Bisimulation Congruences in the Calculus of Looping Sequences

Roberto Barbuti, Andrea Maggiolo-Schettini,
Paolo Milazzo, and Angelo Troina

Dipartimento di Informatica, Università di Pisa
Largo B. Pontecorvo 3, 56127 - Pisa, Italy
{barbuti,maggiolo,milazzo,troina}@di.unipi.it

Abstract. The Calculus of Looping Sequences (CLS) is a calculus suitable to describe biological systems and their evolution. CLS terms are constructed by starting from basic constituents and composing them by means of operators of concatenation, looping, containment and parallel composition. CLS terms can be transformed by applying rewrite rules. We give a labeled transition semantics for CLS by using, as labels, contexts in which rules can be applied. We define bisimulation relations that are congruences with respect to the operators on terms, and we show an application of CLS to the modeling of a biological system and we use bisimulations to reason about properties of the described system.

1 Introduction

In the last few years many formalisms originally developed by computer scientists to model systems of interacting components have been applied to Biology. Among these, there are Petri Nets [9], Hybrid Systems [1], and the π-calculus [6, 12]. Moreover, some new formalisms have been proposed to describe biomolecular and membrane interactions [2, 4, 5, 7, 10, 11]. The formal modeling of biological systems allows the development of simulators and the verification of properties of the described systems.

The π–calculus and new calculi based on it [10, 11] have been particularly successful in the description of biological systems. Interactions of biological components are modeled as communications on channels whose names can be passed. Sharing names of private channels allows describing biological compartments, such as membranes. Calculi such as those proposed in [4, 5, 7] give an abstract description of systems and offer special biologically motivated operators.

In [3] we have presented a new calculus, called Calculus of Looping Sequences (CLS for short), for describing biological systems and their evolution, and we have shown how to use it for modeling interactions among bacteria and bacteriophage viruses as well as bacteria sporulation. In this paper we focus on semantic aspects of the formalism, in particular on bisimulation relations. We define a simplified variant of CLS (that we still call CLS in the following), we study bisimulations for it, and we apply such relations on the CLS model of a real example of gene regulation.

The calculus we propose is more general than those in [4, 5, 7], which could be encoded into CLS (see [3] for an example), and with respect to the π–calculi and calculi based on the π–calculus [10, 11], which are more expressive, it has the advantage of allowing the definition of bisimulations that are congruences.

The terms of our calculus are constructed by starting from basic constituent elements and composing them by means of operators of sequencing, looping, containment and parallel composition. Sequencing can be used to describe biological elements such as DNA fragments and proteins. DNA fragments can be modeled as sequences of nucleotides or as sequences of genes; proteins can be modeled as sequences of amino acids or as sequences of interaction sites. Looping allows tying up the ends of a sequence, thus creating a circular sequence of the constituent elements. We assume that the elements of a circular sequence can rotate, and this motivates the terminology of looping sequence. A looping sequence can represent a membrane. This description of membranes is finer than the one given in specialized membrane calculi (see e.g. [4, 11]) as it allows representing interaction of membrane constituent elements. The containment operator can be used to represent that an element is inside the membrane, and parallel composition expresses juxtaposition of elements.

A structural congruence relation allows considering as equivalent terms that are intended to represent the same biological system. The evolution of a system is described by a set of rewrite rules to be applied to terms. The definition of the rewrite rules depends on the system and the evolution one wants to represent.

Bisimilarity is widely accepted as the finest extensional behavioural equivalence one may want to impose on systems. It may be used to verify a property of a system by assessing the bisimilarity of the considered system with a system one knows to enjoy that property. The notion of congruence is very important for a compositional account of behavioural equivalence. This is true, in particular, for complex systems such as biological ones.

To define bisimilarity of systems, these must have semantics based on labeled transition relations capturing potential external interactions between systems and their environment. A labeled transition semantics for CLS is derived from rewrite rules by using as labels contexts in which rules can be applied, in the style of Sewell [13] and Leifer and Milner [8]. We define bisimulation relations and we show them to be congruences with respect to the operators on terms.

The main difference between the definition of CLS we give in this paper with respect to the one in [3], is the presence of some constraints on the syntax of terms which simplifies the definition of the labeled transition relation for the calculus. We model an example of gene regulation, namely the regulation of the lactose operon in E. coli, to show that the new variant of the calculus, though simple, is expressive enough to model real systems. We use bisimulations to obtain an equivalent simplified model and to verify a property of the described system.

2 Calculus of Looping Sequences

In this section we introduce the Calculus of Looping Sequences (CLS).

Definition 1 (Terms). Terms T, looping sequences S_L, elementary sequences S, *and* elementary constituents E *of CLS are given by the following grammar:*

$$\begin{array}{rcl} T & ::= & S \quad \big| \quad S_L \quad \big| \quad T \mid T \quad \big| \quad S_L \rfloor T \\ S_L & ::= & \big(S\big)^L \\ S & ::= & E \quad \big| \quad \epsilon \quad \big| \quad S \cdot S \\ E & ::= & a \quad \big| \quad b \quad \big| \quad c \quad \big| \quad \ldots \end{array}$$

We denote with $\mathcal{E}$ the set of elementary constituents $a, b, c, \ldots$.

An elementary sequence S may be either an element in $\mathcal{E}$ or the empty sequence ϵ or a concatenation of elementary sequences. An example of elementary sequence is $a \cdot b \cdot c$. We denote with $\mathcal{S}$ the set of elementary sequences.

A looping sequence S_L is obtained by applying the looping operator $\big(\_\big)^L$ to an elementary sequence S. A term T may be either an elementary sequence S, or a looping sequence S_L, or the combination of a looping sequence and a term by means of the containment operator $\rfloor$, or the combination of two terms by means of the parallel composition operator $\mid$.

A looping sequence $\big(S\big)^L$ is a closed circular sequence of the elements constituting the elementary sequence S. Term $\big(S\big)^L \rfloor T$ represents the containment of term T in the looping sequence $\big(S\big)^L$. The set of all terms is denoted by $\mathcal{T}$.

Brackets can be used to indicate the order of application of the operators in a term. We assume that the $\rfloor$ operator has precedence over the $\mid$ operator, therefore $S \rfloor T_1 \mid T_2$ stands for $(S \rfloor T_1) \mid T_2$. Moreover, from the definition of CLS terms, the $\rfloor$ operator is right–associative, therefore $S_1 \rfloor S_2 \rfloor T$ denotes $S_1 \rfloor (S_2 \rfloor T)$. In Fig. 1 we show some examples of CLS terms and their visual representation.

Definition 2 (Structural Congruence). *The structural congruence relations* $\equiv_S$ *and* $\equiv_T$ *are the least congruence relations on elementary sequences and on terms, respectively, satisfying the following axioms:*

$$S_1 \cdot (S_2 \cdot S_3) \equiv_S (S_1 \cdot S_2) \cdot S_3 \qquad S \cdot \epsilon \equiv_S \epsilon \cdot S \equiv_S S$$

$$T_1 \mid T_2 \equiv_T T_2 \mid T_1 \qquad T_1 \mid (T_2 \mid T_3) \equiv_T (T_1 \mid T_2) \mid T_3 \qquad \epsilon \equiv_T \big(\epsilon\big)^L$$

$$T \mid \epsilon \equiv_T T \qquad \big(S\big)^L \rfloor \epsilon \equiv_T \big(S\big)^L \qquad \big(S_1 \cdot S_2\big)^L \rfloor T \equiv_T \big(S_2 \cdot S_1\big)^L \rfloor T$$

Axioms of the structural congruence state the associativity of $\cdot$ and $\mid$, the commutativity of the latter and the neutral role of ϵ and $\big(\epsilon\big)^L$. We remark that $\big(\epsilon\big)^L \rfloor T \not\equiv T$. Moreover, axiom $\big(S_1 \cdot S_2\big)^L \rfloor T \equiv_T \big(S_2 \cdot S_1\big)^L \rfloor T$ says that elementary sequences in a looping can rotate.

Note that $\equiv_S$ can be lifted to $\equiv_T$, in the sense that if $S \equiv_S S'$ then $S \equiv_T S'$. Moreover, note that the first operand of the $\rfloor$ operator is not a general term,

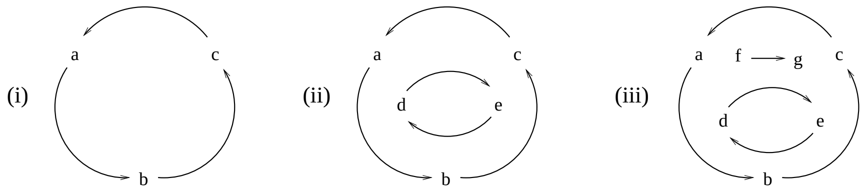

Fig. 1. (i) represents $\big(a \cdot b \cdot c\big)^L$; (ii) represents $\big(a \cdot b \cdot c\big)^L \rfloor \big(d \cdot e\big)^L$; (iii) represents $\big(a \cdot b \cdot c\big)^L \rfloor (\big(d \cdot e\big)^L \mid f \cdot g)$.

but a looping sequence $\big(S\big)^L$, which is an element of S_L, hence $\equiv_T$ cannot be applied to it, but $\equiv_S$ can be applied to the sequence S. In the following, for simplicity, we will use $\equiv$ in place of $\equiv_T$.

Now, we define rewrite rules, which can be used to describe the transformation of terms by giving a transition relation as the semantics of rule applications. Let us consider a set of term variables TV ranged over by $X, Y, Z, \ldots$, a set of elementary sequence variables SV ranged over by $\widetilde{x}, \widetilde{y}, \widetilde{z}, \ldots$, and a set of element variables $\mathcal{X}$ ranged over by $x, y, z, \ldots$. All these sets are possibly infinite and pairwise disjoint. We denote by $\mathcal{V}$ the set of all variables, $\mathcal{V} = TV \cup SV \cup \mathcal{X}$.

An *instantiation* is a function $\sigma : \mathcal{V} \rightarrow \mathcal{T}$; let Σ be the set of all the possible instantiations. With $\mathcal{T}_{\mathcal{V}}$ we denote the set of CLS terms which may also contain variables in $\mathcal{V}$ and, given $T \in \mathcal{T}_{\mathcal{V}}$, with $T\sigma$ we denote the term obtained by replacing each occurrence of each variable $X_{\mathcal{V}} \in \mathcal{V}$ appearing in T with the corresponding term $\sigma(X_{\mathcal{V}})$. An instantiation σ must respect the type of variables, thus for $X \in TV$, $\widetilde{x} \in SV$ and $x \in \mathcal{X}$ we have $\sigma(X) \in \mathcal{T}$, $\sigma(\widetilde{x}) \in \mathcal{S}$ and $\sigma(x) \in \mathcal{E}$, respectively.

Definition 3 (Rewrite Rule). *A rewrite rule is a pair of terms (T, T'), denoted with $T \mapsto T'$, where $T, T' \in \mathcal{T}_{\mathcal{V}}$, $T \not\equiv \epsilon$ and such that $Var(T') \subseteq Var(T)$. We denote with $\Re$ the infinite set of all the possible rewrite rules.*

A rewrite rule (T, T') states that a ground term $T\sigma$, obtained by instantiating variables in T by some instantiation function σ, can be transformed into the ground term $T'\sigma$. The rewrite rules must be applied to terms only if they occur in a legal context. Contexts are defined as follows.

Definition 4 (Contexts). Contexts $\mathcal{C}$ *are given by the following grammar:*

$$\mathcal{C} ::= \square \quad \big| \quad \mathcal{C} \mid T \quad \big| \quad T \mid \mathcal{C} \quad \big| \quad \big(S\big)^L \rfloor \mathcal{C}$$

where $T \in \mathcal{T}$ and $S \in \mathcal{S}$. Context $\square$ is called the empty context*.*

By definition, every context contains a single $\square$.

Let us assume $C, C' \in \mathcal{C}$. With $C[T]$ we denote the term obtained by replacing $\square$ with T in C; with $C[C']$ we denote the context composition, whose result is the context obtained by replacing $\square$ with C' in C.

Definition 5 (Reduction Semantics). *Given a set of rewrite rules* $\mathcal{R} \subseteq \Re$, *the* reduction semantics *of CLS is the least relation satisfying the following inference rule:*

$$\frac{T \mapsto T' \in \mathcal{R} \qquad T\sigma \not\equiv \epsilon \qquad \sigma \in \Sigma \qquad C \in \mathcal{C}}{C[T\sigma] \to C[T'\sigma]}$$

Definition 6 (Parallel contexts). *Parallel contexts* $\mathcal{C}_P$ *are a subset of contexts given by the following grammar, where* $T \in \mathcal{T}$*:*

$$\mathcal{C}_P ::= \square \quad \big| \quad \mathcal{C}_P \mid T \quad \big| \quad T \mid \mathcal{C}_P$$

Contexts are used for defining a labeled semantics of CLS terms.

Definition 7 (Labeled Semantics). *Given a set of rewrite rules* $\mathcal{R} \subseteq \Re$, *the* labeled semantics *of CLS is the labeled transition system given by the following inference rules:*

$$\text{(rule\_appl)}\ \frac{T \mapsto T' \in \mathcal{R} \quad C[T''] \equiv T\sigma \quad T'' \not\equiv \epsilon \quad \sigma \in \Sigma \quad C \in \mathcal{C}}{T'' \xrightarrow{C} T'\sigma}$$

$$\text{(cont)}\ \frac{T \xrightarrow{\square} T'}{T'' \rfloor T \xrightarrow{\square} T'' \rfloor T'} \qquad \text{(par)}\ \frac{T \xrightarrow{C} T' \quad C \in \mathcal{C}_P}{T \mid T'' \xrightarrow{C} T' \mid T''}$$

where the dual version of the (par) *rule is omitted.*

The labeled semantics is similar to the one in [13] for ground term rewriting. A transition $T \xrightarrow{C} T'$ indicates that the application of the context C to T creates an instance of the left part of a rewrite rule, with target instance T'. Intuitively, the transition makes observable the context C, which, when filled with the term T, can be reduced to T', namely $C[T] \mapsto T'$ is an instance of a rewrite rule. Note that, since in rule (rule_appl) $T'' \not\equiv \epsilon$, the context C used as label cannot provide completely the left part of the rule. Differently with respect to [13], we allow to observe the context in the reduction of a subterm of a parallel composition. For example, if $C[T] \mapsto T'$ is an instance of a rewrite rule, then we have that $T|T" \xrightarrow{C} T'|T"$ (rule (par)).

The following proposition states that the labeled semantics is equivalent to the reduction semantics when the context is empty. The proof is immediate.

Proposition 1. $T \to T' \iff T \xrightarrow{\square} T'$.

Lemma 1 gives a property of parallel contexts, and Lemma 2 gives the labeled semantics with respect to context composition.

Lemma 1. *Given* $T, T' \in \mathcal{T}$ *and a parallel context* $C \in \mathcal{C}_P$, *it holds that:* $C[T]|T' \equiv C[T|T']$.

Proof. Since $C \in \mathcal{C}_P$ there exists T_C such that $C[T] = T_C|T$, and moreover we have that $(T_C|T)|T' \equiv T_C|(T|T') = C[T|T']$.

Lemma 2. $T \xrightarrow{C[C']} T' \iff C'[T] \xrightarrow{C} T'$.

Proof. By induction on the depth of the derivation tree of $T \xrightarrow{C[C']} T'$.

- *Base*. Derivation trees of depth 1 are obtained by rule (rule_appl).
  $T \xrightarrow{C[C']} T' \iff$ there exists $T_1 \mapsto T_1' \in \mathcal{R}$ such that $T_1\sigma = C[C'[T]]$ and $T_1'\sigma = T'$ for some instantiation function $\sigma \iff C'[T] \xrightarrow{C} T'$.
- *Induction step*. We assume that the thesis holds for depth n.
  - (par). We first prove the direction $\Longrightarrow$. Let us assume $T = T_1|T_2$; then $T' = T_1'|T_2$, $T_1 \xrightarrow{C[C']} T_1'$ and $C[C'] \in \mathcal{C}_P$. We have $C'[T_1] \xrightarrow{C} T_1'$ by induction hypothesis, which implies $C'[T_1]|T_2 \xrightarrow{C} T_1'|T_2$ (by applying rule (par)), and hence $C'[T] \xrightarrow{C} T'$, since $T' = T_1'|T_2$, $C' \in \mathcal{C}_P$ and by Lemma 1. The direction $\Longleftarrow$ can be proven symmetrically.
  - (cont). This case is trivial because $C[C'] = \square$. □

We introduce a notion of *strong bisimilarity* between CLS terms.

Definition 8 (Strong Bisimulation). *A binary relation R on terms is a* strong bisimulation *if, given T_1, T_2 such that $T_1 R T_2$, the two following conditions hold:*
$T_1 \xrightarrow{C} T_1' \implies \exists T_2'$ *such that* $T_2 \xrightarrow{C} T_2'$ *and* $T_1' R T_2'$
$T_2 \xrightarrow{C} T_2' \implies \exists T_1'$ *such that* $T_1 \xrightarrow{C} T_1'$ *and* $T_2' R T_1'$.
The strong bisimilarity $\sim$ *is the largest of such relations.*

The strong bisimilarity $\sim$ is a congruence with respect to CLS operators.

Proposition 2 (Strong Congruence). *The relation $\sim$ is a congruence.*

Proof. We show that $\mathcal{S} \stackrel{def}{=} \{\, (C[T_1], C[T_2]) \mid T_1 \sim T_2$ and $C \in \mathcal{C}\}$ is a bisimulation. In particular, we note that $\sim \subseteq \mathcal{S}$ and $T_1 \mathcal{S} T_2 \implies C[T_1]\ \mathcal{S}\ C[T_2]$. Finally, given $T_1 \sim T_2$, we prove by induction on the derivation of $C[T_1] \xrightarrow{C'} T_1'$ that $C[T_1] \xrightarrow{C'} T_1' \implies \exists T_2'. C[T_2] \xrightarrow{C'} T_2'$ and $T_1' \mathcal{S} T_2'$. A detailed proof can be found in Appendix A.1. □

We denote with $\stackrel{\square}{\Longrightarrow}$ a sequence of zero or more transitions $\xrightarrow{\square}$, and with $\stackrel{C}{\Longrightarrow}$, where $C \neq \square$, the sequence of transitions such that $T \stackrel{C}{\Longrightarrow} T'$ if and only if there exist $T_1, T_2 \in \mathcal{T}$ such that $T \stackrel{\square}{\Longrightarrow} T_1 \xrightarrow{C} T_2 \stackrel{\square}{\Longrightarrow} T'$. We have two lemmas.

Lemma 3. *If one of the following two conditions holds:* (i) $C, C' \in \mathcal{C}_P$, (ii) $C = \square$, $C' \in \mathcal{C}$, *then* $T \stackrel{C}{\Longrightarrow} T' \iff C'[T] \stackrel{C}{\Longrightarrow} C'[T']$.

Proof. By definition of $\stackrel{C}{\Longrightarrow}$ and of the labeled semantics. □

Lemma 4. $T \stackrel{C[C']}{\Longrightarrow} T' \iff C'[T] \stackrel{C}{\Longrightarrow} T'$.

Proof. First of all, it is worth noticing that, by Lemma 3, $T \stackrel{\square}{\Longrightarrow} T' \iff C[T] \stackrel{\square}{\Longrightarrow} C[T']$ for any context C. Now, $T \stackrel{C[C']}{\Longrightarrow} T' \iff$ there exist T_1, T_2 such that $T \stackrel{\square}{\Longrightarrow} T_1 \xrightarrow{C[C']} T_2 \stackrel{\square}{\Longrightarrow} T'$. By Lemma 2, we have that $C'[T_1] \xrightarrow{C} T_2$, and hence $C'[T] \stackrel{\square}{\Longrightarrow} C'[T_1] \xrightarrow{C} T_2 \stackrel{\square}{\Longrightarrow} T'$, that is $C'[T] \stackrel{C}{\Longrightarrow} T'$. □

Most of the time we want to consider bisimilarity without taking into account system internal moves. This relation is usually called *weak bisimilarity.*

Definition 9 (Weak Bisimulation). *A binary relation R on terms is a* weak bisimulation *if, given T_1, T_2 such that $T_1 R T_2$, the two following conditions hold:*
$T_1 \xrightarrow{C} T_1' \implies \exists T_2'$ *such that* $T_2 \stackrel{C}{\Longrightarrow} T_2'$ *and* $T_1' R T_2'$
$T_2 \xrightarrow{C} T_2' \implies \exists T_1'$ *such that* $T_1 \stackrel{C}{\Longrightarrow} T_1'$ *and* $T_2' R T_1'$.
The weak bisimilarity $\approx$ *is the largest of such relations.*

Proposition 3 (Weak Congruence). *The relation $\approx$ is a congruence.*

Proof. Similar to the proof of Proposition 2, by using Lemmas 3 and 4. A detailed proof can be found in Appendix A.2. □

Example 1. Consider the following set of rules:

$$\mathcal{R} = \{ \quad a \cdot c \mid b \mapsto e \mid b \quad , \quad d \mid b \mapsto f \quad , \quad e \mapsto a \cdot c \quad , \quad f \mapsto f \quad \}$$

We have that $a \cdot c \sim d$, because $a \cdot c \xrightarrow{\square|b} e \mid b \xrightarrow{\square} a \cdot c \mid b \xrightarrow{\square} e \mid b \xrightarrow{\square} \ldots$ and $d \xrightarrow{\square|b} f \mid b \xrightarrow{\square} f \mid b \xrightarrow{\square} \ldots$, and $e \approx d$, because $e \xrightarrow{\square} a \cdot c \xrightarrow{\square|b} e \mid b \xrightarrow{\square} a \cdot c \mid b \xrightarrow{\square} e \mid b \xrightarrow{\square} \ldots$. On the other hand, $a \cdot c \not\sim d \mid b$ and $a \cdot c \not\approx d \mid b$.

One may also be interested in comparing the behaviour of terms whose evolution is given by the application of two possibly different sets of rewrite rules. To this aim we define *CLS systems* as pairs consisting of a CLS term and a set of rewrite rules.

Definition 10 (System). *A* CLS System *is a pair $\langle T, \mathcal{R} \rangle$ with $T \in \mathcal{T}$, $\mathcal{R} \subseteq \Re$.*

Given a system $\langle T, \mathcal{R} \rangle$, we write $\mathcal{R} : T \xrightarrow{C} T'$ to mean that the transition $T \xrightarrow{C} T'$ is performed by applying a rule in $\mathcal{R}$, and we write $\mathcal{R} : T \stackrel{C}{\Longrightarrow} T'$ to mean that the sequence of transitions $T \stackrel{C}{\Longrightarrow} T'$ is performed by applying rules in $\mathcal{R}$. Now, we introduce strong and weak bisimilarities between CLS systems. With abuse of notation we denote such relations with $\sim$ and $\approx$, respectively.

Definition 11 (Strong Bisimulation on Systems). *A binary relation R on CLS systems is a* strong bisimulation *if, given $\langle T_1, \mathcal{R}_1 \rangle$ and $\langle T_2, \mathcal{R}_2 \rangle$ such that $\langle T_1, \mathcal{R}_1 \rangle R \langle T_2, \mathcal{R}_2 \rangle$, the two following conditions hold:*
$\mathcal{R}_1 : T_1 \xrightarrow{C} T_1' \implies \exists T_2'$ *such that* $\mathcal{R}_2 : T_2 \xrightarrow{C} T_2'$ *and* $\langle T_1', \mathcal{R}_1 \rangle R \langle T_2', \mathcal{R}_2 \rangle$
$\mathcal{R}_2 : T_2 \xrightarrow{C} T_2' \implies \exists T_1'$ *such that* $\mathcal{R}_1 : T_1 \xrightarrow{C} T_1'$ *and* $\langle T_2', \mathcal{R}_2 \rangle R \langle T_1', \mathcal{R}_1 \rangle$.
The strong bisimilarity $\sim$ *is the largest of such relations.*

Definition 12 (Weak Bisimulation on Systems). *A binary relation R on CLS systems is a* weak bisimulation *if, given $\langle T_1, \mathcal{R}_1\rangle$ and $\langle T_2, \mathcal{R}_2\rangle$ such that $\langle T_1, \mathcal{R}_1\rangle R \langle T_2, \mathcal{R}_2\rangle$, the two following conditions hold:*

$$\mathcal{R}_1 : T_1 \xrightarrow{C} T_1' \implies \exists T_2' \text{ such that } \mathcal{R}_2 : T_2 \stackrel{C}{\Longrightarrow} T_2' \text{ and } \langle T_1', \mathcal{R}_1\rangle R \langle T_2', \mathcal{R}_2\rangle$$

$$\mathcal{R}_2 : T_2 \xrightarrow{C} T_2' \implies \exists T_1' \text{ such that } \mathcal{R}_1 : T_1 \stackrel{C}{\Longrightarrow} T_1' \text{ and } \langle T_2', \mathcal{R}_2\rangle R \langle T_1', \mathcal{R}_1\rangle.$$

The weak bisimilarity $\approx$ *is the largest of such relations.*

If we fix a set of rewrite rules, strong and weak bisimilarities on CLS systems correspond to strong and weak bisimilarities on terms, respectively. Namely, for a given $\mathcal{R} \in \Re$, $\langle T_1, \mathcal{R}\rangle \sim \langle T_2, \mathcal{R}\rangle$ if and only if $T_1 \sim T_2$ and $\langle T_1, \mathcal{R}\rangle \approx \langle T_2, \mathcal{R}\rangle$ if and only if $T_1 \approx T_2$. However, as we show in the following example, bisimilarity relations introduced for CLS systems are not congruences.

Example 2. Let $\mathcal{R}_1 = \{a \mid b \mapsto c\}$ and $\mathcal{R}_2 = \{a \mid d \mapsto c\ ,\ \ b \mid e \mapsto c\}$. We have that $\langle a, \mathcal{R}_1\rangle \approx \langle e, \mathcal{R}_2\rangle$ and $\langle b, \mathcal{R}_1\rangle \approx \langle d, \mathcal{R}_2\rangle$, but $\langle a \mid b, \mathcal{R}_1\rangle \not\approx \langle e \mid d, \mathcal{R}_2\rangle$.

Even if bisimilarity on CLS systems are not congruences, they allow us to define equivalence relations on sets of rewrite rules.

Definition 13 (Rules Equivalence). *Two sets of rewrite rules $\mathcal{R}_1$ and $\mathcal{R}_2$ are strongly (resp. weakly) equivalent, denoted $\mathcal{R}_1 \simeq \mathcal{R}_2$ (resp. $\mathcal{R}_1 \cong \mathcal{R}_2$), if and only if for any term $T \in \mathcal{T}$ it holds $\langle T, \mathcal{R}_1\rangle \sim \langle T, \mathcal{R}_2\rangle$ (resp. $\langle T, \mathcal{R}_1\rangle \approx \langle T, \mathcal{R}_2\rangle$).*

Example 3. Given $\mathcal{R}_1 = \{a \mapsto c\}$, $\mathcal{R}_2 = \{a \mapsto f\}$ and $\mathcal{R}_3 = \{a \mapsto b\ ,\ \ b \mapsto c\}$, we have that $\mathcal{R}_1 \simeq \mathcal{R}_2$, but $\mathcal{R}_1 \not\simeq \mathcal{R}_3$ and $\mathcal{R}_1 \cong \mathcal{R}_2$.

Now, if we resort to equivalent rules, we can prove congruence results on CLS systems.

Proposition 4 (Congruences on Systems). *Given $\mathcal{R}_1 \simeq \mathcal{R}_2$ (resp. $\mathcal{R}_1 \cong \mathcal{R}_2$) and $\langle T, \mathcal{R}_1\rangle \sim \langle T', \mathcal{R}_2\rangle$ (resp. $\langle T, \mathcal{R}_1\rangle \approx \langle T', \mathcal{R}_2\rangle$), for any $C \in \mathcal{C}$ we have $\langle C[T], \mathcal{R}_1\rangle \sim \langle C[T'], \mathcal{R}_2\rangle$ (resp. $\langle C[T], \mathcal{R}_1\rangle \approx \langle C[T'], \mathcal{R}_2\rangle$).*

Proof. Since $\mathcal{R}_1 \simeq \mathcal{R}_2$ we have that $\langle T, \mathcal{R}_1\rangle \sim \langle T, \mathcal{R}_2\rangle$; moreover, by hypothesis, $\langle T, \mathcal{R}_1\rangle \sim \langle T', \mathcal{R}_2\rangle$, and therefore $\langle T, \mathcal{R}_2\rangle \sim \langle T', \mathcal{R}_2\rangle$. Now, since the set of rewrite rules is the same ($\mathcal{R}_2$), by the congruence results for CLS terms, we have $\langle C[T], \mathcal{R}_2\rangle \sim \langle C[T'], \mathcal{R}_2\rangle$. Again, since $\mathcal{R}_1 \simeq \mathcal{R}_2$, we have $\langle C[T], \mathcal{R}_1\rangle \sim \langle C[T], \mathcal{R}_2\rangle$, and hence, $\langle C[T], \mathcal{R}_1\rangle \sim \langle C[T'], \mathcal{R}_2\rangle$. The proof is identical for $\cong$ and $\approx$ instead of $\simeq$ and $\sim$, respectively. $\square$

3 An Application to the Modeling of Gene Regulation

In this section we develop a CLS model of the regulation process of the lactose operon in E. coli (Escherichia coli), we use the weak bisimulation on terms to simplify the model and the weak bisimulation on systems to prove a property.

E. coli is a bacterium often present in the intestine of many animals. As most bacteria, it is often exposed to a constantly changing physical and chemical

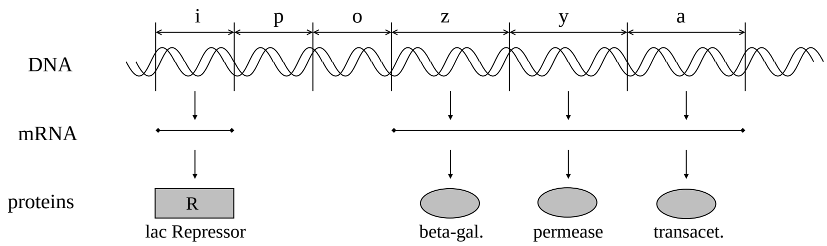

Fig. 2. The lactose operon.

environment, and reacts to changes in its environment through changes in the kinds of proteins it produces.

In general, in order to save energy, bacteria do not synthesize degradative enzymes (which are proteins) unless the substrates for these enzymes are present in the environment. For example, E. coli does not sinthesize the enzymes that degrade lactose (a sugar) unless lactose is in the environment. This phenomenon is called *enzyme induction* or, more generally, *gene regulation* since it is obtained by controlling the transcription of some genes into the corresponding proteins.

Let us consider the lactose degradation example in E. coli. Two enzymes are required to start the breaking process: the *lactose permease*, which is incorporated in the membrane of the bacterium and actively transports the sugar into the cell (without this enzyme lactose can enter the bacterium anyway, but much more slowly), and the *beta galactosidase*, which splits lactose into glucose and galactose. The bacterium produces also the *transacetylase* enzyme, whose function is unknown, but is surely related with the usage of lactose.

The sequence of genes in the DNA of E. coli which produces the described enzymes, is known as the *lactose operon* (see Fig. 2). It is composed by six genes: the first three (i, p, o) regulate the production of the enzymes, and the last three (z, y, a), called *structural genes*, are transcribed (when allowed) into the mRNA for beta galactosidase, lactose permease and transacetylase, respectively[1].

The regulation process is as follows (see Fig. 3): gene i encodes the *lac Repressor*, which in the absence of lactose, binds to gene o (the *operator*). Transcription of structural genes into mRNA is performed by the RNA polymerase enzyme, which usually binds to gene p (the *promoter*) and scans the operon from left to right by transcribing the three structural genes z, y and a into a single mRNA fragment. When the lac Repressor is bound to gene o, it becomes an obstacle for the RNA polymerase, and transcription of the structural genes is not performed. On the other hand, when lactose is present inside the bacterium, it binds to the Repressor and this cannot stop any more the activity of the RNA polymerase. In this case transcription is performed and the three enzymes for lactose degradation are synthesized.

[1] We recall that in protein synthesis first the DNA of one or more genes is transcribed into a piece of mRNA, then the mRNA is translated into one or more proteins.

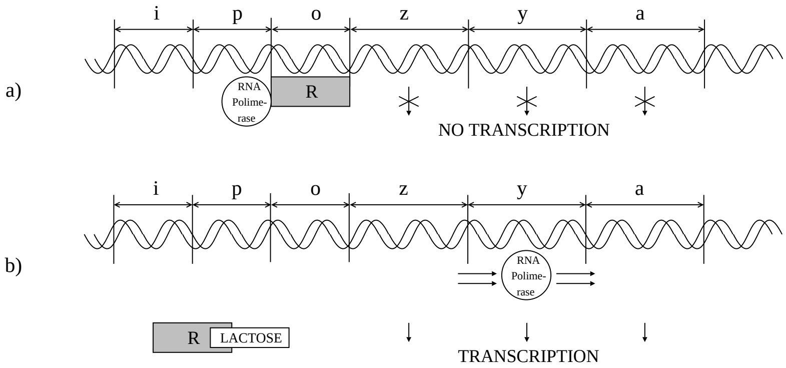

Fig. 3. The regulation process. In the absence of lactose (case a) the lac Repressor binds to gene o and precludes the RNA polymerase from trascribing genes z,y and a. When lactose is present (case b) it binds to and inactivates the lac Repressor.

Now we describe how to model the gene regulation process with CLS. For the sake of simplicity we give a partial model, in the sense that we describe how the transcription of the structural genes is activated when the lactose is in the environment, but we do not describe how the transcription of such genes is stopped when the lactose disappears. Moreover, in order to simplify the example, we assume that genes are transcribed directly into proteins (thus avoiding the modeling of the mRNA), that the lac Repressor is transcribed from gene i without the need of the RNA polymerase and that it can be produced only once. Finally, we assume that one RNA polymerase is present inside the bacterium.

We model the membrane of the bacterium as the looping sequence $\big(m\big)^L$, where the elementary constituent m generically denotes the whole membrane surface in normal conditions. Moreover, we model the lactose operon as the sequence $lacI \cdot lacP \cdot lacO \cdot lacZ \cdot lacY \cdot lacA$ ($lacI$–A for short), in which each element corresponds to a gene, and we replace $lacO$ with RO in the sequence when the lac Repressor is bound to gene o. When the lac Repressor is unbound, it is modeled by the elementary constituent $repr$. Finally, we model the RNA polymerase as the elementary constituent $polym$, a molecule of lactose as the elementary constituent $LACT$, and beta galactose, lactose permease and transacetylase enzymes as elementary constituents $betagal, perm$ and $transac$, respectively.

When no lactose is present the bacterium is modeled by the following term:

$$Ecoli \ ::= \ \big(m\big)^L \rfloor \, (lacI\text{–}A \mid polym)$$

The transcription of the DNA is modeled by the following set of rules:

$$lacI \cdot \widetilde{x} \quad \mapsto \quad lacI' \cdot \widetilde{x} \mid repr \qquad \text{(R1)}$$

$$polym \mid \widetilde{x} \cdot lacP \cdot \widetilde{y} \quad \mapsto \quad \widetilde{x} \cdot PP \cdot \widetilde{y} \qquad \text{(R2)}$$

$$\widetilde{x} \cdot PP \cdot lacO \cdot \widetilde{y} \quad \mapsto \quad \widetilde{x} \cdot lacP \cdot PO \cdot \widetilde{y} \qquad \text{(R3)}$$

$$\widetilde{x} \cdot PO \cdot lacZ \cdot \widetilde{y} \mapsto \widetilde{x} \cdot lacO \cdot PZ \cdot \widetilde{y} \tag{R4}$$
$$\widetilde{x} \cdot PZ \cdot lacY \cdot \widetilde{y} \mapsto \widetilde{x} \cdot lacZ \cdot PY \cdot \widetilde{y} \mid betagal \tag{R5}$$
$$\widetilde{x} \cdot PY \cdot lacA \mapsto \widetilde{x} \cdot lacY \cdot PA \mid perm \tag{R6}$$
$$\widetilde{x} \cdot PA \mapsto \widetilde{x} \cdot A \mid transac \mid polym \tag{R7}$$

Rule (R1) describes the transcription of gene i into the lac Repressor. After transcription $lacI$ becomes $lacI'$ to avoid further productions of the lac Repressor. Rule (R2) describes the binding of the RNA polymerase to gene p. We denote the complex formed by the binding RNA polymerase to a gene $lac\_$ with the elementary constituent $P\_$. Rules (R3)–(R6) describe the scanning of the DNA performed by the RNA polymerase and the consequent production of enzymes. Rule (R3) can be applied (and the scanning can be performed) only when the sequence contains $lacO$ instead of RO, that is when the lac Repressor is not bound to gene o. Finally, in rule (R7) the RNA polymerase terminates the scanning and releases the sequence.

The following rules describe the binding of the lac Repressor to gene o, and what happens when lactose is present in the environment of the bacterium:

$$repr \mid \widetilde{x} \cdot lacO \cdot \widetilde{y} \mapsto \widetilde{x} \cdot RO \cdot \widetilde{y} \tag{R8}$$
$$LACT \mid \left(m \cdot \widetilde{x}\right)^L \rfloor X \mapsto \left(m \cdot \widetilde{x}\right)^L \rfloor (X \mid LACT) \tag{R9}$$
$$\widetilde{x} \cdot RO \cdot \widetilde{y} \mid LACT \mapsto \widetilde{x} \cdot lacO \cdot \widetilde{y} \mid RLACT \tag{R10}$$

Rule (R8) describes the binding of the lac Repressor to gene o, rule (R9) models the passage of the lactose through the membrane of the bacterium and rule (R10) the removal of the lac Repressor from gene o operated by the lactose. In this rule the elementary constituent $RLACT$ denotes the binding of the lactose to the lac Repressor.

Finally, the following rules describe the behaviour of the enzymes synthesized when lactose is present, and their degradation:

$$\left(\widetilde{x}\right)^L \rfloor (perm \mid X) \mapsto \left(perm \cdot \widetilde{x}\right)^L \rfloor X \tag{R11}$$
$$LACT \mid \left(perm \cdot \widetilde{x}\right)^L \rfloor X \mapsto \left(perm \cdot \widetilde{x}\right)^L \rfloor (LACT \mid X) \tag{R12}$$
$$betagal \mid LACT \mapsto betagal \mid GLU \mid GAL \tag{R13}$$
$$perm \mapsto \epsilon \tag{R14}$$
$$betagal \mapsto \epsilon \tag{R15}$$
$$transac \mapsto \epsilon \tag{R16}$$

Rule (R11) describes the incorporation of the lactose permease in the membrane of the bacterium, rule (R12) the transporation of lactose from the environment to the interior performed by the lactose permease, and rule (R13) the decomposition of the lactose into glucose (denoted GLU) and galactose (denoted GAL) performed by the beta galactosidase. Finally, rules (R14),(R15) and (R16) describe degradation of the lactose permease, the beta galactosidase and the transacetylase enzymes, respectively.

Let us denote the set of rewrite rules $\{(R1), \ldots, (R16)\}$ as $\mathcal{R}_{lac}$, and the lactose operon $lacI' \cdot lacP \cdot lacO \cdot lacZ \cdot lacY \cdot lacA$, after the production of the lac Repressor, as $lacI'{-}A$. An example of possible sequence of transitions which can be performed by the term $Ecoli$ by applying rules in $\mathcal{R}_{lac}$ is the following:

$$
\begin{array}{rcl}
Ecoli & \stackrel{\Box}{\Longrightarrow} & (m)^L \rfloor (lacI' \cdot lacP \cdot RO \cdot lacZ \cdot lacY \cdot lacA \mid polym) \\
& \stackrel{LACT|\Box}{\Longrightarrow} & (m)^L \rfloor (lacI'{-}A|polym|RLACT) \\
& \stackrel{\Box}{\Longrightarrow} & (perm \cdot m)^L \rfloor (lacI'{-}A|betagal|transac|polym|RLACT) \\
& \stackrel{LACT|\Box}{\Longrightarrow} & (perm \cdot m)^L \rfloor (lacI'{-}A|betagal|transac|polym|RLACT|GLU|GAL)
\end{array}
$$

In the example, by applying rules (R1) and (R8), $Ecoli$ produces the lac Repressor, which binds to gene o in the lactose operon. Then, the bacterium interacts with an environment containing a molecule of lactose (represented by the context $LACT|\Box$): by applying rule (R9) the lactose enters the membrane of the bacterium and by applying rule (R10) it binds to the lac Repressor. Then, a sequence of internal transitions are performed by applying rules (R2)–(R7) and (R11): the result is the transcription of the structural genes and the incorporation of the lactose permease in the membrane of the bacterium. Finally, the term interacts with an environment containing another molecule of lactose, which enters the bacterium and is decomposed into GLU and GAL. The rules applied in this phase are (R12) and (R13).

Note that, if one starts from $Ecoli$, every time (R11) can be applied, also (R9) can be applied giving the same results. Therefore, rule (R11) seems to be redundant. Nevertheless, rule (R11) describes a precise phenomenon, namely the action performed by the lactose permease, which is modeled by no other rule. The difference between rules (R9) and (R11) is that the latter describes a much faster event. However, since quantitative aspects are not considered in the calculus, the difference between the two rules does not appear.

The model can be simplified. Let us denote by T the term $lacP \cdot lacO \cdot lacZ \cdot lacY \cdot lacA \mid repr$. Note that T behaves as $lacI{-}A$ apart from the transcription of the lac Repressor, which is already present. Therefore, the transition system derived from T corresponds to the one derived form $lacI{-}A$ apart from some $\Box$–labeled transitions obtained by the application of rule (R1). As a consequence, $T \approx lacI{-}A$. Now, since $\approx$ is a congruence, we may replace $lacI{-}A$ with T in $Ecoli$, thus obtaining an equivalent term.

Now we use the weak bisimulation defined on CLS systems to verify a simple property of the described system, namely that by starting from a situation in which the lac Repressor is bound to gene o, and none of the three enzymes produced by the lactose operon is present (which is a typical stable state of the system), production of such enzymes can start only if lactose appears.

In order to verify this property with the bisimulation relation we defined, we need to modify the rules of the model in such a way that the event of starting the production of the three enzymes becomes observable. We can obtain this result, for instance, by replacing rule (R10) with the rule

$$
\begin{array}{l}
(\widetilde{w})^L \rfloor (\widetilde{x} \cdot RO \cdot \widetilde{y} \mid LACT \mid X) \mid START \quad \mapsto \\
\qquad\qquad (\widetilde{w})^L \rfloor (\widetilde{x} \cdot lacO \cdot \widetilde{y} \mid RLACT \mid X) \qquad\qquad \text{(R10bis)}
\end{array}
$$

We choose to modify (R10) because we know that, after applying rule (R10), the three enzymes can be produced freely, and we add to the rule the interaction with the artificial element $START$ in the environment in order to obtain $\square|START$ as a transition label every time the rule is applied to the term.

The property we want to verify is satisfied by the system $\langle T_1, \mathcal{R}\rangle$, where $\mathcal{R}$ consists of the following four rules:

$$
\begin{array}{lcll@{\qquad}lcll}
T_1 \mid LACT & \mapsto & T_2 & \text{(R1')} & T_2 \mid START & \mapsto & T_3 & \text{(R3')} \\
T_2 \mid LACT & \mapsto & T_2 & \text{(R2')} & T_3 \mid LACT & \mapsto & T_3 & \text{(R4')}
\end{array}
$$

for some ground terms T_1, T_2 and T_3.

It can be proved that the system $\langle T_1, \mathcal{R}\rangle$ is weakly bisimilar to the system $\langle EcoliRO, (\mathcal{R}_{lac} \setminus \{R_{10}\}) \cup \{(R_{10bis)}\}\rangle$, where:

$$EcoliRO = (m)^L \rfloor lacI' \cdot PP \cdot RO \cdot lacZ \cdot lacY \cdot lacA$$

In particular, the bisimulation relation associates (the system containing) term T_1 with (the system containing) term $EcoliRO$, term T_2 with all the terms representing a bacterium containing at least one molecule of lactose with the Lac repressor bound to gene o, and, finally, term T_3 with all the terms in which the repressor has left gene o and is bound to the lactose.

4 Conclusions

We have presented a variant of CLS, we have given the calculus a labeled semantics and we have defined bisimulation relations on terms and on systems of the calculus. We have proven bisimilarities to be congruences and shown an example of application of CLS to the modeling of a biological system.

As future work, we plan to develop a quantitative extension of the calculus in which speed of events are modeled as rates of rewrite rule applications.

References

1. R. Alur, C. Belta, F. Ivancic, V. Kumar, M. Mintz, G.J. Pappas, H. Rubin and J. Schug. "Hybrid Modeling and Simulation of Biomolecular Networks". Hybrid Systems: Computation and Control, LNCS 2034, pages 19–32, Springer, 2001.
2. R. Barbuti, S. Cataudella, A. Maggiolo-Schettini, P. Milazzo and A. Troina. "A Probabilistic Model for Molecular Systems". Fundamenta Informaticae, volume 67, pages 13–27, 2005.
3. R. Barbuti, A. Maggiolo-Schettini, P. Milazzo and A. Troina. "A Calculus of Looping Sequences for Modelling Microbiological Systems". Fundamenta Informaticae, volume 72, pages 1–15, 2006 (in press: draft available at `http://www.di.unipi.it/~milazzo/`).
4. L. Cardelli. "Brane Calculi. Interactions of Biological Membranes". CMSB'04, LNCS 3082, pages 257–280, Springer, 2005.
5. N. Chabrier-Rivier, M. Chiaverini, V. Danos, F. Fages and V. Schachter. "Modeling and Querying Biomolecular Interaction Networks". Theoretical Computer Science, volume 325, number 1, pages 25-44, 2004.

6. M. Curti, P. Degano, C. Priami and C.T. Baldari. "Modelling Biochemical Pathways through Enhanced pi-calculus". Theoretical Computer Science, volume 325, number 1, pages 111–140, 2004.
7. V. Danos and C. Laneve. "Formal Molecular Biology". Theoretical Computer Science, volume 325, number 1, pages 69–110, 2004.
8. J. Leifer and R. Milner. "Deriving Bisimulation Congruences for Reactive Systems". CONCUR'00, LNCS 1877, pages 243–258, Springer, 2000.
9. H. Matsuno, A. Doi, M. Nagasaki and S. Miyano."Hybrid Petri Net Representation of Gene Regulatory Network". Pacific Symposium on Biocomputing, World Scientific Press, pages 341–352, 2000.
10. C. Priami and P. Quaglia "Beta Binders for Biological Interactions". CMSB'04, LNCS 3082, pages 20–33, Springer, 2005.
11. A. Regev, E.M. Panina, W. Silverman, L. Cardelli and E. Shapiro. "BioAmbients: An Abstraction for Biological Compartments". Theoretical Computer Science, volume 325, number 1, pages 141–167, 2004.
12. A. Regev, W. Silverman and E.Y. Shapiro. "Representation and Simulation of Biochemical Processes Using the pi-calculus Process Algebra". Pacific Symposium on Biocomputing, World Scientific Press, pages 459–470, 2001.
13. P. Sewell. "From Rewrite Rules to Bisimulation Congruences". Theoretical Computer Science, volume 274, pages 183–230, 2002.

A Proofs

A.1 Proof of Proposition 2

We show that $\mathcal{S} \stackrel{def}{=} \{\ (C[T_1], C[T_2])\ |T_1 \sim T_2$ and C is a context$\}$ is a bisimulation. First of all, it is worth noting that $\mathcal{S}$ includes $\sim$ because $C[T_1] = T_1$ when $C = \square$. Moreover, the following implication holds:

$$T_1 \mathcal{S} T_2 \implies C[T_1]\mathcal{S}C[T_2] \tag{1}$$

because $T_1\mathcal{S}T_2$ implies $\exists C'.T_1 = C'[T_1'], T_2 = C'[T_2']$ for some $T_1', T_2' \in \mathcal{T}$ such that $T_1' \sim T_2'$. Hence $C[C'[T_1']]\mathcal{S}C[C'[T_2']]$, that is $C[T_1]\mathcal{S}C[T_2]$.

Now, since $\sim$ is a symmetric relation, we have only to show that given $T_1 \sim T_2$ the following holds: $C[T_1] \xrightarrow{C'} T_1' \implies \exists T_2'.C[T_2] \xrightarrow{C'} T_2'$ and $T_1'\mathcal{S}T_2'$.

We prove this by induction on the depth of the derivation tree of $C[T_1] \xrightarrow{C'} T_1'$:

– *Base case* (rule_appl). There exists $T \mapsto T_1' \in \mathcal{R}$ such that $C'[C[T_1]] \equiv T\sigma$ for some instantiation function σ. This implies $T_1 \xrightarrow{C'[C]} T_1'$ and, since $T_1 \sim T_2$, there exists T_2' such that $T_2 \xrightarrow{C'[C]} T_2'$ with $T_1' \sim T_2'$. Finally, $T_2 \xrightarrow{C'[C]} T_2'$ implies $C[T_2] \xrightarrow{C'} T_2'$ by Lemma 2 and $T_1' \sim T_2'$ implies $T_1'\mathcal{S}T_2'$.

– *Induction step* (par). In this case $C = C_1[C_2]$ for some C_2 and where $C_1 = \square|T$ for some T. Hence, $C[T_1] = C_1[C_2[T_1]]$ and by the premise of the inference rule we obtain $C_2[T_1] \xrightarrow{C'} T_1''$. It follows that $T_1' = C_1[T_1'']$. By applying the induction hypothesis we have that there exists T_2'' such that $C_2[T_2] \xrightarrow{C'} T_2''$ and $T_1''\mathcal{S}T_2''$, hence, by applying the (par) rule, $C_1[C_2[T_2]] \xrightarrow{C'} C_1[T_2'']$, that is

$C[T_2] \xrightarrow{C'} T_2'$. Finally, by the closure of $\mathcal{S}$ to contexts given in (1), we have $C_1[T_1'']\mathcal{S}C_1[T_2'']$, that is $T_1'\mathcal{S}T_2'$.

– *Induction step* (cont). In this case $C' = \Box$ and $C = C_1[C_2]$ for some C_2 and where $C_1 = T \rfloor \Box$ for some T. Hence, $C[T_1] = C_1[C_2[T_1]]$ and by the premise of the inference rule we obtain $C_2[T_1] \xrightarrow{\Box} T_1''$. It follows that $T_1' = C_1[T_1'']$. By applying the induction hypothesis we have that there exists T_2'' such that $C_2[T_2] \xrightarrow{\Box} T_2''$ and $T_1''\mathcal{S}T_2''$, hence, by applying the (cont) rule, $C_1[C_2[T_2]] \xrightarrow{\Box} C_1[T_2'']$, that is $C[T_2] \xrightarrow{\Box} T_2'$. Finally, by the closure of $\mathcal{S}$ to contexts given in (1), we have $C_1[T_1'']\mathcal{S}C_1[T_2'']$, that is $T_1'\mathcal{S}T_2'$. $\Box$

A.2 Proof of Proposition 3

We show that $\mathcal{S} \stackrel{def}{=} \{ (C[T_1], C[T_2]) \,|T_1 \approx T_2 \text{ and } C \text{ is a context}\}$ is a weak bisimulation. First of all it is worth noting that $\mathcal{S}$ includes $\approx$ because $C[T_1] = T_1$ when $C = \Box$. Moreover, the following implication holds:

$$T_1\mathcal{S}T_2 \implies C[T_1]\mathcal{S}C[T_2] \tag{2}$$

because $T_1\mathcal{S}T_2$ implies $\exists C'.T_1 = C'[T_1'], T_2 = C'[T_2']$ for some $T_1', T_2' \in \mathcal{T}$ such that $T_1' \approx T_2'$. Hence $C[C'[T_1']]\mathcal{S}C[C'[T_2']]$ that is $C[T_1]\mathcal{S}C[T_2]$.

Now, since $\approx$ is a symmetric relation, we have only to show that given $T_1 \approx T_2$ the following holds: $C[T_1] \xrightarrow{C'} T_1' \implies \exists T_2'.C[T_2] \stackrel{C'}{\Longrightarrow} T_2'$ and $T_1'\mathcal{S}T_2'$.

We prove this by induction on the depth of the derivation tree of $C[T_1] \xrightarrow{C'} T_1'$:

– *Base case* (rule_appl). There exists $T \mapsto T_1' \in \mathcal{R}$ such that $C'[C[T_1]] \equiv T\sigma$ for some instantiation function σ. This implies $T_1 \xrightarrow{C'[C]} T_1'$ and, since $T_1 \approx T_2$, there exists T_2' such that $T_2 \stackrel{C'[C]}{\Longrightarrow} T_2'$ with $T_1' \approx T_2'$. Finally, $T_2 \stackrel{C'[C]}{\Longrightarrow} T_2'$ implies $C[T_2] \stackrel{C'}{\Longrightarrow} T_2'$ by Lemma 4 and $T_1' \approx T_2'$ implies $T_1'\mathcal{S}T_2'$.

– *Induction step* (par). In this case $C = C_1[C_2]$ for some C_2 and where $C_1 = \Box|T$ for some T. Hence, $C[T_1] = C_1[C_2[T_1]]$ and by the premise of the inference rule we obtain $C_2[T_1] \xrightarrow{C'} T_1''$. It follows $T_1' = C_1[T_1'']$. By applying the induction hypothesis we have that there exists T_2'' such that $C_2[T_2] \stackrel{C'}{\Longrightarrow} T_2''$ and $T_1''\mathcal{S}T_2''$, hence, by Lemma 3, $C_1[C_2[T_2]] \stackrel{C'}{\Longrightarrow} C_1[T_2'']$, that is $C[T_2] \stackrel{C'}{\Longrightarrow} T_2'$. By the closure of $\mathcal{S}$ to contexts given in (2), we have $C_1[T_1'']\mathcal{S}C_1[T_2'']$, that is $T_1'\mathcal{S}T_2'$.

– *Induction step* (cont). In this case $C' = \Box$ and $C = C_1[C_2]$ for some C_2 and where $C_1 = T \rfloor \Box$ for some T. Hence, $C[T_1] = C_1[C_2[T_1]]$ and by the premise of the inference rule we obtain $C_2[T_1] \xrightarrow{\Box} T_1''$. It follows that $T_1' = C_1[T_1'']$. By applying the induction hypothesis we have that there exists T_2'' such that $C_2[T_2] \stackrel{\Box}{\Longrightarrow} T_2''$ and $T_1''\mathcal{S}T_2''$, hence, by Lemma 3, $C_1[C_2[T_2]] \stackrel{\Box}{\Longrightarrow} C_1[T_2'']$, that is $C[T_2] \stackrel{\Box}{\Longrightarrow} T_2'$. Finally, by the closure of $\mathcal{S}$ to contexts given in (2), we have $C_1[T_1'']\mathcal{S}C_1[T_2'']$, that is $T_1'\mathcal{S}T_2'$. $\Box$