Contextual equivalence for higher-order π-calculus revisited

Alan Jeffrey
CTI, DePaul University
ajeffrey@cs.depaul.edu

Julian Rathke*
COGS, University of Sussex
julianr@cogs.susx.ac.uk

October 2002

Abstract

The higher-order π-calculus is an extension of the π-calculus to allow communication of abstractions of processes rather than names alone. It has been studied intensively by Sangiorgi in his thesis where a characterisation of a contextual equivalence for higher-order π-calculus is provided using labelled transition systems and *nor a* bisimulations. Unfortunately the proof technique used there requires a restriction of the language to only allow finite types.

We revisit this calculus and offer an alternative presentation of the labelled transition system and a novel proof technique which allows us to provide a fully abstract characterisation of contextual equivalence using labelled transitions and bisimulations for higher-order π-calculus with recursive types also.

1 Introduction

It is evident that there is growing interest in the study of mobile code in process languages [2, 1, 7, 12]. It is also clear that there is some relationship between the use of higher-order features and mobility. Indeed, code mobility can be expressed as communication of process abstractions. For this reason then it is important for us to develop a clear understanding of the use of higher-order features in process languages.

Work towards this began several years ago with various proposals for higher-order versions of known calculi [11, 3], including the higher-order π-calculus or HOπ [8]. This calculus was studied intensively by Sangiorgi and one of his achievements was to provide a translation of the higher-order language which supports code mobility, to a first-order π

In this paper we present an alternative description of labelled transition systems and normal bisimulations for HOπ

,	::=		**Value Types**
		·	Unit type
		ch[ ]	Channel type
		→ ◇	Abstraction type
		Z	Type variable
		rec Z.	Recursive type
,	::=		**Terms**
		$v \cdot w$	Application
		$v(x:$)	Input
		$v\langle w \rangle$	Output
		if $v = w$ then else	Matching
		ν($a:$) . ( )	Name creation
		‖	Concurrency
		$*$	Repetition
		0	Termination
v, w	::=		**Values**
		·	Unit value
		a	Channel name
		x	Variable
		($x:$)	Abstractions

Figure 1: The Syntax

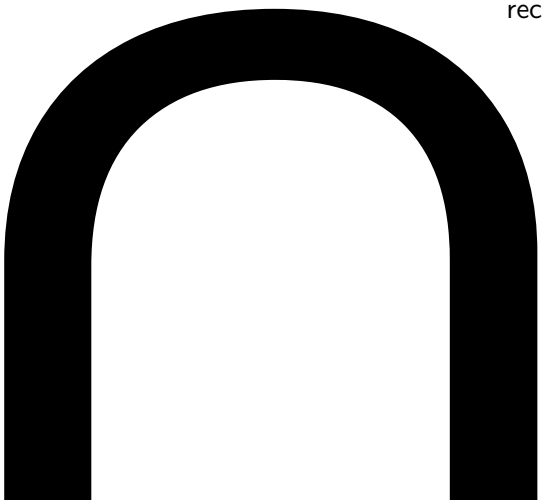
following axioms hold

(comm)	$a\langle v \rangle$ ‖ $a(x)$	⟶	‖ (x) · v	
(β – redn)	(x) · v	⟶	$[v/x]$	
(cond—tt)	if $a = a$ then else	⟶		
(cond—ff)	if $a = b$ then else	⟶		($a \neq b$)

In a standard notation we write $\Longrightarrow$ to denote the reflexive, transitive closure of -⟶ .

We introduce a simple type system for the language which comprises types for channels and abstractions. We also allow recursive types of the form rec Z. where rec . forms a binder and Z is drawn from a countably infinite supply of type variables. We must insist that for any rec Z. that Z does not appear unguarded in , that is to say that any free occurrence of Z lies within a subexpression of of the form ch[ ] or → ◇. To allow us to infer recursive types for terms we make use of type isomorphism. We define this by letting $\sim_{so}$ be the least congruence on types which includes

rec Z.



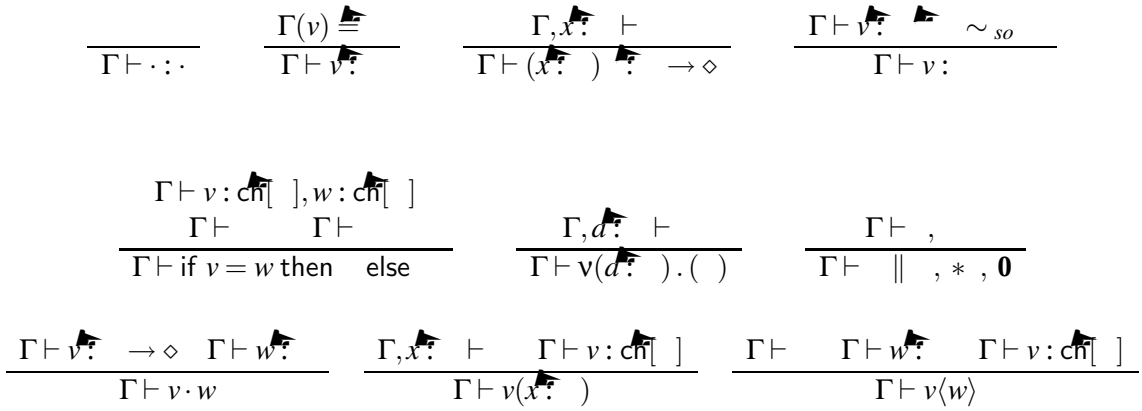


Figure 2: The Typing Rules

well-typed process, , closed if it can be typed as $\Delta \vdash$ for some closed Δ. It is easily shown that subject reduction holds for closed terms for the reduction relation and type inference system given.

2.1 Contextual equivalence

We will now define an appropriate notion of behavioural equivalence based on contexts and barbs.

Contexts are defined by extending the syntax of processes by allowing typed holes $[\ _\Gamma\ ]$ in terms. The type inference system is extended to contexts by using the rule

$$\frac{}{\Gamma, \Gamma' \vdash [\ \cdot_\Gamma\ ]}$$

We write $C[]$ to denote contexts with at most one hole and $C[\ ]$ for the term which results from substituting into the hole.

For any given channel name a such that $\Delta \vdash a : \mathsf{ch}[\cdot]$ we write $\Delta \models \Downarrow a$ if there exists some $'$, $''$ such that

Reduction closure: A type-indexed relation is reduction-closed whenever $\Delta \models$ and $\longrightarrow$ $'$ implies there exists some $'$ such that $\Longrightarrow$ $'$ and $\Delta \models$ $'$ $'$.

Contextuality: A type-indexed relation is contextual whenever $\Gamma' \models$ ^{o} and $\Gamma \vdash C[\cdot_{\Gamma'}]$ implies $\Gamma \models C[\ ]$ ^{o} $C[\ ]$.

Barb preservation: A type-indexed relation is barb-preserving if $\Delta \models$ and $\Delta \models \Downarrow a$ implies $\Delta \models \Downarrow a$.

Definition 2.1 (Contextual equivalence) Let $\cong$ be the open extension of the largest type-indexed relation which is symmetric, reduction-closed, contextual and barb-preserving. □

For technical convenience it will be useful to work with a lighter definition of contextuality. We say that a relation is $\|$-contextual if it is preserved by all contexts of the form $[\ _{\Gamma}\ ] \|$ and we let $\cong_p$ denote the open extension of the largest typed relation over processes which is symmetric, $\|$-contextual, reduction-closed and barb-preserving. The following lemma demonstrates that this lighter definition is sufficient.

Lemma 2.2 (Context lemma) $\Gamma \models \cong$ *and on y* $\Gamma \models \cong_p$

Proof: The 'only if' direction is immediate. For the converse it is sufficient to show that $\cong_p$ is preserved by each process operator of HOπ. The majority of these are straightforward so we only show the case for input prefixing. Suppose then (without loss of generality) that $\Delta, a : \mathsf{ch}[\ ], x : \models \cong_p$. We [illegible] show that

3 Labelled transitions

We will use a labelled transition system to characterize $\cong$ over higher-order π-calculus terms. The style of the labelled transition system differs a little from previous transition systems offered for HOπ. Most notably, the nodes of the transition system are described using an augmented syntax rather than process terms alone. Specifically, for each drawn from a countable set of names disjoint from $\mathcal{N}$ and , we introduce two new operators:

$$\tau \qquad \text{and} \qquad \langle \ \Leftarrow v \rangle$$

with the intuitive reading that τ is an indirect reference to an abstraction and $\langle \ \Leftarrow v \rangle$ stores the abstraction to which refers so that access to v is provided through interaction with . The augmented syntax for nodes is given the grammar of configurations C obtained by extending Figure 1 with:

$$\begin{array}{rcl} v & ::= & \ldots \text{(as Figure 1)} \ldots \mid \tau \\ C & ::= & \mid \langle \ \Leftarrow v \rangle \mid \nu d : \ .(C) \mid C \parallel C \end{array}$$

We impose a syntactic restriction on the augmented syntax so that in any configuration C for any given then $\langle \ \Leftarrow v \rangle$ appears at most once in C

C

$$\frac{\Delta \vdash v : \quad \text{a base type}}{(\Delta\,;\,\Theta \vdash a(x : \quad)\;) \xrightarrow{a\langle v\rangle?} (\Delta\,;\,\Theta \vdash (x : \quad)\;\cdot v)}$$

$$\frac{\Theta(\;) \equiv \qquad \Delta \vdash w : \quad \text{a base type}}{(\Delta\,;\,\Theta \vdash \langle\;\Leftarrow v\rangle) \xrightarrow{\langle w\rangle?} (\Delta\,;\,\Theta \vdash v \cdot w \parallel \langle\;\Leftarrow v\rangle)}$$

$$\frac{\Delta \vdash v : \quad \text{a base type}}{(\Delta\,;\,\Theta \vdash a\langle v\rangle\;) \xrightarrow{a\langle v\rangle!} (\Delta\,;\,\Theta \vdash\;)}$$

$$\frac{\Theta(\;) \equiv \quad \text{a base type}}{(\Delta\,;\,\Theta \vdash \tau\;\cdot v) \xrightarrow{\langle v\rangle!} (\Delta\,;\,\Theta \vdash \mathbf{0})}$$

Figure 5: Basic first-order labelled transition rules

the reference graph of $\langle\;\Leftarrow v\rangle \parallel C$ has the node removed and any edges such that

$$' \mapsto \quad \mapsto$$

for $'$, $\neq$, are replaced with an edge

$$' \mapsto$$

all other edges involving are removed. So if node is involved in a cyci70nf3.099610Td7phgnod(occu830Td[(are

So either, $\alpha_1 = \alpha_2$ in which case $C_1 \equiv C_2$ or $\alpha_1 \neq \alpha_2$ and

$$C_1' \equiv C_3' \parallel \langle\, \alpha_2 \Leftarrow v_2 \rangle \qquad \text{and} \qquad C_2' \equiv C_3' \parallel \langle\, \alpha_1 \Leftarrow v_1 \rangle$$

We notice that

$$\begin{array}{lrcl}
 & C_1 & \equiv & C_1'[v_1/\tau_{\alpha_1}] \\
 & & \equiv & (C_3' \parallel \langle\, \alpha_2 \Leftarrow v_2\rangle)[v_1/\tau_{\alpha_1}] \\
 & & \equiv & C_3'[v_1/\tau_{\alpha_1}] \parallel \langle\, \alpha_2 \Leftarrow v_2[v_1/\tau_{\alpha_1}]\rangle \\
(\text{acyclicity implies } \tau_{\alpha_2} \notin v_2[v_1/\tau_{\alpha_1}]) & & \twoheadrightarrow & C_3'[v_1/\tau_{\alpha_1}][v_2[v_1/\tau_{\alpha_1}]/\tau_{\alpha_2}] \\
 & & \equiv & C_3'[v_1[v_2[v_1/\tau_{\alpha_1}]/\tau_{\alpha_2}]/\tau_{\alpha_1}, v_2[v_1/\tau_{\alpha_1}]/\tau_{\alpha_2}] \\
(\text{acyclicity}) & & \equiv & C_3'[v_1[v_2/\tau_{\alpha_2}]/\tau_{\alpha_1}, v_2[v_1/\tau_{\alpha_1}]/\tau_{\alpha_2}] \\
(\text{def}) & & \equiv & C_3
\end{array}$$

By a symmetric argument we see that $C_2 \twoheadrightarrow C_3'[v_2[v_1/\tau_{\alpha_1}]/\tau_{\alpha_2}, v_1[v_2/\tau_{\alpha_2}]/\tau_{\alpha_1}]$ and, by definition, this is just C_3 so we have $C_2 \twoheadrightarrow C_3$. Thus $\twoheadrightarrow$ is strongly confluent for acyclic terms and hence $\langle\langle\cdot\rangle\rangle$ is well-defined. □

Lemma 4.2 (Composition/Decomposition) *For* $\Delta\,;\,\Theta \vdash C, D$

If $\langle\langle C \parallel D\rangle\rangle \equiv E$ *and*

$$(\Delta\,;\,\Theta \vdash C) \xrightarrow{\alpha} (\Delta,\Delta'\,;\,\Theta,\Theta' \vdash C') \qquad \textit{and} \qquad (\Delta\,;\,\Theta \vdash D) \xrightarrow{\bar{\alpha}} (\Delta,\Delta'\,;\,\Theta,\Theta' \vdash D')$$

then there exists a E' *such that* $E \twoheadrightarrow E'$

It is easy to see that $\langle\langle C \parallel D\rangle\rangle \longrightarrow \langle\langle \nu\Delta', \Delta'' . ((x : \blacktriangleright \quad) \cdot v \parallel C'' \parallel \quad \parallel D'')\rangle\rangle$ let us call the target of this reduction E'. We simply need to check

$$\begin{array}{rcl} E' & \equiv & \langle\langle \nu\Delta', \Delta'' . ((x : \blacktriangleright \quad) \cdot v \parallel C'' \parallel \quad \parallel D'')\rangle\rangle \\ (\tau \notin v) & \equiv & \langle\langle \nu\Delta' . ((x : \blacktriangleright \quad) \cdot \tau \parallel C'') \parallel \nu\Delta'' . (\langle \Leftarrow v\rangle \parallel \quad \parallel D'')\rangle\rangle \\ & \equiv & \langle\langle C' \parallel D'\rangle\rangle \end{array}$$

Case: $\Delta\,;\,\Theta \vdash C \xrightarrow{\nu\,.\,\langle\tau\rangle?} \Delta\,;\,\Theta, \blacktriangleright : \quad \vdash C'$ and $\Delta\,;\,\Theta \vdash D \xrightarrow{\nu\,.\,\langle\tau\rangle!} \Delta\,;\,\Theta, \blacktriangleright : \quad \vdash D'$. Again, by inspection we see that

- $C \equiv \nu\Delta' . (\langle \Leftarrow v\rangle \parallel C'')$
- $C' \equiv \nu\Delta' . (v \cdot \tau \parallel \langle \Leftarrow v\rangle \parallel C'')$
- $D \equiv \nu\Delta'' . (\tau \cdot w \parallel D'')$
- $D' \equiv \nu\Delta'' . (\langle \Leftarrow w\rangle \parallel D'')$

Note that the previous proposition tells us that $(C \parallel D)$ must be acyclic — in particular, $\tau \notin v$. Here we see that

$$\begin{array}{rcl} \langle\langle C \parallel D\rangle\rangle & \equiv & \langle\langle \nu\Delta', \Delta'' . (\langle \Leftarrow v\rangle \parallel C'' \parallel \tau \cdot w \parallel D'')\rangle\rangle \\ (\tau \notin v) & \equiv & \langle\langle \nu\Delta', \Delta'' . (\langle \Leftarrow v\rangle \parallel C'' \parallel v \cdot w \parallel D'')\rangle\rangle \\ (\tau \notin v, w, C'', D'') & \equiv & \langle\langle \nu\Delta', \Delta'' . (\langle \Leftarrow v\rangle \parallel C'' \parallel v \cdot \tau \parallel \langle \Leftarrow w\rangle \parallel D'')\rangle\rangle \\ & \equiv & \langle\langle C' \parallel D'\rangle\rangle \end{array}$$

Case:

Corollary 4.5 (Soundness) *For a ter s , of HOπ*

$$\Gamma \models \approx^{o} \qquad p\ es \quad \Gamma \models \cong$$

Proof: Follows from the previous theorem and Lemma 2.2. □

5 Completeness of bisimilarity for contextual equivalence

The interactions described by the labelled transition system are not obviously derived by genuine contextual observations in HOπ because of the use of the extra syntax for indirect references. In order to show completeness of our bisimilarity for contextual equivalence we must demonstrate that the indirect references are in fact definable as terms of the language proper. Following Sangiorgi [10], we implement the implicit protocol outlined by the indirect references by using the following translation of the augmented terms into HOπ:

$$\begin{array}{rcll}
[\![\ _1 : \ _1, \ldots, \ _n : \ _n]\!] & = & \ _1 : \mathsf{ch}[\ _1], \ldots, \ _n : \mathsf{ch}[\ _n] & \\
[\![\Gamma \,;\, \Theta \vdash C]\!] & = & \Gamma, [\![\Theta]\!] \vdash [\![C]\!]_\Theta & \\
[\![\tau\ ]\!]_\Theta & = & (x : \ )\ \langle x \rangle \mathbf{0} & \text{if } \Theta(\ ) = \\
[\![\langle\ \Leftarrow v \rangle]\!]_\Theta & = & *\ [\![v]\!]_\Theta &
\end{array}$$

The translation acts homomorphically on all other terms. We abuse notation here by using identifiers as channel names in the translation. It is evident that this translation is well-defined in the sense that the translation of well-typed augmented terms are indeed well-typed terms of HOπ.

We would now like to prove a correspondence between reductions from the terms of the augmented syntax and reductions between their translations. However, we note that in translating a term containing both $\langle\ \Leftarrow v\rangle$ and τ we provide matching input and output prefixes, which, in HOπ may create a communication which was not possible in the source term. This turns out not to be of particular concern to us though as we see that if we starting with terms of HOπ, then terms reachable by transitions are *ba anced* in the following sense: we call a term C of the augmented language *ba anced* if for each then C contains at most one of τ

By inspecting the translation $[\![\cdot]\!]$ and using the fact that C is balanced we see that

$$[\![C]\!] \xrightarrow{\mathsf{h}} \longrightarrow \qquad \text{implies} \qquad [\![C]\!] \longrightarrow \xrightarrow{\mathsf{h}}$$

thus we may assume that the first reduction in (†) above is not of the form $\xrightarrow{\mathsf{h}}$. This means that $[\![C]\!] \longrightarrow [\![C']\!] \Longrightarrow$ for some C' such that $C \longrightarrow C'$. It is clear that C' is also balanced so we mayapply the inducti

$[\![$

to obtain a D such that $C' \Longrightarrow D'$ and $[\![C']\!]$

C

$$\longrightarrow C' \Longrightarrow D \qquad \text{and} \qquad [\![C]\!] \longrightarrow [\![C']\!] \Longrightarrow [\![D]\!] \xrightarrow{\mathsf{h}}{}^{*}$$

as required. □

Proposition 5.2 *For eac* α, Δ *and res c anne s* δ, δ' *o appropr ate type ven by* α *and* Δ *t ere ex sts a process* ${}^{\Delta}_{\alpha}$ *de ned n F ure n HO*π *suc t at*

$$\Delta\,;\,\Theta \vdash C \xrightarrow{\alpha} \Delta,\Delta'\,;\,\Theta,\Theta' \vdash C'$$

t en

$$\Delta, [\![\Theta,\Theta']\!], \delta : \mathsf{ch}[\ {}_{0}], \delta' : \mathsf{ch}[\cdot] \vdash {}^{\Delta,[\![\Theta]\!]}_{\alpha}$$

and oreover or ba anced D

$$(\Delta\,;\,\Theta \vdash D) \stackrel{\alpha}{\Longrightarrow} (\Delta,\Delta'\,;\,\Theta,\Theta' \vdash D')$$

and on y $\Delta\,;\,\Theta \vdash D$ *and*

$${}^{\Delta,[\![\Theta]\!]}_{\alpha} \parallel [\![D]\!]_{\Theta} \Longrightarrow \nu\Delta'\,.\,(\delta\langle\Delta'\rangle \parallel \ ) \qquad w\ t \qquad [\![D']\!]_{\Theta,\Theta'} \xrightarrow{\mathsf{h}}{}^{*} \ .$$

Proof: It is straightforward to check that $\Delta, [\![\Theta,\Theta']\!], \delta : \mathsf{ch}[\ {}_{0}], \delta' : \mathsf{ch}[\cdot] \vdash {}^{\Delta}_{\alpha}$ whenever

$$\Delta\,;\,\Theta \vdash C \xrightarrow{\alpha} \Delta,\Delta'\,;\,\Theta,\Theta' \vdash C'.$$

For the remainder, to show the 'only if' direction we use Lemma 5.1 Part 1 to reduce our obligation to the case of a single transition $\xrightarrow{\alpha}$, and we must consider each label α. By way of example we show the case for $\alpha = \tau\langle v\rangle$ (…

$(\Delta\,;\,\Theta \vdash$

$$\xrightarrow{\alpha} (\Delta\,;\,\Theta,\ :\ \ \vdash D').$$

then we know that

$$D \equiv \nu\Delta''\,.\,(\tau\ \cdot v \parallel D'')$$

and

$$D' \equiv \nu\Delta''\,.\,(\langle\ \Leftarrow v\rangle \parallel D'').$$

We see that for $\sim_{so} \ \rightarrow \diamond$

$$\begin{array}{rcl} {}^{\Delta,[\![\Theta]\!]}_{\alpha} \parallel [\![D]\!]_{\Theta} & \equiv & (x:\ )(*\ (y:\ )x\cdot y \parallel (\delta\langle\rangle \oplus \delta'\langle\rangle)) \parallel \nu\Delta''\,.\,(((z:\ )\ \langle z\rangle\mathbf{0})\cdot[\![v]\!]_{\Theta} \parallel [\![D'']\!]_{\Theta}) \\ & \Longrightarrow & (\delta\langle\rangle \oplus \delta'\langle\rangle) \parallel \nu\Delta''\,.\,(*\ (y:\ )[\![v]\!]_{\Theta}\cdot y \parallel [\![D'']\!]_{\Theta}) \\ & \Longrightarrow & \delta\langle\rangle \parallel [\![D']\!]_{\Theta,\ :} \end{array}$$

as required.

For the converse direction we suppose that

$$\overset{\Delta,[\![\Theta]\!]}{\alpha} \parallel [\![D]\!]_\Theta \Longrightarrow \nu\Delta' \,.\, (\delta\langle\Delta'\rangle \parallel \quad)$$

Again, we must perform a case analysis on α. We show the case in which α is $\nu \;.\; \langle\tau\rangle$? (the other cases can be treated similarly). We know Δ' is empty so $\overset{\Delta,[\![\Theta]\!]}{\alpha} \parallel [\![D]\!]_\Theta \Longrightarrow \delta\langle\rangle \parallel \quad$. Note that $\overset{\Delta,[\![\Theta]\!]}{\alpha}$ has no reductions of its own and can only interact with $[\![D]\!]_\Theta$ so we can detail the assumed reductions as

$$\overset{\Delta,[\![\Theta]\!]}{\alpha} \parallel [\![D]\!]_\Theta \Longrightarrow \quad \overset{\Delta,[\![\Theta]\!]}{\alpha} \parallel \quad_0 \longrightarrow (\delta\langle\rangle \oplus \delta'\langle\rangle) \parallel \quad_1 \Longrightarrow \delta$$

$$
\begin{array}{lcll}
\overset{\Delta}{d\langle v\rangle?} & = & d\langle v\rangle(\delta\langle\rangle \oplus \delta'\langle\rangle) & \\
\overset{\Delta}{d\langle v\rangle!} & = & d(x:\quad)\text{if } x = v \text{ then } (\delta\langle\rangle \oplus \delta'\langle\rangle) \text{ else } \mathbf{0} & \text{where } \Delta(d) = \mathsf{ch}[\quad] \\
\overset{\Delta}{\nu b.d\langle b\rangle?} & = & \nu b:\quad .(d\langle b\rangle(\delta\langle b\rangle \oplus \delta'\langle\rangle)) & \text{where } \Delta(d) = \mathsf{ch}[\quad] \\
\overset{\Delta}{\nu b.d\langle b\rangle!} & = & d(x:\quad)\text{if } x \notin \Delta \text{ then } (\delta\langle x\rangle \oplus \delta'\langle\rangle) \text{ else } \mathbf{0} & \text{where } \Delta(d) = \mathsf{ch}[\quad] \\
\overset{\Delta}{\nu .d\langle \tau\rangle?} & = & d\langle(x:\quad)\ \langle x\rangle\mathbf{0}\rangle(\delta\langle\rangle \oplus \delta'\langle\rangle) & \text{where } \Delta(d) = \mathsf{ch}[\quad] \text{ and } \quad \sim_{so} \quad \to\diamond \\
\overset{\Delta}{\nu .d\langle \tau\rangle!} & = & d(x:\quad)(*\,(y:\quad)x\cdot y \parallel (\delta\langle\rangle \oplus \delta'\langle\rangle)) & \text{where } \Delta(d) = \mathsf{ch}[\quad] \text{ and } \quad \sim_{so} \quad \to\diamond
\end{array}
$$

($\oplus$ represents an encoding of internal choice in HOπ)

Figure 6: Testing processes for labelled transitions

Theorem 5.4 (Completeness) *For a c osed ter s , HOπ*

$$\Delta \models \quad \cong_p \qquad p \ \ es \qquad \Delta \models \quad \approx$$

Proof: We define over terms of the augmented language to be

$$\Delta\,;\,\Theta \models C \quad D \qquad \text{iff} \qquad \Delta, [\![\Theta]\!] \models [\![C]\!]_\Theta \cong_p [\![D]\!]_\Theta$$

and show that is a bisimulation. Take $\Delta\,;\,\Theta \models C \quad D$ and suppose that

$$(\Delta\,;\,\Theta \vdash C) \xrightarrow{\alpha} (\Delta, \Delta'\,;\,\Theta, \Theta' \vdash C').$$

We know from Proposition 5.2 that

$$\Delta, [\![\Theta, \Theta']\!], \delta : \mathsf{ch}[\ _0], \delta' : \mathsf{ch}[\cdot] \vdash \ \overset{\Delta,[\![\Theta]\!]}{\alpha}$$

and that

$$\overset{\Delta,[\![\Theta]\!]}{\alpha} \parallel [\![C]\!]_\Theta \Longrightarrow \nu\Delta'.(\delta\langle\Delta'\rangle \parallel \quad)$$

with $[\![C']\!]_{\Theta,\Theta'} \xrightarrow{\mathsf{h}^*}$. We know that

$$\Delta, [\![\Theta]\!] \models [\![C]\!]_\Theta \cong_p [\![D]\!]_\Theta$$

by the definition of

and we now must show that $\Delta, \Delta' \; ; \; \Theta, \Theta$

[7] J. Riely and M. Hennessy. A typed language for distributed mobile processes. In *roc O L.* ACM Press, 1998.

[8] D. Sangiorgi. *Express n Mob ty n rocess A ebras F rst-Order and H er-Order arad s*. PhD thesis, University of Edinburgh, 1993.

[9] D. Sangiorgi. Bisimulation for higher-order process calculi. *In or at on and Co putat on*, 131(2):141–178, 1996.

[10] Davide Sangiorgi and David Walker. *e p -ca cu us A eory of ob e processes*. Cambridge University Press, 2001.

[11] B. Thomsen. *Ca cu for H er-Order Co un cat n yste s*. PhD thesis, University of London, 1990.

[12] Jan Vitek and Giuseppe Castagna. Seal: A framework for secure mobile computations. In *Internet ro ra n Lan ua es*, volume 1686 of *LNC* . Springer-Verlag, 1999.