A Theory of Encodings and Expressiveness

Rob van Glabbeek*

School of Informatics, University of Edinburgh, UK
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 well-known case study: the encoding of the synchronous in the asynchronous π-calculus. Several applications of this theory to the relative expressiveness of CSP, ACP, CCS and the π-calculus are reviewed.

1 Introduction

This paper, like [53], aims at answering the question what it means for one language to encode another one, and making the resulting definition applicable to order system description languages like CCS, CSP, ACP and the π-calculus with respect to their expressive power.

To this end it proposes a unifying concept of valid translation between two languages *up to* a semantic equivalence or preorder $\mathrel{\wasysym}$. Roughly, a valid translation up to $\mathrel{\wasysym}$ of a source language $\mathscr{L}$ into a target language $\mathscr{L}'$ is a mapping from the expressions of $\mathscr{L}$ to those of $\mathscr{L}'$ that preserves their meaning, i.e. such that the meaning of the translation of an expression is semantically equivalent to the meaning of the expression being translated. This requires a semantic equivalence (or preorder) $\mathrel{\wasysym}$ that is meaningful for both $\mathscr{L}$ and $\mathscr{L}'$. Additionally, or implicitly, the translation is also required to be *compositional*. This means that the translation of a composed expression is completely determined by the translations of the argument expressions and a translation of the composition mechanism.

Languages can be ordered by their expressiveness up to the chosen equivalence or preorder $\mathrel{\wasysym}$ according to the existence of valid translations between them. When considering multiple choices of $\mathrel{\wasysym}$, one obtains multiple expressiveness hierarchies between languages. If one language is at least as expressive as another up to a certain equivalence, it is certainly at least as expressive up to a coarser, or less discriminating, equivalence. This way, the choice of $\mathrel{\wasysym}$ is a measure for the quality of the translation, and thereby for the degree in which the source language can be expressed in the target. Any language is as expressive as any other up to the universal relation, whereas almost no two languages are equally expressive up to the identity relation.

The concept of a valid translation between system description languages (or *process calculi*) was first formally defined by Boudol [13], and has been generalised by me in [39, 42]. The present paper contributes a further generalisation. These generalisations concern the class of languages to which the definition is applicable, and the class of relations $\mathrel{\wasysym}$ that may be used as parameter.

My definition applies to languages whose semantics interprets the operators and recursion constructs as operations on a set of values, called a *domain*. This includes the special case of *closed-term languages*, in which the domain in which the language is interpreted consists of the closed expressions from the language itself. This special case was the sole focus of [13, 53] and most other work in this area.

*Supported by Royal Society Wolfson Fellowship RSWF\R1\221008

An extended abstract of this paper appeared in the proceedings of FoSSaCS 2018 [46].

As in [42], my aim is to generalise the concept of a valid 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 exist. At the same time, I try 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.

Contributions This paper contributes a generalisation of the notion of a valid translation up to $\backsim$ from [13, 39, 42], and compares it with other notions of valid translation found in the literature. A language $\mathscr{L}'$ is said to be at least as expressive as a language $\mathscr{L}$ up to $\backsim$ iff there exists a valid translation up to $\backsim$ of $\mathscr{L}$ into $\mathscr{L}'$. I show that the relation "being at least as expressive as" is a preorder on languages.

I contribute five general theorems on valid translations. The first, a *congruence transfer property*, says that if $\mathscr{L}'$ is at least as expressive as $\mathscr{L}$ up to $\backsim$, and $\approx$ is a congruence for $\mathscr{L}'$ that is coarser than or equal to $\backsim$, then $\approx$ also is a congruence for $\mathscr{L}$. This theorem is useful for separation results. It implies, for instance, that there exists no encoding of CCS into CSP that is valid up to strong bisimilarity. Here I employ this general theorem with $\backsim$ in the rôle of strong bisimilarity and $\approx$ in the role of weak bisimilarity, using that weak bisimilarity fails to be a congruence for the $+$ of CCS.

The second theorem describes a congruence transfer property that goes in the other direction: if $\mathscr{T}$ is a translation of $\mathscr{L}$ into $\mathscr{L}'$ that is valid up to $\backsim$, and $\approx$ is a congruence for $\mathscr{L}$ that is coarser than or equal to $\backsim$, then $\approx$ also is a congruence for the fragment of $\mathscr{L}'$ that is obtained as the image of $\mathscr{T}$.

My main result is a *congruence closure property* for valid translations: if a translation $\mathscr{T}$ between $\mathscr{L}$ and $\mathscr{L}'$ is valid up to a semantic equivalence $\approx$, then it is valid even up to an equivalence that

- on $\mathscr{L}$ coincides with the congruence closure of $\approx$
- on the image of $\mathscr{L}$ within $\mathscr{L}'$ also coincides with the congruence closure of $\approx$, and
- melts each equivalence class of $\mathscr{L}$ with exactly one of $\mathscr{L}'$.

As an illustration, I recall a translation by Boudol from a synchronous into an asynchronous fragment of the π-calculus, which has been shown to be valid up to weak barbed bisimilarity $\dot{\approx}$. It might be argued that this is a weak result, as $\dot{\approx}$ is a rather course semantic equivalence. However, $\dot{\approx}$ has been used in the literature as method to define or characterise meaningful equivalences on process calculi, namely as the congruence closures of $\dot{\approx}$. The familiar notion of weak bisimulation congruence on CCS arises as the congruence closure of $\dot{\approx}$ [74], as does the familiar notion of early weak congruence on the π-calculus [101]. Consequently, the above congruence closure theorem lifts this weak result about Boudol's translation to a stronger one.

Another general theorem states that, under some mild side conditions, if a translation $\mathscr{T}$ between $\mathscr{L}$ and $\mathscr{L}'$ is valid up to $\approx$, and $\sim$ is a congruence for $\mathscr{L}'$ that is finer than or equal to $\approx$, then $\mathscr{T}$ is even valid up to a an extension of $\sim$ to the disjoint union of $\mathscr{L}$ and $\mathscr{L}'$, still finer than or equal to $\approx$, that also is a congruence for $\mathscr{L}$. I will refer to this result as *congruence reflection*. Its application is in combining the features of two languages by translating them into a common target language.

For languages that do not feature variables ranging over the domain in which the language is interpreted, I require valid translations to be compositional. However, any language can be upgraded with variables at no extra cost, and for languages $\mathscr{L}$ and $\mathscr{L}'$ with variables I define a translation from $\mathscr{L}$ to $\mathscr{L}'$ as function that maps the *open terms* of $\mathscr{L}$ to semantically equivalent open terms of $\mathscr{L}'$. This requirement alone entails compositionality: if $\mathscr{L}'$ is at least as expressive as $\mathscr{L}$, i.e., if there exists a valid translation from $\mathscr{L}$ to $\mathscr{L}'$, then there exists such a valid translation that moreover is compositional.

Overview of the paper Section 2 formally defines the concept of a language and a translation between languages, states that *validity* of translations is the central concept that this paper and many others aim to describe, and defines expressiveness in terms of validity. Subsequently, I propose a notion of validity, and prove the above-mentioned theorems about it, in four stages.

Section 3 deals with closed-term languages without variables, and without recursion or other non-traditional term-building constructs. Here the definition of validity, which incorporates compositionality, as well as the proofs of the promised theorems, is fairly straightforward. Section 4 illustrates my approach on a well-known case study: the encoding of a synchronous into an asynchronous fragment of the π-calculus. Section 5 argues that my congruence closure result allows verifications in a source language of a valid translation to be mimicked in the target language, and that one needs congruence reflection for combining the features of multiple languages by translating them into a common target.

Sections 6–9 generalise this work to closed-term languages with variables, still without variable binding or term-building constructs other than n-ary operators. Section 6 deals with compositionality, and Section 7 with the definition of validity. Here I find eight plausible generalisations of the notion of validity from Section 3, of which, after some deliberation on their properties, I choose one to continue with. One of the reasons for choosing this notion is that compositionality is not incorporated in the definition, but comes for free, in the sense that any valid translation up to a preorder $\mathrel{\wr\!\sim}$ can be modified into such a translation that moreover is compositional. Section 9 extends the other general theorems to this chosen notion of validity. Section 8 discusses the difference between absolute and relative expressiveness. My paper generally deals with relative expressiveness, but absolute expressiveness can be captured by one of the alternative notions of validity from Section 7.

Section 10 generalises the work to arbitrary closed-term languages, thus dealing with variable binding, recursion, operators with infinitely many arguments, and possible term-building constructs that are yet to be invented, provided they obey a simple postulate. The last stage is Section 11, which lifts the restriction to closed-term languages. It is possible to jump straight from Section 2 to Section 11, reading in between only Section 10 prior to 10.6.

Section 12 studies the generalisation to arbitrary languages of most of the alternative notions of validity contemplated in Section 7. Two of those have been studied earlier in [42], and can be seen as special cases of the notion of validity of the present paper. It is in this sense that the present paper generalises the work in [42].

Section 13 speculates on suitable choices for $\mathrel{\wr\!\sim}$ when comparing the expressiveness of process calculi. In particular, it advocates notions of *barbed bisimilarity*, based on a collection of *barbs* that is external to the calculi under consideration. Section 14 reviews applications of this work to the relative expressiveness of several versions of CSP, ACP, CCS and the π-calculus. It also generalises the notion of being as least as expressive up to $\mathrel{\wr\!\sim}$ from a relation between languages to one between parametrised languages, which are families of languages that differ only in the choice of a parameter, such as a set of atomic actions. Section 15 compares my approach with *full abstraction*, and with the approach of Gorla [53].

Acknowledgement I am grateful to the referees of this paper for their insightful and thorough feedback. It is thanks to them that this version of my paper is easier to read then the original one [45], and includes the motivations of its design decisions.

2 Languages, valid translations, quality criteria 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.

Following [42], 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 mapping $[\![\ ]\!]_{\mathscr{L}} : \mathbb{T}_{\mathscr{L}} \to \mathscr{D}_{\mathscr{L}}$ from $\mathbb{T}_{\mathscr{L}}$ in some set of meanings $\mathscr{D}_{\mathscr{L}}$.

In this paper, I consider single-sorted languages $\mathscr{L}$ in which *expressions* or *terms* are built from variables (taken from an infinite set $\mathscr{X}$) by means of operators (including constants) and additional (e.g. recursion) constructs. Since my main interest is in process calculi, where the expressions of $\mathscr{L}$ denote processes, I sometimes refer to the expressions of $\mathscr{L}$ as *process expressions*, and to the variables from $\mathscr{X}$ as *process variables*. Even so, the proposed framework is equally well applicable to other settings.

Some process calculi employ multiple types of processes. AWN [27], for instance, features sequential processes, parallel processes and networks. Other calculi, such as mCRL2 [56], have only one type of processes but multiple types of *data*. An *action* in mCRL2 can be parametrised with data it carries, and therefore is modelled not as a constant of type process, but as a function from some types of data to the type of processes. Although in this paper I do not provide definitions of translations between such multi-sorted languages, I expect that this can be done along the same lines as proposed here.

Let $\mathscr{L}$ be a language that features *variables*, *operators* f of *arity* $n \in \mathbb{N}$, and possibly additional constructs. The set $\mathbb{T}_{\mathscr{L}}$ of *$\mathscr{L}$-expressions* is inductively defined by:

(i) $X \in \mathbb{T}_{\mathscr{L}}$ for each variable $X \in \mathscr{X}$,

(ii) $f(E_1,\ldots,E_n) \in \mathbb{T}_{\mathscr{L}}$ for each n-ary operator f and expressions $E_i \in \mathbb{T}_{\mathscr{L}}$ for $i = 1,\ldots,n$,

(iii) and clauses for the additional constructs, if any.

Examples of additional constructs are the infinite summation $\sum_{i\in I} E_i$ of CCS, which takes arbitrarily many arguments, or the recursion construct $\mu X.E$, that has one argument E and *binds* all occurrences of the variable X in that argument. Another example is the *choice quantifier* $\sum_{x\in D} E$ [65] of mCRL2 [56]. Although this operator has the same shape and serves the same purpose as the infinite sum of CCS, it has only one argument E, and binds a data variable x that may occur in E. In this paper I consider a variant $\sum_{h\in\mathscr{H}} E[f_h]$ of choice quantification that can be added to single sorted languages like CCS. It employs a possibly infinite family $[f_h]_{h\in\mathscr{H}}$ of unary renaming operators, and has only a single argument E. Semantically it behaves the same as the infinite choice between all processes $E[f_h]$ for $h \in \mathscr{H}$.

Definition 2.1 ([42]) A *translation* from a language $\mathscr{L}$ into a language $\mathscr{L}'$ is a mapping $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$.

I will use the word *encoding* as a synonym for translation. In almost all work on (relative) expressiveness, including [13, 107, 39, 99, 12, 8, 76, 78, 83, 20, 80, 6, 7, 82, 77, 81, 93, 19, 18, 57, 84, 94, 9, 17, 109, 52, 53, 64, 92, 89, 42, 91, 1, 88, 58, 59, 108, 33, 34, 35, 36, 90, 2, 44, 49, 37, 63, 87, 47, 24, 48, 102], the key ingredient is some notion of *validity* of encodings. A translation is *valid* if and only if it satisfies certain *quality criteria*. The choice of these quality criteria varies from one paper to another, and is often adapted to the expressiveness result a paper sets out to prove. The following definition is a common understanding of virtually all papers on expressiveness.

Definition 2.2 Language $\mathscr{L}'$ is at least as *expressive* as $\mathscr{L}$ iff a valid translation from $\mathscr{L}$ into $\mathscr{L}'$ exists.

This way, *expressiveness results* can be obtained by exhibiting a valid translation between two languages, and *separation results* by giving an argument that no valid translation between two given languages exists.

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. For this reason, I use a single set $\mathscr{X}$ of variables for all languages, except for languages that do not feature process variables at all.

The set $fv(E) \subseteq \mathscr{X}$ of variables that occur *free* in an expression $E \in \mathbb{T}_{\mathscr{L}}$ is defined inductively by

(i) $fv(X) = \{X\}$ for $X \in \mathscr{X}$,
(ii) $fv(f(E_1,\ldots,E_n)) = \bigcup_{i=1}^{n} fv(E_i)$,
(iii) and dedicated clauses for each of the additional constructs, for instance $fv(\sum_{i\in I} E_i) = \bigcup_{i\in I} fv(E_i)$, $fv(\mu X.E) = fv(E)\backslash\{X\}$ and $fv(\sum_{h\in\mathscr{H}} E[f_h]) = fv(E)$.

An expression E is *closed* if $fv(E) = \emptyset$. Let $\mathrm{T}_{\mathscr{L}} \subseteq \mathbb{T}_{\mathscr{L}}$ denote the set of closed terms of $\mathscr{L}$.

3 Closed-term languages without variables or additional constructs

In this section I present my theory of encodings and expressiveness for languages $\mathscr{L} = (\mathbb{T}_{\mathscr{L}}, [\![\ ]\!]_{\mathscr{L}})$ that are special in three ways:[1]

(i) They have no process variables. So $\mathbb{T}_{\mathscr{L}} = \mathrm{T}_{\mathscr{L}}$.
(ii) They do not feature any additional syntactic constructs (beyond operators).
(iii) $\mathscr{D}_{\mathscr{L}} = \mathrm{T}_{\mathscr{L}}$ and $[\![\ ]\!]_{\mathscr{L}} : \mathrm{T}_{\mathscr{L}} \to \mathrm{T}_{\mathscr{L}}$ is the identity function.

Languages satisfying (iii) are called *closed-term languages*. In such languages there is no separation between syntax and semantics. Here I call languages satisfying (i)-(iii) *simple*.

3.1 Compositionality for languages without variables or additional constructs

Let $\mathscr{X}_n = \{X_1, X_2, ..., X_n\}$ be a set of n specific variables, called *holes*. Given a language $\mathscr{L}$, let $\mathscr{L}[\mathscr{X}_n]$ be the extension of $\mathscr{L}$ with the variables from $\mathscr{X}_n$.[2] An n-ary *$\mathscr{L}$-context* C is an expression in the language $\mathscr{L}[\mathscr{X}_n]$, i.e., $C \in \mathbb{T}_{\mathscr{L}[\mathscr{X}_n]}$. A ternary context C is often introduced as $C[\ ,\ ,\ ]$ to remind the reader of the 3 holes that may occur in it. Each hole may occur in C multiple times, or not at all. For $C \in \mathbb{T}_{\mathscr{L}[\mathscr{X}_n]}$ and $E_i \in \mathbb{T}_{\mathscr{L}}$ for each $i = 1,\ldots,n$ one writes $C[E_1,\ldots,E_n]$ for the result of substituting E_i for X_i in C, for each $i = 1,\ldots,n$.

Definition 3.1 Let $\mathscr{L}$ and $\mathscr{L}'$ be languages without variables or additional syntactic constructs (beyond operators). A translation $\mathscr{T}$ from $\mathscr{L}$ into a language $\mathscr{L}'$ is *compositional* if for each n-ary operator f of $\mathscr{L}$ there exists an n-ary $\mathscr{L}'$-context C_f such that $\mathscr{T}(f(E_1,\ldots,E_n)) = C_f[\mathscr{T}(E_1),\ldots,\mathscr{T}(E_n)]$.

Proposition 3.2 Let $\mathscr{L}$ and $\mathscr{L}'$ be languages without variables or additional syntactic constructs (beyond operators). A translation $\mathscr{T}$ from $\mathscr{L}$ into a language $\mathscr{L}'$ is compositional iff for each n-ary context D of $\mathscr{L}$ there exists an n-ary $\mathscr{L}'$-context C such that $\mathscr{T}(D[E_1,\ldots,E_n]) = C[\mathscr{T}(E_1),\ldots,\mathscr{T}(E_n)]$.

Proof: "If" follows by taking the context $D := f(X_1,\ldots,X_n)$ for a given n-ary operator f of $\mathscr{L}$.
"Only if" follows by structural induction on D.

Let $D = X_i \in \mathscr{X}_n$. Then $D[E_1,\ldots,E_n] = E_i$. Take $C := D = X_i$. Then

$$\mathscr{T}(D[E_1,\ldots,E_n]) = \mathscr{T}(E_i) = C[\mathscr{T}(E_1),\ldots,\mathscr{T}(E_n)].$$

[1] The material in Section 3.1 only assumes (i) and (ii).
[2] If $\mathscr{L}$ already employs a set of variables $\mathscr{X}$, one takes $\mathscr{X}_n \cap \mathscr{X} = \emptyset$.

Let $D = f(D_1,\ldots,D_k)$ for a k-ary operator f of $\mathscr{L}$. By induction, for each subcontext D_i of D there exists a $\mathscr{L}'$-context C_i such that $\mathscr{T}(D_i[E_1,\ldots,E_n]) = C_i[\mathscr{T}(E_1),\ldots,\mathscr{T}(E_n)]$. Take $C := C_f(C_1,\ldots,C_k)$. Then
$$\begin{aligned}\mathscr{T}(D[E_1,\ldots,E_n]) &= \mathscr{T}(f(D_1[E_1,\ldots,E_n],\ldots,D_k[E_1,\ldots,E_n]))\\ &= C_f[\mathscr{T}(D_1[E_1,\ldots,E_n]),\ldots,\mathscr{T}(D_k[E_1,\ldots,E_n])]\\ &= C_f[C_1[\mathscr{T}(E_1),\ldots,\mathscr{T}(E_n)],\ldots,C_k[\mathscr{T}(E_1),\ldots,\mathscr{T}(E_n)]] = C[\mathscr{T}(E_1),\ldots,\mathscr{T}(E_n)].\ \square\end{aligned}$$

Any compositional translation between languages $\mathscr{L}$ and $\mathscr{L}'$ without variables or additional syntactic constructs trivially generalises to a compositional translation between the extensions $\mathscr{L}[\mathscr{X}_n]$ and $\mathscr{L}'[\mathscr{X}_n]$ of these languages with variables (or holes). The generalisation is defined by $\mathscr{T}(X) := X$ for each $X \in \mathscr{X}_n$, and for the rest is completely determined by the requirement of compositionality. From this perspective, the context C of Proposition 3.2 can be characterised as $\mathscr{T}(D)$.

Corollary 3.3 Let $\mathscr{L},\mathscr{L}'$ be languages without variables or additional constructs. A translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ is compositional iff $\mathscr{T}(D[E_1,...,E_n]) = \mathscr{T}(D)[\mathscr{T}(E_1),...,\mathscr{T}(E_n)]$ for each n-ary $\mathscr{L}$-context D.

3.2 Validity for simple languages

In order to compare two simple languages $\mathscr{L}$ and $\mathscr{L}'$ w.r.t. their expressive power, I normally employ a semantic equivalence or preorder $\mathrel{\scriptstyle\sim}$ that is defined (at least) on $\mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{L}'}$. Intuitively, $P' \mathrel{\scriptstyle\sim} P$ with $P \in \mathrm{T}_{\mathscr{L}}$ and $P' \in \mathrm{T}_{\mathscr{L}'}$ means that the processes P and P' are sufficiently alike for our purposes, so that one can accept a translation of P into P'.

I will be chiefly interested in the case that $\mathrel{\scriptstyle\sim}$ is an equivalence—hence the choice of a symbol that looks like $\sim$. However, to establish Observation 3.6 and Theorem 3.7 below, it suffices to know that $\mathrel{\scriptstyle\sim}$ is reflexive and transitive. My convention is that the dotted end of $\mathrel{\scriptstyle\sim}$ points to a translation and the other end to an original—without offering an intuition for the possible asymmetry.[3]

For many process calculi, a labelled transition relation between their closed expressions is defined, so that the expressions in $\mathrm{T}_{\mathscr{L}}$ and $\mathrm{T}_{\mathscr{L}'}$ can be seen as states in labelled transition systems (LTSs). A candidate for $\mathrel{\scriptstyle\sim}$ could be any semantic equivalence on LTSs; by taking the disjoint union of two LTSs, such an equivalence also relate states from different LTSs, and thus is defined on $\mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{L}'}$.

Although for some applications the disjoint union may be appropriate, in most cases an expression that exists in both $\mathrm{T}_{\mathscr{L}}$ and $\mathrm{T}_{\mathscr{L}'}$ will have the same meaning in both languages. This applies in particular when $\mathscr{L}'$ is a sublanguage of $\mathscr{L}$. In such cases, $\mathrel{\scriptstyle\sim}$ needs to be defined on $\mathrm{T}_{\mathscr{L}} \cup \mathrm{T}_{\mathscr{L}'}$.

Given a translation $\mathscr{T}$, it is strictly speaking not necessary that $\mathrel{\scriptstyle\sim}$ is defined on all of $\mathrm{T}_{\mathscr{L}'}$. Writing $\mathrm{T}_{\mathscr{T}(\mathscr{L})}$ for $\{\mathscr{T}(P) \in \mathrm{T}_{\mathscr{L}'} \mid P \in \mathrm{T}_{\mathscr{L}}\}$, it suffices that $\mathrel{\scriptstyle\sim}$ is defined on $\mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{T}(\mathscr{L})}$.

Definition 3.4 A translation $\mathscr{T}$ is *valid* up to $\mathrel{\scriptstyle\sim}$ iff it is compositional and $\mathscr{T}(P) \mathrel{\scriptstyle\sim} P$ for each $P \in \mathrm{T}_{\mathscr{L}}$.

3.3 A hierarchy of expressiveness preorders on simple languages

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

Observation 3.5 Let $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be a translation from $\mathscr{L}$ into $\mathscr{L}'$, with $\mathscr{L},\mathscr{L}'$ simple languages, and let $\mathrel{\scriptstyle\sim}$ be finer than $\mathrel{\approx}$. If $\mathscr{T}$ is valid up to $\mathrel{\scriptstyle\sim}$, then it is also valid up to $\mathrel{\approx}$.

The quality of a translation depends on the choice of the equivalence or preorder up to which it is valid. Any two languages are equally expressive up to the universal equivalence, relating any two processes. Hence, the equivalence or preorder needs to be chosen carefully to match the intended applications of

[3] In line with this, forthcoming relations **R** between concepts at the source and target of a translation will be systematically oriented with the target concepts on the left—see e.g. Definition 7.3.

the languages under comparison. In general, as shown by Observation 3.5, using a finer equivalence or preorder 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 or preorder yields a stronger claim.

Observation 3.6 The identity is a valid translation up to any preorder from any language into itself.

Theorem 3.7 If valid translations up to $\precsim$ exists from $\mathscr{L}$ into $\mathscr{L}'$ and from $\mathscr{L}'$ into $\mathscr{L}''$, then there is a valid translation up to $\precsim$ from $\mathscr{L}$ into $\mathscr{L}''$.

Proof: Let $\mathscr{T}_1 : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be a valid translation up to $\precsim$ from $\mathscr{L}$ to $\mathscr{L}'$, and let $\mathscr{T}_2 : \mathbb{T}_{\mathscr{L}'} \to \mathbb{T}_{\mathscr{L}''}$ be one from $\mathscr{L}'$ to $\mathscr{L}''$. I will show that the translation $\mathscr{T} := \mathscr{T}_2 \circ \mathscr{T}_1 : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}''}$ from $\mathscr{L}$ into $\mathscr{L}''$, given by $\mathscr{T}(P) = \mathscr{T}_2 \circ \mathscr{T}_1(P) = \mathscr{T}_2(\mathscr{T}_1(P))$, is valid up to $\precsim$.

Let D be an n-ary $\mathscr{L}$-context. Since $\mathscr{L}_1$ and $\mathscr{L}_2$ are compositional, by Proposition 3.2 there is an $\mathscr{L}'$-context C such that $\mathscr{T}_1(D[P_1,\ldots,P_n]) = C[\mathscr{T}_1(P_1),\ldots,\mathscr{T}_1(P_n)]$, and hence an $\mathscr{L}''$-context B such that $\mathscr{T}_2(C[Q_1,\ldots,Q_n]) = B[\mathscr{T}_2(Q_1),\ldots,\mathscr{T}_2(Q_n)]$. Thus

$$\mathscr{T}(D[P_1,\ldots,P_n]) = \mathscr{T}_2(\mathscr{T}_1(D[P_1,\ldots,P_n])) = \mathscr{T}_2(C[\mathscr{T}_1(P_1),\ldots,\mathscr{T}_1(P_n)]) = B[\mathscr{T}_2(\mathscr{T}_1(P_1)),\ldots,\mathscr{T}_2(\mathscr{T}_1(P_n))] = B[\mathscr{T}(P_1)),\ldots,\mathscr{T}(P_n))],$$

so $\mathscr{T}$ is compositional. Furthermore $\mathscr{T}(P) = \mathscr{T}_2(\mathscr{T}_1(P)) \precsim \mathscr{T}_1(P) \precsim P$, using the transitivity of $\precsim$. □

Theorem 3.7 and Observation 3.6 show that the relation "being at least as expressive as up to $\precsim$" is a preorder on simple languages.

3.4 Congruence transfer properties for simple languages

Definition 3.8 Let $\mathscr{L}$ be a simple language. An equivalence relation $\sim$ on $\mathrm{T}_{\mathscr{L}}$ is a *congruence* for $\mathscr{L}$ if

$$P_1 \sim Q_1 \wedge \ldots \wedge P_n \sim Q_n \Rightarrow f(P_1,\ldots,P_n) \sim f(Q_1,\ldots,Q_n)$$

for each n-ary operator f of $\mathscr{L}$, and each $P_i, Q_i \in \mathrm{T}_{\mathscr{L}}$ $(i = 1,\ldots,n)$.

Proposition 3.9 Let $\mathscr{L}$ be a simple language. An equivalence relation $\sim$ on $\mathrm{T}_{\mathscr{L}}$ is a congruence iff

$$P_1 \sim Q_1 \wedge \ldots \wedge P_n \sim Q_n \Rightarrow C[P_1,\ldots,P_n] \sim C[Q_1,\ldots,Q_n]$$

for each n-ary $\mathscr{L}$-context C, and each $P_i, Q_i \in \mathrm{T}_{\mathscr{L}}$ $(i = 1,\ldots,n)$.

Proof: Exactly as in the proof of Proposition 3.2, "if" follows by taking the context $C := f(X_1,\ldots,X_n)$, and "only if" by a straightforward structural induction on C. □

Proposition 3.10 Let $\mathscr{L}$ be a simple language. An equivalence relation $\sim$ on $\mathrm{T}_{\mathscr{L}}$ is a congruence iff

$$P \sim Q \Rightarrow C[P] \sim C[Q]$$

for each unary $\mathscr{L}$-context C, and each $P, Q \in \mathrm{T}_{\mathscr{L}}$.

Proof: "Only if" is a trivial consequence of Proposition 3.9.

"If" follows by the transitivity of $\sim$. Let $P_1 \sim Q_1 \wedge \ldots \wedge P_n \sim Q_n$. Then

$$C[P_1,P_2,\ldots,P_n] \sim C[Q_1,P_2\ldots,P_n] \sim C[Q_1,Q_2\ldots,P_n] \sim \ldots \sim C[Q_1,Q_2,\ldots,Q_n].$$

Here the first step follows since $P_1 \sim Q_1 \Rightarrow D[P_1] \sim D[Q_1]$ where $D := C[\ ,P_2,\ldots,P_n]$ is the unary context obtained from C by filling in P_i for X_i for $i = 2,...,n$. Likewise, the second step follows by taking the unary context $C[Q_1,\ \ ,P_3,\ldots,P_n]$, and so on □

The following result is a handy tool in proving separation results.

Theorem 3.11 Suppose $\mathscr{L}'$ is at least as expressive as $\mathscr{L}$ up to an equivalence or preorder $\lesssim$, and let $\approx \supseteq \lesssim$ be a congruence for $\mathscr{L}'$ that is coarser than or equal to $\lesssim$. Then $\approx$ is also a congruence for $\mathscr{L}$.

Proof: By definition, there is a translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ that is valid up to $\lesssim$, and thus certainly up to $\approx$. Let $P,Q \in \mathrm{T}_{\mathscr{L}}$ with $P \approx Q$ and let D be a unary $\mathscr{L}$-context. Let $C := \mathscr{T}(D)$ – see Corollary 3.3. Then $\mathscr{T}(P) \approx P \approx Q \approx \mathscr{T}(Q)$, so $D[P] \approx \mathscr{T}(D[P]) \stackrel{3.3}{=} C[\mathscr{T}(P)] \approx C[\mathscr{T}(Q)] \stackrel{3.3}{=} \mathscr{T}(D[Q]) \approx D[Q]$. □

I also contribute a reverse direction of this congruence transfer property, but its validity is limited to the image of $\mathscr{L}$ under a given translation $\mathscr{T}$. Using Proposition 3.10 as a guideline for defining a congruence for the target language $\mathscr{L}'$ of $\mathscr{T}$, I weaken this concept to a congruence for the image $\mathscr{T}(\mathscr{L})$ under $\mathscr{T}$ of the source language $\mathscr{L}$, by limiting the expressions $P',Q' \in \mathrm{T}_{\mathscr{L}'}$ that figure in this definition to those of the form $\mathscr{T}(P)$ and $\mathscr{T}(Q)$, and the contexts C to those of the form $\mathscr{T}(D)$.

Definition 3.12 Let $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be a translation from $\mathscr{L}$ into $\mathscr{L}'$. An equivalence relation $\sim$ on $\mathrm{T}_{\mathscr{T}(\mathscr{L})}$ is a *congruence for* $\mathscr{T}(\mathscr{L})$ if

$$\mathscr{T}(P) \sim \mathscr{T}(Q) \Rightarrow \mathscr{T}(D)[\mathscr{T}(P)] \sim \mathscr{T}(D)[\mathscr{T}(Q)]$$

for each unary $\mathscr{L}$-context D, and each $P,Q \in \mathrm{T}_{\mathscr{L}}$.

Theorem 3.13 Let $\mathscr{T}$ be a translation of $\mathscr{L}$ into $\mathscr{L}'$ that is valid up to $\lesssim$ and let $\approx \supseteq \lesssim$ be a congruence for $\mathscr{L}$ that is coarser than or equal to $\lesssim$. Then $\approx$ is also a congruence for $\mathscr{T}(\mathscr{L})$.

Proof: Let $P,Q \in \mathrm{T}_{\mathscr{L}}$ with $\mathscr{T}(P) \approx \mathscr{T}(Q)$ and let $C := \mathscr{T}(D)$ with D be a unary $\mathscr{L}$-context. Then $P \approx \mathscr{T}(P) \approx \mathscr{T}(Q) \approx Q$, so $C[\mathscr{T}(P)] \stackrel{3.3}{=} \mathscr{T}(D[P]) \approx D[P] \approx D[Q] \approx \mathscr{T}(D[Q]) \stackrel{3.3}{=} C[\mathscr{T}(Q)]$. □

Based on its proof, Theorem 3.11 can be sharpened by merely assuming that $\approx$ is a congruence for $\mathscr{T}(\mathscr{L})$.

Theorem 3.11* Let $\mathscr{T}$ be a translation of $\mathscr{L}$ into $\mathscr{L}'$ that is valid up to $\lesssim$ and let $\approx \supseteq \lesssim$ be a congruence for $\mathscr{T}(\mathscr{L})$ that is coarser than or equal to $\lesssim$. Then $\approx$ is also a congruence for $\mathscr{L}$. □

3.5 Congruence reflection for simple languages

The following result says that any congruence relation on the target language of a compositional translation can be extended to the source language in such a way that it also is a congruence on the source language, and the translation is valid up to this extended relation.

Theorem 3.14 Let $\mathscr{T}$ be a compositional translation between simple languages $\mathscr{L}$ and $\mathscr{L}'$. Then any equivalence relation $\sim \subseteq \mathrm{T}_{\mathscr{T}(\mathscr{L})} \times \mathrm{T}_{\mathscr{T}(\mathscr{L})}$ that is a congruence for $\mathscr{T}(\mathscr{L})$ can be extended to an equivalence $\sim_{\mathscr{T}}$ on $\mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{T}(\mathscr{L})}$, such that $\mathscr{T}$ is valid up to $\sim_{\mathscr{T}}$, and $\sim_{\mathscr{T}}$ also is a congruence for $\mathscr{L}$.

Moreover, if $\mathscr{T}$ is valid up to $\approx$ and $\sim \subseteq \approx$ on $\mathrm{T}_{\mathscr{T}(\mathscr{L})}$, then $\sim_{\mathscr{T}} \subseteq \approx$.

Proof: Write $P \stackrel{\mathscr{T}}{\to} R$ for $P,R \in \mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{T}(\mathscr{L})}$ if either $R = P \in \mathrm{T}_{\mathscr{L}}$ or $R = P \in \mathrm{T}_{\mathscr{T}(\mathscr{L})}$ or $P \in \mathrm{T}_{\mathscr{L}}$ and $R = \mathscr{T}(P) \in \mathrm{T}_{\mathscr{T}(\mathscr{L})}$. Define $\sim_{\mathscr{T}}$ on $\mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{T}(\mathscr{L})}$ by $P \sim_{\mathscr{T}} Q$ iff $P \stackrel{\mathscr{T}}{\to} R \sim S \stackrel{\mathscr{T}}{\leftarrow} Q$ for some $R,S \in \mathrm{T}_{\mathscr{T}(\mathscr{L})}$. This relation is reflexive, symmetric and transitive, since $\sim$ is. Thus $\sim_{\mathscr{T}}$ is an equivalence relation. On $\mathrm{T}_{\mathscr{T}(\mathscr{L})}$ the relation $\sim_{\mathscr{T}}$ coincides with $\sim$. By construction, $\mathscr{T}(P) \sim_{\mathscr{T}} P$ for all $P \in \mathrm{T}_{\mathscr{L}}$, so $\mathscr{T}$ is valid up to $\sim_{\mathscr{T}}$. That $\sim_{\mathscr{T}}$ is a congruence for $\mathscr{L}$ follows by Theorem 3.11* (taking $\approx := \lesssim := \sim_{\mathscr{T}}$).

The last statement of Theorem 3.14 follows directly from the definition of $\sim_{\mathscr{T}}$. □

3.6 A congruence closure property for translations between simple languages

For any equivalence relation $\approx$ on $\mathrm{T}_{\mathscr{L}}$ there exists a largest congruence $\approx^c_{\mathscr{L}}$ contained in $\approx$, called the *congruence closure* of $\approx$ on $\mathscr{L}$. It is characterised by $P \approx^c_{\mathscr{L}} Q$ iff $C[P] \approx C[Q]$ for every unary $\mathscr{L}$-context C, and denoted $\approx^c$ when $\mathscr{L}$ is understood.

Lemma 3.15 Let $\mathscr{T}$ be a translation of $\mathscr{L}$ into $\mathscr{L}'$ that is valid up to $\approx$. If $\mathscr{T}(P) = \mathscr{T}(Q)$ then $P \approx^c_{\mathscr{L}} Q$.

Proof: Suppose $\mathscr{T}(P) = \mathscr{T}(Q)$ for some $P, Q \in \mathrm{T}_{\mathscr{L}}$. Let D be a unary $\mathscr{L}$-context and $C := \mathscr{T}(D)$. Then $\mathscr{T}(D[P]) \stackrel{3.3}{=} C[\mathscr{T}(P)] = C[\mathscr{T}(Q)] \stackrel{3.3}{=} \mathscr{T}(D(Q))$, so $D(P) \approx \mathscr{T}(D(P)) = \mathscr{T}(D(Q)) \approx D(Q)$. □

Given a compositional translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$, I also speak of the *congruence closure* $\approx^c_{\mathscr{T}(\mathscr{L})}$ of $\approx$ on $\mathscr{T}(\mathscr{L})$. This is the largest congruence for $\mathscr{T}(\mathscr{L})$ that is contained in $\approx$. It can be characterised by $\mathscr{T}(P) \approx^c_{\mathscr{T}(\mathscr{L})} \mathscr{T}(Q)$ iff $C[\mathscr{T}(P)] \approx C[\mathscr{T}(Q)]$ for every unary $\mathscr{L}'$-context C of the form $\mathscr{T}(D)$.

In many applications, processes $P \in \mathrm{T}_{\mathscr{L}}$ are only meaningful up to a semantic equivalence $\sim$, and the intended semantic domain could just as well be seen as the set of $\sim$-equivalence classes of processes. For this purpose it is essential that $\sim$ is a congruence for $\mathscr{L}$. Often $\sim$ is the congruence closure of a coarser semantic equivalence $\approx$, so that two processes end up being identified iff they are $\approx$-equivalent in every context. We will see an example of this in Section 4, with $\dot{\approx}$ in the rôle of $\approx$ and $\cong^c$ in the rôle of $\sim = \approx^c$. Now Theorem 3.16 below says that if a translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ is valid up to $\approx$, then it is even valid up to an equivalence $\approx^c_{\mathscr{T}}$ that extends both $\approx^c_{\mathscr{L}}$ and $\approx^c_{\mathscr{T}(\mathscr{L})}$, and melts each equivalence class of $\mathrm{T}_{\mathscr{L}}$ with exactly one of $\mathrm{T}_{\mathscr{T}(\mathscr{L})}$, and vice versa.

Theorem 3.16 Let $\mathscr{T}$ be a translation between simple languages $\mathscr{L}$ and $\mathscr{L}'$ that is valid up to a semantic equivalence $\approx$. Then $\mathscr{T}$ is valid even up to a semantic equivalence $\approx^c_{\mathscr{T}}$, contained in $\approx$, such that

(1) the restriction of $\approx^c_{\mathscr{T}}$ to $\mathrm{T}_{\mathscr{L}}$ is the largest congruence for $\mathscr{L}$ contained in $\approx$,
(2) the restriction of $\approx^c_{\mathscr{T}}$ to $\mathrm{T}_{\mathscr{T}(\mathscr{L})}$ is the largest congruence for $\mathscr{T}(\mathscr{L})$ that is contained in $\approx$, and
(3) each equivalence class of $\approx^c_{\mathscr{T}}$ is the union of an equivalence class of $\approx^c_{\mathscr{L}}$ and one of $\approx^c_{\mathscr{T}(\mathscr{L})}$.

Proof: Define $\approx^c_{\mathscr{T}}$ on $\mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{T}(\mathscr{L})}$ by $R \approx^c_{\mathscr{T}} S$ iff $R \stackrel{\mathscr{T}}{\leftarrow} P \approx^c_{\mathscr{L}} Q \stackrel{\mathscr{T}}{\rightarrow} S$ for some $P, Q \in \mathrm{T}_{\mathscr{L}}$. Trivially, this relation is symmetric, since $\approx^c_{\mathscr{L}}$ is. It is reflexive, since $\approx^c_{\mathscr{L}}$ is reflexive and each $R \in \mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{T}(\mathscr{L})}$ has the form $P \in \mathrm{T}_{\mathscr{L}}$ or $\mathscr{T}(P)$ with $P \in \mathrm{T}_{\mathscr{L}}$. Finally, it is transitive by Lemma 3.15 and the transitivity of $P \approx^c_{\mathscr{L}} Q$. Thus $\approx^c_{\mathscr{T}}$ is an equivalence relation.

That $\approx^c_{\mathscr{T}} \subseteq \approx$ holds is because $P \approx \mathscr{T}(P)$ and $\approx^c_{\mathscr{L}} \subseteq \approx$. That $\mathscr{T}$ is valid up to $\approx^c_{\mathscr{T}}$ holds by definition. Property (1) of Theorem 3.16 holds by construction. Regarding Property (2), by Theorem 3.13 (taking $\sim := \backsim := \approx^c_{\mathscr{T}}$) $\approx^c_{\mathscr{T}}$ is a congruence for $\mathscr{T}(\mathscr{L})$, contained in $\approx$. To show that it is the largest, let $\sim$ be any other congruence for $\mathscr{T}(\mathscr{L})$ contained in $\approx$. Let $\sim_{\mathscr{T}}$ be the extension of $\sim$ to $\mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{T}(\mathscr{L})}$ as defined in the proof of Theorem 3.14: $P \sim_{\mathscr{T}} Q$ iff $P \stackrel{\mathscr{T}}{\rightarrow} R \sim S \stackrel{\mathscr{T}}{\leftarrow} Q$ for some $R, S \in \mathrm{T}_{\mathscr{T}(\mathscr{L})}$. This definition entails that $\sim_{\mathscr{T}} \subseteq \approx$, and by Theorem 3.14 $\sim_{\mathscr{T}}$ is a congruence for $\mathscr{L}$.

Now suppose $R \sim S$ with $R, S \in \mathrm{T}_{\mathscr{T}(\mathscr{L})}$. Let $R = \mathscr{T}(P)$ and $S = \mathscr{T}(Q)$. Then $P \sim_{\mathscr{T}} Q$. Since $\sim_{\mathscr{T}}$ is a congruence for $\mathscr{L}$ contained in $\approx$, and $\approx^c_{\mathscr{L}}$ is the largest such congruence, one has $P \approx^c_{\mathscr{L}} Q$, and thus $R \approx^c_{\mathscr{T}} S$. This shows that $\sim \subseteq \approx^c_{\mathscr{T}}$ and yields Property (2).

Each equivalence class of $\approx^c_{\mathscr{T}}$ contains a process $P \in \mathrm{T}_{\mathscr{L}}$ and its translation $\mathscr{T}(P) \in \mathrm{T}_{\mathscr{T}(\mathscr{L})}$. Using this, Property (3) is a simple consequence of (1) and (2). □

4 Example: translating a synchronous into an asynchronous π-calculus

As an illustration of the concepts introduced above, consider the π-calculus $m\pi$ as presented by Milner in [71], i.e., the one of Sangiorgi and Walker [101] without matching, τ-prefixing, and choice.

Given a set of *names* $\mathcal{N}$, the set $\mathrm{T}_{m\pi} = \mathbb{T}_{m\pi}$ of *process expressions*, *processes* or *terms* P of the calculus is given by

$$P ::= \mathbf{0} \;\mid\; \bar{x}y.P \;\mid\; x(z).P \;\mid\; P|P' \;\mid\; (\nu z)P \;\mid\; !P$$

with x, y, z ranging over $\mathcal{N}$. One usually writes $\bar{x}y$ for $\bar{x}y.\mathbf{0}$ and $x(z)$ for $x(z).\mathbf{0}$. The π-calculus is commonly presented as a closed-term language, in that the semantic value associated with a (closed) term is simply itself. Yet, the real semantics is given by a reduction relation between processes, defined below.

Definition 4.1 An occurrence of a name z in π-calculus process $P \in \mathrm{T}_{m\pi}$ is *bound* if it occurs within a subexpression $x(z).P'$ or $(\nu z)P'$ of P; otherwise it is *free*. Let $\mathsf{n}(P)$ (resp. $\mathsf{bn}(P)$) be the set of names occurring (bound) in $P \in \mathrm{T}_{m\pi}$.

Structural congruence, $\equiv$, is the smallest congruence relation on processes satisfying

$$\begin{array}{rcl@{\qquad}rcl@{\qquad}rcll}
P_1|(P_2|P_3) & \equiv & (P_1|P_2)|P_3 & !P & \equiv & P|!P & (\nu w)(P|Q) & \equiv & P|(\nu w)Q & \text{if } w \notin \mathsf{n}(P) \\
P_1|P_2 & \equiv & P_2|P_1 & (\nu z)\mathbf{0} & \equiv & \mathbf{0} & x(z).P & \equiv & x(w).P\{^w/_z\} & \text{if } w \notin \mathsf{n}(P) \\
P|\mathbf{0} & \equiv & P & (\nu z)(\nu w)P & \equiv & (\nu w)(\nu z)P & (\nu z)P & \equiv & (\nu w)P\{^w/_z\} & \text{if } w \notin \mathsf{n}(P).
\end{array}$$

Here $P\{^w/_z\}$ denotes the process obtained by replacing each free occurrence of z in P by w.[4] [5]

Definition 4.2 The *reduction relation*, $\rightarrow \subseteq \mathrm{T}_{m\pi} \times \mathrm{T}_{m\pi}$, is generated by the following rules.

$$\frac{z \notin \mathsf{bn}(Q)}{\bar{x}z.P|x(y).Q \rightarrow P|Q\{^z/_y\}} \qquad \frac{P \rightarrow P'}{P|Q \rightarrow P'|Q} \qquad \frac{P \rightarrow P'}{(\nu z)P \rightarrow (\nu z)P'} \qquad \frac{Q \equiv P \quad P \rightarrow P' \quad P' \equiv Q'}{Q \rightarrow Q'}$$

Let $\Longrightarrow$ be the reflexive and transitive closure of $\rightarrow$. The observable behaviour of π-calculus processes is often stated in terms of the outputs they can produce, abstracting from the value communicated on an output channel.

Definition 4.3 Let $x \in \mathcal{N}$. A process P has a *strong output barb* on x, notation $P{\downarrow_{\bar{x}}}$, if P can perform an output action $\bar{x}z$. This is defined inductively:

$$(\bar{x}z.(P)){\downarrow_{\bar{x}}} \qquad \frac{P{\downarrow_{\bar{x}}}}{(P|Q){\downarrow_{\bar{x}}}} \qquad \frac{Q{\downarrow_{\bar{x}}}}{(P|Q){\downarrow_{\bar{x}}}} \qquad \frac{P{\downarrow_{\bar{x}}} \quad x \neq z}{((\nu z)P){\downarrow_{\bar{x}}}} \qquad \frac{P{\downarrow_{\bar{x}}}}{(!P){\downarrow_{\bar{x}}}}$$

A process P has a *weak output barb* on x, notation $P{\Downarrow_{\bar{x}}}$, if there is a P' with $P \Longrightarrow P'{\downarrow_{\bar{x}}}$.

A common semantic equivalence applied in the π-calculus is *weak barbed congruence* [74, 100, 101].

Definition 4.4 *Weak (output) barbed bisimilarity* is the largest symmetric relation $\dot{\approx} \subseteq \mathrm{T}_{m\pi} \times \mathrm{T}_{m\pi}$ such that

- $P \dot{\approx} Q$ and $P{\downarrow_{\bar{x}}}$ implies $Q{\Downarrow_{\bar{x}}}$, and
- $P \dot{\approx} Q$ and $P \Longrightarrow P'$ implies $Q \Longrightarrow Q'$ for some Q' with $P' \dot{\approx} Q'$.

Weak barbed congruence, $\cong^c$, is the largest congruence included in $\dot{\approx}$.

Often *input barbs*, defined similarly, are included in the definition of weak barbed bisimilarity [101]. This is known to induce the same notion of weak barbed congruence [101].[6]

[4] Here I apply $P\{^w/_z\}$ only when $w \notin \mathsf{bn}(P)$. When extending the definition of $P\{^w/_z\}$ to arbitrary names w, one additionally needs to rename bound occurrences of w in P so as to avoid capture of free names.

[5] A more common approach is to merely require $w \notin \mathsf{fn}(P) := \mathsf{n}(P) \backslash \mathsf{bn}(P)$ above, and likewise omit the premise in the first rule of Definition 4.2. My definition is simpler, as it doesn't involve implicit renaming of bound names. It yields the same result, since, using the last two rules for $\equiv$, any bound occurrence of w in P or z in Q can always be renamed first.

[6] This remark does not extend to the asynchronous π-calculus, defined below. There the restriction to output barbs is meaningful.

Example 4.5 $\bar{x}z \not\cong^c (\nu u)(\bar{x}u|u(v).\bar{v}z)$. For[7] let $P := \bar{x}z$ and $Q := (\nu u)(\bar{x}u|u(v).\bar{v}z)$, and take the unary context $C[\ ] := X_1|x(u).\bar{u}v$. Then $C[Q] \to (\nu u)\big((u(v).\bar{v}z)|\bar{u}v\big) \to (\bar{v}z)\downarrow_{\bar{v}}$ but $(C[P])\not\Downarrow_{\bar{v}}$.

The asynchronous π-calculus, as introduced by Honda & Tokoro in [61] and by Boudol in [14], is the sublanguage $a\pi$ of the fragment $m\pi$ of the π-calculus presented above where all subexpressions $\bar{x}y.P$ have the form $\bar{x}y.\mathbf{0}$. *Asynchronous barbed congruence*, $\cong^c_a$, is the largest congruence *for the asynchronous π-calculus* included in $\dot{\approx}$. As all expressions used in Example 4.5 belong to $a\pi$, one even has $\bar{x}z \not\cong^c_a (\nu u)(\bar{x}u|u(v).\bar{v}z)$.[8] Since $a\pi$ is a sublanguage of $m\pi$, $\cong^c_a$ is at least as coarse an equivalence as $\cong^c$, i.e. $\cong^c \subseteq \cong^c_a$. The inclusion is strict, since $!x(z).\bar{x}z.\mathbf{0} \cong^c_a \mathbf{0}$, yet $!x(z).\bar{x}z.\mathbf{0} \not\cong^c \mathbf{0}$ [101]. This situation is illustrated in Figure 1, which depicts the sets of closed terms $\mathrm{T}_{m\pi}$ and $\mathrm{T}_{a\pi}$ of the (mini) π-calculus $m\pi$ and its asynchronous fragment $a\pi$, respectively, together with some relevant equivalence relations on these sets. When seeing $\mathrm{T}_{a\pi}$ as a strict subset of $\mathrm{T}_{m\pi}$, the equivalence $\cong^c$ on the right is the exact same as weak barbed congruence $\cong^c$ on the left: the congruence closure w.r.t. $m\pi$ of weak barbed bisimilarity $\dot{\approx}$. Since $\cong^c_a$ is the congruence closure of $\dot{\approx}$ w.r.t. a smaller set of operators, it comes out coarser.

As $a\pi$ is a sublanguage of $m\pi$, trivially $m\pi$ is at least as expressive as $a\pi$. This is testified by the identity function mapping $\mathrm{T}_{a\pi}$ into $\mathrm{T}_{m\pi}$, which surely is a valid translation up to any semantic equivalence. Many specialists on π-calculi agree that $a\pi$ is in fact equally expressive as $m\pi$. This is supported by the following encoding of $m\pi$ into $a\pi$.

Boudol [14] defined a translation $\mathscr{T}$ from $m\pi$ to $a\pi$ inductively as follows:

$$\begin{array}{rcll}
\mathscr{T}(\mathbf{0}) & = & \mathbf{0} & \\
\mathscr{T}(\bar{x}z.P) & = & (\nu u)\big(\bar{x}u|u(v).(\bar{v}z|\mathscr{T}(P))\big) & \text{choosing } u,v \notin \mathsf{n}(P)\cup\{x,z\},\ u\neq v \\
\mathscr{T}(x(y).P) & = & x(u).(\nu v)\big(\bar{u}v|v(y).\mathscr{T}(P)\big) & \text{choosing } u,v \notin \mathsf{n}(P))\cup\{x\},\ u\neq v \\
\mathscr{T}(P|Q) & = & \mathscr{T}(P)|\mathscr{T}(Q) & \\
\mathscr{T}(!P) & = & !\mathscr{T}(P) & \\
\mathscr{T}((\nu x)P) & = & (\nu x)\mathscr{T}(P) &
\end{array}$$

Example 4.5 shows that $\mathscr{T}$ is not valid up to $\cong^c$. In fact, it is not even valid up to $\cong^c_a$. However, as shown in [49], it is valid up to $\dot{\approx}$. (The compositionality of this translation will be further discussed in Section 14.3.) Thus the translation $\mathscr{T}$ induces a bijection between the equivalence classes of $\dot{\approx}$ on $\mathrm{T}_{m\pi}$ and those on the image $\mathrm{T}_{\mathscr{T}(m\pi)} \subseteq \mathrm{T}_{a\pi}$ of $\mathscr{T}$ within $a\pi$. This bijection is indicated in *red* in Figure 1.

By Theorem 3.16 Boudol's translation $\mathscr{T}$ is also valid up to an equivalence $\dot{\approx}^c_{\mathscr{T}}$, defined on the disjoint union of $\mathrm{T}_{m\pi}$ and $\mathrm{T}_{\mathscr{T}(m\pi)} \subseteq \mathrm{T}_{a\pi}$, i.e., on the processes of the source language $m\pi$ and those processes in the target language $a\pi$ that arise as translations of source-language processes. This equivalence is contained in $\dot{\approx}$, and on the source domain $\mathrm{T}_{m\pi}$ coincides with $\cong^c = \dot{\approx}^c_{m\pi}$, the congruence closure of $\dot{\approx}$ on $m\pi$. Moreover, on $\mathrm{T}_{\mathscr{T}(m\pi)}$ it coincides with $\dot{\approx}^c_{\mathscr{T}(m\pi)}$, the congruence closure of $\dot{\approx}$ on $\mathscr{T}(m\pi)$.

As $\cong^c_a$ is a congruence for all of $a\pi$ on all of $\mathrm{T}_{a\pi}$, and contained in $\dot{\approx}$, it is certainly a congruence for $\mathscr{T}(m\pi)$, and thus contained in $\dot{\approx}^c_{\mathscr{T}(m\pi)}$. This inclusion turns out to be strict. As an illustration of that, note that $\bar{x}z|\bar{x}z \cong^c \bar{x}z.\bar{x}z$. (This follows since these processes are strong (early) bisimilar [101] and thus strong full bisimilar by [101, Def. 2.2.2].) Consequently, their translations must be related by $\dot{\approx}^c_{\mathscr{T}(m\pi)}$.

[7] This word "For" at the beginning of a sentence is a coordinating conjunction. "A. For B" can be read as "A, because B". Here B is a sentence, or group of sentences, that explains why the statement A is true. I use it a lot when A and B are large sentences that themselves consist of comma-separated parts. This way, the period "binds weaker" than the various commas within A and B.

[8] In fact, all expressions from Example 4.5 belong to the *localised π-calculus* Lπ [66], a subcalculus of $a\pi$ in which a received name y may not be used in input position $y(z).R$. Consequently, P and Q are even distinguished by the congruence closure of $\dot{\approx}$ w.r.t. Lπ.

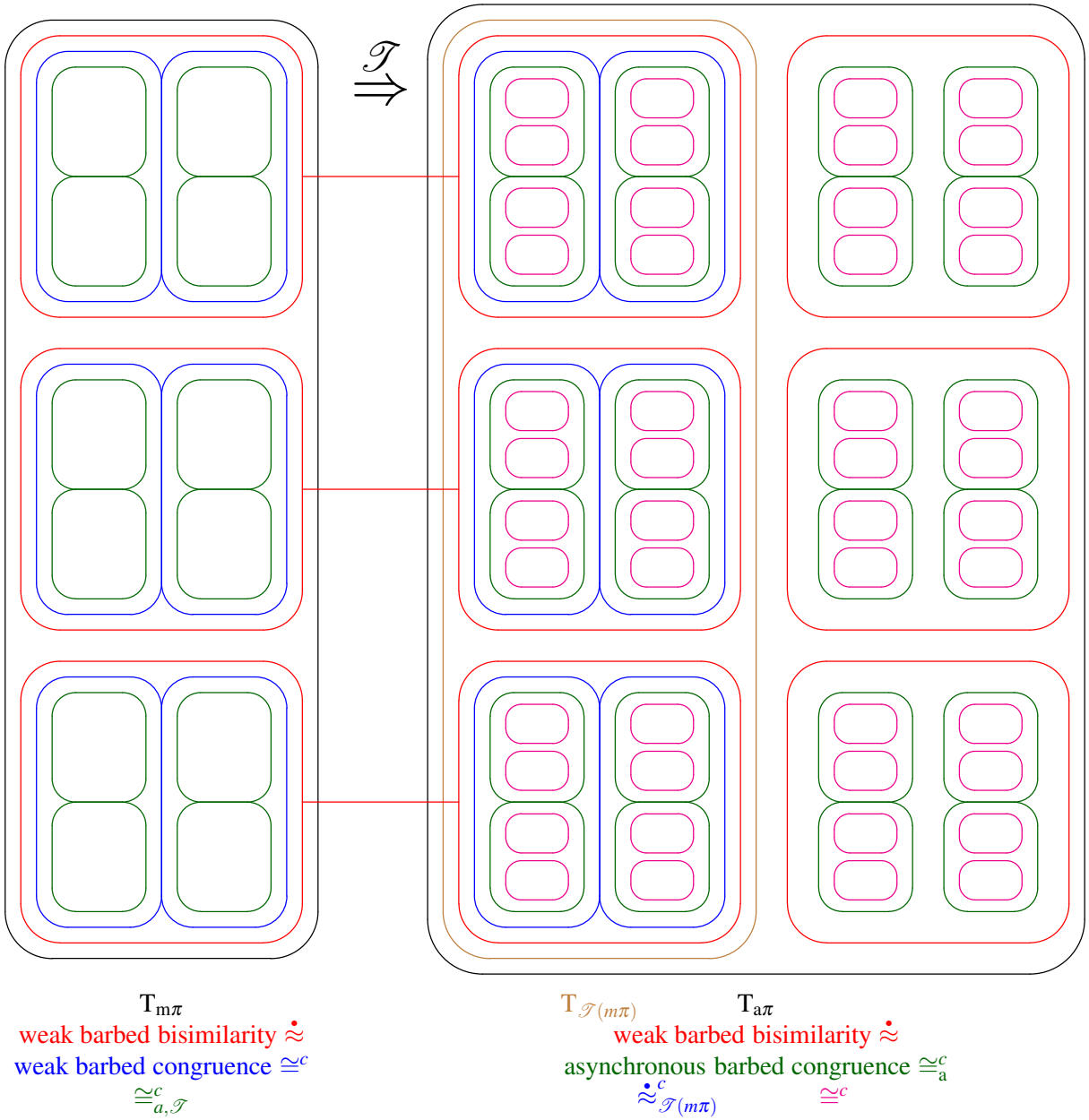


Figure 1: Semantic equivalences for the (a)synchronous π-calculus

So, for distinct $u,v,y,w,x,z \in \mathcal{N}$,

$$(\nu u)\big(\bar{x}u|u(v).(\bar{v}z|\mathbf{0})\big)\big|(\nu u)\big(\bar{x}u|u(v).(\bar{v}z|\mathbf{0})\big) \mathrel{\dot{\approx}^c_{\mathscr{T}(m\pi)}} (\nu y)\big(\bar{x}y|u(w).\big(\bar{w}z|(\nu u)\big(\bar{x}u|u(v).(\bar{v}z|\mathbf{0})\big)\big)\big)\big).$$

Yet, these processes are not $\cong^c_a$-equivalent, as can be seen by putting them in a context $x(y).x(y).\bar{r}(s)|X_1$. In this context, only the left-hand side has a weak barb $\Downarrow_{\bar{r}}$.

5 Integrating language features through translations

Theorem 3.16 shows how valid translations are satisfactory for comparing the expressiveness of simple languages. If there is a valid translation $\mathscr{T}$ from $\mathscr{L}$ to $\mathscr{L}'$ up to $\approx$, then all truths that can be expressed in terms of $\mathscr{L}$ can be mimicked in $\mathscr{L}'$. For the congruence classes of $\approx^c_{\mathscr{L}}$ translate bijectively to congruence classes of an induced equivalence relation on $\mathrm{T}_{\mathscr{T}(\mathscr{L})} \subseteq \mathrm{T}_{\mathscr{L}'}$, and all operations on those congruence classes that can be performed by contexts of $\mathscr{L}$ have a perfect counterpart in terms of contexts of $\mathscr{T}(\mathscr{L})$. This state of affairs was illustrated in the previous section on Boudol's translation from a synchronous to an asynchronous π-calculus.

The expected behaviour of processes in $\mathrm{T}_{\mathscr{T}(\mathscr{L})}$, when being interpreted as translations of processes from $\mathscr{L}$, can be disturbed by composition with processes from $\mathscr{L}'$ outside of $\mathrm{T}_{\mathscr{T}(\mathscr{L})}$. This was illustrated by the context employed at the very end of the previous section.

There is a desirable application of translations between simple languages that is not achieved by Theorem 3.16, namely to combine the powers of two languages into one unified language. If both languages $\mathscr{L}_1$ and $\mathscr{L}_2$ have valid translations into a language $\mathscr{L}'$, then everything that can be done with $\mathscr{L}_1$ can be mimicked in a fragment of $\mathscr{L}'$, and all that can be done with $\mathscr{L}_2$ can be mimicked in another fragment of $\mathscr{L}'$. In order for these two fragments to combine, one would like to employ a single congruence relation on $\mathscr{L}'$ that specialises to congruence relations for $\mathscr{T}_1(\mathscr{L}_1)$ and $\mathscr{T}_2(\mathscr{L}_2)$, which form the counterparts of relevant congruence relations for the source languages $\mathscr{L}_1$ and $\mathscr{L}_2$. This is offered by Theorem 3.14.

In terms of the translation $\mathscr{T}$ from $m\pi$ to $a\pi$, the equivalence $\cong^c_a$ on $\mathrm{T}_{a\pi}$, which is strictly finer than $\mathrel{\dot{\approx}^c_{\mathscr{T}(m\pi)}}$, is the right congruence relation to consider for $a\pi$. By Theorem 3.14 it extends to an equivalence $\cong^c_{a,\mathscr{T}}$ on the disjoint union $\mathrm{T}_{m\pi} \uplus \mathrm{T}_{a\pi}$, such that the restriction to $\mathrm{T}_{m\pi}$ is a congruence for $m\pi$. Necessarily, this congruence on $\mathrm{T}_{m\pi}$ distinguishes the $m\pi$-processes $\bar{x}z|\bar{x}z$ and $\bar{x}z.\bar{x}z$, since their translations are distinguished by $\cong^c_a$. Hence $\cong^c_{a,\mathscr{T}}$ is strictly finer than $\cong^c$. It could be considered to have a non-interleaving flavour. Here it is important that the union of $\mathrm{T}_{m\pi}$ and $\mathrm{T}_{a\pi}$ on which this congruence is defined is required to be disjoint. For if one considers $\mathrm{T}_{a\pi}$ as a subset of $\mathrm{T}_{m\pi}$, then one obtains that the restriction of $\cong^c_{a,\mathscr{T}}$ to that subset (1) coincides with $\cong^c_a$ and (2) is strictly finer than $\cong^c$. This contradicts the fact that $\cong^c$ is strictly finer than $\cong^c_a$.

6 Compositionality for languages with variables, but without binding

In this section I generalise the notion of compositionality from Section 3.1 to languages that do have process variables, yet do not feature any additional syntactic constructs, beyond operators.

For languages with process variables, a natural requirement on translations is that $\mathscr{T}(X) = X$ for each process variable $X \in \mathscr{X}$. It says that a process in the source language whose identity is not yet disclosed, translates into a process in the target language whose identity is not yet disclosed. As I use the same collection of variables for all languages with variables, defining $\mathscr{T}(X) := X$ instead of $\mathscr{T}(X) := Y$

for a variable $Y \neq X$ simplifies the bookkeeping when substituting processes for these variables. I stick this requirement in the definition of compositionality.

Definition 6.1 Let $\mathscr{L}$ and $\mathscr{L}'$ be languages with variables from $\mathscr{X}$, but without additional syntactic constructs (beyond operators). A translation $\mathscr{T}$ from $\mathscr{L}$ into a language $\mathscr{L}'$ is *compositional* if $\mathscr{T}(X) = X$ for each $X \in \mathscr{X}$, and for each n-ary operator f of $\mathscr{L}$ there exists an n-ary $\mathscr{L}'$-context C_f such that $\mathscr{T}(f(E_1,\ldots,E_n)) = C_f[\mathscr{T}(E_1),\ldots,\mathscr{T}(E_n)]$ for any $\mathscr{L}$-expressions $E_1,\ldots,E_n \in \mathbb{T}_{\mathscr{L}}$.

I will present an alternative formulation of Definition 6.1 that does not involve the auxiliary concept of a context. The main idea is that it doesn't matter at all that the holes $X_1,...,X_n$ that appear in contexts are fresh variables, i.e., chosen outside $\mathscr{X}$; one can just as well pick some variables $X_1,...,X_n \in \mathscr{X}$. When doing this, a context is nothing else than an open term (= an expression that may contain variables).

Lemma 6.2 Let an *exterior $\mathscr{L}$-context* be defined as in Section 3.1, i.e., with $\mathscr{X}_n \cap \mathscr{X} = \emptyset$, whereas an *interior $\mathscr{L}$-context* is defined for some arbitrary but fixed choices $\mathscr{X}_n \subseteq \mathscr{X}$. Then the versions of Definition 6.1 that use interior vs. exterior contexts yield the same concept of compositionality.

Proof: In the equation $\mathscr{T}(f(E_1,\ldots,E_n)) = C_f[\mathscr{T}(E_1),\ldots,\mathscr{T}(E_n)]$ no variables occur in either the left- or the right-hand side. All that matters is that the expression $\mathscr{T}(E_i)$ is substituted for the occurrences in C_f of the variable called X_i. Whether $X_i \in \mathscr{X}$ is irrelevant. □

Since the variables that occur in an open term $E \in \mathbb{T}_{\mathscr{T}}$ are not numbered or ordered, there is a priori no notation $E[E_1,...,E_n]$—as one has for contexts—denoting the substitution of expression E_i for the i^{th} hole or variable X_i. Instead I have to use ordinary substitutions.

Definition 6.3 Let $\mathscr{L}$ be a language with variables from $\mathscr{X}$, but without additional syntactic constructs (beyond operators). A *substitution* in $\mathscr{L}$ is a function $\sigma : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$ from the variables to the $\mathscr{L}$-expressions—it is *closed*[9] if $\sigma(X) \in \mathrm{T}_{\mathscr{L}}$ for all X. 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 occurrence of a variable $X \in \mathscr{X}$ is replaced by $\sigma(X)$.

If σ is a closed substitution, then $E[\sigma]$ is a closed expression, for each $E \in \mathbb{T}_{\mathscr{L}}$.

Let $\sigma : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$ be a substitution in $\mathscr{L}$, and $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ a translation of $\mathscr{L}$ into $\mathscr{L}'$. Then their composition $\mathscr{T} \circ \sigma : \mathscr{X} \to \mathbb{T}_{\mathscr{L}'}$ is a substitution in $\mathscr{L}'$. Using this, I can present the alternative formulation of Definition 6.1:

Proposition 6.4 Let $\mathscr{L}$ and $\mathscr{L}'$ be languages with variables from $\mathscr{X}$, but without additional syntactic constructs (beyond operators). A translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ is compositional iff $\mathscr{T}(X) = X$ for each $X \in \mathscr{X}$ and $\mathscr{T}(E[\sigma]) = \mathscr{T}(E)[\mathscr{T} \circ \sigma]$ for each $E \in \mathbb{T}_{\mathscr{L}}$ and $\sigma : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$.

Proof: Let $\mathscr{T}$ be a translation from $\mathscr{L}$ into $\mathscr{L}'$ that satisfies the conditions of Proposition 6.4. I will show that it is compositional. Using Lemma 6.2 I may use interior contexts in Definition 6.1, for some arbitrary but fixed choices $\mathscr{X}_n \subseteq \mathscr{X}$. Let f be an n-ary operator of $\mathscr{L}$, and let $\mathscr{X}_n = \{X_1,...,X_n\}$. Take $E := f(X_1,...,X_n)$ and let $C_f := \mathscr{T}(E)$. I have to show that $\mathscr{T}(f(E_1,\ldots,E_n)) = C_f[\mathscr{T}(E_1),\ldots,\mathscr{T}(E_n)]$ for any $\mathscr{L}$-expressions $E_1,\ldots,E_n \in \mathbb{T}_{\mathscr{L}}$. Let the substitution $\sigma : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$ satisfy $\sigma(X_i) = E_i$ for $i = 1,...,n$. Then $\mathscr{T}(f(E_1,\ldots,E_n)) = \mathscr{T}(E[\sigma]) = \mathscr{T}(E)[\mathscr{T} \circ \sigma] = C_f[\mathscr{T}(E_1),\ldots,\mathscr{T}(E_n)]$.

Now assume that $\mathscr{T}$ is a compositional translation from $\mathscr{L}$ into $\mathscr{L}'$. Let $\sigma : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$ be a substitution. With structural induction on E I show that $\mathscr{T}(E[\sigma]) = \mathscr{T}(E)[\mathscr{T} \circ \sigma]$ for each $E \in \mathbb{T}_{\mathscr{L}}$.

As induction base, let $E = X \in \mathscr{X}$ be a variable, and let $\sigma(X) = F$. Then

$$\mathscr{T}(E[\sigma]) = \mathscr{T}(F) = X[\mathscr{T} \circ \sigma] = \mathscr{T}(E)[\mathscr{T} \circ \sigma].$$

[9] While it may be more appropriate to call a substitution $\sigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ *closing* rather than closed, I here follow widespread terminology.

As induction step, let $E = f(E_1,...,E_n)$. By induction, assume that $\mathscr{T}(E_i[\sigma]) = \mathscr{T}(E_i)[\mathscr{T}\circ\sigma]$ for $i = 1,...,n$. Since $\mathscr{T}$ is compositional, there is an $\mathscr{L}'$-context C_f such that

$$\begin{array}{rcl}\mathscr{T}(E[\sigma]) &=& \mathscr{T}(f(E_1[\sigma],\ldots,E_n[\sigma]))\\ &=& C_f[\mathscr{T}(E_1[\sigma]),\ldots,\mathscr{T}(E_n[\sigma])]\\ &=& C_f[\mathscr{T}(E_1)[\mathscr{T}\circ\sigma],\ldots,\mathscr{T}(E_n)[\mathscr{T}\circ\sigma]]\\ &=& C_f[\mathscr{T}(E_1),\ldots,\mathscr{T}(E_n)][\mathscr{T}\circ\sigma]\\ &=& \mathscr{T}(f(E_1,\ldots,E_n))[\mathscr{T}\circ\sigma]\\ &=& \mathscr{T}(E)[\mathscr{T}\circ\sigma].\end{array}$$ □

The characterisation of compositionality from Proposition 6.4 not only bypasses the concept of a context; it also characterises compositionality in a way that does not refer to the operators of the language. As such it forms a candidate definition for languages that also feature other constructs, besides operators. In fact, it applies perfectly to languages that feature the infinite sum of CCS; here a term may contain infinitely many variables, so that infinitary contexts ought to be used. For this application the characterisation of Proposition 6.4 is preferable to the one of Corollary 3.3. However, in the presence of binding, as in $\mu X.E$, the notion of compositionality from Proposition 6.4 turns out to be too restrictive—see Section 10.5.

7 Validity for simple languages with variables

In this section I generalise the definition of validity from Section 3.2 to languages $\mathscr{L} = (\mathbb{T}_{\mathscr{L}}, [\![\ ]\!]_{\mathscr{L}})$ that satisfy restrictions (ii) and (iii) of Section 3, but do feature process variables. I refer to such languages as *simple languages with variables*. Until Section 10, I still restrict attention to translations $\mathscr{T}$ such that $\mathscr{T}(P)$ is a closed expression for each closed expression P.

The simplest way to generalise Definition 3.4 to simple languages with variables is to stick to the formulation of Definition 3.4, that is, to require compositionality, as well as $\mathscr{T}(P) \leftrightsquigarrow P$ for each closed expression $P \in \mathrm{T}_{\mathscr{T}}$. In this section, I will call this *A-validity*. A possible alternative is to require something like $\mathscr{T}(E) \leftrightsquigarrow E$ even for open expressions $E \in \mathbb{T}_{\mathscr{T}}$. However, it is a priori not clear how to define $\leftrightsquigarrow$ on open expressions from different languages.

Let $\mathscr{L}$ be a simple language with variables. Any semantic equivalence or preorder defined on $\mathrm{T}_{\mathscr{L}}$ naturally extends to $\mathbb{T}_{\mathscr{L}}$ by

$$F \leftrightsquigarrow E \qquad \text{iff} \qquad F[\varsigma] \leftrightsquigarrow E[\varsigma] \text{ for each closed substitution } \varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}. \tag{1}$$

A problem with defining $F \leftrightsquigarrow E$ when E and F are open expressions from different languages $\mathscr{L}$ and $\mathscr{L}'$, is that a substitution $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ in $\mathscr{L}$ generally is not also a substitution in $\mathscr{L}'$. Below I will contemplate three possible solutions to this problem, giving rise to three notions of validity up to an equivalence or preorder $\leftrightsquigarrow$. In this section, I will address those notions as B-, C- and D-validity. I will also consider A^--, B^--, C^-- and D^--validity, obtained by dropping the requirement of compositionality. These notions can be ordered by the strength of their requirements as follows.

$$\begin{array}{ccccccc}\mathrm{A} & \preceq & \mathrm{B} & \preceq & \mathrm{C} & \preceq & \mathrm{D}\\ \curlyvee| & & \curlyvee| & & \curlyvee| & & \curlyvee|\\ \mathrm{A}^- & \preceq & \mathrm{B}^- & \preceq & \mathrm{C}^- & \preceq & \mathrm{D}^-\end{array}$$

This means that when a translation is D-valid, it is certainly A^--valid, etc. These 8 notions of a valid translation, and their associated expressiveness preorders, although defined for simple languages with

variables, can also be applied to simple languages without variables, namely by extending both the source and target language with variables before evaluating their relative expressiveness.

In the end, I pick one of those notions as the concept of validity that I propose in this paper. My choice is guided by the following considerations.

The requirement $\mathscr{T}(E) \mathrel{\wr\!\sim} E$ is natural, and semantic. It says that the meaning of a translated object should be equivalent to the meaning of the original. This requirement is just as much applicable to translations between process calculi as to translations between French and English.

Compositionality is a syntactic or structural requirement. Whereas it can surely be deemed a good property of translations, in my view it does not directly say something about their correctness. For that purpose I would intuitively hold the semantic requirement $\mathscr{T}(E) \mathrel{\wr\!\sim} E$ to be the only relevant one. Hence I am predisposed towards notions A^-, B^-, C^- and D^- over A, B, C and D.

Nevertheless, as discussed in Section 8, simply dropping compositionality from Definition 3.4 yields a unacceptably weak criterion (A^--validity), validating many intuitively unsatisfactory translations. This issue is mitigated by extending the requirement $\mathscr{T}(E) \mathrel{\wr\!\sim} E$ to open expressions E. This fits with intuition: one wants to preserve the meaning of any expression under translation, not just the closed ones. Hence I am predisposed towards one of the notions B, C or D over A.

Below I will reject B^--validity based on an example of an intuitively unsatisfactory translation that is deemed B^--valid. A^--validity is a useful concept, but, as discussed in Section 8, it is too weak to capture the notion of expressiveness I try to formalise in this paper. D^--validity and D-validity are also useful concepts, and they describe the same notion of expressiveness, but they are too strong to capture the form of expressiveness I try to formalise in this paper. In particular, they reject the validity up to $\dot{\approx}$ of the encoding from Section 4 of $m\pi$ into $a\pi$.

The notions of A-, B- and C-validity turn out to coincide, and although C^--validity is slightly weaker, I show that it describes the same notion of expressiveness. Based on this, I settle on C^--validity as the notion of validity proposed in this paper.

7.1 D-validity

To generalise (1) to the case that $E \in \mathbb{T}_{\mathscr{L}}$ and $F \in \mathbb{T}_{\mathscr{L}'}$, when $\mathrel{\wr\!\sim}$ is a binary relation on $\mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{L}'}$, I relate two closed substitutions $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ and $\eta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ by

$$\eta \mathrel{\wr\!\sim} \varsigma \quad \Leftrightarrow \quad \varsigma(X) \mathrel{\wr\!\sim} \eta(X) \text{ for each } X \in \mathscr{X}.$$

Using this, I define

$$F \mathrel{\wr\!\sim} E \qquad \text{iff} \qquad F[\eta] \mathrel{\wr\!\sim} E[\varsigma] \text{ for each } \varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}} \text{ and } \eta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'} \text{ with } \eta \mathrel{\wr\!\sim} \varsigma$$

and D-validity requires $\mathscr{T}(E) \mathrel{\wr\!\sim} E$ in the above sense, for each $E \in \mathbb{T}_{\mathscr{L}}$. Additionally, I require that there exists at least one pair $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ and $\eta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ with $\eta \mathrel{\wr\!\sim} \varsigma$, for otherwise any translation would be gratuitously valid.

D^--validity was proposed in [42] under the name *correctness*. [42, Theorem 3] shows that if any correct translation from $\mathscr{L}$ to $\mathscr{L}'$ up to an equivalence $\sim$ exists, then there even exists a compositional translation that is correct up to $\sim$. In view of Definition 2.2, in which only the existence of a valid translation between two languages matters, the notions of D^--validity and D-validity thus give rise to the same family of expressiveness preorders between languages.

Proposition 1 in [42] says that if a correct translation up to $\sim$ from $\mathscr{L}$ to $\mathscr{L}'$ exists, then $\sim$ is a congruence for $\mathscr{L}$. Since weak barbed bisimilarity, $\dot{\approx}$, is not a congruence for $m\pi$, it follows that there

can exist no translation from $m\pi$ to $a\pi$ that is D^--valid up to $\dot{\approx}$. Thus D- and D^--validity are too demanding concepts to justify the validity up to $\dot{\approx}$ of Boudol's translation. As such they are not the right generalisations to languages with variables of the notion of a valid translation from Section 3.2.

7.2 B-validity

This approach refrains from giving a general definition of $F \leftrightsquigarrow E$ when $E \in \mathbb{T}_{\mathscr{L}}$ and $F \in \mathbb{T}_{\mathscr{L}'}$, but it does so for the special case that $F = \mathscr{T}(E)$, with $\mathscr{T}$ a translation from $\mathscr{L}$ into $\mathscr{L}'$ that satisfies $\mathscr{T}(P) \leftrightsquigarrow P$ for each closed expression $P \in \mathrm{T}_{\mathscr{T}}$. Namely

$$\mathscr{T}(E) \leftrightsquigarrow E \qquad \text{iff} \qquad \mathscr{T}(E)[\mathscr{T} \circ \varsigma] \leftrightsquigarrow E[\varsigma] \text{ for each closed substitution } \varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}.$$

This equation uses on the right-hand side a closed substitution in $\mathscr{L}$, and on the left-hand side its translation as a closed substitution in $\mathscr{L}'$. Using this yields the following alternative definition of validity.

Definition 7.1 A translation $\mathscr{T}$ between simple languages with variables is *B-valid* up to $\leftrightsquigarrow$ if it is compositional and $\mathscr{T}(E)[\mathscr{T} \circ \varsigma] \leftrightsquigarrow E[\varsigma]$ for each $E \in \mathbb{T}_{\mathscr{L}}$ and each closed substitution $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ in $\mathscr{L}$.

Trivially, B-validity implies A-validity, as the latter can be obtained by applying Definition 7.1 only to closed expressions $E \in \mathrm{T}_{\mathscr{L}}$. In fact, B-validity is the same as A-validity. Namely, by applying compositionality of $\mathscr{T}$ and Proposition 6.4, the requirement $\mathscr{T}(E)[\mathscr{T} \circ \varsigma] \leftrightsquigarrow E[\varsigma]$ of Definition 7.1 can be rewritten as $\mathscr{T}(E[\varsigma]) \leftrightsquigarrow E[\varsigma]$, and as $E[\varsigma]$ is a closed expression, this is already implied by A-validity.

Also note that B^--validity is implied by D^--validity. Namely, a D^--valid translation $\mathscr{T}$ satisfies $\mathscr{T}(P) \leftrightsquigarrow P$ for each closed expression $P \in \mathrm{T}_{\mathscr{T}}$. Consequently, $\mathscr{T} \circ \varsigma \leftrightsquigarrow \varsigma$ for each $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ in $\mathscr{L}$.

Example 7.2 Let the language $\mathscr{L}$ feature a constant ε and a unary operator f. Its expressions model the states of the 1-person game "build a high stack of cards". The constant ε models the initial state of the game, in which the stack of cards built so far is empty. The operator f models the only legal move in this game, namely adding a card to the stack. Clearly, all terms in this language have the form $f^n(\varepsilon)$ or $f^n(X)$, where $X \in \mathscr{X}$. Here $f^n(E)$ denotes the result of applying the operator f n times to the expression E. The expression $f^n(X)$ models the extension of an unspecified game X by playing n more cards. The game awards 1 point for each card in the stack. The equivalence relation $\sim$ on $\mathrm{T}_{\mathscr{L}}$ declares two closed terms equivalent iff they model states of the game which earn the same number of points. This turns out to be the identity relation on $\mathrm{T}_{\mathscr{L}}$.

Let the language $\mathscr{L}'$ model a more challenging version of this game, with black and red cards. It features the constant ε denoting the empty stack, and unary operators b and r of adding a black or a red card to the stack. Again one can earn 1 point for each card added, but only as long as the cards in the stack alternate between red and black. Whenever the current stack is nonempty, and one accidentally adds another card with the same colour as the topmost card in the stack, all cards already played are taken away, and the points earned are reset to 0. The equivalence relation $\sim$ extends to $\mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{L}'}$ by declaring two closed terms equivalent iff they model states which earn the same number of points. For example, $bbb(\varepsilon) \sim b(\varepsilon)$ and $fff(\varepsilon) \sim brb(\varepsilon)$. Note that one doesn't have $bb(X) \sim \varepsilon$.

We now wonder whether language $\mathscr{L}'$ is at least as expressive and as $\mathscr{L}$, and to this end consider the following translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$.

$$\begin{array}{rclcrcl} \mathscr{T}(f^{2n}(\varepsilon)) & := & (rb)^n(\varepsilon) & \qquad & \mathscr{T}(f^{2n}(X)) & := & (rb)^n(X) \\ \mathscr{T}(f^{2n+1}(\varepsilon)) & := & (rb)^n r(\varepsilon) & & \mathscr{T}(f^{2n+1}(X)) & := & b(rb)^n(X) \end{array}$$

To show that this translation is B$^-$-valid, I have to establish that

$$\begin{array}{llll} (rb)^n(\varepsilon) & \sim & f^{2n}(\varepsilon) & \qquad (rb)^n(\mathscr{T}(f^k(\varepsilon))) \sim f^{2n}(f^k(\varepsilon)) \\ (rb)^n r(\varepsilon) & \sim & f^{2n+1}(\varepsilon) & \qquad b(rb)^n(\mathscr{T}(f^k(\varepsilon))) \sim f^{2n+1}(f^k(\varepsilon)) \end{array}$$

for each $n,k \in \mathbb{N}$. As a result of the careful design of $\mathscr{T}$, this is indeed the case. Namely, if $k > 0$, the topmost card in the translation of the closed expression $f^k(\varepsilon)$ is red, whereas if $m > 0$, the bottom card in the translation of the game extension $f^m(X)$ is black. Consequently there are never two adjacent cards with the same colour in $\mathscr{T}(f^m(X))[\varsigma]$, where the closed substitution ς sends $f^k(\varepsilon)$ to $\mathscr{T}(f^k(\varepsilon))$.

Yet $\mathscr{T}$ fails to be compositional. Compositionality requires that $\mathscr{T}(f(E)) = C[\mathscr{T}(E)]$, for a unary context C that is independent of $E \in \mathbb{T}_{\mathscr{L}}$. This requirement is not met here, as $\mathscr{T}(f(\varepsilon)) = r(\varepsilon)$, yet $\mathscr{T}(f(X)) = b(X)$. I conjecture that no compositional A-valid translation from $\mathscr{L}$ into $\mathscr{L}'$ exists.

Intuitively, I find $\mathscr{T}$ an unsatisfactory translation. Namely, since $\mathscr{T}(f(\varepsilon)) = r(\varepsilon)$, one would have to admit that $r(\varepsilon)$ is a good translation of $f(\varepsilon)$, and therefore the result of substituting $r(\varepsilon)$ for X in $\mathscr{T}(f(X))$—which is $br(\varepsilon)$—should be a good translation of $ff(\varepsilon)$. Based on this, using the same reasoning once more, the result of substituting $br(\varepsilon)$ for X in $\mathscr{T}(f(X))$—which is $bbr(\varepsilon)$—should be a good translation of $fff(\varepsilon)$. Yet this is not the case.

7.3 C-validity

Based on Example 7.2, which classifies an intuitively unsatisfactory translation as B$^-$-valid, I reject B$^-$-validity as my proposed notion. Instead I propose a new concept that lays strictly in between B$^-$-validity and D$^-$-validity. Its definition is inspired by the reasoning in Example 7.2.

Definition 7.3 A *semantic translation* from $\mathscr{L}$ into $\mathscr{L}'$ is a relation $\mathbf{R} \subseteq \mathrm{T}_{\mathscr{L}'} \times \mathrm{T}_{\mathscr{L}}$ such that $\forall P \in \mathrm{T}_{\mathscr{L}}.\, \exists P' \in \mathrm{T}_{\mathscr{L}'}.\, P' \mathbf{R} P$.

Intuitively, $P' \mathbf{R} P$ says that the closed $\mathscr{L}'$-expression P' is a good translation or *counterpart* of the closed $\mathscr{L}$-expression P. Every closed term of $\mathscr{L}$ needs to have a counterpart in $\mathscr{L}'$—possibly multiple ones. For closed substitutions $\eta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ and $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$, I write $\eta \mathbf{R} \varsigma$ iff $\eta(X) \mathbf{R} \varsigma(X)$ for each $X \in \mathscr{X}$.

Definition 7.4 A translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ is *correct* w.r.t. a semantic translation $\mathbf{R}$ if $\mathscr{T}(E)[\eta] \mathbf{R} E[\varsigma]$ for all expressions $E \in \mathbb{T}_{\mathscr{L}}$ and all closed substitutions $\eta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ and $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ with $\eta \mathbf{R} \varsigma$.

Thus $\mathscr{T}$ is correct iff the meaning of the translation of an expression E is a counterpart of the meaning of E, no matter what values are filled in for the variables, provided that the value filled in for a given variable X occurring in the translation $\mathscr{T}(E)$ is a counterpart of the value filled in for X in E.

Definition 7.5 A translation $\mathscr{T}$ is *C$^-$-valid* up to $\rightsquigarrow$ iff it is correct w.r.t. some semantic translation $\mathbf{R} \subseteq \rightsquigarrow$.

Clearly, D$^-$-validity implies C$^-$-validity: it can be obtained by strengthening the requirement on $\mathbf{R}$ of Definition 7.5 to $\mathbf{R} = \rightsquigarrow \restriction (\mathrm{T}_{\mathscr{L}'} \times \mathrm{T}_{\mathscr{L}})$. Moreover, if $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$ then $\mathscr{T}(P) \mathbf{R} P$ for each closed expression $P \in \mathrm{T}_{\mathscr{L}}$, and thus $(\mathscr{T} \circ \varsigma) \mathbf{R} \varsigma$ for each closed substitution $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$. This entails that C$^-$-validity implies B$^-$-validity.

Proposition 7.6 A-, B- and C-validity coincide.

Proof: We already know that C-validity implies B-validity, and B-validity implies A-validity. Let $\mathscr{T}$ be an A-valid translation from $\mathscr{L}$ into $\mathscr{L}'$. Define $\mathbf{R} \subseteq \mathrm{T}_{\mathscr{L}'} \times \mathrm{T}_{\mathscr{L}}$ by $P' \mathbf{R} P$ iff $P' = \mathscr{T}(P)$. Then $\mathbf{R}$ is a semantic translation and $\mathbf{R} \subseteq \rightsquigarrow$. Let $E \in \mathbb{T}_{\mathscr{L}}$, and let $\eta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ and $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ be closed substitutions with $\eta \mathbf{R} \varsigma$. Then $\eta = \mathscr{T} \circ \varsigma$. Since $\mathscr{T}$ is compositional, Proposition 6.4 yields $\mathscr{T}(E[\varsigma]) = \mathscr{T}(E)[\eta]$, that is, $\mathscr{T}(E)[\eta] \mathbf{R} E[\varsigma]$. Hence $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$ and thus C-valid up to $\rightsquigarrow$. □

The above proof also shows that the concept of C-validity is unaffected if in Definition 7.5 one would require $\mathbf{R}$ to be a(n inverse) function. For C$^-$-validity, on the other hand, allowing general $\mathbf{R}$ increases the class of valid translations.

Example 7.7 Take the variant of Example 7.2 with the same syntax, but without the penalty in $\mathscr{L}'$ on playing the same card twice in a row, so that now $bbb(\varepsilon) \sim fff(\varepsilon) \sim brb(\varepsilon)$. Then the translation $\mathscr{T}$ defined in Example 7.2 is C$^-$-valid. This can be seen by taking $\mathbf{R} := \sim \restriction (\mathrm{T}_{\mathscr{L}'} \times \mathrm{T}_{\mathscr{L}})$. It even is D$^-$-valid. Yet this translation is not correct w.r.t. any semantic translation $\mathbf{R}$ that is a function.

Although a C$^-$-valid translation is not necessarily compositional, I show that it can always be converted into a C-valid translation, i.e., a C$^-$-valid translation that is compositional. Hence the concept of one language being at least as expressive as another is the same, regardless whether it is based on A-, B-, C- or C$^-$-validity.

Theorem 7.8 If a translation between simple languages with variables exists that is C$^-$-valid up to a preorder $\lesssim$, then there exists a compositional translation that is C-valid up to $\lesssim$.

Proof: Enumerate a countable subset of $\mathscr{X}$ as X_1, X_2, etc., so that each term E with $fv(E) \subseteq \{X_1, \ldots, X_n\}$ can be seen as an n-ary context. Given a translation $\mathscr{T}_0$ that is C$^-$-valid up to $\lesssim$, and thus correct w.r.t. a semantic translation $\mathbf{R} \subseteq \lesssim$, define the translation $\mathscr{T}$ inductively by

$$\begin{array}{ll} \mathscr{T}(X) := X & \text{for } X \in \mathscr{X} \\ \mathscr{T}(f(E_1,...,E_n)) := \mathscr{T}_0(f(X_1,...,X_n))[\mathscr{T}(E_1),...,\mathscr{T}(E_n)]. & \end{array}$$

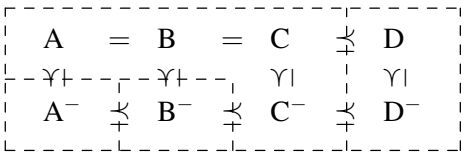
This translation is compositional by construction. It remains to show that $\mathscr{T}$ is A-valid, and thus C-valid, i.e., that $\mathscr{T}(P) \lesssim P$ for each closed expression $P \in \mathrm{T}_{\mathscr{L}}$. I fact, I show that $\mathscr{T}(P)\ \mathbf{R}\ P$, and do this with structural induction on P. I assume that there exists a closed expression $P_0 \in \mathrm{T}_{\mathscr{L}}$, since otherwise the statement holds trivially.

So let $P = f(P_1,...,P_n)$ and assume that $\mathscr{T}(P_i)\ \mathbf{R}\ P_i$ for $i = 1,...,n$. Let $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ and $\eta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ be closed substitutions satisfying $\varsigma(X_i) = P_i$ and $\eta(X_i) = \mathscr{T}(P_i)$ for $i = 1,...,n$; let $\varsigma(X) = P_0$ and $\eta(X) = \mathscr{T}_0(P_0)$ for each variable $X \notin \{X_1,...,X_n\}$. Then $\eta\ \mathbf{R}\ \varsigma$. Therefore

$$\begin{array}{lcll} \mathscr{T}(f(P_1,...,P_n)) & = & \mathscr{T}_0(f(X_1,...,X_n))[\mathscr{T}(P_1),...,\mathscr{T}(P_n)] & \text{(by definition of } \mathscr{T}) \\ & = & \mathscr{T}_0(f(X_1,...,X_n))[\eta] & \text{(by definition of } \eta) \\ & \mathbf{R} & f(X_1,...,X_n)[\varsigma] & \text{(by Definition 7.4 for } \mathscr{T}_0) \\ & = & f(P_1,...,P_n) & \text{(by definition of } \varsigma). \end{array}$$ □

7.4 Conclusion

In this section I contemplated eight definitions of a valid translation for simple languages with variables, giving rise to six different concepts of a valid translation and four different notions of expressiveness, as indicated on the right.



I pick C$^-$-validity as my preferred notion, and henceforth call is *validity*. It induces the same notion of expressiveness as A-, B- and C-validity, which generalise the notion defined in Section 3.2 for simple languages without variables. The appeal of C$^-$-validity is that it only involves a single semantic requirement, with matches intuition and could be applied to translations between French and English just as well as to translations between process calculi. The notions of A-, B- and C-validity on the other hand also involve the syntactic requirement of compositionality.

D- and D$^-$-validity are stronger requirements, which do not recognise the validity up to $\dot{\approx}$ of Boudol's translation of the synchronous mini-π calculus into its asynchronous fragment. A$^-$-validity is a weaker requirement that will be discussed in Section 8, and B$^-$-validity is an intermediate notion that I felt to be unsatisfactory in the light of Example 7.2.

8 Absolute versus relative expressiveness

In the literature, when comparing the expressiveness of system description languages, it is common to distinguish *absolute* and *relative* expressiveness [84]. The present paper is about relative expressiveness, broadly defined in Definition 2.2. Absolute expressiveness compares the class of systems or processes that can expressed in a language. A typical way to refute that a language $\mathscr{L}$ is as most as expressive as a language $\mathscr{L}'$ in the absolute sense is to exhibit a concrete system or process, such as *mutual exclusion* or *leader election*, that can be satisfactorily represented in $\mathscr{L}$, but not in $\mathscr{L}'$.

My preferred formalisation of absolute expressiveness involves a semantic equivalence (or preorder) $\mathcal{R}$, similar to my formalisation of relative expressiveness (in Definitions 3.4 and 7.5). Language $\mathscr{L}'$ matches the absolute expressiveness of $\mathscr{L}$ iff each process that can be denoted by a closed term of $\mathscr{L}$ can, up to $\mathcal{R}$, also be denoted in $\mathscr{L}'$. The below definition makes this precise for closed-term languages.

Definition 8.1 Language $\mathscr{L}'$ has at least the same absolute expressiveness as language $\mathscr{L}$ up to $\mathcal{R}$ iff $\forall P \in \mathrm{T}_{\mathscr{L}}.\ \exists Q \in \mathrm{T}_{\mathscr{L}'}.\ Q \mathcal{R} P$.

The next observation says that this is the case iff there exists a translation from $\mathscr{L}$ to $\mathscr{L}'$ that is A$^-$-valid as defined in Section 7. A$^-$-validity was obtained from A-validity (Definition 3.4) by dropping the requirement of compositionality.

Observation 8.2 Language $\mathscr{L}'$ has at least the same absolute expressiveness as language $\mathscr{L}$ up to $\mathcal{R}$ iff there exists a translation $\mathscr{T} : \mathrm{T}_{\mathscr{L}} \to \mathrm{T}_{\mathscr{L}'}$ from $\mathscr{L}$ into $\mathscr{L}'$ such that $\mathscr{T}(P) \mathcal{R} P$ for all $P \in \mathrm{T}_{\mathscr{L}}$.

Example 8.3 Let $\mathscr{L}$ be the recursion-free fragment of the language CCS, as defined in Section 14.6, and let $\mathscr{L}'$ be the fragment of $\mathscr{L}$ that lacks the parallel composition operator. Strong bisimilarity between CCS processes is defined in Section 14.4. Due to the CCS expansion theorem [70], each $\mathscr{L}$-expression can be rewritten into an $\mathscr{L}'$ expression—by eliminating the parallel composition operator—that denotes a strongly bisimilar state in a labelled transition system. Hence $\mathscr{L}$ and $\mathscr{L}'$ have the very same absolute expressiveness. Yet the relative expressiveness of $\mathscr{L}$ is larger than that of $\mathscr{L}'$, because the parallel composition operator cannot be mimicked by an $\mathscr{L}'$-context without parallel composition.

This example shows the key difference between relative and absolute expressiveness: when dealing with relative expressiveness one wants not only the processes that can be denoted in $\mathscr{L}$ to be denotable in $\mathscr{L}'$ as well, but also the operators on processes. Observation 8.2 shows that A$^-$-validity measures absolute expressiveness, whereas A-validity, or C$^-$-validity, measures relative expressiveness.

9 Congruence transfer, reflection and closure properties with variables

I show that all results on validity from Section 3 generalise smoothly to simple languages with variables, now using the notion of validity from Definition 7.5. Moreover, I reformulate these results so as to avoid reference to the operators of a language, as well as to the concept of a context.

Definition 3.4 defined the validity of a translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ between languages $\mathscr{L}$ and $\mathscr{L}'$ up to a preorder $\mathcal{R}$ that by default is defined at least on $\mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{L}'}$. At the end of Section 3.2, however,

it was remarked that in fact it suffices to assume that $\rightsquigarrow$ is defined on $\mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{T}(\mathscr{L})}$. When working with Definition 7.5 one may encounter relations $\mathbf{R} \subseteq \mathrm{T}_{\mathscr{L}'} \times \mathrm{T}_{\mathscr{L}}$ such that $dom(\mathbf{R}) \not\subseteq \mathrm{T}_{\mathscr{T}(\mathscr{L})}$. To deal with such a case, I simply allow $\rightsquigarrow$ to be defined on any *subset* of $\mathrm{T}_{\mathscr{L}} \uplus \mathrm{T}_{\mathscr{L}'}$. The requirement $\mathbf{R} \subseteq \rightsquigarrow$ of Definition 7.5 implies this subset to contain $\mathrm{T}_{\mathscr{L}} \uplus \mathbf{R}(\mathscr{L})$, where $\mathbf{R}(\mathscr{L}) := \{Q \in \mathrm{T}_{\mathscr{L}'} \mid \exists P \in \mathrm{T}_{\mathscr{L}}.\ Q\,\mathbf{R}\,P\}$. Since $\mathscr{T}(P)\,\mathbf{R}\,P$ for each closed expression $P \in \mathrm{T}_{\mathscr{L}}$, one moreover has $\mathrm{T}_{\mathscr{T}(\mathscr{L})} \subseteq \mathbf{R}(\mathscr{L})$.

9.1 A hierarchy of expressiveness preorders on simple languages with variables

Observation 9.1 Let $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be a translation from $\mathscr{L}$ into $\mathscr{L}'$, with $\mathscr{L}, \mathscr{L}'$ simple languages with variables, and let $\rightsquigarrow$ be finer than $\approx$. If $\mathscr{T}$ is valid up to $\rightsquigarrow$, then it is also valid up to $\approx$.

Observation 9.2 The identity is a valid translation up to any preorder from any language into itself.

Theorem 9.3 If valid translations up to $\rightsquigarrow$ exists from $\mathscr{L}$ into $\mathscr{L}'$ and from $\mathscr{L}'$ into $\mathscr{L}'$, then there is a valid translation up to $\rightsquigarrow$ from $\mathscr{L}$ into $\mathscr{L}''$.

Proof: Let $\mathscr{T}_1 : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be a valid translation up to $\rightsquigarrow$ from $\mathscr{L}$ to $\mathscr{L}'$, and let $\mathscr{T}_2 : \mathbb{T}_{\mathscr{L}'} \to \mathbb{T}_{\mathscr{L}''}$ be one from $\mathscr{L}'$ to $\mathscr{L}''$. I will show that the translation $\mathscr{T} := \mathscr{T}_2 \circ \mathscr{T}_1 : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}''}$ from $\mathscr{L}$ into $\mathscr{L}''$, given by $\mathscr{T}(E) = \mathscr{T}_2 \circ \mathscr{T}_1(E) = \mathscr{T}_2(\mathscr{T}_1(E))$, is valid up to $\rightsquigarrow$.

For $i = 1, 2$ let $\mathscr{T}_i$ be correct w.r.t. the semantic translation $\mathbf{R}_i \subseteq \rightsquigarrow$. Let $\mathbf{R} := \mathbf{R}_2 \circ \mathbf{R}_1 \subseteq \mathrm{T}_{\mathscr{L}''} \times \mathrm{T}_{\mathscr{L}}$ be the relation given by $P''\,\mathbf{R}\,P$ iff $\exists P' \in \mathrm{T}_{\mathscr{L}'}.\ P''\,\mathbf{R}_2\,P' \wedge P'\,\mathbf{R}_1\,P$—it is again a semantic translation, and satisfies $\mathbf{R}_2 \circ \mathbf{R}_1 \subseteq \rightsquigarrow$, using the transitivity of $\rightsquigarrow$. Now let $E \in \mathbb{T}_{\mathscr{L}}$, $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ and $\theta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}''}$, with $\theta\,\mathbf{R}\,\varsigma$. Then there is a substitution $\eta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ with $\theta\,\mathbf{R}_2\,\eta\,\mathbf{R}_1\,\varsigma$. Hence, applying Definition 7.4, $\mathscr{T}_2(\mathscr{T}_1(E))[\theta]\,\mathbf{R}_2\,\mathscr{T}_1(E)[\eta]\,\mathbf{R}_1\,E[\varsigma]$, so $\mathscr{T}(E)[\theta]\,\mathbf{R}\,E[\varsigma]$ and $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$. □

9.2 The concept of a congruence for simple languages with variables

Definition 3.8 of a congruence for $\mathscr{L}$, as well as Propositions 3.9 and 3.10, apply verbatim to simple languages with variables. However, similar to the treatment of compositionality in Section 6, I reformulate this definition so as to avoid reference to the operators of $\mathscr{L}$, as well as to the concept of a context.

Proposition 9.4 Let $\mathscr{L}$ be a simple language with variables. An equivalence relation $\sim$ on $\mathrm{T}_{\mathscr{L}}$ is a congruence for $\mathscr{L}$ iff $E[\varsigma_1] \sim E[\varsigma_2]$ for any $E \in \mathbb{T}_{\mathscr{L}}$ and any $\varsigma_1, \varsigma_2 : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ with $\varsigma_1 \sim \varsigma_2$.

Proof: Suppose $\sim$ satisfies the condition of Proposition 9.4. I will show that it is congruence. Let f be an n-ary operator of $\mathscr{L}$ and suppose $P_i, Q_i \in \mathrm{T}_{\mathscr{L}}$ satisfy $P_i \sim Q_i$ for $i = 1, ..., n$. Take $E := f(X_1, ..., X_n)$ for certain variables $X_1, ..., X_n \in \mathscr{X}$ and let the substitutions $\varsigma_2, \varsigma_1 : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ satisfy $\varsigma_2(X_i) = P_i$ and $\varsigma_1(X_i) = Q_i$ for $i = 1, ..., n$, so that $\varsigma_2 \sim \varsigma_1$. Then $f(P_1, ..., P_n) = E[\varsigma_2] \sim E[\varsigma_1] = f(Q_1, ..., Q_n)$.

Now suppose that $\sim$ is a congruence. Let $\varsigma_1, \varsigma_2 : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ be substitutions with $\varsigma_1 \sim \varsigma_2$. With structural induction on $E \in \mathbb{T}_{\mathscr{L}}$ I show that $E[\varsigma_1] \sim E[\varsigma_2]$.

As induction base, let $E = X$ be a variable. Let $\varsigma_1(X) = P$ and $\varsigma_2(X) = Q$. Then $E[\varsigma_1] = P \sim Q = E[\varsigma_2]$.

As induction step, let $E = f(E_1, ..., E_n)$. By induction, assume that $E_i[\varsigma_1] \sim E_i[\varsigma_2]$ for $i = 1, ..., n$. Then, applying Definition 3.8, $E[\varsigma_1] = f(E_1[\varsigma_1], ..., E_n[\varsigma_1]) \sim f(E_1[\varsigma_2], ..., E_n[\varsigma_2]) = E[\varsigma_2]$. □

I now contribute an analogue of Proposition 3.10. Here $\varsigma_1 \sim^1 \varsigma_2$, for substitutions $\varsigma_1, \varsigma_2 : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$, means that $\varsigma_1(X) \sim \varsigma_2(X)$ for some variable $X \in \mathscr{X}$, and $\varsigma_1(Y) = \varsigma_2(Y)$ for all other variables $Y \neq X$.

Proposition 9.5 Let $\mathscr{L}$ be a simple language with variables. An equivalence relation $\sim$ on $\mathrm{T}_{\mathscr{L}}$ is a congruence for $\mathscr{L}$ iff $E[\varsigma_1] \sim E[\varsigma_2]$ for any $E \in \mathbb{T}_{\mathscr{L}}$ and any $\varsigma_1, \varsigma_2 : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ with $\varsigma_1 \sim^1 \varsigma_2$.

Proof: Suppose that $\sim$ satisfies the property of Proposition 9.5. It suffices to show that it then also satisfies the stronger property of Proposition 9.4. So let $E \in \mathbb{T}_{\mathscr{L}}$ and $\varsigma_1, \varsigma_2 : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ with $\varsigma_1 \sim \varsigma_2$. Define $\varsigma_2' : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ by $\varsigma_2'(X) := \varsigma_2(X)$ if X occurs in E, and $\varsigma_2'(X) := \varsigma_1(X)$ otherwise. Since E is finite it has only $n \in \mathbb{N}$ variables. Thus there are substitutions $\varsigma_1^i : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ for $i = 0, \ldots, n$ such that $\varsigma_1^0 = \varsigma_1$, $\varsigma_1^i \sim^1 \varsigma_1^{i+1}$ for all $0 \leq i < n$, and $\varsigma_1^n = \varsigma_2'$. Now $E[\varsigma_1] = E[\varsigma_1^0] \sim E[\varsigma_1^1] \sim \ldots \sim E[\varsigma_1^n] = E[\varsigma_2]$. □

Definition 3.12 of a congruence for $\mathscr{T}(\mathscr{L})$ was only stated for the case that $\mathscr{T}(D)$ is a context that features no other holes or variables than those from D. In that case, similar to Lemma 6.2, one may just as well assume the single hole in D and in $\mathscr{T}(D)$ to be a variable $X_1 \in \mathscr{X}$, so that D is a open $\mathscr{L}$-expression. The resulting notion of a congruence for $\mathscr{T}(\mathscr{L})$ is easily seen to be a special case of the following generalisation of this concept to translations between simple languages with variables.

Definition 9.6 Let $\mathscr{L}, \mathscr{L}'$ be simple languages with variables, and $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ a translation from $\mathscr{L}$ into $\mathscr{L}'$. A subset $\mathbf{W}$ of $\mathrm{T}_{\mathscr{L}'}$ is *closed* under $\mathscr{T}(\mathscr{L})$ if $\mathscr{T}(E)[\eta] \in \mathbf{W}$ for any expression $E \in \mathbb{T}_{\mathscr{L}}$ and substitution $\eta : \mathscr{X} \to \mathbf{W}$. An equivalence $\sim$ on a closed set $\mathbf{W}$ is a *congruence* for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ if $\mathscr{T}(E)[\eta_1] \sim \mathscr{T}(E)[\eta_2]$ for any $E \in \mathbb{T}_{\mathscr{L}}$ and any $\eta_1, \eta_2 : \mathscr{X} \to \mathbf{W}$ with $\eta_1 \sim \eta_2$.

If $\mathbf{W} \subseteq \mathrm{T}_{\mathscr{L}'}$ is closed under $\mathscr{T}(\mathscr{L})$, then surely $\mathrm{T}_{\mathscr{T}(\mathscr{L})} \subseteq \mathbf{W}$. The parameter $\mathbf{W}$ was not mentioned in Definition 3.12; there is was simply instantiated with $\mathrm{T}_{\mathscr{T}(\mathscr{L})}$. This is not always appropriate in the present setting with variables, since it is not guaranteed that $\mathrm{T}_{\mathscr{T}(\mathscr{L})}$ is closed. Instead, the following proposition gives a default choice of $\mathbf{W}$.

Proposition 9.7 Let $\mathscr{T}$ be a translation from $\mathscr{L}$ into $\mathscr{L}'$ that is correct w.r.t. a semantic translation $\mathbf{R} \subseteq \mathrm{T}_{\mathscr{L}'} \times \mathrm{T}_{\mathscr{L}}$. Then $\mathbf{R}(\mathscr{L})$ is closed under $\mathscr{T}(\mathscr{L})$.

Proof: Let $E \in \mathbb{T}_{\mathscr{L}}$ and $\eta : \mathscr{X} \to \mathbf{R}(\mathscr{L})$. Take $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ with $\eta \; \mathbf{R} \; \varsigma$. Then $\mathscr{T}(E)[\eta] \; \mathbf{R} \; E[\varsigma]$ by Definition 7.4. Since $E[\varsigma] \in \mathrm{T}_{\mathscr{L}}$ one has $\mathscr{T}(E)[\eta] \in \mathbf{R}(\mathscr{L})$. □

I now show that for each translation $\mathscr{T}$, valid up to $\backsim$, there exists a smallest semantic relation $\mathbf{R} \subseteq \backsim$ for which $\mathscr{T}$ is correct. The corresponding set $\mathbf{R}(\mathscr{L})$ is the smallest set that is closed under $\mathscr{T}(\mathscr{L})$.

Proposition 9.8 Let $\mathscr{T}$ be a translation from $\mathscr{L}$ into $\mathscr{L}'$ that is correct w.r.t. a semantic translation $\mathbf{R}' \subseteq \mathrm{T}_{\mathscr{L}'} \times \mathrm{T}_{\mathscr{L}}$ and let $\mathbf{W} \subseteq \mathrm{T}_{\mathscr{L}'}$ be closed under $\mathscr{T}(\mathscr{L})$. Then there $\mathscr{T}$ is correct w.r.t. a semantic translation $\mathbf{R} \subseteq \mathbf{R}'$ for which $\mathbf{R}(\mathscr{L}) \subseteq \mathbf{W}$.

Proof: For each ordinal λ define the relation $\mathbf{R}_\lambda \subseteq \mathrm{T}_{\mathscr{L}'} \times \mathrm{T}_{\mathscr{L}}$ as follows:

- $P' \; \mathbf{R}_0 \; P$ iff $P \in \mathrm{T}_{\mathscr{L}}$ and $P' = \mathscr{T}(P) \in \mathrm{T}_{\mathscr{L}'}$,
- $P' \; \mathbf{R}_{\lambda+1} \; P$ if $P = E[\varsigma]$ and $P' = \mathscr{T}(E)[\eta]$ for some $E \in \mathbb{T}_{\mathscr{L}}$ and substitutions ς, η with $\varsigma \; \mathbf{R}_\lambda \; \eta$,
- $\mathbf{R}_\lambda = \bigcup_{\kappa<\lambda} \mathbf{R}_\kappa$ if λ is a limit ordinal.

A straightforward induction yields that this is an increasing sequence: $\kappa < \lambda \Rightarrow \mathbf{R}_\kappa \subseteq \mathbf{R}_\lambda$. Thus, it must have a limit, which I call $\mathbf{R}$. Clearly, $\mathbf{R}$ is a semantic translation, and $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$. Another straightforward induction yields $\mathbf{R} \subseteq \mathbf{R}'$. A third induction yields $\mathbf{R}(\mathscr{L}) \subseteq \mathbf{W}$. □

Given a translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ and a set $\mathbf{W} \subseteq \mathrm{T}_{\mathscr{L}'}$ that is closed under $\mathscr{T}(\mathscr{L})$, I speak of the *congruence closure* $\approx^c_{\mathscr{T}(\mathscr{L}),\mathbf{W}}$ of $\approx$ w.r.t. $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$. This is the largest congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ that is contained in $\approx$. It can be characterised by $P \approx^c_{\mathscr{T}(\mathscr{L}),\mathbf{W}} Q$ for $P, Q \in \mathbf{W}$ iff $F[\eta_1] \approx F[\eta_2]$ for all $\mathscr{L}'$-expressions F of the form $\mathscr{T}(E)$ and all substitutions $\eta_1, \eta_2 : \mathscr{X} \to \mathbf{W}$ with $\eta_1(X) = P$ and $\eta_2(X) = Q$ for some $X \in \mathscr{X}$ and $\eta_1(Y) = \eta_2(Y)$ for all variables $Y \neq X$.

9.3 Congruence transfer properties for simple languages with variables

Using the above, I now generalise the remaining results of Section 3.4.

Theorem 9.9 Suppose $\mathscr{L}'$ is at least as expressive as $\mathscr{L}$ up to an equivalence or preorder $\mathrel{\scriptstyle\sim}$, and let $\approx \supseteq \mathrel{\scriptstyle\sim}$ be a congruence for $\mathscr{L}'$ that is coarser than or equal to $\mathrel{\scriptstyle\sim}$. Then $\approx$ is also a congruence for $\mathscr{L}$.

Proof: By assumption, there exists a valid translation $\mathscr{T}$ up to $\mathrel{\scriptstyle\sim}$ from $\mathscr{L}$ into $\mathscr{L}'$. Using Definition 7.5, it must be correct w.r.t. some semantic translation $\mathbf{R} \subseteq \mathrel{\scriptstyle\sim} \subseteq \approx$. Let $E \in \mathbb{T}_{\mathscr{L}}$ and $\varsigma_1, \varsigma_2 : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ with $\varsigma_1 \approx \varsigma_2$. As $\mathbf{R}$ is a semantic translation, there must be closed substitutions $\eta_1, \eta_2 : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ with $\eta_1 \mathbf{R} \varsigma_1$ and $\eta_2 \mathbf{R} \varsigma_2$. Then $\eta_1 \approx \varsigma_1 \approx \varsigma_2 \approx \eta_2$, so by Definition 7.4, $E[\varsigma_1] \mathbf{R}^{-1} \mathscr{T}(E)[\eta_1] \approx \mathscr{T}(E)[\eta_2] \mathbf{R} E[\varsigma_2]$. □

Based on its proof, Theorem 9.9 can be sharpened by merely assuming that $\approx$ is a congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{R}(\mathscr{L})$, and in view of Proposition 9.8, we may take $\mathbf{R}$ so that $\mathbf{R}(\mathscr{L}) \subseteq \mathbf{W}$ for a given closed set $\mathbf{W}$.

Theorem 9.9* Let $\mathscr{T}$ be a translation of $\mathscr{L}$ into $\mathscr{L}'$ that is valid up to $\mathrel{\scriptstyle\sim}$ and let $\approx \supseteq \mathrel{\scriptstyle\sim}$ be a congruence for $\mathscr{T}(\mathscr{L})$ on a closed set $\mathbf{W}$. Then $\approx$ is also a congruence for $\mathscr{L}$. □

Theorem 9.10 Let $\mathscr{T}$ be a translation of $\mathscr{L}$ into $\mathscr{L}'$ that is correct w.r.t. a semantic translation $\mathbf{R}$ and let $\approx \supseteq \mathbf{R}$ be a congruence for $\mathscr{L}$. Then $\approx$ is also a congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{R}(\mathscr{L})$.

Proof: Let $E \in \mathbb{T}_{\mathscr{L}}$ and $\eta_1, \eta_2 : \mathscr{X} \to \mathbf{R}(\mathscr{L})$ with $\eta_1 \approx \eta_2$. By the definition of $\mathbf{R}(\mathscr{L})$ there exists closed substitutions $\varsigma_1, \varsigma_2 : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ with $\eta_1 \mathbf{R} \varsigma_1$ and $\eta_2 \mathbf{R} \varsigma_2$. Thus $\varsigma_1 \approx \eta_1 \approx \eta_2 \approx \varsigma_2$, so by Definition 7.4 and Proposition 9.4, $\mathscr{T}(E)[\eta_1] \mathbf{R} E[\varsigma_1] \approx E[\varsigma_2] \mathbf{R}^{-1} \mathscr{T}(E)[\eta_2]$. □

9.4 Congruence reflection and closure for simple languages with variables

Theorem 3.14 generalises trivially to simple languages with variables.[10] Theorem 9.12 below moreover establishes a version of this result that does not require $\mathscr{T}$ to be compositional. The price to pay for that is that $\sim$ must contain the equivalence relation $\equiv_{\mathbf{R}}$ defined next.

Definition 9.11 Given any semantic translation $\mathbf{R} \subseteq \mathrm{T}_{\mathscr{L}'} \times \mathrm{T}_{\mathscr{L}}$, let $\equiv_{\mathbf{R}} \subseteq (\mathrm{T}_{\mathscr{L}} \uplus \mathbf{R}(\mathscr{L}))^2$ be the smallest equivalence relation on $\mathrm{T}_{\mathscr{L}} \uplus \mathbf{R}(\mathscr{L})$ containing $\mathbf{R}$.

Theorem 9.12 Let $\mathscr{T}$ be a translation between simple languages with variables $\mathscr{L}$ and $\mathscr{L}'$ that is correct w.r.t. a semantic translation $\mathbf{R}$. Then any equivalence relation $\sim \subseteq \mathbf{R}(\mathscr{L}) \times \mathbf{R}(\mathscr{L})$ that contains the restriction of $\equiv_{\mathbf{R}}$ to $\mathbf{R}(\mathscr{L})$ and is a congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{R}(\mathscr{L})$ can be extended to an equivalence $\sim_{\mathbf{R}}$ on $\mathrm{T}_{\mathscr{L}} \uplus \mathbf{R}(\mathscr{L})$, such that $\mathscr{T}$ is valid up to $\sim_{\mathbf{R}}$, and $\sim_{\mathbf{R}}$ also is a congruence for $\mathscr{L}$.

Proof: Define $\sim_{\mathbf{R}}$ on $\mathrm{T}_{\mathscr{L}} \uplus \mathbf{R}(\mathscr{L})$ by $P \sim_{\mathbf{R}} Q$ iff $P \equiv_{\mathbf{R}} R \sim S \equiv_{\mathbf{R}} Q$ for some $R, S \in \mathbf{R}(\mathscr{L})$. Since the restriction of $\equiv_{\mathbf{R}}$ to $\mathbf{R}(\mathscr{L})$ is contained in $\sim$, this relation is transitive. Trivially, it is also symmetric, since $\equiv_{\mathbf{R}}$ and $\sim$ are. It is reflexive, since $\sim$ is reflexive and for each $P \in \mathrm{T}_{\mathscr{L}} \uplus \mathbf{R}(\mathscr{L})$ there is an $R \in \mathbf{R}(\mathscr{L})$ with $P \equiv_{\mathbf{R}} R$. Thus $\sim_{\mathbf{R}}$ is an equivalence relation. On $\mathrm{T}_{\mathscr{T}(\mathscr{L})}$ the relation $\sim_{\mathbf{R}}$ coincides with $\sim$. Since $\mathbf{R} \subseteq \equiv_{\mathbf{R}} \subseteq \sim_{\mathbf{R}}$, $\mathscr{T}$ is valid up to $\sim_{\mathbf{R}}$. That $\sim_{\mathscr{T}}$ is a congruence for $\mathscr{L}$ follows by Theorem 9.9*. □

Theorem 9.14 below is my generalisation of Theorem 3.16; it needs a bit of preparation.

Lemma 9.13 If a translation $\mathscr{T}$ is correct w.r.t. a semantic translation $\mathbf{R}$, then $\equiv_{\mathbf{R}}$ is a congruence for $\mathscr{L}$.

[10] Its proof establishes A-validity of $\mathscr{T}$ up to $\sim_{\mathscr{T}}$, which is the same as C-validity and implies (C^--)validity.

Proof: Let $E \in \mathbb{T}_{\mathscr{L}}$ and $\varsigma_1, \varsigma_2 : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ with $\varsigma_1 \equiv^1_{\mathbf{R}} \varsigma_2$. Using Proposition 9.5 it suffices to show that $E[\varsigma_1] \equiv_{\mathbf{R}} E[\varsigma_2]$.

Let $X \in \mathscr{X}$ be such that $\varsigma_1(X) \equiv_{\mathbf{R}} \varsigma_2(X)$ and $\varsigma_1(Y) = \varsigma_2(Y)$ for all $Y \neq X$. Then, for some $n \geq 0$ there are $P_0, ..., P_n \in \mathrm{T}_{\mathscr{L}}$ and $Q_1, ..., Q_n \in \mathrm{T}_{\mathbf{R}(\mathscr{L})}$ with $\varsigma_1(X) = P_0 \,\mathbf{R}^{-1}\, Q_1 \,\mathbf{R}\, P_1 \,\mathbf{R}^{-1}\, Q_2 \,\mathbf{R}\, P_2 \,\mathbf{R}^{-1} \cdots \mathbf{R}\, P_n = \varsigma_2(X)$. For $i = 0, \ldots, n$ let $\varsigma_2^i : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ be given by $\varsigma_2^i(X) = P_i$ and $\varsigma_2^i(Y) = \varsigma_2(Y)$ for $Y \neq X$, and for $i = 1, \ldots, n$ let $\eta^i : \mathscr{X} \to \mathrm{T}_{\mathbf{R}(\mathscr{L})}$ be given by $\eta^i(X) = Q_i$ and $\eta^i(Y) = \eta(Y)$ for $Y \neq X$, using some $\eta : \mathscr{X} \to \mathrm{T}_{\mathbf{R}(\mathscr{L})}$ such that $\eta \;\mathbf{R}\; \varsigma_1$. Then $\varsigma_1 = \varsigma_2^0 \,\mathbf{R}^{-1}\, \eta^1 \,\mathbf{R}\, \varsigma_2^1 \,\mathbf{R}^{-1}\, \eta^2 \,\mathbf{R}\, \varsigma_2^2 \,\mathbf{R}^{-1} \cdots \mathbf{R}\, \varsigma_2^n = \varsigma_2$. So by Definition 7.4, $E[\varsigma_2^0] \,\mathbf{R}^{-1}\, \mathscr{T}(E)[\eta^1] \,\mathbf{R}\, E[\varsigma_2^1] \,\mathbf{R}^{-1}\, \mathscr{T}(E)[\eta^2] \,\mathbf{R}\, E[\varsigma_2^2] \,\mathbf{R}^{-1} \cdots \mathbf{R}\, E[\varsigma_2^n]$. Thus $E[\varsigma_1] \equiv_{\mathbf{R}} E[\varsigma_2]$. □

Theorem 9.14 Let $\mathscr{T}$ be a translation between simple languages with variables $\mathscr{L}$ and $\mathscr{L}'$ that is valid up to an equivalence $\approx$. Then $\mathscr{T}$ is valid even up to an equivalence $\approx^c_{\mathscr{T}}$, contained in $\approx$, such that

(1) the restriction of $\approx^c_{\mathscr{T}}$ to $\mathrm{T}_{\mathscr{L}}$ is the largest congruence for $\mathscr{L}$ contained in $\approx$,

and there exists a closed set $\mathbf{W}$ with $\mathrm{T}_{\mathscr{T}(\mathscr{L})} \subseteq \mathbf{W} \subseteq \mathrm{T}_{\mathscr{L}'}$ such that

(2) the restriction of $\approx^c_{\mathscr{T}}$ to $\mathbf{W}$ is the largest congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ that is contained in $\approx$, and

(3) each equivalence class of $\approx^c_{\mathscr{T}}$ is the union of an equivalence class of $\approx^c_{\mathscr{L}}$ and one of $\approx^c_{\mathscr{T}(\mathscr{L}),\mathbf{W}}$.

Proof: By assumption the translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ is correct w.r.t. a semantic translation $\mathbf{R} \subseteq \approx$. Take $\mathbf{W} := \mathbf{R}(\mathscr{L})$. Let $\approx^c_{\mathscr{L},\mathbf{R}}$, the *congruence closure* of $\approx$ w.r.t. $\mathscr{L}$ and $\mathbf{R}$, be the binary relation on $\mathrm{T}_{\mathscr{L}} \uplus \mathbf{W}$ defined by $R \approx^c_{\mathscr{L},\mathbf{R}} S$ iff $R \equiv_{\mathbf{R}} P \approx^c_{\mathscr{L}} Q \equiv_{\mathbf{R}} S$ for some $P, Q \in \mathrm{T}_{\mathscr{L}}$. Here $\approx^c_{\mathscr{L}}$ is the largest congruence for $\mathscr{L}$, defined on $\mathrm{T}_{\mathscr{L}}$, that is contained in $\approx$ (cf. Section 3.6).

As $\approx$ is an equivalence relation defined om $\mathrm{T}_{\mathscr{L}} \uplus \mathbf{W}$ that contains $\mathbf{R}$, and $\equiv_{\mathbf{R}}$ is the smallest such equivalence, one has $\equiv_{\mathbf{R}} \subseteq \approx$. By Lemma 9.13 $\equiv_{\mathbf{R}}$ is a congruence for $\mathscr{L}$. Since $\equiv_{\mathbf{R}}$ restricted to $\mathrm{T}_{\mathscr{L}}$ is a congruence for $\mathscr{L}$ contained in $\approx$, and $\approx^c_{\mathscr{L}}$ is the largest congruence for $\mathscr{L}$ contained in $\approx$, one has $P \equiv_{\mathbf{R}} Q \Rightarrow P \approx^c_{\mathscr{L}} Q$ for all $P, Q \in \mathrm{T}_{\mathscr{L}}$. From this it follows that $\approx^c_{\mathscr{L},\mathbf{R}}$ is transitive. Trivially, this relation is also symmetric, since $\equiv_{\mathbf{R}}$ and $\approx^c_{\mathscr{L}}$ are. It is reflexive, since $\approx^c_{\mathscr{L}}$ is reflexive and for each $R \in \mathrm{T}_{\mathscr{L}} \uplus \mathbf{W}$ there is a $P \in \mathrm{T}_{\mathscr{L}}$ with $R \equiv_{\mathbf{R}} P$. Thus, $\approx^c_{\mathscr{L},\mathbf{R}}$ is an equivalence relation. Since $\approx^c_{\mathscr{L}}$ and $\mathbf{R}$, and hence also $\equiv_{\mathbf{R}}$, are contained in $\approx$, so is $\approx^c_{\mathscr{L},\mathbf{R}}$. Moreover, $\mathbf{R} \subseteq \equiv_{\mathbf{R}} \subseteq \approx^c_{\mathscr{L},\mathbf{R}}$, so $\mathscr{T}$ is valid up to $\approx^c_{\mathscr{L},\mathbf{R}}$. It remains to check properties (1)–(3).

Since $\equiv_{\mathbf{R}}$ restricted to $\mathrm{T}_{\mathscr{L}}$ is included in $\approx^c_{\mathscr{L}}$, Property (1) of Theorem 9.14, saying that $\approx^c_{\mathscr{L},\mathbf{R}}$ restricted to $\mathrm{T}_{\mathscr{L}}$ coincides with $\approx^c_{\mathscr{L}}$, follows immediately from the definition of $\approx^c_{\mathscr{L},\mathbf{R}}$.

Regarding Property (2), by Theorem 9.10 (taking $\approx := \approx^c_{\mathscr{L},\mathbf{R}}$) $\approx^c_{\mathscr{L},\mathbf{R}}$ is a congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$, contained in $\approx$. To show that it is the largest, let $\sim \subseteq \mathbf{W} \times \mathbf{W}$ be any other congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ contained in $\approx$. By Lemma 9.13 and Theorem 9.10 also $\equiv_{\mathbf{R}}$ is a congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ contained in $\approx$. Let $\sim' \subseteq \mathbf{W} \times \mathbf{W}$ be the smallest equivalence on $\mathbf{W}$ containing both $\sim$ and $\equiv_{\mathbf{R}}$; one has $R \sim' S$ if $R = R_0 \sim S_0 \equiv_{\mathbf{R}} R_1 \sim S_1 \equiv_{\mathbf{R}} \cdots \equiv_{\mathbf{R}} R_k \sim S_k = S$ for some $k \geq 0$. Using Proposition 9.4, also $\sim'$ is a congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ contained in $\approx$.

Let $\sim_{\mathbf{R}}$ be the extension of $\sim'$ to $\mathrm{T}_{\mathscr{L}} \uplus \mathbf{W}$ as defined in the proof of Theorem 9.12: $P \sim_{\mathbf{R}} Q$ iff $P \equiv_{\mathbf{R}} R \sim' S \equiv_{\mathbf{R}} Q$ for some $R, S \in \mathbf{R}(\mathscr{L})$. This definition entails that $\sim_{\mathbf{R}} \subseteq \approx$, and by Theorem 9.12 $\sim_{\mathbf{R}}$ is a congruence for $\mathscr{L}$.

Now suppose $R \sim S$ with $R, S \in \mathbf{W}$. Take $P, Q \in \mathrm{T}_{\mathscr{L}}$ such that $R \,\mathbf{R}\, P$ and $S \,\mathbf{R}\, Q$. Then $P \sim_{\mathbf{R}} Q$. Since $\sim_{\mathbf{R}}$ is a congruence for $\mathscr{L}$ contained in $\approx$, and $\approx^c_{\mathscr{L}}$ is the largest such congruence, one has $P \approx^c_{\mathscr{L}} Q$, and thus $R \approx^c_{\mathscr{L},\mathbf{R}} S$. This shows that $\sim \subseteq \approx^c_{\mathscr{L},\mathbf{R}}$ and yields Property (2).

Each equivalence class of $\approx^c_{\mathscr{L},\mathbf{R}}$ contains a process $P \in \mathrm{T}_{\mathscr{L}}$ and its translation $\mathscr{T}(P) \in \mathbf{W}$. Using this, Property (3) is a simple consequence of (1) and (2). □

10 Variable binding

In this section I will generalise my theory of encodings and expressiveness to languages that merely satisfy requirement (iii) of Section 3; they may feature variables as well as additional syntactic constructs (beyond operators). I call such languages *closed-term languages*. In this paper I allow languages with any additional syntactic constructs, as long as they satisfy a postulate that will be formulated in Section 10.4.

Just like an n-ary operator f builds a new term from an n-tuple of subterms, the additional constructs that I consider build new terms from collections $\vec{E}$ of subterms. These collections may be finite or infinite, and whereas my notation $\vec{E}$ suggests that these terms are arranged in a list, or vector, I also allow sets, multisets, or other collections of subterms. When an additional construct is called $\mathscr{C}$, the term it builds from the subterms $\vec{E}$ can be abstractly denoted as $\mathscr{C}(\vec{E})$, even though notation differs in concrete cases. In case of the recursion construct $\mu X.E$ mentioned in Section 2, one has $\mathscr{C} = \mu X.$ and $\vec{E}$ is required to be a singleton term E.

Such a construct may bind certain variables within some of its arguments—the *scope* of the binding. An occurrence of a variable X in an expression is *bound* if it occurs within the scope of a construct that binds X, and *free* otherwise.

10.1 Substitution and α-conversion

I generalise Definition 6.3 to general languages.

Definition 10.1 A *substitution* in a language $\mathscr{L}$ is a function $\sigma : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$ from the variables to the $\mathscr{L}$-expressions—it is *closed* if $\sigma(X) \in \mathrm{T}_{\mathscr{L}}$ for all $X \in \mathscr{X}$. 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 $X \in \mathscr{X}$ is replaced by $\sigma(X)$**, after a suitable renaming of the bound variables in** E **to prevent variable capture**. Here *capture* is a situation where a free occurrence of a variable in $\sigma(Y)$, with $Y \in \mathscr{X}$, would become bound in $E[\sigma]$.

Definition 10.2 α*-conversion* is the act of renaming all occurrences of a bound variable X within the scope of its binding into another variable, say Y, while avoiding variable capture.

Write $E \stackrel{\alpha}{=} F$ if expression E can be converted into F by multiple acts of α-conversion.

The result of applying a substitution is determined only up to $\stackrel{\alpha}{=}$. For this reason it makes sense to employ a semantic equivalence $\sim$ with $\stackrel{\alpha}{=} \subseteq \sim$. However, if σ is a closed substitution, then $E[\sigma]$ is uniquely determined, without of the need for α-conversion, for there are no variables that could be captured.

Observation 10.3 If $E \stackrel{\alpha}{=} F$ then $fv(E) = fv(F)$.

For substitutions $\sigma, \xi : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$ write $\sigma \stackrel{\alpha}{=} \xi$ if $\sigma(X) \stackrel{\alpha}{=} \xi(X)$ for all $X \in \mathscr{X}$.

Observation 10.4 If $E \stackrel{\alpha}{=} F$ and $\sigma \stackrel{\alpha}{=} \xi$ then $E[\sigma] \stackrel{\alpha}{=} F[\xi]$.

The following notation and observation are used below.

Definition 10.5 The composition $\xi \bullet \sigma$ of substitutions $\sigma, \xi : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$ is given by $(\xi \bullet \sigma)(X) = \sigma(X)[\xi]$.

Observation 10.6 $E[\sigma][\xi] \stackrel{\alpha}{=} E[\xi \bullet \sigma]$.

10.2 Variables versus holes

In the presence of binding, there are two sensible ways to define substitution, and to capture both in one go, one makes a distinction between *variables* and *holes*. Both serve as basic building blocks for expressions, just like constants. Let $\mathscr{X}$ be the collection of variables, and $\mathscr{H}$ the collection of holes one may use in a given language $\mathscr{L}$. Then clause (i) in Section 2, the induction base in the definition of the $\mathscr{L}$-expressions, should be amended to

(i) $X \in \mathbb{T}_{\mathscr{L}}$ for each variable $X \in \mathscr{X}$ and for each hole $X \in \mathscr{H}$.

Only variables can be bound; an occurrence of a hole in an $\mathscr{L}$-expression is always free. An expression without holes is called a *term*, and one with holes a *context*.

Example 10.7 Consider the expression $\mu X.(aX + bY)$, with $Y \neq X$. Here $a\_$ is a unary action prefix operator and $+$ a choice, as occur in CCS and many other process calculi. This expression denotes a process that can perform the action a and subsequently behave like itself—this is the binding at work—or perform the action b and behave like the process Y, which is as of yet not specified. In other words, $\mu X.(aX + bY)$ can perform infinitely many a-actions in succession, or finitely many, followed by a b, and whatever is specified by Y. Now if σ is a substitution with $\sigma(X) = c\mathbf{0}$ (a process that first performs action c and then stops) and $\sigma(Y) = d\mathbf{0}$, then $(\mu X.(aX + bY))[\sigma] = \mu X.(aX + bd\mathbf{0})$. Here we see that $d\mathbf{0}$ is substituted for Y, but no substitution is made for X. This is because this variable only occurs bound. However, one does have that $(aX + \mu X.(aX + bY))[\sigma] = ac\mathbf{0} + \mu X.(aX + bd\mathbf{0})$.

Next consider the substitution ξ with $\xi(Y) = dX$. Here is makes a crucial difference whether Y is a variable or a hole. In case Y is a hole, applying the substitution ξ is straightforward:

$$(\mu X.(aX + bY))[\xi] = \mu X.(aX + bdX).$$

However, in case Y is a variable, it is seen as box whose entire contents is outside the scope of the binder μX. To keep this so after substitution, one renames the bound variable X into a fresh variable Z before performing the substitution:

$$(\mu X.(aX + bY))[\xi] = \mu Z.(aZ + bdX).$$

This practice rests on the assumption that $\mu X.(aX + bY)$ is semantically equivalent to $\mu Z.(aZ + bY)$, in the sense that for most purposes either of these expressions can be replaced by the other without ill effects. However, when substituting for holes, such replacements do have ill effects: if Y is a hole then

$$(\mu X.(aX + bY))[\xi] = \mu X.(aX + bdX) \neq \mu Z.(aZ + bdX) = (\mu Z.(aZ + bY))[\xi].$$

This example suggests that variables and holes do not mix well. In all system description languages I know, the valid expressions are terms rather than contexts, that is, they do not contain holes. This is the reason I did not introduce holes in Section 2, and will not employ them below. Contexts are sometimes used in the analysis of languages, but I do not need them in the remainder of this paper. Earlier in this paper I used holes and contexts, but in the sections where this happened I didn't consider variable binding.

10.3 Mixing name binding and process variable binding

Some languages may have multiple binding constructs. As an example I consider a version of the π-calculus, which features *name binding*, that also has the process variable binding construct $\mu X.E$. The process $P := \mu X.(\bar{z}y|x(y).X)$ features both bindings. This process does two things in parallel: (1) it sends the name y on the channel z, and (2) it reads a name y on the channel x, after which it behaves like P again. Given the semantics of $\mu X.E$, the behaviour of P is by definition equal to the behaviour of its unfolding $(\bar{z}y|x(y).X)[^P/_X]$. Here $[^P/_X]$ denotes the substitution that sends X to P, and leaves all other variables untouched. There are essentially two different ways to define the substitution $[^P/_X]$, and thereby the semantics of P.

My default assumption, implicit in Definitions 10.1 and 10.2, is that a substitution for process variables is completely agnostic to binding issues on the level of names. This yields $(\bar{z}y|x(y).X)[^P/_X] =$

$\bar{z}y|x(y).P$. In this approach, process P will, after reading a name b on channel x, send this b on on channel z. This happens because the name y is bound by the binder $x(y)$ and this binding captures the free name y occurring in the P that is substituted for X. Hence from the perspective of the binder $x(y)$, the variable X is treated more like a hole. One can define a concept of α-conversion for names, as is usual in the π-calculus, but this notion is a different one from the one for variables from Definition 10.2.

The alternative approach is to treat variable and name binding on equal footing, and modify Definitions 10.1 and 10.2 to avoid capture of free names as well as variables. In this approach the free name y in P remains free when P is substituted for X; we get $(\bar{z}y|x(y).X)[P/X] = \bar{z}y|x(u).P$, where the bound name y is renamed into u in order to prevent capture of y. Now P keeps sending y on channel z, no matter what its reads to channel x. We can then use a concept of α-conversion that incorporates names and variables in one go.

In what follows, it doesn't matter which is these two approaches is chosen. Observations 10.3–10.6 hold either way.

10.4 A unique decomposition of terms

An expression E of the form $f(E_1,\ldots,E_n)$ can be written as $H[\xi]$, where H is an expression $f(X_1,\ldots,X_n)$ and ξ is a substitution with $\xi(X_i) = E_i$ for $i = 1,\ldots,n$. Here H and $\xi\restriction\{X_1,\ldots,X_n\}$ are completely determined by E, except for the choice of the variables $X_1,\ldots,X_n$. The term H is called a *head* of E. In this paper, for the proofs below, I propose a unique decomposition of expressions E into H and ξ, by making an arbitrary choice for the variables $X_1,\ldots,X_n$. Moreover, I extend this decomposition to all terms E that are not variables. This requires a postulate that says, in essence, that such a decomposition is always possible, and restricting attention to languages satisfying this postulate.

Definition 10.8 ([42]) A term $E \in \mathbb{T}_{\mathscr{L}}$ is a *prefix* of a term F, written $E \leq F$, if $F \stackrel{\alpha}{=} E[\xi]$ for some substitution ξ.

Using Observation 10.6, $\leq$ is reflexive and transitive, and hence a preorder. Write $\equiv$ for the kernel of $\leq$, i.e. $E \equiv F$ iff $E \leq F \wedge F \leq E$. Note that $E \stackrel{\alpha}{=} F$ implies $E \equiv F$. If $E \equiv F$ then E can be converted into F by means of an injective renaming of its variables.

Definition 10.9 ([42]) A 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 a 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.

Example 10.10 Let c be a constant, g a unary operator, f a binary operator, and $\mu X.E$ the recursion construct of Section 2. Then $f(X,Y)$ is a head of the term $f(c,g(c))$, and $\mu X.f(Y,g(g(X)))$ is a head of $\mu X.f(g(c),g(g(X)))$. See [42] for further detail.

Postulate 10.11 ([42]) Each expression E, if not a variable, has a head that is unique up to $\equiv$.

This is easy to show for each common type of system description language, including the ones with the additional constructs described in the introduction to Section 10:

Proof sketch: Call a subterm F of a term E *free* if F doesn't contain any free variables that are bound in E. Note that a subterm G of free subterm F of E is a free subterm of E. A *proper* subterm of E is any subterm different from E itself. A (free) proper subterm F of E is *maximal* if it is not a proper subterm of another (free) subterm of E. Now obtain a head of an expression $E \notin \mathscr{X}$ by changing all its maximal free proper subterm occurrences into fresh variables. □

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 in this paper I consider only languages that satisfy this postulate.

Proposition 10.12 Let $\mathscr{L}$ be a language. There exists a set $\mathbb{H} \subseteq \mathbb{T}_{\mathscr{L}}$ of *standard heads* and a *decomposition* function $decomp : \mathbb{T}_{\mathscr{L}} \backslash \mathscr{X} \to \mathbb{H} \times (\mathscr{X} \to \mathbb{T}_{\mathscr{L}})$ that associates with each term $E \notin \mathscr{X}$ a pair (H, ξ) of a standard head $H \in \mathbb{H}$ and a substitution $\xi : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$, such that

(i) $E \stackrel{\alpha}{=} H[\xi]$,
(ii) $\xi(X)$ is a proper subterm of E for all $X \in fv(H)$, whereas $\xi(X) = X$ when $X \notin fv(H)$,
(iii) if $E' \stackrel{\alpha}{=} E$ then $decomp(E') \stackrel{\alpha}{=} decomp(E)$, and
(iv) if $H \in \mathbb{H}$ and $\xi : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$ then $decomp(H[\xi]) \stackrel{\alpha}{=} (H, \xi)$.

Here $(H', \xi') \stackrel{\alpha}{=} (H, \xi)$ is a shorthand for "$H' = H$ and $\xi'(X) \stackrel{\alpha}{=} \xi(X)$ for all $X \in \mathscr{X}$".

Proof: Pick a representative from each $\equiv$-equivalence class of heads, and call the chosen representatives *standard heads*. By Postulate 10.11, each expression $E \notin \mathscr{X}$ can be written as $E \stackrel{\alpha}{=} H[\xi]$, with ξ some substitution and H the unique standard head of E. Define $decomp(E) := (H, \xi)$ by picking a substitution ξ for which this holds, and which satisfies $\xi(X) = X$ when $X \notin fv(H)$.

Now *decomp* satisfies (i) and (ii) by construction. Regarding (iii), if $E' \stackrel{\alpha}{=} E$ then E' and E must have the same standard head H, so there are substitutions ξ' and ξ such that $H[\xi'] \stackrel{\alpha}{=} E' \stackrel{\alpha}{=} E \stackrel{\alpha}{=} H[\xi]$. This implies $\xi'(X) \stackrel{\alpha}{=} \xi(X)$ for each $X \in fv(H)$, so $decomp(E') \stackrel{\alpha}{=} decomp(E)$. Towards (iv), let $H \in \mathbb{H}$ and $\xi : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$. By construction, H is the standard head of $H[\xi]$. Hence $decomp(H[\xi]) = (H, \xi')$ for some substitution ξ'. Again $H[\xi] \stackrel{\alpha}{=} H[\xi']$, so $decomp(E') \stackrel{\alpha}{=} decomp(E)$. □

10.5 Compositionality

The following definition generalises the concept of compositionality from Section 6 to general languages, that may feature variable binding.

Definition 10.13 A translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ is *compositional* if

(1) $\mathscr{T}(E[\sigma]) \stackrel{\alpha}{=} \mathscr{T}(E)[\mathscr{T} \circ \sigma]$ for each $E \in \mathbb{T}_{\mathscr{L}}$ and $\sigma : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$,
(2) $E \stackrel{\alpha}{=} F$ implies $\mathscr{T}(E) \stackrel{\alpha}{=} \mathscr{T}(F)$ for all $E, F \in \mathbb{T}_{\mathscr{L}}$,
(3) and moreover $\mathscr{T}(X) = X$ for each $X \in \mathscr{X}$.

In the absence of binding, $\stackrel{\alpha}{=}$ is simply equality, and the above definition is exactly the characterisation of compositionality given in Proposition 6.4. Requiring equality rather than $\stackrel{\alpha}{=}$ even in the presence of variable binding is too demanding, as the following example illustrates.

Example 10.14 Take a source language $\mathscr{L}$ that features a unary replication operator !, as in the π-calculus, and a target language $\mathscr{L}'$ that instead has a recursion construct $\mu X.E$; both languages have a constant $\mathbf{0}$. A suitable translation $\mathscr{T}$ satisfies $\mathscr{T}(!X_1) = \mu X.(X|X_1)$ and $\mathscr{T}(\mathbf{0}) = \mathbf{0}$. Applying Definition 10.13(1) with $\sigma(X_1) = \mathbf{0}$ gives $\mathscr{T}(!\mathbf{0}) \stackrel{\alpha}{=} \mu X.(X|\mathbf{0})$, whereas applying it with $\sigma(X_1) = X$ gives $\mathscr{T}(!X) \stackrel{\alpha}{=} \mu Y.(Y|X)$. Here the bound variable X needed to be renamed (into Y) to avoid capture of the free variable X that is substituted for X_1. Furthermore, applying Definition 10.13(1) with $E := !X$ and $\sigma(X) := \mathbf{0}$ gives $\mathscr{T}(!0) \stackrel{\alpha}{=} \mathscr{T}(!X)[\mathscr{T} \circ \sigma] \stackrel{\alpha}{=} \mu Y.(Y|X)[\mathscr{T} \circ \sigma] = \mu Y.(Y|0)$. Since $X \neq Y$, this shows that $\stackrel{\alpha}{=}$ cannot consistently be replaced by $=$.

Lemma 10.15 If $\mathscr{T}_1 : \mathbb{T}_{\mathscr{L}_1} \to \mathbb{T}_{\mathscr{L}_2}$ and $\mathscr{T}_2 : \mathbb{T}_{\mathscr{L}_2} \to \mathbb{T}_{\mathscr{L}_3}$ are compositional translations, then so is their composition $\mathscr{T}_2 \circ \mathscr{T}_1 : \mathbb{T}_{\mathscr{L}_1} \to \mathbb{T}_{\mathscr{L}_3}$, defined by $\mathscr{T}_2 \circ \mathscr{T}_1(E) := \mathscr{T}_2(\mathscr{T}_1(E))$ for all $E \in \mathscr{L}_1$.

Proof: (1) $\mathscr{T}_2(\mathscr{T}_1(E[\sigma])) \stackrel{\alpha}{=} \mathscr{T}_2(\mathscr{T}_1(E)[\mathscr{T}_1 \circ \sigma]) \stackrel{\alpha}{=} \mathscr{T}_2(\mathscr{T}_1(E))[\mathscr{T}_2 \circ \mathscr{T}_1 \circ \sigma])$ for each $\sigma : \mathscr{X} \rightharpoonup \mathbb{T}_{\mathscr{L}_1}$ and $E \in \mathbb{T}_{\mathscr{L}_1}$. Here the derivation of the first $\stackrel{\alpha}{=}$ uses Property (2) of Definition 10.13—and this is the reason for requiring that property.

(2) $E \stackrel{\alpha}{=} F$ implies $\mathscr{T}_1(E) \stackrel{\alpha}{=} \mathscr{T}_1(F)$ and hence $\mathscr{T}_2(\mathscr{T}_1(E)) \stackrel{\alpha}{=} \mathscr{T}_2(\mathscr{T}_1(F))$ for all $E, F \in \mathbb{T}_{\mathscr{L}}$.

(3) $\mathscr{T}_2(\mathscr{T}_1(X)) = \mathscr{T}_2(X) = X$ for each $X \in \mathscr{X}$. □

10.6 Validity for closed-term languages

Next I generalise the notion of validity chosen in Section 7 to general closed-term languages, at the same time dropping the restriction that closed terms must be translated into closed terms. Definitions 7.3, 7.4 and 7.5 can be lifted verbatim.

Definition 7.3 A *semantic translation* from $\mathscr{L}$ into $\mathscr{L}'$ is a relation $\mathbf{R} \subseteq \mathrm{T}_{\mathscr{L}'} \times \mathrm{T}_{\mathscr{L}}$ such that $\forall P \in \mathrm{T}_{\mathscr{L}}.\, \exists P' \in \mathrm{T}_{\mathscr{L}'}.\, P'\ \mathbf{R}\ P$.

Definition 7.4 A translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ is *correct* w.r.t. a semantic translation $\mathbf{R}$ if $\mathscr{T}(E)[\eta]\ \mathbf{R}\ E[\varsigma]$ for all expressions $E \in \mathbb{T}_{\mathscr{L}}$ and all closed substitutions $\eta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ and $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ with $\eta\ \mathbf{R}\ \varsigma$.

Definition 7.5 A translation $\mathscr{T}$ is *valid* up to $\leftrightarrows$ iff it is correct w.r.t. some semantic translation $\mathbf{R} \subseteq \leftrightarrows$.

Given a relation $\mathbf{R} \subseteq \mathrm{T}_{\mathscr{L}} \times \mathrm{T}_{\mathscr{L}'}$, define $\mathbf{R}^{\alpha}$ by $Q\ \mathbf{R}^{\alpha}\ P$ iff $\exists Q', P'.\ Q \stackrel{\alpha}{=} Q'\ \mathbf{R}\ P' \stackrel{\alpha}{=} P$.

Lemma 10.16 If a translation $\mathscr{T}$ between languages $\mathscr{L}$ and $\mathscr{L}'$ is correct w.r.t. a semantic translation $\mathbf{R}$, then it is also correct w.r.t. $\mathbf{R}^{\alpha}$.

Proof: Let $\mathbf{R} \subseteq \mathrm{T}_{\mathscr{L}} \times \mathrm{T}_{\mathscr{L}'}$ be a semantic translation, and $\mathscr{T}$ a translation that is correct w.r.t. $\mathbf{R}$. Let $E \in \mathbb{T}_{\mathscr{L}}$, $\eta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ and $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$, with $\eta\ \mathbf{R}^{\alpha}\ \varsigma$. Then there must be closed substitutions $\eta' : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ and $\varsigma' : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ with $\eta \stackrel{\alpha}{=} \eta'\ \mathbf{R}\ \varsigma' \stackrel{\alpha}{=} \varsigma$. Now $\mathscr{T}(E)[\eta']\ \mathbf{R}\ E[\varsigma']$, as $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$. By Observation 10.4 $\mathscr{T}(E)[\eta] \stackrel{\alpha}{=} \mathscr{T}(E)[\eta']\ \mathbf{R}\ E[\varsigma'] \stackrel{\alpha}{=} E[\varsigma]$, so $\mathscr{T}(E)[\eta]\ \mathbf{R}^{\alpha}\ E[\varsigma]$. It follows that $\mathscr{T}$ is correct w.r.t. $\mathbf{R}^{\alpha}$. □

10.7 Valid translations can be made compositional

This section is devoted to proving a generalisation to closed-term languages of Theorem 7.8, showing that any valid translation can be converted into one that also is compositional.

Theorem 10.17 Suppose $\leftrightarrows \supseteq \stackrel{\alpha}{=}$. Let $\mathscr{L}$ and $\mathscr{L}'$ be closed-term languages. If any valid translation from $\mathscr{L}$ into $\mathscr{L}'$ up to $\leftrightarrows$ exists, then there exists a compositional translation that is valid up to $\leftrightarrows$.

Proof: Let $\mathscr{T}_1$ be a valid translation from $\mathscr{L}$ into $\mathscr{L}'$ up to $\leftrightarrows$. It must be correct w.r.t. a semantic translation $\mathbf{R} \subseteq \leftrightarrows$. Take $\mathbb{H}$ and *decomp* as specified in Proposition 10.12, and define the translation $\mathscr{T}$ inductively by

$$\begin{array}{ll} \mathscr{T}(X) := X & \text{for } X \in \mathscr{X} \\ \mathscr{T}(E) := \mathscr{T}_1(H)[\mathscr{T} \circ \xi] & \text{when } \mathit{decomp}(E) = (H, \xi). \end{array}$$

First I show that $\mathscr{T}$ is compositional. It satisfies Property (3) of Definition 10.13 by construction. I establish Property (2) by structural induction on E. The base case $E \in \mathscr{X}$ is trivial. Suppose $E' \stackrel{\alpha}{=} E \notin \mathscr{X}$. Let $\mathit{decomp}(E') = (H', \xi')$ and $\mathit{decomp}(E) = (H, \xi)$. Then $H' = H$ and $\xi'(X) \stackrel{\alpha}{=} \xi(X)$ for all $X \in \mathscr{X}$, using Proposition 10.12(iii). Thus $\mathscr{T}_1(H') = \mathscr{T}_1(H)$. By induction $\mathscr{T}(\xi'(X)) \stackrel{\alpha}{=} \mathscr{T}(\xi(X))$ for all $X \in \mathscr{X}$, using Proposition 10.12(ii). So $\mathscr{T}(E') = \mathscr{T}_1(H')[\mathscr{T} \circ \xi'] \stackrel{\alpha}{=} \mathscr{T}_1(H)[\mathscr{T} \circ \xi] = \mathscr{T}(E)$ by Observation 10.4.

Next I establish Property (1) by structural induction on E. So let $E \in \mathbb{T}_{\mathscr{L}}$ and $\sigma : \mathscr{X} \to \mathbb{T}_{\mathscr{L}}$. I have to show that $\mathscr{T}(E[\sigma]) \stackrel{\alpha}{=} \mathscr{T}(E)[\mathscr{T} \circ \sigma]$. The case $E \in \mathscr{X}$ is trivial, so let $\mathit{decomp}(E) = (H, \xi)$. By Proposition 10.12(ii) and the induction hypothesis $\mathscr{T}(\xi(X)[\sigma]) \stackrel{\alpha}{=} \mathscr{T}(\xi(X))[\mathscr{T} \circ \sigma]$ for all $X \in \mathscr{X}$. Thus

$$\begin{array}{lll} (\mathscr{T} \circ (\sigma \bullet \xi))(X) = \mathscr{T}((\sigma \bullet \xi)(X)) & & \text{by definition of functional composition } \circ \\ = \mathscr{T}(\xi(X)[\sigma]) & & \text{by definition of the operation } \bullet \text{ on substitutions} \\ \stackrel{\alpha}{=} \mathscr{T}(\xi(X))[\mathscr{T} \circ \sigma] & & \text{derived above} \\ = ((\mathscr{T} \circ \sigma) \bullet (\mathscr{T} \circ \xi))(X) & & \text{by definition of the operations } \circ \text{ and } \bullet. \end{array}$$

This shows that the substitutions $\mathscr{T} \circ (\sigma \bullet \xi)$ and $(\mathscr{T} \circ \sigma) \bullet (\mathscr{T} \circ \xi)$ are equal up to α-conversion, implying

$F[\mathscr{T}\circ(\sigma\bullet\xi)] \stackrel{\alpha}{=} F[(\mathscr{T}\circ\sigma)\bullet(\mathscr{T}\circ\xi)]$ for all terms $F\in\mathbb{T}_{\mathscr{L}'}$, using Observation 10.4.

Hence $\mathscr{T}(E[\sigma]) \stackrel{\alpha}{=} \mathscr{T}(H[\xi][\sigma])$ since $E\stackrel{\alpha}{=}H[\xi]$ by 10.12(i), also using 10.4 and (2)
$\stackrel{\alpha}{=} \mathscr{T}(H[\sigma\bullet\xi])$ by Observation 10.6, again using (2)
$\stackrel{\alpha}{=} \mathscr{T}_1(H)[\mathscr{T}\circ(\sigma\bullet\xi)]$ by definition of $\mathscr{T}$, using 10.12(iv), (2) and 10.4
$\stackrel{\alpha}{=} \mathscr{T}_1(H)[(\mathscr{T}\circ\sigma)\bullet(\mathscr{T}\circ\xi)]$ derived above
$\stackrel{\alpha}{=} (\mathscr{T}_1(H)[\mathscr{T}\circ\xi])[\mathscr{T}\circ\sigma]$ by Observation 10.6
$\stackrel{\alpha}{=} \mathscr{T}(E)[\mathscr{T}\circ\sigma]$ by definition of $\mathscr{T}$.

Note that $\mathbf{R}^\alpha\subseteq\sim$. It remains to be shown that $\mathscr{T}$ is correct w.r.t. $\mathbf{R}^\alpha$, i.e. that $\mathscr{T}(E)[\eta]\ \mathbf{R}^\alpha\ E[\varsigma]$ for all expressions $E\in\mathbb{T}_{\mathscr{L}}$ and all closed substitutions $\eta:\mathscr{X}\to\mathrm{T}_{\mathscr{L}'}$ and $\varsigma:\mathscr{X}\to\mathrm{T}_{\mathscr{L}}$ with $\eta\ \mathbf{R}^\alpha\ \varsigma$. Let η and ς be such substitutions. I proceed with structural induction on E. The base case $E\in\mathscr{X}$ is trivial, so let $\mathit{decomp}(E)=(H,\xi)$. By Proposition 10.12(ii) and the induction hypothesis, $\mathscr{T}(\xi(X))[\eta]\ \mathbf{R}^\alpha\ \xi(X)[\varsigma]$ for all $X\in\mathscr{X}$. Using Definition 10.5, this can be reworded as $\eta\bullet(\mathscr{T}\circ\xi)\ \mathbf{R}^\alpha\ \varsigma\bullet\xi$. (*)

Therefore $\mathscr{T}(E)[\eta] = \mathscr{T}_1(H)[\mathscr{T}\circ\xi][\eta]$ by definition of $\mathscr{T}$
$\stackrel{\alpha}{=} \mathscr{T}_1(H)[\eta\bullet(\mathscr{T}\circ\xi)]$ by Observation 10.6
$\mathbf{R}^\alpha\ H[\varsigma\bullet\xi]$ by (*) above, as $\mathscr{T}_1$ is correct w.r.t. $\mathbf{R}$, and thus w.r.t. $\mathbf{R}^\alpha$
$\stackrel{\alpha}{=} H[\xi][\varsigma]$ by Observation 10.6
$\stackrel{\alpha}{=} E[\varsigma]$ by Proposition 10.12(1) and Observation 10.4. □

10.8 Free-variable respecting translations

In Sections 7–9 I restricted attention to translations $\mathscr{T}:\mathbb{T}_{\mathscr{L}}\to\mathbb{T}_{\mathscr{L}'}$ with the property that $\mathscr{T}(P)\in\mathrm{T}_{\mathscr{L}'}$ for all $P\in\mathrm{T}_{\mathscr{L}}$—that is, closed terms are translated into closed terms. This restriction (only) affected the definitions of A- and B-validity, the argument in Section 7.2 that B^--validity is implied by D^--validity, the argument in Section 7.3 that C^--validity implies B^--validity, Proposition 7.6, Theorem 7.8, Proposition 9.8, and the generalisation of Theorem 3.14 to simple languages with variables.

Throughout the literature 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* [42]. The following result says that for the purpose of comparing the expressiveness of languages (Definition 2.2) restricting attention to (compositional) fvr-translations is quite harmless. The above-mentioned restriction that $\mathscr{T}(P)\in\mathrm{T}_{\mathscr{L}'}$ for all $P\in\mathrm{T}_{\mathscr{L}}$ is even weaker, and thus also quite harmless.

Proposition 10.18 Let $\mathscr{L}$ and $\mathscr{L}'$ be closed-term languages with $\mathrm{T}_{\mathscr{L}}\neq\emptyset$. If any valid translation from $\mathscr{L}$ into $\mathscr{L}'$ up to $\sim$ exists, then there exists a compositional fvr-translation that is valid up to $\sim$.

Proof: Let $\mathscr{T}_2$ be a valid translation from $\mathscr{L}$ into $\mathscr{L}'$ up to $\sim$. It must be correct w.r.t. a semantic translation $\mathbf{R}\subseteq\sim$. Take a pair $(Q,P)\in\mathbf{R}$. For $E\in\mathbb{T}_{\mathscr{L}}$ let $\sigma_E:\mathscr{X}\to\mathbb{T}_{\mathscr{L}}$ and $\zeta_E:\mathscr{X}\to\mathbb{T}_{\mathscr{L}'}$ be the substitutions given by $\sigma_E(X)=P$ and $\zeta_E(X)=Q$ for all $X\notin\mathit{fv}(E)$, and $\sigma_E(X)=\zeta_E(X)=X$ for all $X\in\mathit{fv}(E)$. Define the translation $\mathscr{T}_1:\mathbb{T}_{\mathscr{L}}\to\mathbb{T}_{\mathscr{L}'}$ by $\mathscr{T}_1(E)=\mathscr{T}_2(E)[\zeta_E]$ for all $E\in\mathbb{T}_{\mathscr{L}}$. By construction, $\mathscr{T}_1$ is an fvr-translation. To show that it is correct w.r.t. $\mathbf{R}$, take $E\in\mathbb{T}_{\mathscr{L}}$ and select closed substitutions $\eta:\mathscr{X}\to\mathrm{T}_{\mathscr{L}'}$ and $\varsigma:\mathscr{X}\to\mathrm{T}_{\mathscr{L}}$ with $\eta\ \mathbf{R}\ \varsigma$. For $X\in\mathit{fv}(E)$ one derives that $(\eta\bullet\zeta_E)(X)=\eta(X)\ \mathbf{R}\ \varsigma(X)=(\varsigma\bullet\sigma_E)(X)$, and for $X\notin\mathit{fv}(E)$ that $(\eta\bullet\zeta_E)(X)=Q\ \mathbf{R}\ P=(\varsigma\bullet\sigma_E)(X)$. So $(\eta\bullet\zeta_E)\ \mathbf{R}\ (\varsigma\bullet\sigma_E)$. Thus $\mathscr{T}_1(E)[\eta]=\mathscr{T}_2(E)[\zeta_E][\eta]=\mathscr{T}_2(E)[\eta\bullet\zeta_E]\ \mathbf{R}\ E[\varsigma\bullet\sigma_E]=E[\varsigma]$. Here, in the second step, one has $=$ rather than $\stackrel{\alpha}{=}$, because ζ_E is such that no variable is at risk of being captured in a substitution $F[\zeta_E]$, while η and $\eta\bullet\zeta_E$ are closed substitutions.

Hence there exists an fvr-translation $\mathscr{T}_1$ from $\mathscr{L}$ into $\mathscr{L}'$ that is valid up to $\sim$. Now in the proof of Theorem 10.17, by induction it follows that if $\mathscr{T}_1$ is fvr, then so is $\mathscr{T}$. □

In case $\mathrm{T}_{\mathscr{L}} = \emptyset$, that is, the language $\mathscr{L}$ has no closed terms, each translation from $\mathscr{L}$ to $\mathscr{L}'$ is trivially valid, with the empty set as semantic translation, so $\mathscr{L}$ is the least expressive closed term language.

The next result gives an alternative characterisation of validity for compositional fvr-translations. This is the counterpart of the requirement of A-validity from Section 7.

Proposition 10.19 Suppose $\leftrightsquigarrow \supseteq \stackrel{\alpha}{=}$. A compositional fvr-translation $\mathscr{T}$ is valid up to $\leftrightsquigarrow$ iff $\mathscr{T}(P) \leftrightsquigarrow P$ for each $P \in \mathrm{T}_{\mathscr{L}}$.

Proof: Let $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be valid up to $\leftrightsquigarrow$ and $P \in \mathrm{T}_{\mathscr{L}}$. Let $\mathscr{T}$ be correct w.r.t. the semantic translation $\mathbf{R} \subseteq \leftrightsquigarrow$. Then $\mathscr{T}(P)\ \mathbf{R}\ P$, so $\mathscr{T}(P) \leftrightsquigarrow P$.

Now let $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be a compositional translation such that $\mathscr{T}(P) \leftrightsquigarrow P$ for each $P \in \mathrm{T}_{\mathscr{L}}$. Define $\mathbf{R}$ by $Q\ \mathbf{R}\ P$ iff $Q \stackrel{\alpha}{=} \mathscr{T}(P)$. Then $\mathbf{R}$ is a semantic translation and $\mathbf{R} \subseteq \leftrightsquigarrow$. Let $E \in \mathbb{T}_{\mathscr{L}}$, and let $\eta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ and $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ be closed substitutions with $\eta\ \mathbf{R}\ \varsigma$. Then $\eta \stackrel{\alpha}{=} \mathscr{T} \circ \varsigma$. Property (1) of Definition 10.13 yields $\mathscr{T}(E[\varsigma]) \stackrel{\alpha}{=} \mathscr{T}(E)[\mathscr{T} \circ \varsigma] \stackrel{\alpha}{=} \mathscr{T}(E)[\eta]$, where the second step follows from Observation 10.4. This gives $\mathscr{T}(E)[\eta]\ \mathbf{R}\ E[\varsigma]$. Hence $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$ and $\mathscr{T}$ is valid up to $\sim$. □

10.9 Congruence transfer, reflection and closure properties for closed term languages

In this section I generalise the results from Section 9, and thereby those from Section 3, to arbitrary closed term languages.

First note that Observations 9.1 and 9.2 as well as Theorem 9.3 generalise straightforwardly to general closed term languages.

Definition 3.8 of a congruence is stated in terms of the n-ary operators of a language, and is therefore inadequate for general closed term languages. Instead I will use its characterisation from Proposition 9.4 as the definition.

Definition 10.20 Let $\mathscr{L}$ be a closed term language. An equivalence relation $\sim$ on $\mathrm{T}_{\mathscr{L}}$ is a *congruence* for $\mathscr{L}$ iff $E[\varsigma_1] \sim E[\varsigma_2]$ for any $E \in \mathbb{T}_{\mathscr{L}}$ and any $\varsigma_1, \varsigma_2 : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ with $\varsigma_1 \sim \varsigma_2$.

This is called a *lean* congruence in [43]; in the presence of recursion, a stricter condition, called *full* congruence, is common. That concept is not needed in this paper.

The proof of Proposition 9.5, giving yet another characterisation of a congruence, breaks down for general closed term languages, namely where it uses that a term contains only finitely many variables. I therefore introduce the concept characterised by Proposition 9.5 under a different name.

Definition 10.21 Let $\mathscr{L}$ be a closed term language. An equivalence relation $\sim$ on $\mathrm{T}_{\mathscr{L}}$ is a *1-hole congruence* for $\mathscr{L}$ iff $E[\varsigma_1] \sim E[\varsigma_2]$ for any $E \in \mathbb{T}_{\mathscr{L}}$ and any $\varsigma_1, \varsigma_2 : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ with $\varsigma_1 \sim^1 \varsigma_2$.

Obviously, a congruence is also a 1-hole congruence. An *n-hole congruence* for any finite $n \in \mathbb{N}$ can be defined in the same vein, and the proof Proposition 9.5 shows that a 1-hole congruence $\sim$ is also an n-hole congruence, for any $n \in \mathbb{N}$. However, in the presence of operators with infinitely many arguments, a 1-hole congruence need not be a congruence. This will be illustrated by Example 11.14.

Definition 9.6 of closure under $\mathscr{T}(\mathscr{L})$ and congruence for $\mathscr{T}(\mathscr{L})$ lifts to general closed term languages virtually unchanged.

Definition 10.22 Let $\mathscr{L}, \mathscr{L}'$ be closed term languages, and $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ a translation from $\mathscr{L}$ into $\mathscr{L}'$. A subset $\mathbf{W}$ of $\mathrm{T}_{\mathscr{L}'}$ is *closed* under $\mathscr{T}(\mathscr{L})$ if $P \stackrel{\alpha}{=} Q \in \mathbf{W} \Rightarrow P \in \mathbf{W}$ and $\mathscr{T}(E)[\eta] \in \mathbf{W}$ for any expression $E \in \mathbb{T}_{\mathscr{L}}$ and substitution $\eta : \mathscr{X} \to \mathbf{W}$. An equivalence $\sim$ on a closed set $\mathbf{W}$ is a *congruence* for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ if $\mathscr{T}(E)[\eta_1] \sim \mathscr{T}(E)[\eta_2]$ for any $E \in \mathbb{T}_{\mathscr{L}}$ and any $\eta_1, \eta_2 : \mathscr{X} \to \mathbf{W}$ with $\eta_1 \sim \eta_2$.

Likewise, $\sim$ is a *1-hole congruence* for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ if we merely require $\eta_1 \sim^1 \eta_2$.

Proposition 9.7 extends to general closed term languages unchanged. So does Proposition 9.8, although it applies to fvr-translations only. Also Theorems 9.9 and 9.10 generalise unchanged, and they remain valid when substituting "1-hole congruence" for "congruence".

Theorem 10.23 Suppose $\mathscr{L}'$ is at least as expressive as $\mathscr{L}$ up to an equivalence or preorder $\precsim$, and let $\approx \supseteq \precsim$ be a [1-hole] congruence for $\mathscr{L}'$. Then $\approx$ is also a [1-hole] congruence for $\mathscr{L}$. □

Theorem 10.23* Let $\mathscr{T}$ be a translation of $\mathscr{L}$ into $\mathscr{L}'$ that is valid up to $\precsim$ and let $\approx \supseteq \precsim$ be a [1-hole] congruence for $\mathscr{T}(\mathscr{L})$ on a closed set $\mathbf{W}$. Then $\approx$ is also a [1-hole] congruence for $\mathscr{L}$. □

Theorem 10.24 Let $\mathscr{T}$ be a translation of $\mathscr{L}$ into $\mathscr{L}'$ that is correct w.r.t. a semantic translation $\mathbf{R}$ and let $\approx \supseteq \mathbf{R}$ be a [1-hole] congruence for $\mathscr{L}$. Then $\approx$ is also a [1-hole] congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{R}(\mathscr{L})$.□

The congruence reflection properties of Theorems 3.14 and 9.12 generalise to both congruences and 1-hole congruences, the latter unchanged.

Theorem 10.25 Let $\mathscr{T}$ be a compositional fvr-translation between closed term languages $\mathscr{L}$ and $\mathscr{L}'$. Then any equivalence relation $\sim \subseteq \mathbf{W} \times \mathbf{W}$ on a closed set $\mathbf{W} \subseteq \mathrm{T}_{\mathscr{L}'}$ that contains $\overset{\alpha}{=}$ and is a [1-hole] congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ can be extended to an equivalence $\sim_{\mathscr{T}}$ on $\mathrm{T}_{\mathscr{L}} \uplus \mathbf{W}$, such that $\mathscr{T}$ is valid up to $\sim_{\mathscr{T}}$, and $\sim_{\mathscr{T}}$ also is a [1-hole] congruence for $\mathscr{L}$.

Moreover, if $\mathscr{T}$ is valid up to $\approx$ and $\sim \subseteq \approx$ on $\mathbf{W}$, then $\sim_{\mathscr{T}} \subseteq \approx$.

Proof: Write $P \xrightarrow{\mathscr{T}} R$ for $P,R \in \mathrm{T}_{\mathscr{L}} \uplus \mathbf{W}$ if either $R = P \in \mathrm{T}_{\mathscr{L}}$ or $R = P \in \mathbf{W}$ or $P \in \mathrm{T}_{\mathscr{L}}$ and $R = \mathscr{T}(P) \in \mathbf{W}$. Define $\sim_{\mathscr{T}}$ on $\mathrm{T}_{\mathscr{L}} \uplus \mathbf{W}$ by $P \sim_{\mathscr{T}} Q$ iff $P \xrightarrow{\mathscr{T}} R \sim S \xleftarrow{\mathscr{T}} Q$ for some $R,S \in \mathbf{W}$. This relation is reflexive, symmetric and transitive, since $\sim$ is. Thus $\sim_{\mathscr{T}}$ is an equivalence relation. On $\mathbf{W}$ the relation $\sim_{\mathscr{T}}$ coincides with $\sim$. By construction, $\mathscr{T}(P) \sim_{\mathscr{T}} P$ for all $P \in \mathrm{T}_{\mathscr{L}}$, so by Proposition 10.19 $\mathscr{T}$ is valid up to $\sim_{\mathscr{T}}$. That $\sim_{\mathscr{T}}$ is a [1-hole] congruence for $\mathscr{L}$ follows by Theorem 10.23* (taking $\approx := \precsim := \sim_{\mathscr{T}}$).

The last statement of Theorem 10.25 follows directly from the definition of $\sim_{\mathscr{T}}$. □

Theorem 10.26 Let $\mathscr{T}$ be a translation between closed term languages $\mathscr{L}$ and $\mathscr{L}'$ that is correct w.r.t. a semantic translation $\mathbf{R}$. Then any equivalence relation $\sim \subseteq \mathbf{R}(\mathscr{L}) \times \mathbf{R}(\mathscr{L})$ that contains the restriction of $\equiv_{\mathbf{R}}$ to $\mathbf{R}(\mathscr{L})$ and is a [1-hole] congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{R}(\mathscr{L})$ can be extended to an equivalence $\sim_{\mathbf{R}}$ on $\mathrm{T}_{\mathscr{L}} \uplus \mathbf{R}(\mathscr{L})$, such that $\mathscr{T}$ is valid up to $\sim_{\mathbf{R}}$, and $\sim_{\mathbf{R}}$ also is a [1-hole] congruence for $\mathscr{L}$. □

Lemma 9.13 generalises unchanged, but only yielding a 1-hole congruence.

Lemma 10.27 If a translation $\mathscr{T}$ is correct w.r.t. a semantic translation $\mathbf{R}$, then $\equiv_{\mathbf{R}}$ is a 1-hole congruence for $\mathscr{L}$. □

For any equivalence relation $\approx$ on $\mathrm{T}_{\mathscr{L}}$ there exists a largest 1-hole congruence for $\mathscr{L}$ contained in $\approx$. I will denote this 1-hole congruence by $\approx^{1c}_{\mathscr{L}}$, and call it the *congruence closure* of $\approx$ w.r.t. $\mathscr{L}$. One has $P \approx^{1c}_{\mathscr{L}} Q$ for $P,Q \in \mathrm{T}_{\mathscr{L}}$ iff $E[\varsigma_1] \approx E[\varsigma_2]$ for all $\mathscr{L}$-expressions E and all substitutions $\varsigma_1,\varsigma_2 : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ with $\varsigma_1(X) = P$ and $\varsigma_2(X) = Q$ for some $X \in \mathscr{X}$ and $\varsigma_1(Y) = \varsigma_2(Y)$ for all variables $Y \neq X$. Example 11.25 will illustrate that in general there is no such result for congruences.

Given a translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$, I also speak of the *congruence closure* $\approx^{1c}_{\mathscr{T}(\mathscr{L}),\mathbf{W}}$ of $\approx$ for $\mathscr{T}(\mathscr{L})$ on a closed set $\mathbf{W}$. This is the largest 1-hole congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ that is contained in $\approx$. It can be characterised by $P \approx^{1c}_{\mathscr{T}(\mathscr{L}),\mathbf{W}} Q$ for $P,Q \in \mathbf{W}$ iff $F[\eta_1] \approx F[\eta_2]$ for all $\mathscr{L}'$-expressions F of the form $\mathscr{T}(E)$ and all substitutions $\eta_1,\eta_2 : \mathscr{X} \to \mathbf{W}$ with $\eta_1(X) = P$ and $\eta_2(X) = Q$ for some $X \in \mathscr{X}$ and $\eta_1(Y) = \eta_2(Y)$ for all variables $Y \neq X$.

Finally, Theorem 9.14 generalised unchanged, provided we consistently read "1-hole congruence" for "congruence".

Theorem 10.28 Let $\mathscr{T}$ be a translation between closed term languages $\mathscr{L}$ and $\mathscr{L}'$ that is valid up to an equivalence $\approx$. Then $\mathscr{T}$ is valid even up to an equivalence $\approx^{1c}_{\mathscr{T}}$, contained in $\approx$, such that

(1) the restriction of $\approx^{1c}_{\mathscr{T}}$ to $\mathrm{T}_{\mathscr{L}}$ is the largest 1-hole congruence for $\mathscr{L}$ contained in $\approx$,

and there exists a closed set $\mathbf{W}$ with $\mathrm{T}_{\mathscr{T}(\mathscr{L})} \subseteq \mathbf{W} \subseteq \mathrm{T}_{\mathscr{L}'}$ such that

(2) the restriction of $\approx^{1c}_{\mathscr{T}}$ to $\mathbf{W}$ is the largest 1-hole congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ that is contained in $\approx$, and

(3) each equivalence class of $\approx^{1c}_{\mathscr{T}}$ is the union of an equivalence class of $\approx^{1c}_{\mathscr{L}}$ and one of $\approx^{1c}_{\mathscr{T}(\mathscr{L}),\mathbf{W}}$. □

11 A theory of encodings and expressiveness for general languages

In this section I drop the restriction to closed term languages, and define my notion of a valid translation for the most general class of languages considered in this paper. Such a language is given as a pair $(\mathbb{T}_{\mathscr{L}}, [\![\ ]\!]_{\mathscr{L}})$ with the set $\mathbb{T}_{\mathscr{L}}$ of open terms as described in Sections 2 and 10, and a semantic mapping $[\![\ ]\!]_{\mathscr{L}}$ of type $\mathbb{T}_{\mathscr{L}} \to ((\mathscr{X} \to \mathbf{V}) \to \mathbf{V})$, for some set $\mathbf{V}$ called a *domain*.

For these languages the meaning $[\![E]\!]_{\mathscr{L}}$ of an $\mathscr{L}$-expression E is a function of type $(\mathscr{X} \to \mathbf{V}) \to \mathbf{V}$ for a given sets of *values* $\mathbf{V}$. It associates a value $[\![E]\!]_{\mathscr{L}}(\rho) \in \mathbf{V}$ to E that depends on the choice of a *valuation* $\rho : \mathscr{X} \to \mathbf{V}$. The valuation associates a value from $\mathbf{V}$ with each variable.

The semantic mapping $[\![\ ]\!]_{\mathscr{L}}$ is defined inductively, in the sense that $[\![X]\!]_{\mathscr{L}}(\rho)$ for a variable $X \in \mathscr{X}$ and a valuation ρ is simply $\rho(X) \in \mathbf{V}$, and for each term building construct $\mathscr{C}$, there must be an operation that extracts the meaning $[\![\mathscr{C}(\vec{E})]\!]_{\mathscr{L}}$ of a term $\mathscr{C}(\vec{E})$ from the meaning of its arguments $\vec{E}$. In particular, the interpretation of each n-ary operator f of $\mathscr{L}$ is an n-ary operation $f^{\mathbf{V}} : \mathbf{V}^n \to \mathbf{V}$ on $\mathbf{V}$, and we have

$$[\![f(E_1,\ldots,E_n)]\!]_{\mathscr{L}}(\rho) = f^{\mathbf{V}}([\![E_1]\!]_{\mathscr{L}}(\rho),\ldots,[\![E_n]\!]_{\mathscr{L}}(\rho)).$$

Moreover, $[\![E]\!]_{\mathscr{L}}(\rho)$ only depends on the restriction of ρ to the set $fv(E)$ of variables occurring free in E. If $P \in \mathrm{T}_{\mathscr{L}}$ and $\mathbf{V} \neq \emptyset$ then $[\![P]\!]_{\mathscr{L}}(\rho)$ is independent of the choice of $\rho : \mathscr{X} \to \mathbf{V}$, and hence written $[\![P]\!]_{\mathscr{L}}$.

11.1 Valid translations

Let $\mathscr{L}$ and $\mathscr{L}'$ be two languages of the type considered above, with semantic mappings

$$[\![\ ]\!]_{\mathscr{L}} : \mathbb{T}_{\mathscr{L}} \to ((\mathscr{X} \to \mathbf{V}) \to \mathbf{V}) \qquad \text{and} \qquad [\![\ ]\!]_{\mathscr{L}'} : \mathbb{T}_{\mathscr{L}'} \to ((\mathscr{X} \to \mathbf{V}') \to \mathbf{V}').$$

In order to compare these languages w.r.t. their expressive power I need a semantic equivalence or preorder $\rightsquigarrow$ that is defined on a unifying domain of interpretation $\mathbf{Z}$, with $\mathbf{V}, \mathbf{V}' \subseteq \mathbf{Z}$. Intuitively, $v' \rightsquigarrow v$ with $v \in \mathbf{V}$ and $v' \in \mathbf{V}'$ means that values v and v' are sufficiently alike for our purposes, so that one can accept a translation of an expression with meaning v into an expression with meaning v'.

The material of Section 10 prior to Section 10.6 pertains already to such general languages. Below follow the adaptations of Definitions 7.3, 7.4 and 7.5 for general languages.

Definition 11.1 Let $\mathbf{V}$ and $\mathbf{V}'$ be domains of values in which two languages $\mathscr{L}$ and $\mathscr{L}'$ are interpreted. A *semantic translation* from $\mathbf{V}$ into $\mathbf{V}'$ is a relation $\mathbf{R} \subseteq \mathbf{V}' \times \mathbf{V}$ such that $\forall v \in \mathbf{V}.\, \exists v' \in \mathbf{V}'.\, v'\ \mathbf{R}\ v$.

Intuitively, $v'\ \mathbf{R}\ v$ says that the $\mathscr{L}'$-value v' is a good translation or *counterpart* of the $\mathscr{L}$-value v. Definition 11.1 says that every value in $\mathbf{V}$ needs to have a counterpart in $\mathbf{V}'$—possibly multiple ones. For valuations $\eta : \mathscr{X} \to \mathbf{V}'$ and $\rho : \mathscr{X} \to \mathbf{V}$, I write $\eta\ \mathbf{R}\ \rho$ iff $\eta(X)\ \mathbf{R}\ \rho(X)$ for each $X \in \mathscr{X}$.

Definition 11.2 A translation $\mathscr{T}: \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ is *correct* w.r.t. a semantic translation $\mathbf{R}$ if $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta)\ \mathbf{R}\ [\![E]\!]_{\mathscr{L}}(\rho)$ for all expressions $E \in \mathbb{T}_{\mathscr{L}}$ and all valuations $\eta : \mathscr{X} \to \mathbf{V}'$ and $\rho : \mathscr{X} \to \mathbf{V}$ with $\eta\ \mathbf{R}\ \rho$.

Thus $\mathscr{T}$ is correct iff the meaning of the translation of an expression E is a counterpart of the meaning of E, no matter what values are filled in for the variables, provided that the value filled in for a given variable X occurring in the translation $\mathscr{T}(E)$ is a counterpart of the value filled in for X in E.

Definition 11.3 A translation $\mathscr{T}$ is *valid* up to $\backsim$ iff it is correct w.r.t. some semantic translation $\mathbf{R} \subseteq \backsim$. Language $\mathscr{L}'$ is at least as *expressive* as $\mathscr{L}$ up to $\backsim$ if a translation valid up to $\backsim$ from $\mathscr{L}$ into $\mathscr{L}'$ exists.

A closed term language is as language as above in which $\mathbf{V} = \mathrm{T}_{\mathscr{L}}$—so that a valuation is the same as a closed substitution—and $[\![E]\!]_{\mathscr{L}}(\rho)$ for $E \in \mathbb{T}_{\mathscr{L}}$ and $\rho : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ is defined to be $E[\rho] \in \mathrm{T}_{\mathscr{L}}$. For such languages Definitions 11.1–11.3 above agree with the definitions of Section 10.6.

11.2 α-conversion

The following relation $\stackrel{\alpha}{=}_{\mathscr{L}}$ is a semantic counterpart on the domain $\mathbf{V}$ on which a language $\mathscr{L}$ is interpreted, of the relation $\stackrel{\alpha}{=}$ on $\mathrm{T}_{\mathscr{L}}$.

Definition 11.4 Write $v \stackrel{\alpha}{=}_{\mathscr{L}} w$, with $v, w \in \mathbf{V}$, iff there are terms $E, F \in \mathbb{T}_{\mathscr{L}}$ with $E \stackrel{\alpha}{=} F$, and a valuation $\zeta : \mathscr{X} \to \mathbf{V}$ such that $[\![E]\!]_{\mathscr{L}}(\zeta) = v$ and $[\![F]\!]_{\mathscr{L}}(\zeta) = w$.

This relation is reflexive and symmetric by definition, but not necessarily transitive. For this reason, I work mainly with its transitive closure $\stackrel{\alpha}{=}^*_{\mathscr{L}}$.

Lemma 11.5 Let $E \in \mathbb{T}_{\mathscr{L}}$ and $\rho_1, \rho_2 : \mathscr{X} \to \mathbf{V}$. If $\rho_1 \stackrel{\alpha}{=}_{\mathscr{L}} \rho_2$ then $[\![E]\!]_{\mathscr{L}}(\rho_1) \stackrel{\alpha}{=}^*_{\mathscr{L}} [\![E]\!]_{\mathscr{L}}(\rho_2)$.

Proof: Suppose $\rho_1 \stackrel{\alpha}{=}_{\mathscr{L}} \rho_2$. Then, for each $X \in \mathscr{X}$, $\rho_1(X) \stackrel{\alpha}{=}_{\mathscr{L}} \rho_2(X)$, so there are terms $E_X, F_X \in \mathbb{T}_{\mathscr{L}}$ with $E_X \stackrel{\alpha}{=} F_X$ and a valuation $\zeta_X : \mathscr{X} \to \mathbf{V}$ such that $[\![E_X]\!]_{\mathscr{L}}(\zeta_X) = \rho_1(X)$ and $[\![F_X]\!]_{\mathscr{L}}(\zeta_X) = \rho_2(X)$. By renaming of variables one can assure that $\mathit{fv}(E_X) \cap \mathit{fv}(E_Y) = \emptyset$ for any different $X, Y \in \mathit{fv}(E)$. Here I assume that the set $\mathscr{X}$ of variables is sufficiently large. Note that $\mathit{fv}(F_X) = \mathit{fv}(E_X)$ for all $X \in \mathit{fv}(E)$. Let $\zeta : \mathscr{X} \to \mathbf{V}$ be a valuation satisfying $\zeta(Z) = \zeta_X(Z)$ for any $Z \in \mathit{fv}(E_X)$ with $X \in \mathit{fv}(E)$. Define the substitutions $\sigma, \xi : \mathrm{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}}$ by $\sigma(X) = E_X$ and $\xi(X) = F_X$ for all $X \in \mathit{fv}(E)$ (and for instance $\sigma(Y) = \xi(Y) = Y$ for $Y \notin \mathit{fv}(E)$). Then $[\![\sigma(X)]\!]_{\mathscr{L}}(\zeta) = \rho_1(X)$ and $[\![\xi(X)]\!]_{\mathscr{L}}(\zeta) = \rho_2(X)$ for all $X \in \mathit{fv}(E)$. Hence, using (2), $[\![E]\!]_{\mathscr{L}}(\rho_1) = [\![E]\!]_{\mathscr{L}}([\![\sigma]\!]_{\mathscr{L}}(\zeta)) \stackrel{\alpha}{=}_{\mathscr{L}} [\![E[\sigma]]\!]_{\mathscr{L}}(\zeta)$ and $[\![E]\!]_{\mathscr{L}}(\rho_2) = [\![E]\!]_{\mathscr{L}}([\![\xi]\!]_{\mathscr{L}}(\zeta)) \stackrel{\alpha}{=}_{\mathscr{L}} [\![E[\xi]]\!]_{\mathscr{L}}(\zeta)$. As $\sigma(X) \stackrel{\alpha}{=} \xi(X)$ for all $X \in \mathscr{X}$ one has $E[\sigma] \stackrel{\alpha}{=} E[\xi]$ by Observation 10.4, and thus

$$[\![E]\!]_{\mathscr{L}}(\rho_1) \stackrel{\alpha}{=}_{\mathscr{L}} [\![E[\sigma]]\!]_{\mathscr{L}}(\zeta) \stackrel{\alpha}{=}_{\mathscr{L}} [\![E[\xi]]\!]_{\mathscr{L}}(\zeta) \stackrel{\alpha}{=}_{\mathscr{L}} [\![E]\!]_{\mathscr{L}}(\rho_2). \qquad \square$$

Given a relation $\mathbf{R} \subseteq \mathbf{V}' \times \mathbf{V}$, define $\mathbf{R}^\alpha$ by $w \; \mathbf{R}^\alpha \; v$ iff $\exists w', v'.\ w \stackrel{\alpha}{=}^*_{\mathscr{L}'} w' \; \mathbf{R} \; v' \stackrel{\alpha}{=}^*_{\mathscr{L}} v$. The above lemma enables me to prove the following generalisation of Lemma 10.16.

Lemma 11.6 If a translation $\mathscr{T}$ between languages $\mathscr{L}$ and $\mathscr{L}'$ is correct w.r.t. a semantic translation $\mathbf{R}$, then it is also correct w.r.t. $\mathbf{R}^\alpha$.

Proof: Let $\mathbf{R} \subseteq \mathbf{V}' \times \mathbf{V}$ be a semantic translation, and $\mathscr{T}$ a translation that is correct w.r.t. $\mathbf{R}$. Let $E \in \mathbb{T}_{\mathscr{L}}$, $\eta : \mathscr{X} \to \mathbf{V}'$ and $\rho : \mathscr{X} \to \mathbf{V}$, with $\eta \; \mathbf{R}^\alpha \; \rho$. Then there must be valuations $\eta' : \mathscr{X} \to \mathbf{V}'$ and $\rho' : \mathscr{X} \to \mathbf{V}$ with $\eta \stackrel{\alpha}{=}^*_{\mathscr{L}'} \eta' \; \mathbf{R} \; \rho' \stackrel{\alpha}{=}^*_{\mathscr{L}} \rho$. Now $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta') \; \mathbf{R} \; [\![E]\!]_{\mathscr{L}}(\rho')$, as $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$. By Lemma 11.5 $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \stackrel{\alpha}{=}^*_{\mathscr{L}'} [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta') \; \mathbf{R} \; [\![E]\!]_{\mathscr{L}}(\rho') \stackrel{\alpha}{=}^*_{\mathscr{L}} [\![E]\!]_{\mathscr{L}}(\rho)$, so $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \; \mathbf{R}^\alpha \; [\![E]\!]_{\mathscr{L}}(\rho)$. It follows that $\mathscr{T}$ is correct w.r.t. $\mathbf{R}^\alpha$. $\square$

11.3 The denotation of a substitution

To obtain a generalisation of Theorem 10.17 I need the concept of the *denotation* of a substitution as a transformation of valuations. 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 $\rho : \mathscr{X} \to \mathbf{V}$. 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 for each expression $E \in \mathbb{T}_{\mathscr{L}}$, substitution $\sigma : \mathscr{X} \rightharpoonup \mathbb{T}_{\mathscr{L}}$ and valuation $\rho : \mathscr{X} \to \mathbf{V}$ there exists a term F with $E \stackrel{\alpha}{=} F$ such that $[\![E[\sigma]]\!]_{\mathscr{L}}(\rho) = [\![F[\sigma]]\!]_{\mathscr{L}}(\rho) = [\![F]\!]_{\mathscr{L}}([\![\sigma]\!]_{\mathscr{L}}(\rho))$, and hence

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

In case E is $f(X_1,\ldots,X_n)$ this amounts to $[\![f(E_1,\ldots,E_n)]\!]_{\mathscr{L}}(\rho) = f^{\mathbf{V}}([\![E_1]\!]_{\mathscr{L}}(\rho),\ldots,[\![E_n]\!]_{\mathscr{L}}(\rho))$, but the above is more general and anticipates language constructs other than functions, such as recursion.

11.4 Valid translations can be made compositional

This section shows how Theorem 10.17 generalises to general languages.

Theorem 11.7 Let $\mathscr{L}$ and $\mathscr{L}'$ be languages. Suppose $\backsim \supseteq \stackrel{\alpha}{=}_{\mathscr{L}}$ and $\backsim \supseteq \stackrel{\alpha}{=}_{\mathscr{L}'}$. If any valid translation from $\mathscr{L}$ into $\mathscr{L}'$ up to $\backsim$ exists, then there exists a compositional translation that is valid up to $\backsim$.

Proof: Let $\mathscr{T}_1$ be a valid translation from $\mathscr{L}$ into $\mathscr{L}'$ up to $\backsim$. It must be correct w.r.t. a semantic translation $\mathbf{R} \subseteq \backsim$. By Lemma 11.6 it is also correct w.r.t. $\mathbf{R}^\alpha \subseteq \backsim$. Define $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ as in the proof of Theorem 10.17. That $\mathscr{T}$ is compositional follows exactly as in that proof.

It remains to be shown that $\mathscr{T}$ is correct w.r.t. $\mathbf{R}^\alpha$, i.e. that $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta)\ \mathbf{R}^\alpha\ [\![E]\!]_{\mathscr{L}}(\rho)$ for all expressions $E \in \mathbb{T}_{\mathscr{L}}$ and all valuations $\eta : \mathscr{X} \to \mathbf{V}'$ and $\rho : \mathscr{X} \to \mathbf{V}$ with $\eta\ \mathbf{R}^\alpha\ \rho$. Let η and ρ be such valuations. I proceed with structural induction on E. The base case $E \in \mathscr{X}$ is trivial, so let $\mathit{decomp}(E) = (H,\xi)$. By Proposition 10.12(ii) and the induction hypothesis, $[\![\mathscr{T}(\xi(X))]\!]_{\mathscr{L}'}(\eta)\ \mathbf{R}^\alpha\ [\![\xi(X)]\!]_{\mathscr{L}}(\rho)$ for all $X \in \mathscr{X}$. The valuation $[\![\xi]\!]_{\mathscr{L}}(\eta)$ is defined such that $[\![\xi]\!]_{\mathscr{L}}(\eta)(X) = [\![\xi(X)]\!]_{\mathscr{L}}(\eta)$ for each $X \in \mathscr{X}$. Likewise, $[\![\mathscr{T}\circ\xi]\!]_{\mathscr{L}'}(\eta)(X) = [\![\mathscr{T}(\xi(X))]\!]_{\mathscr{L}'}(\eta)$ for each $X \in \mathscr{X}$. Hence $[\![\mathscr{T}\circ\xi]\!]_{\mathscr{L}'}(\eta)\ \mathbf{R}^\alpha\ [\![\xi]\!]_{\mathscr{L}}(\rho)$.(*)

$$\begin{array}{lll}
\text{Now } [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) = & [\![\mathscr{T}_1(H)[\mathscr{T}\circ\xi]]\!]_{\mathscr{L}'}(\eta) & \text{by definition of } \mathscr{T} \\
\stackrel{\alpha}{=}_{\mathscr{L}'} & [\![\mathscr{T}_1(H)]\!]_{\mathscr{L}'}([\![\mathscr{T}\circ\xi]\!]_{\mathscr{L}'}(\eta)) & \text{by (2)} \\
\mathbf{R}^\alpha & [\![H]\!]_{\mathscr{L}}([\![\xi]\!]_{\mathscr{L}}(\rho)) & \text{as } \mathscr{T}_1 \text{ is correct w.r.t. } \mathbf{R}^\alpha\text{, using (*) above} \\
\stackrel{\alpha}{=}_{\mathscr{L}} & [\![H[\xi]]\!]_{\mathscr{L}}(\rho) & \text{by (2)} \\
\stackrel{\alpha}{=}_{\mathscr{L}} & [\![E]\!]_{\mathscr{L}}(\rho) & \text{since } E \stackrel{\alpha}{=} H[\xi] \text{ by Proposition 10.12(1).}
\end{array}$$ □

11.5 Free-variable respecting translations

Next I show how Proposition 10.18 generalises. I do not generalise Proposition 10.19.

Proposition 11.8 Let $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be a translation from $\mathscr{L}$ into $\mathscr{L}'$ that is correct w.r.t. a semantic translation $\mathbf{R} \subseteq \mathbf{V} \times \mathbf{V}'$. Assume that there exists a pair $(v',v) \in \mathbf{R}$ and a closed term $Q \in \mathrm{T}_{\mathscr{L}'}$ with $[\![Q]\!]_{\mathscr{L}'} = v'$. Then there exists a compositional fvr-translation that is correct w.r.t. $\mathbf{R}$.

Proof: Take a pair $(v',v) \in \mathbf{R}$ and a closed term $Q \in \mathrm{T}_{\mathscr{L}'}$ with $[\![Q]\!]_{\mathscr{L}'} = v'$. For $E \in \mathbb{T}_{\mathscr{L}}$ let $\zeta_E : \mathscr{X} \to \mathbb{T}_{\mathscr{L}'}$ be the substitution given by $\zeta_E(X) = Q$ for all $X \notin \mathit{fv}(E)$ and $\zeta_E(X) = X$ for all $X \in \mathit{fv}(E)$. Define the translation $\mathscr{T}_1 : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ by $\mathscr{T}_1(E) = \mathscr{T}_2(E)[\zeta_E]$ for all $E \in \mathbb{T}_{\mathscr{L}}$. By construction, $\mathscr{T}_1$ is an fvr-translation. To show that it is correct w.r.t. $\mathbf{R}$, take $E \in \mathbb{T}_{\mathscr{L}}$ and select valuations $\eta : \mathscr{X} \to \mathbf{V}'$ and $\rho : \mathscr{X} \to \mathbf{V}$ with $\eta\ \mathbf{R}\ \rho$. Let $\eta' : \mathscr{X} \to \mathbf{V}'$ and $\rho' : \mathscr{X} \to \mathbf{V}$ be the valuations given by $\eta'(X) = \eta(X)$ and $\rho'(X) = \rho(X)$ for $X \in \mathit{fv}(E)$, while $\eta'(X) = v'$ and $\rho'(X) = v$ for $X \notin \mathit{fv}(E)$. Then $\eta'\ \mathbf{R}\ \rho'$. Thus $[\![\mathscr{T}_1(E)]\!]_{\mathscr{L}'}(\eta) = [\![\mathscr{T}_2(E)[\zeta_E]]\!]_{\mathscr{L}'}(\eta) = [\![\mathscr{T}_2(E)]\!]_{\mathscr{L}'}([\![\zeta_E]\!]_{\mathscr{L}'}(\eta)) = [\![\mathscr{T}_2(E)]\!]_{\mathscr{L}'}(\eta')\ \mathbf{R}\ [\![E]\!]_{\mathscr{L}}(\rho') = [\![E]\!]_{\mathscr{L}}(\rho)$. Here the second step with $\stackrel{\alpha}{=}_{\mathscr{L}'}$ for $=$ follows by (2), but one even gets $=$ because ζ_E is such that no variable is at risk of being captured in a substitution $F[\zeta_E]$.

Hence there exists an fvr-translation $\mathscr{T}_1$ from $\mathscr{L}$ into $\mathscr{L}'$ that is valid up to $\backsim$. Now in the proof of Theorem 11.7, by induction it follows that if $\mathscr{T}_1$ is fvr, then so is $\mathscr{T}$. □

11.6 A hierarchy of expressiveness preorders

The results of Sections 3.3 and 9.1 generalise straightforwardly to general languages.

Observation 11.9 Let $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be a translation from $\mathscr{L}$ into $\mathscr{L}'$, and let $\preccurlyeq$ be finer than $\approx$. If $\mathscr{T}$ is valid up to $\preccurlyeq$, then it is also valid up to $\approx$.

Observation 11.10 The identity is a valid translation up to any preorder from any language into itself.

Theorem 11.11 If valid translations up to $\preccurlyeq$ exists from $\mathscr{L}$ into $\mathscr{L}'$ and from $\mathscr{L}'$ into $\mathscr{L}'$, then there is a valid translation up to $\preccurlyeq$ from $\mathscr{L}$ into $\mathscr{L}''$.

Proof: Let $\mathscr{T}_1 : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ be a valid translation up to $\preccurlyeq$ from $\mathscr{L}$ to $\mathscr{L}'$, and let $\mathscr{T}_2 : \mathbb{T}_{\mathscr{L}'} \to \mathbb{T}_{\mathscr{L}''}$ be one from $\mathscr{L}'$ to $\mathscr{L}''$. I will show that the translation $\mathscr{T} := \mathscr{T}_2 \circ \mathscr{T}_1 : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}''}$ from $\mathscr{L}$ into $\mathscr{L}''$, given by $\mathscr{T}(E) = \mathscr{T}_2 \circ \mathscr{T}_1(E) = \mathscr{T}_2(\mathscr{T}_1(E))$, is valid up to $\preccurlyeq$.

For i=1,2 let $\mathscr{T}_i$ be correct w.r.t. the semantic translation $\mathbf{R}_i \subseteq \preccurlyeq$. Here $\mathbf{R}_1 \subseteq \mathbf{V}' \times \mathbf{V}$ and $\mathbf{R}_2 \subseteq \mathbf{V}'' \times \mathbf{V}'$, where $\mathbf{V}$, $\mathbf{V}'$ and $\mathbf{V}''$ are the semantic domains in which the languages $\mathscr{L}$, $\mathscr{L}'$ and $\mathscr{L}''$ are interpreted. Let $\mathbf{R} := \mathbf{R}_2 \circ \mathbf{R}_1 \subseteq \mathbf{V}'' \times \mathbf{V}$ be the relation given by $v'' \,\mathbf{R}\, v$ iff $\exists v' \in \mathrm{T}_{\mathscr{L}'}.\ v'' \,\mathbf{R}_2\, v' \wedge v' \,\mathbf{R}_1\, v$— it is again a semantic translation, and satisfies $\mathbf{R}_2 \circ \mathbf{R}_1 \subseteq \preccurlyeq$, using the transitivity of $\preccurlyeq$.

Now let $E \in \mathbb{T}_{\mathscr{L}}$, $\rho : \mathscr{X} \to \mathbf{V}$ and $\theta : \mathscr{X} \to \mathbf{V}''$, with $\theta \,\mathbf{R}\, \rho$. Then there is a valuation $\eta : \mathscr{X} \to \mathbf{V}'$ with $\theta \,\mathbf{R}_2\, \eta \,\mathbf{R}_1\, \rho$. Hence, applying Definition 11.2, $[\![\mathscr{T}_2(\mathscr{T}_1(E))]\!]_{\mathscr{L}''}(\theta) \,\mathbf{R}_2\, [\![\mathscr{T}_1(E)]\!]_{\mathscr{L}'}(\eta) \,\mathbf{R}_1\, [\![E]\!]_{\mathscr{L}}(\rho)$, so $[\![\mathscr{T}(E)]\!]_{\mathscr{L}''}(\theta) \,\mathbf{R}\, [\![E]\!]_{\mathscr{L}}(\rho)$ and $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$. □

Theorem 11.11 and Observation 11.10 show that the relation "being at least as expressive as up to $\preccurlyeq$" is a preorder on languages.

11.7 The concept of a congruence

Definitions 10.20 and 10.21 of a [1-hole] congruence generalise to arbitrary languages as follows.

Definition 11.12 An equivalence relation $\sim$ is a *congruence* for a language $\mathscr{L}$ interpreted in a semantic domain $\mathbf{V}$ if $[\![E]\!]_{\mathscr{L}}(\rho_1) \sim [\![E]\!]_{\mathscr{L}}(\rho_2)$ for any $\mathscr{L}$-expression E and any valuations $\rho_1, \rho_2 : \mathscr{X} \to \mathbf{V}$ with $\rho_1 \sim \rho_2$.[11]

Lemma 11.5 says that the transitive closure $\stackrel{\alpha}{=}{}^*_{\mathscr{L}}$ of $\stackrel{\alpha}{=}_{\mathscr{L}}$ is a congruence.

Definition 11.13 An equivalence relation $\sim$ is a *1-hole congruence* for a language $\mathscr{L}$ interpreted in a semantic domain $\mathbf{V}$ if $[\![E]\!]_{\mathscr{L}}(\rho_1) \sim [\![E]\!]_{\mathscr{L}}(\rho_2)$ for any $\mathscr{L}$-expression E and any valuations $\rho_1, \rho_2 : \mathscr{X} \to \mathbf{V}$ with $\rho_1 \sim^1 \rho_2$. Here ρ_1, ρ_2 are $\sim^1$*-equivalent*, $\rho_1 \sim^1 \rho_2$, if $\rho_1(X) \sim \rho_2(X)$ for some $X \in \mathscr{X}$ and $\rho_1(Y) = \rho_2(Y)$ for all variables $Y \neq X$.

An *n-hole congruence* for any finite $n \in \mathbb{N}$ can be defined in the same vein, and it is well known and easy to check that a 1-hole congruence $\sim$ is also an n-hole congruence, for any $n \in \mathbb{N}$. However, in the presence of operators with infinitely many arguments, a 1-hole congruence need not be a congruence.

Example 11.14 Let $\mathbf{V}$ be $(\mathbb{N} \times \mathbb{N}) \cup \{\infty\}$, with the well-order $\leq$ on $\mathbf{V}$ inherited lexicographically from the default order on $\mathbb{N}$, and ∞ the largest element. So $(n,m) \leq (n',m')$ iff $n \leq n' \vee (n = m \wedge m \leq m')$. Consider the language $\mathscr{L}$ with constants 0, 1 and (1), interpreted in $\mathbf{V}$ as $(0,0)$, $(1,0)$ and $(0,1)$, respectively, the binary operator $+$, interpreted by $(n_1,m_1) +^{\mathbf{V}} (n_2,m_2) = (n_1{+}n_2, m_1{+}m_2)$ and $\infty + E = E + \infty = \infty$, and the construct $\sup(E_i)_{i \in I}$ that takes any number of arguments (dependent on the size of the index set

[11] This is called a *lean* congruence in [43]; in the presence of recursion, stricter congruence requirements are common. Those are not needed in this paper.

I). The interpretation of sup in $\mathbf{V}$ is to take the supremum of its arguments w.r.t. the well-order $\leq$. In case sup is given finitely many arguments, it simply returns the largest. However $\sup((n,i))_{i\in\mathbb{N}} = (n{+}1,0)$.

Now let the equivalence relation $\sim$ on $\mathbf{V}$ be defined by $(n,m) \sim (n',m')$ iff $n = n'$, leaving ∞ in an equivalence class of its own. This relation is a 1-hole congruence on $\mathscr{L}$. Hence, it is also a 2-hole congruence, so one has

$$\big((n_1,m_1) \sim (n_1',m_1') \wedge (n_2,m_2) \sim (n_2',m_2')\big) \Rightarrow (n_1,m_1)+(n_2,m_2) \sim (n_1',m_1')+(n_2',m_2').$$

Yet it fails to be a congruence: $(n,i) \sim (n,0)$ for all $i \in \mathbb{N}$, but

$$(n{+}1,0) = \sup((n,i))_{i\in\mathbb{N}} \not\sim \sup((n,0))_{i\in\mathbb{N}} = (n,0).$$

The following generalises Definition 10.22 and Propositions 9.7 and 9.8.

Definition 11.15 Let $\mathscr{T}$ be a translation from $\mathscr{L}$ into $\mathscr{L}'$. A subset $\mathbf{W}$ of $\mathbf{V}'$ is *closed* under $\mathscr{T}(\mathscr{L})$ if $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \in \mathbf{W}$ for any expression $E \in \mathbb{T}_{\mathscr{L}}$ and valuation $\eta : \mathscr{X} \to \mathbf{W}$. An equivalence $\sim$ on $\mathbf{W}$ is a *congruence* (respectively *1-hole congruence*) for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ if $\mathbf{W}$ is closed under $\mathscr{T}(\mathscr{L})$ and for any $E \in \mathbb{T}_{\mathscr{L}}$ and $\eta_1,\eta_2 : \mathscr{X} \to \mathbf{W}$ with $\eta_1 \sim \eta_2$ (resp. $\eta_1 \sim^1 \eta_2$) one has $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta_1) \sim [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta_2)$.

A default choice of $\mathbf{W}$ is $\mathbf{R}(\mathbf{V}) := \{v' \in \mathbf{V}' \mid \exists v \in \mathbf{V}.\ v'\ \mathbf{R}\ v\}$.

Proposition 11.16 Let $\mathscr{T}$ be a translation from $\mathscr{L}$ into $\mathscr{L}'$ that is correct w.r.t. a semantic translation $\mathbf{R} \subseteq \mathbf{V}' \times \mathbf{V}$. Then $\mathbf{R}(\mathbf{V})$ is closed under $\mathscr{T}(\mathscr{L})$.

Proof: Let $E \in \mathbb{T}_{\mathscr{L}}$ and $\eta : \mathscr{X} \to \mathbf{R}(\mathbf{V})$. Take $\rho : \mathscr{X} \to \mathbf{V}$ with $\eta\ \mathbf{R}\ \rho$. Then $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta)\ \mathbf{R}\ [\![E]\!]_{\mathscr{L}}(\rho)$ by Definition 11.2. Since $[\![E]\!]_{\mathscr{L}}(\rho) \in \mathbf{V}$ one has $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \in \mathbf{R}(\mathbf{V})$. □

I now show that for each translation $\mathscr{T}$, valid up to $\backsim$, there exists a smallest semantic relation $\mathbf{R} \subseteq \backsim$ for which $\mathscr{T}$ is correct. The corresponding set $\mathbf{R}(\mathbf{V})$ is the smallest set that is closed under $\mathscr{T}(\mathscr{L})$.

Proposition 11.17 Let $\mathscr{T}$ be a fvr-translation from $\mathscr{L}$ into $\mathscr{L}'$ that is correct w.r.t. a semantic translation $\mathbf{R}' \subseteq \mathbf{V}' \times \mathbf{V}$ and let $\mathbf{W} \subseteq \mathbf{V}'$ be closed under $\mathscr{T}(\mathscr{L})$. Then there $\mathscr{T}$ is correct w.r.t. a semantic translation $\mathbf{R} \subseteq \mathbf{R}'$ for which $\mathbf{R}(\mathbf{V}) \subseteq \mathbf{W}$.

Proof: For each ordinal λ define the relation $\mathbf{R}_\lambda \subseteq \mathbf{V}' \times \mathbf{V}$ as follows:

- $[\![P']\!]'_{\mathscr{L}}\ \mathbf{R}_0\ [\![P]\!]_{\mathscr{L}}$ iff $P \in \mathrm{T}_{\mathscr{L}}$ and $P' = \mathscr{T}(P) \in \mathrm{T}_{\mathscr{L}'}$,
- $v'\,\mathbf{R}_{\lambda+1}\,v$ if $v = [\![E]\!]_{\mathscr{L}}(\rho)$ and $v' = [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta)$ for some $E \in \mathbb{T}_{\mathscr{L}}$ and valuations η,ρ with $\eta\,\mathbf{R}_\lambda\,\rho$,
- $\mathbf{R}_\lambda = \bigcup_{\kappa<\lambda} \mathbf{R}_\kappa$ if λ is a limit ordinal.

A straightforward induction yields that this is an increasing sequence: $\kappa < \lambda \Rightarrow \mathbf{R}_\kappa \subseteq \mathbf{R}_\lambda$. Thus, it must have a limit, which I call $\mathbf{R}$. Clearly, $\mathbf{R}$ is a semantic translation, and $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$. Another straightforward induction yields $\mathbf{R} \subseteq \mathbf{R}'$. A third induction yields $\mathbf{R}(\mathbf{V}) \subseteq \mathbf{W}$. □

11.8 Congruence transfer properties

Theorems 10.23 and 10.24 remain valid in the same form.

Theorem 11.18 Suppose $\mathscr{L}'$ is at least as expressive as $\mathscr{L}$ up to an equivalence or preorder $\backsim$, and let $\approx \supseteq \backsim$ be a [1-hole] congruence for $\mathscr{L}'$. Then $\approx$ is also a [1-hole] congruence for $\mathscr{L}$. □

Proof: By assumption, there exists a valid translation $\mathscr{T}$ up to $\backsim$ from $\mathscr{L}$ into $\mathscr{L}'$. Using Definition 11.3, it must be correct w.r.t. some semantic translation $\mathbf{R} \subseteq \backsim \subseteq \approx$. Let $E \in \mathbb{T}_{\mathscr{L}}$ and $\rho_1,\rho_2 : \mathscr{X} \to \mathbf{V}$ with $\rho_1 \approx \rho_2$. As $\mathbf{R}$ is a semantic translation, there must be valuations $\eta_1,\eta_2 : \mathscr{X} \to \mathbf{V}'$ with $\eta_1\ \mathbf{R}\ \rho_1$ and $\eta_2\ \mathbf{R}\ \rho_2$. Thus $\eta_1 \approx \rho_1 \approx \rho_2 \approx \eta_2$, so by Definition 11.2, $[\![E]\!]_{\mathscr{L}}(\rho_1)\ \mathbf{R}^{-1}\ [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta_1) \approx [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta_2)\ \mathbf{R}\ [\![E]\!]_{\mathscr{L}}(\rho_2)$, implying $[\![E]\!]_{\mathscr{L}}(\rho_1) \approx [\![E]\!]_{\mathscr{L}}(\rho_2)$.

The adaptation of this proof to the 1-hole case is straightforward and left to the reader. □

Based on its proof, Theorem 11.18 can be sharpened by merely assuming that $\approx$ is a [1-hole] congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{R}(\mathbf{V})$, and in view of Proposition 11.17 we may take $\mathbf{R}(\mathbf{V}) \subseteq \mathbf{W}$ for a given closed set $\mathbf{W}$.

Theorem 11.18* Let $\mathscr{T}$ be a translation of $\mathscr{L}$ into $\mathscr{L}'$ that is valid up to $\mathrel{\reflectbox{$\rightsquigarrow$}}$ and let $\approx \supseteq \mathrel{\reflectbox{$\rightsquigarrow$}}$ be a [1-hole] congruence for $\mathscr{T}(\mathscr{L})$ on a closed set $\mathbf{W}$. Then $\approx$ is also a [1-hole] congruence for $\mathscr{L}$. □

Theorem 11.19 Let $\mathscr{T}$ be a translation of $\mathscr{L}$ into $\mathscr{L}'$ that is correct w.r.t. a semantic translation $\mathbf{R}$ and let $\approx \supseteq \mathbf{R}$ be a [1-hole] congruence for $\mathscr{L}$. Then $\approx$ is also a [1-hole] congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{R}(\mathbf{V})$.□

Proof: Let $E \in \mathbb{T}_{\mathscr{L}}$ and $\eta_1, \eta_2 : \mathscr{X} \to \mathbf{R}(\mathbf{V})$ with $\eta_1 \approx \eta_2$. By the definition of $\mathbf{R}(\mathbf{V})$ there exists valuations $\rho_1, \rho_2 : \mathscr{X} \to \mathbf{V}$ with $\eta_1 \mathbf{R} \rho_1$ and $\eta_2 \mathbf{R} \rho_2$. Thus $\rho_1 \approx \eta_1 \approx \eta_2 \approx \rho_2$, so by Definitions 11.2 and 11.12, $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta_1) \mathbf{R} [\![E]\!]_{\mathscr{L}}(\rho_1) \approx [\![E]\!]_{\mathscr{L}}(\rho_2) \mathbf{R}^{-1} [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta_2)$.

Again the 1-hole case is straightforward and left to the reader. □

11.9 Congruence reflection

The next theorem generalises Theorem 10.25; the target language may now be arbitrary, although the source is still required to be a closed term language. I also show that the restriction to fvr-translations can be dropped.

Theorem 11.20 Let $\mathscr{T}$ be a compositional translation, valid up to an equivalence $\approx$, from a closed term language $\mathscr{L}$ into a language $\mathscr{L}'$. Then any [1-hole] congruence $\sim \subseteq \mathbf{W} \times \mathbf{W}$ for $\mathscr{T}(\mathscr{L})$ on a closed set $\mathbf{W} \subseteq \mathbf{V}'$ with $\stackrel{\alpha}{=}_{\mathscr{L}'} \subseteq \sim \subseteq \approx$ can be extended to an equivalence $\sim_{\mathscr{T}} \subseteq \approx$ on $\mathrm{T}_{\mathscr{L}} \uplus \mathbf{W}$, such that $\mathscr{T}$ is valid up to $\sim_{\mathscr{T}}$, and $\sim_{\mathscr{T}}$ also is a [1-hole] congruence for $\mathscr{L}$.

Proof: If $\mathrm{T}_{\mathscr{L}} = \emptyset$, the statement is trivial. Otherwise, pick a valuation $\zeta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$. By assumption, $\mathscr{T}$ is correct w.r.t. a semantic translation $\mathbf{R} \subseteq \approx$. In view of Proposition 11.17 I assume that $\mathbf{R}(\mathrm{T}_{\mathscr{L}}) \subseteq \mathbf{W}$. Now pick a valuation $\theta : \mathscr{X} \to \mathbf{W}$ with $\theta \mathbf{R} \zeta$. For $v, w \in \mathrm{T}_{\mathscr{L}} \uplus \mathbf{W}$ write $v \stackrel{\mathscr{T}}{\to} w$ if either $w = v \in \mathrm{T}_{\mathscr{L}}$ or $w = v \in \mathbf{W}$ or $v \in \mathrm{T}_{\mathscr{L}}$ and $w = [\![\mathscr{T}(v)]\!]_{\mathscr{L}'}(\theta) \in \mathbf{W}$. Since $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$, one has $[\![\mathscr{T}(v)]\!]_{\mathscr{L}'}(\theta) \mathbf{R} [\![v]\!]_{\mathscr{L}}(\zeta) = [\![v]\!]_{\mathscr{L}} = v$ for all $v \in \mathrm{T}_{\mathscr{L}}$. Hence $v \stackrel{\mathscr{T}}{\to} w$ implies $v \approx w$. Define $\sim_{\mathscr{T}}$ on $\mathrm{T}_{\mathscr{L}} \uplus \mathbf{W}$ by $v \sim_{\mathscr{T}} v'$ iff

$$v \stackrel{\mathscr{T}}{\to} w \sim w' \stackrel{\mathscr{T}}{\leftarrow} v'$$

for some $w, w' \in \mathbf{W}$. So $\sim_{\mathscr{T}} \subseteq \approx$. This relation is reflexive, symmetric and transitive, since $\sim$ is. Thus $\sim_{\mathscr{T}}$ is an equivalence relation. On $\mathbf{W}$ the relation $\sim_{\mathscr{T}}$ coincides with $\sim$.

Let $\mathbf{R}_{\mathscr{T}} := \{(w, v) \mid v \in \mathrm{T}_{\mathscr{L}} \wedge w \stackrel{\alpha}{=}^*_{\mathscr{L}'} [\![\mathscr{T}(v)]\!]_{\mathscr{L}'}(\theta)\}$. It is a semantic translation with $\mathbf{R}_{\mathscr{T}} \subseteq \sim_{\mathscr{T}}$. To show validity of $\mathscr{T}$ up to $\sim_{\mathscr{T}}$ it suffices to establish that $\mathscr{T}$ is correct w.r.t. $\mathbf{R}_{\mathscr{T}}$. So 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 $\eta \mathbf{R}_{\mathscr{T}} \rho$. I have to show that $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \mathbf{R}_{\mathscr{T}} [\![E]\!]_{\mathscr{L}}(\rho)$. Since $\mathscr{L}$ is a closed-term language, ρ also is a closed substitution, and $[\![E]\!]_{\mathscr{L}}(\rho) := E[\rho]$. Let $\mathscr{T} \circ \rho : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ be the substitution with $(\mathscr{T} \circ \rho)(X) = \mathscr{T}(\rho(X))$ for all X. Then—filling in $(\eta(X), \rho(X))$ for (w, v) in the definition of $\mathbf{R}_{\mathscr{T}}$—$\eta(X) \stackrel{\alpha}{=}^*_{\mathscr{L}'} [\![(\mathscr{T} \circ \rho)(X)]\!]_{\mathscr{L}'}(\theta)$ for all $X \in \mathscr{X}$, i.e., $\eta \stackrel{\alpha}{=}^*_{\mathscr{L}'} [\![\mathscr{T} \circ \rho]\!]_{\mathscr{L}'}(\theta)$—cf. Section 11.3. So, using Lemma 11.5, (2) and compositionality,

$$[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \stackrel{\alpha}{=}^*_{\mathscr{L}'} [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}([\![\mathscr{T} \circ \rho]\!]_{\mathscr{L}'}(\theta)) \stackrel{\alpha}{=}_{\mathscr{L}'} [\![\mathscr{T}(E)[\mathscr{T} \circ \rho])]\!]_{\mathscr{L}'}(\theta) \stackrel{\alpha}{=}_{\mathscr{L}'} [\![\mathscr{T}(E[\rho])]\!]_{\mathscr{L}'}(\theta) \mathbf{R}_{\mathscr{T}} E[\rho].$$

That $\sim_{\mathscr{T}}$ is a [1-hole] congruence for $\mathscr{L}$ follows by Theorem 11.18* (taking $\approx := \mathrel{\reflectbox{$\rightsquigarrow$}} := \sim_{\mathscr{T}}$). □

The following example illustrates why in Theorem 11.20 the restriction that $\mathscr{L}$ must be a closed term language cannot be dropped.

Example 11.21 Let $\mathscr{L}$ be a language with three constants $\top$, **Yes** and **No**. Its semantics is given by $\mathbf{V}=\{+,-\}$, $\top^{\mathbf{V}}=\mathbf{Yes}^{\mathbf{V}}=+$ and $\mathbf{No}^{\mathbf{V}}=-$. Let $\mathscr{L}'$ be a language with the same three constants and a unary operator **next**. Its semantics is given by $\mathbf{V}'=\{0,1,2\}$ with $\top^{\mathbf{V}'}=2$, $\mathbf{Yes}^{\mathbf{V}'}=1$ and $\mathbf{No}^{\mathbf{V}'}=0$, while $\mathbf{next}^{\mathbf{V}'}(n)=(n+1)\mathsf{mod}\,3$. Let $\sim$ be the semantic equivalence on $\mathbf{V}'\cup\mathbf{V}$ given by $-\sim 0\not\sim 1\sim 2\sim +$. Then the identity function $\mathscr{I}$ is a translation from $\mathscr{L}$ into $\mathscr{L}'$ that is valid up to $\sim$. This is witnessed by the semantic translation $\mathbf{R}:=\{(0,-),(1,+),(2,+)\}\subseteq\sim$. The congruence closure $\sim^c$ of $\sim$ on $\mathbf{V}'$ is the identity relation. This relation cannot be extended to $\mathbf{V}\cup\mathbf{V}'$ in such a way that $\mathscr{T}$ remains valid. For 1 and 2 need to be different; yet to make $\mathscr{T}$ valid both need to be related to $+$.

Theorem 11.23 generalises Theorem 10.26 and establishes a version of Theorem 11.20 that does not require $\mathscr{L}$ to be a closed term language, or $\mathscr{T}$ to be compositional. The price to pay for that is that $\sim$ must contain the relation $\equiv_{\mathbf{R}}$ defined next. This requirement rules out the relation $\sim^c$ from Example 11.21.

Definition 11.22 Given any semantic translation $\mathbf{R}\subseteq\mathbf{V}'\times\mathbf{V}'$, let $\equiv_{\mathbf{R}}\subseteq(\mathbf{V}\uplus\mathbf{R}(\mathbf{V}))^2$ be the smallest equivalence relation on $\mathbf{V}\uplus\mathbf{R}(\mathbf{V})$ containing $\mathbf{R}$.

Theorem 11.23 Let $\mathscr{T}$ be a translation between languages $\mathscr{L}$ and $\mathscr{L}'$ that is correct w.r.t. a semantic translation $\mathbf{R}$. Then any equivalence relation $\sim\subseteq\mathbf{R}(\mathbf{V})\times\mathbf{R}(\mathbf{V})$ that contains the restriction of $\equiv_{\mathbf{R}}$ to $\mathbf{R}(\mathbf{V})$ and is a [1-hole] congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{R}(\mathbf{V})$ can be extended to an equivalence $\sim_{\mathbf{R}}$ on $\mathbf{V}\uplus\mathbf{R}(\mathbf{V})$, such that $\mathscr{T}$ is valid up to $\sim_{\mathbf{R}}$, and $\sim_{\mathbf{R}}$ also is a [1-hole] congruence for $\mathscr{L}$. □

Moreover, if $\mathbf{R}\subseteq\approx$ and $\sim\subseteq\approx$ for some equivalence $\approx$, then $\sim_{\mathbf{R}}\subseteq\approx$.

Proof: Define $\sim_{\mathbf{R}}$ on $\mathbf{V}\uplus\mathbf{R}(\mathbf{V})$ by $v\sim_{\mathbf{R}}v'$ iff $v\equiv_{\mathbf{R}}w\sim w'\equiv_{\mathbf{R}}v'$ for some $w,w'\in\mathbf{R}(\mathbf{V})$. Since the restriction of $\equiv_{\mathbf{R}}$ to $\mathbf{R}(\mathbf{V})$ is contained in $\sim$, this relation is transitive. Trivially, it is also symmetric, since $\equiv_{\mathbf{R}}$ and $\sim$ are. It is reflexive, since $\sim$ is reflexive and for each $v\in\mathbf{V}\uplus\mathbf{R}(\mathbf{V})$ there is an $w\in\mathbf{R}(\mathbf{V})$ with $v\equiv_{\mathbf{R}}w$. Thus $\sim_{\mathbf{R}}$ is an equivalence relation. On $\mathbf{R}(\mathbf{V})$) the relation $\sim_{\mathbf{R}}$ coincides with $\sim$. Since $\mathbf{R}\subseteq\equiv_{\mathbf{R}}\subseteq\sim_{\mathbf{R}}$, $\mathscr{T}$ is valid up to $\sim_{\mathbf{R}}$. That $\sim_{\mathbf{R}}$ is a [1-hole] congruence for $\mathscr{L}$ follows by Theorem 11.18*.

The last statement of Theorem 11.23 follows directly from the definition of $\sim_{\mathbf{R}}$. □

11.10 A congruence closure property for translations

The adaptation of Lemma 10.27 to general languages is straightforward.

Lemma 11.24 If a translation $\mathscr{T}$ is correct w.r.t. a semantic translation $\mathbf{R}$, then $\equiv_{\mathbf{R}}$ is a 1-hole congruence for $\mathscr{L}$.

Proof: Let $E\in\mathbb{T}_{\mathscr{L}}$ and $\rho_1,\rho_2:\mathscr{X}\to\mathbf{V}$ with $\rho_1\equiv^1_{\mathbf{R}}\rho_2$. I need to show that $[\![E]\!]_{\mathscr{L}}(\rho_1)\sim[\![E]\!]_{\mathscr{L}}(\rho_2)$.

Let $X\in\mathscr{X}$ be such that $\rho_1(X)\equiv_{\mathbf{R}}\rho_2(X)$ and $\rho_1(Y)=\rho_2(Y)$ for all $Y\neq X$. Then, for some $n\geq 0$ there are $v_0,...,v_n\in\mathbf{V}$ and $w_1,...,w_n\in\mathbf{R}(\mathbf{V})$ with $\rho_1(X)=v_0\;\mathbf{R}^{-1}\;w_1\;\mathbf{R}\;v_1\;\mathbf{R}^{-1}\;w_2\;\mathbf{R}\;v_2\;\mathbf{R}^{-1}\cdots\mathbf{R}\;v_n=\rho_2(X)$. For $i=0,\ldots,n$ let $\nu_i:\mathscr{X}\to\mathbf{V}$ be given by $\nu_i(X)=v_i$ and $\nu_i(Y)=\rho_2(Y)$ for $Y\neq X$, and for $i=1,\ldots,n$ let $\eta_i:\mathscr{X}\to\mathbf{R}(\mathbf{V})$ be given by $\eta_i(X)=w_i$ and $\eta_i(Y)=\eta(Y)$ for $Y\neq X$, using some $\eta:\mathscr{X}\to\mathbf{R}(\mathbf{V})$ such that $\eta\;\mathbf{R}\;\rho_2$. Then $\rho_1=\nu_0\;\mathbf{R}^{-1}\;\eta_1\;\mathbf{R}\;\nu_1\;\mathbf{R}^{-1}\;\eta_2\;\mathbf{R}\;\nu_2\;\mathbf{R}^{-1}\cdots\mathbf{R}\;\nu_n=\rho_2$. So by Definition 11.2, $[\![E]\!]_{\mathscr{L}}(\nu_0)\;\mathbf{R}^{-1}\;[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta_1)\;\mathbf{R}\;[\![E]\!]_{\mathscr{L}}(\nu_1)\;\mathbf{R}^{-1}\;[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta_2)\;\mathbf{R}\;[\![E]\!]_{\mathscr{L}}(\nu_2)\;\mathbf{R}^{-1}\cdots\mathbf{R}\;[\![E]\!]_{\mathscr{L}}(\nu_n)$. Thus $[\![E]\!]_{\mathscr{L}}(\rho_1)\equiv_{\mathbf{R}}[\![E]\!]_{\mathscr{L}}(\rho_2)$. □

It is well known and easy to check that the collection of equivalence relations on any domain $\mathbf{V}$, ordered by inclusion, forms a complete lattice—namely the intersection of arbitrarily many equivalence relations is again an equivalence relation. Likewise, the collection of 1-hole congruences for $\mathscr{L}$ is also a complete lattice, and moreover a complete sublattice of the complete lattice of equivalence relations on $\mathbf{V}$. The latter implies that for any collection C of 1-hole congruence relations, the least equivalence relation that contains all elements of C (exists and) happens to be a 1-hole congruence relation. Again, this is a

property that is well known [55] and easy to prove. It follows that for any equivalence relation $\approx$ there exists a largest 1-hole congruence for $\mathscr{L}$ contained in $\approx$. I will denote this 1-hole congruence by $\approx^{1c}_{\mathscr{L}}$, and call it the *congruence closure* of $\approx$ w.r.t. $\mathscr{L}$. One has $v_1 \approx^{1c}_{\mathscr{L}} v_2$ for $v_1, v_2 \in \mathbf{V}$ iff $[\![E]\!]_{\mathscr{L}}(\rho_1) \approx [\![E]\!]_{\mathscr{L}}(\rho_2)$ for all $\mathscr{L}$-expressions E and all valuations $\rho_1, \rho_2 : \mathscr{X} \to \mathbf{V}$ with $\rho_1(X) = v_1$ and $\rho_2(X) = v_2$ for some $X \in \mathscr{X}$ and $\rho_1(Y) = \rho_2(Y)$ for all variables $Y \neq X$. Such results do not generally hold for congruences.

Example 11.25 Continue Example 11.14, but skipping the operator $+$. Let $\sim_k$ be the equivalence on $\mathbf{V}$ defined by $(n,m) \sim_k (n',m')$ iff $n = n' \wedge (m = m' \vee m, m' \leq k)$. It is easy to check that all $\sim_k$ for $k \in \mathbb{N}$ are congruences on the reduced $\mathscr{L}$, and contained in $\sim$. Yet their least upper bound (in the lattice of equivalence relations on $\mathbf{V}$) is $\sim$, which is not a congruence itself. In particular, there is no largest congruence contained in $\sim$.

When dealing with languages $\mathscr{L}$ in which all operators and other constructs have a finite arity, so that each $E \in \mathbb{T}_{\mathscr{L}}$ contains only finitely many variables, there is no difference between a congruence and a 1-hole congruence, and thus $\sim^{1c}_{\mathscr{L}}$ is a congruence relation for any equivalence $\sim$. I will apply the theory of expressiveness presented in this paper also to languages like CCS that have operators (such as $\sum_{i \in I} E_i$) of infinite arity. However, in all such cases I am currently aware of, the relevant choices of $\mathscr{L}$ and $\sim$ have the property that $\sim^{1c}_{\mathscr{L}}$ is in fact a congruence relation. As an example, consider weak bisimilarity [70]. This equivalence relation fails to be a congruence for $\sum$. However, the coarsest 1-hole congruence contained in this relation, often called *rooted* weak bisimilarity, happens to be a congruence. In fact, when congruence-closing weak bisimilarity w.r.t. the binary sum, the result [40] is also a congruence for the infinitary sum, as well as for all other operators of CCS [70].

Given a translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$, I also speak of the *congruence closure* $\approx^{1c}_{\mathscr{T}(\mathscr{L}),\mathbf{W}}$ of $\approx$ for $\mathscr{T}(\mathscr{L})$ on a closed set $\mathbf{W}$. This is the largest 1-hole congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ that is contained in a given equivalence $\approx$ on $\mathbf{W}$. It can be characterised by $v \approx^{1c}_{\mathscr{T}(\mathscr{L}),\mathbf{W}} w$ for $v, w \in \mathbf{W}$ iff $F[\theta] \approx F(\eta)$ for all $\mathscr{L}'$-expressions F of the form $\mathscr{T}(E)$ and all valuations $\theta, \eta : \mathscr{X} \to \mathbf{W}$ with $\theta(X) = v$ and $\eta(X) = w$ for some $X \in \mathscr{X}$ and $\theta(Y) = \eta(Y)$ for all variables $Y \neq X$.

Now Theorem 10.28 generalises as follows.

Theorem 11.26 Let $\mathscr{T}$ be a translation from a language $\mathscr{L}$, with semantic domain $\mathbf{V}$, into a language $\mathscr{L}'$, with domain $\mathbf{V}'$, valid up to a semantic equivalence $\approx$. Then $\mathscr{T}$ is valid even up to an equivalence $\approx^{1c}_{\mathscr{T}}$, contained in $\approx$, such that

(1) the restriction of $\approx^{1c}_{\mathscr{T}}$ to $\mathbf{V}$ is the largest 1-hole congruence for $\mathscr{L}$ contained in $\approx$,

and there exists a closed set $\mathbf{W}$ with $\mathbf{W} \subseteq \mathbf{V}'$ such that

(2) the restriction of $\approx^{1c}_{\mathscr{T}}$ to $\mathbf{W}$ is the largest 1-hole congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ that is contained in $\approx$, and

(3) each equivalence class of $\approx^{1c}_{\mathscr{T}}$ is the union of an equivalence class of $\approx^{1c}_{\mathscr{L}}$ and one of $\approx^{1c}_{\mathscr{T}(\mathscr{L}),\mathbf{W}}$. □

Proof: By assumption the translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ is correct w.r.t. a semantic translation $\mathbf{R} \subseteq \approx$. Take $\mathbf{W} := \mathbf{R}(\mathbf{V})$. By Proposition 11.16 it is closed under $\mathscr{T}(\mathscr{L})$. Let $\approx^{1c}_{\mathscr{L},\mathbf{R}}$, the *congruence closure* of $\approx$ w.r.t. $\mathscr{L}$ and $\mathbf{R}$, be the binary relation on $\mathbf{V} \uplus \mathbf{W}$ defined by $w_1 \approx^{1c}_{\mathscr{L},\mathbf{R}} w_2$ iff $w_1 \equiv_{\mathbf{R}} v_1 \approx^{1c}_{\mathscr{L}} v_2 \equiv_{\mathbf{R}} w_2$ for some $v_1, v_2 \in \mathbf{V}$. Here $\approx^{1c}_{\mathscr{L}}$ is the largest 1-hole congruence for $\mathscr{L}$, defined on $\mathbf{V}$, that is contained in $\approx$.

As $\approx$ is an equivalence relation defined om $\mathbf{V} \uplus \mathbf{W}$ that contains $\mathbf{R}$, and $\equiv_{\mathbf{R}}$ is the smallest such equivalence, one has $\equiv_{\mathbf{R}} \subseteq \approx$. Since $\equiv_{\mathbf{R}}$ restricted to $\mathbf{V}$ is a 1-hole congruence for $\mathscr{L}$ contained in $\approx$ (by Lemma 11.24), and $\approx^{1c}_{\mathscr{L}}$ is the largest 1-hole congruence for $\mathscr{L}$ contained in $\approx$, one has $v \equiv_{\mathbf{R}} w \Rightarrow v \approx^{1c}_{\mathscr{L}} w$ for all $v, w \in \mathbf{V}$. From this it follows that $\approx^{1c}_{\mathscr{L},\mathbf{R}}$ is transitive. Trivially, this relation is also symmetric, since $\equiv_{\mathbf{R}}$ and $\approx^{c}_{\mathscr{L}}$ are. It is reflexive, since $\approx^{c}_{\mathscr{L}}$ is reflexive and for each $w \in \mathbf{V} \uplus \mathbf{W}$ there is a $v \in \mathbf{V}$ with $w \equiv_{\mathbf{R}} v$. Thus, $\approx^{1c}_{\mathscr{L},\mathbf{R}}$ is an equivalence relation. Since $\approx^{1c}_{\mathscr{L}}$ and $\mathbf{R}$, and hence also $\equiv_{\mathbf{R}}$, are

contained in $\approx$, so is $\approx^{1c}_{\mathscr{L},\mathbf{R}}$. Moreover, $\mathbf{R} \subseteq \equiv_{\mathbf{R}} \subseteq \approx^{1c}_{\mathscr{L},\mathbf{R}}$, so $\mathscr{T}$ is valid up to $\approx^{1c}_{\mathscr{L},\mathbf{R}}$. It remains to check properties (1)–(3).

Since $\equiv_{\mathbf{R}}$ restricted to $\mathbf{V}$ is included in $\approx^{c}_{\mathscr{L}}$, Property (1) of Theorem 11.26, saying that $\approx^{c}_{\mathscr{L},\mathbf{R}}$ restricted to $\mathbf{V}$ coincides with $\approx^{c}_{\mathscr{L}}$, follows immediately from the definition of $\approx^{c}_{\mathscr{L},\mathbf{R}}$.

Regarding Property (2), by Theorem 11.19 $\approx^{1c}_{\mathscr{L},\mathbf{R}}$ is a 1-hole congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W} = \mathbf{R}(\mathbf{V})$, contained in $\approx$. To show that it is the largest, let $\sim$ be any other 1-hole congruence on $\mathbf{W}$ contained in $\approx$. By Lemma 11.24 and Theorem 11.19 also $\equiv_{\mathbf{R}}$ is a 1-hole congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$ contained in $\approx$. Let $\sim' \subseteq \mathbf{W} \times \mathbf{W}$ be the smallest equivalence on $\mathbf{W}$ containing both $\sim$ and $\equiv_{\mathbf{R}}$; one has $w \sim' w'$ iff $w = w_0 \sim w'_0 \equiv_{\mathbf{R}} w_1 \sim w'_1 \equiv_{\mathbf{R}} \cdots \equiv_{\mathbf{R}} w_k \sim w'_k = w'$ for some $k \geq 0$. By Definition 11.13, also $\sim'$ is a 1-hole congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$, contained in $\approx$.

Let $\sim_{\mathbf{R}}$ be the extension of $\sim'$ to $\mathbf{V} \uplus \mathbf{W}$ as defined in the proof of Theorem 11.23: $v \sim_{\mathbf{R}} v'$ iff $v \equiv_{\mathbf{R}} w \sim' w' \equiv_{\mathbf{R}} v'$ for some $w, w' \in \mathbf{R}(\mathscr{L})$. This definition entails that $\sim_{\mathbf{R}} \subseteq \approx$, and by Theorem 11.23 $\sim_{\mathbf{R}}$ is a 1-hole congruence for $\mathscr{L}$.

Now suppose $w \sim w'$ with $w, w' \in \mathbf{W}$. Take $v, v' \in \mathrm{T}_{\mathscr{L}}$ such that $w \mathbf{R} v$ and $w' \mathbf{R} v'$. Then $v \sim_{\mathbf{R}} v'$. Since $\sim_{\mathbf{R}}$ is a 1-hole congruence for $\mathscr{L}$ contained in $\approx$, and $\approx^{c}_{\mathscr{L}}$ is the largest such 1-hole congruence, one has $v \approx^{c}_{\mathscr{L}} v'$, and thus $w \approx^{c}_{\mathscr{L},\mathbf{R}} w'$. This shows that $\sim \subseteq \approx^{c}_{\mathscr{L},\mathbf{R}}$ and yields Property (2).

Each equivalence class of $\approx^{c}_{\mathscr{L},\mathbf{R}}$ contains a process $v \in \mathbf{V}$ and a process $w \in \mathbf{W}$ such that $v \mathbf{R} w$. Using this, Property (3) is a simple consequence of (1) and (2). □

The relation $\approx^{1c}_{\mathscr{T}} := \approx^{1c}_{\mathscr{L},\mathbf{R}}$ constructed above is on $\mathbf{V}$ completely determined by $\mathscr{L}$ and $\approx$. The following example shows that in general the whole relation $\approx^{1c}_{\mathscr{L}}$ is not completely determined by $\mathscr{L}$ and $\approx$.

Example 11.27 Let $\mathscr{L}$ be a language with two constants **Yes** and **No** and a binary operator **same**. Its semantics is given by $\mathbf{V} = \{0, 1, \top, \bot\}$, $\mathbf{Yes}^{\mathbf{V}} = 1$, $\mathbf{No}^{\mathbf{V}} = 0$ and $\mathbf{same}^{\mathbf{V}}(x,y) = 1$ if $x = y$, while $\mathbf{same}^{\mathbf{V}}(x,y) = 0$ if $x \neq y$. The semantic values $\top$ and $\bot$ are not denotable as the interpretation of closed terms. Let $\mathscr{L}'$ be an exact copy of $\mathscr{L}$, except that the semantic values are primed. Let $\approx$ be the semantic equivalence on $\mathbf{V} \cup \mathbf{V}'$ given by $0 \approx 0' \not\approx 1 \approx 1' \not\approx \top \approx \top' \approx \bot \approx \bot'$. Then the identity function $\mathscr{I}$ is a translation from $\mathscr{L}$ into $\mathscr{L}'$ that is valid up to $\approx$. This is witnessed by the semantic translation $\mathbf{R}^{\dagger} := \{(0',0),(1',1),(\top',\bot),(\bot',\top)\} \subseteq \approx$, or, alternatively, by the semantic translation $\mathbf{R} := \{(0',0),(1',1),(\top',\top),(\bot',\bot)\} \subseteq \approx$. Upon taking the congruence closure $\approx^{1}_{\mathscr{L}}$, the four semantic values of $\mathbf{V}$ become inequivalent. Yet, there are two candidates for $\approx^{1c}_{\mathscr{L}}$, namely $\approx^{1c}_{\mathscr{L},\mathbf{R}}$ and $\approx^{1c}_{\mathscr{L},\mathbf{R}^{\dagger}}$.

12 Alternative concepts of a valid translation

While for closed term languages without variables or syntactic constructs beyond operators I proposed just one concept of a valid translation (cf. Section 3), when merely adding variables to those languages I contemplated eight generalisations (cf. Section 7). Of those, one (B^{-}-validity) was rejected as being counterintuitive, and one (C^{-}-validity) was selected as my favourite. This is the concept of validity I generalised to arbitrary languages in Sections 10 and 11. C-validity is compositional C^{-}-validity, and has been treated as well; by Theorem 11.7 it gives rise to the same notion of expressiveness. In this section I will generalise the remaining notions to general languages, and relate them to the the notion of validity chosen here, as well as to my own prior work.

12.1 Absolute expressiveness

In Section 8 defines a concept of absolute expressiveness, parametrised by the choice of a semantic equivalence or preorder $\mathrel{\scriptstyle\sim}$. For each choice of $\mathrel{\scriptstyle\sim}$ it induces an expressiveness preorder on languages.

Absolute expressiveness is a weaker notion than relative expressiveness (cf. Definition 11.3), in the sense that if the relative expressiveness up to $\sim$ of language $\mathscr{L}'$ is at least as great as that of $\mathscr{L}$, then certainly the absolute expressiveness up to $\sim$ of $\mathscr{L}'$ is at least as great as that of $\mathscr{L}$, and Example 8.3 shows that the reverse does not hold.

Definition 8.1 of absolute expressive generalises verbatim to arbitrary languages. Observation 8.2, characterising absolute expressiveness in terms of the existence of an A^--valid translation $\mathscr{T} : \mathrm{T}_{\mathscr{L}} \to \mathrm{T}_{\mathscr{L}'}$ between the closed terms of two languages, applies to general languages too. I do not see any use for A^--valid translations of type $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$, as A^--validity ignores the application of $\mathscr{T}$ to open terms anyway. Neither see I any use in considering translations that are not fvr (cf. Section 10.8) in the sense that $\mathscr{T}(P)$ would be allowed to contain variables.

12.2 A-validity

A generalisation of the concept of A-validity to arbitrary languages has been studied in [42, Section 9]. 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 case there exists a $v \in \mathbf{V}$ for which there is no $\sim$-counterpart $v' \in \mathbf{V}'$, 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, [42] requires

$$\forall v \in \mathbf{V}.\ \exists v' \in \mathbf{V}'.\ v' \sim v. \tag{3}$$

This implies that for any valuation $\rho : \mathscr{X} \to \mathbf{V}$ there is a valuation $\eta : \mathscr{X} \to \mathbf{V}'$ with $\eta \sim \rho$. With this caveat, the notion of A-validity from Section 7, which applied to fvr-translations only (cf. Section 10.8), generalises straightforwardly to arbitrary languages.

Definition 12.1 An fvr-translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ is *A-valid* up to $\sim$ if (3) holds and $\mathscr{T}$ is compositional and satisfies $[\![\mathscr{T}(P)]\!]_{\mathscr{L}'} \sim [\![P]\!]_{\mathscr{L}}$ for all closed $\mathscr{L}$-expressions $P \in \mathrm{T}_{\mathscr{L}}$.

The next proposition says that A-validity gives rise to a weaker (or equal) concept of relative expressiveness than the notion of (C-)validity of Definition 11.3.

Proposition 12.2 If there exists a valid fvr-translation from $\mathscr{L}$ into $\mathscr{L}'$ up to $\sim$ (Definition 11.3) then there exists an A-valid fvr-translation from $\mathscr{L}$ into $\mathscr{L}'$ up to $\sim$.

Proof: Suppose a valid fvr-valid translation up to $\sim$ exists. By Theorem 11.7, together with the last sentence in the proof of Proposition 11.8, there exists a compositional fvr-translation $\mathscr{T}$ that is valid up to $\sim$. Since $\mathscr{T}$ is correct w.r.t. some semantic translation $\mathbf{R} \subseteq \sim$, one has $\forall v \in \mathbf{V}.\exists v' \in \mathbf{V}'.\, v'\ \mathbf{R}\ v$ by Definition 11.1, which implies (3). By Definitions 11.2 and 12.1, $\mathscr{T}$ is A-valid up to $\sim$. □

The reverse implication holds only when assuming that the source language $\mathscr{L}$ has the property that all values in its domain of interpretation are denotable by closed terms. This property can be stated as

$$\forall v \in \mathbf{V}.\ \exists P \in \mathrm{T}_{\mathscr{L}}.\ [\![P]\!]_{\mathscr{L}} = v\,. \tag{4}$$

Theorem 12.3 Let $\mathscr{L}$ satisfy (4), and let $\mathscr{T}$ be an fvr-translation from $\mathscr{L}$ into $\mathscr{L}'$ that is A-valid up to $\sim$, where $\sim$ contains $\overset{\alpha}{=}_{\mathscr{L}}$ and $\overset{\alpha}{=}_{\mathscr{L}'}$. Then $\mathscr{T}$ is valid up to $\sim$.

Proof: Define the semantic translation $\mathbf{R} \subseteq \mathbf{V}' \times \mathbf{V}$ by $\mathbf{R} := \{([\![\mathscr{T}(P)]\!]_{\mathscr{L}'}, [\![P]\!]_{\mathscr{L}}) \mid P \in \mathrm{T}_{\mathscr{L}}\}$. Then $\mathbf{R} \subseteq \sim$, since $\mathscr{T}$ is A-valid up to $\sim$. Since $\sim$ contains $\overset{\alpha}{=}_{\mathscr{L}}$ and $\overset{\alpha}{=}_{\mathscr{L}'}$, also $\mathbf{R}^\alpha \subseteq \sim$, as defined in Section 11.2. Hence it suffices to show that $\mathscr{T}$ is correct w.r.t. $\mathbf{R}^\alpha$. So let $E \in \mathbb{T}_{\mathscr{L}}$, $\rho : \mathscr{X} \to \mathbf{V}$ and $\eta : \mathscr{X} \to \mathbf{V}'$ with $\eta\ \mathbf{R}^\alpha\ \rho$. I need to show that $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta)\ \mathbf{R}^\alpha\ [\![E]\!]_{\mathscr{L}}(\rho)$.

Pick $\bar\rho:\mathscr{X}\to\mathbf{V}$ and $\bar\eta:\mathscr{X}\to\mathbf{V}'$ such that $\eta \stackrel{\alpha}{=}^*_{\mathscr{L}'} \bar\eta\ \mathbf{R}\ \bar\rho \stackrel{\alpha}{=}^*_{\mathscr{L}} \rho$. Then there is a closed substitution $\sigma:\mathscr{X}\to \mathrm{T}_{\mathscr{L}}$ such that $\bar\rho(X)=[\![\sigma(X)]\!]_{\mathscr{L}}$ and $\bar\eta(X)=[\![\mathscr{T}(\sigma(X))]\!]_{\mathscr{L}'}$ for all $X\in X$, i.e., $\bar\rho=[\![\sigma]\!]_{\mathscr{L}}$ and $\bar\eta=[\![\mathscr{T}\circ\sigma]\!]_{\mathscr{L}'}$. Hence,

$$\begin{array}{lll}
[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) & \stackrel{\alpha}{=}^*_{\mathscr{L}'} & [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\bar\eta) \quad \text{by Lemma 11.5}\\
& = & [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}([\![\mathscr{T}\circ\sigma]\!]_{\mathscr{L}'}) \quad \text{derived above}\\
& \stackrel{\alpha}{=}_{\mathscr{L}'} & [\![\mathscr{T}(E)[\mathscr{T}\circ\sigma]]\!]_{\mathscr{L}'} \quad \text{by (2)}\\
& \stackrel{\alpha}{=}_{\mathscr{L}'} & [\![\mathscr{T}(E[\sigma])]\!]_{\mathscr{L}'} \quad \text{by compositionality of } \mathscr{T}\\
& \mathbf{R} & [\![E[\sigma]]\!]_{\mathscr{L}} \quad \text{by definition of } \mathbf{R}\\
& \stackrel{\alpha}{=}_{\mathscr{L}} & [\![E]\!]_{\mathscr{L}}([\![\sigma]\!]_{\mathscr{L}}) \quad \text{by (2)}\\
& = & [\![E]\!]_{\mathscr{L}}(\bar\rho) \quad \text{derived above}\\
& \stackrel{\alpha}{=}^*_{\mathscr{L}} & [\![E]\!]_{\mathscr{L}}(\rho) \quad \text{by Lemma 11.5.}
\end{array}$$ □

Example 2 in [42] shows that this result does not generalise to languages that do not satisfy (4). In such a case an A-valid translation as in Definition 12.1 can be intuitively unsatisfactory, as it may translate an open term E into a term $\mathscr{T}(E)$ that is not related by $\leftrightsquigarrow$ to E when instantiating a variable in E by a value that is undenotable by a closed term. For this reason, [42, Section 9] presents a theory of expressiveness based on A-valid translations solely for source languages satisfying (4). The above results show that for such source languages A-validity up to $\leftrightsquigarrow$ is the same as C-validity up to $\leftrightsquigarrow$, at least under the reasonable assumption that $\leftrightsquigarrow$ contains $\stackrel{\alpha}{=}_{\mathscr{L}}$ and $\stackrel{\alpha}{=}_{\mathscr{L}'}$.

Whereas in Section 7 my choice of C$^-$-validity over A-validity was largely made on aesthetic grounds, here C$^-$-validity has the added advantage that it applies to a larger class of languages. In this way my present paper generalises the work of [42, Section 9].

In [42, Section 9] the definition of A-validity was generalised to translations that need not be fvr. Applied to this corner case, Theorem 12.3 still holds [45, Theorem 7], but Proposition 12.2 does not. Yet, Proposition 12.2 does hold when additionally requiring that also the target language satisfies (4) [45, Section 15]. As this corner case is not intrinsically interesting, one can just as well stick to the way non-fvr-translations are incorporated in (C$^-$-)validity.

12.3 B-validity

In order to generalise Definition 7.1 to general languages, the closed substitution $\varsigma:\mathscr{X}\to\mathrm{T}_{\mathscr{L}}$ needs to be replaced by a valuation $\rho:\mathscr{X}\to\mathbf{V}$. Consequently, the $\mathscr{T}$ in $\mathscr{T}\circ\varsigma$ needs to be replaced by a *semantic counterpart* $\mathbf{T}:\mathbf{V}\to\mathbf{V}'$ of the translation $\mathscr{T}:\mathbb{T}_{\mathscr{L}}\to\mathbb{T}_{\mathscr{L}'}$; it maps the possible meanings of $\mathscr{L}$-expressions into the possible meanings of $\mathscr{L}'$-expressions. Just as the restriction of $\mathscr{T}$ to closed terms employed in the context $\mathscr{T}\circ\varsigma$ in Definition 7.1, it needs to satisfy $\mathbf{T}(v)\leftrightsquigarrow v$ for all $v\in\mathbf{V}$.

Definition 12.4 A translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ is *B-valid* up to $\leftrightsquigarrow$ if it is compositional and there exists a mapping $\mathbf{T}:\mathbf{V}\to\mathbf{V}'$ such that $\mathbf{T}(v)\leftrightsquigarrow v$ for all $v\in\mathbf{V}$ and $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\mathbf{T}\circ\rho)\leftrightsquigarrow[\![E]\!]_{\mathscr{L}}(\rho)$ for each $E\in\mathbb{T}_{\mathscr{L}}$ and each $\rho:\mathscr{X}\to\mathbf{V}$.

Note that the existence of $\mathbf{T}$ implies (3) above.

Proposition 12.5 Any compositional (C-)valid translation up to a preorder $\leftrightsquigarrow$ is B-valid up to $\leftrightsquigarrow$.

Proof: Let $\mathscr{T}$ be correct w.r.t. the semantic translation $\mathbf{R}\subseteq\leftrightsquigarrow$. Take any $\mathbf{T}:\mathbf{V}\to\mathbf{V}'$ with $\mathbf{T}\subseteq\mathbf{R}$. Then $\mathbf{T}(v)\leftrightsquigarrow v$ for all $v\in\mathbf{V}$. Moreover, $\mathbf{T}\circ\rho\ \mathbf{R}\ \rho$ for any valuation $\rho:\mathscr{X}\to\mathbf{V}$. So, by Definition 11.2, $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\mathbf{T}\circ\rho)\ \mathbf{R}\ [\![E]\!]_{\mathscr{L}}(\rho)$, and thus $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\mathbf{T}\circ\rho)\leftrightsquigarrow[\![E]\!]_{\mathscr{L}}(\rho)$. Hence $\mathscr{T}$ is B-valid up to $\leftrightsquigarrow$. □

Together with Theorem 11.7 this implies that if a valid translation up to $\leftrightsquigarrow$ exists, then there is a B-valid translation. Below I establish the reverse, but only for fvr-translations, and preorders $\leftrightsquigarrow$ containing $\stackrel{\alpha}{=}_{\mathscr{L}}$ and $\stackrel{\alpha}{=}_{\mathscr{L}'}$.

Theorem 12.6 Any fvr-translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ between languages $\mathscr{L}$ and $\mathscr{L}'$ that is B-valid up to $\mathbin{\sim}$, for $\mathbin{\sim}$ be a preorder that contains $\overset{\alpha}{=}_{\mathscr{L}}$ and $\overset{\alpha}{=}_{\mathscr{L}'}$, is (C-)valid up to $\mathbin{\sim}$.

Proof: Let $\mathscr{T}$ be a fvr-translation that is B-valid up to $\mathbin{\sim}$, say by means of the semantic counterpart $\mathbf{T} : \mathbf{V} \to \mathbf{V}'$. Let $\mathbf{R} \subseteq \mathbf{V}' \times \mathbf{V}$ be the smallest relation containing $\mathbf{T}^{-1}$ such that $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$. Its existence can be shown as in the proof of Proposition 11.17. I need to show that $\mathbf{R} \subseteq \mathbin{\sim}$.

Claim: If $v' \mathbf{R} v$ then $v' \overset{\alpha}{=}^*_{\mathscr{L}'} [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\mathbf{T} \circ \nu)$ and $[\![E]\!]_{\mathscr{L}}(\nu) \overset{\alpha}{=}^*_{\mathscr{L}} v$ for some $E \in \mathbb{T}_{\mathscr{L}}$ and $\nu : \mathscr{X} \to \mathbf{V}$.

Establishing the validity of this claim is sufficient, because by Definition 12.4 and the transitivity of $\mathbin{\sim}$, using that $\overset{\alpha}{=}_{\mathscr{L}}, \overset{\alpha}{=}_{\mathscr{L}'} \subseteq \mathbin{\sim}$, it immediately implies that $\mathbf{R} \subseteq \mathbin{\sim}$.

I prove the claim by induction on the construction of $\mathbf{R}$ (similar to the proof of Proposition 11.17).

Induction base: If $v' \mathbf{R} v$ because $v' = \mathbf{T}(v)$ the claim holds by taking $E := X$ and $\nu(X) = v$.

Induction step: Let $v' = [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta)$ and $v = [\![E]\!]_{\mathscr{L}}(\rho)$ for some $E \in \mathbb{T}_{\mathscr{L}}$, $\eta : \mathscr{X} \to \mathbf{V}'$ and $\rho : \mathscr{X} \to \mathbf{V}$ with $\eta \mathbf{R} \rho$. By induction one may assume that for each $X \in \mathit{fv}(E)$ there are $E_X \in \mathbb{T}_{\mathscr{L}}$ and $\nu_X : \mathscr{X} \to \mathbf{V}$ such that $\eta(X) \overset{\alpha}{=}^*_{\mathscr{L}'} [\![\mathscr{T}(E_X)]\!]_{\mathscr{L}'}(\mathbf{T} \circ \nu_X)$ and $[\![E_X]\!]_{\mathscr{L}}(\nu_X) \overset{\alpha}{=}^*_{\mathscr{L}} \rho(X)$. Let $(\sigma_X)_{X \in \mathit{fv}(E)}$ be a family of injective renamings of variables with $\mathit{dom}(\sigma_X) = \mathit{fv}(E_X)$, such that the sets $\mathit{range}(\sigma_X) := \sigma_X(\mathit{fv}(E_X))$ for $X \in \mathit{fv}(E)$ are pairwise disjoint. Here I assume the pool of variables to draw from is large enough. Note that $[\![\sigma_X]\!]_{\mathscr{L}}(\nu_X \circ \sigma_X^{-1}) = \nu_X$ and $[\![\sigma_X]\!]_{\mathscr{L}'}(\mathbf{T} \circ \nu_X \circ \sigma_X^{-1}) = \mathbf{T} \circ \nu_X$. Therefore, using (2), $[\![\mathscr{T}(E_X)]\!]_{\mathscr{L}'}(\mathbf{T} \circ \nu_X) \overset{\alpha}{=}_{\mathscr{L}'} [\![\mathscr{T}(E_X)[\sigma_X]]\!]_{\mathscr{L}'}(\mathbf{T} \circ \nu_X \circ \sigma_X^{-1})$ and $[\![E_X[\sigma_X]]\!]_{\mathscr{L}}(\nu_X \circ \sigma_X^{-1}) \overset{\alpha}{=}_{\mathscr{L}} [\![E_X]\!]_{\mathscr{L}}(\nu_X)$. By the compositionality of $\mathscr{T}$, $\mathscr{T}(E_X)[\sigma_X] \overset{\alpha}{=} \mathscr{T}(E_X[\sigma_X])$, so $\eta(X) \overset{\alpha}{=}^*_{\mathscr{L}'} [\![\mathscr{T}(E_X[\sigma_X])]\!]_{\mathscr{L}'}(\mathbf{T} \circ \nu_X \circ \sigma_X^{-1})$.

Now let $\nu : \mathscr{X} \to \mathbf{V}$ be a valuation such that $\nu(Y) = \nu_X \circ \sigma_X^{-1}(Y)$ if $Y \in \mathit{fv}(E_X[\sigma_X])$ for some $X \in \mathit{fv}(E)$. Furthermore, define the substitution $\sigma : \mathit{fv}(E) \to \mathbb{T}_{\mathscr{L}}$ by $\sigma(X) = E_X[\sigma_X]$. Then $\rho(X) \overset{\alpha}{=}^*_{\mathscr{L}} [\![\sigma(X)]\!]_{\mathscr{L}}(\nu)$ for all $X \in \mathit{fv}(E)$, so $v \overset{\alpha}{=}^*_{\mathscr{L}} [\![E]\!]_{\mathscr{L}}([\![\sigma]\!]_{\mathscr{L}}(\nu))$ by Lemma 11.5. Likewise, $\eta(X) \overset{\alpha}{=}^*_{\mathscr{L}'} [\![\mathscr{T}(\sigma(X))]\!]_{\mathscr{L}'}(\mathbf{T} \circ \nu)$ for all $X \in \mathit{fv}(E) \supseteq \mathit{fv}(\mathscr{T}(E))$, so $v' \overset{\alpha}{=}^*_{\mathscr{L}'} [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}([\![\mathscr{T} \circ \sigma]\!]_{\mathscr{L}'}(\mathbf{T} \circ \nu))$. Therefore, (2) yields $v \overset{\alpha}{=}^*_{\mathscr{L}} [\![E[\sigma]]\!]_{\mathscr{L}}(\nu)$ and $v' \overset{\alpha}{=}^*_{\mathscr{L}'} [\![\mathscr{T}(E)[\mathscr{T} \circ \sigma]]\!]_{\mathscr{L}'}(\mathbf{T} \circ \nu)$. By compositionality, $\mathscr{T}(E)[\mathscr{T} \circ \sigma] \overset{\alpha}{=} \mathscr{T}(E[\sigma])$, from which it follows that $v' \overset{\alpha}{=}^*_{\mathscr{L}'} [\![\mathscr{T}(E[\sigma])]\!]_{\mathscr{L}'}(\mathbf{T} \circ \nu)$. □

There exist B-valid translations up to $\mathbin{\sim}$ that are not fvr-translations and are not valid in the sense of Definition 11.3. Examples are hard to find and do not appear very natural. Moreover, some major results I establish about valid translations (Theorems 11.26 and 11.20) do not generalise to such examples. For these reasons I prefer to exclude such examples from my definition of a valid translation. This adds to my preference of C$^-$-validity (or C-validity) over B-validity.

Example 12.7 Let $\mathscr{L}$ be a language with constants $\mathbf{0}$ and $\mathbf{1}$, and a semantics given by $\mathbf{V} = \{a, b\}$, $\mathbf{0}^{\mathbf{V}} = a$ and $\mathbf{1}^{\mathbf{V}} = b$. Let $\mathscr{L}'$ be a language with a unary operator f, and a semantics given by $\mathbf{V}' = \{1, 2, 3, 4\}$ and $f^{\mathbf{V}'}(n) = n+1 \pmod 4$. Finally, let $\approx$ be the equivalence on $\mathbf{V} \cup \mathbf{V}'$ given by $a \approx 1 \approx 2 \not\approx 3 \approx 4 \approx b$.

The translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ given by $\mathscr{T}(\mathbf{0}) = f(X_0)$ and $\mathscr{T}(\mathbf{1}) = f(f(f(X_0)))$ for some $X_0 \in \mathscr{X}$, and $\mathscr{T}(X) = X$ for all $X \in \mathscr{X}$, is compositional by construction, and B-valid up to $\approx$. This is witnessed by its semantic counterpart $\mathbf{T} : \mathbf{V} \to \mathbf{V}'$ given by $\mathbf{T}(a) = 1$ and $\mathbf{T}(b) = 4$.

Since $\mathrm{T}_{\mathscr{L}'} = \emptyset$, there do not exist fvr-translations from $\mathscr{L}$ to $\mathscr{L}'$. Up to symmetry, $\mathscr{T}$ is in fact the only translation from $\mathscr{L}$ to $\mathscr{L}'$ that is B-valid up to $\approx$. For considering that $f^4(n) = n$ and the choice of X_0 is immaterial, any such translation $\mathscr{T}_{k,\ell}$ must satisfy $\mathscr{T}_{k,\ell}(\mathbf{0}) = f^k(X_0)$ and $\mathscr{T}_{k,\ell}(\mathbf{1}) = f^\ell(X_0)$, where $k, \ell \in \{0, 1, 2, 3\}$. The only choices for $\mathbf{T}(a)$ are 1 or 2, and by symmetry one can pick $\mathbf{T}(a) = 1$. Hence, to satisfy $[\![\mathscr{T}_{k,\ell}(\mathbf{0})]\!]_{\mathscr{L}'}(\mathbf{T} \circ \rho) \approx [\![\mathbf{0}]\!]_{\mathscr{L}}(\rho)$ when $\rho(X_0) = a$, one must take $k \in \{0, 1\}$. To also satisfy this formula when $\rho(X_0) = b$, one must take $k = 1$ and $\mathbf{T}(b) = 4$. This forces $\ell \in \{2, 3\} \cap \{3, 0\}$, so $\mathscr{T}_{k,\ell} = \mathscr{T}_{1,3}$.

However, there does not exist a translation from $\mathscr{L}$ to $\mathscr{L}'$ that is valid up to $\approx$. For by symmetry, using Proposition 12.5, $\mathscr{T}$ is the only candidate for such a translation. Suppose $\mathbf{R}$ is a semantic translation w.r.t. which $\mathscr{T}$ is correct. Then $1 \mathbf{R} a$ or $2 \mathbf{R} a$. Suppose $1 \mathbf{R} a$. Take valuations ρ, η with $\eta \mathbf{R} \rho$, such

that $\rho(X_0) = a$ and $\eta(X_0) = 1$. As $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$, one has

$$2 = f^{\mathbf{V}'}(1) = [\![f(X_0)]\!]_{\mathscr{L}'}(\eta) = [\![\mathscr{T}(\mathbf{0})]\!]_{\mathscr{L}'}(\eta)\ \mathbf{R}\ [\![\mathbf{0}]\!]_{\mathscr{L}}(\rho) = a.$$

So one must have $2\ \mathbf{R}\ a$. Now take valuations ρ, η with $\eta\ \mathbf{R}\ \rho$, such that $\rho(X_0) = a$ and $\eta(X_0) = 2$. As $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$, one has

$$3 = f^{\mathbf{V}'}(2) = [\![f(X_0)]\!]_{\mathscr{L}'}(\eta) = [\![\mathscr{T}(\mathbf{0})]\!]_{\mathscr{L}'}(\eta)\ \mathbf{R}\ [\![\mathbf{0}]\!]_{\mathscr{L}}(\rho) = a.$$

This contradicts the requirement that $\mathbf{R} \subseteq \approx$.

Theorem 11.20 does not extend to this translation. Note that $\mathscr{L}$ is a closed-term language. Moreover, $\stackrel{\alpha}{=}_{\mathscr{L}}$ and $\stackrel{\alpha}{=}_{\mathscr{L}'}$ are the identity relations. The only set $\mathbf{W} \subseteq \mathbf{V}'$ that is closed under $\mathscr{T}(\mathscr{L})$ is $\mathbf{V}'$ itself. Any congruence $\sim$ for $\mathscr{L}'$ contained in $\approx$ must distinguish all four semantic values. Suppose $\mathscr{T}$ would be B-valid up to an equivalence $\sim_{\mathscr{T}}$ on $\mathbf{V} \cup \mathbf{V}'$, contained in $\sim$, that on $\mathbf{V}'$ coincides with $\sim$. Let $\mathbf{T}'$ be a semantic counterpart of $\mathscr{T}$. Then $\mathbf{T}'(a) = 1$ or $\mathbf{T}'(a) = 2$. Suppose $\mathbf{T}'(a) = 1$. Take a valuation ρ with $\rho(X_0) = a$. Then

$$2 = f^{\mathbf{V}'}(1) = [\![f(X_0)]\!]_{\mathscr{L}'}(\mathbf{T}' \circ \rho) = [\![\mathscr{T}(\mathbf{0})]\!]_{\mathscr{L}'}(\mathbf{T}' \circ \rho) \sim_{\mathscr{T}} [\![\mathbf{0}]\!]_{\mathscr{L}}(\rho) = a.$$

So one has $2 \sim_{\mathscr{T}} a \sim_{\mathscr{T}} 1$, a contradiction. The case $\mathbf{T}'(a) = 2$ leads to a contradiction in the same way. Hence $\mathscr{T}$ is not B-valid up to such an equivalence $\sim_{\mathscr{T}}$.

12.4 Correctness

A generalisation of the concept of D^--validity to arbitrary languages has been studied in [42, Section 5] under the name *correctness*. With the same motivation as in the beginning of Section 12.2, [42] requires a correct translation to satisfy (3). With this caveat, the generalisation of the notion of D^--validity from Section 7 is straightforward.

Definition 12.8 ([42]) A translation $\mathscr{T}: \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ from $\mathscr{L}$ into $\mathscr{L}'$ is *correct up to* $\leftrightsquigarrow$ iff (3) holds and $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \leftrightsquigarrow [\![E]\!]_{\mathscr{L}}(\rho)$ 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 $\eta \leftrightsquigarrow \rho$.

In the special case of closed-term languages, (3) is equivalent to the requirement in Section 7.1 that there exists at least one pair $\varsigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ and $\eta : \mathscr{X} \to \mathrm{T}_{\mathscr{L}'}$ with $\eta \leftrightsquigarrow \varsigma$.

Observation 12.9 A translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$ is correct up to $\leftrightsquigarrow$ iff the restriction $\mathbf{R}$ of $\leftrightsquigarrow$ to $\mathbf{V}' \times \mathbf{V}$ is a semantic translation, and $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$.

Here the requirement that $\mathbf{R}$ is a semantic translation is equivalent to (3). Following [42], I will study the notion of correctness up to $\leftrightsquigarrow$ only for the case that $\leftrightsquigarrow$ is an equivalence relation.

Proposition 12.10 ([42]) If a correct translation up to an equivalence $\sim$ 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 $\rho_1, \rho_2 : \mathscr{X} \to \mathbf{V}$ with $\rho_1 \sim \rho_2$. By (3) there is a valuation $\eta : \mathscr{X} \to \mathbf{V}'$ with $\eta \sim \rho_1$. So $[\![E]\!]_{\mathscr{L}}(\rho_1) \sim [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \sim [\![E]\!]_{\mathscr{L}}(\rho_2)$. □

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 $\mathbf{U} := \{v \in \mathbf{V}' \mid \exists v \in \mathbf{V}.\ v' \sim v\}$. In [42] this called a *congruence for* $\mathscr{T}(\mathscr{L})$.

Definition 12.11 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 $\mathbf{V}'$ is a *congruence for* $\mathscr{T}(\mathscr{L})$ if $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta_1) \sim [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta_2)$ for any $E \in \mathbb{T}_{\mathscr{L}}$ and $\eta_1, \eta_2 : \mathscr{X} \to \mathbf{U}$ with $\eta_1 \sim \eta_2$.

This is the same as a congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$—see Definition 11.15—with $\mathbf{W} := \mathbf{U}$.

Proposition 12.12 If $\mathscr{T}$ is correct up to $\sim$ then $\mathbf{U}$ is closed under $\mathscr{T}(\mathscr{L})$.

Proof: This follows from Proposition 11.16 by taking $\mathbf{R}$ as in Observation 12.9, so that $\mathbf{U} := \mathbf{R}(\mathbf{V})$. □

Proposition 12.13 ([42]) If $\mathscr{T}$ is a correct translation up to an equivalence $\sim$ from $\mathscr{L}$ into $\mathscr{L}'$, then $\sim$ is a congruence for $\mathscr{T}(\mathscr{L})$.

Proof: A simply application of Theorem 11.19, taking $\mathbf{R}$ as above. □

For general translations $\mathscr{T}$ that are correct w.r.t. a semantic translation $\mathbf{R} \subseteq \sim$, the property of $\sim$ being a congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{R}(\mathbf{V})$ is weaker than that of being a congruence for $\mathscr{T}(\mathscr{L})$—cf. Definition 12.11—for it proceeds by restricting attention to valuations into $\mathbf{R}(\mathbf{V}) \subseteq \mathbf{U}$. Theorem 11.19 does not generalise to larger sets $\mathbf{W} \supset \mathbf{R}(\mathbf{V})$. This will be illustrated by Example 12.15 below.

The following theorem tells that the notion of validity proposed in this paper can be seen as a generalisation of the notion of correctness from [42] that applies to equivalences (and preorders) $\backsim$ that are not necessarily congruences for $\mathscr{L}$ or $\mathscr{T}(\mathscr{L})$.

Theorem 12.14 A translation $\mathscr{T}$ from $\mathscr{L}$ into $\mathscr{L}'$ is correct up to a semantic equivalence $\sim$ iff it is valid up to $\sim$ and $\sim$ is a congruence for $\mathscr{T}(\mathscr{L})$.

Proof: By Observation 12.9 and Definition 11.3 any correct translation up to $\sim$ is surely valid up to $\sim$.

Suppose $\mathscr{T}$ is valid up to $\sim$ and $\sim$ is a congruence for $\mathscr{T}(\mathscr{L})$. Then there is a semantic translation $\mathbf{R} \subseteq \mathbf{V}' \times \mathbf{V}$ such that $\mathbf{R} \subseteq \sim$ and $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$. 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 $\eta \sim \rho$. Let $\theta : \mathscr{X} \to \mathbf{V}'$ be a valuation with $\theta\ \mathbf{R}\ \rho$—it exists since $\mathbf{R}$ is a semantic translation. Now $\theta \sim \rho \sim \eta$, using that $\mathbf{R} \subseteq \sim$, so $\theta, \eta : \mathscr{X} \to \mathbf{U}$ and $\theta \sim \eta$. Hence $[\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\eta) \sim [\![\mathscr{T}(E)]\!]_{\mathscr{L}'}(\theta) \sim [\![E]\!]_{\mathscr{L}}(\rho)$, using that $\sim$ is a congruence for $\mathscr{T}(\mathscr{L})$ and that $\mathscr{T}$ is correct w.r.t. $\mathbf{R}$. □

In may be tempting to conjecture that the above theorem can be reformulated or sharpened by requiring $\sim$ to be a congruence for $\mathscr{L}$, instead of $\mathscr{T}(\mathscr{L})$. The following example shows that such a sharpening of Theorem 12.14 does not hold.

Example 12.15 Let $\mathscr{L}$ be a language with two constants **Yes** and **No** and a unary operator $\neg$. Its semantics is given by $\mathbf{V} = \{0,1\}$, $\mathbf{Yes}^{\mathbf{V}} = 1$, $\mathbf{No}^{\mathbf{V}} = 0$ and $\neg^{\mathbf{V}}(b) = 1-b$. Let $\mathscr{L}'$ be the extension of $\mathscr{L}$ with the constant $\top$ and the semantic value $\top$—so $\mathbf{V}' = \{0,1,\top\}$—with $\top^{\mathbf{V}'} = \top$ and $\neg^{\mathbf{V}'}(\top) = \top$. Let $\sim$ be the semantic equivalence on $\mathbf{V}' \supseteq \mathbf{V}$ given by $0 \not\sim 1 \sim \top$. Then the identity function $\mathscr{I}$ is a translation from $\mathscr{L}$ into $\mathscr{L}'$ that is valid up to $\sim$. This is witnessed by the semantic translation $\mathbf{R} := \{(0,0),(1,1)\} \subseteq \sim$. The relation $\sim$ is a congruence for $\mathscr{L}$. In line with Theorem 11.19 it also is a congruence for $\mathscr{I}(\mathscr{L})$ on $\mathbf{R}(\mathbf{V}) = \{0,1\}$. However it fails to be a congruence for $\mathscr{I}(\mathscr{L})$ (on $\mathbf{U} = \{0,1,\top\}$). So by Proposition 12.13 the translation $\mathscr{I}$ is not correct up to $\sim$. Indeed, for $\rho : \mathscr{X} \to \mathbf{V}$ a valuation that sends $X \in \mathscr{X}$ to 1 and $\eta : \mathscr{X} \to \mathbf{V}'$ a valuation that sends X to $\top$, assuming $\rho(Y) = \eta(Y)$ for other variables Y, one has $\rho \sim \eta$, yet $[\![\mathscr{I}(\neg X)]\!]_{\mathscr{L}'}(\eta) = [\![\neg X]\!]_{\mathscr{L}'}(\eta) = \neg^{\mathbf{V}'}(\eta(X)) = \top \not\sim 0 = \neg^{\mathbf{V}}(\rho(X)) = [\![\neg X]\!]_{\mathscr{L}}(\rho)$.

The following result presents a condition under which the claim of Theorem 11.26 pertains to correctness instead of merely validity.

Theorem 12.16 Let $\mathscr{T}$ be a translation from a language $\mathscr{L}$, with semantic domain $\mathbf{V}$, into a language $\mathscr{L}'$, with domain $\mathbf{V}'$, valid up to a semantic equivalence $\approx$, and suppose the congruence closure $\approx^{\mathrm{l}}_{\mathscr{L}}$ of $\approx$ w.r.t. $\mathscr{L}$ is in fact a congruence. Then $\mathscr{T}$ is correct up to the equivalence $\approx^{\mathrm{lc}}_{\mathscr{T}}$ described in Theorem 11.26.

Proof: By assumption $\mathscr{T}$ is correct w.r.t. a semantic translation $\mathbf{R}$. The proof of Theorem 11.26 builds a relation $\approx^{1c}_{\mathscr{L},\mathbf{R}}$ out of $\approx$, $\mathscr{L}$ and $\mathbf{R}$ that is shown to have the properties required of $\approx^{1c}_{\mathscr{T}}$. One has $\mathbf{R} \subseteq \approx^{1c}_{\mathscr{L},\mathbf{R}}$, the restriction of $\approx^{1c}_{\mathscr{L},\mathbf{R}}$ to $\mathbf{V}$ equals $\approx^{1}_{\mathscr{L}}$, and $\mathbf{R}(\mathbf{V}) = \mathbf{W}$. Using that $\approx^{1}_{\mathscr{L}}$ is a congruence for $\mathscr{L}$, by Theorem 11.19 $\approx^{1c}_{\mathscr{L},\mathbf{R}}$ is a congruence for $\mathscr{T}(\mathscr{L})$ on $\mathbf{R}(\mathbf{V})$. Since $\mathbf{R}(\mathbf{V}) = \mathbf{W}$, which for $\approx^{1c}_{\mathscr{L},\mathbf{R}}$ is the $\mathbf{U}$ used in Definition 12.11, $\approx^{1c}_{\mathscr{L},\mathbf{R}}$ is a congruence for $\mathscr{T}(\mathscr{L})$. So by Theorem 12.14, $\mathscr{T}$ is correct up to $\approx^{1c}_{\mathscr{T}} := \approx^{1c}_{\mathscr{L},\mathbf{R}}$. □

13 Validity up to barbed bisimilarity for process calculi

In this paper I defined a notion of a valid translation up to a semantic equivalence $\sim$ between general system description languages, dealing in Section 10 with the special case of closed-term languages. In this section I zoom in further on process calculi such as CCS [70] and the π-calculus [101], and ask which equivalences $\sim$ to use in studying their relative expressiveness. These languages have in common that their semantics can be given in terms of labelled transition systems, whose states, called *processes*, are the closed expressions in the language, and whose transitions are given by an operational semantics in the style of Plotkin [95].

13.1 Validity up to strong bisimilarity or strong early bisimilarity

For a wide class of process calculi without name binding, the finest equivalence in regular employ is strong bisimilarity [70]. In proving that process calculus $\mathscr{L}'$ is at least as expressive as process calculus $\mathscr{L}$ up to a semantic equivalence $\sim$, the choice of strong bisimilarity for $\sim$ thus yields the strongest result. Accordingly, Robert de Simone [106, 105] showed that a wide class of process calculi, including CCS [70], CSP [15], ACP [10] and SCCS [68], are expressible up to strong bisimilarity in MEIJE [3].

However, strong bisimilarity is not suitable for comparing any two process calculi. As an example, consider the π-calculus with the early operational semantics versus the π-calculus with the late operational semantics [101]. Both operational semantics are meant to convey the same idea on how π-calculus processes interact. Consequently, one would hope and expect that the identity function between these versions of the π-calculus is a valid translation. However, it is not valid up to strong bisimilarity. The late semantics has transitions labelled $x(y)$, where the name y is a variable in which a value received on channel x will be stored, whereas the early semantics has transitions labelled xy, where the name y is a particular value received on channel x. Since these labels have a different shape, and strong bisimilarity requires matching labels, validity up to strong bisimilarity fails. Moreover, the problem cannot be resolved by simply renaming the labels, for the entire meaning of input transitions is different.

In the π-calculus, *strong early bisimilarity* [101] is a much more canonical semantic equivalence than strong bisimilarity, and the identity function *is* a valid translation between the early and late π-calculi up to strong early bisimilarity. However, the definition of early bisimilarity for the π-calculus with the late operational semantics [101] has a very different form than the definition of early bisimilarity for the early π-calculus [101]. The reason is that the same idea needs to be phrased in terms of rather different transition relations. Consequently, taking strong early bisimilarity as a unifying equivalence for comparing the late and early π-calculus appears to be somewhat ad hoc. Moreover, neither strong bisimilarity nor strong early bisimilarity would be suitable to compare, say, the π-calculus with CCS.

13.2 Validity up to strong barbed bisimilarity

A canonical semantic equivalence that can be defined in a uniform way on CCS as well as on the late

and early π-calculi is *strong barbed bisimilarity*, originally proposed by Milner and Sangiorgi in [74]. It is based on the idea that a τ-transition of a process P describes an actual reaction of the process P, whereas a translation labelled $a \neq \tau$ merely indicates a potential reaction of P when synchronising with a communication partner that is willing to engage in the complementary transition $\bar{a}$. (Likewise, in CSP [15, 79] a transition $P \xrightarrow{a} P'$ can be regarded as a potential; one that is not realised when putting the process in a parallel composition involving synchronisation on the action a when the other component cannot partake in such a synchronisation. The only way to be sure that the CSP action a cannot be inhibited by any synchronisation context is to *conceal* it from the environment, using the CSP concealment operator $\_\backslash a$, renaming the action a into τ.) From this perspective, it appears natural to formulate semantic equivalences entirely in terms of the τ-transitions processes can do, and capture the communication potentials (manifested by the other transitions) merely by studying the behaviour of processes in contexts. Accordingly, *(strong) reduction bisimilarity* was defined in [74] as the version of strong bisimilarity that requires the transfer property for τ-transitions only, i.e., strong barbed bisimilarity as defined below, without the first clause, and reading $\xrightarrow{\tau}$ for $\rightarrow$. Naturally, reduction bisimilarity is not a congruence: the CCS processes $a.\mathbf{0}$ and $\mathbf{0}$ are reduction equivalent (for neither can do a τ-transition), yet $a.\mathbf{0}|\bar{a}.\mathbf{0}$ and $\mathbf{0}|\bar{a}.\mathbf{0}$ are not. The purpose of the relation is to define a reasonable semantic equivalence as its congruence closure. Indeed, for divergence-free CCS processes strong bisimilarity turned out to be the congruence closure of reduction bisimilarity [74]. Unfortunately, this failed for processes with divergences (infinite-sequences of τ-transitions) [74], and when lifted to the weak case, weak reduction congruence turned out to be the universal relation, and thus useless [74]. For this reason, reduction bisimilarity needed to be strengthened. The main idea is that processes like $a.\mathbf{0}$ and $\mathbf{0}$ can be told apart by placing them in a context $\_|\bar{a}.\omega$, where ω denotes a success state that might be reached by the environment in which a process is placed. The process $a.\mathbf{0}|\bar{a}.\omega$ can reach this success state by performing a τ-transition, whereas $\mathbf{0}|\bar{a}.\omega$ can not. Writing $P{\downarrow_\omega}$ to indicate that a process P has reached this success state, reduction bisimilarity can be strengthened by requiring that if $P{\downarrow_\omega}$ then any process Q equivalent to P should also satisfy $Q{\downarrow_\omega}$. This yields the notion of strong barbed bisimilarity. The predicate $\_{\downarrow_\omega}$ is called a *barb*. In general, one can use a collection of barbs.

Barbed bisimilarity is defined on closed-term languages $\mathscr{L}$ that are equipped with a binary *reduction relation* $\rightarrow \subseteq \mathrm{T}_{\mathscr{L}} \times \mathrm{T}_{\mathscr{L}}$ between processes and with a collection $\{{\downarrow_\omega} \subseteq \mathrm{T}_{\mathscr{L}} \mid \omega \in \Omega\}$ of unary predicates on processes, called *barbs*.

Definition 13.1 *Strong barbed bisimilarity* is the largest symmetric relation $\stackrel{\bullet}{\sim} \subseteq \mathrm{T}_{\mathscr{L}} \times \mathrm{T}_{\mathscr{L}}$ such that

- $P \stackrel{\bullet}{\sim} Q$ and $P{\downarrow_\omega}$ with $\omega \in \Omega$ implies $Q{\downarrow_\omega}$, and
- $P \stackrel{\bullet}{\sim} Q$ and $P \rightarrow P'$ implies $Q \rightarrow Q'$ for some Q' with $P' \stackrel{\bullet}{\sim} Q'$.

Strong barbed congruence, $\stackrel{\bullet}{\sim}{}^{1c}_{\mathscr{L}}$, is the congruence closure of $\stackrel{\bullet}{\sim}$ w.r.t. the language $\mathscr{L}$.

For process calculi equipped with a labelled transition system semantics, the reductions $P \rightarrow P'$ are simply the τ-transitions. However, for many languages it is possible to give a *reduction semantics* that generates the reductions directly, without first constricting a labelled transition system. Some languages, such as the λ-calculus, come with a natural reduction semantics, even when they have no labelled transition semantics at all.

13.3 On the choice of a collection of barbs

The definition of the barbs on various process calculi is somewhat ad hoc. In [74] only one barb was used, and a process has this barb when it can perform an observable action, i.e., $P{\downarrow}$ iff $P \xrightarrow{a} P'$ for some

process P' and action $a \neq \tau$. An alternative is to introduce a barb a for each action $a \neq \tau$, taking $P{\downarrow_a}$ iff $P \xrightarrow{a} P'$ for some P'. For CCS, both forms of barbs lead to the same notion of strong barbed congruence.

In the π-calculus, x and $\bar{x}$ for $x \in \mathcal{N}$ are taken to be barbs (or just $\bar{x}$), with $P{\downarrow_\omega}$ iff $P \xrightarrow{\omega(y)} P'$ or $P \xrightarrow{\omega y} P'$ for some P'—cf. Section 4. Here, it makes no difference in the resulting notion of strong barbed congruence whether one takes only barbs $\bar{x}$, or also barbs $x \in \mathcal{N}$.

In fact, it does not matter at all how the barbs are defined, but only whether there are enough barbs, and how they propagate upwards through a context (that is, how the validity of $C[P]{\downarrow_\omega}$ is determined by the validity of $P{\downarrow_\omega}$). This is because in determining whether two processes P and Q are strongly barbed congruent, the barbs that help in this determination appear in contexts in which the processes P and Q are placed, rather than in P and Q themselves. For this reason, aiming for a fairly language-independent definition of barbed congruence, I propose the use of *external barbs* instead, defined as follows. Postulate a sufficiently large collection Ω of barbs and add each of them as a fresh constant to the language under consideration. Require them to propagate upwards through a context in the same way as τ-transitions or reduction steps. Thus, whenever the structural operational semantics of the language has a rule

$$\frac{P_i \to Q}{f(P_1,\ldots,P_n) \to R}$$

for some n-ary operator f, possibly with $\xrightarrow{\tau}$ in the rôle of $\to$, then postulate a rule

$$\frac{P_i{\downarrow_\omega}}{f(P_1,\ldots,P_n){\downarrow_\omega}}$$

for each barb $\omega \in \Omega$. Note that this agrees exactly with Definition 4.3 of (strong) output barbs in the π-calculus. It turns out that for CCS, the π-calculus, and in fact any language with a definition of strong barbed congruence that I am aware of, the above use of external barbs yields the same notion of strong barbed congruence as the original approach. It may be felt as a drawback that in defining strong barbed congruence on a language $\mathscr{L}$, $\mathscr{L}$ needs to be extended, namely by the added constants $\omega \in \Omega$. However, as a tool to define strong barbed congruence on $\mathscr{L}$ this is not a big problem, as the definition on the extended language naturally restricts to $\mathscr{L}$.

The treatment of barbs proposed above is strongly inspired by the success action ω in the treatment of *testing equivalences* by De Nicola & Hennessy [25], and by the criterion of *success sensitiveness* imposed by Gorla [53] on translations between process calculi. Both [25] and [53] use only a single success predicate (barb). In [53] $P{\downarrow_\omega}$ is defined to hold iff P has a "top-level unguarded occurrence" of ω. Gorla defines the latter concept only for languages that are equipped with a notion of *structural congruence* $\equiv$ as well as a parallel composition $|$. In that case P has a top-level unguarded occurrence of ω iff $P \equiv Q|\omega$, for some Q [53]. Specialised to the π-calculus, a *(top-level) unguarded* occurrence is one that does not lay strictly within a subterm $\alpha.Q$, where α is τ, $\bar{x}y$ or $x(z)$. As far as I know, for all languages where Gorla's definition of $P{\downarrow_\omega}$ as well as mine apply, they yield the same result.

On CCS strong barbed congruence coincides with strong bisimilarity [74]. On the π-calculus strong barbed congruence coincides with strong early congruence, the congruence closure of strong early bisimilarity [101]. This testifies to the success of the barbed bisimilarity approach to defining canonical semantic equivalences for different process calculi in a uniform way. For many other calculi, strong barbed congruence is the primary *proposal* for a canonical semantic equivalence, and its explicit characterisation in a manner that does not involve quantification over all context is a task that merely follows.

This makes strong barbed bisimilarity an attractive candidate for the equivalence relation $\sim$ up to which translations between process calculi are proven valid. Theorem 11.26 says that if a translation

between two process calculi is valid up to strong barbed bisimilarity, then it is also valid up an equivalence that on the source language coincides with strong barbed congruence, and on the image of the source language within the target language is also the congruence closure of strong barbed bisimilarity.

13.4 Validity up to weak barbed bisimilarity

For "nonprompt encodings" [78], that "allow administrative (or book-keeping) steps to precede a committing step", strong (barbed) bisimilarity is too fine an equivalence to relate source processes and their translations. Here an appealing alternative is *weak barbed bisimilarity*, cf. Definition 4.4. On CCS weak barbed congruence coincides with weak bisimilarity as defined in [70]. On the π-calculus weak barbed congruence coincides with weak early congruence as used in [101], at least for image-finite processes, or in general when allowing infinite sums in the π-calculus [100]. Section 4 mentioned Boudol's encoding of $m\pi$ into $a\pi$ as an example of a translation valid up to weak barbed bisimilarity.

13.5 Validity up to divergence-preserving weak barbed bisimilarity

In many situations one would reject translations between process calculi, that introduce (or eliminate) divergences. This can be captured in my framework by requiring validity up to *divergence-preserving weak (barbed) bisimilarity*, also know as weak (barbed) bisimilarity *with explicit divergence* [11]. It adds to Definition 4.4 the clause

- $P \stackrel{\bullet}{\approx} Q$ and $P\Uparrow$ implies $Q\Uparrow$.

Here $P\Uparrow$ denotes that there are P_i for $i \geq 0$ such that $P = P_0$ and $P_i \rightarrow P_{i+1}$ (or $P_i \stackrel{\tau}{\longrightarrow} P_{i+1}$) for all $i \geq 0$. Boudol's encoding of $m\pi$ into $a\pi$ is even valid up to divergence-preserving weak barbed bisimilarity [49].

13.6 Validity up to divergence-preserving branching barbed bisimilarity

In most industrial applications of process calculi the rôle once played by weak bisimilarity has largely been taken over by branching bisimilarity [51]—cf. [32, 23]—of which both a default version and two divergence-preserving versions are in use [50, 30]. The following proposal captures these through the barbed bisimilarity methodology:

Definition 13.2 *Divergence-preserving branching barbed bisimilarity* is the largest symmetric relation $\stackrel{\bullet}{\leftrightarrows}{}^{\Delta}_{b} \subseteq \mathrm{T}_{\mathscr{L}} \times \mathrm{T}_{\mathscr{L}}$ such that

- $P \stackrel{\bullet}{\leftrightarrows}{}^{\Delta}_{b} Q$ and $P{\downarrow_\omega}$ with $\omega \in \Omega$ implies $Q \Longrightarrow Q^\dagger$ for some $Q^\dagger$ with $P \stackrel{\bullet}{\leftrightarrows}{}^{\Delta}_{b} Q^\dagger$ and $Q^\dagger{\downarrow_\omega}$,
- $P \stackrel{\bullet}{\leftrightarrows}{}^{\Delta}_{b} Q$ and $P \rightarrow P'$ implies $Q \Longrightarrow Q^\dagger$ $(\rightarrow)$ Q' for some $Q^\dagger, Q'$ with $P \stackrel{\bullet}{\leftrightarrows}{}^{\Delta}_{b} Q^\dagger$ and $P' \stackrel{\bullet}{\leftrightarrows}{}^{\Delta}_{b} Q'$, and
- $P \stackrel{\bullet}{\leftrightarrows}{}^{\Delta}_{b} Q$ and $P \rightarrow P_1 \rightarrow P_2 \rightarrow \ldots$ implies $Q \rightarrow Q'$ for some Q' with $P_k \stackrel{\bullet}{\leftrightarrows}{}^{\Delta}_{b} Q'$ for some $k \geq 0$.

Here P $(\rightarrow)$ Q abbreviates $(P = Q) \vee (P \rightarrow Q)$. See [50] for a number of equivalent versions of the last clause, i.e. with $Q \Longrightarrow\!\!\rightarrow Q'$. For *weakly divergence-preserving branching barbed bisimilarity* the last clause is weakened to "$P \stackrel{\bullet}{\leftrightarrows}{}^{\Delta}_{b} Q$ and $P\Uparrow$ implies $Q\Uparrow$", just as for divergence-preserving weak barbed bisimilarity above. Skipping the last clause altogether yields *branching barbed bisimilarity*. On CCS, the congruence closures of these equivalences yield *((weakly) divergence-preserving) branching bisimilarity*, as defined in the literature [30]; the proofs are not essentially different from the ones for strong and weak barbed bisimilarity. Boudol's encoding of $m\pi$ into $a\pi$ is even valid up to divergence-preserving branching barbed bisimilarity [49].

13.7 Example: the Honda-Tokoro encoding

In Section 4 I recalled an encoding by Boudol [14] of the synchronous fragment $m\pi$ of the π-calculus into the asynchronous fragment $a\pi$ of $m\pi$. As shown in [49], it is valid up to weak barbed bisimilarity—and even up to divergence-preserving branching barbed bisimilarity—when taking as barbs all the output barbs $\downarrow_{\bar{x}}$ (Definition 4.3), or also the input barbs $\downarrow_x$. A similar encoding of $m\pi$ into $m\pi$ was proposed by Honda & Tokoro in [61]. It differs from Boudol's encoding only in the clauses for the input and output prefix:

$$\begin{array}{rcll} \mathscr{T}_{\rm HT}(\bar{x}z.P) & := & x(u).(\bar{u}z|\mathscr{T}_{\rm HT}(P)) & \text{choosing } u \notin \mathsf{n}(P)\cup\{x,z\} \\ \mathscr{T}_{\rm HT}(x(y).P) & := & (\nu u)(\bar{x}u|u(y).\mathscr{T}_{\rm HT}(P)) & \text{choosing } u \notin \mathsf{n}(P))\cup\{x\}. \end{array}$$

While intuitively equally plausible as Boudol's encoding, $\mathscr{T}_{\rm HT}$ is slightly more concise, at the price of swapping input and output actions. Interestingly, the encoding $\mathscr{T}_{\rm HT}$ is *not* valid up to $\dot{\approx}$ as defined in Section 4: one has $(\bar{x}z.P)\downarrow_{\bar{x}}$, yet $\mathscr{T}_{\rm HT}(\bar{x}z.P)\Downarrow_x$ but $\mathscr{T}_{\rm HT}(\bar{x}z.P)\not\Downarrow_{\bar{x}}$. This shows that $\mathscr{T}_{\rm HT}$ does not meet the first condition of Definition 4.4.

In [49] a weaker form of weak barbed bisimilarity was introduced that makes $\mathscr{T}_{\rm HT}$ into a valid encoding. Instead of using all barbs $\downarrow_{\bar{x}}$ and $\downarrow_x$, or only the output barbs $\downarrow_{\bar{x}}$, it employs so-called *channel barbs*. A process is said to have the channel barb $x\in\mathcal{N}$ iff it either has the barb $\bar{x}$ or x. While this solves the problem at hand, it feels a bit contra-intuitive that a new kind of barb needs to be introduced just to confirm the intuitive validity of the encoding $\mathscr{T}_{\rm HT}$. A more fundamental solution to this problem is the use of external barbs as advocated in Section 13.3. When defining barbed bisimilarities in terms of external barbs, $\mathscr{T}_{\rm HT}$ is in fact valid up to weak barbed bisimilarity—and even up to divergence-preserving branching barbed bisimilarity.

14 Applications

14.1 CSP

The language CSP [15, 16, 60] is parametrised by a set Σ of *communications*. Here I use the core of CSP studied under the name *Communicated Processes* in [79]. It is given by the following grammar.

$$E,F ::= \mathbf{div} \mid \mathbf{stop} \mid a{\rightarrow}E \mid E\sqcap F \mid E\,\Box\, F \mid E\|_A F \mid E\backslash A \mid f(E) \mid X \mid \mu X.E$$

Here E and F are CSP expressions, $a\in\Sigma$, $A\subseteq\Sigma$, $f:\Sigma\to\Sigma$ and $X\in\mathscr{X}$. The operators in the above grammar are *divergence*, *inaction*, *action prefixing*, *internal* and *external choice*, *parallel composition*, *concealment* and *renaming*. The operators **stop**, $a\rightarrow$, $\sqcap$, $\Box$, $\backslash A$, $f(\_)$ and the recursion construct $\mu X.E$ stem from [15], and **div** and $\|_A$ from [79]. The parallel composition $\|_A$ requires its two arguments to synchronise over the actions in A; it unifies the operators $\|$ and $|||$ from [15, 16], which are the special cases $\|_\Sigma$ and $\|_\emptyset$ of $\|_A$, respectively. The constant **div** can be seen as syntactic sugar for $\mu X.X$. Compared to [16], the above grammar leaves out sequential composition $E;F$ and the constant **skip**.

The domain $\mathbf{V}_{\rm CSP}$ in which this language is interpreted consists of pairs $\langle\mathcal{D},\mathcal{F}\rangle$ of divergences $\mathcal{D}\subseteq\Sigma^*$ and *failure sets* $\mathcal{F}\subseteq\Sigma^*\times\mathscr{P}(\Sigma)$, satisfying some closure properties [16, 79]. Intuitively, $s\in\mathcal{D}$ indicates that the process $\langle\mathcal{D},\mathcal{F}\rangle$ can perform the sequence of observable actions s, after which it engages in an infinite sequence of unobservable actions; $\langle s,\mathcal{Y}\rangle\in\mathcal{F}$ indicates that the process $\langle\mathcal{D},\mathcal{F}\rangle$ can perform the sequence of observable actions s, after which it reaches a state of deadlock, by not engaging in any further actions, even though the environment in which it is running allows it to continue with the actions from $\mathcal{Y}\subseteq\Sigma$. After a divergence s no further information is observable, which semantically is achieved by saturating $\mathcal{D}$ with all possible divergences st and $\mathcal{F}$ with all failure pairs $\langle st,\mathcal{Y}\rangle$.

The semantics $[\![\ ]\!]_{\mathrm{CSP}} : \mathbb{T}_{\mathrm{CSP}} \to ((\mathscr{X} \to \mathbf{V}_{\mathrm{CSP}}) \to \mathbf{V}_{\mathrm{CSP}})$ is defined inductively below. For $E \in \mathbb{T}_{\mathrm{CSP}}$ a CSP expression and $\rho : \mathscr{X} \to \mathbf{V}_{\mathrm{CSP}}$ a valuation of the variables, $[\![E]\!]_{\mathrm{CSP}}(\rho)$ is a pair, of which, following [16], the two components are denoted $\mathscr{D}[\![E]\!]\rho$ and $\mathscr{F}[\![E]\!]\rho$. The following definition stems from [16].

$$
\begin{array}{ll|ll}
\mathscr{D}[\![\mathbf{div}]\!]\rho & = \Sigma^* & \mathscr{F}[\![\mathbf{div}]\!]\rho & = \Sigma^* \times \mathscr{P}(\Sigma) \\
\mathscr{D}[\![\mathbf{stop}]\!]\rho & = \emptyset & \mathscr{F}[\![\mathbf{stop}]\!]\rho & = \{\langle \varepsilon, \mathcal{Y}\rangle \mid \mathcal{Y} \subseteq \Sigma\} \\
\mathscr{D}[\![a{\to}E]\!]\rho & = \{as \mid s \in \mathscr{D}[\![E]\!]\rho\} & \mathscr{F}[\![a{\to}E]\!]\rho & = \{\langle \varepsilon, \mathcal{Y}\rangle \mid a \notin \mathcal{Y}\} \cup \{\langle as, \mathcal{Y}\rangle \mid \langle s, \mathcal{Y}\rangle \in \mathscr{F}[\![E]\!]\rho\} \\
\mathscr{D}[\![E \sqcap F]\!]\rho & = \mathscr{D}[\![E]\!]\rho \cup \mathscr{D}[\![F]\!]\rho & \mathscr{F}[\![E \sqcap F]\!]\rho & = \mathscr{F}[\![E]\!]\rho \cup \mathscr{F}[\![F]\!]\rho \\
\mathscr{D}[\![E \,\Box\, F]\!]\rho & = \mathscr{D}[\![E]\!]\rho \cup \mathscr{D}[\![F]\!]\rho & \mathscr{F}[\![E \,\Box\, F]\!]\rho & = \{\langle \varepsilon, \mathcal{Y}\rangle \mid \langle \varepsilon, \mathcal{Y}\rangle \in \mathscr{F}[\![E]\!]\rho \cap \mathscr{F}[\![F]\!]\rho\} \\
 & & & \quad \cup \{\langle s, \mathcal{Y}\rangle \mid s \neq \varepsilon \wedge \langle s, \mathcal{Y}\rangle \in \mathscr{F}[\![E]\!]\rho \cup \mathscr{F}[\![E]\!]\rho\} \\
 & & & \quad \cup \{\langle s, \mathcal{Y}\rangle \mid s \in \mathscr{D}[\![E \,\Box\, F]\!]\rho\}
\end{array}
$$

Here ε denotes the empty sequence. In the same vain there are clauses for $E\|_A F$, $E\backslash A$ and $f(E)$. The meaning $[\![\mu X.E]\!]_{\mathrm{CSP}}(\rho) \in \mathbf{V}_{\mathrm{CSP}}$ of $\mu X.E$ is the least fixed point of the equation $\mathcal{X} = [\![E]\!](\rho[X \mapsto \mathcal{X}])$, where $\rho[X \mapsto \mathcal{X}]$ is the valuation that differs from from ρ only on the input X, where it yields $\mathcal{X} \in \mathbf{V}_{\mathrm{CSP}}$. This least fixed point exists, because the partial order $\sqsubseteq$ on $\mathbf{V}$, given by $\langle \mathcal{D}, \mathcal{F}\rangle \sqsubseteq \langle \mathcal{D}', \mathcal{F}'\rangle$ iff $\mathcal{D}' \subseteq \mathcal{D}$ and $\mathcal{F}' \subseteq \mathcal{F}$ makes $\mathbf{V}_{\mathrm{CSP}}$ into a c.p.o. on which all relevant functions are continuous [16].

In [79], besides the above denotational semantics of CSP, also an operational semantics is provided. It is given by the binary transition relations $\xrightarrow{\alpha}$ between closed CSP expressions P, Q derived by the rules of Table 1. Here a ranges over Σ and α over $\Sigma \uplus \{\tau\}$, and relabelling operators f are extended to

$$
\begin{array}{llll}
\mathbf{div} \xrightarrow{\tau} \mathbf{div} & (a \to P) \xrightarrow{a} P & P \sqcap Q \xrightarrow{\tau} P & P \sqcap Q \xrightarrow{\tau} Q \\[2ex]
\dfrac{P \xrightarrow{a} P'}{P \,\Box\, Q \xrightarrow{a} P'} & \dfrac{P \xrightarrow{\tau} P'}{P \,\Box\, Q \xrightarrow{\tau} P' \,\Box\, Q} & \dfrac{Q \xrightarrow{a} Q'}{P \,\Box\, Q \xrightarrow{a} Q'} & \dfrac{Q \xrightarrow{\tau} Q'}{P \,\Box\, Q \xrightarrow{\tau} P \,\Box\, Q'} \\[2ex]
\dfrac{P \xrightarrow{\alpha} P' \ (\alpha \notin A)}{P\|_A Q \xrightarrow{\alpha} P'\|_A Q} & \dfrac{P \xrightarrow{a} P' \ \ Q \xrightarrow{a} Q' \ (a \in A)}{P\|_A Q \xrightarrow{a} P'\|_A Q'} & & \dfrac{Q \xrightarrow{\alpha} Q' \ (\alpha \notin A)}{P\|_A Q \xrightarrow{\alpha} P\|_A Q'} \\[2ex]
\dfrac{P \xrightarrow{\alpha} P' \ (\alpha \notin A)}{P\backslash A \xrightarrow{\alpha} P'\backslash A} & \dfrac{P \xrightarrow{a} P' \ (a \in A)}{P\backslash A \xrightarrow{\tau} P'\backslash A} & \dfrac{P \xrightarrow{\alpha} P'}{f(P) \xrightarrow{f(\alpha)} f(P')} & \mu X.E \xrightarrow{\tau} E[\mu X.E/X]
\end{array}
$$

Table 1: Structural operational semantics of CSP

$\Sigma \uplus \{\tau\}$ by $f(\tau) = \tau$. The failures and divergences of a CSP process can be extracted from its operational semantics:

Definition 14.1 Write $P \Longrightarrow Q$ if there are processes $P_0, \ldots, P_n$, with $n \geq 0$, such that $P = P_0$, $P_i \xrightarrow{\tau} P_{i+1}$ for all $0 \leq i < n$, and $P_n = Q$. Write $P \xRightarrow{\alpha} Q$ if there are processes P', Q' with $P \Longrightarrow P' \xrightarrow{\alpha} Q' \Longrightarrow Q$. Write $P \xRightarrow{s} Q$, for $s = a_1 a_2 \ldots a_n \in \Sigma^*$ with $n \geq 0$, if there are processes $P_0, \ldots, P_n$ such that $P = P_0$, $P_i \xRightarrow{a_i} P_{i+1}$ for all $0 \leq i < n$, and $P_n = Q$. Let $\mathcal{I}(P) = \{\alpha \in \Sigma \cup \{\tau\} \mid \exists Q.\, P \xrightarrow{\alpha} Q\}$.

$s \in \Sigma^*$ is a *divergence* of a process $P \in \mathrm{T}_{\mathrm{CSP}}$ if there are P_i for all $i \geq 0$ with $P \xRightarrow{s} P_0 \xrightarrow{\tau} P_1 \xrightarrow{\tau} \ldots$. The *divergence set* of P is $\mathscr{D}(P) := \{st \mid s \text{ is a divergence of } P\}$.

A pair $\langle s, \mathcal{Y}\rangle \in \Sigma^* \times \mathscr{P}(\Sigma)$ such that $P \xRightarrow{s} Q$ for some Q with $\mathcal{I}(Q) \cap (\mathcal{Y} \cup \{\tau\}) = \emptyset$ is a *stable failure* of a process P. The *failure set* of P is $\mathscr{F}(P) = \{\langle s, \mathcal{Y}\rangle \mid s \in \mathscr{D}(P) \text{ or } \langle s, \mathcal{Y}\rangle \text{ is a stable failure of } P\}$.

Olderog & Hoare [79] showed that this operational semantics is consistent with the denotational semantics of CSP, in the sense that $[\![P]\!]_{\mathrm{CSP}} = \langle \mathscr{D}(P), \mathscr{F}(P)\rangle$ for all closed CSP expressions P.

In my framework a language comprises syntax as well as semantics, so the operational version of CSP is formally a different language, with the same syntax as CSP, which I call CSP^{op}. I see CSP^{op} as a closed-term language. I now claim that the language CSP is at least as expressive as CSP^{op}. To formalise this I need to pick a semantic equivalence on $\mathbf{V}_{\mathrm{CSP}} \cup \mathrm{T}_{\mathrm{CSP}}$. I take *failures-divergences equivalence* $\equiv_{\mathcal{FD}}$. On $\mathbf{V}_{\mathrm{CSP}}$ this is simply the identity relation, and for $P,Q \in \mathrm{T}_{\mathrm{CSP}}$ write $P \equiv_{\mathcal{FD}} Q$ iff $\mathscr{D}(P) = \mathscr{D}(Q)$ and $\mathscr{F}(P) = \mathscr{F}(Q)$. Moreover, for $\langle \mathcal{D}, \mathcal{F}\rangle \in \mathbf{V}_{\mathrm{CSP}}$ and $P \in \mathrm{T}_{\mathrm{CSP}}$ let $P \equiv_{\mathcal{FD}} \langle \mathcal{D}, \mathcal{F}\rangle$ iff $\mathscr{D}(P) = \mathcal{D}$ and $\mathscr{F}(P) = \mathcal{F}$. I now claim that the identity function $\mathscr{I}$ on $\mathbb{T}_{\mathrm{CSP}}$ is a valid encoding up to $\equiv_{\mathcal{FD}}$ from CSP^{op} into CSP. The semantic translation $\mathbf{R}_{\mathrm{CSP}}$ witnessing this validity is $\equiv_{\mathcal{FD}} \restriction (\mathbf{V}_{\mathrm{CSP}} \times \mathrm{T}_{\mathrm{CSP}})$. The validity of $\mathscr{I}$ is a direct consequence of the consistency result from [79]. To make the two languages equally expressive, the syntax of CSP needs to be sufficiently enriched to ensure that for each semantic object $\langle \mathcal{D}, \mathcal{F}\rangle \in \mathbf{V}_{\mathrm{CSP}}$ there exists a $P \in \mathrm{T}_{\mathrm{CSP}}$ such that $P \equiv_{\mathcal{FD}} \langle \mathcal{D}, \mathcal{F}\rangle$; this makes $\mathbf{R}_{\mathrm{CSP}}^{-1}$ into a semantic translation. As long as there are undenotable objects $\langle \mathcal{D}, \mathcal{F}\rangle \in \mathbf{V}_{\mathrm{CSP}}$, the language CSP is slightly more expressive than CSP^{op}.

14.2 ACP

The process algebra ACP [10, 5, 29] is parametrised by a set A of *actions* and a commutative and associative *communication function* $\gamma : \mathsf{A} \times \mathsf{A} \to \mathsf{A} \uplus \{\delta\}$.[12] Here I consider the extension ACP$_R$ with *relational renaming*. Its syntax is given by the grammar

$$E,F ::= a \mid \delta \mid E + F \mid E \cdot F \mid E \| F \mid E \lfloor\!\lfloor F \mid E | F \mid \partial_H(E) \mid \mathcal{R}(E) \mid X \mid \langle X | \mathcal{S}\rangle$$

where E and F are ACP$_R$ expressions, $a \in \mathsf{A}$, $H \subseteq \mathsf{A}$, $\mathcal{R} \subseteq \mathsf{A} \times \mathsf{A}$ and $X \in \mathscr{X}$. Moreover, $\mathcal{S}$ is a recursive specification, given by a set $V_{\mathcal{S}} \subseteq \mathscr{X}$ of bound variables and an equation $X = \mathcal{S}_X$, with $\mathcal{S}_X$ an ACP$_R$ expression, for each $X \in V_{\mathcal{S}}$; in the *recursive call* $\langle X|\mathcal{S}\rangle$ one requires that $X \in V_{\mathcal{S}}$.

$$\frac{}{a \xrightarrow{a} \surd} \qquad \frac{P \xrightarrow{a} \surd}{P+Q \xrightarrow{a} \surd} \qquad \frac{Q \xrightarrow{a} \surd}{P+Q \xrightarrow{a} \surd} \qquad \frac{P \xrightarrow{a} P'}{P+Q \xrightarrow{a} P'} \qquad \frac{Q \xrightarrow{a} Q'}{P+Q \xrightarrow{a} Q'} \qquad \frac{P \xrightarrow{a} \surd}{P \cdot Q \xrightarrow{a} Q} \qquad \frac{P \xrightarrow{a} P'}{P \cdot Q \xrightarrow{a} P' \cdot Q}$$

$$\frac{P \xrightarrow{a} \surd}{\mathcal{R}(P) \xrightarrow{b} \surd}\ a\mathcal{R}b \qquad \frac{P \xrightarrow{a} P'}{\mathcal{R}(P) \xrightarrow{b} \mathcal{R}(P')}\ a\mathcal{R}b \qquad \frac{P \xrightarrow{a} \surd}{P \| Q \xrightarrow{a} Q} \qquad \frac{Q \xrightarrow{a} \surd}{P \| Q \xrightarrow{a} P} \qquad \frac{P \xrightarrow{a} P'}{P \| Q \xrightarrow{a} P' \| Q} \qquad \frac{Q \xrightarrow{a} Q'}{P \| Q \xrightarrow{a} P \| Q'}$$

$$\frac{P \xrightarrow{a} \surd,\ Q \xrightarrow{b} \surd}{P \| Q \xrightarrow{c} \surd}\ \gamma(a,b)=c \qquad \frac{P \xrightarrow{a} \surd,\ Q \xrightarrow{b} Q'}{P \| Q \xrightarrow{c} Q'}\ \gamma(a,b)=c \qquad \frac{P \xrightarrow{a} P',\ Q \xrightarrow{b} \surd}{P \| Q \xrightarrow{c} P'}\ \gamma(a,b)=c \qquad \frac{P \xrightarrow{a} P',\ Q \xrightarrow{b} Q'}{P \| Q \xrightarrow{c} P' \| Q'}\ \gamma(a,b)=c$$

$$\frac{P \xrightarrow{a} \surd}{\partial_H(P) \xrightarrow{a} \surd}\ a \notin H \qquad \frac{P \xrightarrow{a} P'}{\partial_H(P) \xrightarrow{a} \partial_H(P')}\ a \notin H \qquad \frac{\langle \mathcal{S}_X|\mathcal{S}\rangle \xrightarrow{a} \surd}{\langle X|\mathcal{S}\rangle \xrightarrow{a} \surd} \qquad \frac{\langle \mathcal{S}_X|\mathcal{S}\rangle \xrightarrow{a} P'}{\langle X|\mathcal{S}\rangle \xrightarrow{a} P'}$$

Table 2: Structural operational semantics of ACP$_R$ (leaving out the rules for $\lfloor\!\lfloor$ and $|$)

[12] If $\gamma(a,b) = c \in \mathsf{A}$, two parallel components may synchronise when executing the actions a and b respectively, and the result of that synchronisation is called c. If $\gamma(a,b) = \delta$, the actions a and b cannot synchronise. In judging associativity, γ is extended to type $(\mathsf{A} \uplus \{\delta\}) \times (\mathsf{A} \uplus \{\delta\}) \to (\mathsf{A} \uplus \{\delta\})$ by defining $\gamma(\delta, a) = \gamma(a, \delta) = \delta$.

The operational semantics of ACP_R is given by the unary and binary transition relations $\xrightarrow{a}\!\!\surd$ and $\xrightarrow{a}$ on $\mathrm{T}_{\mathrm{ACP}}$ derivable by the rules of Table 2, where $a,b,c \in \mathsf{A}$, P,P',Q,Q' are closed ACP expressions, and $\langle \mathcal{S}_X|\mathcal{S}\rangle$ denotes the term $\mathcal{S}_X[\sigma]$, where $\mathcal{S}_X$ is the right-hand side of the equation for X in the recursive specification $\mathcal{S}$, and σ is the substitution that sends Y to the recursive call $\langle Y|\mathcal{S}\rangle$ for each $Y \in V_{\mathcal{S}}$, and satisfies $\sigma(Z) = Z$ for $Z \notin V_{\mathcal{S}}$. Intuitively, $P \xrightarrow{a}\!\!\surd$ says that process P can perform the action a and then terminate successfully. It might seem unfortunate that in Table 2 most rules come in two versions, one with a process on the right-hand side, and one with the termination symbol $\surd$. Two dialects of ACP_R have been proposed that avoid this: $\mathrm{ACP}_R^\varepsilon$ [111] does so by adding a constant ε to the syntax, indicating successful termination, and aprACP_R [4, 39] by dropping successful termination altogether. In aprACP_R the action constants a and the sequential composition $\cdot$ have been replaced by a family of unary action-prefixing operators $a._-$, one for each $a \in \mathsf{A}$.

$$\frac{}{a.P \xrightarrow{a} P} \qquad \frac{P \xrightarrow{a} P'}{P+Q \xrightarrow{a} P'} \qquad \frac{Q \xrightarrow{a} Q'}{P+Q \xrightarrow{a} Q'} \qquad \frac{P \xrightarrow{a} P'}{\mathcal{R}(P) \xrightarrow{b} \mathcal{R}(P')}\ {\scriptstyle a\mathcal{R}b} \qquad \frac{P \xrightarrow{a} P'}{f \bullet P \xrightarrow{f(a)} P'}$$

$$\frac{P \xrightarrow{a} P'}{P \parallel Q \xrightarrow{a} P' \parallel Q} \qquad \frac{Q \xrightarrow{a} Q'}{P \parallel Q \xrightarrow{a} P \parallel Q'} \qquad \frac{P \xrightarrow{a} P',\ Q \xrightarrow{b} Q'}{P \parallel Q \xrightarrow{c} P' \parallel Q'}\ {\scriptstyle \gamma(a,b)=c} \qquad \frac{P \xrightarrow{a} P'}{\partial_H(P) \xrightarrow{a} \partial_H(P')}\ {\scriptstyle a\notin H} \qquad \frac{\langle \mathcal{S}_X|\mathcal{S}\rangle \xrightarrow{a} P'}{\langle X|\mathcal{S}\rangle \xrightarrow{a} P'}$$

Table 3: Structural operational semantics of aprACP_R^{trig} (leaving out the rules for $\between$ and $|$)[13]

Both languages have the same parameters A and γ as ACP_R, for $\mathrm{ACP}_R^\varepsilon$ with the extra condition that $\surd \notin \mathsf{A}$. Tables 3, without the blue rule, and 4 provide their operational semantics. Here a,b,c range over A and α,β,ζ over $\mathsf{A} \uplus \{\surd\}$. In Table 4 each $\mathcal{R}$ has been extended with the pair $(\surd,\surd)$ to become a binary relation over $\mathsf{A} \uplus \{\surd\}$, and γ is extended to type $(\mathsf{A} \uplus \{\surd\}) \times (\mathsf{A} \uplus \{\surd\}) \to (\mathsf{A} \uplus \{\surd,\delta\})$ by $\gamma(\surd,\surd) = \surd$ and $\gamma(a,\surd) = \gamma(\surd,a) = \delta$, thus stipulating that a parallel composition can terminate only when both of its components can do so.

$$\frac{}{a \xrightarrow{a} \varepsilon} \qquad \frac{}{\varepsilon \xrightarrow{\surd} \delta} \qquad \frac{P \xrightarrow{\alpha} P'}{P+Q \xrightarrow{\alpha} P'} \qquad \frac{Q \xrightarrow{\alpha} Q'}{P+Q \xrightarrow{\alpha} Q'} \qquad \frac{P \xrightarrow{a} P'}{P \cdot Q \xrightarrow{a} P' \cdot Q} \qquad \frac{P \xrightarrow{\surd} P' \quad Q \xrightarrow{\alpha} Q'}{P \cdot Q \xrightarrow{\alpha} Q'}$$

$$\frac{P \xrightarrow{a} P'}{P \between Q \xrightarrow{a} P' \parallel Q} \qquad \frac{P \xrightarrow{a} P'}{P \parallel Q \xrightarrow{a} P' \parallel Q} \qquad \frac{Q \xrightarrow{a} Q'}{P \parallel Q \xrightarrow{a} P \parallel Q'} \qquad \frac{P \xrightarrow{\alpha} P',\ Q \xrightarrow{\beta} Q'}{P \parallel Q \xrightarrow{\zeta} P' \parallel Q'}\ {\scriptstyle \gamma(\alpha,\beta)=\zeta} \qquad \frac{P \xrightarrow{\alpha} P',\ Q \xrightarrow{\beta} Q'}{P|Q \xrightarrow{\zeta} P' \parallel Q'}\ {\scriptstyle \gamma(\alpha,\beta)=\zeta}$$

$$\frac{P \xrightarrow{\alpha} P'}{\partial_H(P) \xrightarrow{\alpha} \partial_H(P')}\ {\scriptstyle \alpha\notin H} \qquad \frac{P \xrightarrow{\alpha} P'}{\mathcal{R}(P) \xrightarrow{\beta} \mathcal{R}(P')}\ {\scriptstyle \alpha\mathcal{R}\beta} \qquad \frac{\langle \mathcal{S}_X|\mathcal{S}\rangle \xrightarrow{\alpha} P'}{\langle X|\mathcal{S}\rangle \xrightarrow{\alpha} P'}$$

Table 4: Structural operational semantics of $\mathrm{ACP}_R^\varepsilon$

Below I will also make use of the extension aprACP_R^{trig} of aprACP_R with the *triggering operator* $f\bullet$ of MEIJE [3]. It renames the initial actions of its argument according to the renaming operator $f : \mathsf{A} \to \mathsf{A}$; see the blue rule in Table 3.

[13] In [39, Section 3.5] I showed that up to strong bisimilarity $\between$ and $|$ are expressible in the rest of the language.

14.3 Expressiveness of parametrised languages

I will proceed to investigate the relative expressiveness of ACP_R, aprACP_R, $\mathrm{ACP}_R^\varepsilon$ and CSP. This necessitates extending Definition 2.2 of expressiveness to parametrised languages $\mathscr{L}(\Gamma)$, where the parameter Γ could be a vector such as (A,γ). When comparing $\mathrm{aprACP}_R(\mathsf{A},\gamma)$ with $\mathrm{CSP}(\Sigma)$, the parameters of these languages have a different type, so cannot be chosen equal.

Definition 14.2 A parametrised language $\mathscr{L}'(\Lambda)$ is at least as expressive as a parametrised language $\mathscr{L}(\Gamma)$ iff for each choice Γ_0 of the parameter Γ there exists a choice Λ_0 of the parameter Λ such that $\mathscr{L}'(\Lambda_0)$ is at least as expressive as $\mathscr{L}(\Gamma_0)$.

Section 4 compared the expressiveness of the parametrised languages $m\pi(\mathcal{N})$ and $a\pi(\mathcal{N})$ and found them equally expressive up to $\dot{\approx}$, weak barbed bisimilarity. In that section I was not explicit about the choice of the parameter $\mathcal{N}$—the set of names out which π-calculus expressions are built. However, the translation $\mathscr{T}$ from $m\pi$ to $a\pi$ presented there would not be compositional in the sense of Definition 3.1, if for a given choice $\mathcal{N}_0$ of $\mathcal{N}$, the language $m\pi(\mathcal{N}_0)$ would be simply translated into $a\pi(\mathcal{N}_0)$. Namely, the clauses for $\mathscr{T}(\bar{x}z.P)$ and $\mathscr{T}(x(y).P)$ involve the choice of two names $u,v\in\mathcal{N}_0$ that are not allowed to occur in P. This means that the contexts $C_{\bar{x}z.\_}$ and $C_{x(y).\_}$ that must exist by Definition 3.1 cannot be chosen independently of P, which violates Definition 3.1. This problem is solved by applying Definition 14.2 such that each language $m\pi(\mathcal{N}_0)$ is translated into $m\pi(\mathcal{N}_1)$, for $\mathcal{N}_1:=\mathcal{N}_0\uplus\{u,v\}$: the original set of names to which two fresh names are added, for use in these clauses.

14.4 Sequential composition can be encoded in terms of action prefixing

I will show that up to strong bisimilarity aprACP_R is no less expressive than $\mathrm{ACP}_R^\varepsilon$. Strong bisimilarity is often considered the finest reasonable semantic equivalence for system description languages like CCS and ACP. It is defined on the states S of a labelled transition system $(S,\rightarrow)$ where $\rightarrow\subseteq S\times L\times S$. Here L is the set of all possible transition labels. It extends to a relation between states in different transition systems by considering multiple transition systems as one by taking their disjoint union.

Definition 14.3 *Strong bisimilarity* is the largest symmetric relation $\leftrightarrows\subseteq S\times S$ such that

- $P\leftrightarrows Q$ and $P\xrightarrow{a}P'$ implies $Q\xrightarrow{a}Q'$ for some Q' with $P'\leftrightarrows Q'$.

This specialises to $\mathrm{aprACP}_R(\mathsf{A},\gamma)$ and $\mathrm{ACP}_R^\varepsilon(\mathsf{A},\gamma)$ processes by seeing these as states in labelled transition systems with $L=\mathsf{A}$ and $L=\mathsf{A}\uplus\{\surd\}$ respectively. Since $\mathrm{ACP}_R^\varepsilon$ is an extension of ACP_R with one extra constant, it is at least as expressive as ACP_R, with the identity mapping $\mathscr{I}$ a valid translation from $\mathrm{ACP}_R(\mathsf{A},\gamma)$ to $\mathrm{ACP}_R^\varepsilon(\mathsf{A},\gamma)$, provided one restricts the family of languages $\mathrm{ACP}_R(\mathsf{A},\gamma)$ to those sets A satisfying $\surd\notin\mathsf{A}$. The easiest definition of strong bisimilarity supporting this translation simply defines two ACP_R processes P and Q to be strongly bisimilar iff $\mathscr{I}(P)\leftrightarrows\mathscr{I}(Q)$. That $\mathrm{ACP}_R^\varepsilon$ is strictly more expressive, even in absolute terms, follows since in ACP_R we cannot model any process such as $a+\varepsilon$ with a state that allows successful termination as well as some continuation.

Whether aprACP_R is strictly more or strictly less expressive than ACP_R and $\mathrm{ACP}_R^\varepsilon$ depends on whether we allow family members $\mathrm{aprACP}_R(\mathsf{A},\gamma)$ with $\surd\in\mathsf{A}$ or not. If not, $\mathrm{aprACP}_R(\mathsf{A},\gamma)$ can be translated into ACP_R or $\mathrm{ACP}_R^\varepsilon$ by the translation $\mathscr{T}$ satisfying $\mathscr{T}(a.P):=a\cdot\mathscr{T}(P)$ for all $a\in\mathsf{A}$ that translates all other constructs of aprACP_R homomorphically. Here an n-ary operator f is translated homomorphically iff $\mathscr{T}(f(P_1,\ldots,P_n))=f(\mathscr{T}(P_1),\ldots,\mathscr{T}(P_n))$, and recursion is translated homomorphically iff $\mathscr{T}(\langle X|\{Y=\mathcal{S}_Y\mid Y\in W\}\rangle)=\langle X|\{Y=\mathscr{T}(\mathcal{S}_Y)\mid Y\in W\}\rangle$ with $\mathscr{T}(X)=X$. So the identity translation $\mathscr{I}$ is the one that translates all constructs homomorphically. That the above translation is valid up to $\leftrightarrows$ is straightforward. Strictness follows since even the absolute expressiveness of aprACP_R is less than that of ACP_R or $\mathrm{ACP}_R^\varepsilon$, for in aprACP_R we cannot express any terminating process such as a.

When allowing $\surd \in \mathsf{A}$ at the side of $\mathrm{aprACP}_R(\mathsf{A},\gamma)$, the language $\mathrm{ACP}^{\varepsilon}_R(\mathsf{A}_0,\gamma_0)$ can be translated into $\mathrm{aprACP}^{trig}_R(\mathsf{A}_2,\gamma_2)$, with $\mathsf{A}_2 := \mathsf{A}_1 \uplus H$, $\mathsf{A}_1 := \mathsf{A}_0 \uplus \{\surd\}$ and $H := \{a^{\mathrm{ini}}, a^{\mathrm{post}} \mid a \in \mathsf{A}_1\} \uplus \{\checkmark\}$, where $\gamma_1 : \mathsf{A}_1 \times \mathsf{A}_1 \to \mathsf{A}_1 \uplus \{\delta\}$ extends γ_0 with $\gamma_1(\surd,\surd) = \surd$ and $\gamma_1(a,\surd) = \gamma_1(\surd,a) = \delta$ for $a \in \mathsf{A}_0$, and γ_2 extends γ_1 with $\gamma_2(\checkmark, a^{\mathrm{ini}}) = \gamma_2(a^{\mathrm{ini}},\checkmark) = a^{\mathrm{post}}$ for $a \in \mathsf{A}_1$ (and $\gamma_2(\alpha,\beta) = \delta$ when not defined so far). Let $f_{\surd}, f^{\mathrm{ini}}, f^{\mathrm{post}} : \mathsf{A}_2 \to \mathsf{A}_2$ be the renaming functions (special cases of the renaming relation $\mathcal{R}$ of aprACP_R) defined by $f_{\surd}(\surd) = \checkmark$, $f_{\surd}(\alpha) = \alpha$ for all $\alpha \neq \surd$; $f^{\mathrm{ini}}(a) = a^{\mathrm{ini}}$ for all $a \in \mathsf{A}_1$, $f^{\mathrm{ini}}(\alpha) = \alpha$ for all $\alpha \in H$; $f^{\mathrm{post}}(a^{\mathrm{post}}) = a$ for all $a \in \mathsf{A}_1$, $f^{\mathrm{post}}(\alpha) = \alpha$ for all $\alpha \neq a^{\mathrm{post}}$. Let $\mathscr{T}$ be the translation from $\mathrm{ACP}^{\varepsilon}_R(\mathsf{A}_0,\gamma_0)$ to $\mathrm{aprACP}^{trig}_R(\mathsf{A}_2,\gamma_2)$ defined by

$$\mathscr{T}(\varepsilon) = \surd.\delta \qquad \mathscr{T}(a) = a.\surd.\delta \text{ for all } a \in \mathsf{A}_0 \qquad \mathscr{T}(P \cdot Q) = \partial_H\big(f^{\mathrm{post}}\big(f_{\surd}(\mathscr{T}(P)) \| f^{\mathrm{ini}} \bullet \mathscr{T}(Q)\big)\big)$$

and where the remaining constructs are translated homomorphically. This translation is such that processes $\mathscr{T}(P)$ exhibit transition labels from A_1 only, but the ones from H play an auxiliary rôle in the definition of $\mathscr{T}(P \cdot Q)$. Translation $\mathscr{T}$ renames the termination label $\surd$ of the first process in a sequential composition into the auxiliary label $\checkmark$, and tags all initial actions (including $\surd$) of the second process with a tag $^{\mathrm{ini}}$. The encapsulation operator ∂_H makes sure that none of those actions can occur by themselves. Now $P \cdot Q$ is implemented by means of a parallel composition that forces all initial actions of Q to synchronise with the termination action of P. That this translation is valid up to $\leftrightarroweq$ follows because $\mathbf{R} := \{(P, \mathscr{T}(P)) \mid P \in \mathrm{T}_{\mathrm{ACP}^{\varepsilon}_R}\} \subseteq \leftrightarroweq$, as can be shown by a straightforward structural induction on P. It follows that up to $\leftrightarroweq$ aprACP^{trig}_R is more expressive than $\mathrm{ACP}^{\varepsilon}_R$. In [39] it was shown that up to $\leftrightarroweq$ aprACP_R is equally expressive as aprACP^{trig}_R, i.e., triggering has a valid translation in aprACP_R. Thus, by Theorem 11.11, aprACP_R is more expressive than $\mathrm{ACP}^{\varepsilon}_R$. This time strictness follows since only in aprACP_R can one find a process that performs some action b after $\surd$.

The observation that up to $\leftrightarroweq$ $\mathrm{ACP}^{\varepsilon}_R$ can be encoded into aprACP_R does not come as a surprise. The structural operational semantics of CSP^{op}, aprACP_R and $\mathrm{ACP}^{\varepsilon}_R$ consists of rules of a particular form, described by Robert de Simone [106], which is now known as *De Simone format*; accordingly, these languages are called *De Simone languages*. He showed [105] that up to strong bisimilarity, any De Simone language admits a valid translation into MEIJE. Inspired by De Simone's work, in [39] I showed that up to $\leftrightarroweq$ De Simone language also admit valid translations into aprACP_R. Crucial for this result is that any label that appears in the operational semantics of a source language of the translation may also be included in the parameter A of aprACP_R. For this reason I do allow actions like τ and $\surd$ there.

14.5 Encoding CSP into aprACP^{τ}_R

The language ACP^{τ} extends ACP with a constant τ, denoting an unobservable action, and operators τ_I for $I \subseteq \mathsf{A}$ that abstract from the actions in I by making them unobservable. The same extension applies to the languages $\mathrm{ACP}^{\tau}_R(\mathsf{A},\gamma)$, $\mathrm{ACP}^{\tau\varepsilon}_R(\mathsf{A},\gamma)$ and $\mathrm{aprACP}^{\tau}_R(\mathsf{A},\gamma)$, although in the latter case the extension is with an action-prefixing operator $\tau._-$. Below are the operational rules for the τ_I operators.

$$\frac{P \xrightarrow{a} P'}{\tau_I(P) \xrightarrow{\tau} \tau_I(P')}\ {\scriptstyle a \in I} \qquad \frac{P \xrightarrow{\alpha} P'}{\tau_I(P) \xrightarrow{\alpha} \tau_I(P')}\ {\scriptstyle \alpha \notin I}$$

Naturally, one should now require that $\tau \notin \mathsf{A}$, and α ranges over $\mathsf{A} \uplus \{\tau\}$. Moreover, in the rules of Tables 2 and 3, one can let a,b,c range over $\mathsf{A} \uplus \{\tau\}$, when extending $\mathcal{R}$ with the pair (τ,τ), and stipulating that $\gamma(a,\tau) = \gamma(\tau,a) = \delta$. Up to $\leftrightarroweq$, the language $\mathrm{aprACP}^{\tau}_R(\mathsf{A},\gamma)$ is strictly more expressive than $\mathrm{aprACP}_R(\mathsf{A},\gamma)$; the identity mapping is a valid translation from $\mathrm{aprACP}_R(\mathsf{A},\gamma)$ to $\mathrm{aprACP}^{\tau}_R(\mathsf{A},\gamma)$.

Nevertheless, the family of languages aprACP_R is strictly more expressive than the family aprACP_R^τ. This is because there exists a straightforward translation, valid up to $\leftrightarrows$, from $\mathrm{aprACP}_R^\tau(\mathsf{A}_0,\gamma_0)$ into $\mathrm{aprACP}_R(\mathsf{A}_1,\gamma_1)$ for $\mathsf{A}_1 := \mathsf{A}_0 \uplus \{\tau\}$ and γ_1 extending γ_0 by $\gamma_1(\tau,\alpha) = \gamma_1(\alpha,\tau) = \delta$ for all $\alpha \in \mathsf{A}_1$. This translation simulates the abstraction operators τ_I by renaming operators.

The previous section quoted a general result that implies that there is a translation, valid up to $\leftrightarrows$, from CSP^{op} into aprACP_R. I conjecture that this result cannot be sharpened into a translation, valid up to $\leftrightarrows$, from CSP^{op} into aprACP_R^τ. However, below I present a translation from CSP^{op} into aprACP_R^τ for which I conjecture that it is valid up to *rooted weak bisimilarity* [40], originally called *observation congruence* [70], or even up to *rooted branching bisimilarity* [51].

I translate $\mathrm{CSP}^{op}(\Sigma)$ into $\mathrm{aprACP}_R^\tau(\mathsf{A},\gamma)$, where $\mathsf{A} := \Sigma \uplus \{\tau\} \uplus H$ with $H := \{a_{\mathtt{first}}, a_{\mathtt{next}}, a_{\mathtt{ini}}, a_{\mathtt{syn}}, a_{\mathtt{post}} \mid a \in \Sigma\} \cup \{\mathtt{first}, \mathtt{next}\}$ and γ defined by

$$\begin{array}{rclcrcl} \gamma(a,\mathtt{first}) &=& a_{\mathtt{first}} & & \gamma(a,\mathtt{next}) &=& a_{\mathtt{next}} \\ \gamma(\mathtt{choose},a_{\mathtt{ini}}) &=& a_{\mathtt{post}} & & \gamma(a_{\mathtt{syn}},a_{\mathtt{syn}}) &=& a_{\mathtt{post}} \end{array}$$

for all $a \in \Sigma$, in the understanding that γ is commutative and $\gamma(a,b) = \delta$ when not defined otherwise.

Write a^∞ for the process that perpetually performs the action a. This process is rendered in aprACP_R^τ as $\langle X|\{X = a.X\}\rangle$. Let $H_1 := \Sigma \uplus \{\mathtt{first}, \mathtt{next}\}$. Then, for P an aprACP_R^τ-process with actions from $\Sigma \uplus \{\tau\}$, $\partial_{H_1}(P \| \mathtt{first}.\mathtt{next}^\infty)$ is the process obtained from P by tagging the first visible actions by (the subscript) $\mathtt{first}$, and all subsequent ones by $\mathtt{next}$. Let $f^A, f^{\mathtt{ini}}, f^{\mathtt{post}} : \mathsf{A} \to \mathsf{A}$ be the renaming functions defined by

$$f^A(a) = \begin{cases} a_{\mathtt{syn}} & \text{if } a \in A \\ a & \text{otherwise} \end{cases} \qquad f^{\mathtt{ini}}(\alpha) = \begin{cases} a_{\mathtt{ini}} & \text{if } \alpha = a_{\mathtt{first}} \\ a & \text{if } \alpha = a_{\mathtt{next}} \\ \alpha & \text{otherwise} \end{cases} \qquad f^{\mathtt{post}}(\alpha) = \begin{cases} a & \text{if } \alpha = a_{\mathtt{post}} \\ \alpha & \text{otherwise} \end{cases}$$

—they can be employed as special cases of the relational renaming operator $\mathcal{R}$ of $\mathrm{aprACP}_R^\tau(\mathsf{A},\gamma)$. Now, for P an aprACP_R^τ-process with actions from $\Sigma \uplus \{\tau\}$, $f^{\mathtt{ini}}(\partial_{H_1}[P \| \mathtt{first}.\mathtt{next}^\infty])$ is the process obtained from P by tagging the first visible actions by $\mathtt{ini}$.

Below is my translation $\mathscr{T}$ from $\mathrm{CSP}^{op}(\Sigma)$ into $\mathrm{aprACP}_R^\tau(\mathsf{A},\gamma)$. Processes $\mathscr{T}(P)$ exhibit transition labels from $\Sigma \uplus \{\tau\}$ only.

$$\begin{array}{rclcrcl} \mathscr{T}(\mathbf{div}) &=& \langle X|\{X = \tau.X\}\rangle & & \mathscr{T}(a{\to}E) &=& a.\mathscr{T}(E) \\ \mathscr{T}(\mathbf{stop}) &=& \delta & & \mathscr{T}(E \sqcap F) &=& \tau.\mathscr{T}(E) + \tau.\mathscr{T}(F) \\ \mathscr{T}(E\backslash A) &=& \tau_A(\mathscr{T}(E)) & & \mathscr{T}(X) &=& X \\ \mathscr{T}(f(E)) &=& f(\mathscr{T}(E)) & & \mathscr{T}(\mu X.E) &=& \langle X|\{X = \tau.\mathscr{T}(E)\}\rangle \\ \mathscr{T}(E \|_A F) &=& \partial_H\big(f^{\mathtt{post}}\big(f^A(E) \| f^A(F)\big)\big) & & & & \\ \mathscr{T}(E \,\Box\, F) &=& \partial_H\Big(f^{\mathtt{post}}\Big(f^{\mathtt{ini}}(\partial_{H_1}[E \| \mathtt{first}(\mathtt{next}^\infty)]) \Big\| \mathtt{choose}.\delta \Big\| f^{\mathtt{ini}}(\partial_{H_1}[F \| \mathtt{first}(\mathtt{next}^\infty)])\Big)\Big) & & & & \end{array}$$

This translation is not valid up to strong bisimilarity, as $\mathscr{T}(\tau{\to}\mathbf{stop} \,\Box\, b{\to}c{\to}\mathbf{stop})$ has a trace $b\tau c$, which is not a trace of $\tau{\to}\mathbf{stop} \,\Box\, b{\to}c{\to}\mathbf{stop}$. I conjecture that $\mathscr{T}$ is valid up to rooted branching bisimilarity [51], because that equivalence satisfies the law $a.P = a.(\tau.0 \| P)$.

The results of Sections 14.4 and 14.5 are depicted in the right-lower corner of Figure 2, where the grey dot denotes aprACP_R^τ extended with triggering. Here a black arrow $\mathscr{L} \longrightarrow \mathscr{L}'$ indicates a translation valid up to strong bisimilarity, whereas a red one is conjectured valid up to rooted branching bisimilarity. By Theorem 11.18, the expressiveness hierarchy $\mathrm{CSP}^{op} \longrightarrow \mathrm{aprACP}_R^\tau \longrightarrow \mathrm{aprACP}_R^{\tau,trig} \longrightarrow \mathrm{aprACP}_R$ is strict, since (a) rooted branching bisimilarity is a congruence for $\mathrm{aprACP}_R^{\tau,trig}$ [41] but not for aprACP_R, (b) rooted weak bisimilarity is a congruence for aprACP_R^τ [41] but not for triggering, and (c) weak bisimilarity is a congruence for CSP^{op} [41] but not for the + of aprACP_R^τ.

$$\alpha.P \xrightarrow{\alpha} P \qquad \frac{P_j \xrightarrow{\alpha} P'_j}{\sum_{i\in I} P_i \xrightarrow{\alpha} P'_j}\ (j\in I)$$

$$\frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q} \qquad \frac{P \xrightarrow{a} P',\ Q \xrightarrow{\bar{a}} Q'}{P|Q \xrightarrow{\tau} P'|Q'} \qquad \frac{Q \xrightarrow{\alpha} Q'}{P|Q \xrightarrow{\alpha} P|Q'}$$

$$\frac{P \xrightarrow{\alpha} P'}{P\backslash H \xrightarrow{\alpha} P'\backslash H}\ (\alpha \notin H\cup\overline{H}) \qquad \frac{P \xrightarrow{\alpha} P'}{P[f] \xrightarrow{f(\alpha)} P'[f]} \qquad \frac{\langle \mathcal{S}_X|\mathcal{S}\rangle \xrightarrow{\alpha} P'}{\langle X|\mathcal{S}\rangle \xrightarrow{\alpha} P'}$$

Table 5: Structural operational semantics of CCS

14.6 CCS

CCS [70] is parametrised with a set A of *visible actions*. The set $\overline{\mathsf{A}}$ of *co-actions* is $\overline{\mathsf{A}} := \{\bar{a} \mid a\in\mathsf{A}\}$, and $L := \mathsf{A}\cup\overline{\mathsf{A}}$ is the set of *labels*. The function $\bar{\cdot}$ is extended to L by declaring $\bar{\bar{a}} = a$. Finally, $Act := L \uplus \{\tau\}$ is the set of *actions*. Below, a, b, c, ... range over L and α, β over Act. A *relabelling* is a function $f : L \to L$ satisfying $f(\bar{a}) = \overline{f(a)}$; it extends to Act by $f(\tau) := \tau$. The syntax of CCS is given by

$$E,F ::= \alpha.E \mid \textstyle\sum_{i\in I} E_i \mid E|F \mid E\backslash H \mid E[f] \mid X \mid \langle X|\mathcal{S}\rangle$$

where I is an arbitrary index set, E, F and the E_i are CCS expressions, $\alpha\in Act$, $H\subseteq\mathsf{A}$, f is a relabelling and $X\in\mathscr{X}$. The recursive call $\langle X|\mathcal{S}\rangle$ is the same as for ACP. I use ACP notation here; in [70] the recursive specification $\mathcal{S}$ is written $\{X_i = E_i \mid i\in I\}$ and the recursive call $\langle X_j|\mathcal{S}\rangle$ is displayed as $\mathbf{fix}_j\mathcal{S}$. The transition relations $\xrightarrow{\alpha}$ between closed CCS expressions are derived by the rules of Table 5. One writes P_1+P_2 for $\sum_{i\in I} P_i$ when $I=\{1,2\}$, and $\mathbf{0}$ when $I=\emptyset$.

Here I consider twelve versions of CCS. First of all, CCS$^+$ is obtained by restricting the index sets I of the choice operator $\sum_{i\in I}$ to be finite. Equivalently, it can be presented by dropping that operator altogether, in favour of the binary choice operator $+$ (with the same operational rules as in Table 3) and the constant $\mathbf{0}$ (with no operational rules). It can be deemed an advantage of this version of CCS that there are no operators with infinitely many arguments, but on the other hand the infinite sum is extremely convenient in practical applications, e.g., when modelling the receipt over some channel of an arbitrary natural number. A compromise is found in the language CCS^q, featuring $\mathbf{0}$ and $+$ as well as the choice quantifier $\sum_{h\in\mathscr{H}} E[f_h]$ proposed in Section 2. This language features only operators with finitely many arguments, yet still allows applications of the above-mentioned kind. In general, up to $\leftrightarrows$, the authentic CCS [70] is more expressive than CCS$^+$, with CCS^q sitting in between, as indicated in the second column of Figure 2. However, the operator $\sum_{i\in I}$ where I is required to be countable is already expressible in CCS$^+$, as shown in [39]. This encoding hinges on the use of unguarded recursive specifications with infinitely many equations.

Orthogonal to that hierarchy are the variants of CCS without variables and without the recursion construct, but parametrised with a set of *agent identifiers* X_i, which are extra constants in the language, each equipped with a *defining equation* $X_i \stackrel{\text{def}}{=} P_i$ with $P_i\in\mathrm{T}_{\mathrm{CCS}}$ [69]. The extra parameter of this language CCS$^{\mathrm{AI}}$ can be seen as a recursive specification $\mathcal{S}$, but one without free variables. It is trivial to translate CCS$^{\mathrm{AI}}(\mathsf{A},\mathcal{S})$ into CCS(A), namely by translating each agent identifier X into the recursive call $\langle X|\mathcal{S}\rangle$. Even though the absolute expressiveness of CCS$^{\mathrm{AI}}$ and CCS is the same, the relative expressiveness of CCS is greater, up to $\leftrightarrows$. Namely, the replication operator ! of the π-calculus [101], applied to the variable

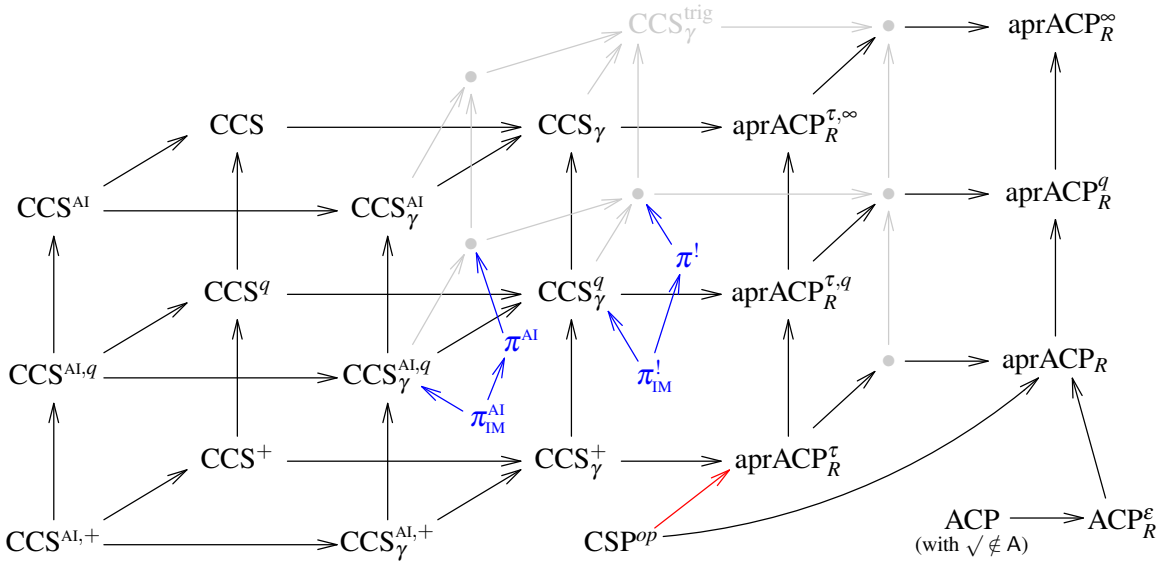


Figure 2: Expressiveness hierarchy between some versions of CCS, CSP, ACP and the π-calculus

X, can be translated into CCS as $\langle Y|\{Y = X\|Y\}\rangle$. Here, the recursive specification $\{Y = X\|Y\}$ features the free variable $X \in \mathscr{X}$, which is not permitted in CCS$^{\rm AI}$. I conjecture that the replication operator cannot be expressed in CCS$^{\rm AI}$.

There exists a trivial translation, valid up to $\leftrightarrows$, from CCS$^+$(A) into aprACP$^\tau_R$(B,γ), where B := A$\uplus\overline{A}\uplus I$, $I := \{\hat{a} \mid a \in {\rm A}\}$ and $\gamma(a,\bar{a}) = \gamma(\bar{a},a) = \hat{a}$ for all $a \in$ A:

$$\mathscr{T}(\mathbf{0}) \;=\; \delta \qquad \mathscr{T}(E\backslash H) \;=\; \partial_H(\mathscr{T}(E)) \qquad \mathscr{T}(E[f]) \;=\; f(\mathscr{T}(E)) \qquad \mathscr{T}(E|F) \;=\; \tau_I(E\|F)$$

and the other constructs are translated homomorphically. When aprACP$^{\tau,\infty}_R$ and aprACP$^{\tau,q}_R$ denote the extensions of aprACP$^\tau_R$ with the infinite choice of CCS and the choice quantifier, respectively, one can find similar translations from CCS into aprACP$^{\tau,\infty}_R$, and from CCS^q into aprACP$^{\tau,q}_R$, as indicated in Figure 2.

In [48] a variant CCS$_\gamma$ of CCS was defined as a middle ground between CCS and aprACP$^{\tau,\infty}_R$. It has three parameters: an alphabet A of *visible actions*, with a subset $\mathscr{S} \subseteq$ A of *synchronisations*, and a commutative *communication function* $\gamma\colon (\mathrm{A}\backslash\mathscr{S})^2 \to \mathscr{S}\uplus\{\tau,\delta\}$. This disables multiway synchronisation. Compared to CCS there are no co-actions, so $Act := \mathrm{A}\uplus\{\tau\}$. The syntax of CCS$_\gamma$ is the same as that of CCS, except that parallel composition is denoted $\|$ rather than $|$, following ACP. This indicates a semantic difference: the rule for communication in the middle of Table 5 is for CCS$_\gamma$ replaced by the corresponding rule from Table 3. Moreover, relabelling operators $f : \mathrm{A} \to Act$ are allowed to rename visible actions into τ, but not vice versa.[14] They are required to satisfy $c \in \mathscr{S} \Rightarrow f(c) \in \mathscr{S}\cup\{\tau\}$. These are the only differences between CCS and CCS$_\gamma$. As indicated in Figure 2, each of the 6 versions of CCS considered above has a CCS$_\gamma$-counterpart.

All variants of CCS and aprACP can also be extended with the triggering operator of MEIJE [3]. As mentioned before, such an extension doesn't increase the expressiveness of aprACP$_R$. For the some of the other languages in Figure 2, its extension with triggering is indicated in grey.

[14] Renaming into τ could already be done in CCS by means of parallel composition. Hence this feature in itself does not add extra expressiveness.

14.7 The π-calculus

The original π-calculus [72, 73]—I here call it π^{AI}—is parametrised with an infinite set $\mathcal{N}$ of *names* and, for each $n \in \mathbb{N}$, a set of $\mathcal{K}_n$ of *agent identifiers* of arity n. Its syntax is given by

$$P,Q ::= \mathbf{0} \;\mid\; \tau.P \;\mid\; \bar{x}y.P \;\mid\; x(z).P \;\mid\; P+Q \;\mid\; P|Q \;\mid\; [x{=}y]P \;\mid\; (\nu z)P \;\mid\; V(y_1,\ldots,y_n)$$

with x,y,z,y_i ranging over $\mathcal{N}$ and $V \in \mathcal{K}_n$. The set $\mathsf{fn}(P)$ of free names of an expression $P \in \mathrm{T}_{\pi^{\text{AI}}}$ is given as in Definition 4.1. Each agent identifier $V \in \mathcal{K}_n$ comes with a unique *defining equation*

$$V(x_1,\ldots,x_n) \stackrel{\text{def}}{=} P$$

where the names x_i are all distinct and $\mathsf{fn}(P) \subseteq \{x_1,\ldots,x_n\}$.

In [101] a version of the π-calculus is standardised whose syntax differs on two counts from π^{AI}. First of all, following [71], the agent identifiers are replaced by a unary replication operator !, where $!P$ is like an infinite parallel composition $P|P|\cdots$. Secondly, the matching operator $[x{=}y]$ may be used only in a disciplined way: an expression $[x{=}y]P$ is allowed only when P has the form $\alpha.Q$, where α is τ, xy or $x(z)$, possibly preceded by other matching operators. Semantically, $[x{=}y]P$ behaves as P when x and y are names of the same channel, and as $\mathbf{0}$ otherwise. I denote this version of the π-calculus by $\pi^{!}_{\text{IM}}$, where IM stands for *implicit matching*. Between π^{AI} and $\pi^{!}_{\text{IM}}$ languages $\pi^{\text{AI}}_{\text{IM}}$ and $\pi^{!}$ can be found, which differ from π^{AI} on only one of these counts. The language $m\pi$ from Section 4 is a sublanguage of $\pi^{!}_{\text{IM}}$.

Notions of structural congruence, a reduction relation and barbs can be defined for these versions of the π-calculus, similar to how this was done in Section 4—see [101] for details. Consequently, $\dot{\sim}$, strong barbed bisimilarity—see Definition 13.1—is well defined on π-calculus expressions. In [48] I extended this definition to CCS_γ by defining a reduction relation and barbs for CCS_γ. As reductions I simply took the τ-transitions, and the barbs where obtained by choosing a suitable set A as parameter of CCS_γ, defining a partial function $O : \mathsf{A} \rightharpoonup \Omega$, with Ω a collection of barbs, and letting $P{\downarrow_\omega}$ iff $P \stackrel{a}{\longrightarrow} P'$ for some P' and $a \in \mathsf{A}$ with $O(a) = \omega$.

In [48] I presented a translation from $\pi^{\text{AI}}_{\text{IM}}$ into $\text{CCS}^{\text{AI},q}_\gamma$ that is valid up to $\dot{\sim}$. It is indicated in blue in Figure 2. In [48] this translation was presented as going from $\pi^{\text{AI}}_{\text{IM}}$ into $\text{CCS}^{\text{AI}}_\gamma$, there called π_{IM} and CCS_γ since agent identifiers where used throughout the paper. However, it is trivial to see that the target of the translation lays within $\text{CCS}^{\text{AI},q}_\gamma$. The same paper also presented an extension of this translation from all of π^{AI} into $\text{CCS}^{\text{AI},q}_\gamma$ extended with the triggering operator, also indicated in Figure 2. Instead of using the above-mentioned barbs that are typical for the π-calculus, the translations of [48] are also valid when using barbs that are external to π and CCS_γ, as advocated in Section 13.3; this follows by a straightforward simplification of the proofs in [48].

Strong early congruence, the main semantic equivalence relation on π-calculus processes employed in [101], arises as the congruence closure of $\dot{\sim}$. In an ad hoc way, namely dependent on the translation $\mathscr{T} : \mathrm{T}_{\pi^{\text{AI}}_{\text{IM}}} \to \mathrm{T}_{\text{CCS}^{\text{AI}}_\gamma}$, strong early congruence can be extended to $\mathrm{T}_{\mathscr{T}(\pi^{\text{AI}}_{\text{IM}})} \subseteq \mathrm{T}_{\text{CCS}^{\text{AI}}_\gamma}$, the image of $\pi^{\text{AI}}_{\text{IM}}$ under $\mathscr{T}$, where it also is the congruence closure of $\dot{\sim}$. Now by Theorem 11.26 the translation $\mathscr{T}$ is not only valid up to $\dot{\sim}$, but even under this so-defined strong early congruence. As a consequence of this, system specifications and verifications carried out in $\pi^{\text{AI}}_{\text{IM}}$ can be replayed in $\text{CCS}^{\text{AI}}_\gamma$.

An earlier translation from the π-calculus into CSP was presented in [98]. However, that translation was not compositional, and accordingly one cannot make as strong a claim about replaying π-calculus specifications and verifications in CSP.

In [48] I also provided two separation results. First of all, there is no translation from $\text{CCS}^{\text{AI},q}_\gamma$ (or $\text{CCS}^{\text{AI},q}$) back to $\mathrm{T}_{\pi^{\text{AI}}_{\text{IM}}}$ that is valid up to $\dot{\sim}$, so up to $\dot{\sim}$ $\text{CCS}^{\text{AI}}_\gamma$ is strictly more expressive than $\pi^{\text{AI}}_{\text{IM}}$. Secondly, there exists no translation valid up to $\dot{\sim}$ of $\pi^{\text{AI}}_{\text{IM}}$ into CCS.

The translation of [48] of $\pi^{\rm AI}_{\rm IM}$ into $\mathrm{CCS}^{\mathrm{AI},q}_{\gamma}$, and of $\pi^{\rm AI}$ into $\mathrm{CCS}^{\mathrm{AI},q,\mathrm{trig}}_{\gamma}$, can be modified into one from $\pi^{!}_{\rm IM}$ into $\mathrm{CCS}^{q}_{\gamma}$, and of $\pi^{!}$ into $\mathrm{CCS}^{q,\mathrm{trig}}_{\gamma}$. Since my translations are compositional, the only extra effort lays in translating the replication operator. Clearly, each $\pi^{!}$-calculus process $!P$ with $fv(P) = \{x_1,\ldots,x_n\}$ can be rendered in $\pi^{\rm AI}$, namely as $V(x_1,\ldots,x_n)$, where $V \in \mathcal{K}_n$ is the agent identifier with the defining equation $V(x_1,\ldots,x_n) \stackrel{\rm def}{=} P|V(x_1,\ldots,x_n)$. However, there is no compositional translation of $\pi^{!}$ into $\pi^{\rm AI}$, translation the replication *operator* into a unary $\pi^{\rm AI}$-context. For this reason, when translating $\pi^{!}_{\rm IM}$ into $\mathrm{CCS}^{q}_{\gamma}$, we cannot translate replication with agent identifiers, so as to stay in $\mathrm{CCS}^{\mathrm{AI},q}_{\gamma}$. Instead, inspired by Example 10.14, any compositional translation $\mathscr{T}$ from $\pi^{\rm AI}_{\rm IM}$ into $\mathrm{CCS}^{\mathrm{AI},q}_{\gamma}$ can be extended to $\pi^{!}_{\rm IM}$ by

$$\mathscr{T}(!P) := \langle X | \{X = \mathscr{T}(P|X)\}\rangle.$$

15 Related work

A first version of the concept of a valid translation from the present paper occurred in Boudol [13] under the name *translation* or *syntax-directed implementation* (Page 21). It applies to closed-term languages only. In [39] I have reformulated Boudol's definition, while dropping the restriction to closed-term languages. This allows (but does not enforce) a clear separation of syntax and semantics, in the tradition of universal algebra. Nevertheless, the definition employed in [39] only deals with the case that all (relevant) elements in the domain in which a language is interpreted are denotable as the interpretations of closed terms. In [42] situations are described where such a restriction is undesirable. In addition, both [13] and [39] 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 many more recent encodings [12, 76, 78, 20, 6, 7, 81, 89].

In [42] I alleviated these two restrictions by proposing two notions of encoding: *correct* and *valid* [15] translations up to $\sim$. Each of them generalises the proposals of [13] and [39]. The former drops the restriction on denotability as well as $\sim$ being a congruence for the whole target language, but it requires $\sim$ to be a congruence for the source language, as well as for the source's image within the target. The latter drops both congruence requirements (and allows $\sim$ to be a preorder rather than an equivalence), 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.

The current paper further generalises the work of [42] by proposing a new notion of a valid translation that incorporates the correct and valid translations of [42] as special cases—see Sections 12.4 and 12.2. It drops the congruence requirements as well as the restriction on denotability.

In [42] I limited attention to languages satisfying

$$\textit{if } E \stackrel{\alpha}{=} F \textit{ then } [\![E]\!]_{\mathscr{L}} = [\![F]\!]_{\mathscr{L}}. \tag{5}$$

This postulate says that the meaning of an expression is invariant under α-conversion. It can be reformulated as the requirement that $\stackrel{\alpha}{=}_{\mathscr{L}}$ is the identity relation—cf. Definition 11.4. This postulate is satisfied by all my intended applications, except for the important class of closed-term languages. Languages like CCS and the π-calculus can be regarded as falling in that class (although it is also possible to declare the meaning of a term under a valuation to be an $\stackrel{\alpha}{=}$-equivalence class of closed terms). To bring this type of application within the scope of my theory, in a previous version of this paper [45, 46] I weakened Postulate (5) by requiring merely that $\stackrel{\alpha}{=}_{\mathscr{L}}$ is an equivalence relation. That weakened postulate was needed

[15] To avoid confusion, the notion of validity from [42] is called *A-validity* in the present paper—see Sections 7 and 12.2.

in the proofs of Theorems 11.23 and 11.26. Here I managed to prove these theorems without using this postulate, which could therefore be dropped.

15.1 Full abstraction

The concept of *full abstraction* stems from Milner [67]. It indicates a particularly nice connection between a denotational and an operational semantics of a language $\mathscr{L}$. Here a denotational semantics is a function $[\![\ ]\!]_{\mathscr{L}}$ as introduced in Section 2, whereas an operational semantics is given by an *evaluation function* $\mathscr{E} : \mathrm{T}_{\mathscr{L}} \to \mathbb{O}$ from the closed terms, there called *programs*, to a set $\mathbb{O}$ of *observations*. Evaluation determines an equivalence relation on programs: $P \approx_{\mathscr{E}} Q$ iff $\mathscr{E}(P) = \mathscr{E}(Q)$. Let $\approx^{1c}_{\mathscr{E}}$ be the congruence closure of $\approx_{\mathscr{E}}$ for the language $\mathscr{L}$, as defined in Section 11.10.

Definition 15.1 ([67]) The semantic function $[\![\ ]\!]_{\mathscr{L}}$ for $\mathscr{L}$ is *fully abstract* w.r.t. $\mathscr{E}$ iff for all $P, Q \in \mathrm{T}_{\mathscr{L}}$

$$[\![P]\!]_{\mathscr{L}} = [\![Q]\!]_{\mathscr{L}} \quad \Leftrightarrow \quad P \approx^{1c}_{\mathscr{E}} Q\,.$$

A semantic function $[\![\ ]\!]_{\mathscr{L}}$ always induces an equivalence relation on $\mathrm{T}_{\mathscr{L}}$ by $P \sim_{\mathscr{L}} Q$ iff $[\![P]\!]_{\mathscr{L}} = [\![Q]\!]_{\mathscr{L}}$. When this semantics is defined inductively as indicated in Section 11, $\sim_{\mathscr{L}}$ must be a congruence. Now $[\![\ ]\!]_{\mathscr{L}}$ is fully abstract w.r.t. $\mathscr{E}$ iff $\sim_{\mathscr{L}} = \approx^{1c}_{\mathscr{E}}$.

Note that any equivalence relation $\approx$ can be extracted from an evaluation function $\mathscr{E} : \mathrm{T}_{\mathscr{L}} \to \mathbb{O}$, namely by taking $\mathbb{O}$ to be the set of $\approx$-equivalence classes of closed terms, with $\mathscr{E}$ mapping each $P \in \mathrm{T}_{\mathscr{L}}$ to its own equivalence class. This makes $\approx_{\mathscr{E}}$ equal to $\approx$. Likewise, each congruence relation $\sim$ on $\mathrm{T}_{\mathscr{L}}$ can be obtained from a semantics $[\![\ ]\!]_{\mathscr{L}}$: take $\mathbf{V}$ to be the set of $\sim$-equivalence classes of closed terms, and for $E \in \mathbb{T}_{\mathscr{L}}$ and $\rho : \mathscr{L} \to \mathbf{V}$ define $[\![E]\!]_{\mathscr{L}}(\rho)$ to be the $\sim$-equivalence class of $E[\sigma]$, where $\sigma : \mathscr{X} \to \mathrm{T}_{\mathscr{L}}$ is a closed substitution that maps each variable X to a member of the $\sim$-equivalence class of closed terms $\rho(X)$. Since $\sim$ is a congruence, this definition is independent of the choice of σ.

Consequently, full abstraction can equally well be stated as a relation between two equivalence relations $\sim$ and $\approx$ on $\mathrm{T}_{\mathscr{L}}$:

$$\sim \text{ is } \textit{fully abstract} \text{ w.r.t. } \mathscr{L} \text{ and } \approx \quad \text{iff} \quad \sim\, = \approx^{1c}_{\mathscr{L}}.$$

It is in this spirit that full abstraction has been employed in [38] and subsequent papers.

Riecke [97] and Shapiro [103] extend the notion of full abstraction to translations between languages. Riecke [97] compares languages $\mathscr{L}_{\mathrm{S}}$ and $\mathscr{L}_{\mathrm{T}}$ with a shared evaluation function $\mathscr{E} : \mathrm{T}_{\mathscr{L}_{\mathrm{S}}} \cup \mathrm{T}_{\mathscr{L}_{\mathrm{T}}} \to 2^{\mathbb{O}}$, associating with each closed expression a set of observations. Write $P \sqsubseteq^{\mathscr{E}} Q$, for $P, Q \in \mathrm{T}_{\mathscr{L}_{\mathrm{S}}} \cup \mathrm{T}_{\mathscr{L}_{\mathrm{T}}}$, if $\mathscr{E}(P) \subseteq \mathscr{E}(Q)$, and let $\sqsubseteq^{\mathscr{E}}_{\mathscr{L}_i}$ be the congruence closure of $\sqsubseteq^{\mathscr{E}}$ w.r.t. $\mathscr{L}_i$, for $i = \mathrm{S}, \mathrm{T}$ (source and target). He calls a translation $\mathscr{T} : \mathbb{T}_{\mathscr{L}_{\mathrm{S}}} \to \mathbb{T}_{\mathscr{L}_{\mathrm{T}}}$ fully abstract iff, for all $P, Q \in \mathrm{T}_{\mathscr{L}_{\mathrm{S}}}$,

$$P \sqsubseteq^{\mathscr{E}}_{\mathscr{L}_{\mathrm{S}}} Q \quad \Leftrightarrow \quad \mathscr{T}(P) \sqsubseteq^{\mathscr{E}}_{\mathscr{L}_{\mathrm{T}}} \mathscr{T}(Q)\,.$$

The same notion occurs earlier in Mitchell [75], although not under the name "full abstraction", and using equivalence relations instead of preorders—taking $P \approx^{\mathscr{E}} Q$ iff $\mathscr{E}(P) = \mathscr{E}(Q)$. He compares the expressive power of programming languages in terms of *abstraction preserving reductions* between them. These are translations that are compositional as well as fully abstract in the above sense. Later work abstracts from the evaluation function $\mathscr{E}$, and casts full abstraction directly in terms of equivalences $\approx^{\mathrm{S}}$ and $\approx^{\mathrm{T}}$ on the source and target languages [99].

Felleisen [28] compares the expressive power of programming languages through a notion of *eliminability* of language constructs. With some effort, this approach can be seen to have significant similarities with the approach of Mitchell [75], although it allows certain degenerate reductions [75].

Whereas most work on expressiveness deals with closed-term languages, allowing a focus on syntax over semantics, Shapiro [103] works entirely on the semantic side, leaving the syntax largely implicit. His languages are triples $(\mathbf{V},\mathscr{L},\approx)$, consisting of a semantic domain $\mathbf{V}$, a collection $\mathscr{L}$ of partial operators on $\mathbf{V}$, and a semantic equivalence $\approx$ on $\mathbf{V}$. A *language embedding* of one such language $(\mathbf{V}_{\rm S},\mathscr{L}_{\rm S},\approx^{\rm S})$ into another $(\mathbf{V}_{\rm T},\mathscr{L}_{\rm T},\approx^{\rm T})$ is defined to be a homomorphism of the partial algebra $(\mathbf{V}_{\rm S},\mathscr{L}_{\rm S})$ into the partial algebra $(\mathbf{V}_{\rm T},\mathscr{L}_{\rm T})$. Recasting this definition in terms of my framework, this is a function $\mathbf{R}$: $\mathbf{V}_{\rm S}\to\mathbf{V}_{\rm T}$ such that there exists a compositional translation $\mathscr{T}:\mathscr{L}_1\to\mathscr{L}_2$ correct w.r.t. $\mathbf{R}$ (as in Definition 11.2). For $i={\rm S},{\rm T}$ let $\approx^i_{\mathscr{L}_i}$ be the congruence closure of $\approx^i$ w.r.t. $\mathscr{L}_i$—Shapiro calls this the *fully-abstract* congruence of $\mathscr{L}_i$. Then a language embedding is deemed fully abstract iff, for all $v,w\in\mathbf{V}_{\rm S}$,

$$v\approx^{\rm S}_{\mathscr{L}_{\rm S}} w \quad\Leftrightarrow\quad \mathscr{T}(v)\approx^{\rm T}_{\mathscr{L}_{\rm T}}\mathscr{T}(w)\,.$$

In [104] these fully abstract language embeddings are used to classify a number of concurrent programming languages by expressive power.

In Nestmann & Pierce [78] the notion of full abstraction was generalised by dropping the requirement that the source and target equivalences being compared need to be congruence closures.

Definition 15.2 A translation $\mathscr{T}:\mathbb{T}_{\mathscr{L}_{\rm S}}\to\mathbb{T}_{\mathscr{L}_{\rm T}}$ is *fully abstract* w.r.t. the equivalences $\approx_{\rm S}\subseteq{\rm T}^2_{\mathscr{L}_{\rm S}}$ and $\approx_{\rm T}\subseteq{\rm T}^2_{\mathscr{L}_{\rm T}}$ if, for all $P,Q\in{\rm T}_{\mathscr{L}_{\rm S}}$,

$$P\approx_{\rm S} Q \quad\Leftrightarrow\quad \mathscr{T}(P)\approx_{\rm T}\mathscr{T}(Q)\,.$$

In this form, full abstraction has found widespread applications [76, 7]. Fu [31], for instance, bases a theory of expressiveness on full abstraction, with divergence-preserving branching barbed bisimilarity in the rôle of $\approx_{\rm S}$ and $\approx_{\rm T}$.

As stressed in [54, 85], the notion of full abstraction is meaningful only in relation to a well-chosen pair of source and target equivalences, and only in combination with a criterion like compositionality. In particular, for each encoding $\mathscr{T}$ and each target equivalence $\approx_{\rm T}$ there exists a source term equivalence $\approx_{\rm S}$, namely $\{(P,Q)\mid\mathscr{T}(P)\approx_{\rm T}\mathscr{T}(Q)\}$, such that $\mathscr{T}$ is fully abstract w.r.t. $\approx_{\rm S}$ and $\approx_{\rm T}$. For each injective encoding $\mathscr{T}$ and each source term relation $\approx_{\rm S}$, there exists $\approx_{\rm T}\subseteq{\rm T}^2_{\mathscr{L}_{\rm T}}$, namely $\{(\mathscr{T}(P),\mathscr{T}(Q))\mid P\approx_{\rm S}Q\}$, such that $\mathscr{T}$ is fully abstract w.r.t. $\approx_{\rm S}$ and $\approx_{\rm T}$. Finally, for each pair $\approx_{\rm S}$ and $\approx_{\rm T}$ such that the cardinality of ${\rm T}_{\mathscr{L}_{\rm T}}/_{\approx_{\rm T}}$ is greater than or equal to the cardinality of ${\rm T}_{\mathscr{L}_{\rm S}}/_{\approx_{\rm S}}$ there exists a translation from $\mathscr{L}_{\rm S}$ to $\mathscr{L}_{\rm T}$ that is fully abstract w.r.t. $\approx_{\rm S}$ and $\approx_{\rm T}$.

Naturally, any translation that is valid up to an equivalence $\approx$ as in Definition 11.3 of the current paper is also fully abstract, namely w.r.t. the same equivalence $\approx$ in the rôle of both $\approx_{\rm S}$ and $\approx_{\rm T}$. Furthermore, validity entails compositionality through Theorem 11.7. Both full abstraction and validity imply an injective translation from the $\approx_{\rm S}$-equivalence classes of closed source terms to the $\approx_{\rm T}$-equivalence classes of closed target terms. However, validity demands that this link between source and target terms is again governed by $\approx$, whereas full abstraction implies no counterpart to this crucial requirement.

A typical example of a valid translation is the encoding of the synchronous into the asynchronous π-calculus described in Section 4. This encoding is valid up to weak barbed bisimilarity $\dot{\approx}$. Consequently it is also fully abstract w.r.t. $\dot{\approx}$ and $\dot{\approx}$ according to Definition 15.2. However, it is not fully abstract w.r.t. $\dot{\approx}$ and $\dot{\approx}$ in the more original sense of Shapiro [103] and others, due to the fact that $\dot{\approx}$ is not a congruence for either the source or the target language. By Theorem 11.26 the same translation is also fully abstract w.r.t the congruence closure of $\dot{\approx}$ on the source language, and a somewhat artificial equivalence on the target language that is the congruence closure of $\dot{\approx}$ under translated source contexts. Although closer, this is still not a full-abstraction result according to Shapiro, as the latter equivalence fails to be a congruence for all of the target language. Finally, by Theorem 11.20, the same encoding is furthermore fully abstract

w.r.t. the congruence closure of $\dot{\approx}$ on the target language, and an artificial congruence on the source language that is strictly finer than the congruence closure of $\dot{\approx}$ on the source language. This is a full abstraction result in the framework of [103]. However, in line with the observations of [54, 85], the latter two full abstraction results should not be regarded as lending additional credibility to this particular encoding of the synchronous into the asynchronous π-calculus. Since Theorems 11.26 and 11.20 hold in great generality, these full abstraction results are merely consequences of the validity of the encoding up to $\dot{\approx}$.

It is well known that Boudol's encoding—described in Section 4—of the synchronous into the asynchronous π-calculus fails to be fully abstract w.r.t. $\cong^c$ and $\cong^c_a$. Here $\cong^c$ is the congruence closure of $\dot{\approx}$ on the source language, and $\cong^c_a$ the congruence closure of $\dot{\approx}$ on the target language. A counterexample was given at the end of Section 4. The same analysis applies to the encoding of Honda & Tokoro [61] reviewed in Section 13.7. In [26] this problem is addressed by proposing a strict subcalculus $SA\pi$ of the target language that contains the image of the source language under of a version Honda & Tokoro's encoding, such that this encoding is fully abstract w.r.t. $\cong^c$ and the congruence closure of $\dot{\approx}$ w.r.t. $SA\pi$. In [96] a similar solution to the same problem was found earlier, but for a variant of Boudol's encoding from the *polyadic* π-calculus to the (monadic) asynchronous π-calculus. They defined a class of *well-typed* expressions in the asynchronous π-calculus, such that the well-typed expressions constitute a subcalculus of the target language that contains the image of the source language under the encoding. Again, the encoding is fully abstract w.r.t. $\cong^c$ and the congruence closure of $\dot{\approx}$ w.r.t. that sublanguage. By Theorem 11.26 such results can always be achieved, namely by taking as target language exactly the image of the source language under the encoding. What the results of [96, 26] add is that the sublanguage may be strictly larger than the image of the source language, and, especially in the case of [26], that its definition is not phrased in terms of the encoding.

15.2 Validity of encodings according to Gorla

In the last twenty 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) [99, 62, 12, 83, 78, 80, 76, 22, 21, 17, 20, 6, 7, 82, 81, 93, 19, 110, 18, 57, 94, 109, 1, 58, 90, 2]; see [52, 53] 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. Several of these criteria are discussed and compared in [77], [84] and [86, 87]. Gorla [53] 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. Since then, several authors have used Gorla's framework as a basis for establishing new valid encodings and separation results [52, 64, 92, 89, 91, 33, 34, 35, 36].

Like Boudol [13] and the present paper, Gorla requires a compositionality condition for encodings. However, his criterion differs on two counts from mine. It is stronger in that the context C_f encoding an operator f—see Definition 3.1—may use each hole (a kind variable corresponding to an argument of f) only once. A translation defined (in part) by $\mathscr{T}(f(P)) = g(\mathscr{T}(P)\|\mathscr{T}(P))$ for instance can be compositional according to Definition 3.1, but not according to Gorla's definition. It is weaker than mine in that the context C_f encoding an operator f may be dependent on the set of names occurring freely in the expressions given as arguments of f. This issue is further discussed in [42]. It is an interesting topic for future research to see if there are any valid encodability results à la [53] that suffer from my proposed strengthening of compositionality.

The second criterion of [53] 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. In [42] I argue that this criterion as formalised in [53] is too restrictive.

The remaining three requirements of Gorla (the 'semantic' requirements) are very close to an instantiation of mine with a particular preorder $\precsim$. If one takes $\precsim$ to be weak barbed bisimilarity with explicit divergence (i.e. relating divergent states with divergent states only—see Section 13.5), using external barbs as defined in Section 13.3, then any valid translation in my sense satisfies Gorla's semantic criteria, provided that the equivalence $\asymp$ on the target language that acts as a parameter in Gorla's third criterion is also taken to be divergence-preserving weak barbed bisimilarity. The precise relationships between the proposals of [42] and [53] are further discussed in [88].

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.

16 Conclusion

This paper contributed a definition of a valid translation between system description languages, parametrised by the choice of a semantic equivalence or preorder $\precsim$. When such a translation between a source and target language exists, the target language is said to be at least expressive as the source, up to $\precsim$. The choice of $\precsim$ is a measure for the quality of the translation, and thus for the degree in which the source languages is no more expressive than the target. Several theorems, reviewed in the introduction, support the use of this framework to order system description languages by their expressive power. Future work could involve the construction of a big map, graph or category, featuring many relevant process calculi as nodes or objects, and the translations as arrows between them, labelled with the finest equivalence or preorder for which they are valid.

References

[1] J. Åman Pohjola & J. Parrow (2014): *Priorities Without Priorities: Representing Preemption in Psi-Calculi*. In J. Borgström & S. Crafa, editors: Proceedings Combined 21st International Workshop on *Expressiveness in Concurrency* and 11th Workshop on *Structural Operational Semantics*, *Electronic Proceedings in Theoretical Computer Science* 160, Open Publishing Association, pp. 2–15, doi:10.4204/EPTCS.160.2.

[2] J. Åman Pohjola & J. Parrow (2016): *The Expressive Power of Monotonic Parallel Composition*. In P. Thiemann, editor: *Programming Languages and Systems:* Proceedings 25th *European Symposium on Programming*, ESOP'16, held as part of the European Joint Conferences on *Theory and Practice of Software*, ETAPS'16, LNCS 9632, Springer, pp. 780–803, doi:10.1007/978-3-662-49498-1_30.

[3] D. Austry & G. Boudol (1984): *Algèbre de processus et synchronisations*. *Theoretical Computer Science* 30(1), pp. 91–131, doi:10.1016/0304-3975(84)90067-7.

[4] J.C.M. Baeten & J.A. Bergstra (1991): *A Survey of Axiom Systems for Process Algebras*. Report P9111, Programming Research Group, Department of Mathematics and Computer Science, University of Amsterdam.

[5] J.C.M. Baeten & W.P. Weijland (1990): *Process Algebra*. Cambridge Tracts in Theoretical Computer Science 18, Cambridge University Press, doi:10.1017/CBO9780511624193.

[6] 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'04, July 2004, Turku, Finland, IEEE Computer Society Press, pp. 22–31, doi:10.1109/LICS.2004.1319597.

[7] 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'05, Lisbon, Portugal, July 2005, LNCS 3580, Springer, pp. 1202–1213, doi:10.1007/11523468_97.

[8] R. Banach & F. van Breugel (1998): *Mobility and Modularity: expressing π-calculus in CCS*. Available at http://www.cs.man.ac.uk/~banach/some.pubs/Pi.CCS.ext.abs.pdf.

[9] R. Beauxis, C. Palamidessi & F.D. Valencia (2008): *On the Asynchronous Nature of the Asynchronous pi-Calculus*. In P. Degano, R. De Nicola & J. Meseguer, editors: *Concurrency, Graphs and Models, Essays Dedicated to Ugo Montanari on the Occasion of His 65th Birthday*, LNCS 5065, Springer, pp. 473–492, doi:10.1007/978-3-540-68679-8_29.

[10] J.A. Bergstra & J.W. Klop (1986): *Algebra of communicating processes*. In J.W. de Bakker, M. Hazewinkel & J.K. Lenstra, editors: *Mathematics and Computer Science*, CWI Monograph 1, North-Holland, Amsterdam, pp. 89–138.

[11] J.A. Bergstra, J.W. Klop & E.-R. Olderog (1987): *Failures without chaos: a new process semantics for fair abstraction*. In M. Wirsing, editor: *Formal Description of Programming Concepts – III, Proceedings of the 3th IFIP WG 2.2 working conference*, Ebberup 1986, North-Holland, Amsterdam, pp. 77–103.

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

[13] G. Boudol (1985): *Notes on algebraic calculi of processes*. In K. Apt, editor: *Logics and Models of Concurrent Systems*, Springer, pp. 261–303, doi:10.1007/978-3-642-82453-1_9. Available at https://inria.hal.science/inria-00076161. NATO ASI Series F13.

[14] G. Boudol (1992): *Asynchrony and the π-calculus (Note)*. Technical Report 1702, INRIA.

[15] 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.

[16] 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.

[17] 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.

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

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

[20] M. Carbone & S. Maffeis (2003): *On the Expressive Power of Polyadic Synchronisation in pi-calculus*. *Nordic Journal of Computing* 10(2), pp. 70–98.

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

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

[23] S. Cranen, J. F. Groote, J. J. A. Keiren, F. P. M. Stappers, E. P. de Vink, W. Wesselink & T. A. C. Willemse (2013): *An Overview of the mCRL2 Toolset and Its Recent Advances*. In N. Piterman & S.A. Smolka, editors: Proceedings *Tools and Algorithms for the Construction and Analysis of Systems*, TACAS '13, LNCS 7795, Springer, pp. 199–213, doi:10.1007/978-3-642-36742-7_15.

[24] I. Cristescu, Th. Given-Wilson & A. Legay (2020): *Expressiveness of concurrent intensionality*. *Theoretical Computer Science* 837, pp. 54–83, doi:10.1016/j.tcs.2020.05.007.

[25] R. De Nicola & M. Hennessy (1984): *Testing equivalences for processes*. *Theoretical Computer Science* 34, pp. 83–133, doi:10.1016/0304-3975(84)90113-0.

[26] W. Du, Z. Yang & H. Zhu (2018): *A Fully Abstract Encoding for Sub Asynchronous Pi Calculus*. In J. Pang, C. Zhang, J. He & J. Weng, editors: Proc. 2018 International Symposium on *Theoretical Aspects of Software Engineering*, TASE'18, IEEE Computer Society, pp. 17–27, doi:10.1109/TASE.2018.00011.

[27] A. Fehnker, R.J. van Glabbeek, P. Höfner, A.K. McIver, M. Portmann & W.L. Tan (2012): *A Process Algebra for Wireless Mesh Networks*. In H. Seidl, editor: *Programming Languages and Systems:* Proc. 21st *European Symposium on Programming*, ESOP'12; held as part of the *European Joint Conferences on Theory and Practice of Software*, ETAPS'12, LNCS 7211, Springer, pp. 295–315, doi:10.1007/978-3-642-28869-2_15.

[28] M. Felleisen (1991): *On the Expressive Power of Programming Languages*. *Science of Computer Programming* 17(1-3), pp. 35–75, doi:10.1016/0167-6423(91)90036-W.

[29] W. J. Fokkink (2000): *Introduction to Process Algebra*. Texts in Theoretical Computer Science, An EATCS Series, Springer, doi:10.1007/978-3-662-04293-9.

[30] W.J. Fokkink, R.J. van Glabbeek & B. Luttik (2017): *Divide and Congruence III: Stability & Divergence*. In R. Meyer & U. Nestmann, editors: Proceedings 28th International Conference on *Concurrency Theory*, CONCUR 2017, *Leibniz International Proceedings in Informatics (LIPIcs)* 85, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 15:1–15:16, doi:10.4230/LIPIcs.CONCUR.2017.15.

[31] Y. Fu (2016): *Theory of interaction*. *Theoretical Computer Science* 611, pp. 1–49, doi:10.1016/j.tcs.2015.07.043.

[32] H. Garavel, F. Lang, R. Mateescu & W. Serwe (2011): *CADP 2010: A Toolbox for the Construction and Analysis of Distributed Processes*. In P.A. Abdulla & K.R.M. Leino, editors: Proceedings *Tools and Algorithms for the Construction and Analysis of Systems*, TACAS '11, LNCS 6605, Springer, pp. 372–387, doi:10.1007/978-3-642-19835-9_33.

[33] T. Given-Wilson (2014): *Expressiveness via Intensionality and Concurrency*. In G. Ciobanu & D. Méry, editors: Proceedings 11th International Colloquium on *Theoretical Aspects of Computing*, ICTAC'14, Bucharest, Romania, 2014, LNCS 8687, Springer, pp. 206–223, doi:10.1007/978-3-319-10882-7_13.

[34] T. Given-Wilson (2014): *On the Expressiveness of Intensional Communication*. In J. Borgström & S. Crafa, editors: Proceedings Combined 21st International Workshop on *Expressiveness in Concurrency* and 11th Workshop on *Structural Operational Semantics*, Rome, Italy, September 2014, *Electronic Proceedings in Theoretical Computer Science* 160, Open Publishing Association, pp. 30–46, doi:10.4204/EPTCS.160.4.

[35] T. Given-Wilson & A. Legay (2015): *On the Expressiveness of Joining*. In S. Knight, I. Lanese, A. Lluch Lafuente & H. Torres Vieira, editors: Proceedings 8th *Interaction and Concurrency Experience*, Grenoble, France, 4-5th June 2015, *Electronic Proceedings in Theoretical Computer Science* 189, Open Publishing Association, pp. 99–113, doi:10.4204/EPTCS.189.9.

[36] T. Given-Wilson & A. Legay (2016): *On the Expressiveness of Symmetric Communication*. In A. Sampaio & F. Wang, editors: Proceedings 13th International Colloquium on *Theoretical Aspects of Computing*, ICTAC'16, LNCS 9965, Springer, pp. 139–157, doi:10.1007/978-3-319-46750-4_9.

[37] Th. Given-Wilson & A. Legay (2019): *On the Expressiveness of Joining and Splitting*. In T. Margaria, S. Graf & K.G. Larsen, editors: *Models, Mindsets, Meta: The What, the How, and the Why Not? - Essays Dedicated to Bernhard Steffen on the Occasion of His 60th Birthday*, LNCS 11200, Springer, pp. 326–355, doi:10.1007/978-3-030-22348-9_20.

[38] R.J. van Glabbeek (1993): *Full Abstraction in Structural Operational Semantics (extended abstract)*. In M. Nivat, C. Rattray, T. Rus & G. Scollo, editors: Proceedings of the 3^{rd} International Conference on *Algebraic Methodology and Software Technology*, AMAST'93, Workshops in Computing, Springer, pp. 75–82. Available at http://theory.stanford.edu/~rvg/abstracts.html#28.

[39] 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*, ACP'94, Workshops in Computing, Springer, pp. 188–217, doi:10.1007/978-1-4471-2120-6_8.

[40] R.J. van Glabbeek (2005): *A Characterisation of Weak Bisimulation Congruence*. In A. Middeldorp, V. van Oostrom, F. van Raamsdonk & R. de Vrijer, editors: *Processes, Terms and Cycles: Steps on the Road to Infinity: Essays Dedicated to Jan Willem Klop on the Occasion of His 60th Birthday*, LNCS 3838, Springer, pp. 26–39, doi:10.1007/11601548_4.

[41] R.J. van Glabbeek (2011): *On Cool Congruence Formats for Weak Bisimulations*. *Theoretical Computer Science* 412(28), pp. 3283–3302, doi:10.1016/j.tcs.2011.02.036. Available at http://theory.stanford.edu/~rvg/abstracts.html#88.

[42] R.J. van Glabbeek (2012): *Musings on Encodings and Expressiveness*. In B. Luttik & M.A. Reniers, editors: Proceedings Combined 19th International Workshop on *Expressiveness in Concurrency* and 9th Workshop on *Structured Operational Semantics*, Newcastle upon Tyne, UK, September 2012, *Electronic Proceedings in Theoretical Computer Science* 89, Open Publishing Association, pp. 81–98, doi:10.4204/EPTCS.89.7.

[43] R.J. van Glabbeek (2017): *Lean and Full Congruence Formats for Recursion*. In: Proceedings 32nd Annual ACM/IEEE Symposium on *Logic in Computer Science*, LICS'17, Reykjavik, Iceland, June 2017, IEEE Computer Society Press, doi:10.1109/LICS.2017.8005142.

[44] R.J. van Glabbeek (2018): *On the Validity of Encodings of the Synchronous in the Asynchronous π-calculus*. *Information Processing Letters* 137, p. 17–25, doi:10.1016/j.ipl.2018.04.015.

[45] R.J. van Glabbeek (2018): *A Theory of Encodings and Expressiveness*. Technical Report, Data61, CSIRO. Available at https://arxiv.org/abs/1805.10415v1. Previous version of the present paper.

[46] R.J. van Glabbeek (2018): *A Theory of Encodings and Expressiveness (extended abstract)*. In C. Baier & U. Dal Lago, editors: Proc. 21st International Conference on *Foundations of Software Science and Computational Structures*, FoSSaCS 2018; held as part of the *European Joint Conferences on Theory and Practice of Software*, ETAPS 2018, LNCS 10803, Springer, pp. 183–202, doi:10.1007/978-3-319-89366-2_10.

[47] R.J. van Glabbeek (2019): *On the Meaning of Transition System Specifications*. In J.A. Pérez & J. Rot, editors: Proceedings Combined 26th International Workshop on *Expressiveness in Concurrency* and 16th Workshop on *Structural Operational Semantics*, *Electronic Proceedings in Theoretical Computer Science* 300, Open Publishing Association, pp. 69–85, doi:10.4204/EPTCS.300.5.

[48] R.J. van Glabbeek (2022): *Comparing the expressiveness of the π-calculus and CCS*. In I. Sergey, editor: *Programming Languages and Systems:* Proceedings 31st *European Symposium on Programming*, ESOP'22; held as part of the *European Joint Conferences on Theory and Practice of Software*, ETAPS'22, LNCS 13240, Springer, p. 548–574, doi:10.1007/978-3-030-99336-8_20.

[49] R.J. van Glabbeek, U. Goltz, C. Lippert & S. Mennicke (2019): *Stronger Validity Criteria for Encoding Synchrony*. In M.S. Alvim, K. Chatzikokolakis, C. Olarte & F. Valencia, editors: *The Art of Modelling Computational Systems: A Journey from Logic and Concurrency to Security and Privacy - Essays Dedicated to Catuscia Palamidessi on the Occasion of Her 60th Birthday*, LNCS 11760, Springer, pp. 182–205, doi:10.1007/978-3-030-31175-9_11.

[50] R.J. van Glabbeek, B. Luttik & N. Trčka (2009): *Branching Bisimilarity with Explicit Divergence*. *Fundamenta Informaticae* 93(4), pp. 371–392, doi:10.3233/FI-2009-109.

[51] R.J. van Glabbeek & W.P. Weijland (1996): *Branching Time and Abstraction in Bisimulation Semantics*. *Journal of the ACM* 43(3), pp. 555–600, doi:10.1145/233551.233556.

[52] D. 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.

[53] D. 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.

[54] D. Gorla & U. Nestmann (2016): *Full abstraction for expressiveness: history, myths and facts*. *Mathematical Structures in Computer Science* 26(4), pp. 639–654, doi:10.1017/S0960129514000279.

[55] G. Grätzer (2010): *Lattice Theory: Foundation*. Springer, doi:10.1007/978-3-0348-0018-1.

[56] J.F. Groote & M.R. Mousavi (2014): *Modeling and Analysis of Communicating Systems*. MIT Press, doi:10.7551/mitpress/9946.001.0001.

[57] B. Haagensen, S. Maffeis & I. Phillips (2008): *Matching Systems for Concurrent Calculi*. *Electronic Notes in Theoretical Computer Science* 194(2), pp. 85–99, doi:10.1016/j.entcs.2007.11.004.

[58] M. Hatzel, C. Wagner, K. Peters & U. Nestmann (2015): *Encoding CSP into CCS*. In S. Crafa & D.E. Gebler, editors: Proceedings of the Combined 22th International Workshop on *Expressiveness in Concurrency* and 12th Workshop on *Structural Operational Semantics*, *Electronic Proceedings in Theoretical Computer Science* 190, Open Publishing Association, pp. 61–75, doi:10.4204/EPTCS.190.5.

[59] M. Hatzel, Chr. Wagner, K. Peters & U. Nestmann (2015): *Encoding CSP into CCS (Extended Version)*. arXiv:1508.01127.

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

[61] K. Honda & M. Tokoro (1991): *An Object Calculus for Asynchronous Communication*. In Pierre America, editor: Proc. *European Conference on Object-Oriented Programming*, ECOOP'91, LNCS 512, Springer, pp. 133–147, doi:10.1007/BFb0057019.

[62] C.T. Jensen (1994): *Interpreting Broadcast Communication in CCS with Priority Choice*. In: *Proceedings of the 6th Nordic Workshop on Programming Theory*, pp. 49–70.

[63] D. Kouzapas, J.A. Pérez & N. Yoshida (2019): *On the relative expressiveness of higher-order session processes*. *Information and Computation* 268, doi:10.1016/j.ic.2019.06.002.

[64] I. Lanese, J.A. Pérez, D. Sangiorgi & A. Schmitt (2010): *On the Expressiveness of Polyadic and Synchronous Communication in Higher-Order Process Calculi*. In S. Abramsky, C. Gavoille, C. Kirchner, F. Meyer auf der Heide & P.G. Spirakis, editors: Proceedings 37th International Colloquium on *Automata, Languages and Programming*, ICALP'10, Part II, LNCS 6199, Springer, pp. 442–453, doi:10.1007/978-3-642-14162-1_37.

[65] Bas Luttik (2003): *On the expressiveness of choice quantification*. *Ann. Pure Appl. Log.* 121(1), pp. 39–87, doi:10.1016/S0168-0072(02)00082-9.

[66] M. Merro & D. Sangiorgi (2004): *On asynchrony in name-passing calculi*. *Mathematical Structures in Computer Science* 14(5), pp. 715–767, doi:10.1017/S0960129504004323.

[67] R. Milner (1975): *Processes: A Mathematical Model for Computing Agents*. In H.E. Rose & J.C. Shepherdson, editors: *Logic Colloquium '73*, *Studies in Logic and the Foundations of Mathematics* 50, Elsevier, pp. 157–173, doi:10.1016/S0049-237X(08)71948-7.

[68] R. Milner (1983): *Calculi for synchrony and asynchrony*. *Theoretical Computer Science* 25, pp. 267–310, doi:10.1016/0304-3975(83)90114-7.

[69] R. Milner (1989): *Communication and Concurrency*. Prentice Hall, Englewood Cliffs.

[70] 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.

[71] R. Milner (1992): *Functions as Processes*. *Mathematical Structures in Computer Science* 2(2), pp. 119–141, doi:10.1017/S0960129500001407.

[72] R. Milner, J. Parrow & D. Walker (1992): *A Calculus of Mobile Processes, I*. *Information and Computation* 100(1), pp. 1–40, doi:10.1016/0890-5401(92)90008-4.

[73] R. Milner, J. Parrow & D. Walker (1992): *A Calculus of Mobile Processes, II*. *Information and Computation* 100(1), pp. 41–77, doi:10.1016/0890-5401(92)90009-5.

[74] R. Milner & D. Sangiorgi (1992): *Barbed Bisimulation*. In W. Kuich, editor: Proceedings 19th International Colloquium on *Automata, Languages and Programming*, ICALP'92, Vienna, Austria, July 1992, LNCS 623, Springer, pp. 685–695, doi:10.1007/3-540-55719-9_114.

[75] J.C. Mitchell (1993): *On Abstraction and the Expressive Power of Programming Languages*. *Science of Computer Programming* 21(2), pp. 141–163, doi:10.1016/0167-6423(93)90004-9. A earlier version was circulated in 1986 under the name *Lisp is not universal*.

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

[77] 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.

[78] U. Nestmann & B.C. Pierce (2000): *Decoding Choice Encodings*. *Information and Computation* 163(1), pp. 1–59, doi:10.1006/inco.2000.2868. An earlier version appeared in Proc. CONCUR'96.

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

[80] C. Palamidessi (2003): *Comparing The Expressive Power of the Synchronous and Asynchronous π-Calculi*. *Mathematical Structures in Computer Science* 13(5), pp. 685–719, doi:10.1017/S0960129503004043.

[81] 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'06, IEEE Computer Society Press, pp. 59–68, doi:10.1109/LICS.2006.39.

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

[83] 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.

[84] J. Parrow (2008): *Expressiveness of Process Algebras*. *Electronic Notes in Theoretical Computer Science* 209, pp. 173–186, doi:10.1016/j.entcs.2008.04.011.

[85] J. Parrow (2016): *General conditions for full abstraction*. *Mathematical Structures in Computer Science* 26(4), pp. 655–657, doi:10.1017/S0960129514000280.

[86] K. Peters (2012): *Translational Expressiveness. Comparing Process Calculi using Encodings*. Ph.D. thesis, TU Berlin, doi:10.14279/depositonce-3416.

[87] K. Peters (2019): *Comparing Process Calculi Using Encodings*. In Jorge A. Pérez & Jurriaan Rot, editors: Proceedings Combined 26th International Workshop on *Expressiveness in Concurrency* and 16th Workshop on *Structural Operational Semantics*, *Electronic Proceedings in Theoretical Computer Science* 300, Open Publishing Association, pp. 19–38, doi:10.4204/EPTCS.300.2.

[88] K. Peters & R.J. van Glabbeek (2015): *Analysing and Comparing Encodability Criteria*. In S. Crafa & D.E. Gebler, editors: Proceedings of the Combined 22th International Workshop on *Expressiveness in Concurrency* and 12th Workshop on *Structural Operational Semantics*, *Electronic Proceedings in Theoretical Computer Science* 190, Open Publishing Association, pp. 46–60, doi:10.4204/EPTCS.190.4.

[89] 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'12; held as part of the *European Joint Conferences on Theory and Practice of Software*, ETAPS'12, LNCS 7213, Springer, pp. 210–224, doi:10.1007/978-3-642-28729-9_14.

[90] K. Peters & U. Nestmann (2016): *Breaking symmetries*. *Mathematical Structures in Computer Science* 26(6), pp. 1054–1106, doi:10.1017/S0960129514000346.

[91] K. Peters, U. Nestmann & U. Goltz (2013): *On Distributability in Process Calculi*. In M. Felleisen & Ph. Gardner, editors: *Programming Languages and Systems:* Proceedings 22nd *European Symposium on Programming*, ESOP'13, held as part of the European Joint Conferences on *Theory and Practice of Software*, ETAPS'13, LNCS 7792, Springer, pp. 310–329, doi:10.1007/978-3-642-37036-6_18.

[92] 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.

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

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

[95] G.D. Plotkin (2004): *A Structural Approach to Operational Semantics*. *The Journal of Logic and Algebraic Programming* 60–61, pp. 17–139, doi:10.1016/j.jlap.2004.05.001. Originally appeared in 1981.

[96] P. Quaglia & D. Walker (2000): *On Synchronous and Asynchronous Mobile Processes*. In J. Tiuryn, editor: Proceedings Third International Conference on *Foundations of Software Science and Computation Structures*, FoSSaCS'00, held as part of the Joint European Conferences on *Theory and Practice of Software*, ETAPS'00, LNCS 1784, Springer, pp. 283–296, doi:10.1007/3-540-46432-8_19.

[97] J.G. Riecke (1991): *Fully Abstract Translations between Functional Languages*. In D.S. Wise, editor: Conference Record of the Eighteenth Annual ACM Symposium on *Principles of Programming Languages*, POPL'91, Orlando, Florida, USA, January 1991, ACM Press, pp. 245–254, doi:10.1145/99583.99617.

[98] A.W. Roscoe (2010): *CSP is Expressive Enough for π*. In: *Reflections on the Work of C.A.R. Hoare*, Springer, pp. 371–404, doi:10.1007/978-1-84882-912-1_16.

[99] D. Sangiorgi (1993): *From π-calculus to Higher-Order π-calculus — and back*. In M.-C. Gaudel & J.-P. Jouannaud, editors: Proc. *Theory and Practice of Software Development*, TAPSOFT'93, International Joint Conference CAAP/FASE, LNCS 668, Springer, pp. 151–166, doi:10.1007/3-540-56610-4_62.

[100] D. Sangiorgi & D. Walker (2001): *On Barbed Equivalences in π-Calculus*. In K.G. Larsen & M. Nielsen, editors: Proceeding 12th International Conference on *Concurrency Theory*, CONCUR'01, Aalborg, Denmark, August 2001, LNCS 2154, Springer, pp. 292–304, doi:10.1007/3-540-44685-0_20.

[101] D. Sangiorgi & D. Walker (2001): *The π-calculus: A Theory of Mobile Processes*. Cambridge University Press.

[102] A. Schmitt, K. Peters & Y. Deng (2022): *Encodability Criteria for Quantum Based Systems*. In M.R. Mousavi & A. Philippou, editors: Proc. International Conference on *Formal Techniques for Distributed Objects, Components, and Systems* – 42nd IFIP WG 6.1, FORTE 2022, held as part of the 17th International Federated Conference on *Distributed Computing Techniques*, DisCoTec 2022, Lucca, Italy, June 2022, LNCS 13273, Springer, pp. 151–169, doi:10.1007/978-3-031-08679-3_10.

[103] E.Y. Shapiro (1991): *Separating Concurrent Languages with Categories of Language Embeddings (Extended Abstract)*. In C. Koutsougeras & J.S. Vitter, editors: Proceedings 23rd Annual ACM Symposium on *Theory of Computing*, STOC'91, ACM, pp. 198–208, doi:10.1145/103418.103423.

[104] E.Y. Shapiro (1992): *Embeddings Among Concurrent Programming Languages (Preliminary Version)*. In R. Cleaveland, editor: Proceedings Third International Conference on *Concurrency Theory*, CONCUR '92, *Stony Brook, NY, USA, August 1992*, LNCS 630, Springer, pp. 486–503, doi:10.1007/BFb0084811.

[105] R. de Simone (1984): *Calculabilité et Expressivité dans l'Algebra de Processus Parallèles* MEIJE. Thèse de 3^e cycle, Univ. Paris 7.

[106] 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.

[107] 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, 1992, LNCS 666, Springer, pp. 609–638, doi:10.1007/3-540-56596-5_49.

[108] A. Valmari (2015): *On constructibility and unconstructibility of LTS operators from other LTS operators*. *Acta Informatica* 52(2-3), pp. 207–234, doi:10.1007/s00236-015-0217-2.

[109] 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.

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

[111] J.L.M. Vrancken (1997): *The Algebra of Communicating Processes With Empty Process*. *Theoretical Computer Science* 177(2), pp. 287–328, doi:10.1016/S0304-3975(96)00250-2.

17 Overview of notation (Skipping notation that is used only in the (sub)section in which it is introduced)

$\mathbb{N}$	set of natural numbers	general mathematics
$\uplus$	disjoint union	general mathematics
$\restriction$	restriction	general mathematics
$\mathbf{R}^{-1}$	inverse of the binary relation $\mathbf{R}$	general mathematics
$\circ$	functional and relational composition	Sects. 3.3 and 11.6
$\mathscr{L}$	a language	Section 2
$\mathbb{T}_{\mathscr{L}}$	set of open terms or expressions in the language $\mathscr{L}$	Section 2
$\mathrm{T}_{\mathscr{L}}$	set of closed terms in the language $\mathscr{L}$	Section 2
$\mathscr{D}_{\mathscr{L}}$	set of meanings expressible in language $\mathscr{L}$	Section 2
$[\![\ ]\!]_{\mathscr{L}} : \mathbb{T}_{\mathscr{L}} \to \mathscr{D}_{\mathscr{L}}$	semantic mapping from expressions to their meanings	Section 2
$[\![\sigma]\!]_{\mathscr{L}}$	denotation of the substitution σ	Section 11.3
$\mathscr{T} : \mathbb{T}_{\mathscr{L}} \to \mathbb{T}_{\mathscr{L}'}$	*translation* or *encoding*, mapping open terms from source language $\mathscr{L}$ to target $\mathscr{L}'$	
$\mathscr{X}$	an infinite set of variables, fixed for this paper	Section 2
$fv(E)$	set of variables occurring free in the expression E	Section 2
$\mu X.E$	recursion construct	Section 2
$\sum_{i\in I} E_i$	infinite summation of CCS	Section 2
$\sum_{h\in\mathscr{H}} E[f_h]$	variant of choice quantification	Section 2
$\mathscr{L}[\mathscr{X}_n]$	extension of language $\mathscr{L}$ with n so-called *holes*	Section 3.1
$C[E_1,\ldots,E_n]$	The result of substituting E_i for the hole X_i in the context C	Section 3.1
$E[\sigma]$	The result of applying the substitution σ to the expression E	Definition 6.3, 10.1
$\mathscr{T}(\mathscr{L})$	the image of language $\mathscr{L}$ under the translation $\mathscr{T}$	informal, see below
$\mathrm{T}_{\mathscr{T}(\mathscr{L})}$	the closed terms in $\mathscr{L}'$ that are translations of ones from $\mathscr{L}$	Section 3.2
$\mathrel{\rotatebox{180}{$\precsim$}}$	a semantic equivalence or preorder	Section 3.2
$\mathrel{\rotatebox{180}{$\precapprox$}}$	a coarser semantic equivalence or preorder	Section 3.3
$\sim$	a semantic equivalence relation	Section 3.4
$\approx$	a coarser semantic equivalence relation	Section 3.4
$\approx^c_{\mathscr{L}}$	congruence closure of $\approx$ on $\mathscr{L}$	Section 3.6
$\approx^c_{\mathscr{T}(\mathscr{L})}$	congruence closure of $\approx$ on $\mathscr{T}(\mathscr{L})$	Section 3.6
$\approx^c_{\mathscr{T}(\mathscr{L}),\mathbf{W}}$	congruence closure of $\approx$ w.r.t. $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$	Section 9.2
$m\pi$	the fragment of the π-calculus studied in [71]	Section 4
$a\pi$	the asynchronous fragment of $m\pi$ studied in [61, 14]	Section 4
$\mathcal{N}$	set of names in the π-calculus	Section 4, 14.7
$\mathsf{fn}(P)$	free names occurring in a π-calculus process P	Section 14.7
$\mathsf{bn}(P)$	bound names occurring in a π-calculus process P	Definition 4.1
$\mathsf{n}(P)$	all names occurring in a π-calculus process P	Definition 4.1

$P \to P'$	reduction relation between processes	Def. 4.3, Sect. 13.2
$P \xrightarrow{a} P'$	labelled transition between processes	Section 14
$P{\downarrow_\omega}$	π-calculus process $P \in \mathrm{T}_\pi$ has a strong barb ω	Def. 4.3, Sect. 13.2
$P{\Downarrow_\omega}$	π-calculus process $P \in \mathrm{T}_\pi$ has a weak barb ω	Definition 4.3
$\dot{\sim} \subseteq \mathrm{T}_\pi \times \mathrm{T}_\pi$	strong barbed bisimilarity	Definition 13.1
$\dot{\approx} \subseteq \mathrm{T}_{m\pi} \times \mathrm{T}_{m\pi}$	weak barbed bisimilarity	Definition 4.4
$\cong^c \subseteq \mathrm{T}_{m\pi} \times \mathrm{T}_{m\pi}$	weak barbed congruence, the congruence closure of $\dot{\approx}$	Definition 4.4
$\cong_a^c \subseteq \mathrm{T}_{a\pi} \times \mathrm{T}_{a\pi}$	asynchronous barbed congr., the congr. closure of $\dot{\approx}$ on $\mathrm{T}_{a\pi}$	Section 4
$\preceq$	ordering by strength on validity requirements on translations	Section 7
$\mathbf{R} \subseteq \mathrm{T}_{\mathscr{L}'} \times \mathrm{T}_{\mathscr{L}}$	$\mathbf{R} \subseteq \mathbf{V}' \times \mathbf{V}$ semantic translation	Definition 7.3, 11.1
$\mathbf{R}(\mathscr{L})$	semantic counterpart of $\mathrm{T}_{\mathscr{L}}$ within $\mathrm{T}_{\mathscr{L}'}$	Section 9
$\sim^1$	1-point lifting of the equivalence relation $\sim$ to substitutions	Section 9.2
$\equiv_{\mathbf{R}}$	smallest equiv. on $\mathrm{T}_{\mathscr{L}} \uplus \mathbf{R}(\mathscr{L})$ or $\mathbf{V} \uplus \mathbf{R}(\mathbf{V})$ containing $\mathbf{R}$	Definition 9.11, 11.22
$\overset{\alpha}{=}$	α-convertibility	Definition 10.2
$\overset{\alpha}{=}_{\mathscr{L}}$	semantic counterpart of α-convertibility	Definition 11.4
$\mathbf{R}^\alpha$	closure of the relation $\mathbf{R}$ under α-conversion	Section 10.6, 11.2
$\bullet$	composition operator for substitutions	Definition 10.5
$\mathbb{H}$, *decomp*	set of standard heads, and decomposition function	Proposition 10.12
$\approx_{\mathscr{L}}^{1c}$	[1-hole] congruence closure of $\approx$ on $\mathscr{L}$	Section 10.9
$\approx_{\mathscr{T}(\mathscr{L}),\mathbf{W}}^{1c}$	[1-hole] congruence closure of $\approx$ w.r.t. $\mathscr{T}(\mathscr{L})$ on $\mathbf{W}$	Section 10.9
$\mathbf{V}, \mathbf{V}'$	semantic domains of the source/target languages $\mathscr{L}$ and $\mathscr{L}'$	Section 11
$\mathbf{R}(\mathbf{V})$	semantic counterpart of $\mathbf{V}$ within $\mathbf{V}'$	Section 11.7