Musings on Encodings and Expressiveness

Rob van Glabbeek
NICTA, Sydney, Australia
School of Computer Science and Engineering, University of New South Wales, Sydney, Australia
rvg@cs.stanford.edu

This paper proposes a definition of what it means for one system description language to encode another one, thereby enabling an ordering of system description languages with respect to expressive power. I compare the proposed definition with other definitions of encoding and expressiveness found in the literature, and illustrate it on a case study: comparing the expressive power of CCS and CSP.

1 Introduction

This paper aims at answering the question what it means for one language to encode another one, and make this definition applicable to order system description languages like CCS, CSP and the π-calculus with respect to their expressive power.

To this end it proposes a unifying concept of correct translation between two languages, and adapts it to translations *up to* a semantic equivalence, for languages with a denotational semantics that interprets the operators and recursion constructs as operations on a set of values, called a *domain*. Languages can be partially ordered by their expressiveness up to the chosen equivalence according to the existence of correct translations between them.

The concept of a [correct] translation between system description languages (or *process calculi*) was first formally defined by Boudol [7]. There, and in most other related work in this area, the domain in which a system description language is interpreted consists of the closed expressions from the language itself. In [18] I have reformulated Boudol's definition, while dropping the requirement that the domain of interpretation is the set of closed terms. This allows (but does not enforce) a clear separation of syntax and semantics, in the tradition of universal algebra. Nevertheless, the definition employed in [18] only deals with the case that all (relevant) elements in the domain are denotable as the interpretations of closed terms. Examples 1 and 2 herein will present situations where such a restriction is undesirable. In addition, both [7] and [18] require the semantic equivalence $\sim$ under which two languages are compared to be a congruence for both of them. This is too severe a restriction to capture some recent encodings.

The current paper aims to generalise the concept of a correct translation as much as possible, so that it is uniformly applicable in many situations, and not just in the world of process calculi. Also, it needs to be equally applicable to encodability and separation results, the latter saying that an encoding of one language in another does not exists. At the same time, it tries to derive this concept from a unifying principle, rather than collecting a set of criteria that justify a number of known encodability and separation results that are intuitively justified.

In Sections 5 and 9 I propose in fact two notions of encoding: *correct* and *valid* translations up to $\sim$. The former drops the restriction on denotability and $\sim$ being a congruence for the whole target language, but it requires $\sim$ to be a congruence for the source language, as well as the source's image within the target. The latter drops both congruence requirements, but at the expense of requiring denotability by closed terms. In situations where $\sim$ is a congruence for the source language's image within the target language *and* all semantic values are denotable, the two notions agree.

B. Luttik and M.A. Reniers (Eds.): Combined Workshop on Expressiveness in Concurrency and Structural Operational Semantics (EXPRESS/SOS 2012)
EPTCS 89, 2012, pp. 81–98, doi:10.4204/EPTCS.89.7

2 Correct translations and expressiveness

A language consists of *syntax* and *semantics*. The syntax determines the valid expressions in the language. The semantics is given by a mapping $[\![\ ]\!]$ that associates with each valid expression its meaning, which can for instance be an object, concept or statement. This mapping determines the set $\mathscr{D}$ of all objects, concepts or statements that can be denoted in the language, namely as its image.

A correct translation of one language into another is a mapping from the valid expressions in the first language to those in the second, that preserves their meaning, i.e. such that the meaning of the translation of an expression is the same as the meaning of the expression being translated. In order to formalise this, I represent a language $\mathscr{L}$ as a pair $(\mathbb{T}_{\mathscr{L}}, [\![\ ]\!]_{\mathscr{L}})$ of a set $\mathbb{T}_{\mathscr{L}}$ of valid expressions in $\mathscr{L}$ and a surjective mapping $[\![\ ]\!]_{\mathscr{L}} : \mathbb{T}_{\mathscr{L}} \to \mathscr{D}_{\mathscr{L}}$ from $\mathbb{T}_{\mathscr{L}}$ in some set of meanings $\mathscr{D}_{\mathscr{L}}$.

Definition 1 A *translation* from a language $\mathscr{L}$ into a language $\mathscr{L}'$ is a mapping $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$. It is *correct* when $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'} = [\![E]\!]_{\mathscr{L}}$ for all $E \in \mathbb{T}_{\mathscr{L}}$. Language $\mathscr{L}'$ is at least as *expressive* as $\mathscr{L}$ if a correct translation exists.

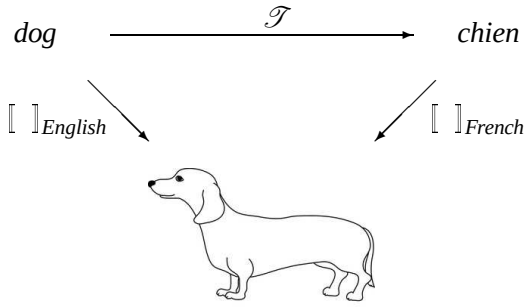

Figure 1: The essence of a correct translation

This fundamental notion is illustrated in Figure 1. It is not hard to see that a correct translation from $\mathscr{L}$ to $\mathscr{L}'$ exists if and only if anything that can be expressed in $\mathscr{L}$ can also be expressed in $\mathscr{L}'$, i.e. iff $\mathscr{D}_{\mathscr{L}} \subseteq \mathscr{D}_{\mathscr{L}'}$.

In this paper I will argue that this simple notion of a correct translation, when instantiated with appropriate proposals for $[\![\ ]\!]$ and $\mathscr{D}$, is a suitable definition of an encoding from one system description language into another, and thereby a suitable basis for classifying such languages w.r.t. expressiveness.

3 Dividing out a semantic equivalence

Definition 2 A *process graph* over an alphabet *Act* is a triple $(S, I, \rightarrow)$ with S a set of *states*, $I \in S$ the *initial state*, and $\rightarrow \subseteq S \times Act \times S$ the *transition relation*.

In other words, a process graph is a labelled transition system equipped with an initial state.

One way to apply the above definition of a translation to system description languages like CCS and CSP would be to take variable-free (and hence recursion-free) versions of those languages, and to define the meaning $[\![P]\!]$ of a CCS or CSP expression P to be the process graph $G_P := (S, P, \rightarrow)$ with as set of states S the set of all CCS/CSP expressions, as initial state the expression P, and $\rightarrow$ being the transition relation generated by the standard structural operational semantics of these languages. A variant of this idea is to reduce S to the states that are *reachable* from P by following transitions.

Now it happens to be case that the reachable part of each process graph that can be denoted by a CSP expression is *isomorphic*, but in general not *equal*, to one that can be denoted by a CCS expression. As

an example consider the CCS and CSP constants for *inaction*. In CCS this constant is called 0 whereas in CSP it is called STOP. The operational semantics generates no outgoing transitions of either process. It is therefore tempting to translate the CSP constant STOP into the CCS constant 0. Yet, this is not a correct translation in the current set-up, as the process graph with initial state 0 and no other states or transitions is different from the one with initial state STOP.

One way to deal with this anomaly is to relax Definition 1 by defining an appropriate semantic equivalence $\sim$ on $\mathscr{D}_{\mathscr{L}} \cup \mathscr{D}_{\mathscr{L}'}$ and merely requiring that the meanings of an expression and its translation are *equivalent*.

Definition 3 A translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ from a language $\mathscr{L}$ into a language $\mathscr{L}'$ is *correct up to* a semantic equivalence $\sim$ on $\mathscr{D}_{\mathscr{L}} \cup \mathscr{D}_{\mathscr{L}'}$ when $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'} \sim [\![E]\!]_{\mathscr{L}}$ for all $E \in \mathbb{T}_{\mathscr{L}}$.

In the example above, an appropriate candidate for $\sim$ could be isomorphism of reachable parts.

In some sense, introducing an appropriate semantic equivalence $\sim$, or maybe a preorder, appears to be the only reasonable way to allow intuitively correct translations, such as of 0 by STOP. Nevertheless, it need not be seen as a relaxation—and hence abandonment—of Definition 1, but rather as an appropriate instantiation. Namely the meaning of a CCS or CSP expression P is no longer a process graph G, but instead the equivalence class $[G]_{\sim}$ of all process graphs in $\mathscr{D}_{\mathrm{CCS}} \cup \mathscr{D}_{\mathrm{CSP}}$ that are equivalent to G.

Observation 1 Let $\mathscr{L} = (\mathbb{T}_{\mathscr{L}}, [\![\ ]\!]_{\mathscr{L}})$ and $\mathscr{L}' = (\mathbb{T}_{\mathscr{L}'}, [\![\ ]\!]_{\mathscr{L}'})$ be two languages, and $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ a correct translation between them up to an equivalence $\sim$ on $\mathscr{D}_{\mathscr{L}} \cup \mathscr{D}_{\mathscr{L}'}$. Then $\mathscr{T}$ is a correct translation between the languages $(\mathbb{T}_{\mathscr{L}}, [\![\ ]\!]^{\sim}_{\mathscr{L}})$ and $(\mathbb{T}_{\mathscr{L}'}, [\![\ ]\!]^{\sim}_{\mathscr{L}'})$, where $[\![E]\!]^{\sim}_{\mathscr{L}}$ is defined to be $[[\![E]\!]_{\mathscr{L}}]_{\sim}$.

Hence, correct translations up to some equivalence can be seen as special cases of correct translations. In doing so, it may appear problematic that the meaning $[\![E]\!]^{\sim}_{\mathscr{L}}$ of an expression $E \in \mathbb{T}_{\mathscr{L}}$ becomes dependent on the semantic domain $\mathscr{D}_{\mathscr{L}'}$ of the other language, namely by $[\![E]\!]^{\sim}_{\mathscr{L}}$ being the class of all processes in $\mathscr{D}_{\mathscr{L}} \cup \mathscr{D}_{\mathscr{L}'}$ that are equivalent with $[\![E]\!]_{\mathscr{L}}$. This worry can be alleviated by using, instead of $\mathscr{D}_{\mathscr{L}} \cup \mathscr{D}_{\mathscr{L}'}$, a natural class of which both $\mathscr{D}_{\mathscr{L}}$ and $\mathscr{D}_{\mathscr{L}'}$ are subsets. In the example above this could for instance be the class of all process graphs (over a suitable alphabet).

4 Translating operators

Up to isomorphism of reachable parts, so certainly up to coarser equivalences such as strong bisimilarity, the variable-free fragments of CSP and CCS with finitary choice are equally expressive. Namely each of them can express exactly the (equivalence classes of) finite process graphs. Here a process graph is finite if it has finitely many states and transitions, and no loops. In fact, these languages do not lose any expressiveness when omitting their parallel compositions, for parallel composition is not needed to denote any finite process graph.

Hence the treatment above does not address the question whether one of the *operators* of one language, such as parallel composition, can be mimicked by an operator or combination of operators in the other. This is to be blamed on the absence of variables. Once we admit variables in the language, the CCS parallel composition corresponds to the CCS expression $X|Y$, where X and Y are process variables, and a correct translation to CSP ought to translate this expression to a valid CSP expression—a CSP context built from CSP operators and the variables X and Y.

Henceforth, I consider single-sorted languages $\mathscr{L}$ in which *expressions* or *terms* are built from variables (taken from a set $\mathscr{X}$) by means of operators (including constants) and possibly recursion constructs.[1] The semantics of such a language is given by a domain of values **D**, and an interpretation of

[1] In Section 7 two postulates will be presented that restrict the class of languages considered in this paper.

each n-ary operator f of $\mathscr{L}$ as an n-ary operation $f^{\mathbf{D}} : \mathbf{D}^n \to \mathbf{D}$ on $\mathbf{D}$. Using the equations

$$[\![X]\!]_{\mathscr{L}}(\rho) = \rho(X) \qquad \text{and} \qquad [\![f(E_1,\ldots,E_n)]\!]_{\mathscr{L}}(\rho) = f^{\mathbf{D}}([\![E_1]\!]_{\mathscr{L}}(\rho),\ldots,[\![E_n]\!]_{\mathscr{L}}(\rho))$$

this allows an inductive definition of the meaning $[\![E]\!]_{\mathscr{L}}$ of an $\mathscr{L}$-expression E as a function of type $(\mathscr{X}\to\mathbf{D})\to\mathbf{D}$, associating a value $[\![E]\!]_{\mathscr{L}}(\rho)\in\mathbf{D}$ to E that depends on the choice of a *valuation* $\rho\colon\mathscr{X}\to\mathbf{D}$. The valuation associates a value from $\mathbf{D}$ with each variable. Moreover, $[\![E]\!]_{\mathscr{L}}(\rho)$ only depends on the restriction of ρ to those variables that occur free in E. In this setting, the class $\mathscr{D}_{\mathscr{L}}$ of possible meanings of $\mathscr{L}$-expressions is a subclass of $(\mathscr{X}\to\mathbf{D})\to\mathbf{D}$. Hence, a translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ between two such languages $\mathscr{L}$ and $\mathscr{L}'$ that employ the same set $\mathscr{X}$ of variables and are interpreted in the same domain $\mathbf{D}$ is correct when $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\rho) = [\![E]\!]_{\mathscr{L}}(\rho)$ for all $E \in \mathbb{T}_{\mathscr{L}}$ and all valuations $\rho : \mathscr{X} \to \mathbf{D}$.

Since normally the names of variables are irrelevant and the cardinality of the set of variables satisfies only the requirement that it is "sufficiently large", no generality is lost by insisting that two (system description) languages whose expressiveness is being compared employ the same set of (process) variables. On the other hand, two languages $\mathscr{L}$ and $\mathscr{L}'$ may be interpreted in different domains of values $\mathbf{D}$ and $\mathbf{D}'$. Without dividing out a semantic equivalence, one must insist that $\mathbf{D} \subseteq \mathbf{D}'$; otherwise no correct translation from $\mathscr{L}$ into $\mathscr{L}'$ exists. When $\mathbf{D} \subseteq \mathbf{D}'$ also $(\mathscr{X} \to \mathbf{D}) \subseteq (\mathscr{X} \to \mathbf{D}')$, so any function $(\mathscr{X} \to \mathbf{D}') \to \mathbf{D}'$ restricts to a function $(\mathscr{X} \to \mathbf{D}) \to \mathbf{D}'$. For the purpose of comparing the expressive power of $\mathscr{L}$ and $\mathscr{L}'$, the semantics of $\mathscr{L}'$ can be taken to be the mapping $[\![\ ]\!]_{\mathscr{L}'} : \mathbb{T}_{\mathscr{L}'} \to ((\mathscr{X} \to \mathbf{D}) \to \mathbf{D}')$, where $[\![E]\!]_{\mathscr{L}'}(\rho)$ with $E \in \mathbb{T}_{\mathscr{L}'}$ is considered for valuations $\rho : \mathscr{X} \to \mathbf{D}$ only. This restriction entails that when translating $\mathscr{L}$ into $\mathscr{L}'$ I compare the meaning of $\mathscr{L}$-expressions and their translations only under valuations within the domain $\mathbf{D}$ in which $\mathscr{L}$ is interpreted. A translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ from $\mathscr{L}$ to $\mathscr{L}'$ remains correct when $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\rho) = [\![E]\!]_{\mathscr{L}}(\rho)$ for all $E \in \mathbb{T}_{\mathscr{L}}$ and all valuations $\rho : \mathscr{X} \to \mathbf{D}$.

Example 1 Let $\mathscr{L}$ be the language whose syntax consists of a binary operator $+$, interpreted as addition in the domain $\mathbb{N}$ of the natural numbers. So $\mathbb{T}_{\mathscr{L}}$ contains expressions such as $X + (Y + Z)$. $\mathscr{L}'$ is the language with unary operators e^x and $\ln(x)$, interpreted as exponentiation and the natural logarithm on the reals $\mathbb{R}$, as well as the binary operator $\times$ of multiplication. If you do not like partial functions, the domain $\mathbb{R}$ can be extended with a special value $\perp$ to capture undefined outcomes. Note that $\mathbb{N} \subset \mathbb{R}$. Using that $\ln(e^x) = x$, the $\mathscr{L}$-expression $X + Y$ can be translated into the $\mathscr{L}'$-expression $\ln(e^X \times e^Y)$. Using this, a translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ is defined inductively by $\mathscr{T}(X) := X$ and $\mathscr{T}(E + F) := \ln(e^{\mathscr{T}(E)} \times e^{\mathscr{T}(E)})$.

5 Correct translations up to a congruence

This section aims at integrating the instantiations of the notion of a correct translation proposed in Sections 3 and 4. Let $\mathscr{L}$ and $\mathscr{L}'$ be two languages of the type considered in Section 4, with semantic mappings $[\![\ ]\!]_{\mathscr{L}} : \mathbb{T}_{\mathscr{L}} \to ((\mathscr{X} \to \mathbf{V}) \to \mathbf{V})$ and $[\![\ ]\!]_{\mathscr{L}'} : \mathbb{T}_{\mathscr{L}'} \to ((\mathscr{X} \to \mathbf{V}') \to \mathbf{V}')$. Here $\mathbf{V}$ and $\mathbf{V}'$ are domains of interpretation prior to quotienting by an appropriate semantic equivalence; they might be sets of process graphs with as states closed CCS expressions and closed CSP expressions, respectively. In order to compare these languages w.r.t. their expressive power I need a semantic equivalence $\sim$ that is defined on a unifying domain of interpretation $\mathbf{Z}$, with $\mathbf{V}, \mathbf{V}' \subseteq \mathbf{Z}$. Let $\mathbf{U} := \{v \in \mathbf{V}' \mid \exists v \in \mathbf{V}.\ v' \sim v\}$.

Definition 4 Two valuations $\eta, \rho : \mathscr{X} \to \mathbf{Z}$ are *$\sim$-equivalent*, $\eta \sim \rho$, if $\eta(X) \sim \rho(X)$ for each $X \in \mathscr{X}$.

In case there exists a $v \in \mathbf{V}$ for which there is no $\sim$-equivalent $v' \in \mathbf{V}'$, there is no correct translation from $\mathscr{L}$ into $\mathscr{L}'$ up to $\sim$. Namely, the semantics of $\mathscr{L}$ describes, among others, how any $\mathscr{L}$-operator evaluates the argument value v, and this aspect of the language has no counterpart in $\mathscr{L}'$. Therefore, I will require

$$\forall v \in \mathbf{V}.\ \exists v' \in \mathbf{V}'.\ v' \sim v. \tag{1}$$

This implies that for any valuation $\rho : \mathscr{X} \to \mathbf{V}$ there is a valuation $\eta : \mathscr{X} \to \mathbf{V}'$ with $\eta \sim \rho$.

Definition 5 A translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ is *correct up to* $\sim$ iff (1) holds and $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \sim [\![E]\!]_{\mathscr{L}}(\rho)$ for all $E \in \mathbb{T}_{\mathscr{L}}$ and all valuations $\eta : \mathscr{X} \to \mathbf{V}'$ and $\rho : \mathscr{X} \to \mathbf{V}$ with $\eta \sim \rho$.

Note that a correct translation as defined in Section 4 is exactly a correct translation up to the identity relation. If a correct translation up to $\sim$ from $\mathscr{L}$ into $\mathscr{L}'$ exists, then $\sim$ must be a congruence for $\mathscr{L}$.

Definition 6 An equivalence relation $\sim$ is a *congruence* for a language $\mathscr{L}$ interpreted in a semantic domain $\mathbf{V}$ if $[\![E]\!]_{\mathscr{L}}(\nu) \sim [\![E]\!]_{\mathscr{L}}(\rho)$ for any $\mathscr{L}$-expression E and any valuations $\nu, \rho : \mathscr{X} \to \mathbf{V}$ with $\nu \sim \rho$.

Proposition 1 If a correct translation up to $\sim$ from $\mathscr{L}$ into $\mathscr{L}'$ exists, then $\sim$ is a congruence for $\mathscr{L}$.

Proof: Let $\mathscr{T}$ be a correct translation up to $\sim$ from $\mathscr{L}$ into $\mathscr{L}'$. Let $E \in \mathbb{T}_{\mathscr{L}}$ and let $\nu, \rho : \mathscr{X} \to \mathbf{V}$ with $\nu \sim \rho$. By (1) there is a valuation $\eta : \mathscr{X} \to \mathbf{V}'$ with $\eta \sim \nu$. Hence $[\![E]\!]_{\mathscr{L}}(\nu) \sim [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \sim [\![E]\!]_{\mathscr{L}}(\rho)$.□

The existence of a correct translation up to $\sim$ from $\mathscr{L}$ into $\mathscr{L}'$ does not imply that $\sim$ is a congruence for $\mathscr{L}'$. However, $\sim$ has the properties of a congruence for those expressions of $\mathscr{L}'$ that arise as translations of expressions of $\mathscr{L}$, when restricting attention to valuations into $\mathbf{U}$. I call this a *congruence for* $\mathscr{T}(\mathscr{L})$.

Definition 7 Let $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be a translation from $\mathscr{L}$ into $\mathscr{L}'$. An equivalence $\sim$ on $\mathbb{T}_{\mathscr{L}'}$ is a *congruence for* $\mathscr{T}(\mathscr{L})$ if $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\nu) \sim [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta)$ for any $E \in \mathbb{T}_{\mathscr{L}}$ and $\nu, \eta : \mathscr{X} \to \mathbf{U}$ with $\nu \sim \eta$.

Proposition 2 If a correct translation up to $\sim$ from $\mathscr{L}$ into $\mathscr{L}'$ exists, then $\sim$ is a congruence for $\mathscr{T}(\mathscr{L})$.

Proof: Let $\mathscr{T}$ be correct up to $\sim$ from $\mathscr{L}$ into $\mathscr{L}'$. Let $E \in \mathbb{T}_{\mathscr{L}}$ and let $\nu, \eta : \mathscr{X} \to \mathbf{U}$ with $\nu \sim \eta$. By definition of $\mathbf{U}$ there is a $\rho : \mathscr{X} \to \mathbf{V}$ with $\rho \sim \nu$. Hence $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\nu) \sim [\![E]\!]_{\mathscr{L}}(\rho) \sim [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta)$. □

In the rest of this section I will show how the concept of a correct transition up to $\sim$ can be seen as an instantiation of the notion of correct translation, analogously to the situation in Section 3. To this end I need to unify the types of the semantic mappings $[\![\ ]\!]_{\mathscr{L}}$ and $[\![\ ]\!]_{\mathscr{L}'}$, say as $[\![\ ]\!]_{\mathscr{L}} : \mathbb{T}_{\mathscr{L}} \to ((\mathscr{X} \to \mathbf{E}) \to \mathbf{D})$ and $[\![\ ]\!]_{\mathscr{L}'} : \mathbb{T}_{\mathscr{L}'} \to ((\mathscr{X} \to \mathbf{E}) \to \mathbf{D})$.[2] This unification process involves dividing out the semantic equivalence $\sim$, as well as changing the type of a semantic mapping without tampering with the essence of its meaning. Below I propose two methods for doing so. The first method applies when $\sim$ is a congruence for both $\mathscr{L}$ and $\mathscr{L}'$, whereas the second merely requires that it is a congruence for $\mathscr{L}$. In both cases, the semantic mappings $[\![\ ]\!]_{\mathscr{L}}$ and $[\![\ ]\!]_{\mathscr{L}'}$ can be understood to be of types $\mathbb{T}_{\mathscr{L}} \to ((\mathscr{X} \to \mathbf{V}) \to \mathbf{Z})$ and $\mathbb{T}_{\mathscr{L}'} \to ((\mathscr{X} \to \mathbf{V}') \to \mathbf{Z})$, respectively. Dividing out $\sim$ yields the quotient domain $\mathbf{D} := \mathbf{Z}/\sim := \{[z]_\sim \mid z \in \mathbf{Z}\}$, consisting of the $\sim$-equivalence classes of elements of $\mathbf{Z}$, together with the mappings $[\![\ ]\!]^\sim_{\mathscr{L}} : \mathbb{T}_{\mathscr{L}} \to ((\mathscr{X} \to \mathbf{V}) \to \mathbf{D})$ and $[\![\ ]\!]^\sim_{\mathscr{L}} : \mathbb{T}_{\mathscr{L}'} \to ((\mathscr{X} \to \mathbf{V}') \to \mathbf{D})$, where $[\![E]\!]^\sim_{\mathscr{L}}(\rho) := [[\![E]\!]_{\mathscr{L}}(\rho)]_\sim$.

5.1 Translations up to a congruence for both languages

Let $\sim$ be a congruence for both $\mathscr{L}$ and $\mathscr{L}'$. Take $\mathbf{W} := \{\nu'' \in \mathbf{Z} \mid \exists \nu \in \mathbf{V}.\ \nu \sim \nu''\}$ and likewise $\mathbf{W}' := \{\nu'' \in \mathbf{Z} \mid \exists \nu' \in \mathbf{V}'.\ \nu' \sim \nu''\}$. Furthermore, $\mathbf{C} := \mathbf{W}/\sim$ and $\mathbf{C}' := \mathbf{W}'/\sim$. By (1), $\mathbf{W} \subseteq \mathbf{W}'$ and $\mathbf{C} \subseteq \mathbf{C}' \subseteq \mathbf{D}$.

Now $[\![\ ]\!]^\sim_{\mathscr{L}}$ can be recast as a function of type $\mathbb{T}_{\mathscr{L}} \to ((\mathscr{X} \to \mathbf{C}) \to \mathbf{D})$; namely by defining $[\![E]\!]^\sim_{\mathscr{L}}(\theta)$ with $\theta : \mathscr{X} \to \mathbf{C}$ to be $[\![E]\!]^\sim_{\mathscr{L}}(\rho)$, for any valuation $\rho : \mathscr{X} \to \mathbf{V}$ such that $\theta(X) = [\rho(X)]_\sim$ for all $X \in \mathscr{X}$. The congruence property of $\sim$ ensures that the value $[\![E]\!]^\sim_{\mathscr{L}}(\theta) \in \mathbf{D}$ is independent of the choice of the representatives $\rho(X)$ in the equivalence classes $\theta(X)$.

Likewise, $[\![\ ]\!]^\sim_{\mathscr{L}'}$ can be recast as a function of type $\mathbb{T}_{\mathscr{L}'} \to ((\mathscr{X} \to \mathbf{C}') \to \mathbf{D})$, which, as in Section 4, can be restricted to a function of type $\mathbb{T}_{\mathscr{L}'} \to ((\mathscr{X} \to \mathbf{C}) \to \mathbf{D})$. A translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ from $\mathscr{L}$ into $\mathscr{L}'$ can be defined to be *correct up to* $\sim$ when (1) holds and $[\![\mathscr{T}(E)]\!]^\sim_{\mathscr{L}'}(\theta) = [\![E]\!]^\sim_{\mathscr{L}}(\theta)$ for all $E \in \mathbb{T}_{\mathscr{L}}$ and all valuations $\theta : \mathscr{X} \to \mathbf{C}$. It is not hard to check that this definition agrees with Definition 5.

[2] In fact, it suffices to obtain mappings $[\![\ ]\!]_{\mathscr{L}} : \mathbb{T}_{\mathscr{L}} \to ((\mathscr{X} \to \mathbf{E}) \to \mathbf{D})$ and $[\![\ ]\!]_{\mathscr{L}'} : \mathbb{T}_{\mathscr{L}'} \to ((\mathscr{X} \to \mathbf{E}') \to \mathbf{D}')$ satisfying $((\mathscr{X} \to \mathbf{E}) \to \mathbf{D}) \subseteq ((\mathscr{X} \to \mathbf{E}') \to \mathbf{D}')$, and hence $\mathbf{E}' = \mathbf{E}$ and $\mathbf{D} \subseteq \mathbf{D}'$. However, any mapping $[\![\ ]\!]_{\mathscr{L}} : \mathbb{T}_{\mathscr{L}} \to ((\mathscr{X} \to \mathbf{E}) \to \mathbf{D})$ is also a mapping $[\![\ ]\!]_{\mathscr{L}} : \mathbb{T}_{\mathscr{L}} \to ((\mathscr{X} \to \mathbf{E}) \to \mathbf{D}')$, so one can just as well use $\mathbf{D}'$ for $\mathbf{D}$.

5.2 Translations up to a congruence for the source language

Let $\sim$ be a congruence for $\mathscr{L}$. Recast $[\![\ ]\!]^{\sim}_{\mathscr{L}}$ as a function of type $\mathbb{T}_{\mathscr{L}} \to ((\mathscr{X} \to \mathbf{U}) \to \mathbf{D})$ by defining $[\![E]\!]^{\sim}_{\mathscr{L}}(\eta)$ with $\eta : \mathscr{X} \to \mathbf{U}$ to be $[\![E]\!]^{\sim}_{\mathscr{L}}(\rho)$, for any valuation $\rho : \mathscr{X} \to \mathbf{V}$ with $\rho \sim \eta$. The congruence property of $\sim$ ensures that the value $[\![E]\!]^{\sim}_{\mathscr{L}}(\eta) \in \mathbf{D}$ is independent of the choice of the representative valuation ρ.

Since $\mathbf{U} \subseteq \mathbf{V}$ also $(\mathscr{X} \to \mathbf{U}) \subseteq (\mathscr{X} \to \mathbf{V})$, and therefore any function $(\mathscr{X} \to \mathbf{V}) \to \mathbf{D}$ restricts to a function $(\mathscr{X} \to \mathbf{U}) \to \mathbf{D}$. This way, $[\![\ ]\!]^{\sim}_{\mathscr{L}'}$ can be recast as a function of type $\mathbb{T}_{\mathscr{L}'} \to ((\mathscr{X} \to \mathbf{U}) \to \mathbf{D})$ as well, and unification is achieved. Now a translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ from $\mathscr{L}$ into $\mathscr{L}'$ can be defined to be *correct up to* $\sim$ when (1) holds and $[\![\mathscr{T}(E)]\!]^{\sim}_{\mathscr{L}'}(\eta) = [\![E]\!]^{\sim}_{\mathscr{L}}(\eta)$ for all $E \in \mathbb{T}_{\mathscr{L}}$ and all valuations $\eta : \mathscr{X} \to \mathbf{U}$. It is straightforward that this definition agrees with Definition 5.

6 A hierarchy of expressiveness preorders

An equivalence $\sim$ on a class $\mathbf{Z}$ is said to be *finer, stronger,* or *more discriminating* than another equivalence $\approx$ on $\mathbf{Z}$ if $p \sim q \Rightarrow p \approx q$ for all $p, q \in \mathbf{Z}$.

Theorem 1 Let $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be a translation from $\mathscr{L}$ into $\mathscr{L}'$, and let $\sim, \approx$ be congruences for $\mathscr{T}(\mathscr{L})$, with $\sim$ finer than $\approx$. If $\mathscr{T}$ is correct up to $\sim$, then it is also correct up to $\approx$.

Proof: Let $\mathbf{U}^{\approx} := \{v' \in \mathbf{V}' \mid \exists v \in \mathbf{V}.\ v \approx v''\}$. Let $\mathscr{T}$ be correct up to $\sim$. Then $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \sim [\![E]\!]_{\mathscr{L}}(\rho)$ for all $E \in \mathbb{T}_{\mathscr{L}}$ and all $\eta : \mathscr{X} \to \mathbf{V}'$ and $\rho : \mathscr{X} \to \mathbf{V}$ with $\eta \sim \rho$. To establish that $\mathscr{T}$ also is correct up to $\approx$, let $E \in \mathbb{T}_{\mathscr{L}}$, $v : \mathscr{X} \to \mathbf{V}'$ and $\rho : \mathscr{X} \to \mathbf{V}$ with $v \approx \rho$. Take $\eta : \mathscr{X} \to \mathbf{V}'$ with $\eta \sim \rho$—it exists by (1). Then $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \sim [\![E]\!]_{\mathscr{L}}(\rho)$ and hence $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \approx [\![E]\!]_{\mathscr{L}}(\rho)$. By (1) both η and v are of type $\mathscr{X} \to \mathbf{U}^{\approx}$. Since $\approx$ is a congruence for $\mathscr{T}(\mathscr{L})$ and $v \approx \eta$, $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(v) \approx [\![\mathscr{T}(E)]\!]_{\mathscr{L}}(\eta) \approx [\![E]\!]_{\mathscr{L}'}(\rho)$.□

When it is necessary to divide out a semantic equivalence, the quality of a translation depends on the choice of this equivalence. In no way would I want to suggest that a language $\mathscr{L}'$ is at least as expressive as $\mathscr{L}$ when there is a correct translation of $\mathscr{L}$ up to *some* equivalence—the equivalence does *not* appear in the scope of an existential quantifier. In fact, this would make any two languages equally expressive, namely by using the universal equivalence, relating any two processes. Instead, the equivalence needs to be chosen carefully to match the intended applications of the languages under comparison. In general, as show by Theorem 1, using a finer equivalence yields a stronger claim that one language can be encoded in another. On the other hand, when separating two languages $\mathscr{L}$ and $\mathscr{L}'$ by showing that $\mathscr{L}$ *cannot* be encoded in $\mathscr{L}'$, a coarser equivalence generally yields a stronger claim.

The following corollary of Theorem 1 is a powerful tool for proving the nonexistence of translations.

Corollary 1 If there is a correct translation up to $\sim$ from $\mathscr{L}$ into $\mathscr{L}'$, and $\approx$ is a congruence for $\mathscr{L}'$ that is coarser than $\sim$, then $\approx$ is a congruence for $\mathscr{L}$.

Proof: By combining Theorem 1 and Proposition 1. □

Proposition 3 If $\sim$ is a congruence for a language $\mathscr{L}$, then the identity is a correct translation up to $\sim$ from $\mathscr{L}$ into itself.

Proof: Immediately from Definitions 5 and 6. □

Theorem 2 If correct translations up to $\sim$ exists from $\mathscr{L}_1$ into $\mathscr{L}_2$ and from $\mathscr{L}_2$ into $\mathscr{L}_3$, then there is a correct translation up to $\sim$ from $\mathscr{L}_1$ into $\mathscr{L}_3$.

Proof: For $i = 1,2,3$ let $[\![\ ]\!]_{\mathscr{L}_i} : \mathbb{T}_{\mathscr{L}_i} \to ((\mathscr{X} \to \mathbf{V}_i) \to \mathbf{V}_i)$, and for $k = 1,2$ let $\mathscr{T}_k : \mathbb{T}_{\mathscr{L}_k} \to \mathbb{T}_{\mathscr{L}_{k+1}}$ be correct translations up to $\sim$ from $\mathscr{L}_k$ to $\mathscr{L}_{k+1}$. I will show that the translation $\mathscr{T}_2 \circ \mathscr{T}_1 : \mathbb{T}_{\mathscr{L}_1} \to \mathbb{T}_{\mathscr{L}_3}$ from $\mathscr{L}_1$ to $\mathscr{L}_3$, given by $\mathscr{T}_2 \circ \mathscr{T}_1(E) = \mathscr{T}_2(\mathscr{T}_1(E))$, is a correct up to $\sim$.

By assumption, $[\![\mathscr{T}_1(E)]\!]_{\mathscr{L}_2}(\eta) \sim [\![E]\!]_{\mathscr{L}_1}(\rho)$ for all $E \in \mathbb{T}_{\mathscr{L}_1}$ and all $\eta : \mathscr{X} \to \mathbf{V}_2$ and $\rho : \mathscr{X} \to \mathbf{V}_1$ with $\eta \sim \rho$, and likewise $[\![\mathscr{T}_2(F)]\!]_{\mathscr{L}_3}(\nu) \sim [\![F]\!]_{\mathscr{L}_2}(\eta)$ for all $F \in \mathbb{T}_{\mathscr{L}_2}$ and all $\nu : \mathscr{X} \to \mathbf{V}_3$ and $\eta : \mathscr{X} \to \mathbf{V}_2$ with $\nu \sim \eta$. Let $E \in \mathbb{T}_{\mathscr{L}_1}$, $\nu : \mathscr{X} \to \mathbf{V}_3$ and $\rho : \mathscr{X} \to \mathbf{V}$ with $\nu \sim \rho$; I need to show that $[\![\mathscr{T}_2 \circ \mathscr{T}_1(E)]\!]_{\mathscr{L}_3}(\nu) \sim [\![E]\!]_{\mathscr{L}_1}(\rho)$.

Let $\eta : \mathscr{X} \to \mathbf{V}_2$ be a valuation with $\eta \sim \rho$—it exists by (1). Then $\nu \sim \eta$. Taking $F := \mathscr{T}_1(E)$ one obtains $[\![\mathscr{T}_2(\mathscr{T}_1(E))]\!]_{\mathscr{L}_3}(\nu) \sim [\![\mathscr{T}_1(E)]\!]_{\mathscr{L}_2}(\eta) \sim [\![E]\!]_{\mathscr{L}_1}(\rho)$. □

Definition 8 A language $\mathscr{L}'$ *can express* or *is at least as expressive as* a language $\mathscr{L}$ *up to* $\sim$, if there exists a correct translation up to $\sim$ from $\mathscr{L}$ into $\mathscr{L}'$.

Theorem 2 shows that this relation is transitive. Restricted to languages for which $\sim$ is a congruence, it is even a preorder.

7 Compositionality

A substitution in $\mathscr{L}$ is a partial function $\sigma : \mathscr{X} \rightharpoonup \mathbb{T}_{\mathscr{L}}$ from the variables to the $\mathscr{L}$-expressions. For a given $\mathscr{L}$-expression $E \in \mathbb{T}_{\mathscr{L}}$, $E[\sigma] \in \mathbb{T}_{\mathscr{L}}$ denotes the $\mathscr{L}$-expression E in which each free occurrence of a variable $X \in \mathit{dom}(\sigma)$ is replaced by $\sigma(X)$, while renaming bound variables in E so as to avoid a free variable Y occurring in an expression $\sigma(X)$ ending up being bound in $E[\sigma]$. In general, a given expression $E \in \mathbb{T}_{\mathscr{L}}$ can be written in several ways as $F[\sigma]$. For instance, if $\mathscr{L}$ features a binary operator f, a unary operator g and a constant c, then the term $f(c,g(c)) \in \mathbb{T}_{\mathscr{L}}$ can be written as $F[\sigma]$ with

- $F = f(X,Y)$, $\sigma(X) = c$ and $\sigma(Y) = g(c)$, or
- $F = f(X,g(Y))$, $\sigma(X) = c$ and $\sigma(Y) = c$, or
- $F = f(c,g(X))$ and $\sigma(X) = c$.

Likewise, in case $\mathscr{L}$ contains a recursion construct $\mathbf{fix}_X S$, where S is a set of recursion equations $Y = E_Y$, then the expression $\mathbf{fix}_X\{X = f(g(c),g(g(X)))\}$, in which the variable X is bound, can be written as $F[\sigma]$ with $F = \mathbf{fix}_X\{X = f(Y,g(g(X)))\}$ and $\sigma(Y) = g(c)$.

Definition 9 A term $E \in \mathbb{T}_{\mathscr{L}}$ is a *prefix* of a term F, written $E \leq F$, if $F \stackrel{\alpha}{=} E[\sigma]$ for some substitution σ. Here $\stackrel{\alpha}{=}$ denotes *α-recursion*, renaming of bound variables while avoiding capture of free variables.

Since $E[\mathit{id}] = E$, where $\mathit{id} : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$ is the identity, and $E[\sigma][\xi] \stackrel{\alpha}{=} E[\xi \bullet \sigma]$, where the substitution $\xi \bullet \sigma$ is given by $(\xi \bullet \sigma)(X) = \sigma(X)[\xi]$, it follows that $\leq$ is reflexive and transitive, and hence a preorder. Write $\equiv$ for the kernel of $\leq$, i.e. $E \equiv F$ iff $E \leq F \wedge F \leq E$. If $E \equiv F$ then E can be converted into F by means of an injective renaming of its variables.

Definition 10 An term $H \in \mathbb{T}_{\mathscr{L}}$ is a *head* if H is not a single variable and $E \leq H$ implies that E is single variable or $E \equiv H$. It is a *head of* another term F if it is a head, as well as a prefix of F.

$f(X,Y)$ is a head of $f(c,g(c))$, and $\mathbf{fix}_X\{X = f(Y,g(g(X)))\}$ is a head of $\mathbf{fix}_X\{X = f(g(c),g(g(X)))\}$.

Postulate 1 Each expression E, if not a variable, has a head, which is unique up to $\equiv$.

This is easy to show for each common type of system description language, and I am not aware of any counterexamples. However, while striving for maximal generality, I consider languages with (recursion-like) constructs that are yet to be invented, and in view of those, this principle has to be postulated rather than derived. This means that here I consider only languages that satisfy this postulate. I also limit attention to languages where the meaning of an expression is invariant under α-recursion.

Postulate 2 If $E \stackrel{\alpha}{=} F$ then $[\![E]\!]_{\mathscr{L}} = [\![F]\!]_{\mathscr{L}}$.

The semantic mapping $[\![\ ]\!]_{\mathscr{L}} : \mathbb{T}_{\mathscr{L}} \to ((\mathscr{X} \to \mathbf{V}) \to \mathbf{V})$ extends to substitutions σ by $[\![\sigma]\!]_{\mathscr{L}}(\rho)(X) := [\![\sigma(X)]\!]_{\mathscr{L}}(\rho)$ for all $X \in \mathscr{X}$ and $\rho : \mathscr{X} \to \mathbf{V}$—here σ is extended to a total function by $\sigma(Y) := Y$ for all $Y \notin \mathit{dom}(\sigma)$. Thus $[\![\sigma]\!]_{\mathscr{L}}$ is of type $(\mathscr{X} \to \mathbf{V}) \to (\mathscr{X} \to \mathbf{V})$, i.e. a map from valuations to valuations. The inductive nature of the semantic mapping $[\![\ ]\!]_{\mathscr{L}}$ ensures that

$$[\![E[\sigma]]\!]_{\mathscr{L}}(\rho) = [\![E]\!]_{\mathscr{L}}([\![\sigma]\!]_{\mathscr{L}}(\rho)) \tag{2}$$

for all expressions $E \in \mathbb{T}_{\mathscr{L}}$, substitutions $\sigma : \mathscr{X} \rightharpoonup \mathbb{T}_{\mathscr{L}}$ and valuations $\rho : \mathscr{X} \to \mathbf{V}$. In case E is $f(X_1,\ldots,X_n)$ this amounts to $[\![f(E_1,\ldots,E_n)]\!]_{\mathscr{L}}(\rho) = f^{\mathbf{D}}([\![E_1]\!]_{\mathscr{L}}(\rho),\ldots,[\![E_n]\!]_{\mathscr{L}}(\rho))$, but (2) is more general and anticipates language constructs other than functions, such as recursion.

Definition 11 A translation $\mathscr{T}$ from $\mathscr{L}$ to $\mathscr{L}'$ is *compositional* if $\mathscr{T}(E[\sigma]) \stackrel{\alpha}{=} \mathscr{T}(E)[\mathscr{T} \circ \sigma]$ for each $E \in \mathbb{T}_{\mathscr{L}}$ and $\sigma : \mathscr{X} \rightharpoonup \mathbb{T}_{\mathscr{L}}$, and moreover $\mathscr{T}(X) = X$ for each $X \in \mathscr{X}$.

In case $E = f(t_1,\ldots,t_n)$ for certain $t_i \in \mathbb{T}_{\mathscr{L}}$ this amounts to $\mathscr{T}(f(t_1,\ldots,t_n)) \stackrel{\alpha}{=} E_f(\mathscr{T}(t_1),\ldots,\mathscr{T}(t_n))$, where $E_f := \mathscr{T}(f(X_1,\ldots,X_n))$ and $E_f(u_1,\ldots,u_n)$ denotes the result of the simultaneous substitution in this expression of the terms $u_i \in \mathbb{T}_{\mathscr{L}'}$ for the free variables X_i, for $i = 1,\ldots,n$. Again, Definition 11 is more general and anticipates language constructs other than functions, such as recursion.

Theorem 3 If any correct translation from $\mathscr{L}$ to $\mathscr{L}'$ up to $\sim$ exists, then there exists a compositional translation that is correct up to $\sim$.

Proof: Pick a representative from each $\equiv$-equivalence class of terms. With *the head of an expression* E I mean the chosen representative out of the $\equiv$-equivalence class of heads of E. Now each term $E \notin \mathscr{X}$ can uniquely be written as $H[\sigma]$, with H the head of E and $\mathit{dom}(\sigma)$ the set of free variables of H.

Given a correct translation $\mathscr{T}_0$, define the translation $\mathscr{T}$ inductively by

$$\begin{array}{ll} \mathscr{T}(X) := X & \text{for } X \in \mathscr{X} \\ \mathscr{T}(E) := \mathscr{T}_0(H)[\mathscr{T} \circ \sigma] & \text{when } E \stackrel{\alpha}{=} H[\sigma] \text{ as stipulated above.} \end{array}$$

First I show that $\mathscr{T}$ is compositional, using induction on E. So let $E \in \mathbb{T}_{\mathscr{L}}$ and $\xi : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$. I have to show that $\mathscr{T}(E[\xi]) \stackrel{\alpha}{=} \mathscr{T}(E)[\mathscr{T} \circ \xi]$. The case $E \in \mathscr{X}$ is trivial, so let $E \stackrel{\alpha}{=} H[\sigma]$. For each free variable X of H, $\sigma(X)$ is a proper subterm of E, so by the induction hypothesis $\mathscr{T}(\sigma(X)[\xi]) \stackrel{\alpha}{=} \mathscr{T}(\sigma(X))[\mathscr{T} \circ \xi]$.

$$\begin{array}{lll} \text{Thus } (\mathscr{T} \circ (\xi \bullet \sigma))(X) = \mathscr{T}((\xi \bullet \sigma)(X)) & & \text{by definition of functional composition } \circ \\ = \mathscr{T}(\sigma(X)[\xi]) & & \text{by definition of the relation } \bullet \text{ between substitutions} \\ \stackrel{\alpha}{=} \mathscr{T}(\sigma(X))[\mathscr{T} \circ \xi] & & \text{by induction, derived above; trivial if } X \notin \mathit{dom}(\sigma) \\ = ((\mathscr{T} \circ \xi) \bullet (\mathscr{T} \circ \sigma))(X) & & \text{by definition of the relations } \circ \text{ and } \bullet. \end{array}$$

This shows that the substitutions $\mathscr{T} \circ (\xi \bullet \sigma)$ and $(\mathscr{T} \circ \xi) \bullet (\mathscr{T} \circ \sigma)$ are equal up to α-recursion, from which it follows that that $F[\mathscr{T} \circ (\xi \bullet \sigma)] \stackrel{\alpha}{=} (F[\mathscr{T} \circ \sigma])[\mathscr{T} \circ \xi]$ for all terms $F \in \mathbb{T}_{\mathscr{L}'}$.

$$\begin{array}{lll} \text{Hence } \mathscr{T}(E[\xi]) \stackrel{\alpha}{=} \mathscr{T}(H[\sigma][\xi]) & & \text{since } E \stackrel{\alpha}{=} H[\sigma]. \\ \stackrel{\alpha}{=} \mathscr{T}(H[\xi \bullet \sigma]) & & \text{by the identity used already in proving transitivity of } \leq \\ = \mathscr{T}_0(H)[\mathscr{T} \circ (\xi \bullet \sigma)] & & \text{by definition of } \mathscr{T} \\ \stackrel{\alpha}{=} (\mathscr{T}_0(H)[\mathscr{T} \circ \sigma])[\mathscr{T} \circ \xi] & & \text{derived above} \\ = \mathscr{T}(H[\sigma])[\mathscr{T} \circ \xi] & & \text{by definition of } \mathscr{T} \\ \stackrel{\alpha}{=} \mathscr{T}(E)[\mathscr{T} \circ \xi] & & \text{since } E \stackrel{\alpha}{=} H[\sigma]. \end{array}$$

It remains to be shown that $\mathscr{T}$ is correct up to $\sim$, i.e. that $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \sim [\![E]\!]_{\mathscr{L}}(\rho)$ for all terms $E \in \mathbb{T}_{\mathscr{L}}$ and all valuations $\eta : \mathscr{X} \to \mathbf{V}'$ and $\rho : \mathscr{X} \to \mathbf{V}$ with $\eta \sim \rho$. Let η and ρ be such valuations. I proceed with structural induction on E. When handling a term $E \stackrel{\alpha}{=} H[\sigma]$, $\sigma(X)$ is a proper subterm of E for each free variable X of H. So by the induction hypothesis $[\![\mathscr{T}(\sigma(X)]\!]_{\mathscr{L}'}(\eta) \sim [\![\sigma(X)]\!]_{\mathscr{L}}(\rho)$. The valuation $[\![\sigma]\!]_{\mathscr{L}}(\rho)$ is defined such that $[\![\sigma]\!]_{\mathscr{L}}(\rho)(X) = [\![\sigma(X)]\!]_{\mathscr{L}}(\rho)$ for each $X \in \mathscr{X}$. Likewise, $[\![\mathscr{T} \circ \sigma]\!]_{\mathscr{L}'}(\eta)(X) = [\![\mathscr{T}(\sigma(X)]\!]_{\mathscr{L}'}(\eta)$ for each $X \in \mathscr{X}$. Hence $[\![\mathscr{T} \circ \sigma]\!]_{\mathscr{L}'}(\eta) \sim [\![\sigma]\!]_{\mathscr{L}}(\rho)$. (*)

- $[\![\mathscr{T}(X)]\!]_{\mathscr{L}'}(\eta) = [\![X]\!]_{\mathscr{L}'}(\eta) = \eta(X)$ by definitions of $\mathscr{T}$ and $[\![\ ]\!]_{\mathscr{L}'}$
  $\sim \rho(X)$ since $\eta \sim \rho$
  $= [\![X]\!]_{\mathscr{L}}(\rho)$ by definition of $[\![\ ]\!]_{\mathscr{L}}$.
- $[\![\mathscr{T}(H[\sigma])]\!]_{\mathscr{L}'}(\eta) = [\![\mathscr{T}_0(H)[\mathscr{T} \circ \sigma]]\!]_{\mathscr{L}'}(\eta)$ by definition of $\mathscr{T}$
  $= [\![\mathscr{T}_0(H)]\!]_{\mathscr{L}'}([\![\mathscr{T} \circ \sigma]\!]_{\mathscr{L}'}(\eta))$ by (2)
  $\sim [\![H]\!]_{\mathscr{L}}([\![\sigma]\!]_{\mathscr{L}}(\rho))$ by (*) above, as $\mathscr{T}_0$ is a correct translation
  $= [\![H[\sigma]]\!]_{\mathscr{L}}(\rho)$ by (2). □

Hence, for the purpose of comparing the expressive power of languages, correct translations between them can be assumed to be compositional.

8 Comparing the expressive power of CCS and CSP

As an application of my approach, in this section I quantify the degree to which the parallel composition of CSP can be expressed in CCS. It turns out that there exists a correct translation up to trace equivalence, but not up to the version of weak bisimilarity equivalence that takes divergence into account. This combination of an encoding and a separation result is typical when comparing system description languages. Here we see that for applications where divergence and branching time are a concern, the CSP parallel composition cannot be encoded in CCS; however, when linear time reasoning is all that matters, it can.

8.1 CCS

CCS [25] is parametrised with a set $\mathscr{A}$ of *names*. The set $\overline{\mathscr{A}}$ of *co-names* is $\overline{\mathscr{A}} := \{\overline{a} \mid a \in \mathscr{A}\}$, and $\mathscr{L} := \mathscr{A} \cup \overline{\mathscr{A}}$ is the set of *labels*. The function $\overline{\cdot}$ is extended to $\mathscr{L}$ by declaring $\overline{\overline{a}} = a$. Finally, $Act := \mathscr{L} \dot\cup \{\tau\}$ is the set of *actions*. Below, a, b, c, ... range over $\mathscr{L}$ and α, β over Act. A *relabelling function* is a function $f : \mathscr{L} \to \mathscr{L}$ satisfying $f(\overline{a}) = \overline{f(a)}$; it extends to Act by $f(\tau) := \tau$. Let $\mathscr{X}$ be a set $X, Y, \ldots$ of *process variables*. The set $\mathscr{E}$ of CCS terms or *process expressions* is the smallest set including:

$\alpha.E$	for $\alpha \in Act$ and $E \in \mathscr{E}$	*prefixing*
$\sum_{i\in I} E_i$	for I an index set and $E_i \in \mathscr{E}$	*choice*
$E\|F$	for $E, F \in \mathscr{E}$	*parallel composition*
$E\backslash L$	for $L \subseteq \mathscr{L}$ and $E \in \mathscr{E}$	*restriction*
$E[f]$	for f a relabelling function and $E \in \mathscr{E}$	*relabelling*
X	for $X \in \mathscr{X}$	a *process variable*
$\mathbf{fix}_X S$	for $S : \mathscr{X} \nrightarrow \mathscr{E}$ and $X \in dom(S)$	*recursion*.

One writes $E_1 + E_2$ for $\sum_{i\in I} E_i$ with $I = \{1,2\}$, and 0 for $\sum_{i\in\emptyset} E_i$. A partial function $S : \mathscr{X} \nrightarrow \mathscr{E}$ is called a *recursive specification*. The variables in its domain $dom(S)$ are called *recursion variables* and the equations $Y = S(Y)$ for $Y \in dom(S)$ *recursion equations*. A recursive specification $S : \mathscr{X} \nrightarrow \mathscr{E}$ is traditionally written as $\{Y = S(Y) \mid Y \in dom(S)\}$.

CCS is traditionally interpreted in the domain $\mathrm{T}_{\mathrm{CCS}}$ of closed CCS expressions up to α-recursion. Hence a valuation $\rho : \mathscr{X} \to \mathrm{T}_{\mathrm{CCS}}$, valuating each variable as a closed CCS expression, is just a closed substitution. The semantic mapping $[\![\ ]\!]_{\mathrm{CCS}}$ is given by $[\![E]\!]_{\mathrm{CCS}}(\rho) := E[\rho]$—a CCS expression E evaluates, under the valuation $\rho : \mathscr{X} \to \mathrm{T}_{\mathrm{CCS}}$, to the result of performing the substitution ρ on E. In fact, this is a common way to provide many system description languages with a semantics. Consequently, the distinction between syntax and semantics can, to a large extent, be dropped. It is for this reason that the semantic interpretation function $[\![\ ]\!]$ rarely occurs in papers on CCS-like languages.

The "real" semantics of CCS is given by the labelled transition relation $\to \subseteq \mathrm{T}_{\mathrm{CCS}} \times Act \times \mathrm{T}_{\mathrm{CCS}}$ between closed CCS expressions. The transitions $p \xrightarrow{\alpha} q$ with $p, q \in \mathrm{T}_{\mathrm{CCS}}$ and $\alpha \in Act$ are derived from

$$\alpha.E \xrightarrow{\alpha} E \qquad \frac{E_j \xrightarrow{\alpha} E'_j}{\sum_{i\in I} E_i \xrightarrow{\alpha} E'_j}\ (j \in I)$$

$$\frac{E \xrightarrow{\alpha} E'}{E|F \xrightarrow{\alpha} E'|F} \qquad \frac{E \xrightarrow{a} E',\ F \xrightarrow{\bar{a}} F'}{E|F \xrightarrow{\tau} E'|F'} \qquad \frac{F \xrightarrow{\alpha} F'}{E|F \xrightarrow{\alpha} E|F'}$$

$$\frac{E \xrightarrow{\alpha} E',\ \alpha \notin L \cup \bar{L}}{E\backslash L \xrightarrow{\alpha} E'\backslash L} \qquad \frac{E \xrightarrow{\alpha} E'}{E[f] \xrightarrow{f(\alpha)} E'[f]} \qquad \frac{S(X)[\mathbf{fix}_Y S/Y]_{Y\in dom(S)} \xrightarrow{\alpha} E}{\mathbf{fix}_X S \xrightarrow{\alpha} E}$$

Table 1: Structural operational semantics of CCS

the rules of Table 1. Formally a transition $p \xrightarrow{\alpha} q$ is part of the transition relation of CCS if there exists a well-founded, upwards branching tree (a *proof* of the transition) of which the nodes are labelled by transitions, such that

- the root is labelled by $p \xrightarrow{\alpha} q$, and
- if φ is the label of a node n and K is the set of labels of the nodes directly above n, then $\frac{K}{\varphi}$ is a rule from Table 1, with closed CCS expressions substituted for the variables $E, F, \ldots$.

8.2 CSP

CSP [8, 29, 9, 24] is parametrised with a set $\mathscr{A}$ of *communications*; $Act := \mathscr{A} \dot{\cup} \{\tau\}$ is the set of *actions*. Below, a, b range over $\mathscr{A}$ and α, β over Act. The set $\mathscr{E}$ of CSP terms is the smallest set including:

STOP		*inaction*
DIV		*divergence*
$(a \to E)$	for $a \in \mathscr{A}$ and $E \in \mathscr{E}$	*prefixing*
$E \,\Box\, F$	for $E, F \in \mathscr{E}$	*external choice*
$E \sqcap F$	for $E, F \in \mathscr{E}$	*internal choice*
$E \|_A F$	for $E, F \in \mathscr{E}$ and $A \subseteq \mathscr{A}$	*parallel composition*
E/b	for $b \in \mathscr{A}$ and $E \in \mathscr{E}$	*concealment*
$f(E)$	for $E \in \mathscr{E}$ and $f : Act \to Act$ with $f(\tau) = \tau$ and $f^{-1}(a)$ finite	*renaming*
X	for $X \in \mathscr{X}$	a *process variable*
$\mu X \cdot E$	for $E \in \mathscr{E}$ and $X \in \mathscr{X}$	*recursion*.

As in [29], I here leave out the guarded choice $(x : B \to P(x))$ and the constant RUN of [8], and the inverse image and sequential composition operator, with constant SKIP, of [8, 9]. The semantics of CSP was originally given in quite a different way [8, 9], but [29] provided an operational semantics of CSP in the same style as the one of CCS, and showed its consistency with the original semantics. It is this operational semantics I will use here; it is given by the rules in Table 2. Let $\mathscr{L} := \mathscr{A}$.

8.3 Trace semantics and convergent weak bisimilarity

I will compare the expressive power of CCS and CSP up two semantic equivalences: a linear time and a branching time equivalence. For the former I take *trace equivalence* [23] and for the latter a version of weak bisimilarity that takes divergence into account [22, 40, 1, 44]—called *convergent weak bisimilarity* in [17]. Unlike the standard weak bisimilarity of [25], this relation is finer than the failures-divergences semantics of [8, 29, 9, 24].

$$\mathrm{DIV} \xrightarrow{\tau} \mathrm{DIV} \qquad (a \to E) \xrightarrow{a} E \qquad E \sqcap F \xrightarrow{\tau} E \qquad E \sqcap F \xrightarrow{\tau} F$$

$$\frac{E \xrightarrow{a} E'}{E \,\Box\, F \xrightarrow{a} E'} \qquad \frac{F \xrightarrow{a} F'}{E \,\Box\, F \xrightarrow{a} F'} \qquad \frac{E \xrightarrow{\tau} E'}{E \,\Box\, F \xrightarrow{\tau} E' \,\Box\, F} \qquad \frac{F \xrightarrow{\tau} F'}{E \,\Box\, F \xrightarrow{\tau} E \,\Box\, F'}$$

$$\frac{E \xrightarrow{\alpha} E' \ (\alpha \notin A)}{E \|_A F \xrightarrow{\alpha} E' \|_A F} \qquad \frac{E \xrightarrow{a} E' \ \ F \xrightarrow{a} F' \ (a \in A)}{E \|_A F \xrightarrow{a} E' \|_A F'} \qquad \frac{F \xrightarrow{\alpha} F' \ (\alpha \notin A)}{E \|_A F \xrightarrow{\alpha} E \|_A F'}$$

$$\frac{E \xrightarrow{b} E'}{E/b \xrightarrow{\tau} E'/b} \qquad \frac{E \xrightarrow{\alpha} E' \ (\alpha \neq b)}{E/b \xrightarrow{\alpha} E'/b} \qquad \frac{E \xrightarrow{\alpha} E'}{f(E) \xrightarrow{f(\alpha)} f(E')} \qquad \mu X \cdot E \xrightarrow{\tau} E[\mu X \cdot E/X]$$

Table 2: Structural operational semantics of CSP

The relation $\Rightarrow \subseteq \mathrm{T}_{\mathrm{CCS}} \times \mathscr{L}^* \times \mathrm{T}_{\mathrm{CCS}}$ is the transitive closure of $\to$ that abstracts from τ-steps. Formally, $\Longrightarrow$ is the transitive closure of $\xrightarrow{\tau}$ and $p \overset{a_1 \cdots a_n}{\Longrightarrow} q$ for $n \geq 0$ holds iff there are $p_0, p_1, \ldots, p_n$ with $p_0 = p$, $p_{i-1} \Longrightarrow \xrightarrow{a_i} p_i$ for $i = 1, \ldots, n$, and $p_n \Longrightarrow q$. Below, T is a set that contains $\mathrm{T}_{\mathrm{CCS}}$ and $\mathrm{T}_{\mathrm{CSP}}$.

Definition 12 The set $T(p) \subseteq \mathscr{L}^*$ of *traces* of a process $p \in \mathrm{T}$ is given by $s \in T(p)$ iff $\exists p'.\ p \overset{s}{\Longrightarrow} p'$. Two processes $p, q \in \mathrm{T}$ are *trace equivalent* if $T(p) = T(q)$.

Definition 13 A relation $\mathscr{B} \subseteq \mathrm{T} \times \mathrm{T}$ is a *weak bisimulation* [25] if

- for any $p, p', q \in \mathrm{T}$ and $s \in \mathscr{L}^*$ with $p \mathscr{B} q$ and $p \overset{s}{\Longrightarrow} p'$, there is a q' with $q \overset{s}{\Longrightarrow} q'$ and $p' \mathscr{B} q'$,
- for any $p, q, q' \in \mathrm{T}$ and $s \in \mathscr{L}^*$ with $p \mathscr{B} q$ and $q \overset{s}{\Longrightarrow} q'$, there is a p' with $p \overset{s}{\Longrightarrow} p'$ and $p' \mathscr{B} q'$.

Two processes $p, q \in \mathrm{T}$ are *weakly bisimilar*, $p \leftrightarrows_w q$, if they are related by a weak bisimulation.

All we need to know about the *convergent* weak bisimilarity ($\leftrightarrows_w^{\downarrow}$) is that a process that has a divergence cannot be related to a divergence-free process, and that restricted to divergence-free processes it coincides with weak bisimilarity. Here a process *has a divergence* if it can do an infinite sequence of transitions that from some point onwards are all labelled τ.

Trace equivalence and (convergent) weak bisimilarity are congruences for CSP. The (convergent) weak bisimilarity fails to be a congruence for the $+$ of CCS, a problem that is commonly solved by taking its congruence closure. I do not need to do this when translating CSP into CCS, because correct translations need not be a congruence for the whole target language.

Note that even when restricting CCS to just 0, action prefixing and $+$, there is no correct translation of this language into CSP up to the congruence closure of $\leftrightarrows_w^{\downarrow}$—this is a direct consequence of Corollary 1.

8.4 A correct translation of CSP into CCS up to trace equivalence

For any choice of a CSP set of communications $\mathscr{A}$, I create a CCS set of names $\mathscr{B}$ and construct a translation from CSP with communications from $\mathscr{A}$ into CCS with names from $\mathscr{B}$.

Let $\mathscr{B} := \{a, a', a'' \mid a \in \mathscr{A}\}$, consisting of 3 disjoint copies of $\mathscr{A}$. For $A \subseteq \mathscr{A}$, let S_A be the recursive specification given by the single CCS equation $\{X = \sum_{a \in A} \overline{a}.a'.a''.a'.X + \sum_{a \in \mathscr{A} - A} \overline{a}.a''.X\}$ and S'_A be the recursive specification given by the single CCS equation $\{X = \sum_{a \in A} \overline{a}.\overline{a'}.\overline{a'}.X + \sum_{a \in \mathscr{A} - A} \overline{a}.a''.X\}$. Now, up to trace equivalence, and assuming that P features names from $\mathscr{A}$ only, $(P|\mathbf{fix}_X S_A)\backslash\mathscr{A}$ is a process that differs from P by the replacement of each a-transition by a sequence of transitions $a'a''a'$ if $a \in A$,

and by the single transition a'' otherwise. Likewise, $(P|\mathbf{fix}_X S'_A)\backslash\mathscr{A}$ differs from P by the replacement of each a-transition by a $\overline{a}'\overline{a}'$ if $a \in A$, and a'' otherwise. Let $\mathscr{A}' := \{a' \mid a \in \mathscr{A}\}$, and let the relabelling function f be such that $f(a'') = a$. Then the following is a correct translation of CSP into CCS up to trace equivalence.

$\mathscr{T}(X) = X$
$\mathscr{T}(\mu X \cdot E) = \mathbf{fix}_X\{X = \mathscr{T}(E)\}$
$\mathscr{T}(a \to E) = a.\mathscr{T}(E)$
$\mathscr{T}(\text{STOP}) = \mathscr{T}(\text{DIV}) = 0$
$\mathscr{T}(E \sqcap F) = \mathscr{T}(E \,\Box\, F) = \mathscr{T}(E) + \mathscr{T}(F)$
$\mathscr{T}(E/b) = (\mathscr{T}(E)|\mathbf{fix}_X\{X = \overline{b}.X\})\backslash\{b\}$
$\mathscr{T}(f(E)) = \mathscr{T}(E)[f]$
$\mathscr{T}(E \|_A F) = \big(\big((\mathscr{T}(E)|\mathbf{fix}_X S_A)\backslash\mathscr{A} \,\big|\, (\mathscr{T}(F)|\mathbf{fix}_X S'_A)\backslash\mathscr{A}\big)\backslash\mathscr{A}'\big)[f]$

8.5 The untranslatability of CSP into CCS up to convergent weak bisimilarity

In this section I show that there is no translation of CSP into CCS up to convergent weak bisimilarity. Suppose that $\mathscr{T}$ is such a translation. Let $\rho : \mathscr{X} \to \mathrm{T}_{\mathrm{CSP}}$ and $\eta : \mathscr{X} \to \mathrm{T}_{\mathrm{CCS}}$ satisfy $\rho(X) = \rho(Y) = (b \to \text{STOP}) \,\Box\, (b \to (c \to \text{STOP}))$ and $\eta(X) = \eta(Y) = b.0 + b.c.0$. Then $\rho \leftrightarrows_w^{\downarrow} \eta$. So

$$\mathscr{T}(X\|_{\{b,c\}}Y)[\eta] = [\![\mathscr{T}(X\|_{\{b,c\}}Y)]\!]_{\mathrm{CCS}}(\eta) \leftrightarrows_w^{\downarrow} [\![X\|_{\{b,c\}}Y]\!]_{\mathrm{CSP}}(\rho) \leftrightarrows_w^{\downarrow} b.0 + b.c.0.$$

Let $\nu : \mathscr{X} \to \mathrm{T}_{\mathrm{CCS}}$ satisfy $\nu(X) = \nu(Y) = b.0$. By the same reasoning as above

$$\mathscr{T}(X\|_{\{b,c\}}Y)[\nu] \leftrightarrows_w^{\downarrow} b.0.$$

Since $b.0$ has no divergence, neither does $\mathscr{T}(X\|_{\{b,c\}}Y)[\nu]$, so there must be a state $p \in \mathrm{T}_{\mathrm{CCS}}$ with $\mathscr{T}(X\|_{\{b,c\}}Y)[\nu] \Longrightarrow p \stackrel{\tau}{\not\longrightarrow}$. By [5, Proposition 7.1 (or 8)], it follows from the operational semantics of CCS that if $E[\sigma] \xrightarrow{\alpha} q$ for $E \in \mathbb{T}_{\mathrm{CCS}}$, $\sigma : \mathscr{X} \to \mathrm{T}_{\mathrm{CCS}}$ and $q \in \mathrm{T}_{\mathrm{CCS}}$, then q must have the form $F[\sigma']$ with $F \in \mathbb{T}_{\mathrm{CCS}}$ and for each variable W that occurs free in F there is a variable Z that occurs free in E, such that either $\sigma(Z) = \sigma'(W)$ or $\sigma(Z) \xrightarrow{\beta} \sigma'(W)$ for some $\beta \in Act$[3]—moreover, F depends on E and on the existence of the β-transitions, but not any other property of σ. So, for some $n \geq 0$,

$$\mathscr{T}(X\|_{\{b,c\}}Y)[\nu] \xrightarrow{\tau} E_1[\nu_1] \xrightarrow{\tau} E_2[\nu_2] \xrightarrow{\tau} \ldots \xrightarrow{\tau} E_n[\nu_n] \stackrel{\tau}{\not\longrightarrow}$$

where, for any free variable Z of E_i, $\nu_i(Z)$ is either 0 or $b.0$. This execution path can be simulated by

$$\mathscr{T}(X\|_{\{b,c\}}Y)[\eta] \xrightarrow{\tau} E_1[\eta_1] \xrightarrow{\tau} E_2[\eta_2] \xrightarrow{\tau} \ldots \xrightarrow{\tau} E_n[\eta_n] \stackrel{\tau}{\not\longrightarrow}$$

where $\eta_i(Z) = b.0 + b.c.0$ iff $\nu_i(Z) = b.0$ and $\eta_i(Z) = 0$ iff $\nu_i(Z) = 0$—i.e. always choosing $\eta(Z) \xrightarrow{b} 0$ over $\eta(Z) \xrightarrow{b} c.0$. By the properties of $\leftrightarrows_w^{\downarrow}$, $E_n[\eta_n] \leftrightarrows_w^{\downarrow} b.0 + b.c.0$. So there is a process $E_{n+1}[\eta_{n+1}]$ with $E_n[\eta_n] \xrightarrow{b} E_{n+1}[\eta_{n+1}] \Longrightarrow \xrightarrow{c}$. It must be that $E_{n+1}[\eta_{n+1}] \leftrightarrows_w^{\downarrow} c.0$.

The only rule in the structural operational semantics of CCS that has multiple premises has a conclusion with label τ. Furthermore, any rule with a τ-labelled premise, has a τ-labelled conclusion. Hence, since the transition $E_n[\eta_n] \xrightarrow{b} E_{n+1}[\eta_{n+1}]$ is not labelled τ, its proof has only one branch. This branch could stem from a transition from $\eta(X)$ or from $\eta(Y)$, but not both. W.l.o.g. I assume it does not stem from $\eta(X)$.

[3] In general multiple occurrences of Z in E may give rise to different associated variables W in F.

Let $\xi : \mathscr{X} \to \mathrm{T}_{\mathrm{CCS}}$ satisfy $\xi(X) = b.0$ and $\xi(Y) = b.0 + b.c.0$. Since in the proofs of the transitions in the above path from $\mathscr{T}(X\|_{\{b,c\}}Y)[\eta]$ the transition $\eta(X) \xrightarrow{b} c.0$ is never used, that path can be simulated by

$$\mathscr{T}(X\|_{\{b,c\}}Y)[\xi] \xrightarrow{\tau} E_1[\xi_1] \xrightarrow{\tau} E_2[\xi_2] \xrightarrow{\tau} \ldots \xrightarrow{\tau} E_n[\xi_n] \xrightarrow{b} E_{n+1}[\xi_{n+1}].$$

Note that $\mathscr{T}(X\|_{\{b,c\}}Y)[\xi] \leftrightarrows_w^{\downarrow} b.0$. Due to the properties of $\leftrightarrows_w^{\downarrow}$ the above derivation can be extended with

$$E_{n+1}[\xi_{n+1}] \xrightarrow{\tau} E_{n+2}[\xi_{n+2}] \xrightarrow{\tau} \ldots \xrightarrow{\tau} E_{n+k}[\xi_{n+k}]$$

ending in a *deadlock* state, where no further transitions are possible. This derivation, in turn, can be simulated by

$$E_{n+1}[\eta_{n+1}] \xrightarrow{\tau} E_{n+2}[\eta_{n+2}] \xrightarrow{\tau} \ldots \xrightarrow{\tau} E_{n+k}[\eta_{n+k}],$$

still ending in a deadlock state. This contradicts $E_{n+1}[\eta_{n+1}] \leftrightarrows_w^{\downarrow} c.0$. □

9 Valid translations up to a preorder

Let $\mathscr{L}$ and $\mathscr{L}'$ be languages with $[\![\ ]\!]_{\mathscr{L}} : \mathbb{T}_{\mathscr{L}} \to ((\mathscr{X} \to \mathbf{V}) \to \mathbf{V})$ and $[\![\ ]\!]_{\mathscr{L}'} : \mathbb{T}_{\mathscr{L}'} \to ((\mathscr{X} \to \mathbf{V}') \to \mathbf{V}')$. In this section I explore an alternative for the notion of a correct translation up to an equivalence $\sim$. This alternative doesn't have a build-in requirement that $\sim$ must be a congruence for $\mathscr{L}$;[4] however it only deals with semantic values denotable by closed terms.

Let $\mathrm{T}_{\mathscr{L}}$ be the set of closed $\mathscr{L}$-expressions, i.e. having no free variables. The meaning $[\![P]\!]_{\mathscr{L}}(\rho)$ of a closed term $P \in \mathrm{T}_{\mathscr{L}}$ is independent of the valuation $\rho : \mathscr{X} \to \mathbf{V}$, and hence denoted $[\![P]\!]_{\mathscr{L}}$.

Definition 14 A translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ *respects* $\sim$ if (1) holds and $[\![\mathscr{T}(P)]\!]_{\mathscr{L}'}(\eta) \sim [\![P]\!]_{\mathscr{L}}$ for all closed $\mathscr{L}$-expressions $P \in \mathrm{T}_{\mathscr{L}}$ and all valuations $\eta : \mathscr{X} \to \mathbf{U}$, with $\mathbf{U} := \{v \in \mathbf{V}' \mid \exists v \in \mathbf{V}.\ v' \sim v\}$.

Observation 2 If $\mathscr{T}$ is a correct translation from $\mathscr{L}$ into $\mathscr{L}'$ up to $\sim$, then it respects $\sim$.

Usually one employs translations $\mathscr{T}$ with the property that for any $E \in \mathbb{T}_{\mathscr{L}}$ any free variable of $\mathscr{T}(E)$ is also a free variable of E—I call these *free-variable respecting translations*, or *fvr-translations*. If there is at least one $Q \in \mathrm{T}_{\mathscr{L}'}$ with $[\![Q]\!]_{\mathscr{L}'} \in \mathbf{U}$, then any translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ can be modified to an fvr-translation $\mathscr{T}^\circ$ from $\mathscr{L}$ into $\mathscr{L}'$, namely by substituting Q for all free variables of $\mathscr{T}(E)$ that are not free in E. This modification preserves the properties of respecting $\sim$ and of being correct up to $\sim$. An fvr-translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ *respects* $\sim$ iff $[\![\mathscr{T}(P)]\!]_{\mathscr{L}'} \sim [\![P]\!]_{\mathscr{L}}$ for all closed $\mathscr{L}$-expressions $P \in \mathrm{T}_{\mathscr{L}}$.

Observation 3 Let $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be an fvr-translation from $\mathscr{L}$ into $\mathscr{L}'$, and let $\sim, \approx$ be equivalences (or preorders) on a class $\mathbf{Z} \subseteq \mathbf{V} \cup \mathbf{V}'$, with $\sim$ finer than $\approx$. If $\mathscr{T}$ respects $\sim$, then it also respects $\approx$.

The identity is a $\sim$-respecting fvr-translation from any language into itself.

If $\sim$-respecting fvr-translations exists from $\mathscr{L}_1$ into $\mathscr{L}_2$ and from $\mathscr{L}_2$ into $\mathscr{L}_3$, then there is a $\sim$-respecting fvr-translation from $\mathscr{L}_1$ into $\mathscr{L}_3$.

Respecting an equivalence or preorder is a very weak correctness requirement for translations. In spite of the separation result of Section 8.5, there trivially exists a translation from CSP to CCS that respects $\leftrightarrows_w^{\downarrow}$, or even strong bisimilarity. This follows from the observation that—thanks to the arbitrary index sets I and $dom(S)$ that may be used for choice and recursion—up to $\leftrightarrows_w^{\downarrow}$ every process graph is denotable by a CCS expression. In particular, compositionality is in no way implied by respect for an equivalence. It therefore makes sense to add compositionality as a separate requirement. The following shows that also the notion of a compositional $\sim$-respecting transition is a bit too weak.

[4] Moreover, it may be a preorder rather than an equivalence.

Example 2 Let $\mathscr{L}'$ be the language CCS without the recursion construct, but interpreted in a domain of arbitrary process graphs (similar to the graph model of ACP [2]). Let $\mathscr{L}$ be the same language, but with an extra operator $\_/\mathscr{L}$ that relabels all transitions into τ. The compositional translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ with $\mathscr{T}(X/\mathscr{L}) := 0$ respects $\leftrightarrows_w^{\downarrow}$. This is because the interpretation of any closed $\mathscr{L}$-expression is a process graph without infinite paths, and after relabelling all transitions into τ such a graph is equivalent to 0. Yet, there are process graphs G—those with infinite paths—that cannot be denoted by closed $\mathscr{L}$-expressions, and for which $G/\mathscr{L} \not\leftrightarrows_w^{\downarrow} 0$, demonstrating that $\mathscr{T}$ should not be seen as a valid translation.

Based on this, I add the denotability of all semantic values as a requirement of a valid translation.

Definition 15 A translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ is *valid up to* $\sim$ if it is compositional and respects $\sim$, while $\mathscr{L}$ satisfies

$$\forall v \in \mathbf{V}.\ \exists P \in \mathrm{T}_{\mathscr{L}}.\ [\![P]\!]_{\mathscr{L}} = v\,. \tag{3}$$

The following theorem (in combination with Theorem 3 and Observation 2) shows that this notion of a valid translation is consistent with the notion of a correct translation, and can be seen as extending that notion to situations where $\sim$ is not known to be a congruence.

Theorem 4 Let $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be a translation from $\mathscr{L}$ into $\mathscr{L}'$, and $\sim$ be a congruence for $\mathscr{T}(\mathscr{L})$. If $\mathscr{T}$ is valid up to $\sim$, then it is correct up to $\sim$.

Proof: Suppose $\mathscr{T}$ is valid up $\sim$. Then $[\![\mathscr{T}(P)]\!]_{\mathscr{L}'}(\eta) \sim [\![P]\!]_{\mathscr{L}}$ for all all closed $\mathscr{L}$-expressions $P \in \mathrm{T}_{\mathscr{L}}$ and all valuations $\eta : \mathscr{X} \to \mathbf{U}$. To establish that $\mathscr{T}$ is correct up to $\sim$, let $E \in \mathbb{T}_{\mathscr{L}}$ and let $\eta : \mathscr{X} \to \mathbf{V}'$ and $\rho : \mathscr{X} \to \mathbf{V}$ be valuations with $\eta \sim \rho$. So $\eta : \mathscr{X} \to \mathbf{U}$. I need to show that $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \sim [\![E]\!]_{\mathscr{L}}(\rho)$.

Let $\sigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ be a substitution with $[\![\sigma(X)]\!]_{\mathscr{L}} = \rho(X)$ for all $X \in \mathscr{X}$—such a substitution exists by (3). Furthermore, define $\nu : \mathscr{X} \to \mathbf{V}'$ by $\nu(X) := [\![\mathscr{T}(\sigma(X))]\!]_{\mathscr{L}'}(\eta)$ for all $X \in \mathscr{X}$. Since $\mathscr{T}$ respects $\sim$ I have $\nu(X) \sim \rho(X)$ for all $X \in \mathscr{X}$; thus $\eta \sim \rho \sim \nu$ and also $\nu : \mathscr{X} \to \mathbf{U}$.

$$\begin{aligned}
\text{Hence } [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) &\sim [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\nu) && \text{since } \sim \text{ is a congruence for } \mathscr{T}(\mathscr{L})\\
&= [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}([\![\mathscr{T}\circ\sigma]\!]_{\mathscr{L}'}(\eta)) && \text{expanding the definition of } \nu\\
&= [\![\mathscr{T}(E)[\mathscr{T}\circ\sigma]]\!]_{\mathscr{L}'}(\eta) && \text{by (2)}\\
&= [\![\mathscr{T}(E[\sigma])]\!]_{\mathscr{L}'}(\eta) && \text{by compositionality of } \mathscr{T}\\
&\sim [\![E[\sigma]]\!]_{\mathscr{L}} && \text{since } \mathscr{T} \text{ respects } \sim\\
&= [\![E]\!]_{\mathscr{L}}([\![\sigma]\!]_{\mathscr{L}}) && \text{by (2)}\\
&= [\![E]\!]_{\mathscr{L}}(\rho) && \text{by definition of } \rho. \qquad \square
\end{aligned}$$

10 Related work

The greatest expressibility result presented so far is by De Simone [39], who showed that a wide class of languages, including CCS, SCCS, CSP and ACP, are expressible up to strong bisimulation equivalence in MEIJE. Vaandrager [41] established that this result crucially depends on the use of unguarded recursion, and its noncomputable consequences. *Effective* versions of CCS, SCCS, MEIJE and ACP, not using unguarded recursion, are incapable of expressing all effective De Simone languages. Nevertheless, [18] isolated a *primitive effective* dialect of ACP (featuring primitive recursive renaming operators) in which a large class of primitive effective languages, including primitive effective versions of CCS, SCCS, CSP and MEIJE, can be encoded. All these results fall within the scope of the notion of translation and expressibility from [7] and [18], and use strong bisimulation as underlying equivalence.

In the last few years, a great number of encodability and separation results have appeared, comparing CCS, Mobile Ambients, and several versions of the π-calculus (with and without recursion; with mixed choice, separated choice or asynchronous) [6, 26, 28, 33, 16, 15, 10, 11, 14, 30, 3, 4, 32, 27, 31, 37, 13,

43, 12, 21, 34, 38, 42, 36, 35]; see [19, 20] for an overview. Many of these results employ different and somewhat ad-hoc criteria on what constitutes a valid encoding, and thus are hard to compare with each other. Gorla [20] collected some essential features of these approaches and integrated them in a proposal for a valid encoding that justifies most encodings and some separation results from the literature.

Like Boudol [7] and the present paper, Gorla requires a compositionality condition for encodings. However, his criterion is weaker than mine (cf. Definition 11) in that the expression E_f encoding an operator f may be dependent on the set of *names* occurring freely in the expressions given as arguments of f. The reason for this weakening appears to be that it provides a method for freeing up names that need to be fresh because of the special rôle they play in the translation, but might otherwise occur in the expressions being translated.

To address the problem of freeing up names I advocate a slightly different approach, already illustrated in Section 8.4: Most languages with names are parametrised with the set of names that are allowed in expressions. So instead of the single language CCS, there is an incarnation CCS($\mathscr{A}$) for each choice of names $\mathscr{A}$. Likewise, there is an incarnation CSP($\mathscr{A}$) of CSP for each $\mathscr{A}$. A priori, these parameters need not be related. So rather than insisting that for every $\mathscr{A}$ the language CCS($\mathscr{A}$) encodes CSP($\mathscr{A}$), I merely require that for each $\mathscr{A}$ there exists a $\mathscr{B}$ such that CCS($\mathscr{B}$) encodes CSP($\mathscr{A}$). Now the translations obviously are also parametrised by the choice of $\mathscr{A}$, and they may use names in $\mathscr{B}-\mathscr{A}$ as names that are guaranteed to be fresh. It is an interesting topic for future research to see if there are any valid encodability results à la [20] that suffer from my proposed strengthening of compositionality.

The second criterion of [20] is a form of invariance under name-substitution. It serves to partially undo the effect of making the compositionality requirement name-dependent. In my setting I have not yet found the need for such a condition. This criterion as formalised in [20] is too restrictive. It forbids the translation of the input process $a(x).E$ from value-passing CCS [25] into the CCS expression $\sum_{v\in\mathscr{V}} a_v.E[v/x]$, where $\mathscr{V}$ is a given (possibly infinite) set of data values. The problem is that a renaming of the single name a occurring in an expression E of value-passing CCS, say into b, would require renaming infinitely many names a_v occurring in $\mathscr{T}(E)$ into b_v, which is forbidden in [20]. Yet this translation, from [25], appears entirely justified intuitively.

The remaining three requirements of Gorla might be seen as singling our a particular preorder $\sqsubseteq$ for comparing terms and their translations. Since in [20], as in [7], the domain of interpretation consists of the closed expressions, and $\sqsubseteq$ is generally not a congruence for the source or target languages, one needs to compare with the approach of Section 9, where $\sim$ is allowed to be a preorder. The preorder presupposes a transition system with τ-transitions (reduction), and a notion of a success state; and compares processes based on these attributes only.

Hence Gorla's criteria are very close to an instantiation of mine with a particular preorder. Further work is needed to sort out to what extent the two approaches have relevant differences when evaluating encoding and separation results from the literature. Another topic for future work is to sort out how dependent known encoding and separation results are on the chosen equivalence or preorder.

As a concluding remark, many separation results in the literature[14, 30, 31, 37, 38, 21] are based on the assumption that parallel composition translates homomorphically, i.e. $\mathscr{T}(E|F) = \mathscr{T}(E)|\mathscr{T}(F)$.[5] This applies for instance to the proof in [21] that there is no valid encoding from the asynchronous π-calculus into CCS. In [20] this assumption is relaxed, but the separation proof of [20] hinges crucially on the too restrictive form of Gorla's second criterion. Whether the asynchronous π-calculus is expressible in CCS is therefore still wide open.

Acknowledgement My thanks to an EXPRESS/SOS referee for careful proofreading.

[5] This assumption is often defended by the theory that non-homomorphic translations reduce the degree of concurrency of the source process—a theory I do not share. Note that my translation of CSP into CCS in Section 8.4 is not homomorphic.

References

[1] S. Abramsky (1987): *Observation equivalence as a testing equivalence*. *Theoretical Computer Science* 53, pp. 225–241, doi:10.1016/0304-3975(87)90065-X.

[2] J.C.M. Baeten & W.P. Weijland (1990): *Process Algebra*. Cambridge Tracts in Theoretical Computer Science 18, Cambridge University Press.

[3] M. Baldamus, J. Parrow & B. Victor (2004): *Spi Calculus Translated to π-Calculus Preserving May-Tests*. In: Proceedings 19th IEEE Symposium on *Logic in Computer Science* (LICS 2004), July 2004, Turku, Finland, IEEE Computer Society, pp. 22–31, doi:10.1109/LICS.2004.1319597.

[4] M. Baldamus, J. Parrow & B. Victor (2005): *A Fully Abstract Encoding of the pi-Calculus with Data Terms*. In L. Caires, G.F. Italiano, L. Monteiro, C. Palamidessi & M. Yung, editors: Proceedings 32nd *International Colloquium on Automata, Languages and Programming,* ICALP 2005, Lisbon, Portugal, July 2005, LNCS 3580, Springer, pp. 1202–1213, doi:10.1007/11523468_97.

[5] B. Bloom, W.J. Fokkink & R.J. van Glabbeek (2004): *Precongruence Formats for Decorated Trace Semantics*. *Transactions on Computational Logic* 5(1), pp. 26–78, doi:10.1145/963927.963929. Available at http://theory.stanford.edu/~rvg/abstracts.html#48.

[6] M. Boreale (1998): *On the Expressiveness of Internal Mobility in Name-Passing Calculi*. *Theor. Comput. Sci.* 195(2), pp. 205–226, doi:10.1016/S0304-3975(97)00220-X.

[7] G. Boudol (1985): *Notes on algebraic calculi of processes*. In K. Apt, editor: *Logics and Models of Concurrent Systems*, Springer, pp. 261–303. NATO ASI Series F13.

[8] S.D. Brookes, C.A.R. Hoare & A.W. Roscoe (1984): *A theory of communicating sequential processes*. *Journal of the ACM* 31(3), pp. 560–599, doi:10.1145/828.833.

[9] S.D. Brookes & A.W. Roscoe (1985): *An improved failures model for communicating processes*. In S.D. Brookes, A.W. Roscoe & G. Winskel, editors: *Seminar on Concurrency*, LNCS 197, Springer, pp. 281–305, doi:10.1007/3-540-15670-4_14.

[10] N. Busi, M. Gabbrielli & G. Zavattaro (2003): *Replication vs. Recursive Definitions in Channel Based Calculi*. In J.C.M. Baeten, J.K. Lenstra, Parrow J & G.J. Woeginger, editors: Proceedings 30th *International Colloquium on Automata, Languages and Programming,* ICALP 2003, Eindhoven, The Netherlands, LNCS 2719, Springer, pp. 133–144, doi:10.1007/3-540-45061-0_12.

[11] N. Busi, M. Gabbrielli & G. Zavattaro (2009): *On the expressive power of recursion, replication and iteration in process calculi*. *Mathematical Structures in Computer Science* 19(6), pp. 1191–1222, doi:10.1017/S096012950999017X.

[12] D. Cacciagrano, F. Corradini, J. Aranda & F.D. Valencia (2008): *Linearity, Persistence and Testing Semantics in the Asynchronous Pi-Calculus*. *Electr. Notes Theor. Comput. Sci.* 194(2), pp. 59–84, doi:10.1016/j.entcs.2007.11.006.

[13] D. Cacciagrano, F. Corradini & C. Palamidessi (2007): *Separation of synchronous and asynchronous communication via testing*. *Theor. Comput. Sci.* 386(3), pp. 218–235, doi:10.1016/j.tcs.2007.07.009.

[14] M. Carbone & S. Maffeis (2003): *On the Expressive Power of Polyadic Synchronisation in pi-calculus*. *Nord. J. Comput.* 10(2), pp. 70–98.

[15] L. Cardelli, G. Ghelli & A.D. Gordon (2002): *Types for the Ambient Calculus*. *Inf. Comput.* 177(2), pp. 160–194, doi:10.1006/inco.2001.3121.

[16] L. Cardelli & A.D. Gordon (2000): *Mobile ambients*. *Theor. Comput. Sci.* 240(1), pp. 177–213, doi:10.1016/S0304-3975(99)00231-5.

[17] R.J. van Glabbeek (1993): *The Linear Time – Branching Time Spectrum II; The semantics of sequential systems with silent moves (extended abstract)*. In E. Best, editor: Proceedings *CONCUR'93,* 4^{th} International Conference on *Concurrency Theory,* Hildesheim, Germany, August 1993, LNCS 715, Springer, pp. 66–81, doi:10.1007/3-540-57208-2_6.

[18] R.J. van Glabbeek (1994): *On the expressiveness of ACP (extended abstract)*. In A. Ponse, C. Verhoef & S.F.M. van Vlijmen, editors: Proceedings First Workshop on the *Algebra of Communicating Processes,* ACP94, Utrecht, The Netherlands, May 1994, Workshops in Computing, Springer, pp. 188–217. Available at `http://theory.stanford.edu/~rvg/abstracts.html#31`.

[19] Daniele Gorla (2010): *A taxonomy of process calculi for distribution and mobility*. *Distributed Computing* 23(4), pp. 273–299, doi:10.1007/s00446-010-0120-6.

[20] Daniele Gorla (2010): *Towards a unified approach to encodability and separation results for process calculi*. *Information and Computation* 208(9), pp. 1031–1053, doi:10.1016/j.ic.2010.05.002.

[21] B. Haagensen, S. Maffeis & I. Phillips (2008): *Matching Systems for Concurrent Calculi*. *Electr. Notes Theor. Comput. Sci.* 194(2), pp. 85–99, doi:10.1016/j.entcs.2007.11.004.

[22] M. Hennessy & G.D. Plotkin (1980): *A term model for CCS*. In P. Dembiński, editor: Proc. 9^{th} Symposium on *Mathematical Foundations of Computer Science*, LNCS 88, Springer, pp. 261–274, doi:10.1007/BFb0022510.

[23] C.A.R. Hoare (1980): *Communicating sequential processes*. In R.M. McKeag & A.M. Macnaghten, editors: *On the construction of programs – an advanced course*, Cambridge University Press, pp. 229–254.

[24] C.A.R. Hoare (1985): *Communicating Sequential Processes*. Prentice Hall, Englewood Cliffs.

[25] R. Milner (1990): *Operational and algebraic semantics of concurrent processes*. In J. van Leeuwen, editor: *Handbook of Theoretical Computer Science*, chapter 19, Elsevier Science Publishers B.V. (North-Holland), pp. 1201–1242. Alternatively see *Communication and Concurrency*, Prentice-Hall, Englewood Cliffs, 1989, of which an earlier version appeared as *A Calculus of Communicating Systems*, LNCS 92, Springer, 1980.

[26] U. Nestmann (2000): *What is a "Good" Encoding of Guarded Choice? Inf. Comput.* 156(1-2), pp. 287–319, doi:10.1006/inco.1999.2822.

[27] U. Nestmann (2006): *Welcome to the Jungle: A Subjective Guide to Mobile Process Calculi*. In C. Baier & H. Hermanns, editors: Proceedings 17th International Conference on *Concurrency Theory,* CONCUR 2006, Bonn, Germany, August 2006, LNCS 4137, Springer, pp. 52–63, doi:10.1007/11817949_4.

[28] U. Nestmann & B.C. Pierce (2000): *Decoding Choice Encodings*. *Inf. Comput.* 163(1), pp. 1–59, doi:10.1006/inco.2000.2868.

[29] E.-R. Olderog & C.A.R. Hoare (1986): *Specification-oriented semantics for communicating processes*. *Acta Informatica* 23, pp. 9–66, doi:10.1007/BF00268075.

[30] C. Palamidessi (2003): *Comparing The Expressive Power Of The Synchronous And Asynchronous Pi-Calculi*. *Mathematical Structures in Computer Science* 13(5), pp. 685–719, doi:10.1017/S0960129503004043.

[31] C. Palamidessi, V.A. Saraswat, F.D. Valencia & B Victor (2006): *On the Expressiveness of Linearity vs Persistence in the Asychronous Pi-Calculus*. In: Proceedings 21th IEEE Symposium on *Logic in Computer Science* (LICS 2006), August 2006, Seattle, WA, USA, IEEE Computer Society, pp. 59–68, doi:10.1109/LICS.2006.39.

[32] C. Palamidessi & F.D. Valencia (2005): *Recursion vs Replication in Process Calculi: Expressiveness*. *Bulletin of the EATCS* 87, pp. 105–125.

[33] J. Parrow (2000): *Trios in concert*. In G.D. Plotkin, C. Stirling & M. Tofte, editors: *Proof, Language, and Interaction, Essays in Honour of Robin Milner*, The MIT Press, pp. 623–638.

[34] J. Parrow (2008): *Expressiveness of Process Algebras*. *Electr. Notes Theor. Comput. Sci.* 209, pp. 173–186, doi:10.1016/j.entcs.2008.04.011.

[35] K. Peters & U. Nestmann (2012): *Is It a "Good" Encoding of Mixed Choice?* In L. Birkedal, editor: Proceeding 15th International Conference on *Foundations of Software Science and Computational Structures,* FOSSACS 2012; held as part of the *European Joint Conferences on Theory and Practice of Software,* ETAPS 2012, Tallinn, Estonia, March/April 2012, LNCS 7213, Springer, pp. 210–224, doi:10.1007/978-3-642-28729-9_14.

[36] K. Peters, J.-W. Schicke & U. Nestmann (2011): *Synchrony vs Causality in the Asynchronous Pi-Calculus*. In B. Luttik & F. Valencia, editors: Proceedings 18th International Workshop on *Expressiveness in Concurrency*, *EPTCS* 64, pp. 89–103, doi:10.4204/EPTCS.64.7.

[37] I. Phillips & M.G. Vigliotti (2006): *Leader election in rings of ambient processes*. *Theor. Comput. Sci.* 356(3), pp. 468–494, doi:10.1016/j.tcs.2006.02.004.

[38] I. Phillips & M.G. Vigliotti (2008): *Symmetric electoral systems for ambient calculi*. *Inf. Comput.* 206(1), pp. 34–72, doi:10.1016/j.ic.2007.08.005.

[39] R. de Simone (1985): *Higher-level synchronising devices in* MEIJE*-SCCS*. *Theoretical Computer Science* 37, pp. 245–267, doi:10.1016/0304-3975(85)90093-3.

[40] C. Stirling (1987): *Modal logics for communicating systems*. *Theoretical Computer Science* 49, pp. 311–347, doi:10.1016/0304-3975(87)90012-0.

[41] F.W. Vaandrager (1993): *Expressiveness Results for Process Algebras*. In J.W. de Bakker, W.P. de Roever & G. Rozenberg, editors: *Proceedings REX Workshop on Semantics: Foundations and Applications*, Beekbergen, The Netherlands, June 1992, LNCS 666, Springer, pp. 609–638, doi:10.1007/3-540-56596-5_49.

[42] C. Versari, N. Busi & R. Gorrieri (2009): *An expressiveness study of priority in process calculi*. *Mathematical Structures in Computer Science* 19(6), pp. 1161–1189, doi:10.1017/S0960129509990168.

[43] M.G. Vigliotti, I. Phillips & C. Palamidessi (2007): *Tutorial on separation results in process calculi via leader election problems*. *Theor. Comput. Sci.* 388(1-3), pp. 267–289, doi:10.1016/j.tcs.2007.09.001.

[44] D.J. Walker (1990): *Bisimulation and divergence*. *Information and Computation* 85(2), pp. 202–241, doi:10.1016/0890-5401(90)90048-M.