Information and Computation **156**, 345–385 (2000)
doi:10.1006/inco.1999.2818, available online at http://www.idealibrary.com on IDEAL®

Bisimilarity of Open Terms[1]

Arend Rensink

Faculty of Informatics, University of Twente, Postbus 217, 7500 AE Enschede, The Netherlands
E-mail: rensink@cs.utwente.nl

Traditionally, in process calculi, relations over open terms, i.e., terms with free process variables, are defined as extensions of closed-term relations: two open terms are related if and only if all their closed instantiations are related. Working in the context of bisimulation, in this paper we study a different approach; we define semantic models for open terms, so-called *conditional transition systems*, and define bisimulation directly on those models. It turns out that this can be done in at least two different ways, one giving rise to De Simone's *formal hypothesis* bisimilarity and the other to a variation which we call *hypothesis-preserving* bisimilarity (denoted $\sim^{fh}$ and $\sim^{hp}$, respectively). For open terms, we have (strict) inclusions $\sim^{fh} \subset \sim^{hp} \subset \sim^{ci}$ (the latter denoting the standard "closed instance" extension); for closed terms, the three coincide. Each of these relations is a congruence in the usual sense. We also give an alternative characterisation of $\sim^{hp}$ in terms of nonconditional transitions, as *substitution-closed* bisimilarity (denoted $\sim^{sb}$). Finally, we study the issue of *recursion congruence*: we prove that each of the above relations is a congruence with respect to the recursion operator; however, for $\sim^{ci}$ this result holds under more restrictive conditions than for $\sim^{fh}$ and $\sim^{hp}$.

1. INTRODUCTION

An important mathematical tool for reasoning about the implementation and verification of systems is that of an *implementation relation*. This is a binary relation between system descriptions expressing that (the behaviour specified by) one description is a valid implementation of (the behaviour described by) another. The criteria for implementation relations generally include at least reflexivity (so every system implements itself) and transitivity (so implementation steps can be combined), meaning that they are preorders; in many cases they are also symmetric and, hence, are in fact equivalence relations.

[1] This work has been partially supported by the European HCM network EXPRESS (Expressiveness of Languages for Concurrency), while the author was employed at the University of Hildesheim.

0890-5401/00 $35.00

The "system descriptions" on which implementation relations are based are models of system behaviour. By far the most popular such models are *labelled transition systems* (LTSs), which are directed, edge-labelled graphs whose nodes correspond to the states of the system (with a distinguished node representing the start state) and whose edges represent system (inter)actions. An overview of LTS-based implementation relations has been drawn up by Van Glabbeek in [20, 22]. If the class of labelled transition systems is given by $\mathbf{T}$, a typical implementation relation can be denoted $\lesssim \subseteq \mathbf{T} \times \mathbf{T}$.

On the other hand, in practice one would prefer to specify and reason about systems not on the level of their behaviour models, but rather through a more abstract and readable *language* of some kind. Formally, such a language is given by the term algebra $\mathbb{T}_\Sigma$ generated by a signature Σ; the underlying behaviour models are given by a mapping from $\mathbb{T}_\Sigma$ to the semantic domain. For instance, if we are using LTSs, we can write $[\![t]\!] \in \mathbf{T}$ to denote the model corresponding to $t \in \mathbb{T}_\Sigma$. Typically, the semantic mapping could be defined through a set of *structural operational rules* that define the operational intention of the operators in Σ; this gives rise to an LTS in which the states are terms in $\mathbb{T}_\Sigma$. An implementation relation $\lesssim$ can then be lifted from $\mathbf{T}$ to $\mathbb{T}_\Sigma$ by defining

$$t \lesssim u \Leftrightarrow [\![t]\!] \lesssim [\![u]\!]$$

for arbitrary $t, u \in \mathbb{T}_\Sigma$. This imposes a further criterion on implementation relations: namely, they had better be (pre-)*congruences* with respect to the operators of the signature, where $\lesssim$ is said to be a precongruence w.r.t. an n-ary operator $op \in \Sigma$ if $\forall 1 \leqslant i \leqslant n.\ t_i \lesssim u_i$ implies $op(t_1, ..., t_n) \lesssim op(u_1, ..., u_n)$.

A further step is to extend the term algebra with *term variables*, giving rise to a language $\mathbb{T}_\Sigma(V)$, where V is the universe of variables. Term variables are used for at least two different purposes: first, to allow reasoning on the level of the language, for instance using (in)equational proof systems for implementation relations or structural rules for operational semantics; and second, to define higher-order language constructors, in the form of *binders* $bnd(x, \_)$ for every $x \in V$. The best known binder in process calculi is the *recursion operator*, $bnd(x, \_) = \text{rec } x.\_$, which recursively binds x to its operand. Furthermore, recently there has been a growing interest in *higher-order calculi*, featuring functional binders that bind variables to actual parameters to be provided by application or communication (see, e.g., [43]).

The primary operation on free (i.e., nonbound) term variables is their *substitution* by an actual term. We use $t[u/x]$ to denote the replacement, within $t \in \mathbb{T}_\Sigma(V)$, of every occurrence of the variable $x \in V$ by (a copy of) $u \in \mathbb{T}_\Sigma(V)$. (This notion of substitution is modified by the presence of binders $bnd(y, t')$ in t: the y-occurrences in t' cannot be replaced by substitution, and moreover, substitution should not cause y-occurrences in u to become bound. These problems are well documented in the context of the λ-calculus; see [6].) Two typical substitution-based rules of an inequational proof system are

$$\frac{t \sqsubseteq u}{v[t/x] \sqsubseteq v[u/x]}, \quad \frac{t \sqsubseteq u}{t[v/x] \sqsubseteq u[v/x]}.$$

The left-hand side expresses *preservation by insertion* of the relation $\sqsubseteq$, which is quite close to the property of being a precongruence mentioned above (in the presence of binders, the two notions are not quite the same, due to the problems described before); the right-hand side expresses *preservation by instantiation*. We call a relation *substitutive* if it is preserved by both insertion and instantiation.

In the usual setup, no model in $\mathbf{T}$ exists for open terms $t, u \in \mathbb{T}_\Sigma(V)$; therefore, t and u are not subject to direct comparison by any implementation relation over $\mathbf{T}$ lifted to $\mathbb{T}_\Sigma$ in the standard way recalled above. (An alternative setup can be found in [32, 21, 15], where an operational semantics is defined directly over open terms wherein the term variables are treated as actions; i.e., they may appear as transition labels. Unfortunately, this approach is only viable as long as the process calculus under consideration contains no operator for parallel composition.) On the other hand, intuitively, for an implementation relation to make sense at all for open terms, it should at least be preserved by instantiation; in other words, this can be regarded as a necessary criterion for an extension of $\lesssim$ to $\mathbb{T}_\Sigma(V)$. Now if we substitute every free variable x in t and u by a closed term $v_x \in \mathbb{T}_\Sigma$, the resulting terms t' and u' are also closed and thus do have models $[\![t']\!], [\![u']\!] \in \mathbf{T}$. They are therefore subject to $\lesssim$-comparison. Hence, preservation by instantiation implies that t and u may only be related by (an extension of) $\lesssim$ if $t' \lesssim u'$ for all possible choices of the v_x. In fact, the standard way to extend $\lesssim$ to open terms is to turn this necessary criterion into a sufficient condition; that is, to define $t \lesssim u$ for $t, u \in \mathbb{T}_\Sigma(V)$ if and only if $t' \lesssim u'$ for all closed instantiations $t', u' \in \mathbb{T}_\Sigma$ of t, u.

In this paper, we follow a different approach: we extend the class of models $\mathbf{T}$ instead, so as to include term variables explicitly on the level of the semantics, giving rise to an extended model class $\mathbf{T}(V)$. Hence, for all $t \in \mathbb{T}_\Sigma(V)$ there is a model $[\![t]\!] \in \mathbf{T}(V)$. We then define relations $\lesssim$ on the extended model class $\mathbf{T}(V)$ and lift these to $\mathbb{T}_\Sigma(V)$ as before. Since preservation by instantiation is still a necessary criterion, but no longer sufficient, these alternative $\lesssim$-extensions can then in principle be stricter than the standard one.

The paper proceeds as follows. We first define an abstract notion of substitution, in the form of *substitution systems* and then use this to define the class $\mathbf{T}(V)$ of *conditional transition systems* (CTSs); see Section 2 below. Again, one particular way to generate a conditional transition system is by defining a number of operational rules over a given signature Σ; the states of the CTS are then open terms in $\mathbb{T}_\Sigma(V)$. In Section 3, we then study the implementation relation of *bisimilarity* (which is, in fact, an equivalence rather than merely a preorder) over conditional transition systems. Apart from *closed-instance* bisimilarity ($\sim^{\mathsf{ci}}$), which is obtained through the standard approach outlined above, we define two stricter notions: *formal hypothesis* and *hypothesis preserving* bisimilarity (respectively $\sim^{\mathsf{fh}}$ and $\sim^{\mathsf{hp}}$). (The former is originally due to De Simone [14], where it is studied in a restricted setting in which it actually coincides with the latter.) We prove that each of the three bisimilarity relations is preserved by insertion and instantiation. We also give an alternative characterisation of $\sim^{\mathsf{hp}}$ in terms of the standard, nonconditional semantics, as *substitution closed bisimilarity* ($\sim^{\mathsf{sb}}$), which is defined by requiring that the underlying bisimulation relation (and not just the resulting bisimilarity) is preserved by instantiation, in addition to the usual matching criteria. We feel

that this alternative characterisation provides strong evidence for the viability of $\sim^{\mathsf{hp}}$.

We then turn to the recursion operator. A relation is called a *recursion (pre)-congruence* if it satisfies the proof rule:

$$\frac{t \sqsubseteq u}{\operatorname{rec} x.t \sqsubseteq \operatorname{rec} x.u}.$$

Note that the terms in the premise are typically open. As indicated above, this is not implied by substitutivity; it is not the case that $\operatorname{rec} x.t = v[t/y]$ and $\operatorname{rec} x.u = v[u/y]$ for some v—in particular, $v = \operatorname{rec} x.y$ is not satisfactory. In Section 4, we show that $\sim^{\mathsf{ci}}$ is a recursion congruence in a subclass of $\mathbf{T}(V)$, whereas both of the stricter forms of bisimilarity, $\sim^{\mathsf{fh}}$ and $\sim^{\mathsf{hp}}$, are always recursion congruences.

Section 5 contains a comparison of this paper's results with, among others, the work on *SOS formats* by De Simone [14], Bloom, Istrail, and Meyer [8], and Groote and Vaandrager [25], on *contexts* by Larsen and Xinxin [30], on bisimilarity in functional programming by Howe and others [28, 24, 41] and in higher-order calculi by Sangiorgi [42].

2. CONDITIONAL OPERATIONAL SEMANTICS

The principal idea underlying the developments in this paper is to regard the operational semantics of a process calculus not through the usual transition predicates of the form $t \xrightarrow{\alpha} t'$, where t and t' are (usually closed) terms of the calculus and α is some label from a predefined universe, but rather through enriched *conditional* transition predicates of the form $\Delta \vdash t \xrightarrow{\alpha} t'$, where t and t' are *open* terms and Δ is an *operational environment* containing information about their free variables.

We first discuss conditional transitions on an informal level; then we define the concept of substitution and, based on that, the concept of a conditional transition system. Furthermore, we show that the usual structural operational rules can be used to generate conditional transition systems.

2.1. *Notation.* • We use $A \subseteq_{\mathrm{Fin}} B$ to denote that A is a finite subset of B, and $\mathrm{Fin}(B)$ to denote the set of all finite subsets of B.

• If $A \subseteq B$, we use $A \to_{\mathrm{Fin}} B$ to denote the space of functions $f: A \to B$ that are *the identity almost everywhere*, i.e., such that $f(a) \neq a$ for only a finite subset $a \in A' \subseteq_{\mathrm{Fin}} A$. If $f: A \to_{\mathrm{Fin}} B$ such that $f(a) = a$ for all $a \notin \{a_1, ..., a_n\}$, we also write $f = \{f(a_1)/a_1, ..., f(a_n)/a_n\}$ or $f = \{f(a_i)/a_i\}_{i \in \{1, ..., n\}}$.

• A function $f: A \to B$ can be *updated* as

$$f \pm \{b_1/a_1, ..., b_n/a_n\} : x \mapsto \begin{cases} b_i, & \text{if } x = a_i, \\ f(x), & \text{if } x \notin \{a_1, ..., a_n\}. \end{cases}$$

Note that $(f \pm \{b_1/a_1, ..., b_n/a_n\}) : A \to_{\mathrm{Fin}} B$ if $f: A \to_{\mathrm{Fin}} B$.

- If $A \subseteq B$, we use $id: A \to_{\text{Fin}} B$ to denote the identity function, i.e., such that $id(a) = a$ for all $a \in A$.
- V denotes a denumerable universe of variables.

2.1. Conditional Transitions

For the moment, we assume that the concepts of a term (over a given signature) and a transition between terms are known. We draw examples from CCS (see Milner [33]). If one considers transitions between open terms, instead of restricting to closed ones (as is more usual), it soon becomes clear that there are fewer derivable facts than one would wish. For instance, the term $x + y$ has no derivable outgoing transitions; however, if we instantiate x to, say, $a.\mathbf{0}$, the resulting term $a.\mathbf{0} + y$ (which is still open) allows a transition to be derived, in this case $a.\mathbf{0} + y \xrightarrow{a} \mathbf{0}$.

One way to interpret the semantics of open terms in a less roundabout way than through their instantiations is to *predict* or *assume* some facts about the variables. For that purpose, we adopt a technique from type systems (cf. [37, 12]), which ultimately goes back to sequents in formal logic (cf. [19]): namely, we collect the assumptions about the free variables in a term as *hypotheses*, and state the existence of transitions under such hypotheses. This gives rise to *conditional* transitions, which are predicates of the form

$$x_1 \xrightarrow{\alpha_1} x'_1, \ldots, x_n \xrightarrow{\alpha_n} x'_n \vdash t \xrightarrow{\beta} t',$$

where the x_i, x'_i are term variables, the $x_i \xrightarrow{\alpha_i} x'_i$ are hypotheses and t, t' are terms presumably containing those variables. We call the (finite) set $\{x_i \xrightarrow{\alpha_i} x'_i \mid 1 \leqslant i \leqslant n\}$ an (operational) *environment*,[2] and use Δ to range over operational environments. Thus, for instance, the behaviour of the open term $x + y$ can be captured by (among others) the conditional transition $x \xrightarrow{a} x' \vdash x + y \xrightarrow{a} x'$.

The intuition behind a conditional transition of the general form above is the following: *if* the variables x_i, x'_i are instantiated by terms, say t_i, t'_i, in such a way as to satisfy each hypothesised transition $x_i \xrightarrow{\alpha_i} x'_i$ by an actual transition $t_i \xrightarrow{\alpha_i} t'_i$, *then* the instantiated transition $t[t_i/x_i\, t'_i/x'_i]_{1 \leqslant i \leqslant n} \xrightarrow{\beta} t'[t_i/x_i, t'_i/x'_i]_{1 \leqslant i \leqslant n}$ holds (where $t[t_i/x_i]_{i \in I}$ denotes the simultaneous substitution, in t, of all variables x_i by their corresponding images t_i). Consider, for instance, the conditional transition $x \xrightarrow{a} x' \vdash x + y \xrightarrow{a} x'$ above: $a.\mathbf{0} \xrightarrow{a} \mathbf{0}$ is an actual transition satisfying the hypothesis $x \xrightarrow{a} x'$, and indeed, the instantiation

$$\begin{array}{ccc} (x+y)[a.\mathbf{0}/x, \mathbf{0}/x'] & \xrightarrow{a} & x'[a.\mathbf{0}/x, \mathbf{0}/x'] \\ = & & = \\ a.\mathbf{0} + y & & \mathbf{0} \end{array}$$

[2] Often also called a *context*; however, we prefer to reserve the word "context" to reason about congruences.

gives rise to a standard (i.e., unconditional) transition. Furthermore, an analogous property holds if the $t_i \xrightarrow{\alpha_i} t_i'$ themselves are conditional.

2.2. Substitution Systems

We formalise the idea of conditional transitions in as general a setting as feasible without unduly complicating matters. To provide more generality, we prefer to abstract from the syntactic level. First, we give an abstract account of the notion of *substitution*.

2.2. DEFINITION. A *substitution system* is a tuple $(O, X, var, \_[\_])$, where

- O is a set of *objects*, ranged over by o, p, q;
- $X \subseteq O$ is a set of *variables*, ranged over by x, y;
- var: $O \to \mathrm{Fin}(X)$ is a function yielding the *free variables* of an object;
- $\_[\_]: O \times (X \to_{\mathrm{Fin}} O) \to O$ denotes a *substitution* operator: $o[f]$, where $f: X \to_{\mathrm{Fin}} O$ is a *substitution function*, indicates the replacement of every occurrence of each free variable x in the object o by its f-image, $f(x)$.

The following properties are required to hold:

1. $\mathrm{var}(o[f]) = \bigcup_{x \in \mathrm{var}(o)} \mathrm{var}(f(x))$ and $\mathrm{var}(x) = \{x\}$;
2. $o[f] = o[g]$ if and only if $f(x) = g(x)$ for all $x \in \mathrm{var}(o)$;
3. $o[f][g] = o[f[g]]$, where $f[g](x) = f(x)[g]$ for all $x \in X$;
4. $o[x/x]_{x \in \mathrm{var}(o)} = o = x[o/x]$.

This is by no means a complete characterisation of the concept of substitution (for instance, the concept of unification is ignored altogether); however, it suffices for our purpose. Note that, since objects have only a finite number of free variables, and substitution functions are required to be the identity almost everywhere, if X itself is infinite then there are always fresh variables available. In fact, this is the main reason for requiring that substitution functions are the identity almost everywhere. An important special class of substitution functions $\phi : X \to_{\mathrm{Fin}} X$ just map variables onto variables. We sometimes call such functions *alpha-conversions*.

The usual way to generate a substitution system is through syntactic substitution in open terms over a given signature. We formulate a framework which allows for constructs that *bind* variables (in a limited way, sufficient for our purposes; see [17] for a more general scheme). The signature is partitioned into (first-order) operators of arbitrary arity, but without variable binding, and (higher-order) unary binders, which bind a single variable.

2.3. DEFINITION. • A *signature* is a set $\Sigma = \Sigma_O \uplus \Sigma_B$, partitioned into *operators* $op \in \Sigma_O$ with arity $|op| \in \mathbb{N}$, and *binders* $bnd \in \Sigma_B$:

- Given a set of variables X, the *term algebra* $\mathbb{T}_\Sigma(X)$ is given by
  - — variables x, where $x \in X$;
  - — terms $op(t_1, ..., t_{|op|})$, where $op \in \Sigma_O$ and $t_i \in \mathbb{T}_\Sigma(X)$ for all $1 \leqslant i \leqslant |op|$;
  - — bound terms $bnd(x.t)$, where $bnd \in \Sigma_B$, $x \in X$ and $t \in \mathbb{T}_\Sigma(X)$.

• A binary relation $\mathscr{R} \subseteq \mathbb{T}_\Sigma(X) \times \mathbb{T}_\Sigma(X)$ is called an *op*-congruence for $op \in \Sigma_O$ if $(t_i, u_i) \in \mathscr{R}$ for all $1 \leqslant i \leqslant |op|$ implies $(op(t_1, ..., t_{|op|}), op(u_1, ..., u_{|op|})) \in \mathscr{R}$, and a *bnd*-congruence for $bnd \in \Sigma_B$ if $(t, u) \in \mathscr{R}$ implies $(bnd(x.t), bnd(x.u)) \in \mathscr{R}$ for all $x \in X$.

• The function var: $\mathbb{T}_\Sigma(X) \to \mathrm{Fin}(X)$, yielding the set of free variables of a term, is defined by

— $\mathrm{var}(x) = \{x\}$;

— $\mathrm{var}(op(t_1, ..., t_{|op|})) = \bigcup_{1 \leqslant i \leqslant |op|} \mathrm{var}(t_i)$;

— $\mathrm{var}(bnd(x.t)) = \mathrm{var}(t) \backslash \{x\}$.

A term t is called *closed* if $\mathrm{var}(t) = \varnothing$, and *open* otherwise.

• Syntactic substitution is denoted $t[f]$, where $f: X \to_{\mathrm{Fin}} \mathbb{T}_\Sigma(X)$ is the identity almost everywhere, and is defined by

— $x[f] = f(x)$;

— $op(t_1, ..., t_{|op|})[f] = op(t_1[f], ..., t_{|op|}[f])$;

— $bnd(x.t)[f] = bnd(y.t[f + \{y/x\}])$, where $y \notin \mathrm{var}(f(z))$ for all $z \in \mathrm{var}(bnd(x.t))$.

• Bound variable names may be freely renamed; that is, terms are interpreted modulo the smallest congruence $\equiv$ over $\mathbb{T}_\Sigma(X)$ such that $bnd(x.t) \equiv bnd(y.t[y/x])$ for all $bnd \in \Sigma_B$, $t \in T_\Sigma(X)$ and $y \notin \mathrm{var}(t) \backslash \{x\}$.

Note that syntactic substitution is well-defined modulo $\equiv$; in particular, the choice of the variable y is not relevant, except that it must be fresh, i.e., satisfy the side condition in the definition. We give two examples.

2.4. Example. We recall the case of CCS (cf. Milner [33]). Let $\mathscr{A}$ be a global set of (abstract) action names, ranged over by $a, b, c, ...$, and $\bar{\mathscr{A}}$ a (disjoint) set of corresponding co-names, so that $\bar{\mathscr{A}} = \{\bar{a} \mid a \in \mathscr{A}\}$. Let $\mathscr{L} = \mathscr{A} \cup \bar{\mathscr{A}}$ and $\mathscr{L}_\tau = \mathscr{L} \cup \{\tau\}$, ranged over by α, β, γ; let $\bar{\tau} = \tau$ and $\bar{\bar{a}} = a$. CCS is given by the signature $\Sigma = \Sigma_O \cup \Sigma_B$, where

$$\Sigma_O = \{\mathbf{0}, \alpha._-, \_ + \_, \_ | \_, \_ \backslash A, \_[\phi]\},$$
$$\Sigma_B = \{\mathrm{rec}\ \_.\_\},$$

where $\alpha \in \mathscr{L}_\tau$, $A \subseteq \mathscr{A}$, and $\phi: \mathscr{A} \to \mathscr{A}$ (extended to $\mathscr{L}_\tau \to \mathscr{L}_\tau$ by defining $\phi(\bar{a}) = \overline{\phi(a)}$ and $\phi(\tau) = \tau$) are arbitrary, i.e., actually give rise to *sets* of operators.

2.5. Example. Another example is the pure λ-calculus (cf. Barendregt [6]). This has a particularly simple signature,

$$\Sigma_O = \{\_ \cdot \_\},$$
$$\Sigma_B = \{\lambda \_.\_\}.$$

($t \cdot u$ (function application) is more usually denoted by direct juxtaposition: tu.)

At the moment, what interests us about term algebras is the fact that they give rise to substitution systems. We leave the proof of the following up to the reader.

2.6. PROPOSITION. *For an arbitrary signature Σ and set of variables X, $(\mathbb{T}_\Sigma(X)/\equiv, X, \mathrm{var}, \_[\_])$ is a substitution system.*

Given a substitution system and a binary relation $\mathscr{R} \subseteq O^2$ over objects (extended pointwise to $(X \to_{\mathrm{Fin}} O)^2$ by defining $(f, g) \in \mathscr{R}$ if $(f(x), g(x)) \in \mathscr{R}$ for all $x \in X$), there are natural preservation properties of $\mathscr{R}$ with respect to the substitution operator.

2.7. DEFINITION. Let $\mathscr{R} \subseteq O \times O$ be a binary relation over the objects of a substitution system:

- $\mathscr{R}$ is *preserved by instantiation* if $(p, q) \in \mathscr{R}$ implies $\forall f: X \to_{\mathrm{Fin}} O.(p[f], q[f]) \in \mathscr{R}$;
- $\mathscr{R}$ is *preserved by insertion* if $\forall x \in \mathrm{var}(o) : (f(x), g(x)) \in \mathscr{R}$ implies $(o[f], o[g]) \in \mathscr{R}$.
- $\mathscr{R}$ is *substitutive* if it is preserved by instantiation and insertion.

Note that the strength of these semantic properties depends on the degree to which objects can be constructed by instantiation. In particular, in a term algebra over a signature Σ, preservation by insertion is stronger than congruence of the (first-order) operators in Σ_O, but does not imply congruence of the (higher-order) binders in Σ_B, since terms with bound variables cannot always be constructed by instantiation.

2.8. EXAMPLE. Assume that $\mathscr{R}$ is preserved by insertion in the substitution system generated by the CCS-signature (see Example 2.4). It follows that $\mathscr{R}$ is a congruence for all operators of CCS; for instance, if $(t, u) \in \mathscr{R}$ then $(a.t, a.u) = (a.x[t/x], a.x[u/x]) \in \mathscr{R}$. In fact, it also follows that $(\mathrm{rec}\, x.a.x + t, \mathrm{rec}\, x.a.x + u) = ((\mathrm{rec}\, x.a.x + y)[t/y], (\mathrm{rec}\, x.a.x + y)[u/y]) \in \mathscr{R}$ if $x \notin \mathrm{var}(t)$, showing that preservation by instantiation in some cases is applicable to binders as well. However, $(t, u) \in \mathscr{R}$ does not imply $(\mathrm{rec}\, x.t, \mathrm{rec}\, x.u) \in \mathscr{R}$, since due to the rules for substitution w.r.t. bound variables, there is no term v such that $\mathrm{rec}\, x.t = v[t/y]$ for all t. In particular, $v = \mathrm{rec}\, x.y$ does not do the trick.

2.3. Conditional Transition Systems

We now come to the behaviour models that constitute the basis for the results of this paper: conditional transition systems. These form a generalisation of labelled transition systems, in which the states are not completely unstructured objects but rather the objects of a substitution system, and the transitions are not simply labelled binary relations between states but are indexed by an operational environment, already introduced informally above. Let X be a set of variables, and L a set of *labels*, ranged over by α, β.

2.9. DEFINITION. An *operational environment* over X and L is a finite set $\Delta = \{x_i \xrightarrow{\alpha_i} x_i' \mid i \in I\}$ with $x_i, x_i' \in X$ and $\alpha_i \in L$ for all $i \in I$. Δ is called *hierarchical* if all x_i' are distinct and Δ contains no cycles of hypotheses.

The class of operational environments over X is denoted $\mathbf{\Delta}(X)$. The following defines the *source*, *target*, and *root* variables, as well as the set of *all* variables of an operational environment Δ:

$$\begin{aligned} \operatorname{src} \Delta &= \{x \mid (x \xrightarrow{\alpha} x') \in \Delta\} \\ \operatorname{tgt} \Delta &= \{x' \mid (x \xrightarrow{\alpha} x') \in \Delta\} \\ \operatorname{root} \Delta &= \operatorname{src} \Delta \backslash \operatorname{tgt} \Delta \\ \operatorname{var} \Delta &= \operatorname{src} \Delta \cup \operatorname{tgt} \Delta. \end{aligned}$$

If Δ is hierarchical, we can define the *distance* of a variable $x \in \operatorname{var} \Delta$ as the number of hypotheses necessary to reach x from some root variable of Δ:

$$\operatorname{dist}(x) = \begin{cases} 0, & \text{if} \quad x \in \operatorname{root} \Delta, \\ \operatorname{dist}(y) + 1, & \text{if} \quad (y \xrightarrow{\alpha} x) \in \Delta. \end{cases}$$

(Note that this is well defined because Δ is hierarchical, and hence, for all $x \in \operatorname{tgt} \Delta$ there is precisely one $(y \xrightarrow{\alpha} x) \in \Delta$.) We will use the distance as the basis for induction proofs. Finally, the alpha-conversion of an environment Δ by a function $\phi: X \to_{\text{Fin}} X$ is defined by

$$\phi(\Delta) = \{\phi(x) \xrightarrow{\alpha} \phi(x') \mid (x \xrightarrow{\alpha} x') \in \Delta\}.$$

2.10. DEFINITION. A *conditional transition system* (CTS) is a tuple $T = (L, S, X, \operatorname{var}, \_[\_], \vdash)$, where

- L is a set of *labels*, ranged over by α, β, γ;
- S is a set of *states*, ranged over by p, q, s;
- $(S, X, \operatorname{var}, \_[\,]/\_]$ is a substitution system;
- $\vdash \subseteq \mathbf{\Delta}(X) \times (S \times L \times S)$ is a set of *conditional transitions*; $(\Delta, (s, \alpha, s')) \in \vdash$ is denoted $\Delta \vdash s \xrightarrow{\alpha} s'$. A conditional transition is called *pure* if Δ is hierarchical, $\operatorname{var} \Delta \cap \operatorname{var}(s) = \operatorname{root} \Delta$ and $\operatorname{var}(s') \subseteq \operatorname{var}(s) \cup \operatorname{tgt} \Delta$.

The following properties are required to hold:

- Identity axiom: if $(x \xrightarrow{\alpha} x') \in \Delta$, then $\Delta \vdash x \xrightarrow{\alpha} x'$.
- Cut rule: If $\Delta \vdash s \xrightarrow{\alpha} s'$ and $\forall (x \xrightarrow{\beta} x') \in \Delta : \Delta' \vdash f(x) \xrightarrow{\beta} f(x')$, then $\Delta' \vdash s[f] \xrightarrow{\alpha} s'[f]$.

We usually leave the components L, var, and $\_[\_]$ implicit; we write S_T, X_T, and $\vdash_T$ for the remaining components of T, and drop the index T when it is clear from the context. As a further notational convention, we write $s \xrightarrow{\alpha} s'$ rather than $\Delta \vdash s \xrightarrow{\alpha} s'$ if $\Delta = \varnothing$ and omit set brackets for concrete Δ. The class of conditional

transition systems over a set of variables X is denoted $\mathbf{T}(X)$. It can be seen that the subclass of standard transition systems (over closed objects) equals $\mathbf{T}(\varnothing)$.

The *pure* conditional transitions are such that the variables in the source and target states and the operational environment satisfy some further constraints. For instance, $x \xrightarrow{\alpha} x' \vdash s \xrightarrow{\beta} s'$ is only pure if $x \in \mathrm{var}(s)$, $x' \notin \mathrm{var}(s)$, and $\mathrm{var}(s') \subseteq \mathrm{var}(s) \cup \{x'\}$. (The term "pure" for such transitions was taken from [25], where it entails similar conditions for SOS rules; see also below.)

Note that it is *not* required that a CTS contain only pure conditional transitions, or even only conditional transitions with hierarchical environments. The reason for not imposing this requirement is that the cut rule naturally gives rise to impure transitions (see the third item of Example 2.14 below) and, moreover, we need impure transitions for a smooth definition of hypothesis-preserving bisimulation (see the remark after Example 3.5).

2.11. *Notation.* We say that a substitution function (conditionally) *satisfies* an operational environment if the condition in the cut rule holds:

$$\Delta' \vdash f \,\mathbf{sat}\, \Delta \Leftrightarrow \forall (x \xrightarrow{\alpha} x') \in \Delta : \Delta' \vdash f(x) \xrightarrow{\alpha} f(x').$$

The cut rule itself can be pictured graphically (assuming $\Delta' \vdash f \,\mathbf{sat}\, \Delta$) as

$$\boxed{\Delta \vdash s \xrightarrow{\alpha} s'} \xrightarrow{f} \boxed{\Delta' \vdash s[f] \xrightarrow{\alpha} s'[f]}$$

The identity axiom and cut rule in Definition 2.10 are basic rules in the sequent calculus (cf. [19]), a fact that supports the intuition behind conditional transitions discussed above. Some consequences of the identity axiom and cut rule are listed in the following lemma. The proof is straightforward and omitted.

2.12. Lemma. *Let T be a conditional transition system with $\Delta \vdash s \xrightarrow{\alpha} s'$ arbitrary*:

1. *If $\Delta \subseteq \Delta'$, then $\Delta' \vdash s \xrightarrow{\alpha} s'$.*
2. *If $f(x) = x$ for all $x \in \mathrm{var}\ \Delta$, then $\Delta \vdash s[f] \xrightarrow{\alpha} s'[f]$.*
3. *For all $\phi: X \to_{\mathrm{Fin}} X$, $\phi(\Delta) \vdash s[\phi] \xrightarrow{\alpha} s'[\phi]$.*

One method to generate a conditional transition system is by predefining a number of conditional transitions over a given substitution system and closing under the identity axiom and cut rule, i.e., taking the smallest set of conditional transitions that can be derived from the predefined ones by applying the identity axiom and cut rule a finite number of times. In fact, this method almost precisely corresponds to the well-known concept of a structural operational *transition system specification* (TSS) consisting of structural operational rules, in the sense of [8, 25, 16, 17]. Let us make the connection explicit.

2.13. Definition. Let Σ be a signature with $\Sigma_B = \varnothing$:

- A *tree transition* over Σ is a pure conditional transition $\Delta \vdash t \xrightarrow{\alpha} t'$ over Σ such that $t = op(x_1, ..., x_{|op|})$ for some $op \in \Sigma_O$ and distinct $x_i \in V$.

TABLE 1

Pure Conditional Transitions for CCS Terms

$$\vdash \alpha.x \xrightarrow{\alpha} x \qquad R_1$$

$$x \xrightarrow{\alpha} x' \vdash x + y \xrightarrow{\alpha} x' \qquad R_2$$

$$x \xrightarrow{\alpha} x' \vdash y + x \xrightarrow{\alpha} x' \qquad R_3$$

$$x \xrightarrow{\alpha} x' \vdash x \mid y \xrightarrow{\alpha} x' \mid y \qquad R_4$$

$$x \xrightarrow{\alpha} x' \vdash y \mid x \xrightarrow{\alpha} y \mid x' \qquad R_5$$

$$x \xrightarrow{a} x',\ y \xrightarrow{\bar{a}} y' \vdash x \mid y \xrightarrow{\tau} x' \mid y' \qquad R_6$$

$$x \xrightarrow{\alpha} x' \vdash x \backslash A \xrightarrow{\alpha} x' \backslash A \quad \text{for all} \quad \alpha \notin A \cup \bar{A} \qquad R_7$$

$$x \xrightarrow{\alpha} x' \vdash x[\phi] \xrightarrow{\phi(\alpha)} x'[\phi] \qquad R_8$$

- The CTS $T_{\mathscr{S}}$ *generated* by a collection $\mathscr{S}$ of conditional transitions over Σ (pure, tree, or otherwise) is defined as consisting of states $\mathbb{T}_{\Sigma}(V)$, variables V, and as conditional transitions the smallest set $\vdash_{\mathscr{S}}$ including $\mathscr{S}$ and all instances of the identity axiom, and closed under the cut rule.

Note that any generated CTS $T_{\mathscr{S}}$ is indeed (trivially) a CTS. Moreover, any collection of tree transitions equals a TSS in the tree format of [16] (which in turn equals the *pure xyft* format of [25]), with the limitation that we have only finite operational environments. For instance, Table 1 shows the tree transitions that give rise to the operational semantics of the finite (i.e., nonrecursive) part of CCS.

2.14. EXAMPLE. Consider the conditional transitions of Table 1:

- If $\Delta' = \{x \xrightarrow{a} x'\}$ and $f = \{y/x\}$, then $\Delta' \vdash f\,\mathbf{sat}\,\varnothing$. Hence, applying the cut rule to R_1 and f, we obtain

$$\begin{array}{ccccc} \Delta' & \vdash & (\bar{a}.x)[f] & \xrightarrow{\bar{a}} & x[f] \\ = & & = & & = \\ x \xrightarrow{a} x' & \vdash & \bar{a}.y & \xrightarrow{\bar{a}} & y \end{array} .$$

The identity axiom implies $x \xrightarrow{a} x' \vdash x \xrightarrow{a} x'$; hence setting $g = \{\bar{a}.y/y, y/y'\}$, we have $\Delta' \vdash g\,\mathbf{sat}\{x \xrightarrow{a} x', y \xrightarrow{\bar{a}} y'\}$. Applying the cut rule to R_6 and g, we obtain

$$\begin{array}{ccccc} \Delta' & \vdash & (x \mid y)[g] & \xrightarrow{\tau} & (x' \mid y')[g] \\ = & & = & & = \\ x \xrightarrow{a} x' & \vdash & x \mid \bar{a}.y & \xrightarrow{\tau} & x' \mid y \end{array} .$$

- In $T_{\mathscr{S}}$, there can be many "proofs" of a given conditional transition, depending on when and where the cut rule is applied. For instance, to prove $\vdash a.\mathbf{0} + b.\mathbf{0} \xrightarrow{a} \mathbf{0}$, one may either first derive $\vdash a.\mathbf{0} \xrightarrow{a} \mathbf{0}$ and "instantiate" the predefined $x \xrightarrow{a} x' \vdash x + y \xrightarrow{a} x'$ using $f = \{a.\mathbf{0}/x,\ b.\mathbf{0}/y,\ \mathbf{0}/x'\}$, or one may first derive $\vdash a.z + y \xrightarrow{a} z$ and instantiate this further using $f = \{b.\mathbf{0}/y, \mathbf{0}/z\}$.

- All the predefined conditional transitions of Table 1 are pure (in fact, tree). However, the cut rule easily gives rise to impure transitions: for instance,

$x \xrightarrow{\alpha} y \vdash x + b.y \xrightarrow{\alpha} y$, which is not pure since the target variables of the environment and the free variables of the source term overlap.

The issue raised in the second item of the above example deserves some more attention. The following "proof normalisation" lemma restricts the use of the cut rule to a special case, namely where the conditional transition $\Delta \vdash t \xrightarrow{\alpha} t'$ to which it is applied is from the predefined set $\mathscr{S}$. This is in fact precisely the proof strategy one would use when regarding the conditional transitions in $\mathscr{S}$ as SOS rules in the usual sense.

2.15. LEMMA. *Assume Σ is a signature with $\Sigma_B = \emptyset$, and $\mathscr{S}$ is a collection of conditional transitions over Σ. $\vdash_{\mathscr{S}}$ is the smallest set of conditional transitions satisfying*:

- *if $(x \xrightarrow{\alpha} x') \in \Delta$, then $\Delta \vdash_{\mathscr{S}} x \xrightarrow{\alpha} x'$;*
- *if $(\Delta \vdash t \xrightarrow{\alpha} t') \in \mathscr{S}$ and $\Delta' \vdash_{\mathscr{S}} f$ **sat** Δ, then $\Delta' \vdash_{\mathscr{S}} t[f] \xrightarrow{\alpha} t'[f]$.*

Proof. Let $\vdash_s$ be the smallest set generated by the conditions of the lemma. We first show that $\vdash_s$ satisfies the full cut rule. Assume $\Delta \vdash_s t \xrightarrow{\alpha} t'$ and $\Delta' \vdash_s f$ **sat** Δ; we prove by induction on the construction of $\Delta \vdash_s t \xrightarrow{\alpha} t'$ that $\Delta' \vdash_s t[f] \xrightarrow{\alpha} t'[f]$:

- $(t \xrightarrow{\alpha} t') \in \Delta$. Then $\Delta' \vdash_s t[f] \xrightarrow{\alpha} t'[f]$ is part of $\Delta' \vdash_s f$ **sat** Δ.
- There are $(\Delta'' \vdash u \xrightarrow{\alpha} u') \in \mathscr{S}$ and g such that $\Delta \vdash_s g$ **sat** Δ'', $t = u[g]$ and $t' = u'[g]$. By the induction hypothesis applied to $\Delta \vdash_s g(x) \xrightarrow{\beta} g(x')$ for all $(x \xrightarrow{\beta} x') \in \Delta''$, it follows that $\Delta' \vdash_s g(x)[f] \xrightarrow{\beta} g(x')[f]$; hence $\Delta' \vdash_s g[f]$ **sat** Δ''. It follows that

$$\begin{array}{ccc} \Delta' \vdash u[g[f]] & \xrightarrow{\alpha} & u'[g[f]] \\ = & & = \\ u[g][f] & & u'[g][f] \\ = & & = \\ t[f] & & t'[f] \end{array}$$

In order to prove the lemma, it is sufficient to show that $\vdash_{\mathscr{S}}$ is included in $\vdash_s$ (the reverse inclusion follows by the fact that $\vdash_{\mathscr{S}}$ obviously satisfies the two conditions in the lemma). This is done by induction on the construction of $\Delta \vdash_{\mathscr{S}} t \xrightarrow{\alpha} t'$. There are three cases:

- Assume $(\Delta \vdash t \xrightarrow{\alpha} t') \in \mathscr{S}$. Then $\Delta \vdash_s t \xrightarrow{\alpha} t'$ by the second condition of the lemma, with $\Delta' = \Delta$ and $f = id$.
- Assume $\Delta \vdash_{\mathscr{S}} t \xrightarrow{\alpha} t'$ is an instance of the identity axiom, i.e., $(t \xrightarrow{\alpha} t') \in \Delta$. Then $\Delta \vdash_{\mathscr{S}} t \xrightarrow{\alpha} t'$ by the first condition of the lemma.
- Assume $\Delta \vdash_{\mathscr{S}} t \xrightarrow{\alpha} t'$ was constructed through the cut rule:

$$\boxed{\Delta \vdash_{\mathscr{S}} t \xrightarrow{\alpha} t'} \xleftarrow{f} \boxed{\Delta' \vdash_{\mathscr{S}} u \xrightarrow{\alpha} u'.}$$

By the induction hypothesis, $\Delta' \vdash_s u \xrightarrow{\alpha} u'$ and $\Delta \vdash_s f$ **sat** Δ', and since (as shown above) $\vdash_s$ satisfies the cut rule, it follows that $\Delta \vdash_s t \xrightarrow{\alpha} t'$. ▮

The important difference between defining a TSS and generating a conditional transition system from tree transitions is that we are working with ordinary (actual) variables and not (formal) meta-variables as in the usual TSS approaches. This difference plays a role as soon as we look at *binders*, since this is where the substitution properties of actual and formal variables differ. Definition 2.13 explicitly restricts the source terms of tree transitions to (first-order) operators and does not account for binders; and indeed, a conditional transition of the form $\Delta \vdash bnd(x.y) \xrightarrow{\alpha} t'$ for some binder *bnd* would in general make little sense. For instance, the conditional transitions of recursive terms, i.e., with an occurrence of the recursion constructor $\mathrm{rec}\_.\_$ on the outside, cannot be generated by a tree-like transition, since in general $\mathrm{rec}\, x.t$ cannot be obtained by instantiating $\mathrm{rec}\, x.y$ (see also Example 2.8 above). Fokkink and Verhoef [17] discuss this point thoroughly and formulate a framework general enough to deal with both actual and formal variables. Since the subject of this paper is bisimulation rather than TSS formats, we do not go deeper into this issue here; see, however, Section 4 below, where we discuss the operational semantics of a specific binder, namely recursion.

2.4. Completeness

For conditional transition systems to really come into their own, they should satisfy a couple of additional properties, which are, in a sense, dual to the identity axiom and the cut rule.

- The identity axiom expresses that all hypothesised transitions are derivable. The dual property is that variables have no outgoing transitions except those that are hypothesised. If this holds, we call a CTS *proper*.
- The cut rule expresses that every instantiation of a conditional transition is again a conditional transition. The dual property is that every outgoing conditional transition of an instantiated object can be derived with the cut rule using an "explanatory" conditional transition from the original object. This can be strengthened slightly by requiring the explanatory transition to be pure. If this holds, we call a CTS *complete*.

2.16. Definition. Let T be a conditional transition system:

- T is called *proper* if $\Delta \vdash x \xrightarrow{\alpha} s'$ implies $(x \xrightarrow{\alpha} s') \in \Delta$;
- T is called *complete* if for all $\Delta \vdash s[f] \xrightarrow{\alpha} s'$, there is a pure transition $\Delta' \vdash s \xrightarrow{\alpha} s''$ and a substitution f' such that $\Delta \vdash f'$ **sat** Δ', $s[f] = s[f']$, and $s' = s''[f']$.

Note that in the completeness property, we can assume w.l.o.g. that $f(x) = x$ for all $x \in \mathrm{tgt}\, \Delta'$ (since f is the identity almost everywhere, and the variables in $\mathrm{tgt}\, \Delta'$ can be converted according to Lemma 2.12.3 to variables on which f is the identity).

2.17. Example. As seen in Example 2.14, Table 1 allows us to derive the impure $x \xrightarrow{a} y \vdash x + b.y \xrightarrow{a} y$. Here, the occurrence of y both as target of the hypothesis

$x \xrightarrow{a} y$ and as free variable of the source term $x + b.y$ is coincidental; for instance, there is an "explanatory" pure transition $x \xrightarrow{a} y \vdash x + z \xrightarrow{a} y$, from which the impure one can be inferred by instantiating z with $b.y$.

If a CTS is proper and complete, it can be proved that every conditional transition can be seen as a (not necessarily injective) alpha-conversion of a pure conditional transition, and moreover, hypotheses not connected to free variables of the source term can always be dropped. These properties are expressed in the following lemma.

2.18. LEMMA. *Let T be a proper and complete conditional transition system:*

1. *If $\Delta \vdash s \xrightarrow{\alpha} s'$, there is a pure $\Delta' \vdash s \xrightarrow{\alpha} s''$ and an alpha-conversion $\phi: X \to_{\mathrm{Fin}} X$ such that $\phi(\Delta') \subseteq \Delta$, $s[\phi] = s$, and $s''[\phi] = s'$.*

2. *If $\Delta \vdash s \xrightarrow{\alpha} s'$ then $\mathrm{var}(s') \subseteq \mathrm{var}(s) \cup \mathrm{var}\,\Delta$.*

3. *If $\Delta, \Delta' \vdash s \xrightarrow{\alpha} s'$ such that $\mathrm{var}\,\Delta' \cap (\mathrm{var}(s) \cup \mathrm{var}\,\Delta) = \varnothing$, then $\Delta \vdash s \xrightarrow{\alpha} s'$.*

Proof. 1. Consider $f = id$; then $s[f] = s$. Due to completeness, there is a pure $\Delta' \vdash s \xrightarrow{\alpha} s''$ and a substitution f' such that $\Delta \vdash f'$ **sat** Δ', $s[f] = s[f']$ (which implies that f' is the identity on $\mathrm{var}(s)$) and $s' = s''[f']$; w.l.o.g. assume that f' is the identity everywhere outside $\mathrm{var}\,\Delta'$.

We construct a function $\phi: X \to_{\mathrm{Fin}} X$ such that $f' = \phi$; this ϕ meets the proof obligation. For $x \notin \mathrm{var}\,\Delta'$ let $\phi(x) = x$; for $x \in \mathrm{var}\,\Delta'$, the proof proceeds by induction on $\mathrm{dist}(x)$:

Base case. If $\mathrm{dist}(x) = 0$ then $x \in \mathrm{root}\,\Delta' \subseteq \mathrm{var}(s)$; hence $f'(x) = f(x) = x$. Let $\phi(x) = x$.

Induction step. Assume the property is fulfilled for all x with $\mathrm{dist}(x) = n$, and assume $\mathrm{dist}(x') = n + 1$; hence $(x \xrightarrow{\beta} x') \in \Delta'$ such that $\mathrm{dist}(x) = n$. It follows that $f'(x) = \phi(x)$; hence, $\Delta \vdash \phi(x) \xrightarrow{\beta} f'(x')$. Since T is proper, it follows that $f'(x') = x''$ such that $(\phi(x) \xrightarrow{\beta} x'') \in \Delta$. Let $\phi(x') = x''$.

2. Clause 1 above guarantees there is a pure $\Delta'' \vdash s \xrightarrow{\alpha} s''$ and an alpha-conversion $\phi: X \to_{\mathrm{Fin}} X$ such that $\phi(\Delta'') \subseteq \Delta \cup \Delta'$, ϕ is the identity on $\mathrm{var}(s)$ and $s' = s''[\phi]$. It follows that $\mathrm{var}(s'') \subseteq \mathrm{var}(s) \cup \mathrm{var}\,\Delta''$, and hence,

$$\mathrm{var}(s') = \phi(\mathrm{var}(s'')) \subseteq \phi(\mathrm{var}(s)) \cup \phi(\mathrm{var}\,\Delta'') \subseteq \mathrm{var}(s) \cup \mathrm{var}\,\Delta.$$

3. Clause 1 above guarantees there is a pure $\Delta'' \vdash s \xrightarrow{\alpha} s''$ and an alpha-conversion $\phi: X \to_{\mathrm{Fin}} X$ such that $\phi(\Delta'') \subseteq \Delta \cup \Delta'$, $s = s[\phi]$ and $s' = s''[\phi]$. W.l.o.g. assume that ϕ is the identity outside $\mathrm{var}\,\Delta''$. It can be proved by a straightforward induction on the distance of the variables in Δ'' that ϕ only maps them to variables of Δ; i.e., $\phi(\Delta'') \subseteq \Delta$. Due to Lemma 2.12.1, this implies $\Delta \vdash s \xrightarrow{\alpha} s'$. ∎

The following result states that the conditional transition systems obtained by "closing up" a number of predefined tree transitions under the identity axiom and cut rule, in the sense of Definition 2.13, are always proper and complete.

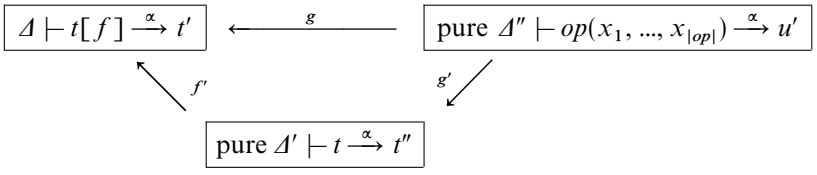
2.19. THEOREM. *If Σ is a signature with $\Sigma_B = \varnothing$, and $\mathscr{S}$ is a collection of tree transitions over Σ, then $T_{\mathscr{S}}$ is a proper and complete CTS.*

Proof. By induction on the construction of the conditional transitions of $T_{\mathscr{S}}$, using Lemma 2.15. T is proper as a direct consequence of Lemma 2.15. Completeness is proved by induction on the construction of $\Delta \vdash_{\mathscr{S}} t[f] \xrightarrow{\alpha} t'$, assuming a "normalised" construction in the sense of Lemma 2.15:

- Assume $t = x \in V$. Then the proof obligations are met by $\Delta' = \{x \xrightarrow{\alpha} x'\}$, $t'' = x'$, and $f' = \{f(x)/x, t'/x'\}$.
- Otherwise $t = op(t_1, ..., t_{|op|})$ for some $op \in \Sigma_O$. By Lemma 2.15, then, $\Delta \vdash t \xrightarrow{\alpha} t'$ was constructed using a cut rule:

$$\boxed{\Delta \vdash t[f] \xrightarrow{\alpha} t'} \xleftarrow{g} \boxed{\text{pure } \Delta'' \vdash op(x_1, ..., x_{|op|}) \xrightarrow{\alpha} u'} \in \mathscr{S}.$$

W.l.o.g. assume g is the identity outside $\{x_1, ..., x_{|op|}\} \cup \text{var } \Delta''$. It follows that $g(x_i) = t_i[f]$ for $1 \leqslant i \leqslant |op|$. We set out to extend the above diagram as



This entails constructing f', g', and Δ' such that

— $t[f'] = t[f]$ and f' is the identity outside $\text{var}(t) \cup \text{var } \Delta'$;
— $g'(x_i) = t_i$ for $1 \leqslant i \leqslant |op|$ and $g = g'[f']$;
— $\Delta \vdash f'$ **sat** Δ' and $\Delta' \vdash g'$ **sat** Δ'';
— Δ' is hierarchical and $\text{var } \Delta' \cap \text{var}(t) = \text{root } \Delta'$.

It follows that $t = u[g']$; furthermore, let $t'' = u'[g']$. Note that since g is the identity outside $\{x_1, ..., x_{|op|}\} \cup \text{var } \Delta''$, f' is the identity outside $\text{var}(t) \cup \text{var } \Delta'$ and $g = g'[f']$, it follows that $\text{var}(t'') \subseteq \text{var}(t) \cup \text{var } \Delta'$. By the cut rule we can derive $\Delta' \vdash t \xrightarrow{\alpha} t''$, which is then pure; and it follows by the properties of substitution that $t[f'] = t[f]$ and $t''[f'] = u'[g'][f'] = u'[g'[f']] = u'[g] = t'$; hence the proof obligations are met.

The construction of f', g' and Δ' is stepwise, based on a sequence $\varnothing = \Delta''_0 \subset \cdots \subset \Delta''_n = \Delta''$ where for each $0 < k \leqslant n$, $\Delta''_k = \Delta''_{k-1} \cup \{y_k \xrightarrow{\alpha_k} y'_k\}$ with $y'_k \notin \text{var } \Delta''_{k-1}$ (such a sequence exists because Δ'' is hierarchical). Note that for all $0 < k \leqslant n$, $y_k \in \text{var } \Delta''_{k-1}$ or $y_k = x_i$ for some $1 \leqslant i \leqslant |op|$. By induction on k with $0 \leqslant k \leqslant n$, we construct f'_k, g'_k and Δ'_k such that

— $t[f'_k] = t[f]$ and f'_k is the identity outside $\text{var}(t) \cup \text{var } \Delta'_k$;
— $g'_k(x_i) = t_i$ for all $1 \leqslant i \leqslant |op|$ and $g(y) = g'_k(y)[f'_k]$ for all $y \in \text{var } \Delta'_k$;
— $\Delta'_k \vdash g'_k$ **sat** Δ''_k and $\Delta \vdash f'_k$ **sat** Δ'_k;
— Δ'_k is hierarchical and $\text{var } \Delta'_k \cap \text{var}(t) = \text{root } \Delta'_k$.

Base case. For $k = 0$ (hence $\Delta''_k = \varnothing$) let

$$f'_k = \{ f(x)/y \mid x \in \mathrm{var}(t) \}$$
$$g'_k = \{ t_i/x_i \mid 1 \leqslant i \leqslant |op| \}$$
$$\Delta'_k = \varnothing.$$

Induction step. Assume the construction is done up to $k-1 < n$. By the inner induction hypothesis we have $g(y_k) = g'_{k-1}(y_k)[f'_{k-1}]$. By the outer induction, completeness holds for the transition $\Delta \vdash g(y_k) \xrightarrow{\alpha_k} g(y'_k)$; hence there are a pure $\hat{\Delta} \vdash g'_{k-1}(y_k) \xrightarrow{\beta} v'$ and $\hat{f}$ such that $\Delta \vdash \hat{f}$ **sat** $\hat{\Delta}$, $g'_{k-1}(x)[\hat{f}] = g(y_k)$, and $v'[\hat{f}] = g(y'_k)$. W.l.o.g. assume $\mathrm{tgt}\,\hat{\Delta} \cap (\mathrm{var}(t) \cup \mathrm{tgt}\,\Delta'_{k-1}) = \varnothing$, and let

$$f'_k = f'_{k-1} \pm \{ \hat{f}(z)/z \mid z \in \mathrm{tgt}\,\hat{\Delta} \}$$
$$g'_k = g'_{k-1} \pm \{ v'/y'_k \}$$
$$\Delta'_k = \Delta'_{k-1} \cup \hat{\Delta}.$$

The above requirements are then satisfied by $f' = f'_n$, $g' = g'_n$, and $\Delta' = \Delta'_n$. ∎

In the remainder of this paper, we implicitly assume all conditional transition systems to be proper and complete.

3. BISIMULATION

Now we consider the notion of *bisimulation* for conditional transition systems. At this point, we reap the benefits of our choice to take the more abstract approach of regarding conditional transition systems as semantic models in their own right, rather than strictly in the context of a given signature: the definitions and results of this section are entirely syntax-independent.

We first define three different versions of bisimilarity over open states: the standard *closed instance* bisimilarity ($\sim^{\mathsf{ci}}$), De Simone's *formal hypothesis* bisimilarity ($\sim^{\mathsf{fh}}$), and a variation of the latter, called *hypothesis-preserving* bisimilarity ($\sim^{\mathsf{hp}}$). We then show that $\sim^{\mathsf{fh}}$ is strictly stronger than $\sim^{\mathsf{hp}}$, which in turn is strictly stronger than $\sim^{\mathsf{ci}}$; however, the differences are only in the treatment of *open* states; on closed states, they all coincide. We also show that on complete CTS's, each of the above relations is substitutive (Definition 2.7). Finally, we give an alternative characterisation of $\sim^{\mathsf{hp}}$ in terms of nonconditional transitions, as *substitution closed* bisimilarity ($\sim^{\mathsf{sb}}$)—a result that relies on the existence of states in the conditional transition system that model choice- and action prefix-like operators. The overall situation is depicted by the schema:

$$\sim^{\mathsf{fh}} \subset \sim^{\mathsf{hp}} \subset \sim^{\mathsf{ci}}$$
$$=$$
$$\sim^{\mathsf{sb}} \text{ (if the CTS has choice and prefix states).}$$

3.1. Definitions

First, we extend the standard definition of bisimulation to conditional transition systems. As recounted in the Introduction, the usual approach to defining a semantic equivalence for open terms is to define it initially for closed terms and to extend it to open terms by considering all their closed instantiations. We call the corresponding standard notion of bisimulation *closed-instance bisimulation*. In the following, $S^\bullet$ denotes the set of closed states in S, i.e., $S^\bullet = \{s \in S \mid \mathrm{var}(s) = \varnothing\}$; and $Y \to S^\bullet$, for $Y \subseteq_{\mathrm{Fin}} X$, denotes the class of substitution functions that map all $x \in Y$ to $S^\bullet$ and are the identity on $X \setminus Y$. (Note that, due to Lemmas 2.12.1 and 2.18.3, if $s \in S^\bullet$ then $\Delta \vdash s \xrightarrow{\alpha} s'$ if and only if $s \xrightarrow{\alpha} s'$, in which case, moreover, also $s' \in S^\bullet$; hence as long as we are regarding closed states, hypotheses play no role.)

3.1. Definition. Let T be a conditional transition system:

- A *closed-instance bisimulation* over T is a symmetrical relation $\mathscr{R} \subseteq S^\bullet \times S^\bullet$, such that $(p, q) \in \mathscr{R}$ implies that for all $p \xrightarrow{\alpha} p'$, there is a $q \xrightarrow{\alpha} q'$ with $(p', q') \in \mathscr{R}$.
- Two (arbitrary) states $p, q \in S$ are said to be *closed-instance bisimilar*, denoted $p \sim^{\mathsf{ci}} q$, if there is a closed-instance bisimulation $\mathscr{R}$ such that $(p[f], q[f]) \in \mathscr{R}$ for all $f: \mathrm{var}(p, q) \to S^\bullet$.

It is seen that closed-instance bisimulation does not take hypotheses into account in any way; open states are not matched directly, but are instantiated to a closed state first. To adapt this to the concept of hypotheses, we extend the matching requirement to open states and conditional transitions, by requiring equality of operational environments. That is, to match up a transition $\Delta \vdash p \xrightarrow{\alpha} p'$ with a transition $\Delta' \vdash q \xrightarrow{\beta} q'$, we require not only $\alpha = \beta$ but also $\Delta = \Delta'$. This gives rise to a relation due to De Simone [14], which he called *formal hypothesis bisimilarity*.

3.2. Definition. Let T be a conditional transition system:

- A *formal hypothesis bisimulation* over T is a symmetrical relation $\mathscr{R} \subseteq S \times S$, such that $(p, q) \in \mathscr{R}$ implies that for all $\Delta \vdash p \xrightarrow{\alpha} p'$, there is a $\Delta \vdash q \xrightarrow{\alpha} q'$ with $(p', q') \in \mathscr{R}$.
- Two (arbitrary) states $p, q \in S$ are said to be *formal hypothesis bisimilar*, denoted $p \sim^{\mathsf{fh}} q$, if there is a formal hypothesis bisimulation $\mathscr{R}$ such that $(p, q) \in \mathscr{R}$.

By completeness it follows that we may as well check the matching condition only for pure transitions. An example showing the difference between $\sim^{\mathsf{ci}}$ and $\sim^{\mathsf{fh}}$ is the following (two other examples, in a synchronous setting, were given in [14]).

3.3. Example. Consider an extension of *CCS* with a unary operator $do_a(\_)$ for some $a \in \mathscr{A}$, whose behaviour is described by the conditional transition:

$$x \xrightarrow{a} x' \vdash do_a(x) \xrightarrow{a} \mathbf{0}.$$

Hence, $do_a(t)$ can only "let through" a single a-transition of t, after which it deadlocks. Now consider the terms

$$t = b;\, do_a(x) + b.a.\mathbf{0} + b.\mathbf{0} \quad \begin{matrix} \sim^{\mathsf{ci}} \\ \not\sim^{\mathsf{fh}} \end{matrix} \quad b.a.\mathbf{0} + b.\mathbf{0} = u.$$

The interesting issue is to match the following (unconditional) transition of the left-hand term (where v is an arbitrary instantiation of x):

$$\vdash t[v/x] \xrightarrow{b} do_a(v).$$

If v is closed, $do_a(v)$ can behave in only one of two basic ways: either it has no transition (in which case the b-transition in question can be matched by $u[v/x] \xrightarrow{b} \mathbf{0}$), or it has a single a-transition to $\mathbf{0}$ (in which case the b-transition can be matched by $u[v/x] \xrightarrow{b} a.\mathbf{0}$). Hence, we have $t \sim^{\mathsf{ci}} u$. On the other hand, if $v = x$, then for the above b-transition there is no transition of u that matches up to formal hypothesis bisimulation. The only possibilities are

$$\vdash u \xrightarrow{b} a.\mathbf{0},$$

after which $\vdash a.\mathbf{0} \xrightarrow{a} \mathbf{0}$ has no (unconditional) counterpart in $do_a(x)$, or

$$\vdash u \xrightarrow{b} \mathbf{0},$$

after which $x \xrightarrow{a} x' \vdash do_a(x) \xrightarrow{a} \mathbf{0}$ has no counterpart in $\mathbf{0}$. Hence, we have $t \not\sim^{\mathsf{fh}} u$.

This example relies on a nonstandard operator, $do_a(-)$. A natural question is whether ci- and fh-bisimilarity are also different in standard CCS or other standard process algebras known from the literature, or more generally, if we can establish a restriction on the operational rules of a language so that the two relations coincide. We conjecture that, in fact, $\sim^{\mathsf{ci}}$ and $\sim^{\mathsf{fh}}$ do coincide in most, if not all, of the standard process algebras, including CCS, CSP [27], ACP [4], and LOTOS [9]. We return to this issue in the conclusion of the paper.

At first sight, $\sim^{\mathsf{fh}}$ would seem to be the natural and, indeed, only sensible way to extend bisimulation to conditional transitions. However, there are arguments in favour of yet another variant. Namely, environments can be treated as *persistent*, in that any hypothesis made concerning a variable's behaviour while matching up particular states is retained "for future use." Formally, the corresponding bisimulations are *environment-indexed families* of binary relations, where the index lists the assumptions made in the past and is augmented each time a match is made. (Other examples of indexed bisimulation relations are *history-preserving* bisimulation in [23], *symbolic* bisimulation in [26] and *location* bisimulation in [38, 39].)

3.4. Definition. Let T be a conditional transition system:

- A *hypotheses-preserving bisimulation* over T is a family of symmetrical relations $\mathscr{R}_\Gamma \subseteq S \times S$ for $\Gamma \subseteq_{Fin} \mathbf{\Delta}(X)$, such that $(p, q) \in \mathscr{R}_\Gamma$ implies that for all $\Delta \vdash p \xrightarrow{\alpha} p'$, there is a $\Gamma \cup \Delta \vdash q \xrightarrow{\alpha} q'$ with $(p', q') \in \mathscr{R}_{\Gamma \cup \Delta}$.

• Two states $p, q \in S$ are said to be *hypotheses-preserving bisimilar* under Γ, denoted $\Gamma \vdash p \sim^{\mathsf{hp}} q$, if there is an hypotheses-preserving bisimulation $\mathscr{R}_\Gamma$ such that $(p, q) \in \mathscr{R}_\Gamma$.

We write $p \sim^{\mathsf{hp}} q$ for $\varnothing \vdash p \sim^{\mathsf{hp}} q$. Again, due to completeness, we may as well only check the bisimulation condition for pure $\Delta \vdash p \xrightarrow{\alpha} p'$, where, moreover, tgt Δ is fresh w.r.t. the variables in Γ. Example 3.3 is also applicable to hypotheses-preserving bisimulation; it shows that $\sim^{\mathsf{hp}}$ and $\sim^{\mathsf{ci}}$ are different. The following example illustrates the difference between $\sim^{\mathsf{fh}}$ and $\sim^{\mathsf{hp}}$.[3]

3.5. EXAMPLE. Consider CCS extended with $do_a(\_)$, as in Example 3.3. Now consider the terms

$$t = do_a(x) \,|\, do_a(x) + a.a.\mathbf{0} \qquad \begin{matrix} \not\sim^{\mathsf{fh}} \\ \sim^{\mathsf{hp}} \end{matrix} \qquad a.a.\mathbf{0} = u.$$

The interesting issue is to match the following transition of the left-hand term,

$$x \xrightarrow{a} x' \vdash t \xrightarrow{a} \mathbf{0} \,|\, do_a(x)$$

(or its symmetric equivalent). There is no transition of u that matches this up to formal hypothesis bisimulation; the only possibility is

$$x \xrightarrow{a} x' \vdash u \xrightarrow{a} a.\mathbf{0},$$

after which $\vdash a.\mathbf{0} \xrightarrow{a} \mathbf{0}$ has no (unconditional) counterpart in $\mathbf{0} \,|\, do_a(x)$. Hence, we have $t \not\sim^{\mathsf{fh}} u$. On the other hand, consider (the symmetric closure of) the relations:

$$\mathscr{R}_\varnothing = \{(do_a(x) \,|\, do_a(x) + a.a.\mathbf{0},\ a.a.\mathbf{0}),\ (a.\mathbf{0},\ a.\mathbf{0}),\ (\mathbf{0}, \mathbf{0})\}$$

$$\mathscr{R}_{\{x \xrightarrow{a} x'\}} = \{(\mathbf{0} \,|\, do_a(x),\ a.\mathbf{0}),\ (do_a(x) \,|\, \mathbf{0},\ a.\mathbf{0}),\ (\mathbf{0} \,|\, \mathbf{0},\ \mathbf{0})\}.$$

These form a hypotheses-preserving bisimulation. In particular, for the pair $(\mathbf{0} \,|\, do_a(x),\ a.\mathbf{0}) \in \mathscr{R}_{\{x \xrightarrow{a} x'\}}$, the right-hand side transition $a.\mathbf{0} \xrightarrow{a} \mathbf{0}$ can be matched, given the preserved hypothesis $x \xrightarrow{a} x'$, by $\mathbf{0} \,|\, do_a(x) \xrightarrow{a} \mathbf{0} \,|\, \mathbf{0}$. Hence, we have $t \sim^{\mathsf{hp}} u$.

Note that, typically, the transition $\Gamma \cup \Delta \vdash q \xrightarrow{\alpha} q'$ required in the matching criterion of an hp-bisimulation is *not* pure, since Γ consists of "past" hypotheses, among which may very well be ones having variables of q as target variables. This is the reason why we have not forbidden impure transitions altogether. Some properties of history-preserving bisimilarity are collected in the following proposition.

[3] It should be remarked that in [14], where De Simone first proposed $\sim^{\mathsf{fh}}$, he only considers pure $\Delta \vdash t \xrightarrow{\alpha} t'$ with, moreover, $\mathrm{var}(t') \cap \mathrm{src}\, \Delta = \varnothing$, meaning that source variables of hypotheses may not occur in the target states of the transition any more. Under that restriction, hypotheses made in the past cannot influence any future transitions, which implies that $\sim^{\mathsf{fh}}$ and $\sim^{\mathsf{hp}}$ coincide.

3.6. PROPOSITION. *Let T be a conditional transition system and $\Delta, \Delta' \in \mathbf{\Delta}(X)$:*

1. $\Delta \vdash s \sim^{\mathsf{hp}} s$ *for all* $s \in S$;
2. *If* $\Delta \vdash p \sim^{\mathsf{hp}} s$ *and* $\Delta \vdash s \sim^{\mathsf{hp}} q$, *then* $\Delta \vdash p \sim^{\mathsf{hp}} q$.
3. *If* $\Delta \vdash p \sim^{\mathsf{hp}} q$ *then* $\Delta \cup \Delta' \vdash p \sim^{\mathsf{hp}} q$.
4. *If* $\Delta \uplus \Delta' \vdash p \sim^{\mathsf{hp}} q$ *such that* $\operatorname{var} \Delta' \cap (\operatorname{var}(p) \cap \operatorname{var}(q)) = \varnothing$, *then* $\Delta \vdash p \sim^{\mathsf{hp}} q$.

3.2. Properties

Let us compare the various bisimilarities introduced above. The first property we consider is their relative strength. It turns out that we have a strict inclusion.

3.7. THEOREM. $\sim^{\mathsf{fh}} \subset \sim^{\mathsf{hp}} \subset \sim^{\mathsf{ci}}$.

Proof. We prove the inclusions; their strictness follows from Example 3.3 and Example 3.5, respectively. The proof consists of two parts:

($\sim^{\mathsf{hp}} \subseteq \sim^{\mathsf{ci}}$) We show that hypotheses-preserving bisimilarity gives rise to a ci-bisimulation by considering arbitrary closed instantiations that satisfy the preserved environment. Define

$$\mathscr{R} = \{(p[f], q[f]) \mid \Gamma \vdash p \sim^{\mathsf{hp}} q, f\colon \operatorname{var}(p, q) \to S^\bullet, f \textbf{ sat } \Gamma\}.$$

It follows that $\mathscr{R}$ is symmetrical and $\mathscr{R} \subseteq S^\bullet \times S^\bullet$.

- Let $(p[f], q[f]) \in \mathscr{R}$ be arbitrary, and assume $p[f] \xrightarrow{\alpha} p'$.
- Due to completeness, there is a pure $\Delta \vdash p \xrightarrow{\alpha} p''$ and a substitution f' such that f' **sat** Δ, $p[f] = p[f']$, and $p' = p''[f']$. W.l.o.g. assume $f'(x) = f(x)$ for all $x \notin \operatorname{tgt} \Delta$; hence, f' **sat** Γ.
- It can be proved by induction on the distance of the variables of Δ, using Lemma 2.18.2, that $f'(x) \in S^\bullet$ for all $x \in \operatorname{var} \Delta$.
- Due to hp-bisimulation, $\Gamma \cup \Delta \vdash q \xrightarrow{\alpha} q''$ such that $\Gamma \cup \Delta \vdash p'' \sim^{\mathsf{hp}} q''$.
- It follows that f' **sat** $\Gamma \cup \Delta$; hence $q[f] \xrightarrow{\alpha} q''[f']$ by the cut rule.
- By construction, $(p''[f'], q''[f']) \in \mathscr{R}$.

It follows that $\mathscr{R}$ is a closed-instance bisimulation.

($\sim^{\mathsf{fh}} \subseteq \sim^{\mathsf{hp}}$) The role of the preserved environments is to make the matching requirement of bisimulation easier to satisfy. Therefore, any fh-bisimulation immediately gives rise to an hp-bisimulation.

For all Γ define $\mathscr{R}_\Gamma = \sim^{\mathsf{fh}}$. It follows that each $\mathscr{R}_\Gamma$ is symmetrical.

- Let $(p, q) \in \mathscr{R}_\Gamma$ be arbitrary, and assume $\Delta \vdash p \xrightarrow{\alpha} p'$.
- Due to fh-bisimulation, it follows that $\Delta \vdash q \xrightarrow{\alpha} q'$ such that $p' \sim^{\mathsf{fh}} q'$.
- Due to Lemma 2.12.1, it follows that $\Gamma \cup \Delta \vdash q \xrightarrow{\alpha} q'$.
- By construction, $(p', q') \in \mathscr{R}_{\Gamma \cup \Delta}$.

It follows that $\mathscr{R}_\Gamma$ is an hypotheses-preserving bisimulation. ∎

The next point is that, although we have three different notions of bisimilarity, their difference lies in treatment of free variables. This is in accordance with our intention to study *open* bisimulation. Formally, this means that on closed states, the relations should coincide. In the following proposition, $\sim \restriction S^\bullet$ abbreviates $\sim \cap (S^\bullet \times S^\bullet)$.

3.8. THEOREM. $(\sim^{\mathsf{ci}} \restriction S^\bullet) = (\sim^{\mathsf{hp}} \restriction S^\bullet) = (\sim^{\mathsf{fh}} \restriction S^\bullet)$.

Proof. Due to Theorem 3.7, it suffices to show $\sim^{\mathsf{ci}} \restriction S^\bullet \subseteq \sim^{\mathsf{fh}}$. For this purpose, we show that $\mathscr{R} = (\sim^{\mathsf{ci}} \restriction S^\bullet)$ is an fh-bisimulation. Let $(p, q) \in \mathscr{R}$ be arbitrary, and assume a pure $\Delta \vdash p \xrightarrow{\alpha} p'$. It follows that Δ is hierarchical and root $\Delta \subseteq \mathrm{var}(p) = \varnothing$, implying $\Delta = \varnothing$. Due to ci-bisimulation, $q \xrightarrow{\alpha} q'$ such that $p' \sim^{\mathsf{ci}} q'$, or identically, $\Delta \vdash q \xrightarrow{\alpha} q'$ such that $(p', q') \in \mathscr{R}$. ∎

We now come to the substitutivity of bisimilarity, consisting of the dual properties of preservation by instantiation and insertion (cf. Definition 2.7).

3.9. THEOREM. $\sim^{\mathsf{ci}}$, $\sim^{\mathsf{hp}}$, *and* $\sim^{\mathsf{fh}}$ *are substitutive.*

First, we consider preservation by instantiation. For $\sim^{\mathsf{ci}}$, this property is built into the definition, since $p \sim^{\mathsf{ci}} q$ iff $p[f] \sim^{\mathsf{ci}} q[f]$ for all closed instantiations f. For $\sim^{\mathsf{hp}}$ and $\sim^{\mathsf{fh}}$, the situation is less straightforward.

3.10. LEMMA. *If* $\Gamma \vdash p \sim^{\mathsf{hp}} q$ *and* $\Delta \vdash f$ **sat** Γ, *then* $\Delta \vdash p[f] \sim^{\mathsf{hp}} q[f]$.

Proof. It suffices to show that the following is an hp-bisimulation:

$$\mathscr{R}_\Gamma = \{(p[f], q[f]) \mid \Gamma' \vdash p \sim^{\mathsf{hp}} q, \Gamma \vdash f \textbf{ sat } \Gamma'\}.$$

It immediately follows that all $\mathscr{R}_\Gamma$ are symmetrical.

- Let $(p[f], q[f]) \in \mathscr{R}_\Gamma$ such that $\Delta \vdash p[f] \xrightarrow{\alpha} p'$.
- By completeness, there is a pure $\Delta' \vdash p \xrightarrow{\alpha} p''$ and a substitution f' such that $\Delta \vdash f'$ **sat** Δ', $p[f] = p[f']$, and $p' = p''[f']$. W.l.o.g. assume $f'(x) = f(x)$ for all $x \notin \mathrm{tgt}\ \Delta'$.
- Due to $\Gamma' \vdash p \sim^{\mathsf{hp}} q$, there is a $\Gamma' \cup \Delta' \vdash q \xrightarrow{\alpha} q''$ such that $\Gamma' \cup \Delta' \vdash p'' \sim^{\mathsf{hp}} q''$.
- Since $q[f'] = q[f]$, the cut rule implies $\Gamma' \cup \Delta' \vdash q[f] \xrightarrow{\alpha} q''[f']$.
- Since $\Gamma \cup \Delta \vdash f'$ **sat** $\Gamma' \cup \Delta'$, by construction we have $(p''[f'], q''[f']) \in \mathscr{R}_{\Gamma \cup \Delta}$. ∎

3.11. LEMMA. *If* $p \sim^{\mathsf{fh}} q$, *then* $p[f] \sim^{\mathsf{fh}} q[f]$ *for arbitrary* f.

Proof. It suffices to prove that the following is an fh-bisimulation:

$$\mathscr{R} = \{(p[f], q[f]) \mid p \sim^{\mathsf{fh}} q\}.$$

It is clear that $\mathscr{R}$ is symmetrical.

- Let $(p[f], q[f]) \in \mathscr{R}$ be arbitrary, and assume $\Delta \vdash p[f] \xrightarrow{\alpha} p'$.
- By completeness, there is a pure $\Delta' \vdash p \xrightarrow{\alpha} p''$ and a substitution f' such that $\Delta \vdash f'$ **sat** Δ', $p[f] = p[f']$, and $p' = p''[f']$. W.l.o.g. assume $f'(x) = f(x)$ for all $x \notin \mathrm{tgt}\,\Delta'$.
- Due to $p \sim^{\mathsf{fh}} q$, there is a $\Delta' \vdash q \xrightarrow{\alpha} q''$ such that $p'' \sim^{\mathsf{fh}} q''$.
- Since $q[f'] = q[f]$, the cut rule implies $\Delta \vdash q[f] \xrightarrow{\alpha} q''[f']$.
- By construction, $(p''[f'], q''[f']) \in \mathscr{R}$. ∎

A more involved property to prove is that of *preservation by insertion*. For the case of closed-instance bisimilarity, we restrict ourselves to closed instantiations; the general case follows easily.

3.12. Lemma. *If* $f, g: \mathrm{var}(s) \to S^\bullet$ *such that* $f \sim^{\mathsf{ci}} g$, *then* $s[f] \sim^{\mathsf{ci}} s[g]$.

Proof. It suffices to show that the following is a ci-bisimulation relation:

$$\mathscr{R} = \{(s[f], s[g]) \mid f, g: \mathrm{var}(s) \to S^\bullet, f \sim^{\mathsf{ci}} g\}.$$

It is immediately clear that $\mathscr{R}$ is symmetrical.

- Let $(s[f], s[g]) \in \mathscr{R}$ be arbitrary, and consider $s[f] \xrightarrow{\alpha} s'$.
- By completeness, there is a pure $\Delta \vdash s \xrightarrow{\alpha} s''$ and a substitution f' such that f' **sat** Δ, $s[f'] = s[f]$ and $s''[f'] = s'$. W.l.o.g. assume $f'(x) = f(x)$ for all $x \notin \mathrm{tgt}\,\Delta$.
- We construct g': $\mathrm{var}(s) \to S^\bullet$ with $g'(x) = g(x)$ for all $x \notin \mathrm{tgt}\,\Delta$, $f' \sim^{\mathsf{ci}} g'$ and g' **sat** Δ. Let $g'(x) = g(x)$ for all $x \notin \mathrm{var}\,\Delta$; the images of the $x \in \mathrm{var}\,\Delta$ are constructed by induction on $\mathrm{dist}(x)$.

Base case. If $\mathrm{dist}(x) = 0$ then $x \in \mathrm{root}\,\Delta$; let $g'(x) = g(x)$.

Induction step. Otherwise $(y \xrightarrow{\beta} x) \in \Delta$, where (by the induction hypothesis) $f'(y) \sim^{\mathsf{ci}} g'(y)$; let $g'(x)$ be such that $g'(y) \xrightarrow{\beta} g'(x)$ and $f'(x) \sim^{\mathsf{ci}} g'(x)$ (this exists by $f'(y) \xrightarrow{\beta} f'(x)$ and closed-instance bisimilarity).

- $s[g] = s[g']$, and by the cut rule, $s[g'] \xrightarrow{\alpha} s''[g']$.
- By construction, $(s''[f'], s''[g']) \in \mathscr{R}$. ∎

3.13. Lemma. *If* $\Gamma \vdash f \sim^{\mathsf{hp}} g$, *then* $\Gamma \vdash s[f] \sim^{\mathsf{hp}} s[g]$ *for arbitrary* s.

Proof. This consists of proving that the following is an hp-bisimulation relation: for all Γ,

$$\mathscr{R}_\Gamma = \{(s[f], s[g]) \mid \Gamma \vdash f \sim^{\mathsf{hp}} g\}.$$

It is clear that all $\mathscr{R}_\Gamma$ are symmetrical.

- Assume $(s[f], s[g]) \in \mathscr{R}_\Gamma$ and $\Delta \vdash s[f] \xrightarrow{\alpha} s'$.
- By completeness, there is a pure $\Delta' \vdash s \xrightarrow{\alpha} s''$ and a substitution f' such that $\Delta \vdash f'$ **sat** Δ', $s[f'] = s[f]$ and $s''[f'] = s'$. W.l.o.g. assume $f'(x) = f(x)$ for all $x \notin \mathrm{tgt}\,\Delta'$.

• We construct $g' : X \to_{\mathrm{Fin}} S$ such that $g'(x) = g(x)$ for all $x \notin \mathrm{tgt}\, \Delta'$, $\Gamma \cup \Delta \vdash f' \sim^{\mathsf{hp}} g'$ and $\Gamma \cup \Delta \vdash g'$ **sat** Δ'. Let $g'(x) = g(x)$ for all $x \notin \mathrm{var}\, \Delta'$. (Note that due to Proposition 3.6.2, $\Gamma \vdash f \sim^{\mathsf{hp}} g$ implies $\Gamma \cup \Delta \vdash f \sim^{\mathsf{hp}} g$.) The images of the $x \in \mathrm{var}\, \Delta'$ are constructed by induction on $\mathrm{dist}(x)$.

Base case. If $\mathrm{dist}(x) = 0$ then $x \in \mathrm{root}\, \Delta'$; let $g'(x) = g(x)$.

Induction step. Otherwise $(y \xrightarrow{\beta} x) \in \Delta'$, where (by the induction hypothesis) $\Gamma \cup \Delta \vdash f'(y) \sim^{\mathsf{hp}} g'(y)$. Let $g'(x)$ be such that $\Gamma \cup \Delta \vdash g'(y) \xrightarrow{\beta} g'(x)$ and $\Gamma \cup \Delta \vdash f'(x) \sim^{\mathsf{hp}} g'(x)$ (which exists due to $\Delta \vdash f'(y) \xrightarrow{\beta} f'(x)$ and history preserving bisimilarity).

• $s[g] = s[g']$, and by the cut rule, $\Gamma \cup \Delta \vdash s[g'] \xrightarrow{\alpha} s''[g']$.

• By construction, $(s''[f'], s''[g']) \in \mathscr{R}_{\Gamma \cup \Delta}$. ∎

3.14. Lemma. *If $f \sim^{\mathsf{fh}} g$, then $s[f] \sim^{\mathsf{fh}} s[g]$ for arbitrary s.*

Proof. It suffices to show that the following is an fh-bisimulation:

$$\mathscr{R} = \{(s[f], s[g]) \mid f \sim^{\mathsf{fh}} g\}.$$

It is clear that $\mathscr{R}$ is symmetrical.

• Let $(s[f], s[g]) \in \mathscr{R}$ be arbitrary, and consider $\Delta \vdash s[f] \xrightarrow{\alpha} s'$.

• By completeness, there is a pure $\Delta' \vdash s \xrightarrow{\alpha} s''$ and a substitution f' such that $\Delta \vdash f'$ **sat** Δ', $s[f'] = s[f]$ and $s''[f'] = s'$. W.l.o.g. assume $f'(x) = f(x)$ for all $x \notin \mathrm{tgt}\, \Delta'$.

• We construct $g' : X \to_{\mathrm{Fin}} S$ such that $g'(x) = g(x)$ for all $x \notin \mathrm{tgt}\, \Delta'$, $f' \sim^{\mathsf{fh}} g'$ and $\Delta \vdash g'$ **sat** Δ'. Let $g'(x) = g(x)$ for all $x \notin \mathrm{var}\, \Delta'$. The images of the $x \in \mathrm{var}\, \Delta'$ are constructed by induction on $\mathrm{dist}(x)$.

Base case. If $\mathrm{dist}(x) = 0$ then $x \in \mathrm{root}\, \Delta'$; let $g'(x) = g(x)$.

Induction step. Otherwise $(y \xrightarrow{\beta} x) \in \Delta'$, where (by the induction hypothesis) $f'(y) \sim^{\mathsf{fh}} g'(y)$. Let $g'(x)$ be such that $\Delta \vdash g'(y) \xrightarrow{\beta} g'(x)$ and $f'(x) \sim^{\mathsf{fh}} g'(x)$ (which exists due to $\Delta \vdash f'(y) \xrightarrow{\beta} f'(x)$ and formal hypothesis bisimilarity).

• $s[g] = s[g']$ and by the cut rule, $\Delta \vdash s[g'] \xrightarrow{\alpha} s''[g']$.

• By construction, $(s''[f'], s''[g']) \in \mathscr{R}$. ∎

With respect to history-preserving bisimilarity, we can conclude the following combined property.

3.15. Corollary. *If $\Gamma \vdash t \sim^{\mathsf{hp}} u$, where $x \notin \mathrm{var}\, \Gamma$ and $\Delta \vdash t' \sim^{\mathsf{hp}} u'$, then $\Gamma \cup \Delta \vdash t[t'/x] \sim^{\mathsf{hp}} u[u'/x]$.*

3.3. Substitutive Bisimilarity

Yet another variant of bisimilarity can be defined by considering bisimulations over open terms, as in $\sim^{\mathsf{fh}}$ and $\sim^{\mathsf{hp}}$, but matching up only unconditional transitions, as in $\sim^{\mathsf{ci}}$. By itself, this does not yield a useful relation; for instance, every variable x would be identified with the deadlock constant $\mathbf{0}$, since neither has any

unconditional transitions. This deficiency, however, can be repaired by strengthening the bisimulation criterion: in addition to the usual matching of transitions, one explicitly requires bisimulations to be preserved by instantiation. Hence, $(x, \mathbf{0})$ could never appear in any bisimulation relation because, after applying $\{a.\mathbf{0}/x\}$, the terms x and $\mathbf{0}$ develop quite distinct transitions. This principle gives rise to the following definition.

3.16. DEFINITION. Let T be a (conditional or standard) transition system:

- A *substitutive bisimulation* is a symmetric relation $\mathscr{R} \subseteq S \times S$ preserved by instantiation, such that $(p, q) \in \mathscr{R}$ implies that for all $p \xrightarrow{\alpha} p'$ there is a $q \xrightarrow{\alpha} q'$ with $(p', q') \in \mathscr{R}$.
- Two states p and q are said to be *substitutive bisimilar*, denoted $p \sim^{\mathsf{sb}} q$, if there is a substitutive bisimulation $\mathscr{R}$ such that $(p, q) \in \mathscr{R}$.

Hence, substitutive bisimilarity differs from closed-instance bisimilarity in two respects: it is defined directly over open states, and the bisimulation relations are preserved by instantiation. However, the definition of substitutive bisimilarity is independent of operational environments and conditional transitions, just as for closed-instance bisimilarity. It is, therefore, surprising that substitutive bisimilarity coincides with hypotheses-preserving bisimilarity, under an assumption concerning the existence of particular kinds of states.

3.17. DEFINITION. Let T be a conditional transition system:

- For $\alpha \in L$, an α-*prefix* is a state $s_\alpha \in S$ with $\mathrm{var}(s_\alpha) = \{x\}$ for some $x \in X$, such that $\Gamma \vdash s_\alpha \xrightarrow{\beta} s'$ iff $\alpha = \beta$ and $s' = x$.
- A *choice state* is a state $s_+ \in S$ with $\mathrm{var}(s_+) = \{x, y\}$ for some distinct $x, y \in X$, such that $\Gamma \vdash s_+ \xrightarrow{\alpha} s'$ iff either $\Gamma \vdash x \xrightarrow{\alpha} s'$ or $\Gamma \vdash y \xrightarrow{\alpha} s'$.

Clearly, for instance, the prefix and choice operators of CCS give rise to these respective states. Accordingly, we will write $\alpha.p$ for $s_\alpha[p/x]$ and $p + q$ for $s_+[p/x, q/y]$. The following theorem then states the correspondence of hypothesis-preserving and substitutive bisimilarity.

3.18. THEOREM. *In a CTS with choice and prefix states,* $\sim^{\mathsf{hp}} = \sim^{\mathsf{sb}}$.

The proof is based on the idea that the effect of a hierarchical operational environment can be mimicked by substitutions that "precisely" satisfy the environment. Concretely, we construct a so-called *characteristic substitution* for every hierarchical environment Γ, which is a substitution function f_Γ with f_Γ **sat** Γ, defined by

$$f_\varnothing = id$$

$$f_{x \xrightarrow{\alpha} x', \Gamma} = \{x + \alpha.x'/x\}[f_\Gamma].$$

Intuitively (using an informal sum notation) it follows that

$$f_\Gamma : x \mapsto x + \sum \{\alpha.f_\Gamma(x') \mid (x \xrightarrow{\alpha} x') \in \Gamma\}.$$

3.19. EXAMPLE. If $\Gamma = \{x \xrightarrow{a} y, x \xrightarrow{b} z, z \xrightarrow{a} z'\}$, the characteristic substitution is given by

$$f_\Gamma = \{(x + b.(z + a.z'')) + a.y/x, z + a.z''/z\}$$

It can be seen that $f_\Gamma(x) \xrightarrow{\gamma} s$ iff either $\gamma = a$ and $s = f_\Gamma(y)$ or $\gamma = b$ and $s = f_\Gamma(z)$, and $f_\Gamma(z) \xrightarrow{\gamma} s$ iff $\gamma = a$ and $s = f_\Gamma(z')$. This precisely mimics the content of Γ.

The following proposition lists the most important properties of characteristic substitution functions.

3.20. PROPOSITION. *Let T be a conditional transition system, and let $\Gamma, \Delta \in \mathbf{\Delta}(X)$ be hierarchical:*

1. *If* $\operatorname{var}\Gamma \cap \operatorname{tgt}\Delta = \varnothing$, *then* $s[f_\Gamma][f_\Delta] = s[f_{\Gamma \cup \Delta}]$ *for arbitrary* s.
2. $\Delta \vdash f_\Gamma(x) \xrightarrow{\alpha} s$ *iff either* $s = x'$, *where* $(x \xrightarrow{\alpha} x') \in \Delta$, *or* $s = f_\Gamma(x')$, *where* $(x \xrightarrow{\alpha} x') \in \Gamma$.
3. f_Γ **sat** Γ *and* $\Delta \vdash f_\Gamma$ **sat** Δ *for arbitrary* Δ.
4. $s[f_\Gamma] \xrightarrow{\alpha} s'$ *implies* $\Gamma \vdash s \xrightarrow{\alpha} s''$ *such that* $s' = s''[f_\Gamma]$.

Proof. 1. It follows by construction that $f_\Gamma(x)[f_\Delta] = f_{\Gamma \cup \Delta}(x)$ for all $x \in \operatorname{var}\Gamma$ and $f_\Delta(x) = f_{\Gamma \cup \Delta}(x)$ for all $x \in \operatorname{var}\Delta \backslash \operatorname{var}\Gamma$. This implies the property.

2. By the construction of Γ and the properties of the action prefix and choice states.

3. Directly from Property 2 above.

4. By completeness, there is a pure $\Delta \vdash s \xrightarrow{\alpha} s_0'$ and a substitution f' such that f' **sat** Δ, $s[f_\Gamma] = s[f']$, and $s' = s_0'[f']$. W.l.o.g. assume $f'(x) = f_\Gamma(x)$ for all $x \notin \operatorname{tgt}\Delta$.

We construct a function $\phi: X \to_{\mathrm{Fin}} X$ such that $\phi(x) = x$ for all $x \in \operatorname{var}(s)$ and $f'(x) = \phi[f_\Gamma]$ for all $x \in X$, and $\phi(\Delta) \subseteq \Gamma$. By the cut rule and Lemma 2.12.1 and 3, it then follows that $\Gamma \vdash s \xrightarrow{\alpha} s_0'[\phi]$, and by the properties of substitution it follows that $s_0'[\phi][f_\Gamma] = s_0'[\phi[f_\Gamma]] = s_0'[f'] = s'$. Hence, $s'' = s_0'[\phi]$ then meets the proof obligations.

For all $x \notin \operatorname{var}\Delta$ let $\phi(x) = x$. The construction of the images of $x \in \operatorname{var}\Delta$ is by induction on $\operatorname{dist}(x)$.

Base case. If $\operatorname{dist}(x) = 0$ then $x \in \operatorname{root}\Delta \subseteq \operatorname{var}(s)$; let $\phi(x) = x$.

Induction step. Otherwise, $(y \xrightarrow{\beta} x) \in \Delta$ such that $f'(y) \xrightarrow{\beta} f'(x)$ and (by the induction hypothesis) $f'(y) = f_\Gamma(\phi(y))$. By the construction of f_Γ, it follows that $f'(x) = f_\Gamma(z)$ for some z such that $(\phi(y) \xrightarrow{\beta} z) \in \Delta$. Let $\phi(x) = z$. ∎

Proof of Theorem 3.18. First we prove $\sim^{\mathsf{hp}} \subseteq \sim^{\mathsf{sb}}$. It suffices to show $\mathscr{R} = \{(p, q) \mid \varnothing \vdash p \sim^{\mathsf{hp}} q\}$ is a substitutive bisimulation. This is easily seen to hold: $\mathscr{R}$ is preserved by instantiation as a consequence of Lemma 3.10, and the matching criterion of substitutive bisimulation is a special case of that of hypothesis preserving

bisimulation. (Note that this part of the proof does *not* depend on either characteristic substitutions or the existence of action prefix and choice states.)

Now we prove $\sim^{\mathsf{sb}} \subseteq \sim^{\mathsf{hp}}$. It suffices to show that the following is an hp-bisimulation:

$$\mathscr{R}_\Gamma = \{(p[\phi], q[\phi]) \mid \exists \text{ hierarchical } \Gamma' : p[f_{\Gamma'}] \sim^{\mathsf{sb}} q[f_{\Gamma'}], \phi(\Gamma') \subseteq \Gamma\}.$$

Clearly, all $\mathscr{R}_\Gamma$ are symmetrical.

- Let $(p[\phi], q[\phi]) \in \mathscr{R}_\Gamma$ be arbitrary, and consider $\Delta \vdash p[\phi] \xrightarrow{\alpha} p'$.
- Due to completeness, there is a pure $\Delta' \vdash p \xrightarrow{\alpha} p''$ and a substitution f' such that $\Delta \vdash f'$ **sat** Δ', $p[f'] = p[\phi]$ and $p''[f'] = p'$. W.l.o.g. assume $\mathrm{tgt}\,\Delta' \cap \mathrm{var}\,\Gamma' = \varnothing$ and $f'(x) = \phi(x)$ for all $x \notin \mathrm{tgt}\,\Delta'$.
- By induction on the distance of the variables in Δ', it can be shown that $f' = \phi'$ such that $\phi'(\Delta') \subseteq \Delta$.
- Due to Proposition 3.20.3 and the cut rule, it follows that $\Delta' \vdash p[f_{\Gamma'}] \xrightarrow{\alpha} p''[f_{\Gamma'}]$.
- Due to Proposition 3.20.3 and 1 and the cut rule, $p[f_{\Gamma'}][f_{\Delta'}] \xrightarrow{\alpha} p''[f_{\Gamma'}][f_{\Delta'}] = p''[f_{\Gamma' \cup \Delta'}]$.
- Due to $p[f_{\Gamma'}] \sim^{\mathsf{sb}} q[f_{\Gamma'}]$ it follows that $p[f_{\Gamma'}][f_{\Delta'}] \sim^{\mathsf{sb}} q[f_{\Gamma'}][f_{\Delta'}]$.
- Due to Proposition 3.20.1 and bisimilarity, $q[f_{\Gamma' \cup \Delta'}] \xrightarrow{\alpha} q'$ such that $p''[f_{\Gamma' \cup \Delta'}] \sim^{\mathsf{sb}} q'$.
- Due to Proposition 3.20.4, $\Gamma' \cup \Delta' \vdash q \xrightarrow{\alpha} q''$ for some q'' such that $q' = q''[f_{\Gamma' \cup \Delta'}]$.
- By Lemma 2.12.1 and 3, $\phi'(\Gamma' \cup \Delta') \subseteq \Gamma \cup \Delta \vdash q[\phi] = q[\phi'] \xrightarrow{\alpha} q''[\phi']$.
- By construction, $(p''[\phi'], q''[\phi']) \in \mathscr{R}_{\Gamma \cup \Delta}$. ∎

4. RECURSION

Most of the developments so far have been framed in terms of conditional transition systems and rely mainly on their properness and completeness. One the one hand, this makes the results quite general, but on the other, one may be interested in concrete applications. In Definition 2.13, we showed one particular way to generate a (proper and complete) CTS, using *tree transitions*; however, we also saw that these are only useful to deal with operators, and not with binders.

In the absence of a theory to treat binders systematically, we now turn to a specific binder whose use is almost universally accepted in process algebra, namely *recursion* (sometimes, equivalently, through finite sets of equations rather than an explicit binder) (see, e.g., [27, 33, 4]).

We have seen (Example 2.8) that substitutivity is quite a strong property; it implies preservation by the nonbinding operators of a signature. On the other hand, substitutivity does not imply binder congruence. In this section, we investigate whether the various bisimilarity relations discussed in the previous section are *recursion congruences*, i.e., are preserved by the rec-binder of CCS (cf. Example 2.4).

4.1. The Operational Semantics of Recursion

Let us first capture the operational behaviour of recursion in our framework. As we have seen, generating a CTS from tree rules as in Definition 2.13 is not sufficient for this purpose. Instead, we explicitly introduce a *recursion rule* to generate the transitions of recursive terms.

4.1. DEFINITION. Let Σ be a signature with $\Sigma_B = \{\mathrm{rec}\}$. The CTS $T^{\mathrm{rec}}_{\mathscr{S}}$ generated by a collection $\mathscr{S}$ of conditional transitions over Σ_O is defined as consisting of states $\mathbb{T}_\Sigma(V)$, variables V, and as conditional transitions the smallest set $\vdash_{\mathscr{S}}$, including $\mathscr{S}$ and all instances of the identity axiom, and closed under the cut rule and the

- Recursion rule. If $\Delta \vdash_{\mathscr{S}} t[\mathrm{rec}\, x.t/x] \xrightarrow{\alpha} t'$, then $\Delta \vdash_{\mathscr{S}} \mathrm{rec}\, x.t \xrightarrow{\alpha} t'$.

Again, this allows the application of the cut rule and the recursion rule in any order. To regulate this, we extend the previous proof normalisation result (Lemma 2.15).

4.2. LEMMA. *Assume Σ is a signature with $\Sigma_B = \{\mathrm{rec}\}$ and $\mathscr{S}$ is a collection of conditional transitions over Σ_O; $\vdash_{\mathscr{S}}$ is the smallest set of conditional transitions satisfying*:

- *if $(x \xrightarrow{\alpha} x') \in \Delta$, then $\Delta \vdash_{\mathscr{S}} x \xrightarrow{\alpha} x'$;*
- *if $(\Delta \vdash t \xrightarrow{\alpha} t') \in \mathscr{S}$ and $\Delta' \vdash_{\mathscr{S}} f$ **sat** Δ, then $\Delta' \vdash_{\mathscr{S}} t[f] \xrightarrow{\alpha} t'[f]$;*
- *if $\Delta \vdash_{\mathscr{S}} t[\mathrm{rec}\, x.t/x] \xrightarrow{\alpha} t'$, then $\Delta \vdash_{\mathscr{S}} \mathrm{rec}\, x.t \xrightarrow{\alpha} t'$.*

Proof. A straightforward adaptation of the proof of Lemma 2.15.

Furthermore, as for the case without recursion (cf. Theorem 2.19), if $\mathscr{S}$ contains only tree transitions then the properness and completeness of $T^{\mathrm{rec}}_{\mathscr{S}}$ are automatic.

4.3. THEOREM. *If Σ is a signature with $\Sigma_B = \{\mathrm{rec}\}$ and $\mathscr{S}$ is a collection of tree transitions over Σ, then $T^{\mathrm{rec}}_{\mathscr{S}}$ is a proper and complete CTS.*

Proof. The proof is by induction on the construction of the conditional transitions $\Delta \vdash_{\mathscr{S}} t[f] \xrightarrow{\alpha} t'$ according to Lemma 4.2. It is analogous to the proof of Theorem 2.19, except that we have to consider an extra case.

- $t = \mathrm{rec}\, x.t_0$. W.l.o.g. assume $x \notin \mathrm{var}(t[f])$; hence $t[f] = \mathrm{rec}\, x.t_0[f]$. By Lemma 4.2, $\Delta \vdash t_0[f][t[f]/x] \xrightarrow{\alpha} t'$ is constructed in fewer steps than $\Delta \vdash t[f] \xrightarrow{\alpha} t'$; hence the induction hypothesis applies. Since $t_0[f][t[f]/x] = t_0[t/x][f]$, by the induction hypothesis there are a pure $\Delta' \vdash t_0[t/x] \xrightarrow{\alpha} t''$ and f' such that $\Delta \vdash f'$ **sat** Δ', $t_0[t/x][f'] = t[t/x][f]$ and $t''[f'] = t'$. But then also $\Delta' \vdash t \xrightarrow{\alpha} t''$ by the recursion rule, which is pure as well. Since $f'(x) = f(x)$ for all $x \in \mathrm{var}(t_0[t/x]) = \mathrm{var}(t)$, it follows that $t[f'] = t[f]$; hence we are done. ∎

It turns out, somewhat surprisingly, that closed-instance bisimilarity is *not* a recursion congruence for all $T^{\mathrm{rec}}_{\mathscr{S}}$. A problem occurs if there is *look-ahead* in the hypotheses of the transitions in $\mathscr{S}$, i.e., sequences of assumptions $(x \xrightarrow{\beta} x')$, $(x' \xrightarrow{\gamma} x'') \in \Delta$ for some $(\Delta \vdash t \xrightarrow{\alpha} t') \in \mathscr{S}$. If we forbid look-ahead by requiring $\mathrm{src}\,\Delta \cap \mathrm{tgt}\,\Delta = \varnothing$ for all transitions in $\mathscr{S}$, $\sim^{\mathrm{ci}}$ becomes a recursion congruence. On

the other hand, both hypothesis-based bisimilarity relations are always recursion congruences. This is stated in the following theorem, whose proof follows later.

4.4. THEOREM. *Let Σ be a signature with $\Sigma_B = \{\mathrm{rec}\}$ and $\mathscr{S}$, a set of tree transitions over Σ.*

1. *$\sim^{\mathsf{ci}}$ is a recursion congruence over $T^{\mathrm{rec}}_{\mathscr{S}}$ if the transitions in $\mathscr{S}$ have no look-ahead.*
2. *$\sim^{\mathsf{fh}}$ and $\sim^{\mathsf{hp}}$ are recursion congruences over $T^{\mathrm{rec}}_{\mathscr{S}}$.*

4.2. Bisimulation Up-to

To prove Theorem 4.4 for $\sim^{\mathsf{ci}}$, we adopt a proof technique used in [33] for CCS, called *up-to bisimulation*; see also [36]. This is based on the following idea: to prove $\mathrm{rec}\, x.t \sim^{\mathsf{ci}} \mathrm{rec}\, x.u$ under the assumption that $t \sim^{\mathsf{ci}} u$, we prove $v[\mathrm{rec}\, x.t/x] \sim^{\mathsf{ci}} v[\mathrm{rec}\, x.u/x]$ for arbitrary v; then in particular, $\mathrm{rec}\, x.t = x[\mathrm{rec}\, x.t/x] \sim^{\mathsf{ci}} x[\mathrm{rec}\, x.u/x] = \mathrm{rec}\, x.u$.

$v[\mathrm{rec}\, x.t/x] \sim^{\mathsf{ci}} v[\mathrm{rec}\, x.u/x]$ is proved by induction on the proof depth of the *initial* transitions of these terms. However, look-ahead in the hypotheses of an operator gives rise to assumptions about *noninitial* transitions; thus, the induction breaks down. The following example shows that $\sim^{\mathsf{ci}}$ may indeed fail to be a congruence for operators whose behaviour relies on look-ahead.

4.5. EXAMPLE. Consider an extension of CCS with *negative action prefix*, *determinisation*, and *left-merge*. The family of *negative action prefix* operators $\alpha^{-1}.\_$ is operationally defined by the transition

$$x \xrightarrow{\alpha} x',\ x' \xrightarrow{\beta} x'' \vdash \alpha^{-1}.x \xrightarrow{\beta} x''.$$

Hence $\alpha^{-1}.t$ "predicts" the behaviour of t, after an initial α, in such a way that any choice made during that initial α-transition is wiped out. For instance, $\alpha^{-1}.(\alpha.\mathbf{0} + \alpha.\beta.\mathbf{0})$ and $\beta.\mathbf{0}$ have bisimilar behaviours. a-determinisation, denoted $((\_))_a$, is then obtained by restricting to a-steps and extending the prediction at every step:

$$x \xrightarrow{a} x' \vdash ((x))_a \xrightarrow{a} ((a^{-1}.x))_a.$$

Furthermore, we recall the *left-merge* operator $\_ \between \_$ from [4], adapted to CCS parallel composition:

$$x \xrightarrow{\beta} x' \vdash x \between y \xrightarrow{\beta} x' \mid y.$$

Now consider the terms

$$t = a.\mathbf{0} + ((x \between a.\mathbf{0}))_a \sim^{\mathsf{ci}} a.\mathbf{0} + ((x \between a.\mathbf{0}))_a + a.((x))_a = u.$$

This relation holds because the only subterm that could make a difference, namely $a.((x))_a$ in u, for every closed instantiation v of x equals one of the subterms of t.

Either v can do no a's at all, in which case $a.((v))_a$ is simulated by $a.\mathbf{0}$; or v can do at most $n \in \mathbb{N}$ a's in succession, in which case $a.((v))_a$ can do *precisely* $n+1$ a's, which is simulated by $((v \between a.\mathbf{0}))_a$; or v can do a^∞, in which case $a.((v))_a$ is equivalent to rec $y.a.y$, which in turn is equivalent to $((v \between a.\mathbf{0}))_a$. On the other hand,

$$\text{rec } x.t \not\sim^{\mathbf{ci}} \text{rec } x.u.$$

Here, the right-hand side can do an infinite a-sequence by choosing the $a.((-))_a$-branch at each successive step:

$$\begin{aligned} \text{rec } a.u &\xrightarrow{a} ((\text{rec } a.u))_a \xrightarrow{a} ((a^{-1}.((\text{rec } a.u))_a))_a \\ &\xrightarrow{a} ((a^{-1}.((a^{-1}.((\text{rec } a.u))_a))_a))_a \\ &\xrightarrow{a} \cdots. \end{aligned}$$

The left-hand side cannot match this; rec $x.t$ can be unfolded only finitely many times during the initial transition. Hence at some depth, the $a.\mathbf{0}$-branch must be chosen.

Proof of Theorem 4.4. Let Σ be a signature with $\Sigma_B = \{\text{rec}\}$, and let $\mathscr{S}$ be a collection of transitions over Σ without look-ahead. Let

$$\mathscr{R} = \{(v[\text{rec } x.t/x], v[\text{rec } x.u/x]) \mid t, u, v \in \mathbb{T}_{\mathscr{S}}(\{x\}),\ t \sim^{\mathbf{ci}} u\}.$$

We prove that $\mathscr{R}$ is a **ci**-bisimulation *up to* $\sim^{\mathbf{ci}}$, in the sense of [35]. That is, we prove that for all $(t_0, u_0) \in \mathscr{R}$, if $t_0 \xrightarrow{\alpha} t_0'$, then $u_0 \xrightarrow{\alpha} u_0' \sim^{\mathbf{ci}} u_0''$, such that $(t_0', u_0'') \in \mathscr{R}$. This implies $\mathscr{R} \subseteq \sim^{\mathbf{ci}}$. Since $v = x$ is a special case in the definition of $\mathscr{R}$, it follows that rec $x.t \sim^{\mathbf{ci}}$ rec $x.u$ whenever $t \sim^{\mathbf{ci}} u$, where t and u contain at most x as a free variable. The result for general t and u follows by the definition of $\sim^{\mathbf{ci}}$ through closed instantiations.

The proof proceeds by induction on the "normal proof" construction of $\vdash_{\mathscr{S}} v[\text{rec } x.t/x] \xrightarrow{\alpha} v'$, according to Lemma 4.2:

- $v[\text{rec } x.t/x] = y$ and $v' = y'$ such that $(y \xrightarrow{\alpha} y') \in \Delta$. This is not applicable, since $v[\text{rec } x.t/x]$ is a closed term.
- $v[\text{rec } x.t/x] = op(x_1, ..., x_{|op|})[f]$ and $v' = t'[f]$ for some $(\Delta \vdash op(x_1, ..., x_{|op|}) \xrightarrow{\alpha} t') \in \mathscr{S}$ and $\vdash_{\mathscr{S}} f$ **sat** Δ. It follows that $v = op(v_1, ..., v_{|op|})$ such that $x_i[f] = v_i[\text{rec } x.t/x]$ for $1 \leqslant i \leqslant |op|$. Let $\Delta = \{y_k \xrightarrow{\beta_k} y_k' \mid k \in K\}$.

Since there is no look-ahead, in Δ, src Δ = root $\Delta \subseteq \{x_i \mid 1 \leqslant i \leqslant |op|\}$; hence for all $k \in K$, $y_k = x_{i_k}$ for some $1 \leqslant i_k \leqslant |op|$. Then $\vdash_{\mathscr{S}} f$ **sat** Δ implies $v_{i_k}[\text{rec } x.t/x] \xrightarrow{\beta_k} f(y_k')$ for all $k \in K$. By the induction hypothesis, $\vdash_{\mathscr{S}} v_{i_k}[\text{rec } x.u/x] \xrightarrow{\beta_k} v_k' \sim^{\mathbf{ci}} v_k''$ for some v_k' and v_k'' such that $(f(y_k'), v_k'') \in \mathscr{R}$, which in turn implies $f(y_k') = t_k'[\text{rec } x.t/x]$ and $v_k'' = t_k'[\text{rec } x.u/x]$ for some t_k'. Now let

$$\begin{aligned} g &= \{v_{i_k}[\text{rec } x.u/x]/y_k \mid k \in K\} \cup \{v_k'/y_k' \mid k \in K\} \\ g' &= \{v_{i_k}[\text{rec } x.u/x]/y_k \mid k \in K\} \cup \{v_k''/y_k' \mid k \in K\} \\ h &= \{v_{i_k}/y_k \mid k \in K\} \cup \{t_k'/y_k' \mid k \in K\}. \end{aligned}$$

It follows that $\vdash_{\mathscr{S}} g$ **sat** Δ, hence (by the cut rule and $op(x_1, ..., x_{|op|})[g] = v[\text{rec } x.u/x]) \vdash_{\mathscr{S}} v[\text{rec } x.u/x] \xrightarrow{\alpha} t'[g]$. Moreover, $t'[g] \sim^{\mathbf{ci}} t'[g']$ since $\sim^{\mathbf{ci}}$ is preserved by insertion (cf. Theorem 3.9), and $t'[f] = t'[h][\text{rec } x.t/x]$ and $t'[g'] = t'[h][\text{rec } x.u/x]$ by construction of g' and h, implying $(t'[f], t'[g']) \in \mathscr{R}$.

- $v[\text{rec } x.t/x] = \text{rec } y.v_0$ and $\vdash_{\mathscr{S}} v_0[\text{rec } y.v_0/y] \xrightarrow{\alpha} v'$. There are two subcases:

— $v = x = y$ and $v_0 = t$. Since the construction of $\vdash_{\mathscr{S}} v_0[\text{rec } y.v_0/y] \xrightarrow{\alpha} v'$ takes fewer steps, by the induction hypothesis there is a $\vdash_{\mathscr{S}} t[\text{rec } x.u/x] \xrightarrow{\alpha} t' \sim^{\mathbf{ci}} v''$ such that $(v', v'') \in \mathscr{R}$. Due to $t \sim^{\mathbf{ci}} u$ and the fact that $\sim^{\mathbf{ci}}$ is preserved by instantiation, it follows that $t[\text{rec } x.u/x] \sim^{\mathbf{ci}} u[\text{rec } x.u/x]$; hence, there is a $\vdash_{\mathscr{S}} u[\text{rec } x.u/x] \xrightarrow{\alpha} u'$ with $t' \sim^{\mathbf{ci}} u'$; hence, also $u' \sim^{\mathbf{ci}} v''$. Since (by the recursion rule) $\vdash_{\mathscr{S}} \text{rec } x.u \xrightarrow{\alpha} u'$, we are done.

— $v = \text{rec } y.v_1$ with $y \neq x$ and $v_0 = v_1[\text{rec } x.t/x]$. Since $\text{var}(v_0) \subseteq \{y\}$, it follows that

$$v_0[\text{rec } y.v_0/y] = v_1[\text{rec } x.t/x][v[\text{rec } x.t/x]/y] = v_1[v/y][\text{rec } x.t/x].$$

By the induction hypothesis, therfore, $\vdash_{\mathscr{S}} v_1[v/y][\text{rec } x.u/x] \xrightarrow{\alpha} v_1' \sim^{\mathbf{ci}} v''$ such that $(v', v'') \in \mathscr{R}$. Furthermore,

$$v_1[v/y][\text{rec } x.u/x] = v_1[\text{rec } x.u/x][\text{rec } y.v_1[\text{rec } x.u/x]/y]$$

and, hence, the recursion rule implies $\vdash_{\mathscr{S}} \text{rec } y.v_1[\text{rec } x.u/x] \xrightarrow{\alpha} v_1'$. Finally,

$$\text{rec } y.v_1[\text{rec } x.u/x] = (\text{rec } y.v_1)[\text{rec } x.u/x] = v[\text{rec } x.u/x];$$

hence we are done. ∎

4.3. *Open Approximants*

The proof of Theorem 4.4 for $\sim^{\mathbf{fh}}$ and $\sim^{\mathbf{hp}}$ relies on a notion of *open approximants*:

$$\text{rec}^i\, x.t = \begin{cases} x, & \text{if } \ i = 0, \\ t = [\text{rec}^{i-1}\, x.t/x], & \text{otherwise.} \end{cases}$$

Hence, $\text{rec}^i\, x.t$ is obtained by repeatedly (viz. i times) substituting t for x, starting with x. The idea is that for larger i, the behaviour of $\text{rec}^i\, x.t$ gets closer and closer to that of $\text{rec}\, x.t$. The difference with the usual approximants lies in the choice of starting point, i.e., the definition of $\text{rec}^0\, x.t$, which in the standard case equals the bottom element of a c.p.o., usually either **0** or some special divergence constant.

The following "additivity property" of open approximants is used in the proof below; it is proved by a straightforward induction on the approximation depth.

4.6. Proposition. *For all* $i, j \in \mathbb{N}$, $(\text{rec}^i\, x.t)[\text{rec}^j\, x.t/x] = \text{rec}^{i+j}\, x.t$.

Based on open approximants, we now define an approximation relation over arbitrary terms, such that t approaches u if u contains recursive subterms and t is obtained from u by replacing some of them by open approximants. For this purpose, we assume a family of variables $\{\hat{x}_i \mid i \in \mathbb{N}\} \subseteq V$ not used anywhere else. The approximation relation then looks as

$$u <_{x,t} v \Leftrightarrow \exists \hat{t} : x \notin \mathrm{var}(\hat{t}),\ u = \hat{t}[\mathrm{rec}^i\, x.t/\hat{x}_i]_{\hat{x}_i \in \mathrm{var}(\hat{t})},\ v = \hat{t}[\mathrm{rec}\, x.t/\hat{x}_i]_{\hat{x}_i \in \mathrm{var}(\hat{t})}.$$

We call $\hat{t}$ the *seed term* of the approximation relation, of which the terms u and v under comparison are instantiations. $u <_{x,t} v$ expresses that u approaches v, in the sense that u can be obtained from v by replacing some of the subterms $\mathrm{rec}\, x.t$ of v by open approximants. The following proposition contains an alternative characterisation for $<_{x,t}$.

4.7. PROPOSITION. *$<_{x,t}$ is the smallest insertion-preserved relation such that $\mathrm{rec}^i\, x.t <_{x,t} \mathrm{rec}\, x.t$ for all $i \in \mathbb{N}$.*

Proof. It is clear that $\mathrm{rec}^i\, x.t <_{x,t} \mathrm{rec}\, x.t$ for all $i \in \mathbb{N}$ (take $\hat{x}_i$ as seed term). We now prove that $<_{x,t}$ is preserved by insertion. Consider $u \in \mathbb{T}_\Sigma(V)$ and $f, g: X \to_{\mathrm{Fin}} \mathbb{T}_\Sigma(V)$ such that $f(y) <_{x,t} g(y)$ for all $y \in \mathrm{var}(u)$, with seed term $\hat{t}_y$. We construct a new seed term $\hat{t} = u[h]$, where $h = \{\hat{t}_y/y \mid y \in \mathrm{var}(u)\}$. It follows that

$$u[f] = u[h][\mathrm{rec}^i\, x.t/\hat{x}_i]_{\hat{x}_i \in \mathrm{var}(\hat{t})} = \hat{t}[\mathrm{rec}^i\, x.t/\hat{x}_i]_{\hat{x}_i \in \mathrm{var}(\hat{t})}$$
$$u[g] = u[h][\mathrm{rec}\, x.t/\hat{x}_i]_{\hat{x}_i \in \mathrm{var}(\hat{t})} = \hat{t}[\mathrm{rec}\, x.t/\hat{x}_i]_{\hat{x}_i \in \mathrm{var}(\hat{t})},$$

proving $u[f] <_{x,t} u[g]$.

Finally, let $<$ be the smallest insertion-preserved relation such that $\mathrm{rec}^i\, x.t < \mathrm{rec}\, x.t$ for all $i \in \mathbb{N}$; we have to prove $<_{x,t} \subseteq <$. Assume $u <_{x,t} v$ with seed term $\hat{t}$, and let

$$f = \{\mathrm{rec}^i\, x.t/\hat{x}_i \mid \hat{x}_i \in \mathrm{var}(\hat{t})\}$$
$$g = \{\mathrm{rec}\, x.t/\hat{x}_i \mid \hat{x}_i \in \mathrm{var}(\hat{t})\}.$$

Since $y < y$ for all $y \in V$, it follows that $f(y) < g(y)$ for all $y \in \mathrm{var}(\hat{t})$; since $<$ is preserved by insertion, we may conclude $u = \hat{t}[f] < \hat{t}[g] = v$. ∎

The following lemma expresses the fundamental behavioural relation between terms u and v such that $u <_{x,t} v$. It assumes a signature Σ with $\Sigma_B = \{\mathrm{rec}\}$ and a collection $\mathscr{S}$ of tree transitions over Σ.

4.8. PROPOSITION. *Let $u <_{x,t} v$:*

1. *If $\Delta \vdash_{\mathscr{S}} u \xrightarrow{\alpha} u'$ such that $x \notin \mathrm{var}\, \Delta$, then $\Delta \vdash_{\mathscr{S}} v \xrightarrow{\alpha} v'$ such that $u' <_{x,t} v'$.*
2. *If $\Delta \vdash_{\mathscr{S}} v \xrightarrow{\alpha} v'$, then there is a $k \in \mathbb{N}$ such that $\Delta \vdash_{\mathscr{S}} u[\mathrm{rec}^k\, x.t/x] \xrightarrow{\alpha} u'$ and $u' <_{x,t} v'$.*

Proof. Assume $\hat{t}$ is the seed term of $u <_{x,t} v$.

1. This part of the proof depends on an induction on the largest i such that $\hat{x}_i \in \mathrm{var}(\hat{t})$ (since $\mathrm{var}(\hat{t})$ is finite, there is such a largest i.) Let us write $u <^k_{x,t} v$ to signal that $\forall \hat{x}_i \in \mathrm{var}(\hat{t}) : i \leqslant k$. Note that $<^k_{x,t}$ is also preserved by insertion (proved by a slight variation on the proof of Proposition 4.7). Instead of the clause in the lemma, we now prove the following strengthened property:

If $u <^k_{x,t} v$ and $\Delta \vdash_{\mathscr{S}} u \xrightarrow{\alpha} u'$ with $x \notin \mathrm{var}\,\Delta$, then $\Delta \vdash_{\mathscr{S}} v \xrightarrow{\alpha} v'$ such that $u' <^k_{x,t} v'$.

The proof proceeds by induction on k.

Base case. If $k = 0$, then $u = \hat{t}[x/\hat{x}_0]$ and (since $x \notin \mathrm{var}(\hat{t})$) also $\hat{t} = u[\hat{x}_0/x]$, thus $v = u[\mathrm{rec}\, x.t/x]$. Since $x \notin \mathrm{var}\,\Delta$, by Lemma 2.12.2 we may conclude $\Delta \vdash_{\mathscr{S}} v \xrightarrow{\alpha} v'$ with $v' = u'[\mathrm{rec}\, x.t/x]$, where $u' <_{x,t} v'$ with seed term $u'[\hat{x}_0/x]$.

Induction step. Assume that the property holds whenever $u <^k_{x,t} v$, and consider $u <^{k+1}_{x,t} v$. By completeness (Theorem 4.3), there is a pure $\Delta' \vdash_{\mathscr{S}} \hat{t} \xrightarrow{\alpha} \hat{t}'$ and an f' such that $\Delta \vdash_{\mathscr{S}} f'$ **sat** Δ', $u = \hat{t}[f']$ (implying that for all $y \in \mathrm{var}(\hat{t})$, $f'(y) = \mathrm{rec}^i\, x.t$ if $y = \hat{x}_i$ and $f'(y) = y$ if $y \notin \{\hat{x}_i\}_{i \leqslant k+1}$) and $u' = \hat{t}'[f']$. It follows that there is a sequence $\varnothing = \Delta'_0 \subset \cdots \subset \Delta'_n = \Delta'$ where for all $1 \leqslant m \leqslant n$, $\Delta'_m = \Delta'_{m-1} \cup \{y_m \xrightarrow{\beta_m} y'_m\}$ with $y_m \in \mathrm{var}(\hat{t}) \cup \mathrm{var}\,\Delta'_{m-1}$ and $y'_m \notin \mathrm{var}\,\Delta'_{m-1}$. By induction on m, we construct a substitution function g_m with

— $g_m(\hat{x}_i) = \mathrm{rec}\, x.t$ for all $i \leqslant k+1$ and $g_m(y) = y$ if $y \in \mathrm{var}(\hat{t}) \backslash \{\hat{x}_i\}_{i \leqslant k+1}$;

— $f'(z) <^k_{x,t} g_m(z)$ for all $z \in tgt\,\Delta'_m$;

— $\Delta \vdash_{\mathscr{S}} g_m$ **sat** Δ'_m.

It then follows that $\hat{t}[g_n] = v$ and (since $\mathrm{var}(\hat{t}') \subseteq \mathrm{var}(\hat{t}) \cup \mathrm{var}\,\Delta'$ and $<^{k+1}_{x,t}$ is preserved by insertion) $u' <^{k+1}_{x,t} v'$ with $v' = \hat{t}'[g_n]$; moreover, by the cut rule, $\Delta \vdash_{\mathscr{S}} \hat{t}[g_n] \xrightarrow{\alpha} v'$. The construction of the g_m is by induction on m.

Base case. $g_0 = f' \pm \{\mathrm{rec}\, x.t/x_i\}_{i \leqslant k+1}$.

Induction step. Assume that g_m satisfies the requirements, and consider the transition $\Delta \vdash_{\mathscr{S}} f'(y_{m+1}) \xrightarrow{\beta_{m+1}} f'(y'_{m+1})$. There are three cases.

• $y_{m+1} = \hat{x}_i$ for some $1 \leqslant i \leqslant k+1$. It follows that $f'(y_{m+1}) = \mathrm{rec}^i\, x.t$ and $g_m(y_{m+1}) = \mathrm{rec}\, x.t$. Due to $\mathrm{rec}^i\, x.t <^k_{x,t} t[\mathrm{rec}\, x.t/x]$, by the outer induction hypothesis $\Delta \vdash_{\mathscr{S}} t[\mathrm{rec}\, x.t/x] \xrightarrow{\beta_{m+1}} v'_{m+1}$ such that $f'(y'_{m+1}) <^k_{x,t} v'_{m+1}$. But then (by the recursion rule) also $\Delta \vdash_{\mathscr{S}} g_m(y_{m+1}) \xrightarrow{\beta_{m+1}} v'_{m+1}$; hence $g_{m+1} = g_m \pm \{v'_{m+1}/y'_{m+1}\}$ satisfies the requirements.

• $y_{m+1} \in \mathrm{var}(\hat{t}) \backslash \{\hat{x}_i\}_{i \leqslant k+1}$. It follows that $f'(y_{m+1}) = y_{m+1}$ and hence (since the CTS is proper) $(y_{m+1} \xrightarrow{\beta_{m+1}} y'_{m+1}) \in \Delta$. Then $g_{m+1} = g_m$ satisfies the requirements.

• $y_{m+1} \in tgt\,\Delta'_m$. In that case, by the inner induction hypothesis, $f'(y_{m+1}) <^k_{x,t} g_m(y_{m+1})$; hence, by the outer induction hypothesis, $\Delta \vdash_{\mathscr{S}} g_m(y_{m+1}) \xrightarrow{\beta_{m+1}} v'_{m+1}$ such that $f'(y'_{m+1}) <^k_{x,t} v'_{m+1}$. Then $g_{m+1} = g_m \pm \{v'_{m+1}/y'_{m+1}\}$ satisfies the requirements.

2. By induction on the construction of $\Delta \vdash_{\mathscr{S}} v \xrightarrow{\alpha} v'$ according to Lemma 4.2. There are three cases:

- $v = y$ and $v' = y'$ such that $(y \xrightarrow{\alpha} y') \in \Delta$. It follows that $\hat{t} = y$ and, hence, $u = y$. Hence, setting $u' = y'$, it follows that $\Delta \vdash u \xrightarrow{\alpha} u'$ and $u' <_{x,t} v'$.

- $v = op(y_1, ..., y_{|op|})[f]$ and $v' = t'[f]$ for some $(\Delta' \vdash op(y_1, ..., y_{|op|}) \xrightarrow{\alpha} t') \in \mathscr{S}$ such that $\Delta \vdash_{\mathscr{S}} f$ **sat** Δ'. It follows that $u = op(u_1, ..., u_{|op|})$ such that $u_j <_{x,t} y_j[f]$ for $1 \leqslant j \leqslant |op|$. Let $Y = \{y_j \mid 1 \leqslant j \leqslant |op|\}$.

Since Δ' is hierarchical, it can be constructed through a sequence $\varnothing = \Delta'_0 \subset \Delta'_1 \subset \cdots \subset \Delta'_n = \Delta'$, where for all $1 \leqslant m \leqslant n$, $\Delta'_m = \Delta'_{m-1} \cup \{z_m \xrightarrow{\beta_m} z'_m\}$ with $z_m \in Y \cup \text{var}\,\Delta'_{m-1}$ and $z'_m \notin \text{var}\,\Delta'_{m-1}$. By induction on m, we prove that there is a constant k_m and a substitution function g_m such that

— $g_m(y_j) = u_j[\text{rec}^{k_m} x.t/x]$ for $1 \leqslant j \leqslant |op|$,
— $g_m(z) <_{x,t} f(z)$ for all $z \in Y \cup \text{var}\,\Delta'_m$;
— $\Delta \vdash_{\mathscr{S}} g_m$ **sat** Δ'_m.

It follows that $\Delta \vdash_{\mathscr{S}} g_n$ **sat** Δ'; hence (due to the cut rule) $\Delta \vdash_{\mathscr{S}} u[\text{rec}^{k_n} x.t/x] = op(y_1, ..., y_{|op|})[g_n] \xrightarrow{\alpha} t'[g_n] = u'$, with $v' = t'[f] <_{x,t} t'[g_n] = u'$ since $<_{x,t}$ is preserved by insertion (Proposition 4.7) and $\text{var}(t') \subseteq Y \cup \text{var}\,\Delta'$.

Base case. Let $k_0 = 0$ and $g_0 = \{u_j/y_j\}_{1 \leqslant j \leqslant |op|}$.

Induction step. Assume that k_m and g_m satisfy the assumptions ($m < n$). It follows that $g_m(z_{m+1}) <_{x,t} f(z_{m+1})$ by the inner induction hypothesis; hence $\exists k$: $\Delta \vdash_{\mathscr{S}} g_m(z_{m+1})[\text{rec}^k z.t/x] \xrightarrow{\beta_{m+1}} u'_{m+1}$ such that $f(z'_{m+1}) <_{x,t} u'_{m+1}$ by the outer induction hypothesis. Let $k_{m+1} = k_m + k$ and $g_{m+1} = g_m[\text{rec}^k x.t/x] \pm \{u'_{m+1}/z'_{m+1}\}$.

- $v = \text{rec}\, y.v_0$ and $\Delta \vdash_{\mathscr{S}} v_0[v/y] \xrightarrow{\alpha} v'$. There are two cases.

— $x = y$ and $v_0 = t$. It follows that $v = \text{rec}\, x.t$ and $u = \text{rec}^i x.t$ for some $i \in \mathbb{N}$; hence $t <_{x,t} t[\text{rec}\, x.t/x] = v_0[v/x]$. By the induction hypothesis, it follows that there is a k such that $\Delta \vdash t[\text{rec}^k x.t/x] \xrightarrow{\alpha} u'$ with $u' <_{x,t} v'$. Note that $t[\text{rec}^k x.t/x] = \text{rec}^{k+1} x.t$. Due to the cut rule and Proposition 4.6, it follows that $\Delta \vdash \text{rec}^{i+k+1} x.t \xrightarrow{\alpha} u''$ with $u'' = u'[\text{rec}^i x.t/x]$; moreover, it is not difficult to see that $u'' <_{x,t} v'$. Again due to Proposition 4.6, $\text{rec}^{i+k+1} x.t = u[\text{rec}^{k+1} x.t/x]$; hence, we are done.

— $x \neq y$; w.l.o.g. assume $y \notin \text{var}(t)$. Then $u = \text{rec}\, y.u_0$ and $\hat{t} = \text{rec}\, y.\hat{t}_0$ such that $u_0 <_{x,t} v_0$ with seed term $\hat{t}_0$. It follows that $u_0[u/y] <_{x,t} v_0[v/y]$ with seed term $\hat{t}_0[\hat{t}/y]$; hence by the induction hypothesis, there is a $k \in \mathbb{N}$ such that

$$\Delta \vdash \quad u_0[u/y][\text{rec}^k x.t/x] \quad \xrightarrow{\alpha} u'$$

$$=$$

$$u_0[\text{rec}^k x.t/x][\text{rec}\, y.u_0[\text{rec}^k x.t/x]/y]$$

and $u' <_{x,t} v'$. By the recursion rule it then follows that $\Delta \vdash_{\mathscr{S}} \text{rec}\, y.u_0[\text{rec}^k x.t/x] \xrightarrow{\alpha} u'$. Since $\text{rec}\, y.u_0[\text{rec}^k x.t/x] = u[\text{rec}^k x.t/x]$, we are done.

The above lemma enables us to prove that both $\sim^{\mathsf{fh}}$ and $\sim^{\mathsf{hp}}$ are recursion congruences.

Proof of Theorem 4.4.2. The proof for $\sim^{\mathsf{fh}}$ runs along the following lines. Assume $t \sim^{\mathsf{fh}} u$; by repeated application of Theorem 3.9, it follows that $\mathrm{rec}^k\, x.t \sim^{\mathsf{fh}} \mathrm{rec}^k\, x.u$ for all $k \in \mathbb{N}$. Now consider

$$\mathscr{R} = \{(t_0, u_0) \mid \exists t_1 <_{x,t} t_0, u_1 <_{x,u} u_0 : t_1 \sim^{\mathsf{fh}} u_1\}.$$

We prove this to be a formal hypothesis bisimulation relation. Note that $\mathscr{R}$ is symmetrical.

- Assume $(t_0, u_0) \in \mathscr{R}$ and $\Gamma \vdash t_0 \xrightarrow{\alpha} t_0'$; hence $x \notin \mathrm{var}\,\Gamma$.
- Due to Proposition 4.8.2, there is a $\Gamma \vdash t_1[\mathrm{rec}^k\, x.t/x] \xrightarrow{\alpha} t_1'$ such that $t_1' <_{x,t} t_0'$;
- Since $t_1 \sim^{\mathsf{fh}} u_1$ and $\mathrm{rec}^k\, x.t \sim^{\mathsf{fh}} \mathrm{rec}^k\, x.u$, due to Theorem 3.9 $t_1[\mathrm{rec}^k\, x.t/x] \sim^{\mathsf{fh}} u_1[\mathrm{rec}^k\, x.u/x]$;
- There is a $\Gamma \vdash u_1[\mathrm{rec}^k\, x.u/x] \xrightarrow{\alpha} u_1'$ such that $t_1' \sim^{\mathsf{fh}} u_1'$;
- Due to Proposition 4.8.1, there is a $\Gamma \vdash u_0 \xrightarrow{\alpha} u_0'$ such that $u_1' <_{x,u} u_0'$;
- By construction, $(t_0', u_0') \in \mathscr{R}$.

Since $x \sim^{\mathsf{fh}} x$, $x <_{x,t} \mathrm{rec}\, x.t$ and $x <_{x,u} \mathrm{rec}\, x.u$, it follows that $(\mathrm{rec}\, x.t, \mathrm{rec}\, x.u) \in \mathscr{R}$; this concludes the proof.

For $\sim^{\mathsf{hp}}$, the proof is slightly more complicated, due to the growing indices in history preserving bisimulations. Assume $t \sim^{\mathsf{hp}} u$; by repeated application of Theorem 3.9, it follows that $\mathrm{rec}^k\, x.t \sim^{\mathsf{hp}} \mathrm{rec}^k\, x.u$ for all $k \in \mathbb{N}$. Now for all Γ with $x \notin \mathrm{var}\,\Gamma$ consider

$$\mathscr{R}_\Gamma = \{(t_0, u_0) \mid \exists t_1 <_{x,t} t_0, u_1 <_{x,u} u_0 : \Gamma \vdash t_1 \sim^{\mathsf{hp}} u_1\}.$$

We prove this to be an hp-bisimulation. Note that $\mathscr{R}_\Gamma$ is symmetrical.

- Assume $(t_0, u_0) \in \mathscr{R}_\Gamma$ and a pure $\Delta \vdash t_0 \xrightarrow{\alpha} t_0'$; w.l.o.g. assume $x \notin \mathrm{var}\,\Delta$.
- Due to Proposition 4.8.2, there is a $\Delta \vdash t_1[\mathrm{rec}^k\, x.t/x] \xrightarrow{\alpha} t_1'$ such that $t_1' <_{x,t} t_0'$;
- Due to Corollary 3.15, $\Gamma \vdash t_1[\mathrm{rec}^k\, x.t/x] \sim^{\mathsf{hp}} u_1[\mathrm{rec}^k\, x.u/x]$;
- There is a $\Gamma \cup \Delta \vdash u_1[\mathrm{rec}^k\, x.u/x] \xrightarrow{\alpha} u_1'$ such that $\Gamma \cup \Delta \vdash t_1' \sim^{\mathsf{hp}} u_1'$;
- Due to Proposition 4.8.1, there is a $\Gamma \cup \Delta \vdash u_0 \xrightarrow{\alpha} u_0'$ such that $u_1' <_{x,u} u_0'$;
- By construction, $(t_0', u_0') \in \mathscr{R}_{\Gamma \cup \Delta}$.

Since $\varnothing \vdash x \sim^{\mathsf{hp}} x$, $x <_{x,t} \mathrm{rec}\, x.t$ and $x <_{x,u} \mathrm{rec}\, x.u$, it follows that $(\mathrm{rec}\, x.t, \mathrm{rec}\, x.u) \in \mathscr{R}_\varnothing$; this concludes the proof. ∎

5. CONCLUSIONS

5.1. Related Work

There is a large amount of related work from different fields, some of which we already mentioned in the course of the paper.

SOS Formats. Conditional transitions have much in common with Plotkin style operational rules. We already discussed the similarities in Section 2; the essential difference is the nature of the variables used in the formalism, which for operational rules are *meta-variables* but for conditional transitions are *term variables*. It is

precisely this fact that allowed us to prove the *completeness* of conditional transition systems, and thereby the *substitutivity* of the various open term bisimilarities, also in the presence of recursion in the language (Theorem 4.3). As we saw before (Example 2.8), in the presence of binders such as recursion, substitutivity is a stronger property than preservation by the first-order (i.e., nonbinding) operators.

With respect to the operational behaviour of first-order operators, we have shown that one particular way to generate complete CTS's is by using *conditional tree transitions*; in the analogy above (i.e., replacing term variables by meta-variables), these correspond to Fokkink and Van Glabbeek's *tree rules* [16] or, alternatively, the *pure xyft* subformat of Groote and Vaandrager [25].

It should be noted that the concept of formal hypothesis bisimilarity can be formulated just as well on the basis of meta-variables as on term variables (as we did): the hypotheses to be matched are then the premises of operational "ruloids" (that is, fragments of derivation trees) rather than the environments of conditional transitions. In fact, this was the original definition of $\sim^{\mathsf{fh}}$ in [14] and has also been studied by Aceto, Bloom, and Vaandrager in [2] (who go into the issue of equational proof systems for this relation).

Formal Hypothesis Bisimilarity. The principle of formal hypothesis bisimilarity also comes up in a different setting in Larsen's work with Xinxin on the operational semantics of contexts [29, 30]. The *context systems* in [30] are easily seen to give rise to conditional transition systems, in the following sense: each transduction $C \xrightarrow{\mathbf{a}}_{\mathbf{b}} C'$ corresponds to a vector of conditional transitions $\Gamma \vdash t_i \xrightarrow{a_i} t'_i$ for $1 \leqslant i \leqslant |\mathbf{a}|$, where $\Gamma = \{x_j \xrightarrow{b_j} x'_j \mid 1 \leqslant j \leqslant |\mathbf{b}|\}$ with all x_j, x'_j distinct, such that $C(x_1, ..., x_{|\mathbf{b}|}) = \mathbf{t}$ and $C'(x'_1, ..., x'_{|\mathbf{b}|}) = \mathbf{t}'$. (Alternatively, one can interpret transductions as vectors of De Simone-format ruloids.) Context bisimulation as defined in [30] then precisely corresponds to $\sim^{\mathsf{fh}}$. An important characteristic is that the format of transductions prevents copies of "old" variables from the source term to remain in the target term of a transition (i.e., the t'_i may contain none of the x_j). Among other things, this makes it impossible to capture general recursion using contexts—the encoding presented in [10] does not seem to be adequate for our purpose. At the same time, this restriction obviates the re-use of old hypotheses; hence $\sim^{\mathsf{fh}}$ and hypothesis-preserving bisimilarity ($\sim^{\mathsf{hp}}$) coincide in this setting (see also Footnote 2 to Example 3.5).

A characterisation of open terms as *tiles* has been proposed by Gadducci and Montanari in [18]. The corresponding operational semantics unifies the principles of the contexts in [30] with the principles of *conditional rewriting* in [31]; [18] also defines the notion of $\sim^{\mathsf{fh}}$ in this setting. Since tiles seem to be strictly more general than our conditional transitions, it should also be possible to define the corresponding notion of $\sim^{\mathsf{hp}}$ as different from $\sim^{\mathsf{fh}}$.

The formulation of $\sim^{\mathsf{hp}}$ seems to be completely new. Although its definition is more complex than that of $\sim^{\mathsf{fh}}$, a strong argument in its favour is the alternative characterisation as substitutive bisimilarity (see Theorem 3.18). It would be interesting to compare the two relations on the basis of their equational theories; see also below, where we conjecture that $\sim^{\mathsf{ci}} = \sim^{\mathsf{hp}} = \sim^{\mathsf{fh}}$ under certain, rather weak, conditions.

Mobile and higher-order calculi. Bisimilarity, originally developed in the setting of transition systems and "classical" process algebra, has been extended to other fields as well. Among these are mobile calculi such as the π-calculus (cf. [34]) and higher-order calculi (cf. [43]). In mobile calculi, some of the issues concerning the instantiation of channel names are analogous to the issues studied in this paper regarding term variables. For instance, the *distinctions* between channel names used in [34] as indices to bisimilarity are comparable to our hypotheses over term variables used to index hp-bisimilarity; and the hyperbisimulation recently proposed for the *fusion* calculus (see [40, 44]) can be seen as a counterpart to sb-bisimilarity.

The connection is stronger in the case of higher order calculi. These depend crucially on term variables, which can be bound as a result of communication. In fact, input action prefix becomes a binder rather than an ordinary operator as in CCS. As a consequence, a naive extension of the usual notion for bisimilarity to this setting results in a relation that is not a congruence. Sangiorgi has extensively studied this problem in [42]; he shows that a major criterion for bisimulation to give rise to a congruence is a notion of substitutivity very much like the one underlying our substitution-closed bisimilarity ($\sim^{\mathsf{sb}}$). The difference between his context bisimulation and $\sim^{\mathsf{sb}}$ mainly seems due to the fact that [42] takes the closed-instance approach, defining bisimulation directly over closed terms only. A definition of *weak* context bisimulation whose formulation is still closer to $\sim^{\mathsf{sb}}$ is studied in [5].

Another field where bisimulation has gained a foothold is that of *functional calculus*; see, e.g., [24, 28, 41]. Again, the operational semantics and bisimilarity are defined over closed terms and extended to open terms using the closed-instance definition. Here, too, substitutivity is a necessary condition for bisimilarity to be a congruence.

It would seem that in both settings described above, the coarsest congruences within $\sim^{\mathsf{ci}}$ and $\sim^{\mathsf{sb}}$ (extended appropriately) coincide, or are at least closer than in the "classical" case. If so, then (given the coincidence of $\sim^{\mathsf{sb}}$ and $\sim^{\mathsf{hp}}$) it might be worthwhile to adapt the principle of conditional transitions to those formalisms. This is an area for further study.

Recursion Congruence. The issue of *recursion congruence* for bisimilarity has received scant attention in the literature, especially in the work on SOS formats.[4] One of the few results appears to be the congruence proof for the specific CCS signature in [33], which is based on the technique of *up-to* bisimulation (see also [36]). The same basic technique (albeit in a different formulation, explicitly based on co-induction) is used in the congruence proofs for the functional calculi mentioned above. We have shown in Section 4.2 that there are limitations to this technique which prevent its application as soon as there is look-ahead in the hypotheses of an operator rule. The hypothesis-based relations, on the other hand, give rise to a very different proof technique using *open approximations*, which does not suffer from these limitations.

[4] Note, however, that any implementation relation that generates a c.p.o. in which recursion yields least fixpoints is automatically a recursion (pre-)congruence; in particular, in a domain where equality coincides with bisimilarity, such as studied in, e.g., [1, 3], bisimilarity is bound to be a recursion congruence.

Recursion congruence is merely a special case of congruence with respect to a *binder* in the language. As far as we know, no general results exist about binder congruences. The first step to formulate (let alone prove) such a result would be to come up with a format for SOS rules that recognises the binding of term variables as a primitive concept. The only current work in this direction that we are aware of is Fokkink and Verhoef [17], who, however, investigate conservativity rather than congruence.

5.2. *Extensions*

The results of this paper raise several questions that might deserve looking into.

Difference between the bisimilarities. Examples 3.3 and 3.5, showing that ci-, hp-, and fh-bisimilarity are, in general, different relations, relied on an auxiliary operator introduced especially for this purpose. At present, we do not know precisely what aspects of this operator are really necessary to show up the difference, or if the relations are still different in, say, CCS without extensions. Formulated more generally, it is unclear under what circumstances, for instance what conditions on the operational rules, the various notions of bisimulation studied in this paper remain distinct. However, we have the following conjecture.

5.1. CONJECTURE. *If a conditional transition system does not contain a state with a finite number, greater than* 1, *of nonbisimilar closed instantiations, then* $\sim^{\mathsf{ci}}$ *and* $\sim^{\mathsf{fh}}$ (*and, hence, also* hp-*bisimilarity*) *coincide.*

The idea is that if a state has only a single closed instantiation (modulo the equivalence under consideration) then all bisimilarities studied in this paper coincide anyway (Theorem 3.8); whereas if it has an infinite number of nonequivalent instantiations, it should be possible to reconstruct its conditional transitions from the transitions of its instantiations.

For instance, Examples 3.3 and 3.5 crucially rely on the fact the open term $do_a(x)$ has only two nonbisimilar closed instantiations, viz. $do_a(\mathbf{0}) \sim^{\mathsf{ci}} \mathbf{0}$ and $do_a(a.\mathbf{0}) \sim^{\mathsf{ci}} a.\mathbf{0}$; all other instantiations are bisimilar to one of these two. On the other hand, no open term t in CCS, CSP, or ACP has this feature, since if a free variable $x \in \mathrm{var}(t)$ plays any role in t's behaviour at all, then at least the internal moves (τ-transitions) of a term u substituted for x propagate to τ-transitions of the entire term, $t[u/x]$, and hence, $t[u_1/x] \not\sim^{\mathsf{ci}} t[u_2/x]$ as soon as u_1 and u_2 are distinguishable by their τ-transitions, e.g., if $u_1 = \tau^i.\mathbf{0}$ and $u_2 = \tau^j.\mathbf{0}$ for $i \neq j$. Thus, we conjecture that in those languages, $\sim^{\mathsf{ci}} = \sim^{\mathsf{hp}} = \sim^{\mathsf{fh}}$.

Weak Bisimilarity. The principles exposed in this paper can be extended to weak bisimilarity relations without fundamental problems. However, the substitutivity of the corresponding relations can only be guaranteed if the states of a conditional transition system are sufficiently insensitive to internal moves of their variables. In fact, the results of Bloom [7], who studied SOS formats for which weak bisimilarity is a congruence, carry over to weak open term bisimilarity.

The above discussion regarding the distinction between the closed-instance and hypothesis-based notions of bisimilarity also applies to the case of weak bisimilarity. In

contrast to the situation for strong bisimilarity however, CSP, for instance, allows us to construct terms that are weakly ci-bisimilar but not weakly fh-bisimilar; e.g., in the version of [11]:

$$b \to STOP \,\Box\, b \to a \to STOP, \qquad b \to STOP \,\Box\, b \to a \to STOP \,\Box\, b \to (x \parallel (a \to STOP)),$$

where $\alpha \to \_$ is equivalent to $\alpha.\_$ in CCS, *STOP* is equivalent to **0** in CCS, $\_ \Box \_$ is equivalent to $\_ + \_$ in CCS, and $\_ \parallel \_$ denotes the synchronised execution of its operands, as formalised by the conditional transitions (where we have used the CCS notation for internal steps as τ-transitions),

$$x \xrightarrow{\tau} x' \vdash x \parallel y \xrightarrow{\tau} x' \parallel y$$

$$y \xrightarrow{\tau} y' \vdash x \parallel y \xrightarrow{\tau} x \parallel y'$$

$$x \xrightarrow{a} x',\ y \xrightarrow{a} y' \vdash x \parallel y \xrightarrow{a} x' \parallel y'.$$

The operative subterm in this example is $x \parallel (a \to STOP)$, which in fact plays precisely the same role as $do_a(x)$ in Examples 3.3 and 3.5. Analogous terms can be formulated in ACP and LOTOS.

Further themes. Without further comment, we list some more possible themes for future research:

- It would be interesting to study the equational theory of $\sim^{\mathsf{hp}}$, as well as decision algorithms, and compare the results with those for $\sim^{\mathsf{fh}}$, as (partially) reported in [2].
- It might be useful to apply the idea of conditional transitions to the functional and higher order process calculi mentioned above and to investigate if they provide additional insight in the various bisimilarity relations being proposed in that setting (see [24, 28, 42]).
- In line with the SOS formats defined for operators, which guarantee that (standard) bisimilarity is a congruence (see [14, 8, 25]), one could try to formulate a format for *binder rules* that guarantees, first, the completeness of the resulting (conditional) transition systems in the sense of Definition 2.16 and second, that bisimilarity is a binder congruence.

ACKNOWLEDGMENTS

I thank Rob van Glabbeek, Roberto Gorrieri and Frits Vaandrager for discussions on the subject of this paper, and Luca Aceto, Michael Baldamus, Robert De Simone, Ugo Montanari, and Davide Sangiorgi for helpful pointers.

Received January 1998; final manuscript received June 14, 1999

REFERENCES

1. Abramsky, S. (1991), A domain equation for bisimulation, *Inform. and Comput.* **92**, 161–218.
2. Aceto, L., Bloom, B., and Vaandrager, F. (1999), Checking open equations in GSOS systems, Draft.
3. Aceto, L., and Hennessy, M. C. B. (1992), Termination, deadlock, and divergence, *J. Assoc. Comput. Mach.* **39**(1), 147–187.
4. Baeten, J. C. M., and Weijland, W. P. (1990), "Process Algebra," Cambridge Univ. Press, Cambridge.
5. Baldamus, M., and Dingel, J. (1997), Modal characterization of weak bisimulation for higher-order processes, *in* "TAPSOFT '97: Theory and Practice of Software Development" (M. Bidoit and M. Dauchet, Eds.), Lecture Notes in Computer Science, Vol. 1214, pp. 285–296, Springer-Verlag, New York/Berlin.
6. Barendregt, H. P. (1984), "The Lambda Calculus," Studies in Logic and the Foundations of Mathematics, Vol. 103, 2nd ed., North-Holland, Amsterdam.
7. Bloom, B. (1995), Structural operational semantics for weak bisimulation, *Theoret. Comput. Sci.* **146**, 25–685.
8. Bloom, B., Istrail, S., and Meyer, A. R. (1995), Bisimulation can't be traced, *J. Assoc. Comput. Mach.* **42**(1), 232–268.
9. Bolognesi, T., and Brinksma, E. (1987), Introduction to the ISO specification language LOTOS, *Comput. Networks ISDN Systems* **14**, 25–59.
10. Brinksma, E. (1992), On the uniqueness of fixpoints modulo observation congruence, *in* "Concur '92" (W. R. Cleaveland, Ed.), Lect. Notes in Comput. Sci., Vol. 630, Springer-Verlag, New York/Berlin.
11. Brookes, S. D., Hoare, C. A. R., and Roscoe, A. W. (1984), A theory of communicating sequential processes, *J. Assoc. Comput. Mach.* **31**(3), 560–599.
12. Cardelli, L. (1997), Type systems, *in* "Handbook of Computer Science and Engineering" (A. B. Tucker, Ed.), Chap. 103, CRC Press, Boca Raton, FL.
13. Cleaveland, W. R. (Ed.) (1992), "Concur '92," Lecture Notes in Computer Science, Vol. 630, Springer-Verlag, New York/Berlin.
14. De Simone, R. (1985), Higher-level synchronising devices in Meije-SCCS, *Theoret. Comput. Sci.* **37**, 245–267.
15. Fokkink, W. (1996), On the completeness of the equations for the Kleene star in bisimulation, *in* "Algebraic Methodology and Software Technology," Lecture Notes in Computer Science, Vol. 1101, pp. 180–194. [Full report version: University of Utrecht, Faculteit Wijsbegeerte, Preprint 141]
16. Fokkink, W., and van Glabbeek, R. (1996), Ntyft/ntyxt rules reduce to ntree rules, *Inform. and Comput.* **126**, 1–10.
17. Fokkink, W., and Verhoef, C. (1998), A conservative look at operational semantics with variable binding, *Inform. and Comput.* **146**(1), 24–54.
18. Gadducci, F., and Montanari, U. (1996), Tiles, rewriting rules and CCS, *in* "Rewriting Logics and Applications, First International Workshop" (J. Meseguer, Ed.), Electronic Notes in Theoretical Computer Science, Vol. 4, pp. 1–19.
19. Girard, J.-Y., Lafont, Y., and Taylor, P. (1989), "Proofs and Types," Cambridge Tracts in Theoretical Computer Science, Vol. 7, Cambridge Univ. Press, Cambridge.
20. van Glabbeek, R. J. (1990), The linear time-branching time spectrum, *in* "Concur '90" (J. C. M. Baeten and J. W. Klop, Eds.), Lecture Notes in Computer Science, Vol. 458, pp. 278–297, Springer-Verlag, New York/Berlin.
21. van Glabbeek, R. J. (1993), A complete axiomatization for branching bisimulation congruence of finite-state behaviours, *in* "Mathematical Foundations of Computer Science 1993" (A. M. Borzyszkowski and S. Sokołowski, Eds.), Lecture Notes in Computer Science, Vol. 711, pp. 473–484, Springer-Verlag, New York/Berlin.

22. van Glabbeek, R. J. (1993), The linear time-branching time spectrum II: The semantics of sequential systems with silent moves, *in* "Concur '93" (E. Best, Ed.), Lecture Notes in Computer Science, Vol. 715, pp. 66–81, Springer-Verlag, New York/Berlin. [Extended abstract]
23. van Glabbeek, R. J., and Goltz, U. (1989), Equivalence notions for concurrent systems and refinement of actions, *in* "Mathematical Foundations of Computer Science 1989" (A. Kreczmar and G. Mirkowska, Eds.), Lecture Notes in Computer Science, Vol. 379, pp. 237–248, Springer-Verlag, New York/Berlin.
24. Gordon, A. D. (1995), Bisimilarity as a theory of functional programming, *Electron. Notes in Theoret. Comput. Sci.* **1**. [Proc. of 11th Conference on Mathematical Foundations of Programming Semantics]
25. Groote, J. F., and Vaandrager, F. W. (1992), Structured operational semantics and bisimulation as a congruence, *Inform. and Comput.* **100**, 202–260.
26. Hennessy, M. C. B., and Lin, H. (1995), Symbolic bisimulations, *Theoret. Comput. Sci.* **138**(2), 353–389.
27. Hoare, C. A. R. (1985), "Communicating Sequential Processes," Prentice–Hall, Englewood Cliffs, NJ.
28. Howe, D. J. (1996), Proving congruences of bisimulation in functional programming languages, *Inform. and Comput.* **124**, 103–112.
29. Larsen, K. G. (1990), Compositional theories based on an operational semantics of contexts, *in* "Stepwise Refinement of Distributed Systems—Models, Formalisms, Correctness" (J. W. de Bakker, W.-P. de Roever, and G. Rozenberg, Eds.), Lecture Notes in Computer Science, Vol. 430, pp. 487–518, Springer-Verlag, New York/Berlin.
30. Larsen, K. G., and Xinxin, L. (1991), Compositionality through an operational semantics of contexts, *J. Logic Comput.* **1**(6), 761–795.
31. Meseguer, J. (1992), Conditional rewriting logic as a unified model of concurrency, *Theoret. Comput. Sci.* **96**, 73–155.
32. Milner, R. (1984), A complete inference system for a class of regular behaviours, *J. Comput. Syst. Sci.* **28**, 439–466.
33. Milner, R. (1989), "Communication and Concurrency," Prentice–Hall, Englewood Cliffs, NJ.
34. Milner, R., Parrow, J., and Walker, D. (1992), A calculus of mobile processes, I and II, *Inform. and Comput.* **100**, 1–77.
35. Milner, R., and Sangiorgi, D. (1992), Barbed bisimulation, *in* "Automata, Languages and Programming" (W. Kuich, Ed.), Lecture Notes in Computer Science, Vol. 623, pp. 685–695, Springer-Verlag, New York/Berlin.
36. Milner, R., and Sangiorgi, D. (1992), The problem of "weak bisimulation up to," *in* "Concur '92" (W. R. Cleaveland, Ed.), Lect. Notes in Comput. Sci., Vol. 630, Springer-Verlag, New York/Berlin.
37. Mitchell, J. C. (1990), Type systems for programming languages, *in* "Handbook of Theoretical Computer Science" (J. van Leeuwen, Ed.), Formal Models and Semantics, Vol. B, Chap. 8, pp. 365–458, Elsevier, Amsterdam.
38. Montanari, U., Pistore, M., and Yankelevich, D. (1996), Efficient minimization up to location equivalence, "Programming Languages and Systems—ESOP '96" (H. R. Nielson, Ed.), pp. 265–279, Springer-Verlag, New York/Berlin.
39. Mukund, M., and Nielsen, M. (1992), CCS, locations and asynchronous transition systems, *in* "Foundations of Software Technology and Theoretical Computer Science" (R. Shyamasundar, Ed.), Lecture Notes in Computer Science, Vol. 652, pp. 328–341, Springer-Verlag, Berlin. [Report version: PB–395, Computer Science Department, Aarhus University, May 1992]
40. Parrow, J., and Victor, B. (1998), The fusion calculus: Expressiveness and symmetry in mobile processes, *in* "Symposium on Logic in Computer Science," Comput. Soc. Press, New York.
41. Sands, D. (1997), From SOS rules to proof principles: An operational metatheory for functional languages, *in* "24TH ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages," ACM, New York.

42. Sangiorgi, D. (1996), Bisimulation for higher-order calculi, *Inform. and Comput.* **131**(2), 141–178.

43. Thomsen, B. (1995), A theory of higher order communicating systems, *Inform. and Comput.* **116**, 38–57.

44. Victor, B. (1998), "The Fusion Calculus: Expressiveness and Symmetry in Mobile Processes," Ph.D. thesis, Uppsala University.