UNIVERSITY
OF TRENTO

DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY

38050 Povo – Trento (Italy), Via Sommarive 14
http://www.dit.unitn.it

CONTROL FLOW ANALYSIS FOR BIOAMBIENTS

Flemming Nielson, Hanne Riis Nielson, Corrado Priami
and Debora Schuch da Rosa

June 2003

Technical Report # DIT-03-036

Control Flow Analysis for BioAmbients

Flemming Nielson, Hanne Riis Nielson
Technical University of Denmark

Corrado Priami, Debora Schuch da Rosa
University of Trento

June 7, 2003

Abstract This paper presents a static analysis for investigating properties of biological systems specified in BioAmbients. We exploit the control flow analysis to decode the bindings of variables induced by communications and to build a relation of the ambients that can interact with each other. We eventually apply our analysis to an example of gene regulation by positive feedback taken from the literature.

1 Introduction and Motivation

Modelling of biological systems is a challenge for computer science [27]. In fact the complexity of these systems is some order of magnitude larger than computer systems ever built. Furthermore, the modelling of dynamical behaviour of biological systems is becoming an urgent need for biologists that are trying to coherently organize the huge amount of data available in the post-genomic era. This paper is a step towards the definition of modelling environments for biologists that can assist them in the definition and analysis of complex systems.

Promising approaches based on process algebras exists to model and simulate the dynamic behaviour of molecular systems. The pioneering work on modeling biochemical systems with a calculus is [10] where a version of the λ-calculus is used. A better account of pathways descriptions is proposed by [28] via a calculus for mobility where processes represent compounds and communications represent interactions. Then, [24] enriched this model with quantitative aspects. Along the same line, we mention also the Bio-calculus proposed in [16].

A process algebra called *Core Formal Molecular Biology* has been recently proposed in [7]. The new calculus builds on the basic primitives of the π-calculus. As in the other language-based models mentioned above, processes represent compounds, sets of processes represent solutions, and their behaviour is given by a set of rewriting rules, driven by suitable side-conditions. The proposed rules are related to the biological realm and mimic typical reactions that occur in biochemical networks, *e.g.* activation, synthesis, complexation etc.

Recently, Regev proposed *BioAmbients* [25], a variant of the Ambient Calculus [6] in which compartments are described as a hierarchy of boundary ambients. This hierarchy can be modified by suitable operations that have an immediate biological interpretation. For example, the *enter* n primitive, that moves an ambient into a (sibling) ambient n, models a compartment entry. Ambients contain compounds that interact via communications. A communication is only possible if the involved processes obey to some constraints, e.g. either they are in the same compartment (*local* communications), or they belong to two parallel compartments (*sibling* communication), or they belong to two ambients one within the other (*parent-child* communication). The original presentation of BioAmbients has been refined in [26, 5].

All the work mentioned above describe the behaviour of molecular systems by relying on a transition

system representation that then can be explored to investigate the properties of interest. The main limitation of this approach is the huge size of the representation. In fact the size of the transition system is exponential in the size of the program representing the behaviour. In other words all the proposals above implement a dynamic analysis of systems.

The classical alternative to dynamic analysis, when the size of the representations is too large, is static analysis [17]. It only needs the text of the program and can infer suitable properties on the behaviour of the system modelled. The technique is much more efficient, but one has to pay a loss in the precision of the properties checked. Historically, static analysis techniques have been developed in the context of optimising compilers and only within the last few years they have been successfully used for validating programs in process calculi. In the classical application domains it is customary that the complete program is available for analysis and hence the techniques have focused on *closed* programs. Previous work have shown that static analysis approaches can handle a variety of the necessary constructs including mobility and communication primitives as in the π-calculus [3], Mobile Ambients [20, 18] and Boxed Ambients [20].

We here introduce a static approach for analysing molecular processes specified in BioAmbients. To the best of our knowledge this is the first attempt at exploiting static analysis in this biological application domain. The aim of the analysis is to keep track of the contents of the ambients and the bindings of the names that may vary when communications occur. The content of ambients is abstracted by annotating specifications with group tags; therefore we cannot distinguish two different ambients annotated with the same group. The initial bindings are recovered by the standard binding rules of the operators of the calculus. According to this information we build two relations describing the bindings of names and the contents of ambients. The relations are updated while scanning the specification and analysing the potential communications that may alter the bindings of names and the potential execution of capabilities that may change the contents of ambients. We show how the analysis works by modelling in BioAmbients an example already published in [24] and specified there in the π-calculus.

The exploitation of the results of our analysis in the biological setting is immediate. For instance, we can use our analysis to establish whether two ambients may interact (e.g., a protein with a degradation factor or with another protein) or whether there exists a flow of information from one molecule to another. Due to the efficiency of the solver of the constraints of the analysis we are able to handle larger molecular networks than dynamic analysis, although our approach only suggests potential interaction. This could be a breaktrough in the analysis of complex pathways to establish relation between elements that are not directly related in the available representations on public databases (e.g., EcoCyc [12], WIT [30], KEGG [22], CSNDB [11], aMAZE [31], GeNet [13], TransPath [32], INETRACT [9], DIP [33], BIND [2], SPAD [1], and Flynets [29]).

The paper is organized as follows. In the next section we recall the basics of BioAmbients. Section 3 introduces the analysis technique and Section 4 then applies it to an example taken from the literature. We eventually draw some conclusions.

2 BioAmbients

BioAmbients [25, 26, 5] differ from Mobile Ambients [6] in two main respects:

- The ambients are nameless entities; however, their roles may be indicated by comments. We shall therefore assume that we have a rudimentary type structure: each ambient belongs to a group and hence we shall write $[P]^{\mu}$ to clarify that the ambient $[P]$ belongs to the group μ.
- The capabilities are based on pure names n with no internal structure. Reactions are synchronous and both the object and the subject must agree on the reaction in order for it to happen; the latter is accomplished by having pairs of capabilities react with each other.

Furthermore, the set of control structures for processes is slightly larger than what is traditionally studied for Mobile Ambients in that it includes non-deterministic choice as well as a general recursion construct

$$\vdash^s_{\mathsf{wf}} 0 \qquad \frac{\vdash^p_{\mathsf{wf}} P}{\vdash^s_{\mathsf{wf}} M.P} \qquad \frac{\vdash^s_{\mathsf{wf}} P}{\vdash^s_{\mathsf{wf}} (n)P} \qquad \frac{\vdash^s_{\mathsf{wf}} P \quad \vdash^s_{\mathsf{wf}} P'}{\vdash^s_{\mathsf{wf}} P + P'} \qquad \frac{\vdash^s_{\mathsf{wf}} P}{\vdash^s_{\mathsf{wf}} \mathsf{rec}\, X.\, P} \qquad \vdash^s_{\mathsf{wf}} X$$

$$\frac{\vdash^s_{\mathsf{wf}} P}{\vdash^p_{\mathsf{wf}} P} \qquad \frac{\vdash^p_{\mathsf{wf}} P}{\vdash^p_{\mathsf{wf}} (n)P} \qquad \frac{\vdash^p_{\mathsf{wf}} P \quad \vdash^p_{\mathsf{wf}} P'}{\vdash^p_{\mathsf{wf}} P \mid P'} \qquad \frac{\vdash^p_{\mathsf{wf}} P}{\vdash^p_{\mathsf{wf}} [P]^\mu}$$

Table 1: Well-formedness predicates: $\vdash^p_{\mathsf{wf}} P$ and $\vdash^s_{\mathsf{wf}} P$.

in the manner of CCS [15].

The syntax of the processes P and the capabilities M are given by:

P	::=	$\mathsf{0}$	inactive process
	|	$(n)P$	binding box for the name n
	|	$[P]^\mu$	ambient enclosing P in group μ
	|	$M.P$	prefixing with capability M
	|	$P \mid P'$	parallel processes
	|	$P + P'$	non-deterministic choice
	|	$\mathsf{rec}\, X.\, P$	recursive process ($X = P$)
	|	X	process variable
M	::=	$\mathsf{enter}\ n \mid \mathsf{accept}\ n$	enter movement
	|	$\mathsf{exit}\ n \mid \mathsf{expel}\ n$	exit movement
	|	$\mathsf{merge{+}}\ n \mid \mathsf{merge{-}}\ n$	merge movement
	|	$n!\{m\} \mid n?\{p\}$	local communication
	|	$n\_!\{m\} \mid n\hat{}?\{p\}$	to child communication
	|	$n\hat{}!\{m\} \mid n\_?\{p\}$	to parent communication
	|	$n\#!\{m\} \mid n\#?\{p\}$	to sibling communication

The $\mathsf{enter}/\mathsf{accept}$ and $\mathsf{exit}/\mathsf{expel}$ capabilities are analogous to the $\mathsf{in}/\overline{\mathsf{in}}$ and $\mathsf{out}/\overline{\mathsf{out}}$ capabilities of Mobile Ambients and its variants, Safe Ambients [14] and Discretionary Ambients [20]. There is no analogue of the $\mathsf{open}/\overline{\mathsf{open}}$ capabilities, rather there is a $\mathsf{merge{+}}/\mathsf{merge{-}}$ construct that disolves the boundary of one ambient and includes its contents in another.

The communication primitives of BioAmbients are somewhat different from those of Mobile Ambients in that they use names as channels and furthermore only names can be exchanged as a result of the communication. As for Mobile Ambients two processes can communicate if they run in parallel within an ambient; this is called local communication. However, they may also communicate if they belong to ambients that are siblings or where one is a child of the other. As for Boxed Ambients [4] the latter gives rise to two kinds of communication depending on whether information flows from the child to the parent or the other way. Compared to the π-calculus [15] the names of channels are used in a localised manner.

The syntax is subject to a well-formedness condition that ensures that a top-level process has no free process variables and that it basically is a parallel composition of a number of processes that each is a sum of processes. The latter condition is formalised by the predicate $\vdash^p_{\mathsf{wf}} P$ defined in Table 1; it makes use of the auxiliary predicate $\vdash^s_{\mathsf{wf}} P$ holding on sums of guarded processes. The well-formedness conditions are somewhat more liberal than the syntactic rules for sum and prefixing put forward in [26].

The semantics is given in the classical way using a congruence relation $\equiv$ and a transition relation $\rightarrow$. The congruence relation is defined in Table 2; here we write $\mathsf{fn}(P)$ for the set of free names in P, $\mathsf{fn}(M)$ resp. $\mathsf{bn}(M)$ for the free resp. bound names of M and we write $P[m/n]$ for the process that is as P except that all free occurrences of n are replaced by m (subject to alpha-renaming of bound names). A similar notation is used for free process variables, $\mathsf{fv}(P)$, and substitutions of free process variables, $P[Y/X]$. The transition relation $\rightarrow$ is defined in Table 3. The well-formedness condition ensures that we have:

Proposition. If $\vdash^p_{\mathsf{wf}} P$ and $P \rightarrow Q$ then there exists Q' such that $\vdash^p_{\mathsf{wf}} Q'$ and $Q' \equiv Q$.

Alpha-renaming of bound names and bound variables:

$$\begin{array}{rcll}
(n)P & \equiv & (m)P[m/n] & \text{if} \quad m \notin \mathsf{fn}(P) \\
n?\{p\}.\,P & \equiv & n?\{q\}.\,P[q/p] & \text{if} \quad q \notin \mathsf{fn}(P) \\
n_?\{p\}.\,P & \equiv & n_?\{q\}.\,P[q/p] & \text{if} \quad q \notin \mathsf{fn}(P) \\
n\hat{}?\{p\}.\,P & \equiv & n\hat{}?\{q\}.\,P[q/p] & \text{if} \quad q \notin \mathsf{fn}(P) \\
n\#?\{p\}.\,P & \equiv & n\#?\{q\}.\,P[q/p] & \text{if} \quad q \notin \mathsf{fn}(P) \\
\mathsf{rec}\,X.\,P & \equiv & \mathsf{rec}\,Y.\,P[Y/X] & \text{if} \quad Y \notin \mathsf{fv}(P)
\end{array}$$

Reordering of parallel processes:

$$\begin{array}{rcl}
P \mid P' & \equiv & P' \mid P \\
(P \mid P') \mid P'' & \equiv & P \mid (P' \mid P'') \\
P \mid 0 & \equiv & P
\end{array}$$

Reordering of sum processes:

$$\begin{array}{rcl}
P + P' & \equiv & P' + P \\
(P + P') + P'' & \equiv & P + (P' + P'') \\
P + 0 & \equiv & P
\end{array}$$

Scope rules for name bindings:

$$\begin{array}{rcll}
(n)0 & \equiv & 0 & \\
(n_1)(n_2)P & \equiv & (n_2)(n_1)P & \text{if} \quad n_1 \neq n_2 \\
(n)(P \mid P') & \equiv & ((n)P) \mid P' & \text{if} \quad n \notin \mathsf{fn}(P') \\
(n)([P]^{\mu}) & \equiv & [(n)P]^{\mu} & \\
(n)(P + P') & \equiv & (n)P + (n)P' & \\
(n)(M.P) & \equiv & M.((n)P) & \text{if} \quad n \notin \mathsf{fn}(M) \cup \mathsf{bn}(M)
\end{array}$$

Table 2: Structural congruence relation: $P \equiv P'$.

Movement of ambients:

$$\begin{array}{r}
[(\mathsf{enter}\ n.\,P + P') \mid P'']^{\mu_1} \mid [(\mathsf{accept}\ n.\,Q + Q') \mid Q'']^{\mu_2} \to [\,[P \mid P'']^{\mu_1} \mid Q \mid Q'']^{\mu_2} \\
[\,[(\mathsf{exit}\ n.\,P + P') \mid P'']^{\mu_1} \mid (\mathsf{expel}\ n.\,Q + Q') \mid Q'']^{\mu_2} \to [P \mid P'']^{\mu_1} \mid [Q \mid Q'']^{\mu_2} \\
[(\mathsf{merge}{+}\ n.\,P + P') \mid P'']^{\mu_1} \mid [(\mathsf{merge}{-}\ n.\,Q + Q') \mid Q'']^{\mu_2} \to [P \mid P'' \mid Q \mid Q'']^{\mu_1}
\end{array}$$

Communication between ambients:

$$\begin{array}{r}
(n!\{m\}.\,P + P') \mid (n?\{p\}.\,Q + Q') \to P \mid Q[m/p] \\
(n_!\{m\}.\,P + P') \mid [(n\hat{}?\{p\}.\,Q + Q') \mid Q']^{\mu} \to P \mid [Q[m/p] \mid Q']^{\mu} \\
[(n\hat{}!\{m\}.\,P + P') \mid P']^{\mu} \mid (n_?\{p\}.\,Q + Q') \to [P \mid P']^{\mu} \mid Q[m/p] \\
[(n\#!\{m\}.\,P + P') \mid P'']^{\mu_1} \mid [(n\#?\{p\}.\,Q + Q') \mid Q'']^{\mu_2} \to [P \mid P'']^{\mu_1} \mid [Q[m/p] \mid Q'']^{\mu_2}
\end{array}$$

Execution in context:

$$\frac{P \to Q}{(n)P \to (n)Q} \qquad \frac{P \to Q}{[P]^{\mu} \to [Q]^{\mu}} \qquad \frac{P \to Q}{P \mid R \to Q \mid R} \qquad \frac{P[\mathsf{rec}\,X.\,P/X] \to Q}{\mathsf{rec}\,X.\,P \to Q}$$

$$\frac{P \equiv P' \quad P' \to Q' \quad Q' \equiv Q}{P \to Q}$$

Table 3: Transition relation: $P \to P'$.

3 Analysis

The aim of the analysis is to keep track of the contents of ambients and the bindings of names; it amounts to an adaption of ideas presented in [20]. We shall use the group structure to identify the ambients; hence if two ambients are annotated with the same group μ then the analysis will not be able to distinguish between them. In the rather simple analysis developed here we shall write **Group** for the set of groups and assume that it is finite.

As we have seen the names are subject to alpha-renaming so they cannot be used to carry information in the analysis. The usual way to circumvent this problem is to assume that each name n has a *canonical name* written $\lfloor n \rfloor$, and then assume that canonical names are preserved under alpha-renaming, i.e. that

$(\mathcal{I},\mathcal{R}) \models^\star \mathsf{0}$	iff	true
$(\mathcal{I},\mathcal{R}) \models^\star (n)P$	iff	$\lfloor n \rfloor \in \mathcal{R}(\lfloor n \rfloor) \wedge (\mathcal{I},\mathcal{R}) \models^\star P$
$(\mathcal{I},\mathcal{R}) \models^\star [P]^\mu$	iff	$\mu \in \mathcal{I}(\star) \wedge (\mathcal{I},\mathcal{R}) \models^\mu P$
$(\mathcal{I},\mathcal{R}) \models^\star M.P$	iff	$(\mathcal{I},\mathcal{R}) \models^\star M \wedge (\mathcal{I},\mathcal{R}) \models^\star P$
$(\mathcal{I},\mathcal{R}) \models^\star P \mid P'$	iff	$(\mathcal{I},\mathcal{R}) \models^\star P \wedge (\mathcal{I},\mathcal{R}) \models^\star P'$
$(\mathcal{I},\mathcal{R}) \models^\star P + P'$	iff	$(\mathcal{I},\mathcal{R}) \models^\star P \wedge (\mathcal{I},\mathcal{R}) \models^\star P'$
$(\mathcal{I},\mathcal{R}) \models^\star \mathsf{rec}\, X.\, P$	iff	$(\mathcal{I},\mathcal{R}) \models^\star P$
$(\mathcal{I},\mathcal{R}) \models^\star X$	iff	true

Table 4: Analysis of processes: $(\mathcal{I},\mathcal{R}) \models^\star P$.

$\lfloor n \rfloor = \lfloor m \rfloor$ resp. $\lfloor p \rfloor = \lfloor q \rfloor$ holds for the alpha-renaming clauses of Table 2. We shall write **Name** for the set of canonical names and once more assume that it is finite. Canonical capabilities are then capabilities using canonical names rather than names; we write **Cap** for those.

The analysis keeps track of the following information:

- An approximation of the contents of ambients of a given group:

  $$\mathcal{I} \subseteq \mathbf{Group} \times (\mathbf{Group} \cup \mathbf{Cap})$$

  So $u \in \mathcal{I}(\mu)$ (standing for $(\mu, u) \in \mathcal{I}$) means that μ may contain u. An ambient may contain other ambients as well as capabilities so the second component contain both possibilities. This component of the analysis is affected by the *movement capabilities*.

- An approximation to the relevant name bindings:

  $$\mathcal{R} \subseteq \mathbf{Name} \times \mathbf{Name}$$

  So $\nu' \in \mathcal{R}(\nu)$ (standing for $(\nu, \nu') \in \mathcal{R}$) means that ν may take on the value ν'. Here ν' will typically be the canonical name of the name being transmitted in the communication. This component of the analysis is affected by the *communication capabilities*.

The judgements of the analysis have the form

$$(\mathcal{I},\mathcal{R}) \models^\star P$$

and express that when P is enclosed within an ambient in group $\star \in \mathbf{Group}$ then $\mathcal{I}$ and $\mathcal{R}$ correctly capture the behaviour of P, i.e. if $P \rightarrow^k P'$ then also $(\mathcal{I},\mathcal{R}) \models^\star P'$.

The analysis is specified in two stages. First we make sure that $\mathcal{I}$ and $\mathcal{R}$ describe the initial process; this is done for processes in Table 4 and for capabilities in Table 5. The clauses of Table 4 simply amount to a straightforward structural traversal of the processes; whenever a name is introduced it must be reflected in the $\mathcal{R}$ component as expressed by the condition $\lfloor n \rfloor \in \mathcal{R}(\lfloor n \rfloor)$ and whenever an ambient is introduced it must be reflected in the $\mathcal{I}$ component as expressed by $\mu \in \mathcal{I}(\star)$. For capabilities we use the judgement $(\mathcal{I},\mathcal{R}) \models^\star M$ defined in Table 5 and explained below. The clauses for parallel processes and sums of processes are equal thereby witnessing the simplicity of the analysis; the same trend is followed in the analysis of recursion.

To understand the analysis of capabilities it is important to observe that the names introduced by $(n)P$ are constants whereas the names introduced in input capabilities (called p above) are variables that may be bound to other names (i.e. constants) as a result of communications. The clauses for processes already ensure that constants stand for themselves in $\mathcal{R}$; initially there will be no requirements on the bindings of the variables of input capabilities, they will be imposed when we study how to mimick the dynamics of the processes. The clauses of Table 5 merely demand that for each possible binding of the names occurring free in the capability (called n and m above), there is a record of the corresponding instantiated capability in the $\mathcal{I}$ component of the analysis.

$(\mathcal{I},\mathcal{R}) \models^\star \mathsf{enter}\ n$	iff $\forall \nu_n : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \Rightarrow \mathsf{enter}\ \nu_n \in \mathcal{I}(\star)$
$(\mathcal{I},\mathcal{R}) \models^\star \mathsf{accept}\ n$	iff $\forall \nu_n : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \Rightarrow \mathsf{accept}\ \nu_n \in \mathcal{I}(\star)$
$(\mathcal{I},\mathcal{R}) \models^\star \mathsf{exit}\ n$	iff $\forall \nu_n : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \Rightarrow \mathsf{exit}\ \nu_n \in \mathcal{I}(\star)$
$(\mathcal{I},\mathcal{R}) \models^\star \mathsf{expel}\ n$	iff $\forall \nu_n : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \Rightarrow \mathsf{expel}\ \nu_n \in \mathcal{I}(\star)$
$(\mathcal{I},\mathcal{R}) \models^\star \mathsf{merge+}\ n$	iff $\forall \nu_n : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \Rightarrow \mathsf{merge+}\ \nu_n \in \mathcal{I}(\star)$
$(\mathcal{I},\mathcal{R}) \models^\star \mathsf{merge-}\ n$	iff $\forall \nu_n : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \Rightarrow \mathsf{merge-}\ \nu_n \in \mathcal{I}(\star)$
$(\mathcal{I},\mathcal{R}) \models^\star n!\{m\}$	iff $\forall \nu_n, \nu_m : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \wedge \nu_m \in \mathcal{R}(\lfloor m \rfloor) \Rightarrow \nu_n!\{\nu_m\} \in \mathcal{I}(\star)$
$(\mathcal{I},\mathcal{R}) \models^\star n?\{p\}$	iff $\forall \nu_n : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \Rightarrow \nu_n?\{\lfloor p \rfloor\} \in \mathcal{I}(\star)$
$(\mathcal{I},\mathcal{R}) \models^\star n\_!\{m\}$	iff $\forall \nu_n, \nu_m : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \wedge \nu_m \in \mathcal{R}(\lfloor m \rfloor) \Rightarrow \nu_n\_!\{\nu_m\} \in \mathcal{I}(\star)$
$(\mathcal{I},\mathcal{R}) \models^\star n\hat{}?\{p\}$	iff $\forall \nu_n : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \Rightarrow \nu_n\hat{}?\{\lfloor p \rfloor\} \in \mathcal{I}(\star)$
$(\mathcal{I},\mathcal{R}) \models^\star n\hat{}!\{m\}$	iff $\forall \nu_n, \nu_m : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \wedge \nu_m \in \mathcal{R}(\lfloor m \rfloor) \Rightarrow \nu_n\hat{}!\{\nu_m\} \in \mathcal{I}(\star)$
$(\mathcal{I},\mathcal{R}) \models^\star n\_?\{p\}$	iff $\forall \nu_n : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \Rightarrow \nu_n\_?\{\lfloor p \rfloor\} \in \mathcal{I}(\star)$
$(\mathcal{I},\mathcal{R}) \models^\star n\#!\{m\}$	iff $\forall \nu_n, \nu_m : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \wedge \nu_m \in \mathcal{R}(\lfloor m \rfloor) \Rightarrow \nu_n\#!\{\nu_m\} \in \mathcal{I}(\star)$
$(\mathcal{I},\mathcal{R}) \models^\star n\#?\{p\}$	iff $\forall \nu_n : \nu_n \in \mathcal{R}(\lfloor n \rfloor) \Rightarrow \nu_n\#?\{\lfloor p \rfloor\} \in \mathcal{I}(\star)$

Table 5: Analysis of capabilities: $(\mathcal{I},\mathcal{R}) \models^\star M$.

Enter/accept:	$\forall \mu, \mu_1, \mu_2, \nu_n$:	$\mathsf{enter}\ \nu_n \in \mathcal{I}(\mu_1) \wedge \mu_1 \in \mathcal{I}(\mu) \wedge$ $\mathsf{accept}\ \nu_n \in \mathcal{I}(\mu_2) \wedge \mu_2 \in \mathcal{I}(\mu)$ $\Rightarrow \mu_1 \in \mathcal{I}(\mu_2)$
Exit/expel :	$\forall \mu, \mu_1, \mu_2, \nu_n$:	$\mathsf{exit}\ \nu_n \in \mathcal{I}(\mu_1) \wedge \mu_1 \in \mathcal{I}(\mu_2) \wedge$ $\mathsf{expel}\ \nu_n \in \mathcal{I}(\mu_2) \wedge \mu_2 \in \mathcal{I}(\mu)$ $\Rightarrow \mu_1 \in \mathcal{I}(\mu)$
Merge:	$\forall \mu, \mu_1, \mu_2, \nu_n$:	$\mathsf{merge+}\ \nu_n \in \mathcal{I}(\mu_1) \wedge \mu_1 \in \mathcal{I}(\mu) \wedge$ $\mathsf{merge-}\ \nu_n \in \mathcal{I}(\mu_2) \wedge \mu_2 \in \mathcal{I}(\mu)$ $\Rightarrow \forall \mu' : \mu' \in \mathcal{I}(\mu_2) \Rightarrow \mu' \in \mathcal{I}(\mu_1)$
To local:	$\forall \mu, \nu_m, \nu_p, \nu_n$:	$\nu_n!\{\nu_m\} \in \mathcal{I}(\mu) \wedge$ $\nu_n?\{\nu_p\} \in \mathcal{I}(\mu)$ $\Rightarrow \nu_m \in \mathcal{R}(\nu_p)$
To child:	$\forall \mu, \mu_c, \nu_m, \nu_p, \nu_n$:	$\nu_n\_!\{\nu_m\} \in \mathcal{I}(\mu) \wedge$ $\nu_n\hat{}?\{\nu_p\} \in \mathcal{I}(\mu_c) \wedge \mu_c \in \mathcal{I}(\mu)$ $\Rightarrow \nu_m \in \mathcal{R}(\nu_p)$
To parent:	$\forall \mu, \mu_c, \nu_m, \nu_p, \nu_n$:	$\nu_n\hat{}!\{\nu_m\} \in \mathcal{I}(\mu_c) \wedge \mu_c \in \mathcal{I}(\mu) \wedge$ $\nu_n\_?\{\nu_p\} \in \mathcal{I}(\mu)$ $\Rightarrow \nu_m \in \mathcal{R}(\nu_p)$
To sibling:	$\forall \mu, \mu_1, \mu_2, \nu_m, \nu_p, \nu_n$:	$\nu_n\#!\{\nu_m\} \in \mathcal{I}(\mu_1) \wedge \mu_1 \in \mathcal{I}(\mu) \wedge$ $\nu_n\#?\{\nu_p\} \in \mathcal{I}(\mu_2) \wedge \mu_2 \in \mathcal{I}(\mu)$ $\Rightarrow \nu_m \in \mathcal{R}(\nu_p)$

Table 6: Closure condition on $\mathcal{I}$ and $\mathcal{R}$.

Finally we make sure that $\mathcal{I}$ and $\mathcal{R}$ also take the dynamics of the process into account; this is formulated by the closure conditions in Table 6. The first three clauses take care of the movement capabilities and the last four of the communication capabilities. In each case the precondition expresses in terms of $\mathcal{I}$ the potential presence of a redex in the semantics and the conclusion then imposes the additional requirements on $\mathcal{I}$ and $\mathcal{R}$ necessary to mimick the semantics.

The semantic correctness of the analysis is expressed by:

Theorem. Assume $P \to Q$, $(\mathcal{I}, \mathcal{R}) \models^\star P$ and $\forall n \in \mathsf{fn}(P) : \lfloor n \rfloor \in \mathcal{R}(\lfloor n \rfloor)$. Then $(\mathcal{I}, \mathcal{R}) \models^\star Q$.

The proof is by induction on $P \to Q$ and uses the following standard lemma:

Lemma. If $P \equiv Q$ then $(\mathcal{I}, \mathcal{R}) \models^\star P$ if and only if $(\mathcal{I}, \mathcal{R}) \models^\star Q$.

The analysis is implemented using the Succinct Solver [19]. This solver works over finite (but not necessarily bounded) universes and accepts as input a static analysis specified as clauses in ALFP (Alternation Free Fixedpoint Logic) and it will then compute their least solution. Actually, the clauses of Tables 4, 5 and 6 are already written in ALFP so the implementation is straightforward.

The Succinct Solver is implemented in Standard ML and exploits a number of clever algorithms and data structures in order to obtain not only a good performance but also a formally predictable time complexity. Compared with other solvers, the Succinct Solver is optimised for handling sparse relations as we believe they frequently appear in context dependent static analysis. In the specification of the analysis above we have not been concerned with these issues at all and our practical experiments have not indicated a need for doing so; as an example when analysing the process to be presented in the next section, the solver will operate over a universe with just 89 atoms and it will construct an $\mathcal{I}$ relation with 75 elements and a $\mathcal{R}$ relation with 33 elements; the computation of these relations takes less than a second.

However, for more complex examples it may be worthwhile to rewrite the analysis to better exploit the representation of relations. As an illustration of what can be done consider for example the analysis of the capability $\mathsf{enter}\ n$ in Table 5: it will give rise to a pair $(\mathsf{enter}\ \nu_n, \star)$ in $\mathcal{I}$ for each possible value ν_n of $\lfloor n \rfloor$ in $\mathcal{R}$. An alternative specification would just include $(\mathsf{enter}\ \lfloor n \rfloor, \star)$ in $\mathcal{I}$ and then inspect $\mathcal{R}$ as part of checking for the presence of a redex in the closure condition of Table 6.

4 Example: Transcriptional Regulation by Positive Feedback

We shall now use BioAmbients to model the same example specified in [24] relying on a variant of the stochastic π-calculus [23]. The system, illustrated in Figure 1 and presented in Table 7, regulates gene expression by positive feedback. It includes two genes ($Gene_A$ and $Gene_{TF}$), their transcribed mRNAs (RNA_A and RNA_{TF}), the corresponding translated proteins ($Protein_A$ and $Protein_{TF}$) and the degradation of both RNA and protein molecules. The events are mediated by interaction with cellular machineries for DNA transcription (*Transcr*), RNA translation (*Transl*) and RNA and protein degradation (RNA_{deg} and $Protein_{deg}$). Each of these interactions involves different molecular motifs (channels *basal*, *utr*, *degm*, and *degp*).

After two sibling communications on the channel *basal* between *Transcr* and both $Gene_A$ and $Gene_{TF}$ and after the movement of ambients originated by the capabilities $\mathsf{expel}\ a/\mathsf{exit}\ a$ and $\mathsf{expel}\ c/\mathsf{exit}\ c$, the ambients RNA_A and RNA_{TF} are both at the top level.

Now the translation mechanism moves $Protein_A$ and $Protein_{TF}$ to the top level through two sibling communications on the channel *utr* between *Transl* and both RNA_A and RNA_{TF} followed by the movements generated by the capabilities $\mathsf{expel}\ b/\mathsf{exit}\ b$ and $\mathsf{expel}\ e/\mathsf{exit}\ e$.

In the configuration reached $Protein_A$ binds $Protein_{TF}$ by accepting $Active_{TF}$ inside itself using the $\mathsf{accept}\ tf/\mathsf{enter}\ tf$ capabilities. Then $Protein_{TF}$ becomes active by expelling the ambient $Bound_{TF}$ with the capabilities $\mathsf{expel}\ atf/\mathsf{exit}\ atf$. For $Bound_{TF}$ there are now three alternatives:

1. Using a sibling communication on the channel *bb2* it first synchronises with the *Kinase* of $Protein_A$, then it synchronises with the parent $Protein_A$ on the channel *bb1*, and eventually it expels the ambient $Active_{TF}$ with the capabilities $\mathsf{expel}\ f/\mathsf{exit}\ f$. The ambient $Active_{TF}$ is now expelled from $Protein_A$ by the capabilities $\mathsf{expel}\ g/\mathsf{exit}\ g$. Then $Active_{TF}$ can interact either with the transcription factor *Transcr* by a sibling communication on the channel *ptail* or with the degradation factor $Protein_{deg}$ by a sibling communication on the channel *degp*. Note that $Bound_{TF}$ can be dissolved

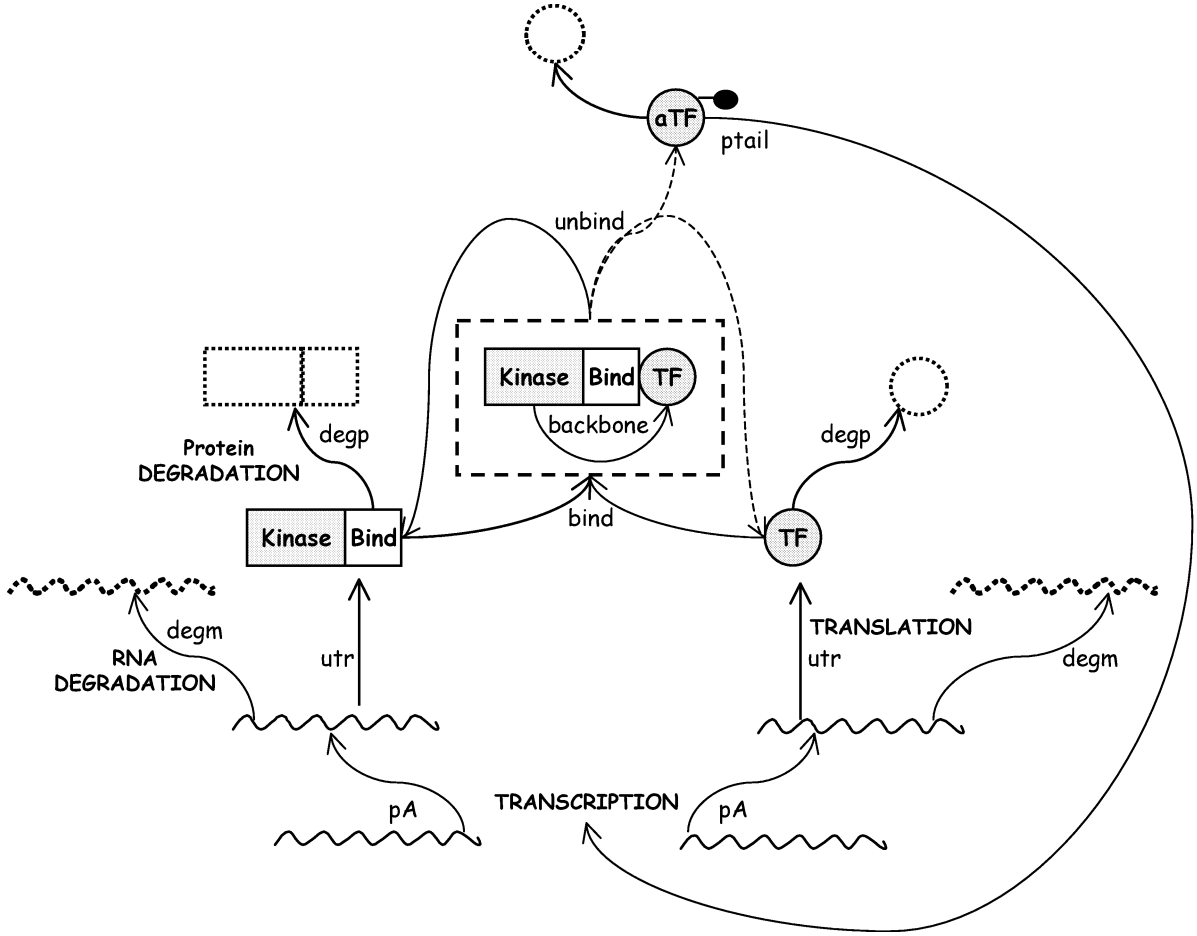

Figure 1: Graphical presentation of Transcriptional Regulation by Positive Feedback [24].

at any time if $Protein_A$ starts a degradation through a sibling interaction along the channel *degp* with $Protein_{deg}$.

2. It can be dissolved after a communication from the parent $Protein_A$ on the channel *bb3* because $Protein_A$ has started a degradation step with a sibling communication on the channel *degp*.

3. It can enter again $Protein_{TF}$ after a communication from the parent along the channel *bb1*.

The specification of the system is reported in Table 7. Note that to avoid a heavy use of parentheses, we write the summation as well as the parallel composition operator immediately under the beginning of the first summand. To aid the analysis we have alpha-renamed the bound variables apart.

The result of analysing the system is displaied in Table 8. Most of the entries of the $\mathcal{I}$ component account for the syntactic structure of the process as displayed in Table 7; the dynamics of the system causes the underlined pairs to be added. These entries clearly confirm the behaviour of the system as described above:

- The pairs $(Gene_A, Protein_A)$, $(\star, Protein_A)$, $(\star, RNA_A)$ and $(Gene_{TF}, Protein_{TF})$, $(\star, Protein_{TF})$, $(\star, RNA_{TF})$ reflect the movement of $Protein_A$, $Protein_{TF}$, RNA_A and RNA_{TF} to the top level.
- The pair $(Protein_A, Protein_{TF})$ witnesses that $Protein_{TF}$ enters $Protein_A$ and the activation of $Bound_{TF}$ inside $Protein_A$ is then reflected by the presence of the pair $(Protein_A, Bound_{TF})$.
- The expelling of $Active_{TF}$ from $Protein_A$ is reflected by the presence of the pair $(\star, Active_{TF})$.

$$
\begin{array}{l}
(a)(b)(c)(d)(e)(f)(g)(bb1)(bb2)(bb3)(basal)(pa)(utr)(degm)(degp)(tf)(atf)(ptail) \\
\quad [\quad \mathsf{rec}\, X_1.\ \ (basal\#?\{x_2\}.\ \mathsf{expel}\ a.\ X_1 \\
\qquad\qquad +pa\#?\{x_1\}.\ \mathsf{expel}\ a.\ X_1) \\
\qquad \mid\mid [\quad \mathsf{rec}\, X_2.\ \mathsf{exit}\ a.\ \ (utr\#?\{x_4\}.\ \mathsf{expel}\ b.\ X_2 \\
\qquad\qquad\qquad\qquad +degm\#?\{x_3\}.\ 0) \\
\qquad\qquad \mid\mid [\ \mathsf{exit}\ b.\ \ \mathsf{rec}\, X_3.\ \mathsf{accept}\ tf.\ \ (bb1\_!\{d\}.\ \ (\mathsf{expel}\ g.\ X_3 \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad +X_3) \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad +degp\#!\{d\}.\ bb3\_!\{d\}.\ bb3\_!\{d\}.\ 0 \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad +degp\#!\{d\}.\ bb3\_!\{d\}.\ 0) \\
\qquad\qquad\qquad \mid\mid [\ \mathsf{rec}\, X_4.\ \ (bb2\#!\{d\}.\ X_4 \\
\qquad\qquad\qquad\qquad\qquad +bb3\tilde{}?\{x_5\}.\ 0)\,]^{Kinase}\,]^{Protein_A}\,]^{RNA_A}\,]^{Gene_A} \\
\\
\quad \mid\mid [\quad \mathsf{rec}\, X_5.\ \ (basal\#?\{y_2\}.\ \mathsf{expel}\ c.\ X_5 \\
\qquad\qquad +pa\#?\{y_1\}.\ \mathsf{expel}\ c.\ X_5) \\
\qquad\quad \mid\mid [\quad \mathsf{rec}\, X_6.\ \mathsf{exit}\ c.\ \ (utr\#?\{y_4\}.\ \mathsf{expel}\ e.\ X_6 \\
\qquad\qquad\qquad\qquad +degm\#?\{y_3\}.\ 0) \\
\qquad\qquad\quad \mid\mid [\ \mathsf{exit}\ e.\ \mathsf{enter}\ tf.\ \mathsf{expel}\ atf.\ \mathsf{accept}\ atf.\ 0 \\
\qquad\qquad\qquad \mid\mid [\ \mathsf{exit}\ atf.\ \ (bb1\tilde{}?\{y_9\}.\ \mathsf{enter}\ atf.\ 0 \\
\qquad\qquad\qquad\qquad\qquad +bb3\tilde{}?\{y_8\}.\ 0 \\
\qquad\qquad\qquad\qquad\qquad +bb2\#?\{y_7\}.\ \ (bb1\tilde{}?\{y_6\}.\ \mathsf{expel}\ f.\ 0 \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad +bb3\tilde{}?\{y_5\}.\ 0)) \\
\qquad\qquad\qquad\quad \mid\mid [\mathsf{exit}\ f.\ \mathsf{exit}\ g.\ \ \mathsf{rec}\, X_7.\ \ (ptail\#!\{d\}.\ X_7 \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad +degp\#?\{y_{10}\}.\ 0)\,]^{Active_{TF}}\,]^{Bound_{TF}}\,]^{Protein_{TF}}\,]^{RNA_{TF}}\,]^{Gene_{TF}} \\
\\
\quad \mid\mid [\ \mathsf{rec}\, X_8.\ \ basal\#!\{d\}.\ X_8 \\
\qquad\qquad +ptail\#?\{z_1\}.\ pa\#!\{d\}.\ X_8\,]^{Transcr} \\
\\
\quad \mid\mid [\ \mathsf{rec}\, X_9.\ utr\#!\{d\}.\ X_9\,]^{Transl} \\
\\
\quad \mid\mid [\ \mathsf{rec}\, X_{10}.\ degm\#!\{d\}.\ X_{10}\,]^{RNA_{deg}} \\
\\
\quad \mid\mid [\ \mathsf{rec}\, X_{11}.\ degp\#!\{d\}.\ X_{11}\,]^{Protein_{deg}}
\end{array}
$$

Table 7: BioAmbient representation of Transcriptional Regulation by Positive Feedback.

As the analysis specifies an over-approximation to the actual contents of the ambients it is actually more interesting to observe the information that is *not* included in $\mathcal{I}$ as this confirms what is definitely not happening. As an example we can see that the ambient *Kinase* does not move at all — only the pair $(Protein_A, Kinase)$ is present in $\mathcal{I}$ — and hence even though there may be several copies of $Protein_A$ in the system they are guaranteed not to get their *Kinase* components mixed up.

The $\mathcal{R}$ component approximates the bindings of the names and since all communications in the example amount to nothing but synchronisation we observe that all variables may end up being bound to the dummy name d. Also we see that all variables may eventually get bound to a value.

5 Conclusion and Further Work

The paper presented a new control flow analysis for BioAmbients, a calculus based on Mobile Ambients and specifically tuned to model biological systems. Our proposal is the first attempt to adopt static analysis techniques for analysing molecular interactions. We established the feasibility of the approach on a case study taken from [24], where a gene regulation by positive feedback is modelled in π-calculus.

The analysis introduced here is a very simple one because it is both context insensitive and flow insensitive. Nevertheless, it has proved very useful for debugging the preliminary versions of our specification. In fact, the basic mechanisms of ambient calculi and π-like calculi are quite different in modelling dimers. In the BioAmbients we can simply decide that one component enters another in the same ambient or that

$\mathcal{I}$: $(Protein_{deg}, degp\#!\{d\})$,

$(RNA_{deg}, degm\#!\{d\})$,

$(Transl, utr\#!\{d\})$,

$(Transcr, pa\#!\{d\}), (Transcr, ptail\#?\{z_1\}), (Transcr, basal\#!\{d\})$,

$(Active_{TF}, degp\#?\{y_{10}\}), (Active_{TF}, ptail\#!\{d\}), (Active_{TF}, \mathsf{exit}\ g), (Active_{TF}, \mathsf{exit}\ f)$,

$(Bound_{TF}, Active_{TF}), (Bound_{TF}, bb3\tilde{}?\{y_5\}), (Bound_{TF}, \mathsf{expel}\ f), (Bound_{TF}, bb1\tilde{}?\{y_6\})$,
$(Bound_{TF}, bb2\#?\{y_7\}), (Bound_{TF}, bb3\tilde{}?\{y_8\}), (Bound_{TF}, \mathsf{expel}\ atf), (Bound_{TF}, bb1\tilde{}?\{y_9\})$,
$(Bound_{TF}, \mathsf{exit}\ atf)$,

$\underline{(Protein_{TF}, Active_{TF})}, (Protein_{TF}, Bound_{TF}), (Protein_{TF}, \mathsf{accept}\ atf)$,
$\overline{(Protein_{TF}, \mathsf{expel}\ atf)}, (Protein_{TF}, \mathsf{expel}\ tf), (Protein_{TF}, \mathsf{exit}\ e)$,

$\underline{(RNA_{TF}, Active_{TF})}, \underline{(RNA_{TF}, Bound_{TF})}, (RNA_{TF}, Protein_{TF}), (RNA_{TF}, degm\#?\{y_3\})$,
$\overline{(RNA_{TF}, \mathsf{expel}\ e)}, (RNA_{TF}, utr\#?\{y_4\}), (RNA_{TF}, \mathsf{exit}\ c)$,

$\underline{(Gene_{TF}, Active_{TF})}, \underline{(Gene_{TF}, Bound_{TF})}, \underline{(Gene_{TF}, Protein_{TF})}, (Gene_{TF}, RNA_{TF})$,
$\overline{(Gene_{TF}, pa\#?\{y_1\})}, (Gene_{TF}, \mathsf{expel}\ c), \overline{(Gene_{TF}, basal\#?\{y_2\})}$,

$(Kinase, bb3\tilde{}?\{x_5\}), (Kinase, bb2\#!\{d\})$,

$\underline{(Protein_A, Active_{TF})}, \underline{(Protein_A, Bound_{TF})}, \underline{(Protein_A, Protein_{TF})}, (Protein_A, Kinase)$,
$\overline{(Protein_A, bb3\_!\{d\})}, \overline{(Protein_A, degp\#!\{d\})}, \overline{(Protein_A, bb3\_!\{d\})}, (Protein_A, bb3\_!\{d\})$,
$(Protein_A, degp\#!\{d\}), (Protein_A, \mathsf{expel}\ g), (Protein_A, bb1\_!\{d\}), (Protein_A, \mathsf{accept}\ tf)$,
$(Protein_A, \mathsf{exit}\ b)$,

$\underline{(RNA_A, Active_{TF})}, (RNA_A, Protein_A), (RNA_A, degm\#?\{x_3\}), (RNA_A, \mathsf{expel}\ b)$,
$\overline{(RNA_A, utr\#?\{x_4\})}, (RNA_A, \mathsf{exit}\ a)$,

$\underline{(Gene_A, Active_{TF})}, \underline{(Gene_A, Protein_A)}, (Gene_A, RNA_A), (Gene_A, pa\#?\{x_1\})$,
$\overline{(Gene_A, \mathsf{expel}\ a)}, \overline{(Gene_A, basal\#?\{x_2\})}$,

$\underline{(\star, Protein_A)}, \underline{(\star, RNA_A)}, \underline{(\star, Active_{TF})}, \underline{(\star, Bound_{TF})}, \underline{(\star, Protein_{TF})}, \underline{(\star, RNA_{TF})}$,
$\overline{(\star, Protein_{deg})}, \overline{(\star, RNA_{deg})}, \overline{(\star, Transl)}, \overline{(\star, Transcr)}, \overline{(\star, Gene_{TF})}, \overline{(\star, Gene_A)}$

$\mathcal{R}$: $(x_1, d), (x_2, d), (x_3, d), (x_4, d), (x_5, d)$,
$(y_1, d), (y_2, d), (y_3, d), (y_4, d), (y_5, d), (y_6, d), (y_7, d), (y_8, d), (y_9, d), (y_{10}, d)$,
(z_1, d),

$(ptail, ptail), (atf, atf), (tf, tf), (degp, degp), (degm, degm), (utr, utr), (pa, pa), (basal, basal)$,
$(bb3, bb3), (bb2, bb2), (bb1, bb1), (g, g), (f, f), (e, e), (d, d), (c, c), (b, b), (a, a)$

Table 8: Analysis result.

two ambients merge to generate a new single ambient including the content of both the merging ones. In the π-calculus we model this situation by letting the constituent of the dimer share a new private channel through a scope extrusion and subsequent closing of the enlarged scope. This difference prevents each π-calculus process in the specification in [24] from being matched by a corresponding ambient, and hence the overall behaviour of the two systems is not easily checked to be equivalent. We used our analysis to check that the interacting entities are the same in both specifications and that the flow of information represented by new bindings is the same in both specification. We iterated the process of specifying the system and analysing it before reaching a BioAmbient specification with the same behaviour as the π-calculus specification. This practical experiment shows how important static analysis is in the modelling phase of biological systems, when we have to write a specification that matches the experimental knowledge available from biological data.

A further development along the line described above is the construction of automatic extractors of process algebra specifications from available databases and subsequent analysis of the specification to validate the knowledge encoded in the databases with the available experimental knowledge from biological literature. Actually, a major problem in modelling biological systems is the selection of parameters that can vary a lot from one publication to another and even from one database to another for the same experiment. To be more accurate in this direction, we are working to extend the semantics as well as the analysis to take stochastic information into account. A suitable approach could be to rely on the enhanced operational semantics [8] where stochastic information is derived by a relabelling function and it is a parameter of the

semantic model [21]. This separation of concerns should allow an easy extension of the analysis presented here and it should also allow to run the analysis solver on the same specification many time with different quantitative parameters thus comparing different experiments.

Acknowledgement. This research has been funded in part by the DEGAS project (number IST-2001-32072) funded by the European Union and by the LoST project (number 21-02-0507) funded by the Danish Natural Science Research Council.

References

[1] Spad signaling pathway database. 2000.

[2] G. D. Bader, I. Donaldson, C. Wolting, B. F. Ouellette, T. Pawson, and C. W. Hogue. Bind-the biomolecular interaction network database. *Nucleic Acids Research*, 29(1):242–245, 2001.

[3] C. Bodei, P. Degano, F. Nielson, and H. Riis Nielson. Static analysis for the π-calculus with applications to security. *Information and Computation*, 168:68–92, 2001.

[4] M. Bugliesi, G. Castagna, and S. Crafa. Boxed Ambients. In *Theoretical Aspects in Computer Science (TACS 2001)*, volume 2215 of *Lecture Notes in Computer Science*, pages 37–63. Springer, 2001.

[5] L. Cardelli. Bioware languages. In *Computer Systems - Papers for Roger Needham*. 2003.

[6] L. Cardelli and A. D. Gordon. Mobile Ambients. In *Foundations of Software Science and Computation Structures (FoSSaCS 1998)*, volume 1378 of *Lecture Notes in Computer Science*, pages 140–155. Springer, 1998.

[7] V. Danos and C. Laneve. Core formal molecular biology. In *European Symposium on Programming (ESOP03), to appear*, 2003.

[8] P. Degano and C. Priami. Enhanced operational semantics: A tool for describing and analysing concurrent systems. *ACM Computing Surveys*, 33,2:135–176, 2001.

[9] K. Eilbeck, A. Brass, N. Paton, and C. Hodgman. Interact: an object oriented protein-protein interaction database. In *Intelligent Systems for Molecular Biology*, volume 7, pages 87–94, Palo Alto, 1999. AAAI Press.

[10] W. Fontana and L. W. Buss. The arrival of the fittest: Toward a theory of biological organization. *Bull. Math. Biol.*, 56:1–64, 1994.

[11] T. Igarashi and T. Kaminuma. Development of a cell signalling networks database. In R. B. Altman, A. K. Dunker, L. Hunter, and T. E. Klein, editors, *Proccedings of the Pacific Symposium of Biocomputing '97*, pages 187–197, Singapore, 1997. World Scientific Press.

[12] P. D. Karp, M. Krummenacker, S. Paley, and J. Wagg. Integrated pathway/genome databases and their role in drug discovery. *Trends in Biotechnology*, 17(7):275–281, 1999.

[13] F. A. Kolpakov, E. A. Ananko, G. B. Kolesov, and N. A. Kolchanov. Genenet: a gene network database and its automated visualization. *Bioinformatics*, 14(8):529–537, 1998.

[14] F. Levi and D. Sangiorgi. Controlling interference in ambients. In *Proceedings of the 27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL 2000)*, pages 352–364. ACM Press, 2000.

[15] R. Milner. *Communicating and Mobile Systems: The pi-Calculus*. Cambridge University Press, 1999.

[16] M. Nagasaki, S. Onami, S. Miyano, and Kitano H. Bio-calculus: Its concept and molecular interaction. *Genome Informatics*, 10:133–143, 1999.

[17] F. Nielson, H. Riis Nielson, and C. Hankin. *Principles of Program Analysis*. 1999.

[18] F. Nielson, H. Riis Nielson, and R. R. Hansen. Validating firewalls using flow logics. *Theoretical Computer Science*, 283(2):381–418, 2002.

[19] F. Nielson, H. Riis Nielson, and H. Seidl. A succinct solver for ALFP. *Nordic Journal of Computing*, 9:335–372, 2002.

[20] H. Riis Nielson, F. Nielson, and M. Buchholtz. Security for mobility. Technical Report WP6-IMM-I01-Int-001, DEGAS, 2002.

[21] C. Nottegar, C. Priami, and P. Degano. Performance evaluation of mobile processes via abstract machines. *IEEE Transactions on Software Engineering*, 27(10), 2001.

[22] H. Ogata, S. Goto, K. Sato, W. Fujibuchi, H. Bono, and M. Kanehisa. Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 27(1):29–34, 2000.

[23] C. Priami. Language-based performance prediction for distributed and mobile systems. *INFCTRL: Information and Computation (formerly Information and Control)*, 175, 2002.

[24] C. Priami, A. Regev, W. Silverman, and E. Shapiro. Application of a stochastic passing-name calculus to representation and simulation of molecular processes. *Information Processing Letters*, 80:25–31, 2001.

[25] A. Regev. *Computational system biology: A calculus for biomolecular knoledge*. PhD thesis, Tel Aviv University, 2003.

[26] A. Regev, E. M. Panina, W. Silverman, L. Cardelli, and E. Shapiro. BioAmbients: An abstraction for biological compartments. 2003. Manuscript available from http://www.luca.demon.co.uk/.

[27] A. Regev and E. Shapiro. Cells as computations. *Nature*, 419:343, 2002.

[28] A. Regev, W. Silverman, and E. Shapiro. Representation and simulation of biochemical processes using the π-calculus process algebra. In *Pacific Symposium of Biocomputing (PSB2001)*, pages 459–470, 2001.

[29] C. Sanchez, C. Lachaize, F. Janody, B. Bellon, L. Roder, J. Euzenat, F. Rechenmann F, and B. Jacq. Grasping at molecular interactions and genetic networks in drosophila melanogaster using flynets, an internet database. *Nucleic Acids Research*, 27(1):89–94, 1999.

[30] E. Selkov, Y. Grechkin, N. Mikhailova, and E. Selkov. Mpw: the metabolic pathways database. *Nucleic Acids Research*, 26(1):43–45, 1998.

[31] J. van Helden, A. Naim, R. Mancuso, M. Eldridge, L. Wernisch, D. Gilbert D, and S. J. Wodak. Representing and analysing molecular and cellular function using the computer. *Biological Chemistry*, 381(9–10):921–935, 2000.

[32] E. Wingender, X. Chen, E. Fricke, R. Geffers, R. Hehl, I. Liebich, M. Krull M, V. Matys, H. Michael, R. Ohnhauser, M. Pruss, F. Schacherer, S. Thiele, and S. Urbach. The transfac system on gene expression regulation. *Nucleic Acids Research*, 29(1):281–283, 2001.

[33] I. Xenarios, E. Fernandez E, L. Salwinski, X. J. Duan, M. J. Thompson, E. M. Marcotte, and D. Eisenberg. Dip: the database of interacting proteins: 2001 update. *Nucleic Acids Research*, 29(1):239–241, 2001.