Abstract Interpretation of Mobile Ambients

René Rydhof Hansen, Jacob Grydholt Jensen,
Flemming Nielson, Hanne Riis Nielson.

Department of Computer Science, Aarhus University,
Ny Munkegade, DK-8000 Aarhus C, Denmark.

E-mail: {rrh,grydholt,fn,hrn}@daimi.au.dk
Web: http://www.daimi.au.dk/~fn/FlowLogic.html

Abstract. We demonstrate that abstract interpretation is useful for analysing calculi of computation such as the ambient calculus (which is based on the π-calculus); more importantly, we show that the entire development can be expressed in a constraint-based formalism that is becoming exceedingly popular for the analysis of functional and object-oriented languages.

The first step of the development is an analysis for counting occurrences of processes inside other processes (for which we show semantic correctness and that solutions constitute a Moore family); the second step is a previously developed control flow analysis that we show how to induce from the counting analysis (and its properties are derived from those of the counting analysis using general results).

1 Introduction

The ambient calculus is a calculus of computation that allows active processes to move between sites; it thereby extends the notion of mobility found in Java (e.g. [6]) where only passive code may move between sites. The untyped calculus was introduced in [3] and a type system for a polyadic variant was presented in [4]. The calculus is based on traditional process algebras (such as the π-calculus) but rather than focusing on communication (of values, channels, or processes) it focuses on the movement of processes between different sites; the sites correspond to administrative domains and are modelled using a notion of ambients. We refer to Sect. 2 for a review of the ambient calculus.

Abstract interpretation is a powerful technique for analysing programs by stepwise development. One starts with an overly precise and costly analysis and then develops more approximate and less costly analyses by carefully choosing appropriate Galois connections; in this way the semantic correctness of the initial analysis carries over to the approximate analyses. This technique has demonstrated its ability to deal successfully with logic languages, imperative languages and functional languages. Recent papers have studied how to apply the technique to calculi of computation such as the π-calculus but have found a need for assuming that processes were on a somewhat simplified form [8, 9].

We show that abstract interpretation can be developed for the ambient calculus (that contains most of the π-calculus as well as a number of other constructs) without the need to assume that processes are on a simplified form. More importantly, we are able to perform the entire development by expressing the analyses in a constraint-based formulation that closely corresponds to formulations that have become popular for the analysis of functional and object-oriented languages. This is likely to make abstract interpretation more accessible to a wider community because often abstract interpretation is being criticised for starting with a "low level" trace-based semantics.

The first step is the development of an analysis for explicitly modelling which processes can be inside what other processes; in order to model accurately what happens when the only process inside some other process actually moves out of the process, the analysis incorporates a counting component. As is customary for constraint-based formulations (in particular our own based on flow logics) this takes the form of specifying a satisfaction relation $\mathcal{C} \models P$ for when the set $\mathcal{C}$ of descriptions of processes satisfies the demands of the program P; here $\mathcal{C}$ is a set of tuples that each describe a set of processes. This approach is very natural for applications such as security and validation where information obtained by other means needs to be checked before it can be used – much the same ideas are found in type systems. We then show that the specification is semantically sound (meaning that, for a suitable extraction function η, if $\mathcal{C} \models P$ and $\eta(P) \in \mathcal{C}$ then $\mathcal{C}$ contains descriptions $\eta(Q)$ of all processes Q that P can evaluate to) and that the set of acceptable solutions has a least element (or more precisely that $\{\mathcal{C} \mid \mathcal{C} \models P\}$ constitutes a Moore family). The details are covered in Sect. 3.

The second step is to show that a previously developed control flow analysis [7] (in the manner of [1, 2]) can in fact be induced from the counting analysis; to show this we first clarify what it means to induce one constraint-based analysis from another. We then show that semantic correctness and the existence of least solutions carry over from the counting analysis. This shows that abstract interpretation is a useful guide also when developing analyses of calculi of computation. It follows that the theoretical properties established in [7] from first principles actually fall out of the general development. We refer to Sect. 4 for the details.

2 The Ambient Calculus

Ambients are introduced in [3] to provide named places with boundaries within which computation can happen. Ambients can be arbitrarily nested and may interact through the use of capabilities.

Syntax. As in [3] the syntax of processes $P, Q \in \textsf{Proc}$ is given by:

$$\begin{array}{rll} P, Q ::= & (\nu\, n)P & \text{restriction} \\ | & \mathbf{0} & \text{inactivity} \\ | & P \mid Q & \text{composition} \\ | & !P & \text{replication} \\ | & n_\mu^{\ell^a}[P] & \text{ambient} \\ | & in^{\ell^t} n_\mu.P & \text{capability to enter } n_\mu \\ | & out^{\ell^t} n_\mu.P & \text{capability to exit } n_\mu \\ | & open^{\ell^t} n_\mu.P & \text{capability to open } n_\mu \\ & & \\ n & & \text{names} \end{array}$$

The restriction $(\nu\, n)P$ introduces the new name n and limits its scope to P; $\mathbf{0}$ does nothing; $P \mid Q$ is P and Q running in parallel; replication provides recursion and iteration as $!P$ represents any number of copies of P in parallel. By $n_\mu^{\ell^a}[P]$ we denote the ambient named n with the process P running inside it. The capabilities $in^{\ell^t} n_\mu$ and $out^{\ell^t} n_\mu$ are used to move ambients whereas $open^{\ell^t} n_\mu$ is used to dissolve the boundary of an ambient; this will be made precise when we define the semantics below. To allow the analyses to deal with the α-equivalence inherent to the calculus, we have added markers $\mu \in \textsf{Mar}$ to the uses of names. For a process to be well-formed, all occurrences of a name in a given scope must have the same marker. As is customary for the flow logics approach to control flow analysis [1, 2] we have also placed labels $\ell^a \in \textsf{Lab}^a$ on ambients and labels $\ell^t \in \textsf{Lab}^t$ on transitions – this is merely a convenient way of indicating "program points" and is useful when developing the analyses. The sets of names, markers and labels are left unspecified but are assumed to be non-empty and disjoint. We write $fn(P)$ for the free names of P.

Semantics. The semantics is given by a structural congruence $P \equiv Q$ and a reduction relation $P \rightarrow Q$ in the manner of the π-calculus.

The congruence relation of Fig. 1 is a straightforward modification of the corresponding table of [3]. Furthermore, processes are identified up to renaming of bound names: $(\nu\, n)P = (\nu\, m)(P\,\{n \leftarrow m\})$ if $m \notin fn(P)$. Well-formedness is preserved under the congruence and the renaming of bound names.

The reduction relation is given in Fig. 2 and is as in [3]. A pictorial representation of the three basic rules is given in Fig. 3. Well-formedness is clearly preserved under reduction.

Example 1. Consider the ambient w that contains a probe p:

$$w_\mu^{\ell^a}[p_{\mu'}^{\ell^{a'}}[out^{\ell_1^t} w_\mu.in^{\ell_2^t} k_{\mu''}.in^{\ell_3^t} w_\mu] \mid P]$$

$$
\begin{array}{ll}
P \equiv P & P \mid Q \equiv Q \mid P \\
P \equiv Q \Rightarrow Q \equiv P & (P \mid Q) \mid R \equiv P \mid (Q \mid R) \\
P \equiv Q \wedge Q \equiv R \Rightarrow P \equiv R & !P \equiv P \mid !P \\
P \equiv Q \Rightarrow (\nu\, n)P \equiv (\nu\, n)Q & P \equiv Q \Rightarrow in^{\ell^t} n_\mu.P \equiv in^{\ell^t} n_\mu.Q \\
P \equiv Q \Rightarrow P \mid R \equiv Q \mid R & P \equiv Q \Rightarrow out^{\ell^t} n_\mu.P \equiv out^{\ell^t} n_\mu.Q \\
P \equiv Q \Rightarrow\, !P \equiv\, !Q & P \equiv Q \Rightarrow open^{\ell^t} n_\mu.P \equiv open^{\ell^t} n_\mu.Q \\
P \equiv Q \Rightarrow n_\mu^{\ell^a}[P] \equiv n_\mu^{\ell^a}[Q] & (\nu\, n)(\nu\, m)P \equiv (\nu\, m)(\nu\, n)P \\
P \mid \mathbf{0} \equiv P & (\nu\, n)(P \mid Q) \equiv P \mid (\nu\, n)Q \\
(\nu\, n)\mathbf{0} \equiv \mathbf{0} & \quad \text{if } n \notin fn(P) \\
!\mathbf{0} \equiv \mathbf{0} & (\nu\, n)(m_{\mu'}^{\ell^a}[P]) \equiv m_{\mu'}^{\ell^a}[(\nu\, n)P] \\
 & \quad \text{if } n \neq m
\end{array}
$$

Fig. 1. Structural congruence

$$\frac{}{n_\mu^{\ell^a}[in^{\ell^t} m_{\mu'}.P \mid Q] \mid m_{\mu'}^{\ell^{a'}}[R] \to m_{\mu'}^{\ell^{a'}}[n_\mu^{\ell^a}[P \mid Q] \mid R]}$$

$$\frac{}{m_{\mu'}^{\ell^{a'}}[n_\mu^{\ell^a}[out^{\ell^t} m_{\mu'}.P \mid Q] \mid R] \to n_\mu^{\ell^a}[P \mid Q] \mid m_{\mu'}^{\ell^{a'}}[R]}$$

$$\frac{}{open^{\ell^t} n_\mu.P \mid n_\mu^{\ell^a}[Q] \to P \mid Q}$$

$$\frac{P \to Q}{n_\mu^{\ell^a}[P] \to n_\mu^{\ell^a}[Q]} \qquad \frac{P \to Q}{(\nu\, n)P \to (\nu\, n)Q}$$

$$\frac{P \to Q}{P \mid R \to Q \mid R} \qquad \frac{P \equiv P' \quad P' \to Q' \quad Q' \equiv Q}{P \to Q}$$

Fig. 2. Reduction relation

The ambient can use the probe to fetch ambients with name k that are willing to be fetched; as an example we have:

$$k_{\mu''}^{\ell^{a''}}[open^{\ell_4^t} p_{\mu'} \mid Q]$$

For convenience we will denote the composition of ambients w and k by P:

$$\mathsf{P} = w_\mu^{\ell^a}[p_{\mu'}^{\ell^{a'}}[out^{\ell_1^t} w_\mu.in^{\ell_2^t} k_{\mu''}.in^{\ell_3^t} w_\mu] \mid P] \mid k_{\mu''}^{\ell^{a''}}[open^{\ell_4^t} p_{\mu'} \mid Q]$$

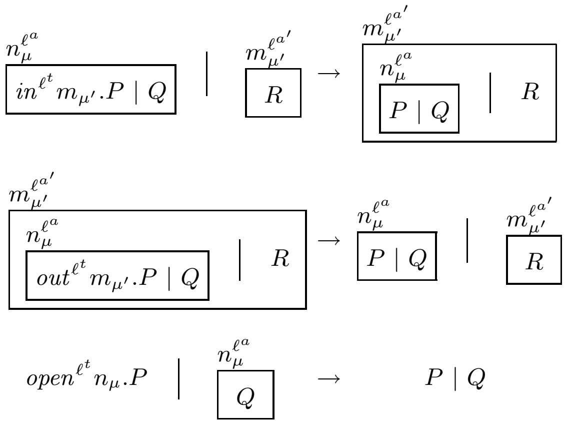

Fig. 3. Pictorial representation of the basic reduction rules

We illustrate the fact that w can use p to fetch k by the reduction sequence:

$$\begin{array}{l}
\quad w_\mu^{\ell^a}[p_{\mu'}^{\ell^{a'}}[out^{\ell_1^t} w_\mu.in^{\ell_2^t} k_{\mu''}.in^{\ell_3^t} w_\mu] \mid P] \mid k_{\mu''}^{\ell^{a''}}[open^{\ell_4^t} p_{\mu'} \mid Q] \\
\to w_\mu^{\ell^a}[P] \mid p_{\mu'}^{\ell^{a'}}[in^{\ell_2^t} k_{\mu''}.in^{\ell_3^t} w_\mu] \mid k_{\mu''}^{\ell^{a''}}[open^{\ell_4^t} p_{\mu'} \mid Q] \\
\to w_\mu^{\ell^a}[P] \mid k_{\mu''}^{\ell^{a''}}[p_{\mu'}^{\ell^{a'}}[in^{\ell_3^t} w_\mu] \mid open^{\ell_4^t} p_{\mu'} \mid Q] \\
\to w_\mu^{\ell^a}[P] \mid k_{\mu''}^{\ell^{a''}}[in^{\ell_3^t} w_\mu \mid Q] \\
\to w_\mu^{\ell^a}[k_{\mu''}^{\ell^{a''}}[Q] \mid P]
\end{array}$$

The reduction sequence shows that w sends p into k; k opens p and enters w. $\square$

We usually consider processes of the form $P_\star$ executing in an environment represented by the ambient $n_{\mu_\star}$ (with label $\ell_\star^a$). This amounts to systems of the form $n_{\mu_\star}^{\ell_\star^a}[P_\star]$, such that neither $\mu_\star$ nor $\ell_\star^a$ occur inside $P_\star$.

3 Occurrence Counting Analysis

In this section we present an analysis that counts occurrences of ambients. In the following, an ambient will be identified by its label $\ell^a \in \mathsf{Lab}^a$ and a capability by its label $\ell^t \in \mathsf{Lab}^t$.

The analysis works with powersets of representations of processes. A process can be described by a triple $(\hat{I}, \hat{H}, \hat{A}) \in \mathsf{InAmb} \times \mathsf{HNam} \times \mathsf{Accum}$; the individual components of the triple are described below.

For each ambient the set of ambients and capabilities contained in it is recorded in the following component:

$$\hat{I} \in \mathsf{InAmb} = \mathcal{P}(\mathsf{Lab}^a \times (\mathsf{Lab}^a \cup \mathsf{Lab}^t))$$

If a process contains an ambient labelled ℓ^a enclosing a capability or ambient labelled ℓ^{at} then $(\ell^a, \ell^{at}) \in \hat{I}$ should hold in order for $(\hat{I}, \hat{H}, \hat{A})$ to be a correct representation of the process.

Each occurrence of an ambient has a marker and to keep track of this information we have the component:

$$\hat{H} \in \textbf{HNam} = \mathcal{P}(\textbf{Lab}^a \times \textbf{Mar})$$

If a process contains an ambient labelled ℓ^a with marker μ then $(\ell^a, \mu) \in \hat{H}$ should hold in order for $(\hat{I}, \hat{H}, \hat{A})$ to be a correct representation of the process.

Furthermore the representation contains information about the number of occurrences of each ambient called the *multiplicity* of the ambient; this information is recorded in the $\hat{A}$-component:

$$\hat{A} \in \textbf{Accum} = \textbf{Lab}^a \to_{\text{fin}} (\textbf{Mult} \setminus \{0\})$$

where $\textbf{Mult} = \{0, 1, \omega\}$ (ω should be read as two or more, and 1 as exactly one) with an "addition" operator $\oplus$:

$\oplus$	0	1	ω
0	0	1	ω
1	1	ω	ω
ω	ω	ω	ω

If a process contains an ambient labelled ℓ^a then $\hat{A}(\ell^a)$ should be 1 or ω (depending on the actual number of occurrences of ℓ^a) in order for $(\hat{I}, \hat{H}, \hat{A})$ to be an acceptable representation of the process.

We say that $(\hat{I}, \hat{H}, \hat{A})$ is *compatible* if whenever $(\ell^a, \ell^{at}) \in \hat{I}$ then $(\{\ell^{at}\} \cap \textbf{Lab}^a) \subseteq \text{dom}(\hat{A})$ and whenever $(\ell^a, \mu) \in \hat{H}$ then $\ell^a \in \text{dom}(\hat{A})$.

A proposed analysis $\mathcal{C}$ is a powerset of representations of processes:

$$\mathcal{C} \in \textbf{CountSet} = \mathcal{P}(\textbf{Count})$$

where $\textbf{Count} = \{(\hat{I}, \hat{H}, \hat{A}) \in \textbf{InAmb} \times \textbf{HNam} \times \textbf{Accum} \mid (\hat{I}, \hat{H}, \hat{A}) \text{ is compatible}\}$.

Representation function. The representation function β^{OC} for a process is defined in terms of an extraction function η^{OC}:

$$\beta_\ell^{\text{OC}}(P) = \{\eta_\ell^{\text{OC}}(P)\}$$

The definition of η^{OC} is given in Fig. 4; it uses the operator $\uplus$ in order to combine representations of processes (note that $\uplus$ produces a compatible triple from two compatible triples):

$$(\hat{I}_1, \hat{H}_1, \hat{A}_1) \uplus (\hat{I}_2, \hat{H}_2, \hat{A}_2) = (\hat{I}_1 \cup \hat{I}_2, \hat{H}_1 \cup \hat{H}_2, \hat{A}_1 \oplus \hat{A}_2)$$

Here $\oplus$ is extended to **Accum** $\times$ **Accum** as follows:

$$(\hat{A}_1 \oplus \hat{A}_2)(\ell) = \begin{cases} \hat{A}_1(\ell) \oplus \hat{A}_2(\ell) & \text{if } \ell \in \mathrm{dom}(\hat{A}_1) \ \wedge \ \ell \in \mathrm{dom}(\hat{A}_2) \\ \hat{A}_1(\ell) & \text{if } \ell \in \mathrm{dom}(\hat{A}_1) \ \wedge \ \ell \notin \mathrm{dom}(\hat{A}_2) \\ \hat{A}_2(\ell) & \text{if } \ell \notin \mathrm{dom}(\hat{A}_1) \ \wedge \ \ell \in \mathrm{dom}(\hat{A}_2) \\ \text{undef} & \text{if } \ell \notin \mathrm{dom}(\hat{A}_1) \ \wedge \ \ell \notin \mathrm{dom}(\hat{A}_2) \end{cases}$$

The notation $[S \mapsto m]$ is used for the finitary function that is only defined on S and that gives the constant value m for all arguments in S. Notice that all ambients "inside" a replication are assigned the multiplicity ω as is natural due to the congruence axiom $!P \equiv P \mid !P$ that ensures that $!P \equiv P \mid \cdots \mid P \mid !P$. It is clear that the $\eta_\ell^{\mathrm{OC}}(P)$ is compatible.

Analysis specification. The specification of the analysis is given in Fig. 5 and is explained below. The specification is compositional and thereby well-defined.

The clause for $in^{\ell^t} n_\mu.P$ first checks the subprocess P. Then all representations of processes in $\mathcal{C}$ are considered. Each time the capability occurs inside an ambient ℓ^a with a sibling $\ell^{a'}$ with the marker μ, a demand on $\mathcal{C}$ is made depending on the multiplicity of ℓ^a. If ℓ^a has the multiplicity 1 it is demanded that the representation where ℓ^a no longer is within its parent $\ell^{a''}$ but has moved inside its sibling $\ell^{a'}$ is recorded in $\mathcal{C}$. Since we do not know whether or not a capability labelled ℓ^t occurs inside an ambient ℓ^a somewhere else in the process we have to make two demands, one where (ℓ^a, ℓ^t) has been removed from the representation and one where it has not. If ℓ^a has the multiplicity ω we cannot make any assumption on the number of ambients labelled ℓ^a that occur in inside the ambient $\ell^{a''}$. Therefore we make two demands on $\mathcal{C}$, one that represents the case that two or more ambients labelled ℓ^a occur inside $\ell^{a''}$ and one that represents the case that ℓ^a occurs exactly once inside $\ell^{a''}$. As for multiplicity 1 we have a demand where ℓ^t still occurs in the process and one where it does not. The clause for out-capabilities is similar.

The clause for $open^{\ell^t} n_\mu.P$ first checks the subprocess P. Then all representations of processes in $\mathcal{C}$ are considered. Each time the capability occurs with a sibling $\ell^{a'}$ with the marker μ, a demand on $\mathcal{C}$ is made depending on the multiplicity of $\ell^{a'}$. If $\ell^{a'}$ has the multiplicity 1 it is demanded that the representation where $\ell^{a'}$ has been removed from the process (using $\setminus$ to reduce the domain of $\hat{A}$) and the ambients and capabilities within it have moved inside the parent ℓ^a is recorded in $\mathcal{C}$. Again we have to make the same demand where ℓ^t no longer occurs inside ℓ^a. If $\ell^{a'}$ has the multiplicity ω a rather involved demand on $\mathcal{C}$ is made. Since we do not know which ambients and capabilities reported to be inside an ambient labelled $\ell^{a'}$ are inside the ambient that is actually opened we have to consider all subsets Z of the ambients and capabilities reported to be inside an ambient $\ell^{a'}$. Furthermore, we have to consider all subsets Y of Z, which represents the ambients and capabilities that only occur inside the ambient being opened. The ambients and capabilities in Z move inside the parent ambient ℓ^a and the ones from Y are removed from the representation. As for the other cases considered

$$
\begin{array}{ll}
\eta_\ell^{\mathrm{OC}}((\nu\, n)P) & = \eta_\ell^{\mathrm{OC}}(P) \\
\eta_\ell^{\mathrm{OC}}(\mathbf{0}) & = (\emptyset, \emptyset, \bot) \\
\eta_\ell^{\mathrm{OC}}(P \mid Q) & = \eta_\ell^{\mathrm{OC}}(P) \uplus \eta_\ell^{\mathrm{OC}}(Q) \\
\eta_\ell^{\mathrm{OC}}(!P) & = \text{ let } (\hat{I}, \hat{H}, \hat{A}) = \eta_\ell^{\mathrm{OC}}(P) \\
 & \quad\ \text{in } (\hat{I}, \hat{H}, [\mathrm{dom}(\hat{A}) \mapsto \omega]) \\
\eta_\ell^{\mathrm{OC}}(n_\mu^{\ell^a}[P]) & = \eta_{\ell^a}^{\mathrm{OC}}(P) \uplus (\{(\ell, \ell^a)\}, \{(\ell^a, \mu)\}, [\{\ell^a\} \mapsto 1]) \\
\eta_\ell^{\mathrm{OC}}(in^{\ell^t}\, n_\mu.P) & = \eta_\ell^{\mathrm{OC}}(P) \uplus (\{(\ell, \ell^t)\}, \emptyset, \bot) \\
\eta_\ell^{\mathrm{OC}}(out^{\ell^t}\, n_\mu.P) & = \eta_\ell^{\mathrm{OC}}(P) \uplus (\{(\ell, \ell^t)\}, \emptyset, \bot) \\
\eta_\ell^{\mathrm{OC}}(open^{\ell^t}\, n_\mu.P) & = \eta_\ell^{\mathrm{OC}}(P) \uplus (\{(\ell, \ell^t)\}, \emptyset, \bot)
\end{array}
$$

Fig. 4. Extraction function for occurrence counting

we have to include a representation where ℓ^t occurs in ℓ^a and one where it does not. Furthermore there is some bookkeeping (represented by U and m) to ensure that the $\hat{H}$ and $\hat{A}$ components are correct in special cases.

3.1 Properties of the Analysis

The following proposition establishes the semantic soundness of the occurrence counting analysis through a subject reduction result:

Proposition 1. *Let $P, Q \in$ **Proc**, $\ell_\star^a \in \mathbf{Lab}^a$ then*

$$\beta_{\ell_\star^a}^{\mathrm{OC}}(P) \subseteq \mathcal{C} \;\wedge\; \mathcal{C} \models^{\mathrm{OC}} P \;\wedge\; P \to Q \;\Rightarrow\; \beta_{\ell_\star^a}^{\mathrm{OC}}(Q) \subseteq \mathcal{C} \;\wedge\; \mathcal{C} \models^{\mathrm{OC}} Q$$

We next show that the set of acceptable solutions constitutes a Moore family. Recall that a subset of a complete lattice, $Y \subseteq L$, is a Moore family if whenever $Y' \subseteq Y$ then $\sqcap Y' \in Y$.

Proposition 2. *The set $\left\{\mathcal{C} \,\middle|\, \mathcal{C} \models^{\mathrm{OC}} P \;\wedge\; \beta_{\ell_\star^a}^{\mathrm{OC}}(P) \subseteq \mathcal{C}\right\}$ is a Moore family for every P and $\ell_\star^a$.*

The Moore family property implies that the counting analysis admits an analysis of every process, and that every process has a *least* or best analysis.

The set $\mathcal{C}$ may be infinite because it is a subset of the infinite set **Count**. However, it is possible to restrict the set **Count** to be finite by restricting all ambient labels and transition labels to be those occurring in the program $P_\star$ of interest; this defines the finite sets $\mathbf{Count}_\star$ and $\mathbf{CountSet}_\star$ of which $\mathcal{C}$ is a member. It follows that the least solution

$$\mathcal{C}_\star = \sqcap \left\{\mathcal{C} \in \mathbf{CountSet}_\star \,\middle|\, \mathcal{C} \models^{\mathrm{OC}} P_\star \;\wedge\; \beta_{\ell_\star^a}^{\mathrm{OC}}(P_\star) \sqsubseteq \mathcal{C}\right\}$$

is in fact computable. However, it is likely to require exponential time to compute $\mathcal{C}_\star$ and this motivates defining a yet coarser analysis.

$$
\begin{array}{ll}
\mathcal{C} \models^{\mathrm{OC}} (\nu\, n)P & \text{iff } \mathcal{C} \models^{\mathrm{OC}} P \\
\mathcal{C} \models^{\mathrm{OC}} \mathbf{0} & \text{always} \\
\mathcal{C} \models^{\mathrm{OC}} P \mid Q & \text{iff } \mathcal{C} \models^{\mathrm{OC}} P \;\wedge\; \mathcal{C} \models^{\mathrm{OC}} Q \\
\mathcal{C} \models^{\mathrm{OC}} !P & \text{iff } \mathcal{C} \models^{\mathrm{OC}} P \\
\mathcal{C} \models^{\mathrm{OC}} n_{\mu}^{\ell^a}[P] & \text{iff } \mathcal{C} \models^{\mathrm{OC}} P \\
\mathcal{C} \models^{\mathrm{OC}} \mathit{in}^{\ell^t} n_{\mu}.P & \text{iff } \mathcal{C} \models^{\mathrm{OC}} P \;\wedge
\end{array}
$$

$$
\begin{array}{l}
\forall(\hat{I},\hat{H},\hat{A}) \in \mathcal{C} : \forall \ell^a, \ell^{a'}, \ell^{a''} : \\
\quad \left(\begin{array}{l} (\ell^a,\ell^t) \in \hat{I} \;\wedge\; (\ell^{a''},\ell^a) \in \hat{I} \;\wedge \\ (\ell^{a''},\ell^{a'}) \in \hat{I} \;\wedge\; (\ell^{a'},\mu) \in \hat{H} \end{array} \right) \Rightarrow \\
\quad\quad \left(\begin{array}{l}
\text{case } \hat{A}(\ell^a) \text{ of} \\
1 : (\hat{I} \setminus \{(\ell^{a''},\ell^a)\} \cup \{(\ell^{a'},\ell^a)\}, \hat{H}, \hat{A}) \in \mathcal{C} \;\wedge \\
\quad (\hat{I} \setminus \{(\ell^{a''},\ell^a),(\ell^a,\ell^t)\} \cup \{(\ell^{a'},\ell^a)\}, \hat{H}, \hat{A}) \in \mathcal{C} \\
\omega : (\hat{I} \cup \{(\ell^{a'},\ell^a)\}, \hat{H}, \hat{A}) \in \mathcal{C} \\
\quad (\hat{I} \setminus \{(\ell^{a''},\ell^a)\} \cup \{(\ell^{a'},\ell^a)\}, \hat{H}, \hat{A}) \in \mathcal{C} \\
\quad (\hat{I} \setminus \{(\ell^a,\ell^t)\} \cup \{(\ell^{a'},\ell^a)\}, \hat{H}, \hat{A}) \in \mathcal{C} \\
\quad (\hat{I} \setminus \{(\ell^{a''},\ell^a),(\ell^a,\ell^t)\} \cup \{(\ell^{a'},\ell^a)\}, \hat{H}, \hat{A}) \in \mathcal{C}
\end{array} \right)
\end{array}
$$

$$
\begin{array}{l}
\mathcal{C} \models^{\mathrm{OC}} \mathit{out}^{\ell^t} n_{\mu}.P \quad \text{iff } \mathcal{C} \models^{\mathrm{OC}} P \;\wedge \\
\forall(\hat{I},\hat{H},\hat{A}) \in \mathcal{C} : \forall \ell^a, \ell^{a'}, \ell^{a''} : \\
\quad \left(\begin{array}{l} (\ell^a,\ell^t) \in \hat{I} \;\wedge\; (\ell^{a'},\ell^a) \in \hat{I} \;\wedge \\ (\ell^{a''},\ell^{a'}) \in \hat{I} \;\wedge\; (\ell^{a'},\mu) \in \hat{H} \end{array} \right) \Rightarrow \\
\quad\quad \left(\begin{array}{l}
\text{case } \hat{A}(\ell^a) \text{ of} \\
1 : (\hat{I} \setminus \{(\ell^{a'},\ell^a)\} \cup \{(\ell^{a''},\ell^a)\}, \hat{H}, \hat{A}) \in \mathcal{C} \\
\quad (\hat{I} \setminus \{(\ell^{a'},\ell^a),(\ell^a,\ell^t)\} \cup \{(\ell^{a''},\ell^a)\}, \hat{H}, \hat{A}) \in \mathcal{C} \\
\omega : (\hat{I} \cup \{(\ell^{a''},\ell^a)\}, \hat{H}, \hat{A}) \in \mathcal{C} \\
\quad (\hat{I} \setminus \{(\ell^a,\ell^t)\} \cup \{(\ell^{a''},\ell^a)\}, \hat{H}, \hat{A}) \in \mathcal{C} \\
\quad (\hat{I} \setminus \{(\ell^{a'},\ell^a)\} \cup \{(\ell^{a''},\ell^a)\}, \hat{H}, \hat{A}) \in \mathcal{C} \\
\quad (\hat{I} \setminus \{(\ell^{a'},\ell^a),(\ell^a,\ell^t)\} \cup \{(\ell^{a''},\ell^a)\}, \hat{H}, \hat{A}) \in \mathcal{C}
\end{array} \right)
\end{array}
$$

$$
\begin{array}{l}
\mathcal{C} \models^{\mathrm{OC}} \mathit{open}^{\ell^t} n_{\mu}.P \text{ iff } \mathcal{C} \models^{\mathrm{OC}} P \;\wedge \\
\forall(\hat{I},\hat{H},\hat{A}) \in \mathcal{C} : \forall \ell^a, \ell^{a'} : \\
\left((\ell^a,\ell^t) \in \hat{I} \;\wedge\; (\ell^a,\ell^{a'}) \in \hat{I} \;\wedge\; (\ell^{a'},\mu) \in \hat{H} \right) \Rightarrow \\
\quad \left(\begin{array}{l}
\text{case } \hat{A}(\ell^{a'}) \text{ of} \\
1 : \forall X \subseteq \{(\ell^a,\ell^t)\} : \\
\quad (\,\hat{I} \setminus (\{(\ell^{a'},\ell') \in \hat{I} \mid \ell' \in \textsf{Lab}\} \cup \{(\ell^a,\ell^{a'})\} \cup X) \cup \{(\ell^a,\ell') \mid (\ell^{a'},\ell') \in \hat{I}\}, \\
\quad\;\; \hat{H} \setminus \{(\ell^{a'},\mu) \mid \mu \in \textsf{Mar}\}, \hat{A} \setminus \{\ell^{a'}\}) \in \mathcal{C} \\
\omega : \forall Z \subseteq \{(\ell^{a'},\ell') \in \hat{I} \mid \ell' \in \textsf{Lab}\} : \forall Y \subseteq Z : \forall X \subseteq \{(\ell^a,\ell^t)\} : \\
\quad \forall U \subseteq \{(\ell^{a'},\mu)\} \text{ s.t. } U \neq \{(\ell^{a'},\mu') \in \hat{H} \mid \mu' \in \textsf{Mar}\} : \forall m \in \{1,\omega\} : \\
\quad ((\hat{I} \setminus (Y \cup X)) \cup \{(\ell^a,\ell') \mid (\ell^{a'},\ell') \in Z\}, \hat{H} \setminus U, \hat{A}[\ell^{a'} \mapsto m]) \in \mathcal{C}
\end{array} \right)
\end{array}
$$

Fig. 5. Specification of analysis with occurrence counting

$$
\begin{array}{lll}
(res) & \beta_\ell^{\mathrm{CF}}((\nu\, n)P) & = \beta_\ell^{\mathrm{CF}}(P) \\
(zero) & \beta_\ell^{\mathrm{CF}}(\mathbf{0}) & = (\emptyset, \emptyset) \\
(par) & \beta_\ell^{\mathrm{CF}}(P \mid Q) & = \beta_\ell^{\mathrm{CF}}(P) \sqcup \beta_\ell^{\mathrm{CF}}(Q) \\
(repl) & \beta_\ell^{\mathrm{CF}}(!P) & = \beta_\ell^{\mathrm{CF}}(P) \\
(amb) & \beta_\ell^{\mathrm{CF}}(n_\mu^{\ell^a}[P]) & = \beta_{\ell^a}^{\mathrm{CF}}(P) \sqcup (\{(\ell, \ell^a)\}, \{(\ell^a, \mu)\}) \\
(in) & \beta_\ell^{\mathrm{CF}}(in^{\ell^t} n_\mu.P) & = \beta_\ell^{\mathrm{CF}}(P) \sqcup (\{(\ell, \ell^t)\}, \emptyset) \\
(out) & \beta_\ell^{\mathrm{CF}}(out^{\ell^t} n_\mu.P) & = \beta_\ell^{\mathrm{CF}}(P) \sqcup (\{(\ell, \ell^t)\}, \emptyset) \\
(open) & \beta_\ell^{\mathrm{CF}}(open^{\ell^t} n_\mu.P) & = \beta_\ell^{\mathrm{CF}}(P) \sqcup (\{(\ell, \ell^t)\}, \emptyset)
\end{array}
$$

Fig. 6. Representation Function for CFA

$$
\begin{array}{lll}
(res) & (\hat{I}, \hat{H}) \models^{\mathrm{CF}} (\nu\, n)P & \text{iff } (\hat{I}, \hat{H}) \models^{\mathrm{CF}} P \\
(zero) & (\hat{I}, \hat{H}) \models^{\mathrm{CF}} \mathbf{0} & \text{always} \\
(par) & (\hat{I}, \hat{H}) \models^{\mathrm{CF}} P \mid Q & \text{iff } (\hat{I}, \hat{H}) \models^{\mathrm{CF}} P \ \wedge\ (\hat{I}, \hat{H}) \models^{\mathrm{CF}} Q \\
(repl) & (\hat{I}, \hat{H}) \models^{\mathrm{CF}} !P & \text{iff } (\hat{I}, \hat{H}) \models^{\mathrm{CF}} P \\
(amb) & (\hat{I}, \hat{H}) \models^{\mathrm{CF}} n_\mu^{\ell^a}[P] & \text{iff } (\hat{I}, \hat{H}) \models^{\mathrm{CF}} P \\
(in) & (\hat{I}, \hat{H}) \models^{\mathrm{CF}} in^{\ell^t} n_\mu.P & \text{iff } (\hat{I}, \hat{H}) \models^{\mathrm{CF}} P \wedge \\
& \multicolumn{2}{l}{\quad \forall \ell^a, \ell^{a'}, \ell^{a''} \in \mathsf{Lab}^a : ((\ell^a, \ell^t) \in \hat{I} \ \wedge\ (\ell^{a''}, \ell^a) \in \hat{I} \ \wedge\ (\ell^{a''}, \ell^{a'}) \in \hat{I}} \\
& \multicolumn{2}{l}{\qquad \wedge\ (\ell^{a'}, \mu) \in H) \ \Rightarrow\ (\ell^{a'}, \ell^a) \in \hat{I}} \\
(out) & (\hat{I}, \hat{H}) \models^{\mathrm{CF}} out^{\ell^t} n_\mu.P & \text{iff } (\hat{I}, \hat{H}) \models^{\mathrm{CF}} P \wedge \\
& \multicolumn{2}{l}{\quad \forall \ell^a, \ell^{a'}, \ell^{a''} \in \mathsf{Lab}^a : ((\ell^a, \ell^t) \in \hat{I} \ \wedge\ (\ell^{a'}, \ell^a) \in \hat{I} \ \wedge\ (\ell^{a''}, \ell^{a'}) \in \hat{I}} \\
& \multicolumn{2}{l}{\qquad \wedge\ (\ell^{a'}, \mu) \in H) \ \Rightarrow\ (\ell^{a''}, \ell^a) \in \hat{I}} \\
(open) & (\hat{I}, \hat{H}) \models^{\mathrm{CF}} open^{\ell^t} n_\mu.P & \text{iff } (\hat{I}, \hat{H}) \models^{\mathrm{CF}} P \wedge \\
& \multicolumn{2}{l}{\quad \forall \ell^a, \ell^{a'} \in \mathsf{Lab}^a : ((\ell^a, \ell^t) \in \hat{I} \ \wedge\ (\ell^a, \ell^{a'}) \in \hat{I} \ \wedge\ (\ell^{a'}, \mu) \in H)} \\
& \multicolumn{2}{l}{\qquad \Rightarrow \left\{(\ell^a, \ell') \mid (\ell^{a'}, \ell') \in \hat{I}\right\} \subseteq \hat{I}}
\end{array}
$$

Fig. 7. Specification of CFA

4 Control Flow Analysis

The control flow analysis can be obtained from the counting analysis by dispensing with the $\hat{A}$ component and by merging the resulting pairs $(\hat{I}, \hat{H})$ into one (by taking their least upper bound). In other words, the analysis works on pairs $(\hat{I}, \hat{H}) \in \mathsf{InAmb} \times \mathsf{HNam}$.

In keeping with the counting analysis, the control flow analysis is defined by a representation function and a specification. The representation function is shown in Fig. 6 and the analysis specification in Fig. 7. The specification of the analysis for the cases (*res*), (*zero*), (*par*), (*repl*) and (*amb*) are merely recursive checks of subprocesses. The case (*in*) states that if some ambient, labelled ℓ^a, has an *in*-capability (denoted by $(\ell^a, \ell^t) \in \hat{I}$) and has a sibling (denoted $(\ell^{a''}, la) \in \hat{I} \wedge (\ell^{a''}, \ell^{a'}) \in \hat{I}$) with the right name (denoted $(\ell^{a'}, \mu) \in \hat{H}$) then the possibility of that ambient using the *in*-capability should also be recorded in $\hat{I}$ (denoted $(\ell^{a'}, \ell^a) \in \hat{I}$). The cases (*out*) and (*open*) are similar.

Example 2. Recall the process, P, of Example 1 and let P and Q equal $\mathbf{0}$ for simplicity. For $\ell^a_\star \in \mathbf{Lab}^a$, the least analysis to $\beta^{\mathrm{CF}}_{\ell^a_\star}(\mathsf{P}) \sqsubseteq (\hat{I}, \hat{H}) \ \wedge \ (\hat{I}, \hat{H}) \models^{\mathrm{CF}} \mathsf{P}$ is given by:

$$\begin{aligned}\hat{I} = \{&(\ell^a_\star, \ell^{a''}), (\ell^a_\star, \ell^{a'}), (\ell^a_\star, \ell^a), (\ell^a, \ell^{a''}), (\ell^a, \ell^{a'}), (\ell^{a'}, \ell^t_1), (\ell^{a'}, \ell^t_2), (\ell^{a'}, \ell^t_3),\\ &(\ell^{a''}, \ell^{a''}), (\ell^{a''}, \ell^{a'}), (\ell^{a''}, \ell^t_1), (\ell^{a''}, \ell^t_2), (\ell^{a''}, \ell^t_3), (\ell^{a''}, \ell^t_4)\}\\ \hat{H} = \{&(\ell^{a''}, \mu''), (\ell^{a'}, \mu'), (\ell^a, \mu)\}\end{aligned}$$

where all counting information of the reduction sequence has been discarded and the remaining information has been marged. $\square$

The control flow analysis was developed in [7] where the formal properties of the analysis were also established and where the analysis was used to validate the protectiveness of a proposed firewall. Below we first develop a notion of how to induce constraint-based analyses and then show that the analysis can also be induced from the counting analysis, whence the formal properties of the control flow analysis also follow from the general framework presented here.

4.1 Systematic Construction of Analyses

The framework presented in this paper lends itself to a systematic approach analogous to the use of Galois connections in abstract interpretation. In what follows we assume that $\sqsubseteq_{\mathcal{A}}$ is an ordering of the complete lattice $\mathcal{A}$ and that $\sqsubseteq_{\mathcal{A}'}$ is an ordering of the complete lattice $\mathcal{A}'$; we shall write $\sqsubseteq$ for both orderings where no confusion can arise. Here a Galois connection between $\mathcal{A}$ and $\mathcal{A}'$, denoted $\mathcal{A}' \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \mathcal{A}$, is a pair of monotone functions, $\alpha : \mathcal{A}' \to \mathcal{A}$ and $\gamma : \mathcal{A} \to \mathcal{A}'$, such that $\mathrm{id}_{\mathcal{A}'} \sqsubseteq \gamma \circ \alpha$ and $\alpha \circ \gamma \sqsubseteq \mathrm{id}_{\mathcal{A}}$. We are now in a position to define the key notion of an *induced satisfaction relation*:

Definition 1 (Induced Satisfaction Relation). *Let* $\mathcal{A}' \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \mathcal{A}$ *be a Galois connection. A satisfaction relation* $\models : \mathbf{Proc} \times \mathcal{A} \to \{\mathsf{tt}, \mathsf{ff}\}$ *is said to be* induced *from another satisfaction relation* $\models' : \mathbf{Proc} \times \mathcal{A}' \to \{\mathsf{tt}, \mathsf{ff}\}$*, when the following holds:*

$$\forall A \in \mathcal{A}, P \in \mathbf{Proc} : A \models P \iff \gamma(A) \models' P$$

Analogously, representation functions may be induced:

Definition 2 (Induced Representation Function). *Let* $\mathcal{A}' \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \mathcal{A}$ *be a Galois connection, then a representation function,* $\beta : \textsf{Proc} \rightarrow \mathcal{A}$ *is said to be* induced *from a representation function,* $\beta' : \textsf{Proc} \rightarrow \mathcal{A}'$ *whenever:*

$$\beta = \alpha \circ \beta'$$

An induced satisfaction relation inherits several important formal properties of the original satisfaction relation, such as subject reduction and Moore family properties. We first show that subject reduction is preserved:

Proposition 3. *Let* $\models : \textsf{Proc} \times \mathcal{A} \rightarrow \{\mathsf{tt}, \mathsf{ff}\}$*,* $\beta : \textsf{Proc} \rightarrow \mathcal{A}$ *and* $\models' : \textsf{Proc} \times \mathcal{A}' \rightarrow \{\mathsf{tt}, \mathsf{ff}\}$*,* $\beta' : \textsf{Proc} \rightarrow \mathcal{A}'$ *be given such that* $\models$ *and* β *are induced from* $\models'$ *and* β' *respectively, via the Galois connection* $\mathcal{A}' \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \mathcal{A}$ *. Then*

$$\beta'(P) \sqsubseteq A' \;\wedge\; A' \models' P \;\wedge\; P \rightarrow Q \;\Rightarrow\; \beta'(Q) \sqsubseteq A' \;\wedge\; A' \models' Q$$

implies

$$\beta(P) \sqsubseteq A \;\wedge\; A \models P \;\wedge\; P \rightarrow Q \;\Rightarrow\; \beta(Q) \sqsubseteq A \;\wedge\; A \models Q$$

Proof. We calculate as follows:

$$\begin{array}{l} \beta(P) \sqsubseteq A \;\wedge\; A \models P \;\wedge\; P \rightarrow Q \\ \quad \Rightarrow\; \beta'(P) \sqsubseteq \gamma(A) \;\wedge\; \gamma(A) \models' P \;\wedge\; P \rightarrow Q \\ \quad \Rightarrow\; \beta'(Q) \sqsubseteq \gamma(A) \;\wedge\; \gamma(A) \models' Q \\ \quad \Rightarrow\; \beta(Q) \sqsubseteq A \;\wedge\; A \models Q \end{array}$$

This concludes the proof. □

Next we show that the Moore family property is also preserved:

Proposition 4. *Let* $\models : \textsf{Proc} \times \mathcal{A} \rightarrow \{\mathsf{tt}, \mathsf{ff}\}$*,* $\beta : \textsf{Proc} \rightarrow \mathcal{A}$ *and* $\models' : \textsf{Proc} \times \mathcal{A}' \rightarrow \{\mathsf{tt}, \mathsf{ff}\}$*,* $\beta' : \textsf{Proc} \rightarrow \mathcal{A}'$ *be given such that* $\models$ *and* β *are induced from* $\models'$ *and* β' *respectively, via the Galois connection* $\mathcal{A}' \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \mathcal{A}$ *. If*

$$\{A' \in \mathcal{A}' \,|\, A' \models' P \;\wedge\; \beta'(P) \sqsubseteq A'\}$$

is a Moore family for every P*, then*

$$\{A \in \mathcal{A} \,|\, A \models P \;\wedge\; \beta(P) \sqsubseteq A\}$$

is also a Moore family for every P*.*

$$
\begin{tikzcd}
\textsf{CountSet} \arrow[rr, shift right, "\alpha^{\mathrm{CF}}"'] & & \textsf{InAmb} \times \textsf{HNam} \arrow[ll, shift right, "\gamma^{\mathrm{CF}}"'] \\
& \textsf{Proc} \arrow[ul, "\beta^{\mathrm{OC}}"] \arrow[ur, "\beta^{\mathrm{CF}}"'] &
\end{tikzcd}
$$

Fig. 8. Galois connection

Proof. Let $P \in \textsf{Proc}$ and $A_i \in \mathcal{A}$ for all $i \in \mathcal{I}$ where $\mathcal{I}$ is an index set. We then calculate as follows:

$$
\begin{array}{l}
\forall i \in \mathcal{I} : (A_i \models P \;\wedge\; \beta(P) \sqsubseteq A_i) \\
\quad \Rightarrow \forall i \in \mathcal{I} : (\gamma(A_i) \models' P \;\wedge\; \beta'(P) \sqsubseteq \gamma(A_i)) \\
\quad \Rightarrow \sqcap\gamma(A_i) \models' P \;\wedge\; \beta'(P) \sqsubseteq \sqcap\gamma(A_i) \\
\quad \Rightarrow \gamma(\sqcap A_i) \models' P \;\wedge\; \beta'(P) \sqsubseteq \gamma(\sqcap A_i) \\
\quad \Rightarrow \sqcap A_i \models P \;\wedge\; \beta(P) \sqsubseteq \sqcap A_i
\end{array}
$$

This concludes the proof. $\square$

4.2 Properties of the Analysis

In order to show that the control flow analysis is induced from the counting analysis we need a Galois connection. In the following $\sqsubseteq$ is the coordinatwise ordering on $\textsf{InAmb} \times \textsf{HNam}$. We define

$$\alpha^{\mathrm{CF}}(\mathcal{C}) = \bigsqcup\{(\hat{I}, \hat{H}) \,|\, (\hat{I}, \hat{H}, \hat{A}) \in \mathcal{C}\}$$

$$\gamma^{\mathrm{CF}}(\hat{I}, \hat{H}) = \{(\hat{I}', \hat{H}', \hat{A}') \,|\, (\hat{I}', \hat{H}') \sqsubseteq (\hat{I}, \hat{H}) \;\wedge\; (\hat{I}', \hat{H}', \hat{A}') \text{ is compatible}\}$$

and note that $\textsf{CountSet} \underset{\alpha^{\mathrm{CF}}}{\overset{\gamma^{\mathrm{CF}}}{\leftrightarrows}} \textsf{InAmb} \times \textsf{HNam}$ is a Galois connection (see Fig. 8). The following proposition then states that the control flow analysis is *induced*, cf. Definition 1, from the analysis with occurrence counting:

Proposition 5. *Let* $P \in \textsf{Proc}$ *and* $\ell \in \textsf{Lab}^a$ *then*

$$(\hat{I}, \hat{H}) \models^{\mathrm{CF}} P \iff \gamma^{\mathrm{CF}}((\hat{I}, \hat{H})) \models^{\mathrm{OC}} P$$

and

$$\beta^{\mathrm{CF}}_{\ell}(P) = \alpha^{\mathrm{CF}}(\beta^{\mathrm{OC}}_{\ell}(P))$$

From the Propositions 1, 3 and 5 and the semantic correctness of the control flow analysis immediately follows:

Corollary 1. *Let* $P, Q \in \textsf{Proc}$ *and* $\ell^a_\star \in \textsf{Lab}^a$ *then*

$$\begin{array}{l}\beta^{\mathrm{CF}}_{\ell^a_\star}(P) \sqsubseteq (\hat{I}, \hat{H}) \ \wedge \ (\hat{I}, \hat{H}) \models^{\mathrm{CF}} P \ \wedge \ P \to Q \ \Rightarrow \\ \quad \beta^{\mathrm{CF}}_{\ell^a_\star}(Q) \sqsubseteq (\hat{I}, \hat{H}) \ \wedge \ (\hat{I}, \hat{H}) \models^{\mathrm{CF}} Q\end{array}$$

Furthermore, the Moore family property of the control flow analysis follows from Propositions 2, 5 and 4:

Corollary 2. *The set* $\left\{(\hat{I}, \hat{H}) \,\middle|\, (\hat{I}, \hat{H}) \models^{\mathrm{CF}} P \ \wedge \ \beta^{\mathrm{CF}}_{\ell^a_\star}(P) \sqsubseteq (\hat{I}, \hat{H})\right\}$ *is a Moore family for every* P *and* $\ell^a_\star$*.*

In [5] it is shown that by restricting the attention to a given system, $n^{\ell^a_\star}_{\mu_\star}[P_\star]$, of size s, it is possible to devise an $O(s^4)$ algorithm for computing the least solution to the control flow analysis.

5 Conclusion

We have shown that abstract interpretation can be formulated in a constraint-based manner that makes it useful for analysing calculi of computation such as the ambient calculus and without the need to assume processes to be on a simplified form. The development mimics the familiar developments of using abstract interpretation to induce general abstract transfer functions from others. A counting analysis is shown to be semantically correct and to have solutions that constitute a Moore family; a previously developed control flow analysis is induced from the counting analysis and its properties derived. In our view this development demonstrates that abstract interpretation and constraint-based analyses naturally complement the use of type systems for analysing calculi of computation.

References

1. C. Bodei, P. Degano, F. Nielson, and H. R. Nielson. Control Flow Analysis for the π-calculus. In *Proc. CONCUR'98*, volume 1466 of *Lecture Notes in Computer Science*, pages 84–98. Springer-Verlag, 1998.
2. C. Bodei, P. Degano, F. Nielson, and H. R. Nielson. Static Analysis of Processes for No Read-Up and No Write-Down. In *Proc. FoSSaCS'99*, volume 1578 of *Lecture Notes in Computer Science*, pages 120–134. Springer-Verlag, 1999.
3. L. Cardelli and A. D. Gordon. Mobile Ambients. In *Proc. FoSSaCS'98*, volume 1378 of *Lecture Notes in Computer Science*, pages 140–155. Springer-Verlag, 1998.
4. L. Cardelli and A. D. Gordon. Types for Mobile Ambients. In *Proc. POPL'99*, pages 79–92. ACM Press, 1999.
5. R. R. Hansen and J. G. Jensen. Flow Logics for Mobile Ambients. M.Sc. thesis, Aarhus University, 1999.

6. T. Jensen, D. LeMétayer, and T. Thorn. Security and Dynamic Class Loading in Java: A Formalisation. Technical report, IRISA, Rennes, 1997.
7. F. Nielson, H. R. Nielson, R. R. Hansen, and J. G. Jensen. Validating Firewalls in Mobile Ambients. In *Proc. CONCUR'99*, Lecture Notes in Computer Science. Springer-Verlag, 1999. To appear.
8. A. Venet. Abstract Interpretation of the π-calculus. In *Analysis and Verification of Multiple-Agent Languages*, volume 1192 of *Lecture Notes in Computer Science*, pages 51–75. Springer-Verlag, 1997.
9. A. Venet. Automatic Determination of Communication Topologies in Mobile Systems. In *Proc. SAS'98*, volume 1503 of *Lecture Notes in Computer Science*, pages 152–167. Springer-Verlag, 1998.