On the existence and decidability of unique decompositions of processes in the applied π-calculus

Jannik Dreier, Cristian Ene, Pascal Lafourcade, Yassine Lakhnech

On the Existence and Decidability of Unique Decompositions of Processes in the Applied π-Calculus

Jannik Dreier[a,b,c,*], Cristian Ene[d], Pascal Lafourcade[e,f], Yassine Lakhnech[d]

[a] *Université de Lorraine, Loria, UMR 7503, Vandoeuvre-lès-Nancy, F-54506, France*
[b] *Inria, Villers-lès-Nancy, F-54600, France*
[c] *CNRS, Loria, UMR 7503, Vandoeuvre-lès-Nancy, F-54506, France*
[d] *Laboratoire VERIMAG, Université Joseph Fourier, Grenoble, France*
[e] *Clermont Université, Université d'Auvergne, LIMOS, Clermont-Ferrand, France*
[f] *CNRS, UMR 6158, LIMOS, Aubière, France*

Abstract

Unique decomposition has been a subject of interest in process algebra for a long time (for example in BPP [1] or CCS [2, 3]), as it provides a normal form and useful cancellation properties. We provide two parallel decomposition results for subsets of the applied π-calculus: we show that every closed normed (i.e. with a finite shortest complete trace) process P can be decomposed uniquely into prime factors P_i with respect to strong labeled bisimilarity, i.e. such that $P \sim_l P_1|\ldots|P_n$. Moreover, we prove that closed finite processes can be decomposed uniquely with respect to weak labeled bisimilarity. We also investigate whether efficient algorithms that compute the unique decompositions exist. The simpler problem of deciding whether a process is in its unique decomposition form is undecidable in general in both cases, due to potentially undecidable equational theories. Moreover, we show that the unique decomposition remains undecidable even given an equational theory with a decidable word problem.

Keywords: Applied π-Calculus, Unique Decomposition, Normal Form, Weak Bisimilarity, Strong Bisimilarity, Cancellation, Decidability, Equational Theory, Word Problem, Process Calculus, Behavioural Equivalence

1. Introduction

Process algebras or calculi are used to formally model and analyze distributed systems. Famous examples include the Calculus of Communicating

[☆] The previous – and officially published – version of this article contained an error in the definition of the norm of processes, pointed out by Daniel Hirschkoff, Matias D. Lee, and Bas Luttik. This version contains the corrected definition.

[*] Corresponding author

Email addresses: jannik.dreier@loria.fr (Jannik Dreier), cristian.ene@imag.fr (Cristian Ene), pascal.lafourcade@udamail.fr (Pascal Lafourcade), yassine.lakhnech@imag.fr (Yassine Lakhnech)

Systems (CCS) due to Milner [4], or Basic Parallel Processes (BPP) [1]. These calculi contain basic operations such as emission and reception of messages as well as parallel composition or interleaving. As an extension to CCS, Milner, Parrow and Walker developed the π-calculus [5], which also features channel passing and scope extrusion. Abadi and Fournet [6] subsequently proposed the applied π-calculus, a variant of the π-calculus designed for the verification of cryptographic protocols. It additionally features user-defined equational theories to model cryptographic primitives and active substitutions.

In a process algebra the question of process decomposition naturally arises: given an equivalence relation $\simeq$ on processes, can we rewrite a process P as $P \simeq P_1|P_2|\ldots|P_n$, where each P_i is prime in the sense that it cannot be rewritten as the parallel composition of two non-zero processes?

More formally, we say that a process P is prime w.r.t. $\simeq$ if for all processes Q and R such that $P \simeq Q|R$ we have that $Q \simeq 0$ or $R \simeq 0$ (where 0 is the empty process). Then we can state the decomposition problem as follows: given an equivalence relation $\simeq$ and a process P, return $P_1, \ldots, P_n$ such that $P \simeq P_1|P_2|\ldots|P_n$ and all P_i are prime. We say that the decomposition w.r.t. $\simeq$ is unique, if for all processes P its decomposition is unique up to $\simeq$ and reordering of the prime processes P_i (due to the associativity and commutativity of "|").

Such a decomposition provides a maximally parallelized version of a given process P. Additionally, if the decomposition is unique, it provides a normal form, and a cancellation result in the sense that $P|Q \simeq P|R$ implies $Q \simeq R$ for all P, Q and R. This is convenient in proofs, for example when proving the equivalence of different security notions in electronic voting [7].

Moreover, if there is a procedure to transform a process into its normal form, such a unique decomposition can also be used to verify the equivalence of two processes [8]: it suffices to verify whether the factors on both sides are identical up to equivalence, associativity and commutativity.

Our Contributions. We provide two decomposition results for subsets of the applied π-calculus. In a first step, we prove that closed normed processes (i.e. with a finite shortest complete trace) can be uniquely decomposed with respect to strong labeled bisimilarity. In the second step we show that every closed finite process (i.e. with a finite longest complete trace) can be uniquely decomposed with respect to weak labeled bisimilarity, the standard notion of bisimilarity in the applied π-calculus. Note that although we require the processes to be finite or normed, no further hypothesis is needed, i.e. they may use the full power of the calculus including channel passing and scope extrusion. As a direct consequence of the uniqueness of the decomposition, we also obtain cancellation results for both cases.

Moreover, we show that in both cases computing the unique decomposition of a process is undecidable in general, due to potentially undecidable equational theories. We also prove that the problem remains undecidable in both cases even if the word problem in the equational theory is decidable.

$M, N ::=$		terms
	a, b, c, n, m, k	names
	x, y, z	variables
	$f(M_1, \ldots, M_l)$	function application

Figure 1: Grammar for terms

Outline of the Paper. In Section 2, we recall the syntax and semantics of the applied π-calculus. In Section 3 we present several notions of equivalence and bisimilarity, and then define the depth and norm of a process in Section 4. In the following we present our unique decomposition for strong and weak bisimilarity in Section 5 and 6, respectively. Finally we discuss the (un)decidability of computing the unique decomposition for a process in Section 7. In Section 8 we review related work concerning unique decomposition of processes, before concluding the paper in Section 9.

2. The applied π-calculus

The applied π-calculus relies on a *type* or *sort* system for terms. It includes a set of *base* types such as `Integer`, `Key` or `Data`. Additionally, if τ is a type, then $\texttt{Channel}\langle\tau\rangle$ is a type (intuitively the type of a channel transmitting terms of type τ).

We suppose a *signature* Σ of functions, which consists of a finite set of *function symbols* with the associated arities and sorts. For example $\texttt{enc}(\mathit{message}, \mathit{key})$, $\texttt{dec}(\mathit{message}, \mathit{key})$ are of arity two with two parameters of sorts `Data` and `Key`, returning a value of type `Data`. A function with arity zero is a *constant*.

Terms in the applied π-calculus are built of *names* (which typically correspond to basic units of data or channels), *variables* (which can represent more complex expressions) and function symbols from the *signature* Σ following the grammar depicted in Figure 1. These terms have to be correct with respect to arities and sorts of the function symbols, variables and names. Variables and names can have every type, and functions take and return only values of base types. We assume infinite sets of names and variables.

Functions typically include encryption and decryption, hashing, signing and so on. Equalities are modeled using an equational theory E which defines a relation $=_E$. A classical example, which describes the correctness of symmetric encryption, is $\texttt{dec}(\texttt{enc}(\mathit{message}, \mathit{key}), \mathit{key}) =_E \mathit{message}$. To simplify the notation we sometimes omit the subscript $_E$ if this is clear from the context that E is the equational theory defining the equality.

n-tuples can be implemented e.g. using a function $\texttt{tuple}_n(M_1, \ldots, M_n)$ and the equations

$$\forall i \colon \texttt{proj}_i(\texttt{tuple}_n(M_1, \ldots, M_n)) = M_i$$

To simplify notation we also write $(M_1, \ldots, M_n)$ for $\texttt{tuple}_n(M_1, \ldots, M_n)$, this assumes the function $\texttt{tuple}_\texttt{n}$ and the destructors $\texttt{proj}_i$ with the equations as defined above.

$P, Q ::=$	plain processes
0	null process
$P \mid Q$	parallel composition
$!P$	replication
$\nu n.P$	name restriction ("new")
$\texttt{if } M = N \texttt{ then } P \texttt{ else } Q$	conditional (M, N terms)
$\texttt{in}(u, x).P$	input on channel u assigned to x
$\texttt{out}(u, M).P$	output of term M on channel u

Figure 2: Grammar for Plain Processes

$A, B, P, Q ::=$	extended processes
P	plain process
$A \mid B$	parallel composition
$\nu n.A$	name restriction
$\nu x.A$	variable restriction
$\{^M/_x\}$	active substitution

Figure 3: Grammar for Extended Processes

There are two types of processes in the applied π-calculus: *plain processes* and *extended* or *active processes*. *Plain processes* are constructed using the grammar depicted in Figure 2. The null process 0 does nothing, the parallel composition $P|Q$ executes P and Q in parallel, and the replication $!P$ executes an unbounded number of copies of P in parallel. The process $\nu n.P$ creates a new, private name n and continues as P. The process $\texttt{if } M = N \texttt{ then } P \texttt{ else } Q$ behaves as P if $N =_E M$ and as Q otherwise. Note the equality with respect to the equational theory, and that we require M and N to have the same type. The process $\texttt{in}(u, x).P$ inputs a message on channel u, assigns it to the variable x of type τ_x and continues as P. We assume that u is of type $\texttt{Channel}\langle\tau_x\rangle$. Finally $\texttt{out}(u, M).P$ outputs M (of type τ_M) on channel u and continues as P. Again, u has to be of type $\texttt{Channel}\langle\tau_M\rangle$.

Extended processes are plain processes or active substitutions as shown in Figure 3. This distinction between extended and plain processes ensures that active substitutions can only occur on the top level or under restrictions, but not under replication (to avoid multiple substitutions defining the same variable), conditionals or input and output. According to the semantics, active substitutions are only created when terms are output, but not when two processes synchronize, and will thus only appear at the top level (see Example 2).

Note that the applied π-calculus does not include the "+"-operator which implements a nondeterministic choice, yet we can implement something similar using a restricted channel (see Example 8). For more details on encoding the operator with respect to different semantics, see [9, 10].

The active substitution $\{^M/_x\}$ replaces all free occurrences of the variable x inside all parallel processes with the term M. We do not allow two active

substitutions to define the same variable, as this might lead to situations with unclear semantics. We also require substitutions to be well-sorted (i.e., respecting the sorts of the variables, names and functions) and cycle-free (i.e., a variable cannot occur inside the substitution defining it; moreover, if y occurs in a substitution defining x, then x cannot occur in the substitution defining y, and so on), and only allow active substitutions on variables of base sorts (this ensures that labeled bisimilarity and observational equivalence coincide, see Section 3). An occurrence of a name n is *bound* if it is in the scope of a restriction νn, an occurrence of a variable x is *bound* if it is in the scope of a restriction νx or of an input $\mathtt{in}(u, x)$. All unbound occurrences of names and variables are *free*. A name or variable is bound or free in a process A if it has a bound or free occurrence in A. We denote by $fv(A)$, $bv(A)$, $fn(A)$, $bn(A)$ the *free variables*, *bound variables*, *free names* or *bound names* of A respectively.

As an additional notation we write $\nu S.A$ for $\nu s_1.\nu s_2 \ldots \nu s_n.A$ where $s_1, \ldots s_n$ are the elements of a set S of variables and names. By abuse of notation we sometimes leave out ".0" at the end of a process. We also write A^k for $A| \ldots |A$ (k times), in particular $A^0 = 0$ as 0 is the neutral element of parallel composition.

To avoid unnecessary parentheses and improve readability of processes, we assume that all operators take precedence over parallel composition, for example we write $0 \mid \mathtt{in}(u, x).\mathtt{out}(u, M)$ for $0 \mid (\mathtt{in}(u, x).\mathtt{out}(u, M))$.

The *frame* $\Phi(A)$ of an extended process A is obtained by replacing all plain processes in A by 0. This frame can be seen as a representation of what is statically known to the environment about a process. The *domain* $dom(\Phi)$ of a frame Φ is the set of free variables for which Φ defines a substitution. By abuse of notation, we also write $dom(A)$ to denote the domain of the frame $\Phi(A)$ of an extended process A. Note that $dom(A) \subseteq fv(A)$, and that as we cannot have two active substitutions for the same variable, $P = Q|R$ implies $dom(P) = dom(Q) \cup dom(R)$ and $dom(Q) \cap dom(R) = \emptyset$. A frame or process is *closed* if all variables are bound or defined by an active substitution. An *evaluation context* $C[\_]$ denotes an extended process with a hole for an extended process. This implies that the hole is not under replication, a conditional, an input or an output, according to the syntax. A context $C[\_]$ closes A when $C[A]$ is closed.

The semantics of the calculus presupposes a notion of *structural equivalence* ($\equiv$), which is defined as the smallest (w.r.t. subset inclusion) equivalence relation on extended processes that is closed under application of evaluation contexts, α-conversion on bound names and bound variables such that the rules in Figure 4 hold.

Note the contagious nature of active substitutions: they apply to every parallel process using rule SUBST. The process $\{^M/_x\}$ on the left hand side of the parallel composition in rule SUBST denotes the active substitution $\{^M/_x\}$, and on the right hand side $A\langle^M/_x\rangle$ denotes the process A where all occurrences of x have been replaced with M. We call $\langle^M/_x\rangle$ an *implicit* substitution.

Example 1. *Consider the following running example, where x and y are vari-*

PAR-0	$A\|0$	$\equiv$	A	
PAR-A	$A\|(B\|C)$	$\equiv$	$(A\|B)\|C$	
PAR-C	$A\|B$	$\equiv$	$B\|A$	
NEW-0	$\nu n.0$	$\equiv$	0	
NEW-C	$\nu u.\nu v.A$	$\equiv$	$\nu v.\nu u.A$	
NEW-PAR	$A\|\nu u.B$	$\equiv$	$\nu u.(A\|B)$	if $u \notin fn(A) \cup fv(A)$
REPL	$!P$	$\equiv$	$P\|!P$	
REWRITE	$\{^M/_x\}$	$\equiv$	$\{^N/_x\}$	if $M =_E N$
ALIAS	$\nu x.\{^M/_x\}$	$\equiv$	0	
SUBST	$\{^M/_x\}\,\|A$	$\equiv$	$\{^M/_x\}\,\|A\,\langle^M/_x\rangle$	

Figure 4: Structural Equivalence

ables, and c, d, k, l, m and n are names:

$$P_{ex} = \nu k.\nu l.\nu m.\nu d.\,(\{^l/_y\}\,|\mathtt{out}(c, enc(n,k))|\mathtt{out}(d,m)|\mathtt{in}(d,x).\mathtt{out}(c,x))$$

We have $fv(P_{ex}) = \{y\}$, $bv(P_{ex}) = \{x\}$, $fn(P_{ex}) = \{n, c\}$, $bn(P_{ex}) = \{k, l, m, d\}$ *and*

$$\Phi(P_{ex}) = \nu k.\nu l.\nu m.\nu d.\,(\{^l/_y\}\,|0|0|0) \equiv \nu k.\nu l.\nu m.\nu d.\,(\{^l/_y\})\,,$$

thus $dom(P_{ex}) = \{y\}$.

Internal Reduction ($\xrightarrow{\tau}$) is the smallest relation on extended processes closed by structural equivalence (i.e., if $P \xrightarrow{\gamma} P'$ then $Q \xrightarrow{\gamma} Q'$ for all processes $Q \equiv P$ and $Q' \equiv P'$) and application of evaluation contexts (i.e., if $P \xrightarrow{\gamma} P'$ then $C[P] \xrightarrow{\gamma} C[P']$ for all evaluation contexts C) such that the rules in Figure 5 hold. Note that in accordance with the original notations [6], we sometimes omit the labels τ_c, τ_t and τ_e, and write $P \to P'$ for $P \xrightarrow{\gamma} P'$ with $\gamma \in \{\tau_c, \tau_t, \tau_e\}$. We also write $P \to^* P'$ for $P \to \ldots \to P'$. Moreover, let $\mathbf{Int} = \{\tau_c, \tau_t, \tau_e\}$ denote the set of labels corresponding to internal reductions or silent transitions.

COMM	$\mathtt{out}(a,x).P \mid \mathtt{in}(a,x).Q$	$\xrightarrow{\tau_c}$	$P \mid Q$
THEN	$\mathtt{if}\ M = M\ \mathtt{then}\ P\ \mathtt{else}\ Q$	$\xrightarrow{\tau_t}$	P
ELSE	$\mathtt{if}\ M = N\ \mathtt{then}\ P\ \mathtt{else}\ Q$	$\xrightarrow{\tau_e}$	Q
	for all ground terms such that $M \neq_E N$		

Figure 5: Internal Reduction

Interactions of extended processes are described using labeled operational semantics ($\xrightarrow{\alpha}$, see Figure 6), where α can be an input or an output of a channel name or variable of base type[1]. More precisely, for all channel names a, terms M, and variables of base type or names of any type u, let $\mathbf{Act} =$

[1]Disallowing variables of channel type is not a limitation. In outputs, variables are only

$\{\mathtt{in}(a, M), \mathtt{out}(a, u), \nu u.\mathtt{out}(a, u)\}$, denote the set of labels of possible external or *visible* transitions, i.e., $\alpha \in \mathbf{Act}$. The free and bound names and variables fn, bn, fv, bv of a transition α are defined analogously to processes, i.e. we obtain the sets by considering α as a process and then applying the function. By construction we have $\mathbf{Act} \cap \mathbf{Int} = \emptyset$.

IN	$\mathtt{in}(a, x).P \xrightarrow{\mathtt{in}(a,M)} P\{^{M}/_{x}\}$
OUT-ATOM	$\mathtt{out}(a, u).P \xrightarrow{\mathtt{out}(a,u)} P$
OPEN-ATOM	$\dfrac{A \xrightarrow{\mathtt{out}(a,u)} A' \qquad u \neq a}{\nu u.A \xrightarrow{\nu u.\mathtt{out}(a,u)} A'}$
SCOPE	$\dfrac{A \xrightarrow{\alpha} A' \qquad u \text{ does not occur in } \alpha}{\nu u.A \xrightarrow{\alpha} \nu u.A'}$
PAR	$\dfrac{A \xrightarrow{\alpha} A' \quad bv(\alpha) \cap fv(B) = bn(\alpha) \cap fn(B) = \emptyset}{A \mid B \xrightarrow{\alpha} A' \mid B}$
STRUCT	$\dfrac{A \equiv B \qquad B \xrightarrow{\alpha} B' \qquad B' \equiv A'}{A \xrightarrow{\alpha} A'}$

Figure 6: Labeled semantics

Labeled *external transitions* are not closed under evaluation contexts. Note that a term M (except for channel names and variables of base type) cannot be output directly. Instead, we have to assign M to a variable, which can then be output. The condition $bv(\alpha) \cap fv(B) = bn(\alpha) \cap fn(B) = \emptyset$ in rule PAR ensures that a bound name or variable in A can only be output if it does not occur in B freely, as otherwise after the transition the name or variable in A would be the same as in B.

Example 2. *Consider our running example process.*

$$P_{ex} = \nu k.\nu l.\nu m.\nu d.\,(\{^{l}/_{y}\} \,|\mathtt{out}(c, enc(n, k))|\mathtt{out}(d, m)|\mathtt{in}(d, x).\mathtt{out}(c, x))$$

used to represent more complex terms. As we cannot apply functions on names and variables of channel type, a channel variable can only be instantiated using a channel name, which can be output directly.

Using an internal reduction, we can derive the following transition[2]*:*

$$\begin{array}{rcll}
P_{ex} & = & \nu k.\nu l.\nu m.\nu d.\,(\{^{l}/_{y}\}\,|\mathtt{out}(c, enc(n,k))|\mathtt{out}(d,m)|\mathtt{in}(d,x).\mathtt{out}(c,x)) & \\
& \equiv & \nu k.\nu l.\nu m.\nu d.(\{^{l}/_{y}\}\,|\mathtt{out}(c, enc(n,k))|\boldsymbol{\nu x.(\{^{m}/_{x}\})}|\mathtt{out}(d,m)| & \\
& & \mathtt{in}(d,x).\mathtt{out}(c,x)) & \textit{by PAR-0, ALIAS} \\
& \equiv & \nu k.\nu l.\nu m.\nu d.(\{^{l}/_{y}\}\,|\mathtt{out}(c, enc(n,k))|\nu x.\boldsymbol{(}\,\{^{m}/_{x}\}\,|\mathtt{out}(d,\boldsymbol{x})| & \\
& & \mathtt{in}(d,x).\mathtt{out}(c,x)\boldsymbol{)}) & \textit{by NEW-PAR, SUBST} \\
& \xrightarrow{\tau_c} & \nu k.\nu l.\nu m.\nu d.\,(\{^{l}/_{y}\}\,|\mathtt{out}(c, enc(n,k))|\nu x.\,(\{^{m}/_{x}\}\,|\mathtt{out}(c,x))) & \\
& \equiv & \nu k.\nu l.\nu m.\nu d.\,(\{^{l}/_{y}\}\,|\mathtt{out}(c, enc(n,k))|\mathtt{out}(c,\boldsymbol{m})) & \\
& & & \textit{by SUBST, ALIAS, NEW-PAR, PAR-0}
\end{array}$$

Similarly, we can also derive an external transition:

$$\begin{array}{l}
P_{ex} \equiv \nu k.\nu l.\nu m.\nu d.(\{^{l}/_{y}\}\,|\nu z.\,(\{^{enc(n,k)}/_{z}\}\,|\mathtt{out}(c,z))\,|\mathtt{out}(d,m)|\mathtt{in}(d,x).\mathtt{out}(c,x)) \\
\quad \xrightarrow{\nu z.\mathtt{out}(c,z)} \nu k.\nu l.\nu m.\nu d.\,(\{^{l}/_{y}\}\,|\,\{^{enc(n,k)}/_{z}\}\,|\mathtt{out}(d,m)|\mathtt{in}(d,x).\mathtt{out}(c,x))
\end{array}$$

3. Observational Equivalence and Labeled Bisimilarity

The applied π-calculus has two notions of equivalence between processes: *observational equivalence* and *labeled bisimilarity*. They can be used to express strong secrecy or privacy properties. For example, one can express vote secrecy in electronic voting as an observational equivalence between two situations where two voters swap their votes [7, 11]. We now discuss multiple notions of equivalence, which are all defined modulo observational equivalence.

Let $A \Downarrow a$ denote that A can send a message on the channel a, i.e. when $A \to^* C[out(a, M).P]$ for some evaluation context $C[\_]$ that does not bind a. Note that C can include other processes parallel to the hole.

Definition 1 (Observational Equivalence [6])**.** Observational equivalence *($\approx$) is the largest symmetric relation $\mathcal{R}$ between closed extended processes with the same domain such that $A\ \mathcal{R}\ B$ implies:*

1. *if $A \Downarrow a$, then $B \Downarrow a$,*
2. *if $A \to^* A'$, then $B \to^* B'$ and $A'\ \mathcal{R}\ B'$ for some B',*
3. *$C[A]\ \mathcal{R}\ C[B]$ for all closing evaluation contexts $C[\_]$.*

The intuition is that two processes are observationally equivalent if each output or internal transition of one process can be simulated by the other, and this holds for all contexts.

Example 3. *Consider the following processes, where f is a function of arity one:*

$$\begin{array}{rcl}
P_0 & = & \nu a.\mathtt{out}(c,a) \\
P_1 & = & \nu a.\nu d.\,(\mathtt{out}(d,a)\ |\ \mathtt{in}(d,y).\mathtt{out}(c,y)) \\
P_2 & = & \nu a.\nu d.\,(\mathtt{out}(d,a)\ |\ \mathtt{in}(d,y).\mathtt{out}(c,(y,f(y))))
\end{array}$$

[2]Here and throughout the rest of the paper we mark the differences between the steps in **bold** for better readability.

Then we have $P_0 \Downarrow c$, $P_1 \Downarrow c$ and $P_2 \Downarrow c$, however $P_0 \not\approx P_2$ and $P_1 \not\approx P_2$ as P_2 outputs the tuple instead of a single value, which can be tested by a context, for example:

$$C[\_] = \mathtt{in}(c, z).\mathtt{if}\ f(\mathtt{proj}_1(z)) = \mathtt{proj}_z(z)\ \mathtt{then}\ \mathtt{out}(e, z)\ \mathtt{else}\ \mathtt{out}(f, z)|\_$$

Yet we have $P_1 \rightarrow \nu a.\nu d.\mathtt{out}(c, a) \equiv P_0$, hence $P_0 \approx P_1$.

As observational equivalence can be difficult to prove because of the all-quantified context, one often uses *labeled bisimilarity* instead. Labeled bisimilarity is defined using the notion of *static equivalence*, which is based on the equivalence of two terms in a given frame. Note that every frame ϕ can be written as $\nu\tilde{n}.\sigma$ modulo structural equivalence, i.e., using rule NEW-PAR.

Definition 2 (Equivalence in a Frame [6])**.** *Two terms M and N are* equal in the frame ϕ*, written $(M = N)\phi$, if and only if for all names $\tilde{n}$ and every substitution σ such that $\phi \equiv \nu\tilde{n}.\sigma$ and $\tilde{n} \cap (fn(M) \cup fn(N)) = \emptyset$ we have $M\sigma =_E N\sigma$.*

Note that M and N cannot contain variables that are restricted in the frame. In the applied π-calculus, restricted names model values that are unknown to the environment, for example keys or fresh random values. In static equivalence, we only consider terms that do not contain restricted values as free names, because otherwise one could use e.g. a secret key to decrypt a value inside M or N, although the key was never output.

Definition 3 (Static Equivalence ($\approx_s$) [6])**.** *Two closed frames ϕ and ψ are* statically equivalent*, written $\phi \approx_s \psi$, when $dom(\phi) = dom(\psi)$ and when for all terms M and N we have $(M = N)\phi$ if and only if $(M = N)\psi$. Two extended processes A and B are statically equivalent ($A \approx_s B$) if their frames are statically equivalent.*

The intuition behind this definition is that two processes are statically equivalent if the previously output terms cannot be distinguished with respect to the equational theory, i.e., all equalities of terms instantiated with the outputs have the same value (true or false) on both sides. Note that this only concerns what is statically known about the process via the active substitutions, but not the possible interactions: we can have two statically equivalent processes where one can do multiple transitions, but the other one none at all.

Example 4 (from [6])**.** *Consider the following frames, where f and g are two functions of arity one with no equations:*

$$\begin{aligned} \phi_0 &= \nu k.\{^{k}/_{x}\} \mid \nu s.\{^{s}/_{y}\} \\ \phi_1 &= \nu k.\{^{f(k)}/_{x}, {}^{g(k)}/_{y}\} \\ \phi_2 &= \nu k.\{^{k}/_{x}, {}^{f(k)}/_{y}\} \end{aligned}$$

Then $\phi_0 \approx_s \phi_1$, but $\phi_1 \not\approx_s \phi_2$ and $\phi_1 \not\approx_s \phi_2$ as $(f(x) = y)\phi_2$, but neither $(f(x) = y)\phi_0$ nor $(f(x) = y)\phi_1$.

We can then define (weak)[3] labeled bisimilarity.

Definition 4 ((Weak) Labeled Bisimilarity ($\approx_l$) [6])**.** (Weak) labeled bisimilarity *is the largest symmetric relation $\mathcal{R}$ on closed extended processes, such that $A\ \mathcal{R}\ B$ implies:*

1. *$A \approx_s B$,*
2. *if $A \to A'$, then $B \to^* B'$ and $A'\ \mathcal{R}\ B'$ for some B',*
3. *if $A \xrightarrow{\alpha} A'$ and $fv(\alpha) \subseteq dom(A)$ and $bn(\alpha) \cap fn(B) = \emptyset$, then $B \to^* \xrightarrow{\alpha} \to^* B'$ and $A'\ \mathcal{R}\ B'$ for some B'.*

As hinted above, labeled bisimilarity is often easier to prove than observational equivalence since there is no quantification over all contexts. However observational equivalence and labeled bisimilarity do not coincide if active substitutions are allowed on variables of channel type [12, 13], as in that case observational equivalence is not closed under the application of evaluation contexts, as the following example illustrates.

Example 5 (From [13])**.** *Consider $A = \nu c.(\mathtt{out}(c, n).\mathtt{out}(a, n) | \{^c/_x\})$ and $B = \nu c.(0 | \{^c/_x\})$. Then $A \approx_l B$ as A and B are statically equivalent and both have no transitions, but not $A \approx B$ as $A|\mathtt{in}(x, y) \Downarrow a$ but $B|\mathtt{in}(x, y) \not\Downarrow a$.*

Here we restricted active substitutions to variables of base sort (see Section 2). Hence (weak) labeled bisimilarity coincides with observational equivalence [12, 14], and is thus closed under the application of evaluation contexts. Note that this implies also that static equivalence is closed under the application of contexts.

Corollary 1. *For all closed extended processes A and B, and all closing evaluation contexts C, $A \approx_s B$ implies $C[A] \approx_s C[B]$.*

Proof. Suppose $A \approx_s B$. Then $\Phi(A) \approx_l \Phi(B)$ as $\Phi(A) \approx_s \Phi(B)$, but neither $\Phi(A)$ nor $\Phi(B)$ can do a transition. Then for all closing evaluation contexts C we have $C[\Phi(A)] \approx_l C[\Phi(B)]$ as weak observational equivalence is closed under the application of closing evaluation contexts, which implies $C[\Phi(A)] \approx_s C[\Phi(B)]$ and $C[A] \approx_s C[B]$. □

In our work on unique decomposition of processes we also consider a stronger version of labeled bisimilarity.

Definition 5 (Strong Labeled Bisimilarity ($\sim_l$))**.** Strong labeled bisimilarity *is the largest symmetric relation $\mathcal{R}$ on closed extended processes, such that $A\ \mathcal{R}\ B$ implies:*

1. *$A \approx_s B$,*

[3] Originally this notion of bisimilarity was only called "labeled bisimilarity" by Abadi and Fournet [6], however we also call it "weak labeled bisimilarity" to distinguish it from "strong labeled bisimilarity".

2. *if* $A \to A'$, *then* $B \to B'$ *and* $A' \mathcal{R} B'$ *for some* B',
3. *if* $A \xrightarrow{\alpha} A'$ *and* $fv(\alpha) \subseteq dom(A)$ *and* $bn(\alpha) \cap fn(B) = \emptyset$, *then* $B \xrightarrow{\alpha} B'$ *and* $A' \mathcal{R} B'$ *for some* B'.

This notion is stronger than weak labeled bisimilarity in the sense that each step on one side has to be matched by exactly one on the other side, whereas in the case of weak labeled bisimilarity a single transition could be simulated using several (internal) transitions.

Example 6. *Consider again the processes from Example 3, where* f *is a function of arity one:*

$$\begin{array}{lcl} P_0 & = & \nu a.\mathtt{out}(c,a) \\ P_1 & = & \nu a.\nu d.(\mathtt{out}(d,a)|(\mathtt{in}(d,y).\mathtt{out}(c,y))) \\ P_2 & = & \nu a.\nu d.(\mathtt{out}(d,a)|(\mathtt{in}(d,y).\mathtt{out}(c,(y,f(y))))) \end{array}$$

Then we have $P_0 \approx_l P_1$ *as* $P_0 \xrightarrow{\nu a.\mathtt{out}(c,a)} 0$ *and* $P_1 \to \nu a.\nu d.\mathtt{out}(c,a) \xrightarrow{\nu a.\mathtt{out}(c,a)} 0$ *(using STRUCT to rewrite* P_1 *at the beginning and to remove* νd *in the second step) and* $P_1 \to \nu a.\nu d.\mathtt{out}(c,a) \approx_l P_0$.

Yet we neither have $P_0 \approx P_2$ *nor* $P_1 \approx P_2$ *as*

$$\begin{array}{ll} P_2 & \to \nu a.\nu d.\mathtt{out}(c,(a,f(a))) \equiv \nu a.\nu d.\nu z.\,\{^{a,f(a)}/_{z}\}\,\mathtt{out}(c,z) \\ & \xrightarrow{\nu z.\mathtt{out}(c,z)} \nu a.\nu d.\,\{^{(a,f(a))}/_{z}\} \end{array}$$

but neither P_0 *nor* P_1 *can produce a frame that is statically equivalent, i.e. a frame where for example* $f(\mathtt{proj}_1(z)) = \mathtt{proj}_2(z)$ *holds.*

Note also that $P_0 \not\approx_l P_1$ *and* $P_0 \not\approx_l P_2$ *as* $P_0 \xrightarrow{\nu a.\mathtt{out}(c,a)} 0$ *but* P_1 *and* P_2 *cannot do an external transition without a previous internal reduction. Similarly to weak labeled bisimilarity above, we have* $P_1 \not\approx_l P_2$ *as the final frames are not statically equivalent.*

Strong labeled bisimilarity is closed under the application of contexts as the following lemma shows.

Lemma 1. *For all closed extended processes* A *and* B*, and all closing evaluation contexts* C*,* $A \sim_l B$ *implies* $C[A] \sim_l C[B]$.

Proof. Let $\mathcal{R}'$ denote the strong bisimilarity relation of $A \sim_l B$. Consider now the following relation $\mathcal{R}$:

$$\mathcal{R} = \{(C[A], C[B]) | (A, B) \in \mathcal{R}', C \text{ evaluation context}\}$$

We now prove that $\mathcal{R}$ satisfies the three conditions of strong labeled bisimilarity. Let $(P, Q) \in \mathcal{R}$.

1. $P \approx_s Q$ by Corollary 1.

2. Suppose $P = C[A] \to P'$. We need to show that $Q \to Q'$ and $P' \mathcal{R} Q'$ for some Q'.

   If $P \xrightarrow{\tau_t} P'$ or $P \xrightarrow{\tau_e} P'$, we have either $C[A] \xrightarrow{\tau_{\{t,e\}}} C'[A]$ or $A \xrightarrow{\tau_{\{t,e\}}} A'$ as applications of structural equivalence cannot introduce or remove processes of the form $\texttt{if } M = M \texttt{ then } P \texttt{ else } Q$, and A is closed (thus no active substitution in C can influence A).
   If $C[A] \xrightarrow{\tau_{\{t,e\}}} C'[A]$ then $Q = C[B] \xrightarrow{\tau_{\{t,e\}}} C'[B]$ and $(P', Q') \in \mathcal{R}$ as $A \approx_s B$. Static equivalence is important since the transition might depend on active substitutions in A or B.
   If $A \xrightarrow{\tau_{\{t,e\}}} A'$ then $B \to B'$ for some B' by $(A, B) \in \mathcal{R}'$, and thus $Q = C[B] \to C[B']$ and $(P', Q') \in \mathcal{R}$.

   If $P \xrightarrow{\tau_c} P'$, we have to distinguish three cases:

   - If $C[A] \xrightarrow{\tau_c} C'[A]$, then $Q = C[B] \xrightarrow{\tau_c} C'[B]$ and $(P', Q') \in \mathcal{R}$ for $Q' = C'[B]$.
   - If $A \xrightarrow{\tau_c} A'$ then $B \to B'$ for some B' by $(A, B) \in \mathcal{R}'$, and thus $Q = C[B] \to C[B']$ and $(P', Q') \in \mathcal{R}$ for $Q' = C[B']$.
   - If the transition is a consequence of a synchronization between two processes in C and A, i.e. $P = C[A] \xrightarrow{\tau_c} C'[A'] = P'$: In this case we have $A \xrightarrow{\alpha} A'$ for $\alpha = \texttt{in}(a, x)$ or $\alpha = \texttt{out}(a, x)$, as active substitutions cannot be defined on variables of channel type. This prevents that their application using structural equivalence operations makes a transition, which is not available using labeled semantics, available for internal reduction (as in Example 5).

     Thus, by $(A, B) \in \mathcal{R}'$, we have $B \xrightarrow{\alpha} B'$ for some B', and thus $Q = C[B] \xrightarrow{\tau_c} C'[B'] = Q'$ and $(P', Q') \in \mathcal{R}$.

   Note that there cannot be any other case: by the semantics, all labeled transitions have to originate in an $\texttt{in}(a, x)$ and an $\texttt{out}(a, u)$. Since these processes cannot be added or removed by structural equivalence, each such process used in a transition $P \xrightarrow{\tau_c} P'$ must be either part of C or A.

3. Now suppose $P \xrightarrow{\alpha} P'$ and $fv(\alpha) \subseteq \mathrm{dom}(P)$ and $bn(\alpha) \cap fn(Q) = \emptyset$. We need to show that $Q \xrightarrow{\alpha} Q'$ and $P' \mathcal{R} Q'$ for some Q'.

   In a first step, we show that such a transition has to originate either in C or in A: if $C[A] \xrightarrow{\alpha} P'$, then either

   - $C[A] \xrightarrow{\alpha} C'[A]$ or
   - $A \xrightarrow{\alpha} A'$ or
   - $A \xrightarrow{\alpha'} A'$ and $\alpha = \nu u.\alpha'$.

   By the semantics, all labeled transitions have to originate either in an $\texttt{in}(a, x)$ or in an $\texttt{out}(a, u)$. Since these processes cannot be added or

removed by structural equivalence, any such process used in a transition $C[A] \xrightarrow{\alpha} P'$ must be either part of C or A.
If it is part of C, it is easy to see that $C[A] \xrightarrow{\alpha} C'[A]$ holds: by the semantics, only restrictions might influence the transition, and by structural equivalence restrictions from A can only be extended around processes in C if the restricted values are not free in C.
If it is part of A, there are two cases: either no restrictions in C are relevant, and we have $A \xrightarrow{\alpha} A'$. Or, we have an output of a value restricted in C, in which case we have $A \xrightarrow{\alpha'} A'$ and $\alpha = \nu u.\alpha'$.
Note that we have $fv(\alpha) \subseteq \mathrm{dom}(A)$ and $fv(\alpha') \subseteq \mathrm{dom}(A)$, respectively: suppose there exists a free variable x in α or α', respectively, which is not in the domain of A. As A is closed and x occurs in A, it must thus be bound. As x cannot be bound using an `in` (then the transition would not be possible according to the semantics), x can only be bound using a restriction. But then, according to the semantics, the restriction must also be part of α or α', contradicting the assumption that x was free.

Now we consider the above cases:

- If $C[A] \xrightarrow{\alpha} C'[A]$ then also $Q = C[B] \xrightarrow{\alpha} C'[B] = Q'$, and $(P', Q') \in \mathcal{R}$.
- If $A \xrightarrow{\alpha} A'$ then $B \xrightarrow{\alpha} B'$ for some B' by $(A, B) \in \mathcal{R}'$, and thus $Q = C[B] \xrightarrow{\alpha} C[B'] = Q'$ with $(P', Q') \in \mathcal{R}$.
- If $A \xrightarrow{\alpha'} A'$ and $\alpha = \nu u.\alpha'$ then $P = C[A] \xrightarrow{\alpha} C'[A']$. Moreover, $B \xrightarrow{\alpha'} B'$ for some B' by $(A, B) \in \mathcal{R}'$, and thus $Q = C[B] \xrightarrow{\alpha} C'[B'] = Q'$ with $(P', Q') \in \mathcal{R}$.

□

Restrictions can only forbid transitions, but not create new ones, as the following lemma shows.

Lemma 2. *Let A be a closed extended process and $X \subseteq dom(A)$. Then $\nu X.A \xrightarrow{\mu} \nu X.A'$ implies $A \xrightarrow{\mu'} A'$ where μ can be a silent or a visible transition, and we have either*

- $\mu' = \mu$ *or*
- *for $x \in X$ and $\mu = \nu x.\mathtt{out}(\mathtt{a}, \mathtt{x})$, $\mu' = \mathtt{out}(\mathtt{a}, \mathtt{x})$*

Proof. Suppose $\nu X.A \xrightarrow{\mu} \nu X.A'$. μ can be the result of a SCOPE or a OPEN-ATOM rule. If μ is the result of a SCOPE-rule, we have $A \xrightarrow{\mu} A'$. Otherwise we have $\mu = \nu x.\mathtt{out}(\mathtt{a}, \mathtt{x})$, and $A \xrightarrow{\mu} A'$ for $\mu' = \mathtt{out}(\mathtt{a}, \mathtt{x})$. □

4. Depth and Norm of Processes

In the following we prove unique decomposition results for different subsets of processes, namely finite and normed processes. This requires to formally define the length of process traces.

The *visible depth* is defined as the length of the longest trace of visible actions, i.e. labeled transitions, not counting internal reductions. Note that this may be infinite for processes including replication. We write $P \not\rightarrow$ if P cannot execute a transition, and $P \xrightarrow{\mu_1\mu_2\ldots\mu_n} P'$ for $P \xrightarrow{\mu_1} P_1 \xrightarrow{\mu_2} P_2 \xrightarrow{\mu_3} \ldots \xrightarrow{\mu_n} P'$. Moreover, we denote by ϵ the empty word, and by ab the concatenantion of traces a and b.

Definition 6 (Visible Depth)**.** *Let* $length_v : (\mathbf{Act} \cup \mathbf{Int})^* \to \mathbb{N}$ *be the function defined by* $length_v(\epsilon) = 0$ *and* $length_v(\mu w) = \begin{cases} 1 + length_v(w) & \text{if } \mu \in \mathbf{Act} \\ length_v(w) & \text{otherwise} \end{cases}$
Then the visible depth $|P|_v \in (\mathbb{N} \cup \{\infty\})$ *of a closed process* P *is defined as follows:*

$$|P|_v = \sup \left\{ length_v(w) : P \xrightarrow{w} P', w \in (\mathbf{Act} \cup \mathbf{Int})^* \right\}$$

The *total depth* is defined as the length of the longest trace of actions (including internal reductions).

Definition 7 (Total Depth)**.** *Let* $length_t : (\mathbf{Act} \cup \mathbf{Int})^* \to \mathbb{N}$ *be the function defined by* $length_t(\epsilon) = 0$ *and* $length_t(\mu w) = 1 + length_t(w)$*. The* total depth $|P|_t \in (\mathbb{N} \cup \{\infty\})$ *of a closed process* P *is defined as follows:*

$$|P|_t = \sup \left\{ length_t(w) : P \xrightarrow{w} P', w \in (\mathbf{Act} \cup \mathbf{Int})^* \right\}$$

The *norm* of a process is defined as the length of the shortest complete trace, including internal reductions, where communications are counted as two. This is to ensure that the norm of $P|Q$ is the sum of the norm of P and the norm of Q.

Definition 8 (Number of internal communications)**.** *Let* $c_{\tau_c}(t) : (\mathbf{Act} \cup \mathbf{Int})^* \to \mathbb{N}$ *be the function defined by* $c_{\tau_c}(\epsilon) = 0$ *and* $c_{\tau_c}(\mu w) = \begin{cases} 1 + c_{\tau_c}(w) & \text{if } \mu = \tau_c \\ c_{\tau_c}(w) & \text{otherwise} \end{cases}$
Moreover, let $min_{\tau_c}(P) = \inf \left\{ c_{\tau_c}(w) : P \xrightarrow{w} P' \not\rightarrow, w \in (\mathbf{Act} \cup \mathbf{Int})^* \right\}$

Definition 9 (Norm of a Process)**.** *Let* $length_n : (\mathbf{Act} \cup \mathbf{Int})^* \to \mathbb{N}$ *be the function defined by* $length_n(\epsilon) = 0$ *and* $length_n(\mu w) = \begin{cases} 1 + length_n(w) & \text{if } \mu \neq \tau_c \\ 2 + length_n(w) & \text{if } \mu = \tau_c \end{cases}$
The norm $\mathcal{N}(P) \in (\mathbb{N} \cup \{\infty\})$ *of a closed process* P *is defined as follows:*

$$\mathcal{N}(P) = \inf \left\{ length_n(w) : P \xrightarrow{w} P' \not\rightarrow, w \in (\mathbf{Act} \cup \mathbf{Int})^*, c_{\tau_c}(w) = min_{\tau_c}(P) \right\}$$

Example 7. *Consider the processes from our running example (Example 1). We have* $|P_{ex}|_v = 2$, $|P_{ex}|_t = 3$ *and* $\mathcal{N}(P_{ex}) = 4$.

The above definitions admit some simple properties.

Lemma 3. *For all closed extended processes* P, Q *and* R *we have*

1. $|P|_v \leq |P|_t$
2. $P = Q|R$ *implies* $|P|_v = |Q|_v + |R|_v$
3. $P = Q|R$ *implies* $|P|_t = |Q|_t + |R|_t$
4. $P = Q|R$ *implies* $\mathcal{N}(P) = \mathcal{N}(Q) + \mathcal{N}(R)$
5. $P = Q|R$ *implies* $|dom(P)| = |dom(Q)| + |dom(R)|$
6. $P \approx_l Q$ *implies* $|P|_v = |Q|_v$
7. $P \sim_l Q$ *implies* $|P|_t = |Q|_t$
8. $P \sim_l Q$ *implies* $\mathcal{N}(P) = \mathcal{N}(Q)$

Proof. See Appendix A. □

Now we define two important subclasses of processes: *finite* processes, i.e. processes with a finite longest complete trace, and *normed* processes, i.e. processes with a finite shortest complete trace.

Definition 10 (Finite and normed processes)**.** *A closed process* P *is called* finite *if* $|P|_t$ *is finite (which implies* $|P|_v$ *is finite). A closed process* P *is called* normed *if* $\mathcal{N}(P)$ *is finite.*

It is easy to see that every finite process is normed, but not all normed processes are finite, as the following example illustrates.

Example 8. *Consider* $P = \nu a.(out(a, m)|(in(a, x).(!in(b, y)))|in(a, x))$. *Note that this process can choose between two possible behaviors: we have* $P \to P' \sim_l 0$ *(hence* P *is normed), but also* $P \to P'' \sim_l !in(b, y)$ *(which has infinite traces). Hence* P *is normed, but not finite.*

It is also clear that not all processes are normed. Consider the following example.

Example 9. *Consider* $P = !(\nu x.out(c, x))$. *It is easy to see that for no sequence of transitions* s *we have* $P \xrightarrow{s} P' \not\to$, *i.e.* P *has no finite traces.*

Note however that all processes without replication ("!") are finite, as no other syntactic element allows to construct infinite traces.

5. Decomposition w.r.t. Strong Labeled Bisimilarity

We begin with the simpler case of strong labeled bisimilarity. Note that $P \sim_l Q$ implies $|P|_t = |Q|_t$ and $\mathcal{N}(P) = \mathcal{N}(Q)$ for all closed processes P and Q by Lemma 3.

We define strong parallel primeness as follows: a process is *prime* if it cannot be decomposed into non-trivial subprocesses (w.r.t. strong labeled bisimilarity). We require the processes to be closed, which is necessary as our bisimulation relation is only defined on closed processes.

Definition 11 (Strongly Parallel Prime)**.** *A closed process P is* strongly parallel prime *if*

- $P \not\sim_l 0$ *and*
- *for all two closed processes Q and R such that $P \sim_l Q|R$, we have $Q \sim_l 0$ or $R \sim_l 0$.*

Example 10. *Consider our running example:*

$$P_{ex} = \nu k.\nu l.\nu m.\nu d.\,(\{^{l}/_{y}\} \mid \mathtt{out}(c, enc(n,k)) \mid \mathtt{out}(d,m) \mid \mathtt{in}(d,x).\mathtt{out}(c,x))$$

We can decompose P_{ex} as follows:

$$P_{ex} \sim_l \nu l.\,\{^{l}/_{y}\} \mid \nu k.\mathtt{out}(c, enc(n,k)) \mid \nu d.(\nu m.\mathtt{out}(d,m)|\mathtt{in}(d,x).\mathtt{out}(c,x))$$

Suppose the first factor $S_1 = \nu l.\,\{^{l}/_{y}\}$ was not prime. Since it allows for no transitions and only defines a variable in the frame, all decompositions of S_1 must be composed of factors consisting of active substitutions and restrictions (modulo $\sim_l$). However, as we cannot have two substitutions defining the same variable, and S_1 defines only one variable, no such decomposition is possible. Hence S_1 is prime.

It is easy to see that the second factor $S_2 = \nu k.\mathtt{out}(c, enc(n,k))$ is prime, as it can only perform one external transition, and has an empty domain. Note that every process with empty domain which cannot do any transition is equivalent to 0 (see also Lemma 4 below).

The third factor

$$S_3 = \nu d.(\nu m.\mathtt{out}(d,m)|\mathtt{in}(d,x).\mathtt{out}(c,x))$$

can only do two transitions, namely $S_3 \to \nu m.\mathtt{out}(c,m) \xrightarrow{\nu m.\mathtt{out}(c,m)} 0$. Suppose S_3 was was not prime, and we could decompose it into two factors, i.e. such that $S_3 \sim_l S_3'|S_3''$. Such a decomposition would imply that both factors can execute at least one transition each – otherwise they would be equivalent to 0 as they have an empty domain, since S_3 has an empty domain (again, see Lemma 4 below). However in that case the transitions of $S_3'|S_3''$ can be executed in every order, whereas in S_3 we have to start with the internal reduction. Hence no such decomposition exists, and S_3 is prime.

Remark 1. *Note also that within a prime factor we can recursively apply the decomposition as our notion of bisimilarity is closed under the application of contexts. For example if we have a prime factor $P = \nu a.P'$, we can bring P' into normal form, i.e. $P' \sim_l P_1'|...|P_n'$, and rewrite $P = \nu a.P'$ as $P \sim_l \nu a.(P_1'|...|P_n')$.*

It is clear that not all processes can be written as a unique decomposition of parallel primes according to our definition.

Example 11. *Consider $!P$ for a process $P \not\sim_l 0$. By definition we have $!P = P|!P$, hence $!P$ is not prime. At the same time every such decomposition contains again $!P$, a non-prime factor, which needs to be decomposed again. Thus there is no decomposition into prime factors.*

However we can show that every closed normed process has a unique decomposition with respect to strong labeled bisimilarity. To achieve this, we need some preliminary lemmas about transitions and the domain of processes. The first lemma captures the fact that every process that cannot perform a transition and has an empty domain, is bisimilar to 0 (the empty process).

Lemma 4. *For every closed process A with $dom(A) = \emptyset$ and $\mathcal{N}(A) = 0$, we have $A \sim_l 0$.*

Proof. Consider the relation $\mathcal{R} = \{(A, 0), (0, A)\}$. We show that it fulfils the conditions of strong labeled bisimilarity:

1. We have $dom(A) = \emptyset = dom(0)$, hence $A \approx_s 0$.
2. Let $(A, 0) \in \mathcal{R}$. Obviously 0 cannot do a transition. Since $\mathcal{N}(A) = 0$, there exists a complete trace of length 0. Thus we have $A \not\rightarrow$, i.e. A cannot do a transition either and the remaining conditions are trivially satisfied. The same is true for $(0, A) \in \mathcal{R}$.

As we have $(A, 0) \in \mathcal{R}$, this gives $A \sim_l 0$, which we wanted to show. □

We also need to show that if a normed process can execute a transition, it can also execute a norm-reducing transition.

Lemma 5. *Let A be a closed normed process with $A \xrightarrow{\mu} A'$ where μ is an internal reduction or visible transition. Then $A \xrightarrow{\mu'} A''$ with $\mathcal{N}(A'') < \mathcal{N}(A)$.*

Proof. As A is normed, we have $\infty > \mathcal{N}(A)$. Moreover, $A \xrightarrow{\mu} A'$ implies $\mathcal{N}(A) > 0$, as this transition contradicts a complete trace of length 0. Hence the shortest complete trace w satisfies $\infty > length_n(w) > 0$. Hence there is a transition μ' with $w = \mu' w'$ which reduces norm, i.e. $A \xrightarrow{\mu'} A''$ with $\mathcal{N}(A'') < \mathcal{N}(A)$. □

In a first step, we prove the existence of a decomposition.

Theorem 1 (Existence of Decomposition)**.** *Every closed normed process P can be expressed as the parallel composition of strong parallel primes, i.e., $P \sim_l P_1 | \ldots | P_n$ where for all $1 \leq i \leq n$, P_i is strongly parallel prime.*

Proof. By induction on the norm of P, and on the size of the domain $dom(P)$.

- If $\mathcal{N}(P) = 0$:
  - If $|dom(P)| = 0$, then $P \sim_l 0$ (by Lemma 4), hence the decomposition is the empty decomposition.
  - If $|dom(P)| > 0$, then $P \not\sim_l 0$, hence P is either strongly parallel prime itself (in which case we are done), or it can be written as $P \sim_l Q|R$, by the definition of strongly parallel prime. As we have $dom(P) = dom(Q) \cup dom(R)$ with $dom(Q) \cap dom(R) = \emptyset$ and $|dom(Q)| > 0$, $|dom(R)| > 0$ (since $Q \not\sim_l 0$ and $R \not\sim_l 0$), we have $|dom(Q)| < |dom(P)|$, $|dom(R)| < |dom(P)|$, hence we

can use the induction hypothesis. Using the induction hypothesis we know that Q and R can be expressed as the parallel composition of strong parallel primes, i.e., we have $Q \sim_l Q_1|\ldots|Q_n$ and $R \sim_l R_1|\ldots|R_n$ where all Q_i and R_i are parallel prime. Hence we have $P = Q_1|\ldots|Q_n|R_1|\ldots|R_n$, which is a parallel composition of strong parallel primes.

- If $\mathcal{N}(P) > 0$:
  - Suppose $|dom(P)| = 0$: P is either strongly parallel prime itself, or can be written as $P \sim_l Q|R$. Then we have $dom(P) = dom(Q) = dom(R) = \emptyset$, and $\mathcal{N}(Q) > 0$, $\mathcal{N}(R) > 0$ by Lemma 4, hence $\mathcal{N}(Q) < \mathcal{N}(P)$, $\mathcal{N}(R) < \mathcal{N}(P)$ by Lemma 3. Using the induction hypothesis we can conclude as above.
  - If $|dom(P)| > 0$, then $P \not\sim_l 0$, hence P is either strongly parallel prime itself, or can be written as $P \sim_l Q|R$. Suppose $\mathcal{N}(Q) > 0$ and $\mathcal{N}(R) > 0$, hence $\mathcal{N}(Q) < \mathcal{N}(P)$, $\mathcal{N}(R) < \mathcal{N}(P)$ and we can apply the induction hypothesis. Suppose w.l.o.g. $\mathcal{N}(Q) = 0 < \mathcal{N}(P)$, then $\mathcal{N}(R) = \mathcal{N}(P)$ by Lemma 3. Since $Q \not\sim_l 0$ this implies $|dom(Q)| > 0$ by Lemma 4, hence $|dom(R)| < |dom(P)|$, and we can use the induction hypothesis to conclude as above.

□

We now show the uniqueness of the decomposition. In a first lemma, we show that the decomposition of all processes with zero norm is unique. As an additional notation, let $\exp(A, R)$ denote the exponent (i.e. the number of occurrences) of prime A in the unique decomposition[4] of R.

Lemma 6 (Uniqueness of Decomposition for Processes with Zero Norm)**.** *The strong parallel decomposition of a closed normed process P with $\mathcal{N}(P) = 0$ is unique up to $\sim_l$ and permutation of the prime factors.*

Proof. By induction on the size of the domain $dom(P)$.

- If $|dom(P)| = 0$, then $P \sim_l 0$ (by Lemma 4), hence the decomposition is the unique empty decomposition. Note that by Lemma 3 any decomposition into factors would imply that these factors also have norm 0 and an empty domain, hence they would also be bisimilar to 0 by Lemma 4.
- If $|dom(P)| > 0$, then $P \not\sim_l 0$. Suppose P is in its decomposition form, and we have a second, different decomposition Q with $P \sim_l Q$:

$$\begin{aligned} P &= A_1^{k_1}|A_2^{k_2}|\ldots|A_n^{k_n} \\ Q &= A_1^{l_1}|A_2^{l_2}|\ldots|A_n^{l_n} \end{aligned}$$

[4]This notation only makes sense if we know that R has a unique decomposition, which however holds in the cases where we employ it during the proof and later on.

where the A_i's are distinct (i.e. for $i \neq j$ we have $A_i \not\sim_l A_j$) and $k_i \geq 0$, $l_i \geq 0$ (w.l.o.g. we can rewrite P and Q this way).

Note that since all factors A_i are prime we have $\forall i\ A_i \not\sim_l 0$, and since we also know $\mathcal{N}(P) = 0$ we have $\forall i\ \mathcal{N}(A_i) = 0$ by Lemma 3. By Lemma 4 we then have $dom(A_i) \neq \emptyset$, which implies $k_i, l_i \leq 1$ as we cannot have two substitutions defining the same variable.

Let m be such that $k_m \neq l_m$. Without loss of generality we assume $1 = k_m > l_m = 0$.

Obviously we have $dom(P) = dom(Q)$. Let $\tilde{v} = dom(P) \setminus dom(A_m)$. Then we have (by Lemmas 3 and 4 and rules NEW-PAR, PAR-0, PAR-C and PAR-A):

$$\nu\tilde{v}.P \equiv A_m | \nu\tilde{v}.P' \sim_l A_m$$

where P' is P without the factor A_m, since $\mathcal{N}(\nu\tilde{v}.P') = 0$ and $dom(\nu\tilde{v}.P') = \emptyset$ by Lemma 3. Similarly

$$\nu\tilde{v}.Q \equiv |_{i \in I}\ \nu\tilde{v}_i.A_i \mid |_{i \notin I}\ \nu\tilde{v}_i.A_i^{l_i} \sim_l |_{i \in I}\ \nu\tilde{v}_i.A_i$$

where $I = \{i | dom(A_i) \cap dom(A_m) \neq \emptyset \text{ and } l_i = 1\}$, and where $\tilde{v}_i = dom(A_i) \cap \tilde{v}$.

By $\nu\tilde{v}.P \sim_l \nu\tilde{v}.Q$ we have $A_m \sim_l |_{i \in I}\ \nu\tilde{v}_i.A_i$. If $|I| = 0$, we have $A_m \sim_l 0$ which contradicts the hypothesis that A_m is prime. Similarly for $|I| > 1$, we have a decomposition for A_m into several processes, which also contradicts A_m prime.

For $|I| = 1$ we have the following cases: Let i denote the only index in I. If $\tilde{v}_i = \emptyset$, we have a contradiction to the distinctness hypothesis of the A_j's since $A_m \sim_l A_i$ with $m \neq i$ as $l_m = 0 \neq l_i = 1$.

If $\tilde{v}_i \neq \emptyset$ we have $dom(A_m) \subset dom(A_i)$, but $dom(A_m) \neq dom(A_i)$. Now consider $\tilde{v}' = dom(Q) \setminus dom(A_i)$. Then - as above - we have:

$$\nu\tilde{v}'.Q \equiv A_i | \nu\tilde{v}'.Q' \sim_l A_i$$

where Q' is Q without the factor A_i. Similarly

$$\nu\tilde{v}'.P \equiv |_{j \in I'}\ \nu\tilde{v}'_j.A_j \mid |_{j \notin I'}\ \nu\tilde{v}'_j.A_j^{l_j} \sim_l |_{j \in I'}\ \nu\tilde{v}'_j.A_j$$

where $I' = \{j | dom(A_j) \cap dom(A_i) \neq \emptyset \text{ and } k_j = 1\}$ and $\tilde{v}'_j = dom(A_j) \cap \tilde{v}'$. By $dom(A_m) \subset dom(A_i)$ we have $m \in I'$, but also $dom(A_m) \neq dom(A_i)$ and $dom(A_i) = dom(|_{j \in I'}\ \nu\tilde{v}'_j.A_j)$. This gives us $|I'| > 1$ as there must be other factors than m to cover the entire domain, hence $A_i \sim_l |_{j \in I'}\ \nu\tilde{v}'_j.A_j$ gives a decomposition of A_i, which contradicts the hypothesis that it is prime.

□

We now show the uniqueness for all normed processes.

Theorem 2 (Uniqueness of Decomposition)**.** *The strong parallel decomposition of a closed normed process P is unique up to $\sim_l$ and permutation of the prime factors.*

Proof. By induction on $\mathcal{N}(P)$, and on the size of the domain $dom(P)$. In the case $\mathcal{N}(P) = 0$, we are done by Lemma 6.

Case $\mathcal{N}(P) > 0$:

- If $|dom(P)| = 0$: Suppose P is in its decomposition form, and we have a second, different decomposition Q with $P \sim_l Q$:

$$\begin{aligned} P &= A_1^{k_1} | A_2^{k_2} | \ldots | A_n^{k_n} \\ Q &= A_1^{l_1} | A_2^{l_2} | \ldots | A_n^{l_n} \end{aligned}$$

  where the A_i's are distinct (i.e. for $i \neq j$ we have $A_i \not\sim_l A_j$) and $k_i \geq 0$, $l_i \geq 0$ (w.l.o.g. we can rewrite P and Q this way).

  By induction hypothesis for every process R with $\mathcal{N}(R) < \mathcal{N}(P)$ the decomposition is unique.

  Let m be such that $k_m \neq l_m$, and that $\mathcal{N}(A_j) > \mathcal{N}(A_m)$ implies $k_j = l_j$ (i.e. A_m has the maximal norm among the factors in which P and Q differ). Without loss of generality we assume $k_m > l_m$.

  We now analyze different cases:

  - If $P = A_m^{k_m}$, i.e. P is the power of a prime:
    By assumption, A_m is the maximal (w.r.t. norm) prime factor in which P and Q differ. Then Q cannot contain any prime factor A_r ($r \neq m$) with a greater norm than A_m. Now, if $k_m = 1$ (i.e., P is prime), then Q is prime as well. It follows that Q is bisimilar to A_m and hence $k_m = 1 = l_m$, contradicting our assumption that $k_m > l_m$.
    If $k_m > 1$:
    - Assume $l_m = 0$. As $k_m > 1$ we have $dom(A_m) = \emptyset$, as we cannot have multiple substitutions for the same variables. As A_m is prime, we must have a transition μ such that $A_m \xrightarrow{\mu} R$, $P \xrightarrow{\mu} P'$ with $\exp(A_m, P') = k_m - 1 > 0$. Moreover, we also have $\mathcal{N}(P') < \mathcal{N}(P)$ by Lemma 5.
      If possible, we choose $\mu \neq \tau_c$. Then, since $P \sim_l Q$, there exists a Q' with $Q \xrightarrow{\mu} Q'$. For every such Q' we have $\exp(A_m, Q') = 0$ since $l_m = 0$, and Q cannot contain any prime factor with greater norm that A_m, i.e. $l_r = 0$ for all A_r with $\mathcal{N}(A_r) > \mathcal{N}(A_m)$. As P' and Q' have a unique prime decomposition by induction hypothesis, we have a contradiction with $\exp(A_m, P') = k_m - 1 > 0 = \exp(A_m, Q')$.
      If no norm-reducing $\mu \neq \tau_c$ exists, we take a norm-reducing $\mu = \tau_c$. Then, since $P \sim_l Q$, there exists a Q' with $Q \xrightarrow{\mu} Q'$. μ must be simulated by one of the factors inside Q, and not by

a synchronization of two factors, as the latter would imply the existence of a $\mu \neq \tau_c$ (also norm-reducing, as it was part of a norm-reducing τ_c), contradicting the assumption that no $\mu \neq \tau_c$ exists. Thus we can conclude as before.

* Hence assume $l_m > 0$: If $A_m \xrightarrow{\mu} R$ with $\mathcal{N}(R) < \mathcal{N}(A_m)$ for $\mu \neq \tau_c$, we have $Q \xrightarrow{\mu} Q'$ and since $P \sim_l Q$ there exists P' with $P \xrightarrow{\mu} P'$. We have $\exp(A_m, P') \geq k_m - 1 > l_m - 1 = \exp(A_m, Q')$ which contradicts $P \sim_l Q$ using the induction hypothesis.
If no such transition μ exists, we have $A_m \xrightarrow{\tau_c} R$, hence $Q \xrightarrow{\tau_c} Q'$ and since $P \sim_l Q$ there exists P' with $P \xrightarrow{\tau_c} P'$ and such that $P' \sim_l Q'$. We know that P cannot simulate this transition using synchronization between the different copies of A_m as this would imply the existence of a visible norm-reducing transition μ (as the transition τ_c is norm-reducing as well). Hence we have again $\exp(A_m, P') \geq k_m - 1 > l_m - 1 = \exp(A_m, Q')$. Moreover we have $P' \sim_l Q'$, and by the induction hypothesis P' and Q' have the same unique decomposition, contradicting $\exp(A_m, P') > \exp(A_m, Q')$.

– If there exists $j \neq m$ such that $k_j > 0$:

Let μ, T be such that $P \xrightarrow{\mu} T$ and $\mathcal{N}(T) < \mathcal{N}(P)$ and for all ν such that $P \xrightarrow{\nu} P'$ with $\mathcal{N}(P') < \mathcal{N}(P)$ we have $\exp(A_m, P') \leq \exp(A_m, T)$. We now show that such μ, T exist. Note that because of Lemma 3 and $\mathcal{N}(P) < \infty$ we have $\mathcal{N}(A_i) < \infty$ for all i with $k_i > 0$.

This gives that if $A_i \xrightarrow{\mu} A_i'$ then $A_i \xrightarrow{\mu'} A_i''$ with $\mathcal{N}(A_i'') < \mathcal{N}(A_i)$ by Lemma 5. Suppose no such μ, T exist. Hence for no A_i with $k_i > 0, i \neq m$ we have $A_i \xrightarrow{\mu} A_i'$, otherwise this would allow a transition that would fulfill the above conditions. Hence (by Lemma 4) we have $dom(A_i) \neq \emptyset$ for every i with $k_i > 0, i \neq m$, which contradicts $|dom(P)| = 0$. Hence such μ, T exist.

Note that $\exp(A_m, T) \geq k_m$, as every transition by a factor different from A_m does not decrease the number of A_m's.

* Assume a μ and T satisfying the requirements above. If possible, we select a $\mu \neq \tau_c$, then - as $P \sim_l Q$ - there exists Q' with $Q \xrightarrow{\mu} Q'$ and $Q' \sim_l T$. Hence $\mathcal{N}(Q') < \mathcal{N}(Q)$ and there exists A_t with $A_t \xrightarrow{\mu} R$.
If $\mathcal{N}(A_t) \leq \mathcal{N}(A_m)$, then we have $\exp(A_m, Q') \leq l_m < k_m \leq \exp(A_m, T)$, which gives the contradiction to the induction hypothesis.
If $\mathcal{N}(A_t) > \mathcal{N}(A_m)$ then $\exp(A_m, Q') = l_m + \exp(A_m, R)$, $t \neq m$ and $k_t = l_t > 0$ as A_m is maximal. Consider now

$$P \xrightarrow{\mu} P' = A_1^{k_1} | \ldots | A_t^{k_t - 1} | \ldots | A_n^{k_n} | R$$

with $\exp(A_m, P') = k_m + \exp(A_m, R)$. Hence $\exp(A_m, Q') = l_m +$

$\exp(A_m, R) < k_m + \exp(A_m, R) = \exp(A_m, P') \leq \exp(A_m, T)$. This contradicts $Q' \sim_l T$ using the induction hypothesis.

* If $\mu = \tau_c$ for all μ and T satisfying the requirements above, we distinguish two different cases: If the transition is matched by only one factor, we can argue as above. If the transition is matched by the synchronization of two factors ($A_r \xrightarrow{\alpha} A'_r$ and $A_s \xrightarrow{\bar{\alpha}} A'_s$, where α is an output and $\bar{\alpha}$ is an input), this implies that we have two visible actions on two different factors, and at least one of them (w.l.o.g. α) is norm-reducing. As all transitions that maximize the number of A_m's are τ_c-transitions, α can only be matched by A_m, thus $A_m \xrightarrow{\alpha} A'_m$. Hence $Q \xrightarrow{\alpha} Q'$ with $\exp(A_m, Q') = l_m - 1$ and for all $P \xrightarrow{\alpha} P'$ we have $\exp(A_m, P') \geq k_m - 1 > l_m - 1 = \exp(A_m, Q')$, which contradicts the induction hypothesis using $P' \sim_l Q'$.

- If $|dom(P)| > 0$: This is essentially the same proof as above. In the first case (P is a power of a prime), we only have to consider the case $k_m = 1$ as $dom(A_m) \neq \emptyset$. Hence P is prime, and then Q is prime as well, and since $1 = k_m > l_m$ we have $Q \sim_l A_j$ for some $j \neq m$, which gives $A_m \sim_l A_j$, which contradicts the distinctness of the prime factors.

  In the second case we have to be more careful when proving that μ and T with the desired properties exist. Once again, we suppose that they do not exist, hence for no A_i with $k_i > 0, i \neq m$ we have $A_i \xrightarrow{\mu} A'_i$, otherwise this would allow a transition that would fulfill the conditions. Hence for every i with $k_i > 0$, $i \neq m$ we have $\mathcal{N}(A_i) = 0$, and by Lemma 4 we have $dom(A_i) \neq \emptyset$. Let $\tilde{v} = dom(P) \setminus dom(A_m)$ and consider

  $$\nu\tilde{v}.P \equiv A_m^{k_m} | \nu\tilde{v}.P' \sim_l A_m^{k_m}$$

  where P' is P without the factor $A_m^{k_m}$. Similarly

  $$\nu\tilde{v}.Q \equiv |_{1 \leq i \leq n} \nu\tilde{v}_i.A_i^{l_i}$$

  where $\tilde{v}_i = dom(A_i) \cap \tilde{v}$.

  As $|dom(A_m^{k_m})| < |dom(P)|$ by induction hypothesis the decomposition of $A_m^{k_m}$ is unique. We cannot have $\tilde{v}_i = \emptyset$ for some $i \neq m$, as this contradicts the uniqueness of the decomposition as $A_i \not\sim_l A_m$. Hence $dom(A_i) \neq \emptyset$ and $l_i = 1$.

  As A_m is prime, for all $i \neq m$ with $l_i > 0$ we have that $\nu\tilde{v}_i.A_i \sim_l A_m | R_i$ for some R_i. More precisely, we have $\nu\tilde{v}_i.A_i \sim_l A_m^l$ for some $l \geq 1$, as every other decomposition of R_i would contradict the primeness of A_m as we could rewrite $A_m^{k_m}$ using the R_i. In fact, since A_m is the biggest (w.r.t. to norm) factor in which P and Q differ, and there are no bigger factors in P, all the factors in Q must have smaller norm. By Lemmas 2 and 4, this gives $l = 1$.

We cannot have $Q \sim_l A_i$ as this would directly give a decomposition of A_i. Hence there has to be another factor A_r which – by Lemma 4 – has either $dom(A_r) \neq \emptyset$ or can execute a norm-reducing transition (or both).

- If $dom(A_r) \neq \emptyset$, consider $\tilde{v}' = dom(Q) \setminus dom(A_i)$. Then – as above – we have:

  $$Q_1 = \nu\tilde{v}'.Q \equiv A_i|\nu\tilde{v}'.Q' \sim_l \nu\tilde{v}'.P = P_1$$

  where Q' is Q without the factor A_i.
- If $A_r \xrightarrow{\eta} A_r'$, we have

  $$Q \sim_l A_i|A_r|S \xrightarrow{\eta} A_i|A_r'|S = Q_1$$

  where S is Q without A_i and A_r. By $P \sim_l Q$ there exists P_1 with $P \xrightarrow{\eta} P_1 \sim_l A_i|A_r'|S$.

In both cases, we have a unique decomposition of Q_1 and P_1 by induction hypothesis. Additionally $\exp(A_i, Q_1) = 1$ (since $dom(A_i) \neq \emptyset$), and by the uniqueness of the decomposition $\exp(A_i, P_1) = \exp(A_i, Q_1) = 1$. Let s be such that $dom(A_s) \cap dom(A_i) \neq \emptyset$, $k_s > 0$. Such s exists because of $dom(A_m) \subsetneq dom(A_i)$ and $dom(P) = dom(Q)$. Then by hypothesis A_s cannot do a transition, and $\exp(A_s, P_1) = \exp(A_s, P) = 1$, which contradicts $\exp(A_i, P_1) = 1$ because of the conflicting domains.

Hence μ and T with the desired properties exist, and the rest of the proof is the same as above.

□

As a direct consequence, we have the following cancellation result.

Lemma 7 (Cancellation Lemma)**.** *For all closed normed processes A, B and C, we have*

$$A|C \sim_l B|C \Rightarrow A \sim_l B$$

Proof. As A, B and C are closed and normed, there exists a unique parallel decomposition for each of them, i.e.

$$\begin{array}{ll} A \sim_l & A_1|\ldots|A_k, \\ B \sim_l & B_1|\ldots|B_l \text{ and} \\ C \sim_l & C_1|\ldots|C_m. \end{array}$$

Thus we have

$$\begin{array}{ll} A|C \sim_l & A_1|\ldots|A_k|C_1|\ldots|C_m \text{ and} \\ B|C \sim_l & B_1|\ldots|B_l|C_1|\ldots|C_m. \end{array}$$

These are prime decomposition, and by Theorem 2 they are unique. As $A|C \sim_l B|C$, they have to be identical. Hence $k + m = l + m$, thus $k = l$. This implies that the decompositions of A and B have to be identical (up to $\sim_l$), which implies $A \sim_l B$. □

6. Decomposition w.r.t. Weak Labeled Bisimilarity

In this part, we discuss unique decomposition with respect to (weak) labeled bisimilarity. Note that $P \approx_l Q$ implies $|P|_v = |Q|_v$ for all closed processes P and Q (cf. Lemma 3).

To obtain our unique decomposition result for weak labeled bisimilarity, we need to define parallel prime with respect to weak labeled bisimilarity.

Definition 12 (Weakly Parallel Prime)**.** *A closed extended process P is* weakly parallel prime*, if*

- $P \not\approx_l 0$ *and*
- *for all two closed processes Q and R such that $P \approx_l Q|R$, we have $Q \approx_l 0$ or $R \approx_l 0$.*

This definition is analogous to strongly parallel prime. However, as the following example shows, in contrast to strong bisimilarity, not all normed processes have a unique decomposition w.r.t. to weak bisimilarity.

Example 12. *Consider $P = \nu a.(out(a, m)|(in(a, x).(!in(b, y)))|in(a, x))$. Then we have $P \approx_l P|P$, hence we have no unique decomposition. Note that this example does not contradict our previous result, as we have $P \not\sim_l P|P$, as $P \to P' \sim_l 0$, but $P|P \to P'' \sim_l P$ and $P|P \not\to P'''$ for all $P''' \sim_l 0$. Hence, w.r.t. strong labeled bisimilarity, P is prime.*

If however we consider normed processes that contain neither restriction ("ν") nor conditionals, we have that every such normed process is finite (and hence has a unique decomposition, as we show below).

Lemma 8. *For every process P that does not contain restriction ("ν") or conditionals ("if then else"), we have that P is finite if and only if P is normed.*

Proof. It is easy to see that every finite process is normed. To show the converse, we use induction on the structure of P.

- $P = 0$: P is obviously finite and normed.
- $P = \{^M/_x\}$: P is finite and normed.
- $P = Q|R$: If $\mathcal{N}(P) < \infty$ then $\mathcal{N}(Q) < \infty$ and $\mathcal{N}(R) < \infty$. By induction hypothesis $|Q|_t < \infty$ and $|R|_t < \infty$, hence $|P|_t < \infty$.
- $P = !Q$: If $\mathcal{N}(P) < \infty$ then $|Q|_t = 0$, hence $|P|_t < \infty$.[5]
- $P = in(u, x).Q$ or $P = out(u, M).Q$: If $\mathcal{N}(P) < \infty$ then $\mathcal{N}(Q) < \infty$. By induction hypothesis $|Q|_t < \infty$, hence $|P|_t < \infty$.

□

[5] Moreover, $Q \approx_l 0$ as $dom(Q) = \emptyset$.

Similarly all processes that do not contain replication are finite.

In the following we show that all finite processes have a unique decomposition w.r.t. to (weak) labeled bisimilarity. To prove this, we need some preliminary lemmas about transitions and the domain of processes.

Lemma 9. *For every closed process A with $A \to^* A'$, we have $dom(A) = dom(A')$.*

Proof. The domain of a process is the set of free variables from its frame for which it defines a substitution. No transition can destroy an existing active substitution. Similarly, if A executes only internal reductions, the only possibility for A to create a new active substitutions is under restrictions (using rule ALIAS) Yet these restrictions cannot be removed as that would require a labeled transition (rule OPEN-ATOM). Hence $dom(A) = dom(A')$. □

Lemma 10. *For every closed process A for which no sequence of transitions $A \to^* \xrightarrow{\alpha} A'$ exists, we have $A \approx_l A'$ for all A' with $A \to^* A'$.*

Proof. Consider the relation $\mathcal{R} = \{(X, Y) | A \to^* X \text{ and } A \to^* Y\}$. We show that it fulfills the conditions of labeled bisimilarity:

1. Obviously we have $A \approx_s A$, which is closed under internal reductions (Lemma 9). Hence for every $(C, D) \in \mathcal{R}$ we have $C \approx_s D$.
2. Let $(C, D) \in \mathcal{R}$. Hence $A \to^* C$ and $A \to^* D$. If $C \to C'$, we have $A \to^* C'$, hence $(C', D) \in \mathcal{R}$ (and symmetrically for $D \to D'$).
3. The last condition is trivially true. Suppose there exists $(C, D) \in \mathcal{R}$ such that $C \xrightarrow{\alpha} C'$, then we have $A \to^* \xrightarrow{\alpha} C'$, which contradicts the hypothesis. The symmetrical case is analogous.

Obviously we have $(A, A') \in \mathcal{R}$ for all A' with $A \to^* A'$. □

The next lemma captures the fact that every process which cannot perform a visible transition and has an empty domain, is weakly bisimilar to 0 (the empty process).

Lemma 11. *If for a closed process A with $dom(A) = \emptyset$ there does not exist a sequence of transitions $A \to^* \xrightarrow{\alpha} A'$, then we have $A \approx_l 0$.*

Proof. Consider the relation $\mathcal{R} = \{(A', 0), (0, A') | A \to^* A'\}$. We show that it fulfills the conditions of labeled bisimilarity:

1. By hypothesis for all $(C, D) \in \mathcal{R}$ we have $\emptyset = dom(A) = dom(C)$ (as internal reductions do not change the active substitutions, Lemma 9) and $dom(D) = dom(0) = \emptyset$, hence $C \approx_s D$.
2. Let $(C, D) \in \mathcal{R}$. Assume w.l.o.g. $A \to^* C$ and $D = 0$. If $C \to C'$, we have $A \to^* C'$, hence $(C', 0) \in \mathcal{R}$ with $0 \to^* 0$. Note that symmetrically 0 cannot perform a transition, hence the condition is trivially true.
3. The last condition is trivially true. Suppose there exists $(C, D) \in \mathcal{R}$ such that $C \xrightarrow{\alpha} C'$, then we have $A \to^* \xrightarrow{\alpha} C'$, which contradicts the hypothesis. Symmetrically by definition 0 cannot perform any transitions at all.

As we have $(A, 0) \in \mathcal{R}$, this gives $A \approx_l 0$, which we wanted to show. □

As a direct consequence, this gives us that every non-zero process with empty domain can do a visible transition.

Corollary 2. *For every closed process A with $dom(A) = \emptyset$ and $A \not\approx_l 0$ there exists a sequence of transitions $A \rightarrow^* \xrightarrow{\alpha} A'$.*

Now we can show in a first step that a decomposition into prime factors exists.

Theorem 3 (Existence of Decomposition)**.** *Every closed finite extended process P can be expressed as the parallel product of parallel primes, i.e. $P \approx_l P_1 | \ldots | P_n$ where for all $1 \leq i \leq n$ P_i is weakly parallel prime.*

Proof. By induction on the visible depth of P, and on the size of the domain $dom(P)$.

- If $|P|_v = 0$:
  - If $|dom(P)| = 0$, then $P \approx_l 0$ (by Lemma 11), hence the decomposition is the empty product.
  - If $|dom(P)| > 0$, then $P \not\approx_l 0$, hence P is either parallel prime itself (in which case we are done), or can be written as $P \approx_l Q|R$ with $Q \not\approx_l 0$ and $R \not\approx_l 0$ (by the definition of parallel prime). As we have $dom(P) = dom(Q) \cup dom(R)$ with $dom(Q) \cap dom(R) = \emptyset$ and $|dom(Q)| > 0$, $|dom(R)| > 0$ (by Lemmas 3 and 11 since $Q \not\approx_l 0$ and $R \not\approx_l 0$), we have $|dom(Q)| < |dom(P)|$ and $|dom(R)| < |dom(P)|$, hence we can use the induction hypothesis to conclude.
- If $|P|_v > 0$:
  - If $|dom(P)| = 0$: P is either parallel prime itself, or can be written as $P \approx_l Q|R$. Then we have $dom(P) = dom(Q) = dom(R) = \emptyset$, and $|Q|_v > 0$, $|R|_v > 0$ (by Corollary 2), hence $|Q|_v < |P|_v$, $|R|_v < |P|_v$ and we can apply the induction hypothesis.
  - If $|dom(P)| > 0$: P is either parallel prime itself, or can be written as $P \approx_l Q|R$. We distinguish cases:

    Suppose $|Q|_v > 0$, $|R|_v > 0$, hence $|Q|_v < |P|_v$, $|R|_v < |P|_v$ by Lemma 3 and we can apply the induction hypothesis.

    Suppose w.l.o.g. $|Q|_v = 0 < |P|_v$, then $|R|_v = |P|_v$. Since $Q \not\approx_l 0$ by Lemma 11 this implies $|dom(Q)| > 0$, hence $|dom(R)| < |dom(P)|$, and we can use the induction hypothesis to conclude.

□

To prove uniqueness we use the following relation on processes.

Definition 13 ("$\succeq$")**.** *For two finite processes P and Q we have $P \succeq Q$ iff*

- $|P|_v > |Q|_v$ *or*
- $|P|_v = |Q|_v$ *and* $P \to^* Q$

i.e. P has either a longer visible trace than Q or P can be reduced to Q using internal reductions.

This is a partial order on finite processes modulo static equivalence. The relation is reflexive as we have $P \to^* P$, and transitive. It is also antisymmetric: Suppose $P \succeq Q$ and $Q \succeq P$, then $|P|_v = |Q|_v$, $P \to^* Q$ and $Q \to^* P$. Since P and Q are finite, we cannot have $P \to^* Q \to^* P$ for $P \not\equiv Q$ as this allows to construct an infinite trace.

Now we can show the uniqueness of the decomposition. Again, we start by considering only processes that cannot do transitions.

Lemma 12 (Uniqueness of Decomposition for Processes with Zero Depth)**.** *The parallel decomposition of a closed finite process P with $|P|_t = 0$ is unique (up to $\approx_l$).*

Proof. The proof is analogous to the proof of Lemma 6, except that we argue modulo weak labeled bisimilarity. For details, see Appendix B. □

Theorem 4 (Uniqueness of Decomposition)**.** *The parallel decomposition of a closed finite process P is unique (up to $\approx_l$).*

Proof. Again, we suppose that we have two different decompositions of P, namely

$$\begin{aligned} P &= A_1^{k_1} | A_2^{k_2} | \ldots | A_n^{k_n} \\ Q &= A_1^{l_1} | A_2^{l_2} | \ldots | A_n^{l_n} \end{aligned}$$

where $P \approx_l Q_f$, the A_i's are distinct (i.e. for $i \neq j$ we have $A_i \not\approx_l A_j$) and $k_i, l_i \geq 0$.

We show that this leads to a contradiction by induction on $a = |P|_t + |Q|_t$, and inside each case by induction on the size of the domain $b = |dom(P)| = |dom(Q)|$. In case $a = 0$ we are done using Lemma 12. Case $a > 0$:

- If $b = 0$: If $P \approx_l 0$ then the (empty) decomposition is unique. Hence suppose $0 \not\approx_l P \approx_l Q$.

  Let m be such that A_m is a maximal (w.r.t. $\succeq$) A_i with $k_i \neq l_i$ (hence $k_m \neq l_m$). Such m exists as we assume two different decompositions, and $\succeq$ is a partial order. Without loss of generality we assume $k_m > l_m$.

  In the following we use multiple times the fact that $P \approx_l Q$ and hence Q can simulate each transition of P and vice versa. Moreover, for our proof it is important that if $P \to^* \xrightarrow{\mu} P'$ such that $|P|_v = |P'|_v + 1$, then the labeled bisimilarity gives us $Q \to^* \xrightarrow{\mu} \to^* Q'$ with $P' \approx_l Q'$, and in Q the prime factors cannot communicate.

  Suppose two prime factors $A_r \xrightarrow{\beta} R$ and $A_s \xrightarrow{\bar{\beta}} S$ communicated (through an internal reduction), then this has consumed at least two visible actions,

hence $|Q'|_v \leq |Q|_v - 2 = |P|_v - 2 = |P'|_v - 1 < |P'|_v$ (cf. Lemma 3). Thus P' and Q' do not have the same visible depth, which contradicts that fact that they are bisimilar.

We now analyze different cases:

- If $P \approx_l A_m^{k_m}$, i.e. P is the power of a prime:
  Note that Q cannot contain a prime factor A_r, $r \neq m$ with $A_r \succeq A_m$: Suppose $l_r > 0$. By assumption, A_m is a maximal (w.r.t. $\succeq$) prime factor in which P and Q differ, hence $k_r = l_r > 0$. This contradicts $P \approx_l A_m^{k_m}$.
  If $k_m = 1$ (i.e. P is prime), then Q is prime as well, and since $1 = k_m > l_m$ we have $Q \approx_l A_j$ for some $j \neq m$, which gives $A_m \approx_l A_j$, which contradicts the distinctness of the prime factors.
  If $k_m > 1$:

  * Assume $l_m = 0$. We have $A_m \rightarrow^* \xrightarrow{\mu} R$ for some $\mu \in \mathbf{Act}$ (by $A_m \not\approx_l 0$ and Corollary 2) with $|R|_v = |A_m|_v - 1$, so $P \rightarrow^* \xrightarrow{\mu} P'$ with $\exp(A_m, P') = k_m - 1 > 0$. Since $P \approx_l Q$, there exists a Q' with $Q \rightarrow^* \xrightarrow{\mu} \rightarrow^* Q'$. For every such Q' we have $\exp(A_m, Q') = 0$ since A_m is maximal (w.r.t. $\succeq$), $l_i = 0$ for all A_i with $|A_i|_v > |A_m|_v$ and since communication between different prime factors – which could through the exchange of secret channels lead to bigger (in the sense of visible depth) new prime factors – is not possible. As P' and Q' have a unique prime decomposition by induction hypothesis, we have a contradiction with $\exp(A_m, P') = k_m - 1 > 0 = \exp(A_m, Q')$.
  * Hence assume $l_m > 0$:
    Suppose $l_m < k_m - 1$:
    As $A_m \rightarrow^* \xrightarrow{\mu} R$, we have $P \rightarrow^* \xrightarrow{\mu} P'$ with $\exp(A_m, P') = k_m - 1$ and since $P \approx_l Q$ there exists Q' with $Q \rightarrow^* \xrightarrow{\mu} \rightarrow^* Q'$. Hence we have $\exp(A_m, P') = k_m - 1 > l_m \geq \exp(A_m, Q')$ which contradicts $P \approx_l Q$ using the induction hypothesis.
    Hence assume $l_m = k_m - 1$:
    We can write $Q \equiv S|A_m^{l_m}$, where S is composed of prime factors. We now show that $S \approx_l A_m$.
    First, since $\emptyset = dom(A_m) = dom(P) = dom(Q) = dom(S)$ we have $S \approx_s A_m$.
    Second, suppose $S \xrightarrow{\mu} S'$. Since we have $P \approx_l Q$, $S|A_m^{l_m} \xrightarrow{\mu} S'|A_m^{l_m} = Q'$ gives us that $P \rightarrow^* \xrightarrow{\mu} \rightarrow^* P'$ (w.l.o.g., when $\mu = \tau$ we have $P \rightarrow^* P'$). Since P' and Q' have smaller total depth, we can apply the induction hypothesis and both Q' and P' have a unique prime decomposition, hence $P' = R|A_m^{k_m - 1}$ where $A_m \rightarrow^* \xrightarrow{\mu} \rightarrow^* R$ (or $A_m \rightarrow^* R$ respectively). By the uniqueness of the decomposition we have $R \approx_l S'$, which is what we wanted

to show.
Third, suppose $A_m \xrightarrow{\mu} R$. As we have $P \approx_l Q$, $P = A_m^{k_m} \xrightarrow{\mu} R|A_m^{k_m-1} = P'$ gives us that $Q \rightarrow^* \xrightarrow{\mu} \rightarrow^* Q'$ (w.l.o.g., otherwise $Q \rightarrow^* Q'$). Since P' and Q' have smaller total depth, we can apply the induction hypothesis and both Q' and P' have the same unique prime decomposition, hence $R|A_m^{k_m-1} = P' \approx_l Q' = S'|A_m^{k_m-1}$ for some S'. Thus $R \approx_l S'$, as we can apply cancellation since the induction hypothesis gives us unique decomposition. Since A_m is the biggest factor in which P and Q differ, all other factors in S cannot be reduced to A_m, and we have $S \rightarrow^* \xrightarrow{\mu} \rightarrow^* S'$ (or $S \rightarrow^* S'$ respectively), which allows to conclude.
Hence $S \approx_l A_m$, which contradicts either the distinctiveness of the prime factors or the fact that A_m is prime, and thus concludes this case.

– If P is not the power of a prime, there exists $j \neq m$ such that $k_j > 0$.

Let $\mu \in \mathbf{Act}$, i and T be such that $k_i > 0$, $A_i \rightarrow^* \xrightarrow{\mu} A_i'$, $P \rightarrow^* \xrightarrow{\mu} A_1^{k_1}|\ldots|A_t^{k_i-1}|\ldots|A_n^{k_n}|A_i' = T$ and $|P|_v = |T|_v + 1$ and for all ν such that $P \rightarrow^* \xrightarrow{\nu} P'$ with $|P|_v = |P'|_v + 1$ we have $\exp(A_m, P') \leq \exp(A_m, T)$. We now show that such μ, T exist.

By $b = 0$ we know that for all A_i with $k_i > 0$ or $l_i > 0$ we have $dom(A_i) = \emptyset$. Hence, by Corollary 2 we have $A_i \rightarrow^* \xrightarrow{\alpha} A_i'$ for all such A_i. We then choose i such that $k_i > 0$ and the transition $A_i \rightarrow^* \xrightarrow{\alpha} A_i'$ maximizes $exp(A_m, T)$. Note that $\exp(A_m, T) \geq k_m$, as a transition by a factor different from A_m does not decrease the number of A_m's, and by the argument above there are other factors which can execute visible transitions.

As $P \approx_l Q$ there exists Q' with $Q \rightarrow^* \xrightarrow{\mu} \rightarrow^* Q'$ and $Q' \approx_l T$. Hence $|Q|_v = |Q'|_v + 1$ and there exists A_t with $A_t \rightarrow^* \xrightarrow{\mu} \rightarrow^* R$ as there can be no communication between the A_i's (as shown above).

* If $|A_t|_v \leq |A_m|_v$ then $\exp(A_m, Q') \leq l_m < k_m \leq \exp(A_m, T)$, which gives the contradiction to the induction hypothesis. Note that as A_m is the maximal prime factor in which P and Q differ, $A_j \rightarrow^* A_m$ implies $k_j = l_j$, hence Q' cannot contain additional A_m as a result of internal reductions - this would imply $\exp(A_j, Q') \neq \exp(A_j, T)$.
* If $|A_t|_v > |A_m|_v$ then $t \neq m$, and $k_t = l_t > 0$ (as A_m is maximal). Consider $P \rightarrow^* \xrightarrow{\mu} \rightarrow^* P' = A_1^{k_1}|\ldots|A_t^{k_t-1}|\ldots|A_n^{k_n}|R$ with $\exp(A_m, P') = k_m + \exp(A_m, R)$. Hence $\exp(A_m, Q') = l_m + \exp(A_m, R) < k_m + \exp(A_m, R) = \exp(A_m, P') \leq \exp(A_m, T)$. This contradicts $Q' \approx_l T$ using the induction hypothesis.

- If $b > 0$: This is essentially the same proof as above. In the first case (P is a power of a prime), we only have to consider the case $k_m = 1$ as

$dom(A_m) \neq \emptyset$. Hence P is prime, and then Q is prime as well, and since $1 = k_m > l_m$ we have $Q \approx_l A_j$ for some $j \neq m$, which gives $A_m \approx_l A_j$, which contradicts the distinctness of the prime factors.

In the second case we have to be more careful when proving that μ and T with the desired properties exist. Once again, we suppose that they do not exist, hence for no A_i with $k_i > 0, i \neq m$ we have $A_i \rightarrow^* \xrightarrow{\mu} A_i'$, otherwise this allows a transition that would fulfill the conditions. Hence (by Lemma 11) we have $dom(A_i) \neq \emptyset$ for all i with $k_i > 0, i \neq m$. Let $\tilde{v} = dom(P) \setminus dom(A_m)$ and consider

$$\nu\tilde{v}.P \equiv A_m^{k_m} | \nu\tilde{v}.P' \approx_l A_m^{k_m}$$

where P' is P without the factor $A_m^{k_m}$. Similarly

$$\nu\tilde{v}.Q \equiv |_i \ \nu\tilde{v}_i.A_i^{l_i}$$

where $\tilde{v}_i = dom(A_i) \cap \tilde{v}$.

As $|dom(A_m^{k_m})| < |dom(P)|$ by induction hypothesis the decomposition of $A_m^{k_m}$ is unique. We cannot have $\tilde{v}_i = \emptyset$ for all $i \neq m$, as this contradicts the uniqueness of the decomposition as $A_i \not\approx_l A_m$. Hence $dom(A_i) \neq \emptyset$ and $l_i = 1$ for $i \neq m$.

As A_m is prime, we have that $\nu\tilde{v}_i.A_i \approx_l A_m | R$ for some $i \neq m$ and R. More precisely, we have $\nu\tilde{v}_i.A_i \approx_l A_m^l$ for some $l \geq 1$, as all other decompositions of R would contradict the primeness of A_m. In fact, since A_m is the biggest (w.r.t. to $\succeq$) factor in which P and Q differ, and there are no bigger factors in P, all the factors in Q must be smaller. Using Lemmas 2 and 11, this gives $l = 1$.

We cannot have $Q \approx_l A_i$ as this would directly give a decomposition of A_i. Hence there has to be another factor A_r which – by Lemma 11 – has either $dom(A_r) \neq \emptyset$ or can execute a visible transition (or both).

- If $dom(A_r) \neq \emptyset$, consider $\tilde{v}' = dom(Q) \setminus dom(A_i)$. Then – as above – we have:

  $$\nu\tilde{v}'.Q \equiv A_i | \nu\tilde{v}'.Q' = Q_1 \approx_l \nu\tilde{v}'.P = P_1$$

  where Q' is Q without the factor A_i.
- If $A_r \rightarrow^* \xrightarrow{\eta} A_r'$, we have

  $$Q \approx_l A_i | A_r | S \rightarrow^* \xrightarrow{\eta} A_i | A_r' | S = Q_1$$

  where S is Q without A_i and A_r. By $P \approx_l Q$ there exists P_1 such that $P \rightarrow^* \xrightarrow{\eta} \rightarrow^* P_1 \approx_l A_i | A_r' | S$.

In both cases, we have a unique decomposition by induction hypothesis. Additionally $\exp(A_i, Q_1) = 1$, and by the uniqueness of the decomposition

$\exp(A_i, P_1) = \exp(A_i, Q_1) = 1$. Let s be such that $dom(A_s) \cap dom(A_i) \neq \emptyset$, $k_s > 0$. Such s exists because of $dom(A_m) \subsetneq dom(A_i)$ and $dom(P) = dom(Q)$. Then by hypothesis A_s cannot do a visible transition, and by Lemma 10 $\exp(A_s, P_1) = \exp(A_s, P) = 1$, which contradicts $\exp(A_i, P_1) = 1$ because of the conflicting domains.

Hence μ and T with the desired properties exist, and the rest of the proof is the same as above.

□

Again we have a cancellation result using the same proof as above.

Lemma 13 (Cancellation Lemma)**.** *For all closed finite processes A, B and C, we have*

$$A|C \approx_l B|C \Rightarrow A \approx_l B$$

Proof. Similar to the proof of Lemma 7. □

7. (Un)decidability of the decomposition

Although we proved in the previous sections that unique decompositions exist, it is not clear whether there are algorithms that compute the decomposition given a process P as input. As the following example shows, it turns out that the problem is at least as difficult as deciding the word problem in an equational theory, i.e. whether an equality holds.

Example 13. *Consider the following process*

$$P = \nu d.\,((\texttt{if } a = b \texttt{ then } \texttt{out}(c, d) \texttt{ else } \nu e.\texttt{out}(c, e))\,|\texttt{out}(c, d))$$

where a and b are ground terms, and c is a free name. P is obviously finite and normed. We can see that its unique decomposition depends (in both cases) on the equation $a = b$.

- *Consider weak labeled bisimilarity:*
  - *If $a = b$ is true, then the unique decomposition P' is*

    $$P \approx_l P' = \nu d.\,(\texttt{out}(c, d)|\texttt{out}(c, d))\,,$$

    i.e. we have one prime factor. A further decomposition is impossible, since the restricted name d is used in both parts.
  - *If $a = b$ is false, then the unique decomposition P' is*

    $$P \approx_l P' = (\nu e.\texttt{out}(c, e))|\,(\nu d.\texttt{out}(c, d))\,,$$

    i.e. we have two prime factors. Here, we can decompose further as the left factor does not rely on d any more.

- *Similarly, consider strong labeled bisimilarity:*
  - *If $a = b$ is true, then the unique decomposition P' is*

    $$P \sim_l P' = \nu d.\,((\texttt{if } a = b \texttt{ then out}(c, d) \texttt{ else } 0)\,|\texttt{out}(c, d))\,,$$

    i.e. we have one prime factor. Note that although we can simplify the `else`*-case to* 0*, we cannot leave out the* `if` *entirely as it results in an internal transition, and we are reasoning up to strong labeled bisimilarity.*
  - *If $a = b$ is false, then the unique decomposition P' is*

    $$P \sim_l P' = (\texttt{if } a = b \texttt{ then } 0 \texttt{ else } \nu e.\texttt{out}(c, e))\,|\nu d.\,(\texttt{out}(c, d))\,,$$

    i.e. we have two prime factors.

Hence the decomposition depends in both cases on whether $a = b$ is true, which is undecidable in general (see [15] for an example of an equational theory with an undecidable word problem).

However in most practical applications in protocol verification the equational theories are decidable. Unfortunately the problem remains undecidable even if the word problem in the equational theory is decidable. To prove this we now define the *Unique Decomposition Decision Problem* (UDDP) for weak (UDDP-W) and strong labeled bisimilarity (UDDP-S).

Problem 1 (Unique Decomposition Decision Problem for Weak Labeled Bisimilarity (UDDP-W))**.**

Input:	*An equational theory, a finite closed processes $P = P_1\,|\ldots|\,P_n$*
Question:	*Is P in the unique decomposition form w.r.t. to weak labeled bisimilarity?*

Problem 2 (Unique Decomposition Decision Problem for Strong Labeled Bisimilarity (UDDP-S))**.**

Input:	*An equational theory, a normed closed processes $P = P_1\,|\ldots|\,P_n$*
Question:	*Is P in the unique decomposition form w.r.t. to strong labeled bisimilarity?*

Remark 2. *An algorithm $\mathcal{A}$ computing the unique decomposition of a process P can be used to solve the above problems as follows. Given the algorithm $\mathcal{A}$, we construct the following algorithm $\mathcal{B}$: Given a process $P = P_1\,|\ldots|\,P_n$ as input to the UDDP problem, execute $\mathcal{A}$ on each factor P_i to obtain $Q_i = \mathcal{A}(P_i)$. If all Q_i consist of only one factor, return true, false otherwise.*

We now show that this algorithms return true if and only if the answer to the UDDP problem is true.

- *If $P_1\,|\ldots|\,P_n$ is the unique decomposition of P into prime factors, then the algorithm cannot decompose any of the factors $P_1, \ldots, P_n$ any more, and hence returns true.*

- *If each $Q_i = \mathcal{A}(P_i)$ consists of only one factor, we know that all P_i are prime. If P contains only prime factors, by the uniqueness of the decomposition (Theorems 2 and 4, respectively), we have that P is in the unique decomposition form.*

Although these problems might appear easier than actually computing the normal form, it turns out that UDDP-S/W is undecidable in most cases, as we show below. In a first step, we now show that the UDDP is undecidable even if the word problem of the equational theory is decidable using a reduction from the *Post Correspondence Problem* (PCP) [16].

Problem 3 (Post Correspondence Problem (PCP) [16])**.**

Input: *Two lists of words $\alpha_1, \ldots, \alpha_n$ and $\beta_1, \ldots, \beta_n$ over an alphabet $\mathcal{A}$ with at least two distinct symbols, and which is disjoint to the set of indices $\mathcal{I}$*

Question: *Is there a finite list of indices $i_1, \ldots, i_n \in \mathcal{I}$ such that $\alpha_{i_1} \ldots \alpha_{i_n} = \beta_{i_1} \ldots \beta_{i_n}$?*

The assumption that $\mathcal{A}$ and $\mathcal{I}$ are disjoint simplifies our proofs, but is not essential as the applied π-calculus supports sorts.

Theorem 5 (Undecidability for Decidable Equational Theories)**.** *The Unique Decomposition Decision Problem for Weak Labeled Bisimilarity (UDDP-W) and the Unique Decomposition Decision Problem for Strong Labeled Bisimilarity (UDDP-S) are undecidable even if the word problem in the equational theory is decidable.*

Proof. We show that an algorithm $\mathcal{A}$ deciding the UDDP-W (or UDDP-S respectively) can be used to construct an algorithm $\mathcal{B}$ that decides the PCP.

Assume that we have an instance $\alpha_1, \ldots, \alpha_n$ and $\beta_1, \ldots, \beta_n$ of the PCP. Let $\alpha_i^1 \alpha_i^2 \ldots \alpha_i^{k_{\alpha_i}} = \alpha_i$ denote the k_{α_i} letters of the word α_i. Consider the following equational theory, where the equations are oriented (from left to right):

$$\begin{aligned}
\mathtt{trans}_\alpha(\mathtt{nil}) &= \mathtt{nil} \\
\forall 1 \leq i \leq n : \mathtt{trans}_\alpha(\mathtt{cons}(i, x)) &= \mathtt{cons}(\alpha_i^1, \mathtt{cons}(\alpha_i^2, \mathtt{cons}(\ldots \\
&\quad \mathtt{cons}(\alpha_i^{k_{\alpha_i}}, \mathtt{trans}_\alpha(x)) \ldots))) \\
\mathtt{trans}_\beta(\mathtt{nil}) &= \mathtt{nil} \\
\forall 1 \leq i \leq n : \mathtt{trans}_\beta(\mathtt{cons}(i, x)) &= \mathtt{cons}(\beta_i^1, \mathtt{cons}(\beta_i^2, \mathtt{cons}(\ldots \\
&\quad \mathtt{cons}(\beta_i^{k_{\beta_i}}, \mathtt{trans}_\beta(x)) \ldots)))
\end{aligned}$$

Now consider the following processes:

$$\begin{aligned}
P_{Sol} = \;& \mathtt{in}(c, x).\mathtt{if}\ x = \mathtt{nil}\ \mathtt{then}\ 0\ \mathtt{else} \\
& \mathtt{if}\ \mathtt{trans}_\alpha(x) = \mathtt{trans}_\beta(x)\ \mathtt{then}\ \mathtt{out}(d, a)\ \mathtt{else}\ \nu f.\mathtt{out}(d, f) \\
P = \;& \nu a.\,(P_{Sol} | \mathtt{out}(d, a))
\end{aligned}$$

The idea is the following: the process P_{Sol} receives a string x on c, and checks if it is not `nil`. It then checks using the equational theory whether x is a solution

to the PCP. The rules of the equational theory contain two functions, $\mathtt{trans}_\alpha$ and $\mathtt{trans}_\beta$, which allow to translate a list of indices into a list of characters (the concatenation of the corresponding words α_i and β_i, respectively). P_{Sol} then simply has to check if these concatenations of words are equal. Hence, if and only if the initial PCP has a solution x, P_{Sol} is able to receive it on channel c, and to output a on channel d (otherwise it outputs f).

Note now that P is finite, and that $\mathtt{trans}_\alpha(x) = \mathtt{trans}_\beta(x)$ is decidable: we only have a finite set of rules, and at each instance only one of them can be applied. The last rule then allows to translate $\mathtt{trans}_x(\mathtt{nil})$ to $\mathtt{nil}$, which then allows to compare $\mathtt{trans}_\alpha$ and $\mathtt{trans}_\beta$.

The algorithm $\mathcal{B}$ then works as follows: Given the input of the PCP, construct the equational theory and processes as above, and return $\mathcal{A}(P)$.

If the instance of the PCP has a solution, there is a trace where P_{Sol} inputs this solution on channel c, and hence outputs a. In such a case the process P cannot be decomposed further due to the shared restricted name a, and hence is in its unique decomposition form. Hence, $\mathcal{A}$ outputs true, which is also the answer to the PCP.

However, if P_{Sol} is unable to output a on channel d as there is no solution which could be input, we have a decomposition with at least two factors, since the restriction can be moved to the $\mathtt{out}(d, a)$ only (as in the above example). Thus, if the instance of the PCP has no solution, P is not in its unique decomposition form, and $\mathcal{A}$ outputs false, which is also the answer to the PCP.

Hence, $\mathcal{B}$ can be used to decide the PCP, which is undecidable [16]. □

Note that in the above equational theory equality (i.e. the word problem) is decidable, but unification is not: by unifying $\mathtt{trans}_\alpha(\mathtt{cons}(x, y))$ with $\mathtt{trans}_\beta(\mathtt{cons}(x, y))$ we could find a solution for the PCP. Yet at least for the UDDP-S, we have undecidability also for equational theories where the word problem and unification are decidable.

Theorem 6. *The Unique Decomposition Decision Problem for Strong Labeled Bisimilarity (UDDP-S) is undecidable in general even if unification and the word problem in the equational theory are decidable.*

Proof. We show again that an algorithm $\mathcal{A}$ deciding UDDP-S can be used to construct an algorithm $\mathcal{B}$ that decides PCP.

Assume that we have an instance $\alpha_1, \ldots, \alpha_n$ and $\beta_1, \ldots, \beta_n$ of the PCP. Let $\alpha_i^1, \ldots, \alpha_i^{k_{\alpha_i}}$ denote the k_{α_i} letters of the word α_i. Now consider the following

processes.

$$
\begin{aligned}
P_i = \;& \mathtt{in}(c,(x,y)).\mathtt{out}(c,(\mathtt{cons}(\alpha_i^1,\mathtt{cons}(\alpha_i^2,\mathtt{cons}(\ldots,\mathtt{cons}(\alpha_i^{k_{\alpha_i}},x)\ldots))),\\
& \qquad \mathtt{cons}(\beta_i^1,\mathtt{cons}(\beta_i^2,\mathtt{cons}(\ldots,\mathtt{cons}(\beta_i^{k_{\beta_i}},y)\ldots))))\\
P_{nil} = \;& \mathtt{out}(c,(\mathtt{nil},\mathtt{nil}))\\
P_f = \;& \nu f.\mathtt{out}(d,f)\\
P_{Sol} = \;& \mathtt{in}(c,(x,y)).\mathtt{if}\ x=\mathtt{nil}\ \mathtt{then}\ P_f\ \mathtt{else}\\
& \mathtt{if}\ x=y\ \mathtt{then}\ \mathtt{out}(d,a)\ \mathtt{else}\ P_f\\
P_{PCP} = \;& \nu c.\,(P_{nil}|!P_1|\ldots|!P_n|P_{Sol})\\
P = \;& \nu a.\,(\nu b.(\mathtt{out}(b,e)|\mathtt{in}(b,z)|\mathtt{in}(b,z).P_{PCP})|\mathtt{out}(d,a))
\end{aligned}
$$

Here `nil` is a constant (i.e. function of arity zero) symbolizing the empty string, and `cons` a function of arity two allowing to construct tuples. Note that we do not associate any equations, hence unification is syntactic and decidable.

The idea is the following. The process P_{nil} outputs two empty strings on the channel c, which is restricted in P_{PCP}. Then the processes P_i encode the possible words: they receive strings x and y on channel c, add their words, and output the result again on c. Finally the process P_{Sol} receives two strings on c, and checks if they are equal. Hence, if and only if the initial PCP has a solution, P_{PCP} is able to output a on channel d (otherwise it will output f).

Note now that P is normed, since there is the transition $P \to P'$ where $P' \sim_l \nu a.\mathtt{out}(d,a)$.

The algorithm $\mathcal{B}$ then works as follows: Given the input of the PCP, construct the processes as above, and return $\mathcal{A}(P)$.

If the instance of the PCP does not have a solution, P_{PCP} is unable to output a on channel d and outputs f. Hence we have a decomposition of P with at least two factors, since the restriction can be moved to the $\mathtt{out}(d,a)$ only (as in the above proof). In this case P is not in its unique decomposition form, hence $\mathcal{A}(P)$ and $\mathcal{B}$ return false.

If the instance of the PCP has a solution, then P_{PCP} is able to output a. In this case the process cannot be decomposed further, and P is in its unique decomposition form. Hence $\mathcal{A}(P)$ and $\mathcal{B}$ return true.

Thus, if and only if the instance of the PCP has a solution, $\mathcal{B}$ returns true. Hence, $\mathcal{B}$ solves the PCP, which is undecidable. □

Note that we cannot use the same proof technique for UDDP-W as the process P_{PCP} is not finite.

However, it remains open whether under the same hypothesis the Unique Decomposition Decision Problem w.r.t. Weak Labeled Bisimilarity (UDDP-W) is decidable. A main issue is that deciding weak labeled bisimilarity for finite processes given a decidable equational theory is still an open problem. Recently Cheval, Cortier and Delaune [17] showed that observational equivalence for determinate processes without replication (hence finite) and a subterm-convergent equational theory is decidable. In the same direction, Liu and Lin [18] developed a proof system for observational equivalence in the applied π-calculus. Their system is sound and complete on finite processes which admit finite partition,

for example simple processes [17]. Yet it is open whether a similar result can be obtained for all finite processes.

Even though the decomposition might not or not efficiently be computable in general, its existence still holds, and can be used e.g. in proofs. Moreover, in practical applications – that rarely use the full expressive power of the applied π-calculus –, a decomposition might still be computable.

8. Related Work

Unique decomposition has been a field of interest in process algebra for a long time. The first results for a subset of CCS were published by Moller and Milner [2, 3]. They showed that finite processes with interleaving can be uniquely decomposed with respect to strong bisimilarity. The same is true for finite processes with parallel composition, where – in contrast to interleaving – the parallel processes can synchronize. They also proved that finite processes with parallel composition can be uniquely decomposed w.r.t. weak bisimilarity.

Later on Christensen [1] proved a unique decomposition result for normed processes (i.e. processes with a finite shortest complete trace) in BPP with interleaving or parallel composition w.r.t. strong bisimilarity.

Luttik and van Oostrom [19] provided a generalization of the unique decomposition results for ordered monoids. They show that if the calculus satisfies certain properties, the unique decomposition result follows directly. Recently Luttik also extended this technique for weak bisimilarity [20].

However, these existing results focus on "pure" calculi such as CCS or BPP or variants thereof. The applied π-calculus, as an "impure" variant of the π-calculus designed for the verification of cryptographic protocols, has a more complex structure and semantics. The main differences are the equational theory to model cryptographic primitives and the active substitutions.

In particular, we cannot apply the general results by Luttik et al. [20, 19]. For their results to hold, it is necessary that all minimal elements (with respect to the transition relation) are equivalent to 0 (the empty process). Yet, in the applied π-calculus, active substitutions are minimal elements with respect to the transition relation, which are different from 0. Thus we cannot apply their results.

Additionally, the applied π-calculus inherits the expressive power of the π-calculus including *channel* or *link passing* (sometimes also called *mobility*) and *scope extrusion*. Consider three parallel processes P, Q and R, where P and Q synchronize using an internal reduction τ_c, i.e. $P|Q|R \xrightarrow{\tau_c} P'|Q'|R$ (see Figures 7 and 8). Channel passing allows a process P to send a channel y he shares with R to process Q (Figure 7). Scope extrusion arises for example when P sends a restricted channel y he shares with R to Q, since the scope after the transition includes Q' (Figure 8). It is of particular importance for unique decomposition since two parallel processes sharing a restricted channel might not be decomposable and hence a simple reduction might "fuse" two prime factors, which is not possible in BPP or CCS.

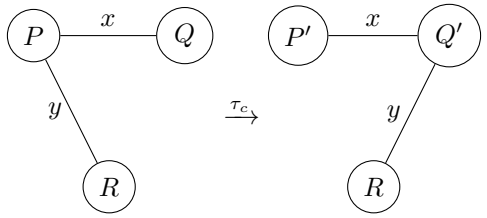

Figure 7: Channel/Link Passing in the applied π-calculus. Note that after the transition P' does not contain y any more, hence there is no link to R.

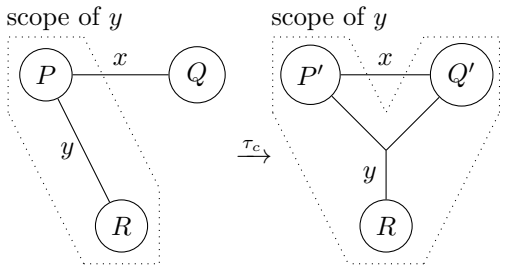

Figure 8: Scope extrusion in the applied π-calculus

An extended abstract presenting the results of Sections 5 and 6 without the detailed proofs was presented at FoSSaCS 2013 [21].

9. Conclusion

We started by recalling the applied π-calculus, in particular its syntax, semantics and notions of equivalence. We then presented two unique decomposition results for subsets of the applied π-calculus. We showed that every closed finite process can be decomposed uniquely with respect to weak labeled bisimilarity, and that every normed process can be decomposed uniquely with respect to strong labeled bisimilarity. Table 1 sums up our results.

Unfortunately it turned out that in general the problem of deciding whether a process is in its unique decomposition form (and thus also computing the unique decomposition) is undecidable. This is due to the complexity of the equational theories (which can be undecidable), yet even for equational theories where the word problem is decidable we were able to prove undecidability using

Type of Process	Strong Bisimilarity ($\sim_l$)	Weak Bisimilarity ($\approx_l$)
finite	unique cf. Thm. 1	unique cf. Thm. 3
normed	unique cf. Thm. 1	(Counter-)Example 8
general	(Counter-)Example 11	(Counter-)Example 11

Table 1: Summary of unique decomposition results for the applied π-calculus

Equational Theory	UDDP-W	UDDP-S
word prob. undecidable	undecidable, Thm. 5	undecidable, Thm. 5
word prob. decidable	undecidable, Thm. 5	undecidable, Thm. 5
unification decidable	open	undecidable, Thm. 6

Table 2: Summary of decidability results for unique decomposition

a reduction from the Post correspondence problem, relying now on restrictions. Moreover, we showed that the unique decomposition decision problem remains undecidable for normed processes even if unification is decidable in the equational theory, again using a reduction from the PCP. It remains open if the problem is decidable for finite processes w.r.t. weak labeled bisimilarity in the same setting. However, the mere existence of the decomposition remains useful in proofs as it provides a normal form with useful properties, even if it might not be computable in general. Moreover, for typical applications in practice, we still expect the computation of the decomposition to be feasible – as future work we would like to identify such decidable subclasses. Table 2 sums up our results.

As the concept of parallel prime decomposition has its inherent limitations with respect to replication ("!", see Example 11), a natural question is to find an extension to provide a normal form even in cases with infinite behavior. A first result in this direction has been obtained by Hirschkoff and Pous [22] for a subset of CCS with top-level replication. They define the *seed* of a process P as the process Q, Q bisimilar to P, of least size (in terms of prefixes) whose number of replicated components is maximal (among the processes of least size), and show that this representation is unique. They also provide a similar normal form result for the restriction-free-π-calculus (i.e. no "ν"). It remains however open if a similar result can be obtained for calculi with restriction such as the applied π-calculus, or even the π-calculus.

Acknowledgements. We would like to thank Bruno Blanchet, Cédric Fournet, Steve Kremer, Olivier Pereira and the anonymous referees for their valuable comments, and Ralf Sasse and José Meseguer for the helpful discussions on the decidability of various equational theories. This work was partly supported by the ANR project ProSe (decision ANR 2010-VERS-004), and conducted with the support of the "Digital trust" Chair from the University of Auvergne Foundation.

References

[1] S. Christensen, Decidability and decompostion in process algebras, Ph.D. thesis, School of Computer Science, University of Edinburgh (1993).

[2] R. Milner, F. Moller, Unique decomposition of processes, Theoretical Computer Science 107 (2) (1993) 357–363.

[3] F. Moller, Axioms for concurrency, Ph.D. thesis, School of Computer Science, University of Edinburgh (1989).

[4] R. Milner, Communication and Concurrency, International Series in Computer Science, Prentice Hall, 1989.

[5] R. Milner, J. Parrow, D. Walker, A calculus of mobile processes, part i, Information and Computation 100 (1) (1992) 1–40.

[6] M. Abadi, C. Fournet, Mobile values, new names, and secure communication, in: Proceedings of the 28th Symposium on Principles of Programming Languages (POPL'01), ACM, New York, 2001, pp. 104–115.

[7] J. Dreier, P. Lafourcade, Y. Lakhnech, Defining privacy for weighted votes, single and multi-voter coercion, in: Proceedings of the 17th European Symposium on Research in Computer Security (ESORICS'12), Vol. 7459 of LNCS, Springer, Pisa, Italy, 2012, pp. 451–468.

[8] J. F. Groote, F. Moller, Verification of parallel systems via decomposition, in: Proceedings of the Third International Conference on Concurrency Theory (CONCUR'92), Springer-Verlag, London, UK, 1992, pp. 62–76.

[9] U. Nestmann, B. C. Pierce, Decoding choice encodings, Information and Computation 163 (1) (2000) 1–59.

[10] C. Palamidessi, O. M. Herescu, A randomized encoding of the pi-calculus with mixed choice, Theoretical Computer Science 335 (2-3) (2005) 373–404.

[11] S. Delaune, S. Kremer, M. Ryan, Verifying privacy-type properties of electronic voting protocols, Journal of Computer Security 17 (2009) 435–487.

[12] J. Liu, A proof of coincidence of labeled bisimilarity and observational equivalence in applied pi calculus, Tech. Rep. ISCAS-SKLCS-11-05, Laboratory for Computer Science, Institute of Software, Chinese Academy of Sciences, available at `http://lcs.ios.ac.cn/ jliu/` (2011).

[13] J. Liu, H. Lin, A complete symbolic bisimulation for full applied pi calculus, Theoretical Computer Science 458 (2012) 76–112.

[14] M. Arapinis, J. Liu, E. Ritter, M. Ryan, Stateful applied pi calculus, in: Principles of Security and Trust - Third International Conference, POST 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014, Proceedings, Vol. 8414 of Lecture Notes in Computer Science, Springer, 2014, pp. 22–41.

[15] C. Marché, The word problem of acd-ground theories is undecidable, International Journal of Foundations of Computer Science 3 (1) (1992) 81–97.

[16] E. L. Post, A variant of a recursively unsolvable problem, Bulletin of the American Mathematical Society 52 (1946) 264–268.

[17] V. Cheval, V. Cortier, S. Delaune, Deciding equivalence-based properties using constraint solving, Theoretical Computer Science 492 (2013) 1–39.

[18] J. Liu, H. Lin, Proof system for applied pi calculus, in: C. Calude, V. Sassone (Eds.), Theoretical Computer Science, Vol. 323 of IFIP Advances in Information and Communication Technology, Springer Berlin Heidelberg, 2010, pp. 229–243.

[19] B. Luttik, V. van Oostrom, Decomposition orders – another generalisation of the fundamental theorem of arithmetic, Theoretical Computer Science 335 (2-3) (2005) 147–186.

[20] B. Luttik, Unique parallel decomposition in branching and weak bisimulation semantics, in: J. C. M. Baeten, T. Ball, F. S. de Boer (Eds.), Theoretical Computer Science - 7th IFIP TC 1/WG 2.2 International Conference, TCS 2012, Amsterdam, The Netherlands, September 26-28, 2012. Proceedings, Vol. 7604 of Lecture Notes in Computer Science, Springer, 2012, pp. 250–264.

[21] J. Dreier, C. Ene, P. Lafourcade, Y. Lakhnech, On unique decomposition of processes in the applied pi-calculus, in: Proceedings of the 16th International Conference Foundations of Software Science and Computation Structures (FOSSACS'13), Vol. 7794 of LNCS, Springer, Rome, Italy, 2013, pp. 50–64.

[22] D. Hirschkoff, D. Pous, On bisimilarity and substitution in presence of replication, in: Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP'10), Vol. 6199 of LNCS, Springer, 2010, pp. 454–465.

Appendix A. Proof of Lemma 3

Let P, Q and R be closed extended processes.

1. By definition of the length_t and length_v functions.
2. Suppose $P = Q|R$. Let w_P denote a maximal (with respect to length_v) complete (i.e. no further transitions are possible) sequence of transitions of P, i.e. $\text{length}_v(w_P) = |P|_v$. By definition of the function length_v we only count external transitions in w, which by rule PAR can originate either from Q or R, hence $|P|_v \leq |Q|_v + |R|_v$. Similarly, let w_Q and w_R denote maximal (with respect to length_v) complete sequence of transitions of Q and R respectively, i.e. $\text{length}_v(w_Q) = |Q|_v$ and $\text{length}_v(w_R) = |R|_v$. Then $w_P = w_Q w_R$ is a complete sequence of transitions of P, hence $|P|_v \geq |Q|_v + |R|_v$, thus $|P|_v = |Q|_v + |R|_v$.
3. Suppose $P = Q|R$. Let w_P denote a maximal (with respect to length_t) complete sequence of transitions of P, i.e. $\text{length}_t(w_P) = |P|_t$. As the length is maximal, there can be no synchronizations between Q and R, as otherwise we can build a longer trace by replacing this synchronization

with two external reductions. Hence all transitions originate either from Q or R, hence $|P|_v \leq |Q|_t + |R|_t$. Similarly, let w_Q and w_R denote maximal (with respect to length_t) complete sequences of transitions of Q and R respectively, i.e. $\text{length}_t(w_Q) = |Q|_t$ and $\text{length}_t(w_R) = |R|_t$. Then $w_P = w_Q w_R$ is a complete sequence of transitions of P, hence $|P|_t \geq |Q|_t + |R|_t$, thus $|P|_t = |Q|_t + |R|_t$.

4. Suppose $P = Q|R$. Let w_Q and w_R denote (one of) the smallest (with respect to length_n) complete (i.e. no further transitions are possible) sequence of transitions of Q and R respectively. Then $w_P = w_Q w_R$ is a complete sequence of transitions of P, hence $\mathcal{N}(P) \leq \mathcal{N}(Q) + \mathcal{N}(R)$. By contradiction, assume $\mathcal{N}(P) < \mathcal{N}(Q) + \mathcal{N}(R)$. Then there is a complete trace w'_P with $\text{length}_n(w'_P) < \text{length}_n(w_P)$. If w'_P contains no synchronizations between Q and R, each transition originates either from Q or R, hence giving shorter complete traces for Q and/or R, contradicting the minimality of w_Q and w_R. If w'_P contains a synchronization between Q and R, this can be rewritten into two external transitions of Q and R, resulting in a complete trace of the same length (by definition of length_n), leading to a contradiction. Hence $\mathcal{N}(P) = \mathcal{N}(Q) + \mathcal{N}(R)$.
5. Suppose $P = Q|R$. Then $dom(P) = dom(Q) \cup dom(R)$ and $dom(Q) \cap dom(R) = \emptyset$ as we cannot have two substitutions defining the same variable. Hence $|dom(P)| = |dom(Q)| + |dom(R)|$.
6. Assume $P \approx_l Q$, but by contradiction w.l.o.g. $|P|_v > |Q|_v$. Let w_s be a sequence of transitions of P with maximal number of visible transitions, i.e. $\text{length}_v(w_s) = |P|_v$. By the definition of $\approx_l$ each visible transition of P can be matched by a visible transition by Q, giving a trace with more visible transitions than $|Q|_v$, leading to a contradiction. Hence $|P|_v = |Q|_v$.
7. Assume $P \sim_l Q$, but by contradiction w.l.o.g. $|P|_t > |Q|_t$. Let w_s be a sequence of transitions of P with maximal number of transitions, i.e. $\text{length}_t(w_s) = |P|_t$. By the definition of $\sim_l$ each transition of P can be matched by a transition by Q, giving a trace with more transitions than $|Q|_t$, leading to a contradiction. Hence $|P|_t = |Q|_t$.
8. Assume $P \sim_l Q$ but by contradiction w.l.o.g. $\mathcal{N}(P) < \mathcal{N}(Q)$. Let w_i be a sequence of transitions of P with minimal number of transitions, i.e. $\text{length}_n(w_i) = \mathcal{N}(P)$, and ending in a state $P' \not\rightarrow$. By the definition of $\sim_l$ each transition of P can be matched by a transition by Q, giving a trace with less transitions than $\mathcal{N}(Q)$ and ending in a state $Q' \not\rightarrow$ as $Q' \sim_l P'$ by definition, leading to a contradiction. Hence $\mathcal{N}(P) = \mathcal{N}(Q)$.

Appendix B. Proof of Lemma 12

Again, we suppose that we have two different decompositions of P, namely

$$\begin{aligned} P &= A_1^{k_1} | A_2^{k_2} | \dots | A_n^{k_n} \\ Q &= A_1^{l_1} | A_2^{l_2} | \dots | A_n^{l_n} \end{aligned}$$

where $P \approx_l Q_f$, the A_i's are distinct (i.e. for $i \neq j$ we have $A_i \not\approx_l A_j$) and $k_i, l_i \geq 0$.

We show that this leads to a contradiction by induction on the size of the domain $b = |dom(P)| = |dom(Q)|$.

- If $b = 0$, then $P \approx_l 0$ (by Lemma 11), hence the decomposition is the unique empty product.
- If $b > 0$, then $P \not\approx_l 0$.

  Note that since $\forall i\ A_i \not\approx_l 0$ and $a = |P|_t + |Q|_t = 0$, we have $dom(A_i) \neq \emptyset$ by Lemma 11, which implies $k_i, l_i \leq 1$ as we cannot have two substitutions defining the same variable.

  Let m be such that $k_m \neq l_m$. Without loss of generality we assume $1 = k_m > l_m = 0$.

  Obviously we have $dom(P) = dom(Q)$. Let $\tilde{v} = dom(P) \setminus dom(A_m)$. Then we have (by Lemma 3 and rules ALIAS and NEW-PAR):

  $$\nu\tilde{v}.P \equiv A_m | \nu\tilde{v}.P' \approx_l A_m$$

  where P' is P without the factor A_m. Similarly

  $$\nu\tilde{v}.Q \equiv |_{i \in I}\ \nu\tilde{v}_i.A_i \mid |_{i \notin I}\ \nu\tilde{v}_i.A_i^{l_i} \approx_l |_{i \in I}\ \nu\tilde{v}_i.A_i$$

  where $I = \{i | dom(A_i) \cap dom(A_m) \neq \emptyset \text{ and } l_i = 1\}$, and where $\tilde{v}_i = dom(A_i) \cap \tilde{v}$.

  By $\nu\tilde{v}.P \approx_l \nu\tilde{v}.Q$ we have $A_m \approx_l |_{i \in I}\ \nu\tilde{v}_i.A_i$. If $|I| = 0$, we have $A_m \approx_l 0$ which contradicts the hypothesis that A_m is prime. Similarly for $|I| > 1$, we have a decomposition for A_m into several processes, which also contradicts A_m prime.

  For $|I| = 1$, i.e. $A_m \approx_l \nu\tilde{v}_i.A_i$ for the only index i in I, we have the following cases: If $\tilde{v}_i = \emptyset$, we have a contradiction to the distinctness hypothesis of the A_j's since $A_m \approx_l A_i$ with $m \neq i$ as $l_m = 0 \neq l_i = 1$.

  If $\tilde{v}_i \neq \emptyset$ we have $dom(A_m) \subsetneq dom(A_i)$. Now consider $\tilde{v}' = dom(Q) \setminus dom(A_i)$. Then - as above - we have:

  $$\nu\tilde{v}'.Q \equiv A_i | \nu\tilde{v}'.Q' \approx_l A_i$$

  where Q' is Q without the factor A_i. Similarly

  $$\nu\tilde{v}'.P \equiv |_{j \in I'}\ \nu\tilde{v}'_j.A_j \mid |_{j \notin I'}\ \nu\tilde{v}'_j.A_j^{l_j} \approx_l |_{j \in I'}\ \nu\tilde{v}'_j.A_j$$

  where $I' = \{j | dom(A_j) \cap dom(A_i) \neq \emptyset \text{ and } k_j = 1\}$ and $\tilde{v}'_j = dom(A_j) \cap \tilde{v}'$. Note now that since $dom(A_m) \subsetneq dom(A_i)$, and as $dom(A_i) = dom(|_{j \in I'}\ \nu\tilde{v}'_j.A_j)$ (by $dom(P) = dom(Q)$) we have $|I'| > 1$, hence $A_i \approx_l |_{j \in I'}\ \nu\tilde{v}'_j.A_j$ gives a decomposition of A_i, which contradicts the hypothesis that it is prime.