq e[\![\vec{\varphi}']\!] \wedge \vec{e}[\![\vec{\varphi}]\!] \subseteq \vec{e}[\![\vec{\varphi}']\!] \\ \vec{\varphi} \mathrel{\vec{\sqsubseteq}^{\Re}} \vec{\varphi}' \Rightarrow e[\![\vec{\varphi}]\!] \sqsubseteq^{\Re} e[\![\vec{\varphi}']\!] \wedge \vec{e}[\![\vec{\varphi}]\!] \sqsubseteq^{\Re} \vec{e}[\![\vec{\varphi}']\!] \\ \vec{F}^{\Re} \in \vec{\mathcal{F}}^{\Re} \overset{\mathrm{m}\vec{\subseteq}}{\longmapsto} \vec{F}^{\Re} \ \wedge \ \vec{F}^{\Re} \in \vec{\mathcal{F}}^{\Re} \overset{\mathrm{m}\vec{\sqsubseteq}^{\Re}}{\longmapsto} \vec{F}^{\Re} \end{array} \tag{46}$$

Proposition 11. *If $b^{\Re} \in \mathbb{P}(\vec{\mathcal{D}}^{\Re} \times \mathcal{D}^{\Re})$ is a total binary relation for all basic operations b then a program defines a total binary relation*[20]*, that is $\forall f \in \vec{f} : \vec{f}^{\Re}[f] \in \mathbb{P}(\vec{\mathcal{D}}^{\Re} \times \mathcal{D}^{\Re})$.*

$\nu : \mathcal{D}^{\Re} \overset{\text{def}}{=} \Upsilon_{\perp}^{\Omega}$ $\vec{\nu} : \vec{\mathcal{D}}^{\Re} \overset{\text{def}}{=} \prod_{v \in \vec{v}} \mathcal{D}^{\Re}$ $\varphi, f^{\Re} : \mathcal{F}^{\Re} \overset{\text{def}}{=} \wp(\vec{\mathcal{D}}^{\Re} \times \mathcal{D}^{\Re})$ $\vec{\varphi}, \vec{f}^{\Re} : \vec{\mathcal{F}}^{\Re} \overset{\text{def}}{=} \prod_{f \in \vec{f}} \mathcal{F}^{\Re}$	$e^{\Re} : \mathcal{E}^{\Re} \overset{\text{def}}{=} \vec{\mathcal{F}}^{\Re} \overset{\mathrm{m}\vec{\sqsubseteq}^{\Re}}{\longmapsto} \wp(\vec{\mathcal{D}}^{\Re} \times \mathcal{D}^{\Re})$ $\vec{e}^{\,\Re} : \vec{\mathcal{E}}^{\Re} \overset{\text{def}}{=} \vec{\mathcal{F}}^{\Re} \overset{\mathrm{m}\vec{\sqsubseteq}^{\Re}}{\longmapsto} \wp(\vec{\mathcal{D}}^{\Re} \times \vec{\mathcal{D}}^{\Re})$ $\vec{F}^{\Re} : \vec{\mathcal{F}}^{\Re} \overset{\mathrm{m}\vec{\sqsubseteq}^{\Re}}{\longmapsto} \vec{\mathcal{F}}^{\Re}$
$k^{\Re}[\![\vec{\varphi}]\!] \overset{\text{def}}{=} \{\langle \vec{\nu}, \underline{k} \rangle \mid \vec{\nu} \in \vec{\mathcal{D}}^{\Re}\}$ $v^{\Re}[\![\vec{\varphi}]\!] \overset{\text{def}}{=} \{\langle \vec{\nu}, \vec{\nu}[v] \rangle \mid \vec{\nu} \in \vec{\mathcal{D}}^{\Re}\}$ $b\vec{e}^{\,\Re}[\![\vec{\varphi}]\!] \overset{\text{def}}{=} \vec{e}^{\,\Re}[\![\vec{\varphi}]\!] \circ b^{\Re}$ $(e_1 \to e_2, e_3)^{\Re}[\![\vec{\varphi}]\!] \overset{\text{def}}{=} \{\langle \vec{\nu}, \perp \rangle \mid \langle \vec{\nu}, \perp \rangle \in e_1^{\Re}[\![\vec{\varphi}]\!]\}$ $\cup \{\langle \vec{\nu}, \Omega \rangle \mid \exists \nu \in \Upsilon^{\Omega} - \{\mathrm{tt}, \mathrm{ff}\} : \langle \vec{\nu}, \nu \rangle \in e_1^{\Re}[\![\vec{\varphi}]\!]\}$ $\cup \{\langle \vec{\nu}, \nu \rangle \mid \langle \vec{\nu}, \mathrm{tt} \rangle \in e_1^{\Re}[\![\vec{\varphi}]\!] \wedge \langle \vec{\nu}, \nu \rangle \in e_2^{\Re}[\![\vec{\varphi}]\!]\}$ $\cup \{\langle \vec{\nu}, \nu \rangle \mid \langle \vec{\nu}, \mathrm{ff} \rangle \in e_1^{\Re}[\![\vec{\varphi}]\!] \wedge \langle \vec{\nu}, \nu \rangle \in e_3^{\Re}[\![\vec{\varphi}]\!]\}$ $f\vec{e}^{\,\Re}[\![\vec{\varphi}]\!] \overset{\text{def}}{=} \vec{e}^{\,\Re}[\![\vec{\varphi}]\!] \circ \vec{\varphi}[f]$ $\vec{e}^{\,\Re}[\![\vec{\varphi}]\!] \overset{\text{def}}{=} \{\langle \vec{\nu}, \vec{\nu}' \rangle \mid \forall v \in \vec{e} : \langle \vec{\nu}, \vec{\nu}'[v] \rangle \in \vec{e}[v]^{\Re}[\![\vec{\varphi}]\!]\}$	
$\perp^{\Re} \overset{\text{def}}{=} \vec{\mathcal{D}}^{\Re} \times \{\perp\}$ $\top^{\Re} \overset{\text{def}}{=} \vec{\mathcal{D}}^{\Re} \times \Upsilon^{\Omega}$ $\varphi \sqsubseteq^{\Re} \varphi' \overset{\text{def}}{=} (\varphi \cap \top^{\Re}) \subseteq (\varphi' \cap \top^{\Re}) \ \wedge \ (\varphi \cap \perp^{\Re}) \supseteq (\varphi' \cap \perp^{\Re})$ $\bigsqcup_{i \in \Delta}^{\Re} \varphi_i \overset{\text{def}}{=} \bigcup_{i \in \Delta} (\varphi_i \cap \top^{\Re}) \cup \bigcap_{i \in \Delta} (\varphi_i \cap \perp^{\Re})$ $\vec{F}^{\Re} \overset{\text{def}}{=} \lambda \vec{\varphi} \bullet \prod_{f \in \vec{f}} \vec{F}[f]^{\Re}[\![\vec{\varphi}]\!]$ $\vec{f}^{\Re} \overset{\text{def}}{=} \mathrm{lfp}_{\vec{\perp}^{\Re}}^{\vec{\sqsubseteq}^{\Re}} \vec{F}^{\Re} = \vec{\bigsqcup}_{\lambda \in \mathbb{O}}^{\Re} \vec{F}^{\Re\,\lambda}(\vec{\perp}^{\Re})$	
$b(\vec{\nu}) \rightsquigarrow \nu \overset{\text{def}}{=} \langle \vec{\nu}, \nu \rangle \in b^{\Re}$ given such that $b^{\Re} \in \mathcal{F}^{\Re}$ $f(\vec{\nu}) \rightsquigarrow \nu \overset{\text{def}}{=} \langle \vec{\nu}, \nu \rangle \in \vec{f}^{\Re}[f]$ $\vec{\varphi}, \vec{\nu} \vdash e \rightsquigarrow \nu \overset{\text{def}}{=} \langle \vec{\nu}, \nu \rangle \in e^{\Re}[\![\vec{\varphi}]\!]$ $\vec{\varphi}, \vec{\nu} \vdash \vec{e} \rightsquigarrow \vec{\nu}' \overset{\text{def}}{=} \langle \vec{\nu}, \vec{\nu}' \rangle \in \vec{e}^{\,\Re}[\![\vec{\varphi}]\!]$	

Fig. 1. Synopsis of the relational semantics of $\prod_{f \in \vec{f}} f\vec{v} \equiv \vec{F}[f]$

3 Forward Strictness Analysis by Abstract Interpretation

By applying Alan Mycroft's approximation to the relational semantics, we find again his dependence-sensitive strictness analysis method by mere calculus. In order to

[20] Partial relations r are represented by a total relation R such that $\langle \vec{\nu}, \Omega \rangle \in R$ when $\forall \nu \in \Upsilon_{\perp} : \langle \vec{\nu}, \Omega \rangle \notin r$.

avoid combinatorial explosion, we shortly explore dependence-free strictness analysis methods and finally suggest a compromise using widenings.

3.1 Definition of Strictness

A function φ is *strict* in its parameters $v \in I$ if the evaluation of φ cannot terminate whenever that of its parameters $v \in I$ does not terminate, that is $\varphi(\bot) = \bot$ when φ has only one parameter. If we define $f(x) \equiv (0 \rightarrow f(x),\ f(x))$ then $f(x) = \Omega$ since 0 is not a boolean so that f is not strict. However, using Mycroft's equations, we conclude that f is strict. Hence differentiating between terminating errors and non-termination is possible but would lead to a different strictness analysis algorithm. Since we want to rediscover Mycroft's algorithm, we can assume, by type checking, that Ω never appears in positions where the abstract equations would differ (i.e. essentially in boolean tests). This is not convincing, e.g. if we want to infer simultaneously type and strictness information. Another solution, as chosen by Mycroft, consists in assimilating Ω and $\bot$ in the semantics. Although commonly accepted, we find that this reasoning is an approximation which should be made explicit. This can be explained by assimilating Ω to $\bot$ in the strictness analysis only, as follows:

Definition 12. $\varphi \in \mathcal{F}^{\mathfrak{R}} = \mathbb{P}(\vec{\mathcal{D}}^{\mathfrak{R}} \times \mathcal{D}^{\mathfrak{R}})$ is *strict* in its parameters $v \in I$ if and only if for all $\vec{\nu} \in \vec{\mathcal{D}}^{\mathfrak{R}}$ and $\nu \in \mathcal{D}^{\mathfrak{R}}$:

$$(\forall v \in I : \vec{\nu}[v] \in \{\bot, \Omega\} \ \wedge\ \varphi(\vec{\nu}) \rightsquigarrow \nu) \ \Rightarrow\ \nu \in \{\bot, \Omega\}$$

Since relation $\varphi(\vec{\nu}) \rightsquigarrow \nu$ is not effectively computable, Mycroft [Myc80, Myc81] uses an approximation φ' of φ for the *approximation ordering* $\subseteq$:

Proposition 13. *For all φ, $\varphi' \in \mathcal{F}^{\mathfrak{R}}$, if φ' is strict in its parameters $v \in I$ and $\varphi \subseteq \varphi'$ then φ is strict in its parameters $v \in I$.*

Moreover this approximation φ' of φ can be effectively computed by abstract interpretation.

3.2 Forward Dependence-Sensitive Strictness Analysis

À la Mycroft Abstraction Mycroft's original idea [Myc80, Myc81] is to approximate functions $\varphi^o \in \mathcal{F}^o = \vec{\mathcal{D}}^o \mapsto \mathcal{D}^o$ (where $\mathcal{D}^o$ is Scott's flat domain) by functions $\varphi^\sharp \in \mathcal{F}^\sharp = \vec{\mathcal{D}}^\sharp \mapsto \mathcal{D}^\sharp$. $\mathcal{D}^\sharp$ is the complete lattice $\{0, 1\}(\leq, 0, 1, \vee, \wedge)$ with $0 \leq 0 < 1 \leq 1$. 0 is the abstraction of $\bot$ (and Ω) and 1 that of any other value $\nu \in \Upsilon$:

$$\begin{array}{lll} \mathcal{D}^\sharp \stackrel{\text{def}}{=} \{0,1\} & \alpha^\sharp_\Upsilon(\nu) \stackrel{\text{def}}{=} 1 & \text{if } \nu \neq \bot \text{ and } \nu \neq \Omega \\ \alpha^\sharp_\Upsilon \in \mathcal{D}^{\mathfrak{R}} \mapsto \mathcal{D}^\sharp & \alpha^\sharp_\Upsilon(\bot) \stackrel{\text{def}}{=} 0 & \\ & \alpha^\sharp_\Upsilon(\Omega) \stackrel{\text{def}}{=} 0 & \end{array} \tag{47}$$

We only slightly deviate from his judicious choice in that instead of considering functions we use a similar abstraction for relations $\varphi^{\mathfrak{R}} \in \mathcal{F}^{\mathfrak{R}}$, using Galois connections.

A set is approximated by the most imprecise approximation of its elements:

$$\alpha^\sharp_{\mathcal{D}}(V) \stackrel{\text{def}}{=} \bigvee\{\alpha^\sharp_\Upsilon(\nu) \mid \nu \in V\} \qquad \gamma^\sharp_{\mathcal{D}}(\nu) \stackrel{\text{def}}{=} \{\nu' \mid \alpha^\sharp_\Upsilon(\nu') \leq \nu\} \tag{48}$$

Proposition 14. *If $\mathcal{D}^\sharp(\leq, \vee)$ is a complete lattice and $\alpha^\sharp_\Upsilon \in \mathcal{D}^{\mathfrak{R}} \mapsto \mathcal{D}^\sharp$ then (48) defines a Galois connection; If $\alpha^\sharp_\Upsilon$ is surjective then it is a Galois surjection:*

$$\wp(\mathcal{D}^{\mathfrak{R}})(\subseteq) \underset{\alpha^\sharp_{\mathcal{D}}}{\overset{\gamma^\sharp_{\mathcal{D}}}{\leftrightarrows}} \mathcal{D}^\sharp(\leq) \tag{49}$$

A set of vectors in $\vec{\mathcal{D}}^{\Re}$ is approximated componentwise ($\vec{\mathcal{D}}^\sharp \stackrel{\text{def}}{=} \prod_{v \in \vec{v}} \mathcal{D}^\sharp$ is a complete lattice for the componentwise ordering $\vec{\nu} \stackrel{\rightarrow}{\leq} \vec{\nu}'$ if and only if $\forall v \in \vec{v} : \vec{\nu}[v] \leq \vec{\nu}'[v]$):

$$\begin{array}{l} \alpha^\sharp_{\vec{\mathcal{D}}}(\vec{V}) \stackrel{\text{def}}{=} \prod_{v \in \vec{v}} \alpha^\sharp_{\mathcal{D}}(\{\vec{\nu}[v] \mid \vec{\nu} \in \vec{V}\}) \\ \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}) \stackrel{\text{def}}{=} \{\vec{\nu}' \in \vec{\mathcal{D}}^{\Re} \mid \forall v \in \vec{v} : \vec{\nu}'[v] \in \gamma^\sharp_{\mathcal{D}}(\vec{\nu}[v])\} \end{array} \tag{50}$$

Proposition 15. *(49) and (50) imply:*

$$\wp(\vec{\mathcal{D}}^{\Re})(\subseteq) \underset{\alpha^\sharp_{\vec{\mathcal{D}}}}{\overset{\gamma^\sharp_{\vec{\mathcal{D}}}}{\leftrightarrows}} \vec{\mathcal{D}}^\sharp(\vec{\leq}) \tag{51}$$

Lemma 16. *If $\alpha^\sharp_\Upsilon$ is surjective then (48) and (50) imply that:*

$$\forall \vec{\nu}^\sharp \in \vec{\mathcal{D}}^\sharp : \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp) : \alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\}) = \vec{\nu}^\sharp \tag{52}$$

The lifting (16) of a Galois connection to higher-order functional spaces for the pointwise ordering can be generalized to relations as follows[21, 22, 23]:

$$\begin{array}{ll} \alpha^\sharp(\phi) \stackrel{\text{def}}{=} \lambda\vec{\nu}^\sharp \bullet \alpha^\sharp_{\mathcal{D}}(\{\nu \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp) : \langle \vec{\nu}, \nu \rangle \in \phi\}) & = \alpha^\sharp_{\mathcal{D}} \circ \phi^\star \circ \gamma^\sharp_{\vec{\mathcal{D}}} \\ \gamma^\sharp(\varphi) \stackrel{\text{def}}{=} \{\langle \vec{\nu}, \nu \rangle \mid \nu \in \gamma^\sharp_{\mathcal{D}} \circ \varphi \circ \alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\})\} & = {}^{\varrho}\gamma^\sharp_{\mathcal{D}} \circ \varphi \circ \alpha^{\sharp\, \imath}_{\vec{\mathcal{D}}} \end{array} \tag{53}$$

We observe that $\alpha^{\sharp\star}(\mathcal{F}^{\Re}) \subseteq \vec{\mathcal{D}}^\sharp \stackrel{\text{m}\vec{\leq}}{\longmapsto} \mathcal{D}^\sharp$ since $\alpha^\sharp(\phi) = \alpha^\sharp_{\mathcal{D}} \circ \phi^\star \circ \gamma^\sharp_{\vec{\mathcal{D}}}$ is the composition of monotonic functions and complete $\cup$-morphisms, hence monotonic. Therefore, we define:

$$\mathcal{F}^\sharp \stackrel{\text{def}}{=} \vec{\mathcal{D}}^\sharp \stackrel{\text{m}\vec{\leq}}{\longmapsto} \mathcal{D}^\sharp \tag{54}$$

which is a complete lattice $\mathcal{F}^\sharp(\leq, \lambda\vec{\nu}\bullet 0, \lambda\vec{\nu}\bullet 1, \vee, \wedge)$ for the pointwise ordering $\varphi \leq \varphi'$ if and only if $\forall \vec{\nu} \in \vec{\mathcal{D}}^\sharp : \varphi(\vec{\nu}) \leq \varphi'(\vec{\nu})$. We have:

Proposition 17. *(49), (51), (53) and (54) imply:*

$$\mathcal{F}^{\Re}(\subseteq) \underset{\alpha^\sharp}{\overset{\gamma^\sharp}{\leftrightarrows}} \mathcal{F}^\sharp(\leq) \tag{55}$$

Prop. 17 is independent of the particular definition of $\alpha^\sharp_\Upsilon$. Taking (47) into account, we have:

Proposition 18 Connection between the relational and strictness semantics. *If $\alpha^\sharp_\Upsilon$ is surjective (hence, in particular, if (47) holds) then:*

$$\mathcal{F}^{\Re}(\subseteq) \underset{\alpha^\sharp}{\overset{\gamma^\sharp}{\leftrightarrows}} \mathcal{F}^\sharp(\leq) \tag{56}$$

A vector of relations in $\vec{\mathcal{F}}^{\Re} = \prod_{f \in \vec{f}} \mathcal{F}^{\Re}$ is approximated componentwise in $\vec{\mathcal{F}}^\sharp \stackrel{\text{def}}{=} \prod_{f \in \vec{f}} \mathcal{F}^\sharp$ as follows:

$$\vec{\alpha}^\sharp(\vec{\phi}) \stackrel{\text{def}}{=} \prod_{f \in \vec{f}} \alpha^\sharp(\vec{\phi}[f]) \qquad \vec{\gamma}^\sharp(\vec{\varphi}) \stackrel{\text{def}}{=} \prod_{f \in \vec{f}} \gamma^\sharp(\vec{\varphi}[f]) \tag{57}$$

---

[21] The *image* of $X \subseteq S$ by relation $\rho \in \wp(S \times T)$ is $\rho^\star(X)$ where $\rho^\star \in \wp(S) \mapsto \wp(T)$ is defined by $\rho^\star(X) \stackrel{\text{def}}{=} \{t \mid \exists s \in X : \langle s, t \rangle \in \rho\}$. In particular for a total function $\varphi \in S \mapsto T$, $\varphi^\star(X) = \{\varphi(s) \mid s \in X\}$.

[22] The *projection* $\varphi^\imath \in S \mapsto T$ of $\varphi \in \wp(S) \mapsto T$ is defined by $\varphi^\imath(s) \stackrel{\text{def}}{=} \varphi(\{s\})$.

[23] If $f \in S \mapsto \wp(T)$ then ${}^{\varrho}f$ is the relation $\{\langle s, t \rangle \mid t \in f(s)\}$.

Proposition 19. *(56) and (57) imply:*

$$\vec{\mathcal{F}}^{\Re}(\vec{\subseteq}) \underset{\vec{\alpha}^\sharp}{\overset{\vec{\gamma}^\sharp}{\leftrightarrows}} \vec{\mathcal{F}}^\sharp(\vec{\leq}) \tag{58}$$

Using Prop. 5, we lift this approximation to function spaces as follows:

$$\alpha^\sharp_{\vec{\mathcal{F}}}(\vec{\phi}) \stackrel{\text{def}}{=} \vec{\alpha}^\sharp \circ \vec{\phi} \circ \vec{\gamma}^\sharp \qquad \gamma^\sharp_{\vec{\mathcal{F}}}(\vec{\varphi}) \stackrel{\text{def}}{=} \vec{\gamma}^\sharp \circ \vec{\varphi} \circ \vec{\alpha}^\sharp \tag{59}$$

Proposition 20. *(58) implies:*

$$\vec{\mathcal{F}}^{\Re} \stackrel{\text{m}\vec{\subseteq}}{\longmapsto} \vec{\mathcal{F}}^{\Re}(\vec{\subseteq}) \underset{\alpha^\sharp_{\vec{\mathcal{F}}}}{\overset{\gamma^\sharp_{\vec{\mathcal{F}}}}{\leftrightarrows}} \vec{\mathcal{F}}^\sharp \stackrel{\text{m}\vec{\leq}}{\longmapsto} \vec{\mathcal{F}}^\sharp(\vec{\leq}) \tag{60}$$

The same idea (19) can be applied to the approximation $e^\sharp \in \mathcal{E}^\sharp$ of the semantics $e^{\Re} \in \mathcal{E}^{\Re}$ of expressions e by defining $\mathcal{E}^\sharp \stackrel{\text{def}}{=} \vec{\mathcal{F}}^\sharp \stackrel{\text{m}\vec{\leq}}{\longmapsto} \vec{\mathcal{D}}^\sharp \stackrel{\text{m}\vec{\leq}}{\longmapsto} \mathcal{D}^\sharp$ and:

$$\alpha^\sharp_{\mathcal{E}}(\varepsilon) \stackrel{\text{def}}{=} \alpha^\sharp \circ \varepsilon \circ \vec{\gamma}^\sharp \qquad \gamma^\sharp_{\mathcal{E}}(\epsilon) \stackrel{\text{def}}{=} \gamma^\sharp \circ \epsilon \circ \vec{\alpha}^\sharp \tag{61}$$

Proposition 21. *(61), (56) and (58) imply:*

$$\mathcal{E}^{\Re}(\vec{\subseteq}) \underset{\alpha^\sharp_{\mathcal{E}}}{\overset{\gamma^\sharp_{\mathcal{E}}}{\leftrightarrows}} \mathcal{E}^\sharp(\vec{\leq}) \tag{62}$$

In order to approximate the semantics $\vec{e}^{\Re}$ of vectors $\vec{e}$ of expressions, we start by approximating elements of $\wp(\vec{\mathcal{D}}^{\Re} \times \vec{\mathcal{D}}^{\Re})$ by elements of the complete lattice $\vec{\mathcal{D}}^\sharp \stackrel{\text{m}\vec{\leq}}{\longmapsto} \vec{\mathcal{D}}^\sharp$ $(\vec{\leq}, \lambda\vec{\nu}\bullet\vec{0}, \lambda\vec{\nu}\bullet\vec{1}, \vec{\vee}, \vec{\wedge})$ for the componentwise ordering $\vec{\varphi} \vec{\leq} \vec{\varphi}'$ if and only if $\forall v \in \vec{v} : \vec{\varphi}[v] \leq \vec{\varphi}'[v]$ so that $\vec{0} = \prod_{v \in \vec{v}} 0$. We define:

$$\begin{array}{l} \alpha^\sharp_\pi(\vec{\phi}) \stackrel{\text{def}}{=} \lambda\vec{\nu}^\sharp\bullet\alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}' \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp) : \langle \vec{\nu}, \vec{\nu}' \rangle \in \vec{\phi}\}) = \alpha^\sharp_{\vec{\mathcal{D}}} \circ \vec{\phi}^\star \circ \gamma^\sharp_{\vec{\mathcal{D}}} \\ \gamma^\sharp_\pi(\vec{\varphi}) \stackrel{\text{def}}{=} \{\langle \vec{\nu}, \vec{\nu}' \rangle \mid \vec{\nu}' \in \gamma^\sharp_{\vec{\mathcal{D}}} \circ \vec{\varphi} \circ \alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\})\} \qquad = {}^{\varrho}\gamma^\sharp_{\vec{\mathcal{D}}} \circ \vec{\varphi} \circ \alpha^{\sharp\,\imath}_{\vec{\mathcal{D}}} \end{array} \tag{63}$$

Proposition 22.

$$\wp(\vec{\mathcal{D}}^{\Re} \times \vec{\mathcal{D}}^{\Re})(\subseteq) \underset{\alpha^\sharp_\pi}{\overset{\gamma^\sharp_\pi}{\leftrightarrows}} \vec{\mathcal{D}}^\sharp \stackrel{\text{m}\vec{\leq}}{\longmapsto} \vec{\mathcal{D}}^\sharp(\vec{\leq}) \tag{64}$$

Finally the approximation $\vec{e}^\sharp \in \vec{\mathcal{E}}^\sharp$ of the semantics $\vec{e}^{\Re} \in \vec{\mathcal{E}}^{\Re}$ of a vector $\vec{e}$ of expressions is defined in $\vec{\mathcal{E}}^\sharp \stackrel{\text{def}}{=} \vec{\mathcal{F}}^\sharp \stackrel{\text{m}\vec{\leq}}{\longmapsto} \vec{\mathcal{D}}^\sharp \stackrel{\text{m}\vec{\leq}}{\longmapsto} \vec{\mathcal{D}}^\sharp$ by:

$$\alpha^\sharp_{\vec{\mathcal{E}}}(\varepsilon) \stackrel{\text{def}}{=} \alpha^\sharp_\pi \circ \varepsilon \circ \vec{\gamma}^\sharp \qquad \gamma^\sharp_{\vec{\mathcal{E}}}(\epsilon) \stackrel{\text{def}}{=} \gamma^\sharp_\pi \circ \epsilon \circ \vec{\alpha}^\sharp \tag{65}$$

Proposition 23. *(65), (61) and (64) imply:*

$$\vec{\mathcal{E}}^{\Re}(\vec{\subseteq}) \underset{\alpha^\sharp_{\vec{\mathcal{E}}}}{\overset{\gamma^\sharp_{\vec{\mathcal{E}}}}{\leftrightarrows}} \vec{\mathcal{E}}^\sharp(\vec{\leq}) \tag{66}$$

Approximation and Computational Orderings Observe that $\vec{\leq}$ is the abstraction of the approximation ordering $\vec{\subseteq}$:

$$\vec{\varphi} \vec{\leq} \vec{\varphi}' \;\Leftrightarrow\; \vec{\gamma}^\sharp(\vec{\varphi}) \vec{\subseteq} \vec{\gamma}^\sharp(\vec{\varphi}') \tag{67}$$

The approximation and computational orderings differ in the concrete semantics but coincide in the abstract semantics[24] :

$$\begin{array}{rcl@{\qquad\qquad}rcl} \phi \subseteq \gamma^\sharp(\varphi) & \Leftrightarrow & \phi \sqsubseteq^\Re \gamma^\sharp(\varphi) & \vec{\phi} \vec{\subseteq} \vec{\gamma}^\sharp(\vec{\varphi}) & \Leftrightarrow & \vec{\phi} \vec{\sqsubseteq}^\Re \vec{\gamma}^\sharp(\vec{\varphi}) \\ \varphi \leq \varphi' & \Leftrightarrow & \gamma^\sharp(\varphi) \sqsubseteq^\Re \gamma^\sharp(\varphi') & \vec{\varphi} \vec{\leq} \vec{\varphi}' & \Leftrightarrow & \vec{\gamma}^\sharp(\vec{\varphi}) \vec{\sqsubseteq}^\Re \vec{\gamma}^\sharp(\vec{\varphi}') \end{array} \tag{68}$$

An immediate consequence is that we have the following Galois surjections:

$$\mathcal{F}^\Re(\sqsubseteq^\Re) \underset{\alpha^\sharp}{\overset{\gamma^\sharp}{\rightleftarrows}} \mathcal{F}^\sharp(\leq) \qquad \vec{\mathcal{F}}^\Re(\vec{\sqsubseteq}^\Re) \underset{\vec{\alpha}^\sharp}{\overset{\vec{\gamma}^\sharp}{\rightleftarrows}} \vec{\mathcal{F}}^\sharp(\vec{\leq}) \tag{69}$$

Forward Dependence-Sensitive Strictness Semantics The abstract semantics of a program $\prod_{f \in \vec{f}} f\vec{v} \equiv \vec{F}[f]$ is specified by $\vec{f}^\sharp$ which we would like to define such that $\vec{f}^\Re = \mathrm{lfp}^{\vec{\sqsubseteq}^\Re}_{\vec{\perp}^\Re} \vec{F}^\Re \vec{\subseteq} \vec{\gamma}^\sharp(\vec{f}^\sharp)$. Corollary 9 provides a method for refining this specification as $\vec{f}^\sharp \stackrel{\text{def}}{=} \vec{\sqcup}^\sharp_{n \in \mathbb{N}} \vec{F}^{\sharp\, n}(\vec{\perp}^\sharp)$ by defining the infimum $\vec{\perp}^\sharp \stackrel{\text{def}}{=} \vec{\alpha}^\sharp(\vec{\perp}^\Re)$, the inductive join $\vec{\sqcup}^\sharp_{i \in \Delta} \vec{\varphi}_i \stackrel{\text{def}}{=} \vec{\alpha}^\sharp(\vec{\sqcup}^\Re_{i \in \Delta} \vec{\gamma}^\sharp(\vec{\varphi}_i))$ and the semantics function $\vec{F}^\sharp$ such that $\vec{\alpha}^\sharp \circ \vec{F}^\Re \circ \vec{\gamma}^\sharp \vec{\leq}^\sharp \vec{F}^\sharp$. Observe that monotonicity and finiteness of the lattice imply continuity and more precisely convergence below ω in (8). We now simplify these formulæ in order to get the formal definition of $\vec{\perp}^\sharp$, $\vec{\sqcup}^\sharp$ and $\vec{F}^\sharp$. The hand-computation presents no difficulties. It is provided just to show that the method is mathematically constructive. The strictness semantics, that is the formal specification of the abstract interpreter for strictness analysis, is entirely determined by the choice of the collecting semantics in Sec. 2.4 and the choice of the approximations (58) and (69). The derivation of $\vec{\perp}^\sharp$, $\vec{\sqcup}^\sharp$ and $\vec{F}^\sharp$ is nothing else than a refinement of this formal specification. This computation is certainly amenable to automation. Moreover, the replacement/simplification/definition introduction strategies are very similar for different abstract domains so that proof strategies should be definable for a given relational semantics so as to guide the automatic derivation of the abstract semantics.

– $\vec{\perp}^\sharp = \vec{\alpha}^\sharp(\vec{\perp}^\Re)$ by definition of $\vec{\perp}^\sharp$
$= \prod_{f \in \vec{f}} \alpha^\sharp(\vec{\perp}^\Re[f])$ by (57)
$= \prod_{f \in \vec{f}} \alpha^\sharp(\vec{Y}^\Omega_\perp \times \{\perp\})$ by (35)
$= \prod_{f \in \vec{f}} \lambda\vec{\nu}^\sharp \cdot \alpha^\sharp_\mathcal{D}(\{\nu \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp) : \langle \vec{\nu}, \nu \rangle \in (\vec{Y}^\Omega_\perp \times \{\perp\})\})$ by (53)
$= \prod_{f \in \vec{f}} \lambda\vec{\nu}^\sharp \cdot \alpha^\sharp_\mathcal{D}(\{\perp \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)\})$ since $\vec{Y}^\Omega_\perp \neq \emptyset$
$= \prod_{f \in \vec{f}} \lambda\vec{\nu}^\sharp \cdot \alpha^\sharp_\mathcal{D}(\{\perp \mid \exists \vec{\nu} \in \{\vec{\nu}' \in \vec{\mathcal{D}}^\Re \mid \forall v \in \vec{v} : \vec{\nu}'[v] \in \gamma^\sharp_\mathcal{D}(\vec{\nu}^\sharp[v])\}\})$ by (50)
$= \prod_{f \in \vec{f}} \lambda\vec{\nu}^\sharp \cdot \alpha^\sharp_\mathcal{D}(\{\perp \mid \exists \vec{\nu} \in \vec{\mathcal{D}}^\Re : \forall v \in \vec{v} : \vec{\nu}[v] \in \{\nu' \mid \alpha^\sharp_Y(\nu') \leq \vec{\nu}^\sharp[v]\}\})$ by (48)
$= \prod_{f \in \vec{f}} \lambda\vec{\nu}^\sharp \cdot \alpha^\sharp_\mathcal{D}(\{\perp\})$ by (47), choosing $\vec{\nu} = \lambda v \cdot \perp$
$= \prod_{f \in \vec{f}} \lambda\vec{\nu}^\sharp \cdot \alpha^\sharp_Y(\perp)$ by (48) and definition of lubs
$= \prod_{f \in \vec{f}} \lambda\vec{\nu}^\sharp \cdot 0$ by (47)

[24] whereas in [CC77a] they also coincide in the semantics collecting invariance properties, where non-termination is ignored, and therefore is obtained by further application of the abstraction $\alpha(V) = V \cup \{\perp\}$ to $\mathcal{D}^\Re$.

– $\vec{\sqcup}^\sharp_{i\in\Delta}\vec{\varphi}_i = \vec{\alpha}^\sharp(\sqcup^\Re_{i\in\Delta}\vec{\gamma}^\sharp(\vec{\varphi}_i))$ by definition of $\vec{\sqcup}$
$= \vec{\vee}^\sharp_{i\in\Delta}\vec{\alpha}^\sharp(\vec{\gamma}^\sharp(\vec{\varphi}_i))$ since $\vec{\alpha}^\sharp$ is a complete $\sqcup^\Re$-morphism by (69)
$= \vec{\vee}^\sharp_{i\in\Delta}\vec{\varphi}_i$ since $\vec{\alpha}^\sharp \circ \vec{\gamma}^\sharp$ is the identity by (69)

– For a program $\prod_{f\,\epsilon\,\vec{f}} f\vec{v} \equiv \vec{F}[f]$, the abstract semantics function $\vec{F}^\sharp$ is defined as an upper approximation of $\vec{\alpha}^\sharp \circ \vec{F}^\Re \circ \vec{\gamma}^\sharp$. We have:

$\vec{\alpha}^\sharp \circ \vec{F}^\Re \circ \vec{\gamma}^\sharp = \lambda\vec{\varphi}\bullet\vec{\alpha}^\sharp\left(\prod_{f\,\epsilon\,\vec{f}}\vec{F}[f]^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi})]\!]\right)$ by (39)
$= \lambda\vec{\varphi}\bullet\prod_{f\,\epsilon\,\vec{f}}\alpha^\sharp(\vec{F}[f]^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi})]\!])$ by (57)
$= \lambda\vec{\varphi}\bullet\prod_{f\,\epsilon\,\vec{f}}\alpha^\sharp_{\mathcal{E}}(\vec{F}[f]^\Re)[\![\vec{\varphi}]\!]$ by (61)

This suggest the definition $\vec{F}^\sharp \stackrel{\text{def}}{=} \lambda\vec{\varphi}\bullet\prod_{f\,\epsilon\,\vec{f}}\vec{F}[f]^\sharp[\![\vec{\varphi}]\!]$ with the condition that for all $f\ \epsilon\ \vec{f}$, we have $\vec{F}[f]^\sharp \geq \alpha^\sharp_{\mathcal{E}}(\vec{F}[f]^\Re)$. This condition is obviously satisfied by generalization to all expressions as $e^\sharp \geq \alpha^\sharp_{\mathcal{E}}(e^\Re)$. We proceed by structural induction on the syntax of e. The basis corresponds to the cases when e is reduced to a constant k or variable v:

• $k^\sharp[\![\vec{\varphi}]\!] = \alpha^\sharp_{\mathcal{E}}(k^\Re)[\![\vec{\varphi}]\!] = \alpha^\sharp \circ k^\Re \circ \vec{\gamma}^\sharp[\![\vec{\varphi}]\!] = \alpha^\sharp(\{\langle\vec{\nu},\ \underline{k}\rangle \mid \vec{\nu} \in \vec{\mathcal{D}}^\Re\}) = \lambda\vec{\nu}^\sharp\bullet\alpha^\sharp_{\mathcal{D}}(\{\underline{k} \mid \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)\}) = \lambda\vec{\nu}^\sharp\bullet\bigvee\{\alpha^\sharp_{\Upsilon}(\underline{k}) \mid \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)\} = \lambda\vec{\nu}^\sharp\bullet 1$, by (61), (40), (53), (48) and (47) since $\underline{k} \in \Upsilon$ and $\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu})$ is not empty.

• $v^\sharp[\![\vec{\varphi}]\!] = \alpha^\sharp_{\mathcal{E}}(v^\Re)[\![\vec{\varphi}]\!] = \alpha^\sharp \circ v^\Re \circ \vec{\gamma}^\sharp[\![\vec{\varphi}]\!] = \alpha^\sharp(\{\langle\vec{\nu},\ \vec{\nu}[v]\rangle \mid \vec{\nu} \in \vec{\mathcal{D}}^\Re\})$ by (61) and (41). By (53) this is equal to $\lambda\vec{\nu}^\sharp\bullet\bigvee\{\alpha^\sharp_{\Upsilon}(\vec{\nu}[v]) \mid \exists\vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)\}$, hence, by (48) and definition (50) of $\gamma^\sharp_{\vec{\mathcal{D}}}$, to $\lambda\vec{\nu}^\sharp\bullet\bigvee\{\alpha^\sharp_{\Upsilon}(\nu) \mid \nu \in \gamma^\sharp_{\mathcal{D}}(\vec{\nu}^\sharp[v])\}$. By (48), this is equal to $\lambda\vec{\nu}^\sharp\bullet\alpha^\sharp_{\mathcal{D}}(\gamma^\sharp_{\mathcal{D}}(\vec{\nu}^\sharp[v])) = \lambda\vec{\nu}^\sharp\bullet\vec{\nu}^\sharp[v]$ since, by (49), $\alpha^\sharp_{\mathcal{D}} \circ \gamma^\sharp_{\mathcal{D}}$ is the identity.

For the induction step, we must prove that $e^\sharp \geq \alpha^\sharp_{\mathcal{E}}(e^\Re)$ where e is $b\vec{e}$, $f\vec{e}$ or $(e_1 \rightarrow e_2,\ e_3)$. We can assume, by induction hypothesis, that $e^\sharp_i \geq \alpha^\sharp_{\mathcal{E}}(e^\Re_i)$, i =1, 2, 3 and that for all $v\ \epsilon\ \vec{e}$, $\vec{e}[v]^\sharp \geq \alpha^\sharp_{\mathcal{E}}(\vec{e}[v]^\Re)$.

• We define $\vec{e}^\sharp[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp \stackrel{\text{def}}{=} \prod_{v\,\epsilon\,\vec{e}}\vec{e}[v]^\sharp[\![\vec{\varphi}]\!]\vec{\nu}$ and first prove that the induction hypothesis implies that $\vec{e}^\sharp \stackrel{\rightarrow}{\geq} \alpha^\sharp_{\vec{\mathcal{E}}}(\vec{e}^\Re)$.

By (65), $\alpha^\sharp_{\vec{\mathcal{E}}}(\vec{e}^\Re) = \alpha^\sharp_{\pi} \circ \vec{e}^\Re \circ \vec{\gamma}^\sharp = \lambda\vec{\varphi}^\sharp\bullet\alpha^\sharp_{\pi}(\vec{e}^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!])$ which, by (63), equals $\lambda\vec{\varphi}^\sharp\bullet\alpha^\sharp_{\vec{\mathcal{D}}} \circ \left(\vec{e}^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\right)^\star \circ \gamma^\sharp_{\vec{\mathcal{D}}} = \lambda\vec{\varphi}^\sharp\bullet\lambda\vec{\nu}^\sharp\bullet\alpha^\sharp_{\vec{\mathcal{D}}}\left(\left(\vec{e}^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\right)^\star\left(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)\right)\right)$. By (50), this is equal to $\lambda\vec{\varphi}^\sharp\bullet\lambda\vec{\nu}^\sharp\bullet\prod_{v\,\epsilon\,\vec{v}}\alpha^\sharp_{\mathcal{D}}(\{\vec{\nu}'[v] \mid \vec{\nu}' \in \left(\vec{e}^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\right)^\star\left(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)\right)\}) =$ LHS. By definition of the image *, $\vec{\nu}\,' \in \left(\vec{e}^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\right)^\star\left(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)\right)$ is equivalent to $\exists\vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp) : \langle\vec{\nu},\ \vec{\nu}\,'\rangle \in \vec{e}^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]$ hence, by definition (42) of $\vec{e}^\Re$, to $\exists\vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp) : \forall v\ \epsilon\ \vec{e} : \langle\vec{\nu},\ \vec{\nu}\,'[v]\rangle \in \vec{e}[v]^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]$ which, for all $v\ \epsilon\ \vec{e}$ implies $\exists\vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp) : \langle\vec{\nu},\ \vec{\nu}\,'[v]\rangle \in \vec{e}[v]^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]$. By (49), $\alpha^\sharp_{\mathcal{D}}$ is monotonic, whence LHS $\stackrel{\rightarrow}{\leq} \lambda\vec{\varphi}^\sharp\bullet\lambda\vec{\nu}^\sharp\bullet\prod_{v\,\epsilon\,\vec{v}}\alpha^\sharp_{\mathcal{D}}(\{\vec{\nu}\,'[v] \mid \exists\vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp) : \langle\vec{\nu},\ \vec{\nu}\,'[v]\rangle \in \vec{e}[v]^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\})$. This is $\lambda\vec{\varphi}^\sharp\bullet\lambda\vec{\nu}^\sharp\bullet\prod_{v\,\epsilon\,\vec{v}}\alpha^\sharp_{\mathcal{D}}(\{\nu \mid \exists\vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp) : \langle\vec{\nu},\ \nu\rangle \in \vec{e}[v]^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\})$, if we let ν be $\vec{\nu}\,'[v]$. By definition (53) of $\alpha^\sharp$, this is equal to $\lambda\vec{\varphi}^\sharp\bullet\lambda\vec{\nu}^\sharp\bullet\prod_{v\,\epsilon\,\vec{v}}\alpha^\sharp(\vec{e}[v]^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!])\vec{\nu}$, hence by definition (61) of $\alpha^\sharp_{\mathcal{E}}$ to $\lambda\vec{\varphi}^\sharp\bullet\lambda\vec{\nu}^\sharp\bullet\prod_{v\,\epsilon\,\vec{v}}\alpha^\sharp_{\mathcal{E}}(\vec{e}[v]^\Re)[\![\vec{\varphi}^\sharp]\!]\vec{\nu} = \prod_{v\,\epsilon\,\vec{v}}\alpha^\sharp_{\mathcal{E}}(\vec{e}[v]^\Re) \stackrel{\rightarrow}{\leq} \prod_{v\,\epsilon\,\vec{v}}\vec{e}[v]^\sharp = \vec{e}^\sharp$ by induction hypothesis and definition of $\vec{e}^\sharp$.

• If e is $b\vec{e}$ then we must define $b\vec{e}^\sharp$ such that $b\vec{e}^\sharp \geq \alpha^\sharp_{\mathcal{E}}(b\vec{e}^\Re)$. We do this by formal hand-computation. This consists is expanding the term $\alpha^\sharp_{\mathcal{E}}(b\vec{e}^\Re)$ in order

to let appear the subterm $\alpha^\sharp_{\vec{\mathcal{E}}}(\vec{e}^{\,\Re})$. By (61), $\alpha^\sharp_{\mathcal{E}}(b\vec{e}^{\,\Re})$ is equal to $\alpha^\sharp \circ b\vec{e}^{\,\Re} \circ \vec{\gamma}^\sharp = \lambda\vec{\varphi}^\sharp.\alpha^\sharp(b\vec{e}^{\,\Re}[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!])$. By (53), this is equal to $\lambda\vec{\varphi}^\sharp.\alpha^\sharp_{\mathcal{D}} \circ (b\vec{e}^{\,\Re}[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!])^\star \circ \gamma^\sharp_{\vec{\mathcal{D}}} = \lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\alpha^\sharp_{\mathcal{D}}\big((b\vec{e}^{\,\Re}[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!])^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp))\big)$. By definition (43) of $b\vec{e}^{\,\Re}$, this is equal to $\lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\alpha^\sharp_{\mathcal{D}}\big((\vec{e}^{\,\Re}[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!] \circ b^\Re)^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp))\big)$ and, since $(\rho \circ \varrho)^\star(X) = \varrho^\star(\rho^\star(X))$, to: $\lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\alpha^\sharp_{\mathcal{D}}\big(b^{\Re\star}(\vec{e}^{\,\Re}[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp))\big)$ = LHS. By (51), $\gamma^\sharp_{\vec{\mathcal{D}}} \circ \alpha^\sharp_{\vec{\mathcal{D}}}$ is extensive so that $b^{\Re\star}(\vec{e}^{\,\Re}[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp))) \subseteq b^{\Re\star}(\gamma^\sharp_{\vec{\mathcal{D}}}(\alpha^\sharp_{\vec{\mathcal{D}}}(\vec{e}^{\,\Re}[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)))))$. Moreover $\alpha^\sharp_{\mathcal{D}}$ is monotonic so that LHS $\leq \lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\alpha^\sharp_{\mathcal{D}}\big(b^{\Re\star}(\gamma^\sharp_{\vec{\mathcal{D}}}(\alpha^\sharp_{\vec{\mathcal{D}}} \circ \vec{e}^{\,\Re}[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]^\star \circ \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)))\big) = \lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\alpha^\sharp_{\mathcal{D}} \circ b^{\Re\star} \circ \gamma^\sharp_{\vec{\mathcal{D}}}(\alpha^\sharp_{\pi}(\vec{e}^{\,\Re}[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!])(\vec{\nu}^\sharp))$ by definition (63) of $\alpha^\sharp_{\pi}$. By (53), this is $\lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\alpha^\sharp(b^\Re)(\alpha^\sharp_{\pi} \circ \vec{e}^{\,\Re} \circ \vec{\gamma}^\sharp(\vec{\varphi}^\sharp)\vec{\nu}^\sharp)$, which, by (65), is equal to $\lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\alpha^\sharp(b^\Re)(\alpha^\sharp_{\vec{\mathcal{E}}}(\vec{e}^{\,\Re})[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp)$ = LHS. This suggests to define $b^\sharp$ as $\alpha^\sharp(b^\Re)$ but in order to allow for further approximations, we assume that $b^\sharp \in \vec{\mathcal{D}}^\sharp \overset{\mathrm{m}\vec{\leq}^\sharp}{\longmapsto} \mathcal{D}^\sharp$ is such that $b^\sharp \geq^\sharp \alpha^\sharp(b^\Re)$. By definition of the pointwise ordering $\leq^\sharp$, it follows that LHS $\leq^\sharp \lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.b^\sharp(\alpha^\sharp_{\vec{\mathcal{E}}}(\vec{e}^{\,\Re})[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp) \leq^\sharp \lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.b^\sharp(\vec{e}^{\,\sharp}[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp)$ by induction hypothesis $\vec{e}^{\,\sharp} \vec{\geq} \alpha^\sharp_{\vec{\mathcal{E}}}(\vec{e}^{\,\Re})$ and monotonicity of $b^\sharp$. We define $b\vec{e}^{\,\sharp} \overset{\text{def}}{=} \lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.b^\sharp(\vec{e}^{\,\sharp}[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp)$ so that we have proved that $b\vec{e}^{\,\sharp} \geq \alpha^\sharp_{\mathcal{E}}(b\vec{e}^{\,\Re})$.

- If e is $f\vec{e}$, $\alpha^\sharp_{\mathcal{E}}(f\vec{e}^{\,\Re}) = \lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\alpha^\sharp_{\mathcal{D}}\big((f\vec{e}^{\,\Re}[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!])^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp))\big)$, as above. By definition (45) of $f\vec{e}^{\,\Re}$, this is: $\lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\alpha^\sharp_{\mathcal{D}}\big((\vec{e}^{\,\Re}[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!] \circ \vec{\gamma}^\sharp(\vec{\varphi}^\sharp)[f])^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp))\big)$. As above, this is: $\lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\alpha^\sharp(\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)[f])(\alpha^\sharp_{\vec{\mathcal{E}}}(\vec{e}^{\,\Re})[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp)$. By definition (57) of $\vec{\gamma}^\sharp$, this is equal to $\lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\alpha^\sharp(\gamma^\sharp(\vec{\varphi}^\sharp[f]))(\alpha^\sharp_{\vec{\mathcal{E}}}(\vec{e}^{\,\Re})[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp)$. But $\alpha^\sharp \circ \gamma^\sharp$ is the identity, a characteristic property of the Galois surjection (56). We get $\lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\vec{\varphi}^\sharp[f](\alpha^\sharp_{\vec{\mathcal{E}}}(\vec{e}^{\,\Re})[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp) \leq^\sharp \lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\vec{\varphi}^\sharp[f](\vec{e}^{\,\sharp}[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp)$ by induction hypothesis $\vec{e}^{\,\sharp} \vec{\geq} \alpha^\sharp_{\vec{\mathcal{E}}}(\vec{e}^{\,\Re})$ and monotonicity of $\vec{\varphi}^\sharp[f] \in \vec{\mathcal{D}}^\sharp \overset{\mathrm{m}\vec{\leq}^\sharp}{\longmapsto} \mathcal{D}^\sharp$. We define $f\vec{e}^{\,\sharp} \overset{\text{def}}{=} \lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\vec{\varphi}^\sharp[f](\vec{e}^{\,\sharp}[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp)$ so that we have proved that $f\vec{e}^{\,\sharp} \geq \alpha^\sharp_{\mathcal{E}}(f\vec{e}^{\,\Re})$.

- It e is $(e_1 \to e_2, e_3)$ then $\alpha^\sharp_{\mathcal{E}}((e_1 \to e_2, e_3)^\Re)$ can be shown, as above, to be equal to $\lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\alpha^\sharp_{\mathcal{D}}\big(((e_1 \to e_2, e_3)^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!])^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp))\big)$. By definition (44) of $(e_1 \to e_2, e_3)^\Re$, this is:

$$\lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\alpha^\sharp_{\mathcal{D}}\Big(\big(\{\langle\vec{\nu},\,\bot\rangle \mid \langle\vec{\nu},\,\bot\rangle \in e_1^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\}$$
$$\cup\,\{\langle\vec{\nu},\,\Omega\rangle \mid \exists\nu \in \Upsilon^\Omega - \{\mathrm{tt},\mathrm{ff}\} : \langle\vec{\nu},\,\nu\rangle \in e_1^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\}$$
$$\cup\,\{\langle\vec{\nu},\,\nu\rangle \mid \langle\vec{\nu},\,\mathrm{tt}\rangle \in e_1^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!] \wedge \langle\vec{\nu},\,\nu\rangle \in e_2^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\}$$
$$\cup\,\{\langle\vec{\nu},\,\nu\rangle \mid \langle\vec{\nu},\,\mathrm{ff}\rangle \in e_1^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!] \wedge \langle\vec{\nu},\,\nu\rangle \in e_3^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\}\big)^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp))\Big)$$

But $(\rho \cup \varrho)^\star(X) = \rho^\star(X) \cup \varrho^\star(X)$ and by (49), $\alpha^\sharp_{\mathcal{D}}$ is the lower adjoint of a Galois connection, hence a complete $\cup$-morphism so that this is:

$$\begin{aligned}\lambda\vec{\varphi}^\sharp.\lambda\vec{\nu}^\sharp.\ \ &\alpha^\sharp_{\mathcal{D}}\big(\{\langle\vec{\nu},\,\bot\rangle \mid \langle\vec{\nu},\,\bot\rangle \in e_1^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\}^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp))\big)\\ \vee\,&\alpha^\sharp_{\mathcal{D}}\big(\{\langle\vec{\nu},\,\Omega\rangle \mid \exists\nu \in \Upsilon^\Omega - \{\mathrm{tt},\mathrm{ff}\} : \langle\vec{\nu},\,\nu\rangle \in e_1^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\}^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp))\big)\\ \vee\,&\alpha^\sharp_{\mathcal{D}}\big(\{\langle\vec{\nu},\,\nu\rangle \mid \langle\vec{\nu},\,\mathrm{tt}\rangle \in e_1^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!] \wedge \langle\vec{\nu},\,\nu\rangle \in e_2^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\}^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp))\big)\\ \vee\,&\alpha^\sharp_{\mathcal{D}}\big(\{\langle\vec{\nu},\,\nu\rangle \mid \langle\vec{\nu},\,\mathrm{ff}\rangle \in e_1^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!] \wedge \langle\vec{\nu},\,\nu\rangle \in e_3^\Re[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\}^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp))\big)\end{aligned}$$

By definition $\rho^\star(X) \stackrel{\text{def}}{=} \{\nu \mid \exists \vec{\nu} \in X : \langle \vec{\nu}, \nu \rangle \in \rho\}$, this is LHS =

$$\begin{array}{ll}\lambda\vec{\varphi}^\sharp{\cdot}\lambda\vec{\nu}^\sharp{\cdot} & \alpha^\sharp_{\mathcal{D}}(\{\bot \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{B}}}(\vec{\nu}^\sharp) : \langle \vec{\nu}, \bot\rangle \in e^\Re_1[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\}) \\ & \vee \alpha^\sharp_{\mathcal{D}}(\{\Omega \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{B}}}(\vec{\nu}^\sharp) : \exists \nu \in \Upsilon^\Omega - \{\mathrm{tt}, \mathrm{ff}\} : \langle \vec{\nu}, \nu\rangle \in e^\Re_1[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\}) \\ & \vee \alpha^\sharp_{\mathcal{D}}(\{\nu \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{B}}}(\vec{\nu}^\sharp) : \langle \vec{\nu}, \mathrm{tt}\rangle \in e^\Re_1[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!] \wedge \langle \vec{\nu}, \nu\rangle \in e^\Re_2[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\}) \\ & \vee \alpha^\sharp_{\mathcal{D}}(\{\nu \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{B}}}(\vec{\nu}^\sharp) : \langle \vec{\nu}, \mathrm{ff}\rangle \in e^\Re_1[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!] \wedge \langle \vec{\nu}, \nu\rangle \in e^\Re_3[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\})\end{array}$$

We now examine each subterm of LHS in turn.

$*$ In the subterm $\alpha^\sharp_{\mathcal{D}}(\{\bot \mid C\})$, the condition $C = \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{B}}}(\vec{\nu}^\sharp) : \langle \vec{\nu}, \bot\rangle \in e^\Re_1[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]$ is either true and, by (48) and (47), this is $\alpha^\sharp_{\mathcal{D}}(\{\bot\}) = \alpha^\sharp_{\Upsilon}(\bot) = 0$ or else C is false and this subterm is $\alpha^\sharp_{\mathcal{D}}(\emptyset) = \vee\emptyset = 0$. We have $0 \vee \nu^\sharp = \nu^\sharp$ since 0 is the infimum of $\mathcal{D}^\sharp$ so that this subterm can be eliminated from LHS.

$*$ The situation is the same for term $\alpha^\sharp_{\mathcal{D}}(\{\Omega \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{B}}}(\vec{\nu}^\sharp) : \exists \nu \in \Upsilon^\Omega - \{\mathrm{tt}, \mathrm{ff}\} : \langle \vec{\nu}, \nu\rangle \in e^\Re_1[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\})$ since $\alpha^\sharp_{\Upsilon}(\Omega) = 0$.

$*$ For the term $T_2 = \alpha^\sharp_{\mathcal{D}}(\{\nu \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{B}}}(\vec{\nu}^\sharp) : \langle \vec{\nu}, \mathrm{tt}\rangle \in e^\Re_1[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!] \wedge \langle \vec{\nu}, \nu\rangle \in e^\Re_2[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\})$, we proceed by cases:

- If $e^\sharp_1[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp = 0$ then $\alpha^\sharp_{\mathcal{E}}(e^\Re_1)[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp = 0$ since $e^\sharp_1 \geq \alpha^\sharp_{\mathcal{E}}(e^\Re_1)$ by induction hypothesis. It follows, by (61), that $\alpha^\sharp(e^\Re_1(\vec{\gamma}^\sharp[\![\vec{\varphi}^\sharp]\!]))(\vec{\nu}^\sharp) = 0$, whence, by (53) that $\alpha^\sharp_{\mathcal{D}}(\{\nu \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{B}}}(\vec{\nu}^\sharp) : \langle \vec{\nu}, \nu\rangle \in e^\Re_1(\vec{\gamma}^\sharp[\![\vec{\varphi}^\sharp]\!])\}) = 0$. From (47) and (48) we derive that for all $\vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{B}}}(\vec{\nu}^\sharp)$ such that $\langle \vec{\nu}, \nu\rangle \in e^\Re_1(\vec{\gamma}^\sharp[\![\vec{\varphi}^\sharp]\!])$, we have $\nu \in \{\bot, \Omega\}$ and therefore ν cannot be tt. In this case $T_2 = \alpha^\sharp_{\mathcal{D}}(\emptyset) = \vee\emptyset = 0$.
- If $e^\sharp_1[\![\vec{\varphi}^\sharp]\!]\vec{\nu} = 1$ then, by monotonicity of $\alpha^\sharp_{\mathcal{D}}$, we have $T_2 \leq \alpha^\sharp_{\mathcal{D}}(\{\nu \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{B}}}(\vec{\nu}^\sharp) : \langle \vec{\nu}, \nu\rangle \in e^\Re_2[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\})$ which, by (53) is equal to $\alpha^\sharp(e^\Re_2[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!])(\vec{\nu}^\sharp)$ hence, by (61) to $\alpha^\sharp_{\mathcal{E}}(e^\Re_2)[\![\vec{\varphi}^\sharp]\!])\vec{\nu}^\sharp, \leq e^\sharp_2[\![\vec{\varphi}^\sharp]\!])\vec{\nu}^\sharp$ by induction hypothesis.
- We conclude that $T_2 \leq e^\sharp_1[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp \wedge e^\sharp_2[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp$ which equals 0 when $e^\sharp_1[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp = 0$ and else is $e^\sharp_2[\![\vec{\varphi}^\sharp]\!])\vec{\nu}^\sharp$.

$*$ The same way, we have $\alpha^\sharp_{\mathcal{D}}(\{\nu \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{B}}}(\vec{\nu}^\sharp) : \langle \vec{\nu}, \mathrm{ff}\rangle \in e^\Re_1[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!] \wedge \langle \vec{\nu}, \nu\rangle \in e^\Re_3[\![\vec{\gamma}^\sharp(\vec{\varphi}^\sharp)]\!]\}) \leq e^\sharp_1[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp \wedge e^\sharp_3[\![\vec{\varphi}^\sharp]\!]\vec{\nu}^\sharp$.

Taking all cases into account, $(e_1 \rightarrow e_2, e_3)^\sharp$ can be defined as $(e^\sharp_1 \wedge e^\sharp_2) \vee (e^\sharp_1 \wedge e^\sharp_3) = e^\sharp_1 \wedge (e^\sharp_2 \vee e^\sharp_3)$. Observe that this is not the best possible approximation since for example, by taking values into account, we could have: $(\mathrm{true} \rightarrow e_2, e_3)^\sharp = e^\sharp_2$ and $(\mathrm{false} \rightarrow e_2, e_3)^\sharp = e^\sharp_3$.

• In conclusion, the definition of $\vec{f}^\sharp$ is given at Fig. 2. According to Cor. 9, our definition of $\vec{f}^\sharp$ is safe by construction:

Proposition 24 Connection of Mycroft's forward dependence-sensitive strictness semantics $\vec{f}^\sharp$ with the relational semantics $\vec{f}^\Re$ of program $\prod_{f \in \vec{f}} f\vec{v} \equiv \vec{F}[f]$.

$$\vec{\alpha}^\sharp(\vec{f}^\Re) = \vec{\alpha}^\sharp(\mathrm{lfp}^{\stackrel{\cdot}{\sqsubseteq}^\Re}_{\bot^\Re} \vec{F}^\Re) \stackrel{\cdot}{\leq} \vec{f}^\sharp = \mathrm{lfp}^{\stackrel{\cdot}{\leq}}_{\bot^\sharp} \vec{F}^\sharp$$

Proposition 25 Safeness of Mycroft's strictness analysis method. *If $\vec{\nu}^\sharp = \prod_{v \in \vec{v}}(v \in I \rightarrow 0, 1)$ and $\vec{f}^\sharp[f]\vec{\nu}^\sharp = 0$ then $\vec{f}^\Re[f]$ is strict in its parameters I.*

Comments on Mycroft's Strictness Analysis Method One strength of Mycroft's strictness analysis method is that it is dependence-sensitive and therefore can detect dependencies between arguments. For example, if we define $f(x, y, z) \equiv (x \rightarrow y, z)$ then we find that $f^\sharp(1, 0, 1) = 1$ and $f^\sharp(1, 1, 0) = 1$ so that f is neither strict in y nor in z but $f^\sharp(1, 0, 0) = 0$ so that it is jointly strict in y and z. More precisely, Sekar, Mishra and Ramakrishnan have proved [SMR91] that "Mycroft's method will deduce a strictness property for program P if and only the property is independent of any constant appearing in any evaluation of P". Hence the only way to improve its power is to take values of variables into account.

In practice, two methods have been proposed in [CC77c] for solving iteratively the system of equations $\prod_{f \in \vec{f}} \vec{f}[f]^\sharp \vec{\nu} = \vec{F}^\sharp(\vec{f}^\sharp)[f]\vec{\nu}$ which are more efficient than the naïve Jacobi iteration $\vec{F}^{\sharp n}(\vec{\bot}^\sharp)$, $n \geq 0$:

(a) We can use first-order chaotic iterations [CC77b] for $\vec{f}^\sharp = \vec{F}^\sharp(\vec{f}^\sharp)$ (using a tabular representation of functions $\vec{f}^\sharp$ or a symbolic representation by e.g. BDDs [Bry86]); This method is expensive since $\vec{f}^\sharp[f](\nu_1, \ldots, \nu_n)$ is computed for all $f \in \vec{f}$, $\nu_i \in \{0, 1\}$ and $i \in \{1, \ldots, n\}$ so that there are $|\mathrm{Dom}\, \vec{f}|.2^n$ possibilities[25].

(b) It is preferable to use second-order chaotic iterations introduced in [CC77c], page 265[26] to compute the subset of $\vec{f}^\sharp[f']\vec{\nu}'$ satisfying $\vec{f}^\sharp[f']\vec{\nu}' = \vec{F}^\sharp(\vec{f}^\sharp)[f']\vec{\nu}'$ which are needed to answer a given strictness question $\vec{f}^\sharp[f]\vec{\nu} = 0$. Again sets of arguments can be represented using BDDs [Bry86] and their refinements.

However, in both cases, exponential worst cases cannot be avoided since Hudak and Young have proved a conjecture due to Albert Meyer stating that "the problem of first-order strictness analysis is complete in deterministic exponential time" [HY86].

3.3 Forward Dependence-Free Strictness Analysis

Thomas Johnsson proposed a backward dependence-free strictness analysis method [Joh81], which was recognized by John Hugues to be considerably more efficient than Mycroft's forward analysis [Hug88]: "there are grounds for believing that backward analysis will be considerably more efficient than forward analysis ... Every context function has only one argument". The difference of forward or backward direction of analysis is a specious explanation of efficiency. Efficiency comes from the idea of dependence-free strictness analysis which can be applied independently of the direction of analysis.

À la Johnsson's Abstraction As shown in example 6.2.0.2 of [CC79b], a dependence-sensitive abstract interpretation can always be transformed into an dependence-free one to make the method less expensive by being less precise. For Mycroft's strictness analysis method, the approximation of $\mathcal{F}^\sharp = \vec{\mathcal{D}}^\sharp \stackrel{\mathrm{m}\leq}{\longmapsto} \mathcal{D}^\sharp$ by

[25] $|S|$ is the cardinality of set S

[26] This technique was latter popularized by Jones and Mycroft [JM86] as *minimal function graphs* understood as a denotational semantics collecting the needed subcalls for a main call (although the corresponding naïve Jacobi iteration is inefficient).

$\nu \quad : \mathcal{D}^\sharp \stackrel{\text{def}}{=} \{0,1\}$ $\vec{v} \quad : \vec{\mathcal{D}}^\sharp \stackrel{\text{def}}{=} \prod_{v\in\vec{v}} \mathcal{D}^\sharp$ $\varphi, f^\sharp : \mathcal{F}^\sharp \stackrel{\text{def}}{=} \vec{\mathcal{D}}^\sharp \stackrel{\text{m}}{\longmapsto} \mathcal{D}^\sharp$ $\vec{\varphi}, \vec{f}^\sharp : \vec{\mathcal{F}}^\sharp \stackrel{\text{def}}{=} \prod_{f\in\vec{f}} \mathcal{F}^\sharp$	$e^\sharp : \mathcal{E}^\sharp \stackrel{\text{def}}{=} \vec{\mathcal{F}}^\sharp \stackrel{\text{m}}{\longmapsto} \vec{\mathcal{D}}^\sharp \stackrel{\text{m}}{\longmapsto} \mathcal{D}^\sharp$ $\vec{e}^\sharp : \vec{\mathcal{E}}^\sharp \stackrel{\text{def}}{=} \vec{\mathcal{F}}^\sharp \stackrel{\text{m}}{\longmapsto} \vec{\mathcal{D}}^\sharp \stackrel{\text{m}}{\longmapsto} \vec{\mathcal{D}}^\sharp$ $\vec{F}^\sharp : \vec{\mathcal{F}}^\sharp \stackrel{\text{m}}{\longmapsto} \vec{\mathcal{F}}^\sharp$
$k^\sharp[\![\varphi]\!]\vec{\nu} \stackrel{\text{def}}{=} 1$ $v^\sharp[\![\varphi]\!]\vec{\nu} \stackrel{\text{def}}{=} \vec{\nu}[v]$ $b\vec{e}^\sharp[\![\varphi]\!]\vec{\nu} \stackrel{\text{def}}{=} b^\sharp(\vec{e}^\sharp[\![\varphi]\!]\vec{\nu})$ where $b^\sharp \geq^\sharp \alpha^\sharp(b^\Re)$ $(e_1 \to e_2, e_3)^\sharp[\![\varphi]\!]\vec{\nu} \stackrel{\text{def}}{=} e_1^\sharp[\![\varphi]\!]\vec{\nu} \wedge (e_2^\sharp[\![\varphi]\!]\vec{\nu} \vee e_3^\sharp[\![\varphi]\!]\vec{\nu})$ $f\vec{e}^\sharp[\![\varphi]\!]\vec{\nu} \stackrel{\text{def}}{=} \varphi[f](\vec{e}^\sharp[\![\varphi]\!]\vec{\nu})$ $\vec{e}^\sharp[\![\varphi]\!]\vec{\nu} \stackrel{\text{def}}{=} \prod_{v\in\vec{e}} \vec{e}[v]^\sharp[\![\varphi]\!]\vec{\nu}$	
$\vec{\bot}^\sharp \stackrel{\text{def}}{=} \prod_{f\in\vec{f}} \lambda\vec{\nu}\bullet 0$ $\vec{F}^\sharp \stackrel{\text{def}}{=} \lambda\vec{\varphi}\bullet \prod_{f\in\vec{f}} \vec{F}[f]^\sharp[\![\varphi]\!]$ $\vec{f}^\sharp \stackrel{\text{def}}{=} \text{lfp}^{\vec{\leq}}_{\vec{\bot}^\sharp} \vec{F}^\sharp = \vec{\bigvee}_{n\in\mathbb{N}} \vec{F}^{\sharp n}(\vec{\bot}^\sharp)$	

Fig. 2. Synopsis of the forward dependence-sensitive strictness semantics of $\prod_{f\in\vec{f}} f\vec{v} \equiv \vec{F}[f]$

$\nu \quad : \mathcal{D}^{\text{:}} \stackrel{\text{def}}{=} \{0,1\}$ $\varphi, f^{\text{:}} : \mathcal{F}^{\text{:}} \stackrel{\text{def}}{=} \prod_{v\in\vec{v}} (\mathcal{D}^{\text{:}} \stackrel{\text{m}}{\longmapsto} \mathcal{D}^{\text{:}})$ $\vec{\varphi}, \vec{f}^{\text{:}} : \vec{\mathcal{F}}^{\text{:}} \stackrel{\text{def}}{=} \prod_{f\in\vec{f}} \mathcal{F}^{\text{:}}$	$e^{\text{:}} : \vec{\mathcal{F}}^{\text{:}} \stackrel{\text{m}}{\longmapsto} \prod_{v\in\vec{v}} (\mathcal{D}^{\text{:}} \stackrel{\text{m}}{\longmapsto} \mathcal{D}^{\text{:}})$ $\vec{e}^{\text{:}} : \vec{\mathcal{F}}^{\text{:}} \stackrel{\text{m}}{\longmapsto} \prod_{v\in\vec{v}} (\mathcal{D}^{\text{:}} \stackrel{\text{m}}{\longmapsto} \prod_{v'\in\vec{e}} \mathcal{D}^{\text{:}})$ $\vec{F}^{\text{:}} : \vec{\mathcal{F}}^{\text{:}} \stackrel{\text{m}}{\longmapsto} \vec{\mathcal{F}}^{\text{:}}$
$k^{\text{:}}[\![\varphi]\!][v]\nu \stackrel{\text{def}}{=} 1$ $v^{\text{:}}[\![\varphi]\!][v]\nu \stackrel{\text{def}}{=} \nu$ $v^{\text{:}}[\![\varphi]\!][v']\nu \stackrel{\text{def}}{=} 1$ if $v' \neq v$ $b\vec{e}^{\text{:}}[\![\varphi]\!][v]\nu \stackrel{\text{def}}{=} b^\sharp(\vec{e}^{\text{:}}[\![\varphi]\!][v]\nu)$ $(e_1 \to e_2, e_3)^{\text{:}}[\![\varphi]\!][v]\nu \stackrel{\text{def}}{=} e_1^{\text{:}}[\![\varphi]\!][v]\nu \wedge (e_2^{\text{:}}[\![\varphi]\!][v]\nu \vee e_3^{\text{:}}[\![\varphi]\!][v]\nu)$ $f\vec{e}^{\text{:}}[\![\varphi]\!][v]\nu \stackrel{\text{def}}{=} \bigwedge_{v'\in\vec{e}} \varphi[f][v'](\vec{e}[v']^{\text{:}}[\![\varphi]\!][v]\nu)$ $\vec{e}^{\text{:}}[\![\varphi]\!][v]\nu \stackrel{\text{def}}{=} \prod_{v'\in\vec{e}} \vec{e}[v']^{\text{:}}[\![\varphi]\!][v]\nu$	
$\vec{\bot}^{\text{:}} \stackrel{\text{def}}{=} \prod_{f\in\vec{f}} \prod_{v\in\vec{v}} \lambda\nu\bullet 0$ $\vec{F}^{\text{:}} \stackrel{\text{def}}{=} \lambda\vec{\varphi}\bullet \prod_{f\in\vec{f}} \vec{F}[f]^{\text{:}}[\![\varphi]\!]$ $\vec{f}^{\text{:}} \stackrel{\text{def}}{=} \text{lfp}^{\vec{\leq}}_{\vec{\bot}^{\text{:}}} \vec{F}^{\text{:}} = \vec{\bigvee}_{n\in\mathbb{N}} \vec{F}^{\text{:}n}(\vec{\bot}^{\text{:}})$	

Fig. 3. Synopsis of the forward dependence-free strictness semantics of $\prod_{f\in\vec{f}} f\vec{v} \equiv \vec{F}[f]$

$\prod_{v\in\vec{v}} (\mathcal{D}^{\text{:}} \stackrel{\text{m}}{\longmapsto} \mathcal{D}^{\text{:}})$ where $\mathcal{D}^{\text{:}} \stackrel{\text{def}}{=} \mathcal{D}^\sharp = \{0,1\}$[27] with $0 \leq 0 < 1 \leq 1$ is the following[28]:

$$\alpha^{\text{:}}(\varphi) = \prod_{v\in\vec{v}} \lambda\nu\bullet\varphi(\vec{1}[v \leftarrow \nu]) \qquad \gamma^{\text{:}}(\phi) = \lambda\vec{\nu}\bullet \bigwedge_{v\in\vec{v}} \phi[v](\vec{\nu}[v]) \tag{70}$$

Example 4. The dependence-free abstraction of the dependence-sensitive strictness analysis $f^\sharp(x,y,z) = x \wedge (y \vee z)$ of function $f(x,y,z) = (x \to y, z)$ is $f^{\text{:}}$ such that

[27] We use the double sharp symbol $^{\text{:}}$ to denote the upper approximation of the sharp symbol $^\sharp$.

[28] For all $v' \in \vec{v}$ such that $v' \neq v$, we have $\vec{1}[v \leftarrow \nu][v'] = 1$ and $\vec{1}[v \leftarrow \nu][v] = \nu$.

$f^{\sharp\!\sharp}[x] = \lambda x{\bullet}x$, $f^{\sharp\!\sharp}[y] = \lambda y{\bullet}1$ and $f^{\sharp\!\sharp}[z] = \lambda z{\bullet}1$. Its concrete form is $\varphi(x, y, z) = x$ so that one can no longer handle $f(x, y, z)$ nicely. □

Observe that if $\varphi \in \mathcal{F}^\sharp = \vec{\mathcal{D}}^\sharp \xrightarrow{\mathrm{m}\leq} \mathcal{D}^\sharp$ and $\alpha^{\sharp\!\sharp}(\varphi)[v](1) = 0$ then, by (70), $\varphi(\vec{1}[v \leftarrow 1])$ $= \varphi(\vec{1}) = 0$ so that, by monotonicity, $\forall \vec{\nu} \in \vec{\mathcal{D}}^\sharp : \varphi(\vec{\nu}) = 0$ proving that $\forall v' \in \vec{v} :$ $\forall \nu \in \mathcal{D}^{\sharp\!\sharp} : \alpha^{\sharp\!\sharp}(\varphi)[v'](\nu) = \varphi(\vec{1}[v' \leftarrow \nu] = 0$. Let us now consider the expression LHS $=$ $\bigwedge_{v' \in \vec{v}} \alpha^{\sharp\!\sharp}(\varphi)[v'](\vec{1}[v \leftarrow \nu][v']) = \alpha^{\sharp\!\sharp}(\varphi)[v](\nu) \wedge \bigwedge\{\alpha^{\sharp\!\sharp}(\varphi)[v'](1) \mid v' \in \vec{v} \wedge v' \neq v\}$. If all $\alpha^{\sharp\!\sharp}(\varphi)[v'](1)$ are equal to 1 then LHS $= \alpha^{\sharp\!\sharp}(\varphi)[v](\nu)$. Else, there exists some $v' \in \vec{v}$ such that $\alpha^{\sharp\!\sharp}(\varphi)[v'](1) = 0$. Then $\alpha^{\sharp\!\sharp}(\varphi)[v](\nu) = 0$ and once again LHS $= \alpha^{\sharp\!\sharp}(\varphi)[v](\nu)$. Intuitively, if the dependence-free strictness analysis shows non-termination of a function for all values ν of its parameter v, this conclusion is drawn without knowing the other parameters v' so that in the analysis the proof of non-termination of this function cannot depend on any of its parameters. This leads to the following definition:

$$\mathcal{F}^{\sharp\!\sharp} \stackrel{\text{def}}{=} \left\{ \phi \in \prod_{v \in \vec{v}} (\mathcal{D}^{\sharp\!\sharp} \xrightarrow{\mathrm{m}\leq} \mathcal{D}^{\sharp\!\sharp}) \;\middle|\; \forall v \in \vec{v} : \lambda\nu{\bullet} \bigwedge_{v' \in \vec{v}} \phi[v'](\vec{1}[v \leftarrow \nu][v']) = \phi[v] \right\} \quad (71)$$

Proposition 26 Connection between the dependence-sensitive and dependence-free strictness semantics.

$$\mathcal{F}^\sharp(\leq) \underset{\alpha^{\sharp\!\sharp}}{\overset{\gamma^{\sharp\!\sharp}}{\leftrightarrows}} \mathcal{F}^{\sharp\!\sharp}(\vec{\leq})$$

For a program $\prod_{f \in \vec{f}} f\vec{v} \equiv \vec{F}[f]$, $\vec{\mathcal{F}}^\sharp = \prod_{f \in \vec{f}} \mathcal{F}^\sharp$ is approximated componentwise by $\vec{\mathcal{F}}^{\sharp\!\sharp} \stackrel{\text{def}}{=} \prod_{f \in \vec{f}} \mathcal{F}^{\sharp\!\sharp}$ as follows:

$$\begin{array}{rr} \vec{\mathcal{F}}^\sharp(\vec{\leq}) \underset{\vec{\alpha}^{\sharp\!\sharp}}{\overset{\vec{\gamma}^{\sharp\!\sharp}}{\leftrightarrows}} \vec{\mathcal{F}}^{\sharp\!\sharp}(\vec{\leq}) & \\ \vec{\alpha}^{\sharp\!\sharp}(\vec{\phi}) = \prod_{f \in \vec{f}} \alpha^{\sharp\!\sharp}(\vec{\phi}[f]) & \vec{\gamma}^{\sharp\!\sharp}(\vec{\varphi}) = \prod_{f \in \vec{f}} \gamma^{\sharp\!\sharp}(\vec{\varphi}[f]) \end{array} \quad (72)$$

Forward Dependence-Free Strictness Semantics The forward dependence-free strictness semantics $\vec{f}^{\sharp\!\sharp} \stackrel{\text{def}}{=} \mathrm{lfp}^{\vec{\leq}^{\sharp\!\sharp}}_{\vec{\bot}^{\sharp\!\sharp}} \vec{F}^{\sharp\!\sharp}$ of a program $\prod_{f \in \vec{f}} f\vec{v} \equiv \vec{F}[f]$ is given at Fig. 3.

The equations which have been proposed by guesswork in the literature are not always optimal. For example in [Joh81] `x` is needed in `f(x)` $\equiv$ `f(x)` but not in `f(x)` $\equiv$ `f(1)`. In both cases, the algorithm of Fig. 3 leads to the conclusion that `f` is strict in `x` (because it is non-terminating). Analogously, for the program: `f(x, y)` $\equiv$ `(x` $\rightarrow$ `y, f(y, x))` the equations: $f^{\sharp\!\sharp}$[x]$x = x$ and $f^{\sharp\!\sharp}$[y]$y = y \vee f^{\sharp\!\sharp}$[y]1 proposed by [Hug88] have the least solution: $f^{\sharp\!\sharp}$[x]$0 = 0$ and $f^{\sharp\!\sharp}$[y]$0 = 1$ whereas the equations of Fig. 3: $f^{\sharp\!\sharp}$[x]$x = x$ and $f^{\sharp\!\sharp}$[y]$y = y \vee (f^{\sharp\!\sharp}$[x]$y \wedge f^{\sharp\!\sharp}$[y]$1)$ are correct and lead to better results: $f^{\sharp\!\sharp}$[x]$0 = 0$ and $f^{\sharp\!\sharp}$[y]$0 = 0$.

The forward dependence-free strictness semantics of Fig. 3 has been obtained constructively by a formal hand-computation, using the abstract interpretation (72) of Mycroft's dependence-sensitive strictness semantics $\vec{f}^\sharp = \mathrm{lfp}^{\vec{\leq}^\sharp}_{\vec{\bot}^\sharp} \vec{F}^\sharp$ given at Fig. 2 and applying Cor. 9, whence it is correct:

Proposition 27 Connection of the forward dependence-free strictness semantics with the dependence-sensitive semantics.

$$\vec{\alpha}^{\sharp\!\sharp}(\vec{\alpha}^\sharp(\vec{f}^{\Re})) = \vec{\alpha}^{\sharp\!\sharp}(\vec{\alpha}^\sharp(\mathrm{lfp}^{\vec{\sqsubseteq}^{\Re}}_{\vec{\bot}^{\Re}} \vec{F}^{\Re})) \vec{\leq} \vec{\alpha}^{\sharp\!\sharp}(\vec{f}^\sharp) = \vec{\alpha}^{\sharp\!\sharp}(\mathrm{lfp}^{\vec{\leq}^\sharp}_{\vec{\bot}^\sharp} \vec{F}^\sharp) \vec{\leq} \vec{f}^{\sharp\!\sharp} = \mathrm{lfp}^{\vec{\leq}^{\sharp\!\sharp}}_{\vec{\bot}^{\sharp\!\sharp}} \vec{F}^{\sharp\!\sharp}$$

Proposition 28 Safeness of the dependence-free strictness analysis method. *If* $\vec{f}^{\sharp}[f][v]0 = 0$ *then function* f *is strict in its parameter* v.

But for suboptimality and the use of different notations, it is interesting to note that the forward equations of Fig. 3 resemble the backward equations proposed by [Joh81], [Hug88] and [DW90]. T. Johnsson [Joh81] uses the dual notations T, F, $\vee$, & instead of 0, 1, $\wedge$, $\vee$. His equations define $\vec{f}^{\sharp}[f][v]0$ directly. When needed, $\vec{f}^{\sharp}[f][v]1$ is approximated by 1 which is suboptimal. J. Hughes [Hug88] uses the notations S, L, &, $\cup$ instead of 0, 1, $\wedge$, $\vee$. The handling of parameters as $f\vec{e}^{\sharp}[\![\vec{\varphi}]\!][v]\nu \stackrel{\text{def}}{=} \vec{\varphi}[f][v](\vec{e}[v]^{\sharp}[\![\vec{\varphi}]\!][v]\nu)$ instead of $f\vec{e}^{\sharp}[\![\vec{\varphi}]\!][v]\nu \stackrel{\text{def}}{=} \bigwedge_{v' \in \vec{e}} \vec{\varphi}[f][v'](\vec{e}[v']^{\sharp}[\![\vec{\varphi}]\!][v]\nu)$ is suboptimal. The equations of the "new low-fidelity first-order forward strictness analysis technique" of [DW90] directly provide $f^{\sharp}(\nu_1, \ldots, \nu_n) = \bigwedge_{i=1}^{n} f^{\sharp}[v_i]\nu_i$ so that $f^{\sharp}[v_i]\nu_i = f^{\sharp}(\vec{1}[v_i \leftarrow \nu_i])$. Forward and backward strictness analysis lead to isomorphic equations hence to isomorphic analysis algorithms. The only difference is between the dependence-sensitive and dependence-free strictness semantics.

3.4 On the Use of Widenings to Mix Dependence-Free and Dependence-Sensitive Forward Strictness Analyses

Since Mycroft's dependence-sensitive strictness analysis is powerful but sometimes expensive and Johnsson's dependence-free strictness analysis is cheaper but less precise, we can attempt a compromise using widenings. One idea is to avoid the combinatorial explosion due to the computation of $f^{\sharp}(\nu_1, \ldots, \nu_n)$ with $\nu_i \in \{0, 1\}$ for all $i \in \{1, \ldots, n\}$ by limiting the number of allowed 0 to a bound κ which can be fixed arbitrarily. For $\kappa = \infty$ we would obtain Mycroft's dependence-sensitive strictness analysis whereas with $\kappa = 1$ we would obtain the dependence-free strictness analysis. The value of κ would have to be fixed experimentally or could be left to the user as a way to adjust the compromise between his available time and computing power resources or could vary during the analysis from ∞ down to 1 so as to avoid exponential analysis times. Calls $f^{\sharp}\vec{\nu}$ where $\vec{\nu} = (\nu_1, \ldots, \nu_n)$ has less than κ 0-valued ν_i (that is $(\sum_{v \in \nu} \neg\nu[v]) \leq \kappa$ where $\neg 0 = 1$ and $\neg 1 = 0$) would be evaluated by Mycroft's method as given at Fig. 2. Calls $f^{\sharp}\vec{\nu}$ with $(\sum_{v \in \nu} \neg\nu[v]) > \kappa$ would be subject to a widening which consists in applying the dependence-free method so that $f^{\sharp}\vec{\nu}$ would be over-estimated by $\bigwedge\{f^{\sharp}(\vec{1}[v \leftarrow 0]) \mid v \in \vec{\nu} \wedge \vec{\nu}[v] = 0\}$. A less drastic approximation of $f^{\sharp}\vec{\nu}$ would be $\bigwedge\{f^{\sharp}\vec{\nu}' \mid \vec{\nu}' \geq \vec{\nu} \wedge (\sum_{v \in \nu} \neg\vec{\nu}'[v]) = \kappa\}$ which consists in approximating $f^{\sharp}(\nu_1, \ldots, \nu_n)$ with more than κ 0-valued ν_i using the dependence-sensitive method with upper approximations having exactly κ zero-parameters.

4 Conclusion

We have shown that Mycroft's seminal dependence-sensitive [Myc80, Myc81] and Johnsson's dependence-free [Joh81] strictness analyses can be constructed by formal hand-derivation from a relational semantics [CC92] within the Galois connection based abstract interpretation framework [CC77a, CC79b]. This was hardly considered to be possible (e.g. [AH87], page 25) and shows that the difficulties that have been encountered with the formalization of strictness analysis [MN83, Nie88] are not intrinsic but mainly due to denotational semantics. Our methodology for designing

an abstract interpreter is constructive and this should be opposed to empirical design methods with a posteriori safeness verification using e.g. logical relations [MJ86].

Abstract interpretation is often opposed to dataflow analysis as being intrinsically costly. This is misunderstanding that widening operators can always be used as a practical compromise between efficiency and precision. For the strictness analysis example, we have suggested a good compromise between efficiency of dependence-free and the precision of the dependence-sensitive strictness analysis algorithms using a user-adjustable threshold for taking partial-dependencies into account.

Acknowledgement We would like to thank Alan Mycroft for numerous judicial comments on the first April 27, 1992 draft of this paper.

References

[AH87] S. Abramsky & C. Hankin, eds. *Abstract Interpretation of Declarative Languages.* Computers and their Applications. Ellis Horwood, 1987.

[Bry86] R. E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Trans. Comput.*, C-35(8), 1986.

[CC77a] P. Cousot & R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In 4^{th} *POPL*, pp. 238–252, Los Angeles, Calif., 1977. ACM Press.

[CC77b] P. Cousot & R. Cousot. Automatic synthesis of optimal invariant assertions: mathematical foundations. In *ACM Symposium on Artificial Intelligence & Programming Languages*, Rochester, New York, SIGPLAN Notices 12(8):1–12, 1977.

[CC77c] P. Cousot & R. Cousot. Static determination of dynamic properties of recursive procedures. In E.J. Neuhold, ed., *IFIP Conference on Formal Description of Programming Concepts*, St-Andrews, N.B., Canada, pp. 237–277. North-Holland, 1977.

[CC79a] P. Cousot & R. Cousot. Constructive versions of Tarski's fixed point theorems. *Pacific J. Math.*, 82(1):43–57, 1979.

[CC79b] P. Cousot & R. Cousot. Systematic design of program analysis frameworks. In 6^{th} *POPL*, pp. 269–282, San Antonio, Texas, 1979. ACM Press.

[CC92] P. Cousot & R. Cousot. Inductive definitions, semantics and abstract interpretation. In 19^{th} *POPL*, pp. 83–94, Albuquerque, N.M., 1992. ACM Press.

[DW90] K. Davis & P. Wadler. Strictness analysis in 4D. In S.L. Peyton Jones, G. Hutton, & C. Kehler Holst, eds., *Functional Programming, Glasgow 1990*, Proc. 1990 Glasgow Workshop on Functional Programming, Ullapool, Scot., pp. 23–43. Springer-Verlag and BCS, 13–15 Aug. 1990.

[Hug88] R. J. M. Hughes. Backwards analysis of functional programs. In D. Bjørner, A. P. Ershov, & N. D. Jones, eds., *Partial Evaluation and Mixed Computation*, Proceedings IFIP TC2 Workshop, Gl Avernæs, Ebberup, 18–24 Oct. 1987, (DK), pp. 187–208. Elsevier, 1988.

[HY86] P. Hudak & J. Young. Higher-order strictness analysis in untyped lambda calculus. In 12^{th} *POPL*, pp. 97–109. ACM Press, Jan. 1986.

[JM81] N. D. Jones & S. S. Muchnick. Complexity of flow analysis, inductive assertion synthesis and a language due to Dijkstra. In S. S. Muchnick & N. D. Jones, eds., *Program Flow Analysis: Theory and Applications*, ch. 12, pp. 380–393. Prentice-Hall, 1981.

[JM86] N. D. Jones & A. Mycroft. Data flow analysis of applicative programs using minimal function graphs: abridged version. In 13^{th} *POPL*, pp. 296–306, St. Petersburg Beach, Fla, 1986. ACM Press.

[Joh81] T. Johnsson. Detecting when call-by-value can be used instead of call-by-need. Res. rep. LPM MEMO 14, Laboratory for Programming Methodology, Department of Computer Science, Chalmers University of Technology, S-412 96 Göteborg, (S), Oct. 1981.

[Mil78] R. Milner. A theory of polymorphism in programming. *J. Comput. Sys. Sci.*, 17(3):348–375, Dec. 1978.

[MJ86] A. Mycroft & N. D. Jones. A relational framework for abstract interpretation. In N. D. Jones & H. Ganzinger, eds., *Programs as Data Objects, Proceedings of a Workshop*, Copenhagen, (DK), 17-19 Oct. 1985, LNCS 215, pp. 156–171. Springer-Verlag, 1986.

[MN83] A. Mycroft & F. Nielson. Strong abstract interpretation using power domains. In J. Diaz, ed., *Tenth ICALP*, LNCS 154, pp. 536–547. Springer-Verlag, 1983.

[Mos90] P. D. Mosses. Denotational semantics. In J. van Leeuwen, ed., *Formal Models and Semantics*, vol. B of *Handbook of Theoretical Computer Science*, ch. 11, pp. 575–631. Elsevier, 1990.

[Myc80] A. Mycroft. The theory and practice of transforming call-by-need into call-by-value. In B. Robinet, ed., *Proc. Fourth International Symposium on Programming*, Paris, (F), 22–24 Apr. 1980, LNCS 83, pp. 270–281. Springer-Verlag, 1980.

[Myc81] A. Mycroft. *Abstract Interpretation and Optimising Transformations for Applicative Programs.* Ph.D. Dissertation, CST-15-81, Department of Computer Science, University of Edinburgh, Edinburgh, Scot., Dec. 1981.

[Nie88] F. Nielson. Strictness analysis and denotational abstract interpretation. *Inf. & Comp.*, 76(1):29–92, 1988.

[SMR91] R. C. Sekar, P. Mishra, & I. V. Ramakrishnan. On the power and limitation of strictness analysis based on abstract interpretation. In *18^{th} POPL*, pp. 37–48, Orlando, Fla, 1991. ACM Press.

Proofs

▪ **Proof of Prop. 2** By (1), $F^0(\bot) = \bot \sqsubseteq F^1(\bot)$. Assume, by hypothesis, that $\forall \lambda \leq \lambda' : F^\lambda(\bot) \sqsubseteq F^{\lambda'}(\bot)$. It $\lambda' = \beta + 1$ is a successor ordinal then in particular $F^\beta(\bot) \sqsubseteq F^{\lambda'}(\bot)$ so that transitivity (1) and monotonicity (2) imply $F^\lambda(\bot) \sqsubseteq F^{\lambda'}(\bot) = F^{\beta+1}(\bot) = F(F^\beta(\bot)) \sqsubseteq F(F^{\lambda'}(\bot)) = F^{\lambda'+1}(\bot)$. It follows that $\forall \lambda \leq \lambda' + 1 : F^\lambda(\bot) \sqsubseteq F^{\lambda'+1}(\bot)$. If λ' is a limit ordinal then $F^{\lambda'}(\bot) = \bigsqcup_{\lambda<\lambda'} F^\lambda(\bot)$ so that $\forall \lambda \leq \lambda' : F^\lambda(\bot) \sqsubseteq F^{\lambda'}(\bot)$ by (1) and definition of least upper bounds. By transfinite induction on λ', we conclude $\forall \lambda \leq \lambda' : F^\lambda(\bot) \sqsubseteq F^{\lambda'}(\bot)$.

We have $F^0(\bot) = \bot \leq \gamma(\bot^\sharp) = \gamma(F^{\sharp 0}(\bot^\sharp))$ by (5). Assume, by hypothesis, that $F^\lambda(\bot) \leq \gamma(F^{\sharp\lambda}(\bot^\sharp))$. Then (6) implies $F^{\lambda+1}(\bot) = F(F^\lambda(\bot)) \leq \gamma(F^\sharp(F^{\sharp\lambda}(\bot^\sharp))) = \gamma(F^{\sharp\lambda+1}(\bot^\sharp))$. By (7) this remains true for limit ordinals. Hence, by transfinite induction induction, $\forall \lambda \in \mathbb{O} : F^\lambda(\bot) \leq \gamma(F^{\sharp\lambda}(\bot^\sharp))$.

Now (1) and (2) imply (3), which, together with (7) for $\lambda = \mathbb{O}$, imply $\mathrm{lfp}^{\sqsubseteq}_{\bot} F = \bigsqcup_{\lambda\in\mathbb{O}} F^\lambda(\bot) \leq \gamma\big(\bigsqcup^\sharp{}_{\lambda\in\mathbb{O}} F^{\sharp\lambda}(\bot^\sharp)\big)$.

▪ **Proof of Prop. 3** By (15), $\alpha_1 \circ \Phi \circ \gamma_0 \leq^\sharp \Psi$ implies $\forall \varphi \in \mathcal{F}_0^\sharp : \alpha_1(\Phi(\gamma_0(\varphi))) \leq^\sharp \Psi(\varphi)$ whence $\forall \varphi \in \mathcal{F}_0^\sharp : \Phi(\gamma_0(\varphi)) \leq \gamma_1(\Psi(\varphi))$ by (14). In particular for $\varphi = \alpha_0(\phi)$, we have $\forall \phi \in \mathcal{F}_0 : \Phi(\gamma_0(\alpha_0(\phi))) \leq \gamma_1(\Psi(\alpha_0(\phi)))$. But (14) implies $\forall \phi \in \mathcal{F}_0 : \phi \leq \gamma_0(\alpha_0(\phi))$ so that $\forall \phi \in \mathcal{F}_0 : \Phi(\phi) \leq \Phi(\gamma_0(\alpha_0(\phi)))$ since $\Phi \in \mathcal{F}_0 \xmapsto{\mathrm{m}\leq} \mathcal{F}_1$. By

transitivity (9), we conclude $\forall \phi \in \mathcal{F}_0 : \Phi(\phi) \leq \gamma_1(\Psi(\alpha_0(\phi)))$ that is $\Phi \leq \gamma_1 \circ \Psi \circ \alpha_0$ by (15).

Reciprocally, if $\Phi \leq \gamma_1 \circ \Psi \circ \alpha_0$ that is $\forall \phi \in \mathcal{F}_0 : \Phi(\phi) \leq \gamma_1(\Psi(\alpha_0(\phi)))$ then $\forall \phi \in \mathcal{F}_0 : \alpha_1(\Phi(\phi)) \leq^\sharp \Psi(\alpha_0(\phi))$ by (14). In particular for $\phi = \gamma_0(\varphi)$, $\forall \varphi \in \mathcal{F}_0^\sharp : \alpha_1(\Phi(\gamma_0(\varphi))) \leq^\sharp \Psi(\alpha_0(\gamma_0(\varphi)))$. But $\alpha_0(\gamma_0(\varphi)) \leq^\sharp \varphi$ by (14) so that $\Psi(\alpha_0(\gamma_0(\varphi))) \leq^\sharp \Psi(\varphi)$ since $\Psi \in \mathcal{F}_0^\sharp \stackrel{\mathrm{m}\leq^\sharp}{\longmapsto} \mathcal{F}_1^\sharp$. By transitivity (10), we conclude $\forall \varphi \in \mathcal{F}_0^\sharp : \alpha_1(\Phi(\gamma_0(\varphi))) \leq^\sharp \Psi(\varphi)$ that is $\alpha_1 \circ \Phi \circ \gamma_0 \leq^\sharp \Psi$ by (15).

▪ **Proof of Cor. 4** The proof of (18) is similar to that of Prop. 3, except that in the reciprocal, we have $\Psi(\alpha_0(\gamma_0(\varphi))) = \Psi(\varphi)$ by (17) hence $\Psi(\alpha_0(\gamma_0(\varphi))) \leq^\sharp \Psi(\varphi)$ by (10).

▪ **Proof of Cor. 5** If we consider a monotone abstract function $\Psi \in \mathcal{F}_0^\sharp \stackrel{\mathrm{m}\leq^\sharp}{\longmapsto} \mathcal{F}_1^\sharp$ then $\gamma_1 \circ \Psi \circ \alpha_0 \in \mathcal{F}_0 \stackrel{\mathrm{m}\leq}{\longmapsto} \mathcal{F}_1$ since it is the composition of monotonic functions and its abstraction $\alpha_1 \circ \gamma_1 \circ \Psi \circ \alpha_0 \circ \gamma_0$ is Ψ since $\alpha_0 \circ \gamma_0$ and $\alpha_1 \circ \gamma_1$ are the identity, proving the Galois surjection property (19).

▪ **Proof of Prop. 6** By (21), $\bot^\sharp \geq^\sharp \alpha(\bot)$ whence $\gamma(\bot^\sharp) \geq \bot$ by (14) proving (5).

If $\varphi \leq \gamma(\phi^\sharp)$ then $\alpha(F(\varphi)) \leq^\sharp \alpha(F(\gamma(\phi^\sharp)))$ since F and α are monotonic by (20) and (14). It follows that $\alpha(F(\varphi)) \leq^\sharp F^\sharp(\phi^\sharp)$ by (22), (15) and transitivity (10) whence $F(\varphi) \leq \gamma(F^\sharp(\phi^\sharp))$ by (14) proving that (6) holds.

If $\forall \beta \leq \beta' < \lambda : \varphi_\beta \sqsubseteq \varphi_{\beta'} \wedge \varphi_\beta \leq \gamma(\phi_\beta^\sharp)$ then $\alpha(\varphi_\beta) \leq^\sharp \phi_\beta^\sharp$ by (14) so that $\alpha\Big(\bigsqcup_{\beta<\lambda} \varphi_\beta\Big) \leq^\sharp \bigsqcup_{\beta<\lambda}^\sharp \phi_\beta^\sharp$ by (23), proving $\bigsqcup_{\beta<\lambda} \varphi_\beta \leq \gamma\Big(\bigsqcup_{\beta<\lambda}^\sharp \phi_\beta^\sharp\Big)$ by (14) whence (7).

By Prop. 2, we conclude that $\mathrm{lfp}_\bot^\sqsubseteq F \leq^\sharp \gamma\Big(\bigsqcup_{\lambda\in\mathbb{O}}^\sharp F^{\sharp\,\lambda}(\bot^\sharp)\Big)$.

▪ **Proof of Lem. 7** By (1), we have $\bot \sqsubseteq \gamma(\bot^\sharp)$ whence $\alpha(\bot) \sqsubseteq^\sharp \bot^\sharp$ by (25) and therefore $\alpha(\bot) = \bot^\sharp$ by (24). By (25), (1) and (24), α is a complete $\bigsqcup$-morphism so that $\alpha(\bigsqcup_{\lambda\in\mathbb{O}} \gamma(\varphi_\lambda^\sharp)) = \bigsqcup_{\lambda\in\mathbb{O}}^\sharp \alpha(\gamma(\varphi_\lambda^\sharp)) = \bigsqcup_{\lambda\in\mathbb{O}}^\sharp \varphi_\lambda^\sharp$ by (25).

▪ **Proof of Lem. 8** If $\forall \lambda \in \mathbb{O} : \alpha(\varphi_\lambda) \leq^\sharp \phi_\lambda^\sharp$ then $\forall \lambda \in \mathbb{O} : \alpha(\varphi_\lambda) \sqsubseteq^\sharp \phi_\lambda^\sharp$ since $\leq^\sharp = \sqsubseteq^\sharp$ and therefore $\bigsqcup_{\lambda\in\mathbb{O}}^\sharp \alpha(\varphi_\lambda) \sqsubseteq^\sharp \bigsqcup_{\lambda\in\mathbb{O}}^\sharp \phi_\lambda^\sharp$ by (24) and definition of lubs. But (1), (24) and (25) imply that α is a complete $\sqcup$-morphism, a property of Galois connections: $\bigsqcup_{\lambda\in\mathbb{O}}^\sharp \alpha(\varphi_\lambda) = \alpha\Big(\bigsqcup_{\lambda\in\mathbb{O}}^\sharp \varphi_\lambda\Big)$. We get $\alpha\Big(\bigsqcup_{\lambda\in\mathbb{O}}^\sharp \varphi_\lambda\Big) \sqsubseteq^\sharp \bigsqcup_{\lambda\in\mathbb{O}}^\sharp \phi_\lambda^\sharp$ whence $\alpha\Big(\bigsqcup_{\lambda\in\mathbb{O}} \varphi_\lambda\Big) \leq^\sharp \bigsqcup_{\lambda\in\mathbb{O}}^\sharp \phi_\lambda^\sharp$ since $\leq^\sharp = \sqsubseteq^\sharp$.

▪ **Proof of Cor. 9** By Lem. 7, we have (26) hence $\alpha(\bot) \geq^\sharp \bot^\sharp$ by (24) and $\leq^\sharp = \sqsubseteq^\sharp$. By Lem. 8, we have (23). We conclude by Prop. 6 that (8) holds.

▪ **Proof of Prop. 13** If $\forall v \in I : \vec{\nu}[v] \in \{\bot, \Omega\}$ and $\varphi(\vec{\nu}) \rightsquigarrow \nu$ then $\langle \vec{\nu}, \nu \rangle \in \varphi$ whence $\langle \vec{\nu}, \nu \rangle \in \varphi'$ since $\varphi \subseteq \varphi'$ that is $\varphi'(\vec{\nu}) \rightsquigarrow \nu$ so that $\nu \in \{\bot, \Omega\}$ since φ' is strict in its parameters $v \in I$ proving that φ is strict in $v \in I$.

▪ **Proof of Prop. 14** By (48), $\alpha_{\mathcal{D}}^\sharp(V) \leq \nu$ is equivalent to $\bigvee\{\alpha_{\mathcal{T}}^\sharp(\nu') \mid \nu' \in V\} \leq \nu$, that is, by definition of lubs to $\forall \nu' \in V : \alpha_{\mathcal{T}}^\sharp(\nu') \leq \nu$ and to $V \subseteq \{\nu' \mid \alpha_{\mathcal{T}}^\sharp(\nu') \leq \nu\}$, which by (48) is equivalent to $V \subseteq \gamma_{\mathcal{D}}^\sharp(\nu)$.

If $\alpha_{\mathcal{T}}^\sharp$ is surjective then $\forall \nu^\sharp \in \mathcal{D}^\sharp : \exists \nu \in \mathcal{D} : \alpha_{\mathcal{T}}^\sharp(\nu) = \nu^\sharp$ so that $\alpha_{\mathcal{D}}^\sharp(\{\nu\}) = \nu^\sharp$ by (48) proving that $\alpha_{\mathcal{D}}^\sharp$ is surjective.

▪ **Proof of Prop. 15** By (50) and definition of the componentwise ordering $\vec{\leq}$, $\alpha^\sharp_{\vec{\mathcal{D}}}(\vec{V}) \vec{\leq} \vec{\nu}$ is equivalent to $\forall v \in \vec{v} : \alpha^\sharp_{\mathcal{D}}(\{\vec{\nu}[v] \mid \vec{\nu} \in \vec{V}\}) \leq \vec{\nu}[v]$, hence by (49) to $\forall v \in \vec{v} : \{\vec{\nu}[v] \mid \vec{\nu} \in \vec{V}\} \subseteq \gamma^\sharp_{\mathcal{D}}(\vec{\nu}[v])$ and to $\forall \vec{\nu} \in \vec{V} : \forall v \in \vec{v} : \vec{\nu}[v] \in \gamma^\sharp_{\mathcal{D}}(\vec{\nu}[v])$ that is $\vec{V} \subseteq \{\vec{\nu}' \in \vec{\mathcal{D}}^\Re \mid \forall v \in \vec{v} : \vec{\nu}'[v] \in \gamma^\sharp_{\mathcal{D}}(\vec{\nu}[v])\}$ which, by (50), is equivalent to $\vec{V} \subseteq \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu})$. By (50), $\alpha^\sharp_{\vec{\mathcal{D}}}$ is surjective since, by (49), $\alpha^\sharp_{\mathcal{D}}$ is surjective.

▪ **Proof of Prop. 16** If $\alpha^\sharp_Y$ is surjective then $\forall \nu^\sharp \in \mathcal{D}^\sharp : \exists \nu \in Y^\Omega_\perp : \alpha^\sharp_Y(\nu) = \nu^\sharp$ whence $\alpha^\sharp_Y(\nu) \leq \nu^\sharp$ proving by definition (48) that $\forall \nu^\sharp \in \mathcal{D}^\sharp : \exists \nu \in \gamma^\sharp_{\mathcal{D}}(\nu^\sharp) : \alpha^\sharp_{\mathcal{D}}(\{\nu\}) = \nu^\sharp$. It follows that $\forall \vec{\nu}^\sharp \in \vec{\mathcal{D}}^\sharp : \exists \vec{\nu} : (\forall v \in \vec{v} : \vec{\nu}[v] \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp[v])) \wedge (\forall v \in \vec{v} : \alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}[v]\}) = \vec{\nu}^\sharp[v])$ so that we conclude by (50).

▪ **Proof of Prop. 17** By definition of the pointwise ordering $\leq$, $\alpha^\sharp(\phi) \leq \varphi$ is equivalent to $\forall \vec{\nu} \in \vec{\mathcal{D}}^\sharp : \alpha^\sharp(\phi)(\vec{\nu}) \leq \varphi(\vec{\nu})$ hence, by (53) to $\forall \vec{\nu} \in \vec{\mathcal{D}}^\sharp : \alpha^\sharp_{\mathcal{D}}\left(\phi^\star(\gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}))\right) \leq \varphi(\vec{\nu})$ thus, by definition of * and (48) to $\forall \vec{\nu} \in \vec{\mathcal{D}}^\sharp : \left(\bigvee\{\alpha^\sharp_{\mathcal{D}}(\{\nu\}) \mid \exists \vec{\nu}' \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}) : \langle \vec{\nu}', \nu\rangle \in \phi\}\right) \leq \varphi(\vec{\nu})$ and, by definition of lubs, to $\forall \vec{\nu} \in \vec{\mathcal{D}}^\sharp : \forall \nu \in \mathcal{D}^\Re : \left(\exists \vec{\nu}' \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}) : \langle \vec{\nu}', \nu\rangle \in \phi\right) \Rightarrow \left(\alpha^\sharp_{\mathcal{D}}(\{\nu\}) \leq \varphi(\vec{\nu})\right)$, which can also be written as $\forall \vec{\nu} \in \vec{\mathcal{D}}^\sharp : \forall \nu \in \mathcal{D}^\Re : \forall \vec{\nu}' \in \vec{\mathcal{D}}^\Re : \left(\langle \vec{\nu}', \nu\rangle \in \phi\right) \Rightarrow \left(\{\vec{\nu}'\} \subseteq \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu})\right) \Rightarrow \left(\alpha^\sharp_{\mathcal{D}}(\{\nu\}) \leq \varphi(\vec{\nu})\right)$. By the Galois connection properties (49) and (51), this is equivalent to LHS $\equiv [\forall \vec{\nu} \in \vec{\mathcal{D}}^\sharp : \forall \nu \in \mathcal{D}^\Re : \forall \vec{\nu}' \in \vec{\mathcal{D}}^\Re : \left(\langle \vec{\nu}', \nu\rangle \in \phi\right) \Rightarrow \left(\left(\alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}'\}) \vec{\leq} \vec{\nu}\right) \Rightarrow \left(\{\nu\} \subseteq \gamma^\sharp_{\mathcal{D}}(\varphi(\vec{\nu}))\right)\right)]$. Observe that in this formula, we have $\{\nu\} \subseteq \gamma^\sharp_{\mathcal{D}}(\varphi(\alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}'\})))$ for $\vec{\nu} = \alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}'\})$. reciprocally, if $\{\nu\} \subseteq \gamma^\sharp_{\mathcal{D}}(\varphi(\alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}'\})))$ then $\alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}'\}) \vec{\leq} \vec{\nu}$, $\varphi \in \mathcal{F}^\sharp$, (54) and (49) imply $\gamma^\sharp_{\mathcal{D}}(\varphi(\alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}'\}))) \subseteq \gamma^\sharp_{\mathcal{D}}(\varphi(\vec{\nu}))$ by monotonicity, whence $\{\nu\} \subseteq \gamma^\sharp_{\mathcal{D}}(\varphi(\vec{\nu}))$ by transitivity. It follows that LHS is equivalent to $\forall \nu \in \mathcal{D}^\Re : \forall \vec{\nu}' \in \vec{\mathcal{D}}^\Re : \left(\langle \vec{\nu}', \nu\rangle \in \phi\right) \Rightarrow \left(\{\nu\} \subseteq \gamma^\sharp_{\mathcal{D}} \circ \varphi \circ \alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}'\})\right)$, that is to $\phi \subseteq \{\langle \vec{\nu}', \nu\rangle \mid \nu \in \gamma^\sharp_{\mathcal{D}}(\varphi(\alpha^{\sharp\,\natural}_{\vec{\mathcal{D}}}(\vec{\nu}')))\}$, hence by definition of ϱ and (53), to $\phi \subseteq \gamma^\sharp(\varphi)$.

▪ **Proof of Prop. 18** Let $\varphi \in \mathcal{F}^\sharp = \vec{\mathcal{D}}^\sharp \xmapsto{\mathrm{m}\vec{\leq}} \mathcal{D}^\sharp$ and $\phi = \{\langle \vec{\nu}, \nu\rangle \mid \vec{\nu} \in \vec{\mathcal{D}}^\Re \wedge \nu \in \gamma^\sharp_{\mathcal{D}}(\varphi(\alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\})))\}$. We show that $\alpha^\sharp(\phi) = \varphi$. By definition, we have $\alpha^\sharp(\phi) = \lambda \vec{\nu}^\sharp \bullet \bigvee\{\alpha^\sharp_{\mathcal{D}}(\{\nu\}) \mid \exists \vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp) : \nu \in \gamma^\sharp_{\mathcal{D}}(\varphi(\alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\})))\}$ which is equal to $\lambda \vec{\nu}^\sharp \bullet \bigvee_{\vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)} \bigvee\{\alpha^\sharp_{\mathcal{D}}(\{\nu\}) \mid \nu \in \gamma^\sharp_{\mathcal{D}}(\varphi(\alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\})))\}$. By (49), $\alpha^\sharp_{\mathcal{D}}$ is a complete $\cup$-morphism so that $\alpha^\sharp(\phi) = \lambda \vec{\nu}^\sharp \bullet \bigvee_{\vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)} \alpha^\sharp_{\mathcal{D}}(\gamma^\sharp_{\mathcal{D}}(\varphi(\alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\}))))$ since $\alpha^\sharp_{\mathcal{D}}(V) = \bigvee\{\alpha^\sharp_{\mathcal{D}}(\{\nu\}) \mid \nu \in V\}$. By (49), $\alpha^\sharp_{\mathcal{D}} \circ \gamma^\sharp_{\mathcal{D}}$ is the identity so that $\alpha^\sharp(\phi)$ is equal to $\lambda \vec{\nu}^\sharp \bullet \bigvee_{\vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)} \varphi(\alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\}))$. $\vec{\nu} \in \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)$ implies $\{\vec{\nu}\} \subseteq \gamma^\sharp_{\vec{\mathcal{D}}}(\vec{\nu}^\sharp)$ hence by (51), $\alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\}) \vec{\leq} \vec{\nu}^\sharp$ proving, by monotonicity and definition of lubs, that $\alpha^\sharp(\phi)(\vec{\nu}^\sharp) \leq \varphi(\vec{\nu}^\sharp)$. Moreover, by (52) and definition of upper bounds, $\varphi(\vec{\nu}^\sharp) \leq \alpha^\sharp(\phi)$ proving, by antisymmetry that we have $\alpha^\sharp(\phi) = \varphi$ hence $\alpha^\sharp$ is surjective.

▪ **Proof of (67)** The approximation ordering is defined componentwise so that (53) implies that $\vec{\gamma}^\sharp(\vec{\varphi}) \vec{\subseteq} \vec{\gamma}^\sharp(\vec{\varphi}')$ is equivalent to $\{\langle \vec{\nu}, \nu\rangle \mid \nu \in \gamma^\sharp_{\mathcal{D}} \circ \vec{\varphi}[f] \circ \alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\})\} \subseteq \{\langle \vec{\nu}, \nu\rangle \mid \nu \in \gamma^\sharp_{\mathcal{D}} \circ \vec{\varphi}'[f] \circ \alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\})\}$, that is to say $\forall f \in \vec{f} : \forall \vec{\nu} \in \vec{\mathcal{D}}^\Re : \gamma^\sharp_{\mathcal{D}} \circ \vec{\varphi}[f] \circ \alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\}) \subseteq \gamma^\sharp_{\mathcal{D}} \circ \vec{\varphi}'[f] \circ \alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\})$ hence, by definition of the Galois surjection (49) to $\forall f \in \vec{f} : \forall \vec{\nu} \in \vec{\mathcal{D}}^\Re : \alpha^\sharp_{\mathcal{D}} \circ \gamma^\sharp_{\mathcal{D}} \circ \vec{\varphi}[f] \circ \alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\}) \leq \vec{\varphi}'[f] \circ \alpha^\sharp_{\vec{\mathcal{D}}}(\{\vec{\nu}\})$.

(49) is a Galois surjection so $\alpha^\sharp_{\mathcal{D}} \circ \gamma^\sharp_{\mathcal{D}}$ is the identity and therefore this is equivalent to $\forall f \in \vec{f} : \forall \vec{\nu} \in \vec{\mathcal{D}}^{\Re} : \vec{\varphi}[f] \circ \alpha^\sharp_{\mathcal{B}}(\{\vec{\nu}\}) \leq \vec{\varphi}'[f] \circ \alpha^\sharp_{\mathcal{B}}(\{\vec{\nu}\})$ which implies, by (52), that $\forall f \in \vec{f} : \forall \vec{\nu}^\sharp \in \vec{\mathcal{D}}^\sharp : \vec{\varphi}[f](\vec{\nu}^\sharp) \leq \vec{\varphi}'[f](\vec{\nu}^\sharp)$ that is $\vec{\varphi} \mathrel{\dot{\leq}} \vec{\varphi}'$ componentwise and pointwise. Reciprocally, $\vec{\gamma}^\sharp$ is monotonic by (58).

- **Proof of (68)** – Assume $\phi \sqsubseteq^{\Re} \gamma^\sharp(\varphi)$ or equivalently, by (37), $(\phi \cap \top^{\Re}) \subseteq (\gamma^\sharp(\varphi) \cap \top^{\Re}) \wedge (\phi \cap \bot^{\Re}) \supseteq (\gamma^\sharp(\varphi) \cap \bot^{\Re})$. By (53), we have $\gamma^\sharp(\varphi) = \{\langle \vec{\nu}, \nu \rangle \mid \nu \in \gamma^\sharp_{\mathcal{D}} \circ \varphi \circ \alpha^\sharp_{\mathcal{B}}(\{\vec{\nu}\})\}$. By (47) and (48), $\forall \nu^\sharp \in \mathcal{D}^\sharp : \bot \in \gamma^\sharp_{\mathcal{D}}(\nu^\sharp)$ so that $\bot^{\Re} \subseteq \gamma^\sharp(\varphi)$ whence $\bot^{\Re} \subseteq (\phi \cap \bot^{\Re})$ and therefore $\bot^{\Re} \subseteq \phi$. It follows that $\phi \sqsubseteq^{\Re} \gamma^\sharp(\varphi)$ is equivalent to $(\phi \cap \top^{\Re}) \subseteq (\gamma^\sharp(\varphi) \cap \top^{\Re})$ that is $\phi \subseteq \gamma^\sharp(\varphi)$ since $\phi = \bot^{\Re} \cup (\phi \cap \top^{\Re}) \subseteq \bot^{\Re} \cup (\gamma^\sharp(\varphi) \cap \top^{\Re}) = (\gamma^\sharp(\varphi) \cap \bot^{\Re}) \cup (\gamma^\sharp(\varphi) \cap \top^{\Re}) = \gamma^\sharp(\varphi)$. The componentwise extention is obvious.

– For the computational ordering, we have $\vec{\gamma}^\sharp(\vec{\varphi}) \mathrel{\dot{\sqsubseteq}}^{\Re} \vec{\gamma}^\sharp(\vec{\varphi}') \Leftrightarrow \vec{\gamma}^\sharp(\vec{\varphi}) \mathrel{\dot{\subseteq}} \vec{\gamma}^\sharp(\vec{\varphi}') \Leftrightarrow \vec{\varphi} \mathrel{\dot{\leq}} \vec{\varphi}'$ by (67).

- **Proof of (69)** By (68) and (56), we have $\phi \sqsubseteq^{\Re} \gamma^\sharp(\varphi) \Leftrightarrow \phi \subseteq \gamma^\sharp(\varphi) \Leftrightarrow \alpha^\sharp(\phi) \leq (\varphi)$. Likewise by (68) and (58), we have $\vec{\phi} \mathrel{\dot{\sqsubseteq}}^{\Re} \vec{\gamma}^\sharp(\vec{\varphi}) \Leftrightarrow \vec{\phi} \mathrel{\dot{\subseteq}} \vec{\gamma}^\sharp(\vec{\varphi}) \Leftrightarrow \vec{\alpha}^\sharp(\vec{\phi}) \mathrel{\dot{\leq}} (\vec{\varphi})$.

- **Proof of Prop. 24** (1) holds since $\vec{\mathcal{F}}^{\Re}(\mathrel{\dot{\sqsubseteq}}^{\Re})$ is a complete lattice. $\mathrel{\dot{\leq}}$ is defined pointwise and $\vec{\mathcal{F}}^\sharp(\mathrel{\dot{\leq}})$ is a complete lattice so that (2), (10), (24) and (15) hold. $\mathrel{\dot{\subseteq}}$ is a partial order on $\vec{\mathcal{F}}^{\Re}$ so that (9) holds. (14) follows from (58). (20) that is $\vec{F}^{\Re} \in \vec{\mathcal{F}}^{\Re} \xmapsto{\mathrm{m}\,\dot{\subseteq}} \vec{F}^{\Re}$ follows from (46). (22) that is $\vec{F}^\sharp \mathrel{\dot{\geq}} \vec{\alpha}^\sharp \circ \vec{F}^{\Re} \circ \vec{\gamma}^\sharp$ has been proved above. (25) follows from (69). Finally, $\mathrel{\dot{\leq}}^\sharp = \mathrel{\dot{\sqsubseteq}}^\sharp = \mathrel{\dot{\leq}}$. We conclude by Cor. 9.

- **Proof of Prop. 25** If $\vec{f}^\sharp[f]\vec{\nu}^\sharp = 0$ then by proposition 24, $\vec{\alpha}^\sharp(\vec{f}^{\Re})[f]\vec{\nu}^\sharp \leq 0$ hence $\alpha^\sharp(\vec{f}^{\Re}[f])\vec{\nu}^\sharp \leq 0$ by (57) which implies $\alpha^\sharp_{\mathcal{D}}(\{\nu \mid \exists \vec{\nu} \in \gamma^\sharp_{\mathcal{B}}(\vec{\nu}^\sharp) : \langle \vec{\nu}, \nu \rangle \in \vec{f}^{\Re}[f]\}) \leq 0$ by equation (53). By (49), (48) and (47), $\{\nu \mid \exists \vec{\nu} \in \gamma^\sharp_{\mathcal{B}}(\vec{\nu}^\sharp) : \langle \vec{\nu}, \nu \rangle \in \vec{f}^{\Re}[f]\} \subseteq \{\Omega, \bot\}$. By definition of $\vec{\nu}^\sharp$ and (50), $\gamma^\sharp_{\mathcal{B}}(\vec{\nu}^\sharp) = \{\vec{\nu} \in \vec{\mathcal{D}}^{\Re} \mid \forall v \in I : \vec{\nu}[v] \in \{\Omega, \bot\}\}$. We get $\forall \nu \in \mathcal{D}^{\Re} : \forall \vec{\nu} \in \vec{\mathcal{D}}^{\Re} : (\forall v \in I : \vec{\nu}[v] \in \{\Omega, \bot\} \wedge \langle \vec{\nu}, \nu \rangle \in \vec{f}^{\Re}[f]) \Rightarrow \nu \in \{\Omega, \bot\}$ proving, by Def. 12, that $\vec{f}^{\Re}[f]$ is strict in I.

- **Proof of Prop. 26** – Assume that $\alpha^{\boldsymbol{:}}(\varphi) \mathrel{\dot{\leq}} \phi$ then, by (70), for all $v \in \vec{v}$ and $\nu \in \mathcal{D}^\sharp$, we have $\varphi(\vec{1}[v \leftarrow \nu]) \leq \phi[v](\nu)$. For all $\vec{\nu}$ in $\vec{\mathcal{D}}^\sharp$ and $v \in \vec{v}$, we have $\vec{\nu} \mathrel{\dot{\leq}} \vec{1}[v \leftarrow \vec{\nu}[v]]$, hence, by monotonicity and transitivity, $\varphi(\vec{\nu}) \leq \varphi(\vec{1}[v \leftarrow \vec{\nu}[v]]) \leq \phi[v](\vec{\nu}[v])$ whence $\varphi(\vec{\nu}) \leq \bigwedge_{v \in \vec{v}} \phi[v](\vec{\nu}[v]) = \gamma^{\boldsymbol{:}}(\phi)(\vec{\nu})$.

– Reciprocally, if for all $\vec{\nu} \in \vec{\mathcal{D}}^\sharp$ we have $\varphi(\vec{\nu}) \leq \gamma^{\boldsymbol{:}}(\phi)(\vec{\nu}) = \bigwedge_{v \in \vec{v}} \phi[v](\vec{\nu}[v])$ then for all $v \in \vec{v}$ and $\nu \in \mathcal{D}^\sharp$, if we let $\vec{\nu} = \vec{1}[v \leftarrow \nu]$ then $\varphi(\vec{1}[v \leftarrow \nu]) \leq \bigwedge_{v' \in \vec{v}} \phi[v'](\vec{1}[v \leftarrow \nu][v']) \leq \phi[v](\nu)$ proving that $\alpha^{\boldsymbol{:}}(\varphi) \mathrel{\dot{\leq}} \phi$.

– We have $\alpha^{\boldsymbol{:}}(\gamma^{\boldsymbol{:}}(\phi)) = \prod_{v \in \vec{v}} \lambda\nu\bullet\gamma^{\boldsymbol{:}}(\phi)(\vec{1}[v \leftarrow \nu]) = \prod_{v \in \vec{v}} \lambda\nu\bullet \bigwedge_{v' \in \vec{v}} \phi[v'](\vec{1}[v \leftarrow \nu][v'])$ by (70). By (71), this is equal to $\prod_{v \in \vec{v}} \phi[v] = \phi$.

- **Proof of Prop. 27** – The infimum is $\vec{\bot}^{\boldsymbol{:}} \stackrel{\text{def}}{=} \vec{\alpha}^{\boldsymbol{:}}(\vec{\bot}^\sharp)$, which by definition of $\vec{\alpha}^{\boldsymbol{:}}$ and $\vec{\bot}^\sharp$ can be simplified into $\prod_{f \in \vec{f}} \alpha^{\boldsymbol{:}}(\vec{\bot}^\sharp[f]) = \prod_{f \in \vec{f}} \alpha^{\boldsymbol{:}}(\lambda\vec{\nu}\bullet 0)$. By definition of $\alpha^{\boldsymbol{:}}$, this is equal to $\prod_{f \in \vec{f}} \prod_{v \in \vec{v}} \lambda\nu\bullet\lambda\vec{\nu}\bullet 0(\vec{1}[v \leftarrow \nu]) = \prod_{f \in \vec{f}} \prod_{v \in \vec{v}} \lambda\nu\bullet 0$.

– We define the inductive join $\vec{\bigvee}_{i \in \Delta} \vec{\varphi}_i \stackrel{\text{def}}{=} \vec{\alpha}^{\boldsymbol{:}}(\bigvee_{i \in \Delta} \vec{\gamma}^{\boldsymbol{:}}(\vec{\varphi}_i))$ which by (72) is equal

to $\prod_{f\epsilon\vec{f}}\alpha^{\sharp}((\vec{\bigvee}_{i\in\Delta}\vec{\gamma}^{\sharp}(\vec{\varphi}_i))[f]) = \prod_{f\epsilon\vec{f}}\alpha^{\sharp}(\vec{\bigvee}_{i\in\Delta}\gamma^{\sharp}(\vec{\varphi}_i[f]))$. By (70), this is equal to $\prod_{f\epsilon\vec{f}}\prod_{v\epsilon\vec{v}}\lambda\nu\bullet\vec{\bigvee}_{i\in\Delta}\vec{\bigwedge}_{v'\epsilon\vec{v}}\vec{\varphi}_i[f][v'](\vec{1}[v\leftarrow\nu][v']) = \prod_{f\epsilon\vec{f}}\prod_{v\epsilon\vec{v}}\vec{\bigvee}_{i\in\Delta}\vec{\varphi}_i[f][v]$ by (71).

– We define $\vec{F}^{\sharp}$ as an upper approximation of $\vec{\alpha}^{\sharp}\circ\vec{F}^{\sharp}\circ\vec{\gamma}^{\sharp}$ by letting $\vec{F}^{\sharp}\stackrel{\text{def}}{=}\lambda\vec{\varphi}\bullet\prod_{f\epsilon\vec{f}}\vec{F}[f]^{\sharp}[\![\vec{\varphi}]\!]$ and by defining $e^{\sharp}[\![\vec{\varphi}]\!]\geq\alpha^{\sharp}(e^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!])$ for all $\vec{\varphi}$ in $\vec{\mathcal{F}}^{\sharp}$. We proceed by case analysis on the expression e:

- $k^{\sharp}[\![\vec{\varphi}]\!] = \alpha^{\sharp}(k^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]) = \alpha^{\sharp}(\lambda\vec{\nu}\bullet 1) = \prod_{v\epsilon\vec{v}}\lambda\nu\bullet 1$.
- $v^{\sharp}[\![\vec{\varphi}]\!] = \alpha^{\sharp}(v^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]) = \alpha^{\sharp}(\lambda\vec{\nu}\bullet\vec{\nu}[v]) = \prod_{v'\epsilon\vec{v}}\lambda\nu\bullet\vec{1}[v'\leftarrow\nu][v]$.
- Assume, by induction hypothesis, that we have $\vec{e}[v']^{\sharp}[\![\vec{\varphi}]\!]\vec{\geq}\alpha^{\sharp}(\vec{e}[v']^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!])$ for all $\vec{\varphi}\in\vec{\mathcal{F}}^{\sharp}$ and $v'\ \epsilon\ \vec{e}$. Then $\alpha^{\sharp}(b\vec{e}^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]) = \alpha^{\sharp}(\lambda\vec{\nu}\bullet b^{\sharp}(\vec{e}^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]\vec{\nu})) = \alpha^{\sharp}(\lambda\vec{\nu}\bullet b^{\sharp}(\prod_{v'\epsilon\vec{e}}\vec{e}[v']^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]\vec{\nu}))\ \vec{\leq}\ \alpha^{\sharp}(\lambda\vec{\nu}\bullet b^{\sharp}(\prod_{v'\epsilon\vec{e}}\gamma^{\sharp}(\alpha^{\sharp}(\vec{e}[v']^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]))\vec{\nu})) = T$ since $\gamma^{\sharp}\circ\alpha^{\sharp}$ is extensive, $b^{\sharp}$ and $\alpha^{\sharp}$ are monotonic. By induction hypothesis and monotonicity, $T\ \vec{\leq}\ \alpha^{\sharp}(\lambda\vec{\nu}\bullet b^{\sharp}(\prod_{v'\epsilon\vec{e}}\gamma^{\sharp}(\vec{e}[v']^{\sharp}[\![\vec{\varphi}]\!])\vec{\nu}))$ which, by definition (70) of $\alpha^{\sharp}$ is $\prod_{v\epsilon\vec{v}}\lambda\nu\bullet b^{\sharp}(\prod_{v'\epsilon\vec{e}}\gamma^{\sharp}(\vec{e}[v']^{\sharp}[\![\vec{\varphi}]\!])\vec{1}[v\leftarrow\nu])$. By definition (70) of $\gamma^{\sharp}$, this is $\prod_{v\epsilon\vec{v}}\lambda\nu\bullet b^{\sharp}(\prod_{v'\epsilon\vec{e}}\bigwedge_{v''\epsilon\vec{v}}\vec{e}[v']^{\sharp}[\![\vec{\varphi}]\!][v''](\vec{1}[v\leftarrow\nu][v'']))$. According to (71), this can be simplified as $\prod_{v\epsilon\vec{v}}\lambda\nu\bullet b^{\sharp}(\prod_{v'\epsilon\vec{e}}\vec{e}[v']^{\sharp}[\![\vec{\varphi}]\!][v]\nu)$. This is $\prod_{v\epsilon\vec{v}}\lambda\nu\bullet b^{\sharp}(\vec{e}^{\sharp}[\![\vec{\varphi}]\!][v]\nu)$ by defining $\vec{e}^{\sharp}[\![\vec{\varphi}]\!][v]\nu\stackrel{\text{def}}{=}\prod_{v'\epsilon\vec{e}}\vec{e}[v']^{\sharp}[\![\vec{\varphi}]\!][v]\nu$ so that $b\vec{e}^{\sharp}[\![\vec{\varphi}]\!][v]\nu\stackrel{\text{def}}{=}b^{\sharp}(\vec{e}^{\sharp}[\![\vec{\varphi}]\!][v]\nu)$.
- Assume that $e_i^{\sharp}[\![\vec{\varphi}]\!]\geq\alpha^{\sharp}(e_i^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!])$ for all $\vec{\varphi}\in\vec{\mathcal{F}}^{\sharp}$ and $i=1,2,3$. We have $\alpha^{\sharp}((e_1\rightarrow e_2,e_3)^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]) = \alpha^{\sharp}(e_1^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]\wedge(e_2^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]\vee e_3^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!])) = T$. By Prop. 26, $\alpha^{\sharp}$ is a complete $\vee$-morphism, hence monotonic whence $T\leq\alpha^{\sharp}(e_1^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!])\wedge(\alpha^{\sharp}(e_2^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!])\vee\alpha^{\sharp}(e_3^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]))$. By induction hypothesis, this is less than or equal to $(e_1\rightarrow e_2,e_3)^{\sharp}[\![\vec{\varphi}]\!]\stackrel{\text{def}}{=}e_1^{\sharp}[\![\vec{\varphi}]\!]\wedge(e_2^{\sharp}[\![\vec{\varphi}]\!]\vee e_3^{\sharp}[\![\vec{\varphi}]\!])$.
- Assume, by induction hypothesis, that $\vec{e}[v']^{\sharp}[\![\vec{\varphi}]\!]\geq\alpha^{\sharp}(\vec{e}[v']^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!])$ for all $\vec{\varphi}\in\vec{\mathcal{F}}^{\sharp}$ and $v'\ \epsilon\ \vec{e}$. Then $\alpha^{\sharp}(f\vec{e}^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]) = \alpha^{\sharp}(\lambda\vec{\nu}\bullet\vec{\gamma}^{\sharp}(\vec{\varphi})[f](\vec{e}^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]\vec{\nu})) = \alpha^{\sharp}(\lambda\vec{\nu}\bullet\gamma^{\sharp}(\vec{\varphi}[f])(\vec{e}^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]\vec{\nu})) = \alpha^{\sharp}(\lambda\vec{\nu}\bullet\bigwedge_{v'\epsilon\vec{v}}\vec{\varphi}[f][v'](\vec{e}[v']^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]\vec{\nu})) = T$. By (70), $\alpha^{\sharp}\circ\gamma^{\sharp}$ is extensive, so that by monotonicity $T\leq\alpha^{\sharp}(\lambda\vec{\nu}\bullet\bigwedge_{v'\epsilon\vec{v}}\vec{\varphi}[f][v'](\gamma^{\sharp}(\alpha^{\sharp}(\vec{e}[v']^{\sharp}[\![\vec{\gamma}^{\sharp}(\vec{\varphi})]\!]))\vec{\nu})) = T'$. By induction hypothesis $T'\leq\alpha^{\sharp}(\lambda\vec{\nu}\bullet\bigwedge_{v'\epsilon\vec{v}}\vec{\varphi}[f][v'](\gamma^{\sharp}(\vec{e}[v']^{\sharp}[\![\vec{\varphi}]\!])\vec{\nu}))$. By definition of $\gamma^{\sharp}$, we get $\alpha^{\sharp}(\lambda\vec{\nu}\bullet\bigwedge_{v'\epsilon\vec{v}}\vec{\varphi}[f][v'](\bigwedge_{v''\epsilon\vec{v}}\vec{e}[v']^{\sharp}[\![\vec{\varphi}]\!][v''](\vec{\nu}[v'']))))$, which, by definition of $\vec{e}^{\sharp}$, is equal to $\alpha^{\sharp}(\lambda\vec{\nu}\bullet\bigwedge_{v'\epsilon\vec{v}}\vec{\varphi}[f][v'](\bigwedge_{v''\epsilon\vec{v}}\vec{e}^{\sharp}[\![\vec{\varphi}]\!][v''](\vec{\nu}[v''])[v']))$, hence, by (70), to $\prod_{v\epsilon\vec{v}}\lambda\nu\bullet\lambda\vec{\nu}\bullet\bigwedge_{v'\epsilon\vec{v}}\vec{\varphi}[f][v'](\bigwedge_{v''\epsilon\vec{v}}\vec{e}^{\sharp}[\![\vec{\varphi}]\!][v''](\vec{\nu}[v''])[v'])(\vec{1}[v\leftarrow\nu])$. This simplifies into $\prod_{v\epsilon\vec{v}}\lambda\nu\bullet\bigwedge_{v'\epsilon\vec{v}}\vec{\varphi}[f][v'](\bigwedge_{v''\epsilon\vec{v}}\vec{e}^{\sharp}[\![\vec{\varphi}]\!][v''](\vec{1}[v\leftarrow\nu][v''])[v'])$ by passing the parameters $\vec{\nu}$. By (70), this is $\prod_{v\epsilon\vec{v}}\lambda\nu\bullet\bigwedge_{v'\epsilon\vec{v}}\vec{\varphi}[f][v'](\vec{e}^{\sharp}[\![\vec{\varphi}]\!][v]\nu[v'])$, which, by definition of $\vec{e}^{\sharp}$, is $\prod_{v\epsilon\vec{v}}\lambda\nu\bullet\bigwedge_{v'\epsilon\vec{v}}\vec{\varphi}[f][v'](\vec{e}[v']^{\sharp}[\![\vec{\varphi}]\!][v]\nu)\stackrel{\text{def}}{=}f\vec{e}^{\sharp}[\![\vec{\varphi}]\!]$. By eliminating the conjunctions $\bigwedge\{\vec{\varphi}[f][v'](\vec{e}[v']^{\sharp}[\![\vec{\varphi}]\!][v]\nu)\mid v'\ \epsilon\ \vec{v}\wedge v'\neq v\}$ we get the suboptimal expression: $\prod_{v\epsilon\vec{v}}\lambda\nu\bullet\vec{\varphi}[f][v](\vec{e}[v]^{\sharp}[\![\vec{\varphi}]\!][v]\nu)$ which can be used to obtain the equations for the examples given in [Hug88].

■ **Proof of Prop. 28** If $\vec{f}^{\sharp}[f][v]0=0$ then $\vec{\alpha}^{\sharp}(\vec{f}^{\sharp})[f][v]0=0$ by Prop. 27 whence $\alpha^{\sharp}(\vec{f}^{\sharp}[f])[v]0=0$ by (72) which implies $\vec{f}^{\sharp}[f](\vec{1}[v\leftarrow 0])=0$ by (70) so that, by Prop. 25, function f is strict in its parameter v.

This article is reprinted from the proceedings of the international conference : "Formal Methods in Programming and their Applications", Academgorodok, Novosibirsk, Russia, June 28 - July 2, 1993, Lecture Notes in Computer Science 735, Dines Bjørner, Manfred Broy, Igor V. Pottosin (Eds.), ©Springer-Verlag Berlin Heidelberg 1993.