Complete Inference Systems for Weak Bisimulation Equivalences in the π-Calculus*

Huimin Lin
Laboratory for Computer Science, Institute of Software,
Chinese Academy of Sciences, P.O.Box 8718, Beijing 100080, China
Email: lhm@ios.ac.cn

Abstract

Proof systems for weak bisimulation equivalences in the π-calculus are presented, and their soundness and completeness are shown. Two versions of π-calculus are investigated, one without and the other with the *mismatch* operator. For each version of the calculus proof systems for both *late* and *early* weak bisimulation equivalences are studied. Thus there are four proof systems in all. These inference systems are related in a natural way: the inference system for early equivalence is obtained from the one for late equivalence by replacing the inference rule for input prefix, while the inference system for the version of π-calculus with mismatch is obtained by adding a single inference rule for mismatch to the one for the version without it. The proofs of the completeness results rely on the notion of *symbolic bisimulation*.

1 Introduction

This paper aims to provide complete proof systems for weak bisimulation equivalences in the π-calculus. *Bisimulation equivalences* are among the most well-developed semantic theories for process calculi. The π-calculus inherits from *CCS* the invisible τ action, so the usual distinction between *strong* and *weak* bisimulations applies: the strong version gives the τ action the same weight as any other normal, or visible, actions, while the weak version ignores τ moves. As τ action is used to represent internal communication and as such is not observable, the weak equivalence is what used for practical applications. On the other hand, for the π-calculus another distinction can be made, namely *late* and *early* bisimulations [11], according to when input variables are instantiated. Thus for the π-calculus we have at least four bisimulation equivalences to study.

*Supported by the National Natural Science Foundation of China. An extended abstract appeared in the proceedings of TAPSOFT'95, Lecture Notes in Computer Science 915.

The original π-calculus paper [10] only gave an axiomatisation for the *late* version of *strong* ground bisimulation. Subsequently, efforts have been made to formulate complete proof systems for other equivalences for this calculus: [12, 2, 7] for both early and late strong bisimulation congruences, [4, 1] for *testing* equivalence, and [13] for (strong) *open bisimulation*. It has been widely conjectured that axiomatisation for weak bisimulations can be obtained by adding Milner's τ-laws to proof systems for strong equivalences [10, 12, 2].

In this paper we shall verify this conjecture by presenting complete proof systems for both late and early weak observation congruences. These proof systems are extensions of those for late and early strong bisimulations, as proposed in [7], with the three τ-laws. They consist of a set of inference rules together with some standard equations. The judgments of the inference systems are of the form

$$C \rhd t = u$$

where t, u are π-calculus terms and C, called a *condition*, is a set of equality or inequality tests between names. It is important to note that C is *not* a construction in the π-calculus; it is a construction only used in the meta language in order to reason about bisimulation equivalences for the calculus. To give a taste of the proof system, here is the inference rule for match:

$$\text{MATCH} \ \frac{C \cup \{x = y\} \rhd t = u \quad C \cup \{x \neq y\} \rhd \mathbf{0} = u}{C \rhd [x = y]t = u}$$

It involves a case analysis: if we can establish $t = u$ under the condition $x = y$, and $\mathbf{0} = u$ under the condition $x \neq y$, then we can conclude $[x = y]t = u$. Here it can be seen clearly how the inequalities *in the meta-language* help to characterise constructions *in the calculus*. Indeed this is the only rule needed for the match operator. For the extension of the π-calculus with the *mismatch* operator, only a single inference rule, which is dual to the above rule for match, needs to be added to the proof systems.

As in *CCS*, weak bisimulations in the π-calculus are defined in terms of weak transition relations that abstract away τ-moves. But care must be taken in dealing with late equivalence: in this case input moves should not be allowed to absorb the τ-moves *after* them, for otherwise the resulted relation would not be transitive. Also as in *CCS*, weak bisimulations are not preserved by the choice operator, so we have to work with observation congruences.

The proofs of the completeness results for these inference systems rely on the notion of *symbolic bisimulations* [5, 6]. In [10] a general notion of bisimulation, $\sim^D$, called *distinction indexed bisimulation*, is introduced, where the index D is a set of *inequations* on names such that $t \sim^D u$ if and only if $t\sigma \stackrel{.}{\sim} u\sigma$ for every substitution σ satisfying D. In the setting of the π-calculus, symbolic bisimulation is a mild generalisation of distinction indexed bisimulation: it is indexed by *conditions* consisting of name inequations *as well as name equations*. This generalisation makes it possible to give a direct definition of symbolic bisimulation (in terms of *symbolic transitional semantics*) instead of as substitution closure on top of ground bisimulation. Such a direct definition facilitates the completeness proofs for the inference systems.

An advantage of working with π-calculus over general message-passing calculi is that the language for the message domain is known: it is simply a set of *channel names* upon which the only allowed operations are tests for equality and inequality. To define the notion of symbolic bisimulation for the π-calculus, we first develop a simple theory of *conditions* which are sets of equalities and inequalities over names, and show that the implication relation between conditions is decidable. A *maximally consistent* condition is a *complete*, or *saturated*, condition, in the sense that adding to it anything not implied by it will result in inconsistency. Maximally consistent conditions characterise substitutions up-to isomorphism, and helps to simplify the proofs of the completeness results considerably.

The rest of this paper is organised as follows: The calculus and its semantics are introduced in the next section. The inference system is presented in Section 3, along with the completeness proof. Section 4 demonstrates how the theory developed for the late equivalence in the previous sections can be carried over to the early case. Section 5 discusses extensions to the calculus. The paper is concluded with Section 6, where related work is also discussed.

2 The π-Calculus And Bisimulations

2.1 Names, Conditions and Substitutions

The basic entities of the π-calculus are *names*, *i.e.* identifiers for communication channels. Let $\mathcal{N}$, ranged over by $x,\ y,\ z,\ \ldots$ be a countably infinite set of names. Vectors of names will be denoted by $\overline{x},\ \overline{y},\ \overline{z},\ \ldots$.

Conditions, ranged over by $C,\ D$, are finite sets of equality or inequality tests on names. We will write $n(C)$ for the set of names appearing in C where trivial identity tests of the form $x = x$ have been omitted. So, for instance, $n(\{x \neq y, z = z\}) = \{x, y\}$. We say C is a condition on a name set V if $n(C) \subseteq V$. *Matches*, ranged over by $M,\ N,\ L$, are special conditions which consist of only name equalities. Matches are part of the π-calculus, while conditions are only used in our meta-theory about the π-calculus, namely in the definition of symbolic bisimulations and in the formulation of inference rules.

Substitutions, ranged over by σ and δ, are partial mappings from $\mathcal{N}$ to $\mathcal{N}$, and $[\overline{y}/\overline{x}]$, where the length of $\overline{x}$ equals that of $\overline{y}$, is the substitution sending $\overline{x}$ to $\overline{y}$. If $\sigma = [\overline{y}/\overline{x}]$ the $n(\sigma) = \{\overline{y}\} \cup \{\overline{x}\}$.

A substitution σ satisfies a condition C, written $\sigma \models C$, if $x\sigma = y\sigma$ for any $x = y \in C$ and $x\sigma \neq y\sigma$ for any $x \neq y \in C$. So the elements in a condition are treated as conjuncts. We write $C \Rightarrow D$ to mean that $\sigma \models C$ implies $\sigma \models D$ for any substitution σ, and $C = D$ to mean $C \Rightarrow D$ and $D \Rightarrow C$. The notation $x =^C y$ will also be used to denote $C \models x = y$. A condition satisfied by every substitution will be denoted by *true*. For a finite set of conditions I, $\sigma \models \bigvee I$ if and only if $\sigma \models C$ for some $C \in I$.

Proposition 2.1 *The relation $C \Rightarrow D$ is decidable.*

Proof: We first generate equivalence classes from the equalities in C, along with a list of pairs of representatives of the equivalence classes such that the pair of representatives of the class of x and the class of y is in the list if and only if $x \neq y$ is in C. Then for each element e of D if e is an equality $a = b$ we check if a, b are in the same equivalence class; if e is an inequality $a \neq b$ we check if the pair consisting of the representatives of the class of a and the class of b are in the list. $\square$

We chose not to include disjunction as a construction operator for conditions since we can do without it. All the definitions and proofs in this paper will not be affected if disjunction is allowed.

A condition C is *consistent* if there are no $x,\ y \in \mathcal{N}$ such that $C \Rightarrow x = y$ and $C \Rightarrow x \neq y$. Otherwise it is *inconsistent*. It is easy to see that C is consistent if and only if there exists a substitution σ such that $\sigma \models C$. We will use *false* to denote inconsistent conditions. It is easy to see that C is consistent if and only if $\sigma \models C$ for some σ. C is *maximally consistent* on $V \subset \mathcal{N}$, written $C \in MC_V$, if it is consistent and for any $x,\ y \in V$ either $C \Rightarrow x = y$ or $C \Rightarrow x \neq y$.

Let $V \subset \mathcal{N}$. D is a *maximally consistent extension* of C on V, written $D \in MCE_V(C)$, if $D \Rightarrow C$, $n(D - C) \subseteq V$ and D is maximally consistent on V. If V is finite then $MCE_V(C)$ is also finite.

Lemma 2.2 $\bigvee MCE_V(C) = C$

Proof: We need to show for any σ, $\sigma \models \bigvee MCE_V(C)$ if and only if $\sigma \models C$. The "only if" part is trivial. For the "if" part, suppose $\sigma \models C$. Let

$$D = \{\, x = y \mid x, y \in V,\ \sigma \models x = y \,\} \cup \{\, x \neq y \mid x, y \in V,\ \sigma \models x \neq y \,\} \cup C$$

Then D is maximally consistent on V. Also $D \Rightarrow C$. Therefore $D \in MCE_V(C)$. By the construction of D, $\sigma \models D$. Hence $\sigma \models \bigvee MCE_V(C)$. $\square$

Lemma 2.3 *Suppose D is maximally consistent on V, C is a condition with $n(C) \subseteq V$. If $\sigma \models D$ and $\sigma \models C$ for some σ (in other words, $D \cup C$ is consistent), then $D \Rightarrow C$.*

Proof: By induction on the size of C. If C is empty then the result holds trivially. Otherwise let $C = C' \cup \{e\}$ with $e \notin C'$. By induction $D \Rightarrow C'$. Since $\sigma \models D$ and $\sigma \models e$, $D \cup \{e\}$ is consistent. Since D is maximally consistent on V and $n(e) \subseteq V$, $D \Rightarrow e$. Therefore $D \Rightarrow C$. $\square$

2.2 The Language And Operational Semantics

The language of π-calculus can be given by the following BNF grammar

$$\begin{array}{rcl} t & ::= & \mathbf{0} \quad | \quad \alpha.t \quad | \quad t+t \quad | \quad [x=y]t \quad | \quad (x)t \quad | \quad t \mid t \quad | \quad A(y_1,...,y_n) \\ \alpha & ::= & \tau \quad | \quad a(x) \quad | \quad \overline{a}x \end{array}$$

Most of these operators are from *CCS* [9]: $\mathbf{0}$ is the process capable of doing no action, $\alpha.t$ is action prefixing, $+$ is non-deterministic choice, $|$ is parallel composition, $(x)t$ is scope restriction. The *match* construction $[x=y]t$ allows the comparison of two names: if x and y are the same name then the process will behave like t; otherwise it will have no action. Each *identifier* A has a defining equation $A(x_1,...,x_n) \stackrel{def}{=} t$ associated with it. The precedences of these operators are as below (in decreasing order):

$$. \quad \text{restriction} \quad \text{match} \quad | \quad +$$

Substitutions bind tighter than any operator.

As in *CCS* τ represents communication. An input-prefixed process $a(x).t$ can receive a name along channel a and then behave like t with the received name in place of x. An output-prefixed process $\overline{a}x.t$ can emit the name x along a and continue like t.

In $a(x).t$ and $(x)t$ x is a bound name with scope t. This induces the notions of bound and free names as usual. The sets of bound and free names of t are denoted by $bn(t)$ and $fn(t)$, respectively. In a defining equation $A(x_1,...,x_n) \stackrel{def}{=} t$ it is required that $fn(t) \subseteq \{x_1,...,x_n\}$. Bound names induce the notion of α-equivalence as usual. In the sequel we will not distinguish between α-equivalent terms and will use $\equiv$ for both syntactical equality and α-equivalence. Furthermore whenever we introduce a binary relation on terms we will always assume it is closed with respect to $\equiv$.

The operational semantics of the language is reported in Figure 1, where a transition is of the form $t \stackrel{\alpha}{\longrightarrow} u$ with α ranging over four kinds of actions: τ, $a(x)$, $\overline{a}x$ and $\overline{a}(x)$. We have omitted the symmetric rules for sum and par. Transitions are defined up-to α-equivalence, i.e. α-equivalent terms are deemed to have the same transitions. Bound names for actions are defined by: $bn(a(x)) = bn(\overline{a}(x)) = \{x\}$ and $bn(\tau) = bn(\overline{a}x) = \emptyset$. Substitutions on actions are defined thus: $\tau\sigma = \tau$, $(ax)\sigma = a\sigma x\sigma$, $(a(x))\sigma = (a\sigma)(x)$ if $x \notin n(\sigma)$, and $(\overline{a}(x))\sigma = (\overline{a}\sigma)(x)$ if $x \notin n(\sigma)$. Note that for bound actions substitution is only defined when the substitution does not interfere with the bound name.

We quote a result from [10] (Lemma 2, page 54) below:

Lemma 2.4 *Suppose that $p \stackrel{a(y)}{\longrightarrow} p'$, where $a = x$ or $a = \overline{x}$ and that $z \notin fn(p)$. Then equally for some $p'' \equiv p'[z/y]$, $p \stackrel{a(z)}{\longrightarrow} p''$.*

The transitions defined in Figure 1 are *concrete* in the sense that they can always be fired regardless of the context in which terms are placed. In the work on general message-passing processes [5, 6] a more abstract form of transitions, called *symbolic transitions*,

$$
\text{pre}\ \frac{}{\alpha.t \xrightarrow{\alpha} t} \qquad \text{match}\ \frac{t \xrightarrow{\alpha} t'}{[x=x]t \xrightarrow{\alpha} t'}
$$

$$
\text{sum}\ \frac{t \xrightarrow{\alpha} t'}{t+u \xrightarrow{\alpha} t'} \qquad \text{par}\ \frac{t \xrightarrow{\alpha} t'}{t \mid u \xrightarrow{\alpha} t' \mid u}\ bn(\alpha) \cap fn(u) = \emptyset
$$

$$
\text{res}\ \frac{t \xrightarrow{\alpha} t'}{(x)t \xrightarrow{\alpha} (x)t'}\ x \notin n(\alpha) \quad \text{com}\ \frac{t \xrightarrow{a(x)} t' \quad u \xrightarrow{\overline{a}y} u'}{t \mid u \xrightarrow{\tau} t'[y/x] \mid u'}
$$

$$
\text{open}\ \frac{t \xrightarrow{\overline{a}x} t'}{(x)t \xrightarrow{\overline{a}(x)} t'}\ a \neq x \qquad \text{close}\ \frac{t \xrightarrow{a(x)} t' \quad u \xrightarrow{\overline{a}(x)} u'}{t \mid u \xrightarrow{\tau} (x)(t' \mid u')}
$$

$$
\text{id}\ \frac{t[y_1,...,y_n/x_1,...,x_n] \xrightarrow{\alpha} t'}{A(y_1,...,y_n) \xrightarrow{\alpha} t'}\ A(x_1,...,x_n) \stackrel{def}{=} t
$$

Figure 1: π-Calculus Transitional Semantics

has been used. A symbolic transition takes the form $t \xrightarrow{b,\alpha} u$, where b is a boolean condition and α is an action. Intuitively b represents the environments under which the action α can actually be fired from t. In the setting of the π-calculus b will be a set of *matches*, i.e. equality tests on names. This kind of transition has also been used in the work on open bisimulation by Sangiorgi [13].

The symbolic transitional semantics of the π-calculus is given in Figure 2, where we use $M,\ N,\ L$ to range over sets of matches. For notational convenience we write MN for the union of M and N. Also the symmetric rules for Sum and Par have been omitted.

Lemma 2.5 *1. If $t \xrightarrow{M,\alpha} u$ then $n(M) \cup fn(\alpha) \subseteq fn(t)$.*

2. If $t \xrightarrow{M,c(x)} u$ where c is a or $\overline{a}$, then for any $z \notin fn(t)$, $t \xrightarrow{M,c(z)} u'$ with $u' \equiv u[z/x]$.

Proof: By transition induction. □

The following technical lemma relates concrete and symbolic transitions.

Lemma 2.6 *1. If $t\sigma \xrightarrow{\alpha} p$ where $bn(\alpha) \cap (fn(t) \cup n(\sigma)) = \emptyset$, then there exist $M,\ \beta,\ t'$ s.t. $\sigma \models M,\ \alpha = \beta\sigma,\ p \equiv t'\sigma$ and $t \xrightarrow{M,\beta} t'$.*

2. If $t \xrightarrow{M,\alpha} t'$ then for any $\sigma \models M$ with $bn(\alpha) \cap (fn(t) \cup n(\sigma)) = \emptyset$, $t\sigma \xrightarrow{\alpha\sigma} p \equiv t'\sigma$.

Proof: We examine the case for bound output here (the other cases are similar to Lemmas 3.2, 3.3 and 3.4 in [6]).
1. Apply induction on why $t\sigma \xrightarrow{\alpha} p$.

$$\text{Pre}\ \frac{}{\alpha.t \xrightarrow{true,\alpha} t} \qquad \text{Match}\ \frac{t \xrightarrow{M,\alpha} t'}{[x{=}y]t \xrightarrow{L,\alpha} t'}\ L{=}\begin{cases} M[x{=}y] & \text{if } x{\neq}y \\ M & \text{if } x{=}y \end{cases}$$

$$\text{Sum}\ \frac{t \xrightarrow{M,\alpha} t'}{t+u \xrightarrow{M,\alpha} t'} \qquad \text{Par}\ \frac{t \xrightarrow{M,\alpha} t'}{t \mid u \xrightarrow{M,\alpha} t' \mid u}\ bn(\alpha)\cap fn(u)=\emptyset$$

$$\text{Res}\ \frac{t \xrightarrow{M,\alpha} t'}{(x)t \xrightarrow{M,\alpha} (x)t'}\ x{\notin}n(M,\alpha) \qquad \text{Open}\ \frac{t \xrightarrow{M,\overline{a}x} t'}{(x)t \xrightarrow{M,\overline{a}(x)} t'}\ x{\notin}n(M,a)$$

$$\text{Com}\ \frac{t \xrightarrow{M,a(x)} t' \quad u \xrightarrow{N,\overline{b}y} u'}{t \mid u \xrightarrow{L,\tau} t'[y/x] \mid u'}\ L{=}\begin{cases} MN[a=b] & \text{if } a{\neq}b \\ MN & \text{if } a{=}b \end{cases}$$

$$\text{Close}\ \frac{t \xrightarrow{M,a(x)} t' \quad u \xrightarrow{N,\overline{b}(x)} u'}{t \mid u \xrightarrow{L,\tau} (x)(t' \mid u')}\ L{=}\begin{cases} MN[a=b] & \text{if } a{\neq}\, b \\ MN & \text{if } a{=}b \end{cases}$$

$$\text{Id}\ \frac{t[y_1,...,y_n/x_1,...,x_n] \xrightarrow{M,\alpha} t'}{A(y_1,...,y_n) \xrightarrow{M,\alpha} t'}\ A(x_1,...,x_n) \stackrel{def}{=} t$$

Figure 2: π-Calculus Symbolic Transitional Semantics

- $((x)t)\sigma \xrightarrow{\overline{a}(x)} p$ because $t\sigma \xrightarrow{\overline{a}x} p$, $a \neq x$ (note that $((x)t)\sigma \equiv (x)t\sigma$ since $x \notin fn((x)t \cup n(\sigma))$). By induction there exist M, β, t' s.t. $\sigma \models M$, $\overline{a}x = \beta\sigma$, $p \equiv t'\sigma$ and $t \xrightarrow{M,\beta} t'$. Since $x \notin n(\sigma)$, β must be of the form $\overline{b}x$ with $b\sigma = a$. Since $x \notin fn((x)t \cup n(\sigma))$, by Lemma 2.5 $b \neq x$ and $x \notin n(M)$. Hence $(x)t \xrightarrow{M,\overline{b}(x)} t'$ by Open.
- $([y = z]t)\sigma \xrightarrow{\overline{a}(x)} p$ because $y\sigma = z\sigma$ and $t\sigma \xrightarrow{\overline{a}(x)} p$. By induction there exist M, β, t' s.t. $\sigma \models M$, $\overline{a}(x) = \beta\sigma$, $p \equiv t'\sigma$ and $t \xrightarrow{M,\beta} t'$. Since $x \notin n(\sigma)$, β must be of the form $\overline{b}(x)$ with $b\sigma = a$. Let $N = M[y = z]$ if $y \neq z$ and $N = M$ otherwise. Then $\sigma \models N$. Hence $[y = z]t \xrightarrow{N,\overline{b}(x)} t'$ by Match.
- The other cases are similar.

2. Apply induction on why $t \xrightarrow{M,\overline{a}(x)} t'$.

- $(x)t \xrightarrow{M,\overline{a}(x)} t'$ because $t \xrightarrow{M,\overline{a}x} t'$, $x \notin n(M,a)$. Let $\sigma \models M$ with $x \notin n(\sigma)$. By induction $t\sigma \xrightarrow{(\overline{a}x)\sigma} p \equiv t'\sigma$. We have $(\overline{a}x)\sigma = (\overline{a}\sigma)x$ and $(\overline{a}(x))\sigma = (\overline{a}\sigma)(x)$. Also $x \neq a\sigma$ because $x \notin n(\sigma)$. Hence $((x)t)\sigma \equiv (x)t\sigma \xrightarrow{(\overline{a}\sigma)(x)} p$ by open.

- $[y = z]t \stackrel{M,\overline{a}(x)}{\longrightarrow} t'$ because $t \stackrel{N,\overline{a}(x)}{\longrightarrow} t'$ with $M = N[y = z]$ if $y \neq z$ and $M = N$ otherwise. Let $\sigma \models M$. Then $\sigma \models N$ and $y\sigma = z\sigma$. By induction $t\sigma \stackrel{(\overline{a}(x))\sigma}{\longrightarrow} p \equiv t'\sigma$. Since $x \notin n(\sigma)$ we know $(\overline{a}(x))\sigma = (\overline{a}\sigma)(x)$. Hence $([y = z]t)\sigma \equiv [y\sigma = z\sigma]t\sigma \stackrel{(\overline{a}\sigma)(x)}{\longrightarrow} p$ by match.

- The other cases are similar.

$\square$

2.3 Weak Bisimulation Equivalences

Strong bisimulations for the π-calculus and their proof systems have been studied in [7], and here we only consider weak equivalences. For the bulk of this paper we will concentrate on *late* bisimulation, and we will sketch in a later section how these results can be carried over to *early* bisimulation in a systematic manner. So until Section 4, the word "bisimulation" means "late bisimulation".

As in *CCS*, weak bisimulation is defined in terms of weak transitions which ignore τ-moves. But late bisimulation insists that an input move from one process be matched by a *single* input move from the other. Therefore in defining late weak transitions input moves should *not* be allowed to absorb τ moves *after* them (for otherwise the resulted bisimulation relation would not be transitive). These τ moves will be taken care of later, when input actions are matched for bisimulation.

Definition 2.7 The (concrete) late weak transition relatios are generated by the following rules

- $p \stackrel{\varepsilon}{\Longrightarrow}_l p$.
- $p \stackrel{\alpha}{\longrightarrow} q$ implies $p \stackrel{\alpha}{\Longrightarrow}_l q$.
- $p \stackrel{\tau}{\longrightarrow}\stackrel{\alpha}{\Longrightarrow}_l q$ implies $p \stackrel{\alpha}{\Longrightarrow}_l q$.
- If α does not have the form $a(x)$ then $p \stackrel{\alpha}{\Longrightarrow}_l\stackrel{\tau}{\longrightarrow} q$ implies $p \stackrel{\alpha}{\Longrightarrow}_l q$.

Let $\hat{\tau}$ be ε, and $\hat{\alpha}$ be α for $\alpha \not\equiv \tau$. We will also write $\Longrightarrow_l$ for $\stackrel{\varepsilon}{\Longrightarrow}_l$. $\square$

Lemma 2.8 *Suppose that $p \stackrel{a(y)}{\Longrightarrow}_l p'$, where $a = x$ or $a = \overline{x}$ and that $z \notin fn(p)$. Then for some $p'' \equiv p'[z/y]$, $p \stackrel{a(z)}{\Longrightarrow}_l p''$.*

Proof: By induction on why $p \stackrel{a(y)}{\Longrightarrow}_l p'$, using Lemma 2.4. $\square$

Definition 2.9 A symmetric relation R is a late weak ground bisimulation if $(p, q) \in R$ implies:

- whenever $p \xrightarrow{a(x)} p'$ with $x \notin fn(p, q)$ then $q \overset{a(x)}{\Longrightarrow}_l q''$ for some q'' s.t for any y there is $q''[y/x] \Longrightarrow_l q'$ and $(p'[y/x], q') \in R$.
- whenever $p \xrightarrow{\alpha} p'$ for any other action α with $bn(\alpha) \cap fn(p, q) = \emptyset$ then $q \overset{\hat{\alpha}}{\Longrightarrow}_l q'$ for some q' and $(p', q') \in R$.

Write $p \dot{\approx}_l q$ if there exists a late weak ground bisimulation R such that $(p, q) \in R$.
t is late weak bisimular to u, written $t \approx_l u$, if $t\sigma \dot{\approx}_l u\sigma$ for every substitution σ. □

Similar to the case of *CCS*, $\approx_l$ is not preserved by $+$. So we have to work with the modified relation:

Definition 2.10 p and q are late ground observation congruent, written $p \dot{\simeq}_l q$, if

- whenever $p \xrightarrow{a(x)} p'$ with $x \notin fn(p, q)$ then $q \overset{a(x)}{\Longrightarrow}_l q''$ for some q'' s.t for any y there is $q''[y/x] \Longrightarrow_l q'$ and $p'[y/x] \approx_l q'$.
- whenever $p \xrightarrow{\alpha} p'$ for any other action α with $bn(\alpha) \cap fn(p, q) = \emptyset$ then $q \overset{\alpha}{\Longrightarrow}_l q'$ for some q' and $p' \approx_l q'$.

And similarly for q.
t is late observation congruent to u, written $t \simeq_l u$, if $t\sigma \dot{\simeq}_l u\sigma$ for every substitution σ.
□

To define the symbolic version of weak bisimulation, we first introduce the symbolic weak transition relation.

Definition 2.11 The late symbolic weak transitions are defined as the least relations satisfying the following rules:

- $t \overset{true,\varepsilon}{\Longrightarrow}_L t$.
- $t \xrightarrow{M,\alpha} u$ implies $t \overset{M,\alpha}{\Longrightarrow}_L u$.
- $t \xrightarrow{M,\tau} \overset{N,\alpha}{\Longrightarrow}_L u$ implies $t \overset{MN,\alpha}{\Longrightarrow}_L u$.
- If α does not have the form $a(x)$ then $t \overset{M,\alpha}{\Longrightarrow}_L \xrightarrow{N,\tau} u$ implies $t \overset{MN,\alpha}{\Longrightarrow}_L u$.

Let $\overset{M}{\Longrightarrow}_L$ be $\overset{M,\varepsilon}{\Longrightarrow}_L$, and $\overset{M,\hat{\alpha}}{\Longrightarrow}_L$ be $\overset{M,\alpha}{\Longrightarrow}_L$ for $\alpha \not\equiv \tau$. □

The two versions of weak transitions can also be related as in the case of single arrows:

Lemma 2.12 *1. If $t\sigma \overset{\alpha}{\Longrightarrow}_l p$ where $bn(\alpha) \cap (fn(t) \cup n(\sigma)) = \emptyset$, then there exist M, β, t' such that $\sigma \models M$, $\alpha = \beta\sigma$, $p \equiv t'\sigma$ and $t \overset{M,\beta}{\Longrightarrow}_L t'$.*

2. If $t \overset{M,\alpha}{\Longrightarrow}_L t'$ *then for any* $\sigma \models M$ *with* $bn(\alpha) \cap (fn(t) \cup n(\sigma)) = \emptyset$, $t\sigma \overset{\alpha\sigma}{\Longrightarrow}_l p \equiv t'\sigma$.

Proof:

1. Apply induction on why $t\sigma \overset{\alpha}{\Longrightarrow}_l p$.

   - $t\sigma \overset{\varepsilon}{\Longrightarrow}_l t\sigma$. Then $t \overset{M,\varepsilon}{\Longrightarrow}_L t$.
   - $t\sigma \overset{\alpha}{\longrightarrow} p$. Then by Lemma 2.6 there exist M, β, t' such that $\sigma \models M$, $\alpha = \beta\sigma$, $p \equiv t'\sigma$ and $t \overset{M,\beta}{\longrightarrow} t'$. Hence $t \overset{M,\beta}{\Longrightarrow}_L t'$.
   - $t\sigma \overset{\alpha}{\Longrightarrow}_l p$ because $t\sigma \overset{\tau}{\longrightarrow} q \overset{\alpha}{\Longrightarrow}_l p$. By Lemma 2.6 there exist M, t'' such that $\sigma \models M$, $q \equiv t''\sigma$ and $t \overset{M,\tau}{\longrightarrow} t''$. Now $t''\sigma \overset{\alpha}{\Longrightarrow}_l p$. So by the induction hypothesis there exist N, β, t' such that $\sigma \models N$, $\alpha = \beta\sigma$, $p \equiv t'\sigma$ and $t'' \overset{N,\beta}{\Longrightarrow}_L t'$. Hence $t \overset{MN,\beta}{\Longrightarrow}_L t'$ and $\sigma \models MN$.
   - α is not an input action and $t\sigma \overset{\alpha}{\Longrightarrow}_l p$ because $t\sigma \overset{\alpha}{\Longrightarrow}_l q \overset{\tau}{\longrightarrow} p$. By the induction hypothesis there exist M, β, t'' such that $\sigma \models M$, $\alpha = \beta\sigma$, $q \equiv t''\sigma$ and $t \overset{M,\beta}{\Longrightarrow}_L t''$. Now $t''\sigma \overset{\tau}{\longrightarrow} p$ because $q \overset{\tau}{\longrightarrow} p$ and $q \equiv t''\sigma$. So by Lemma 2.6 there exist N, t' such that $\sigma \models N$, $p \equiv t'\sigma$ and $t'' \overset{\tau}{\longrightarrow} t'$. Hence $t \overset{MN,\beta}{\Longrightarrow}_L t'$ and $\sigma \models MN$.

2. Apply induction on why $t \overset{M,\alpha}{\Longrightarrow}_L t'$. Details are similar to the above case.

$\square$

We write $\alpha =^C \beta$ to mean

$$\begin{array}{l}
\text{if } \alpha \equiv \tau \text{ then } \beta \equiv \tau \\
\text{if } \alpha \equiv \overline{a}x \text{ then } \beta \equiv \overline{b}y \ \text{ and } \ C \Rightarrow a = b,\ C \Rightarrow x = y \\
\text{if } \alpha \equiv \overline{a}(x) \text{ then } \beta \equiv \overline{b}(x) \ \text{ and } \ C \Rightarrow a = b \\
\text{if } \alpha \equiv a(x) \text{ then } \beta \equiv b(x) \ \text{ and } \ C \Rightarrow a = b
\end{array}$$

Definition 2.13 A condition indexed family of symmetric relations $\mathcal{S} = \{S^C\}$ is a late weak symbolic bisimulation if $(t, u) \in S^C$ implies

whenever $t \overset{M,\alpha}{\longrightarrow} t'$ with $bn(\alpha) \cap fn(t, u, C) = \emptyset$, then for each $D \in MCE_{fn(t,u)}(C \cup M)$ there is a $u \overset{N,\hat{\beta}}{\Longrightarrow}_L u'$ such that $D \Rightarrow N$, $\alpha =^D \beta$, and

- If $\alpha \equiv a(x)$ then for each $D' \in MCE_{fn(t,u)\cup\{x\}}(D)$ there is $u' \overset{N'}{\Longrightarrow}_L u''$ such that $D' \Rightarrow N'$ and $(t', u'') \in S^{D'}$.
- If $\alpha \equiv \overline{a}(x)$ then $(t', u') \in S^{D \cup \{\, x \neq y \mid y \in fn(\alpha.t', \beta.u')\,\}}$.
- Otherwise $(t', u') \in S^D$.

$t \approx_L^C u$ if and only if there exists a late weak symbolic bisimulation $\mathcal{S} = \{S^C\}$ such that $(t,u) \in S^C$. $\square$

Proposition 2.14 *$t \approx_L^C u$ if and only if $t \approx_L^D u$ for any $D \in MCE_{fn(t,u)}(C)$.*

Proof: ($\Longrightarrow$) Suppose $t \approx_L^C u$ and $D \in MCE_{fn(t,u)}(C)$. So there is a late weak symbolic bisimulation $\mathcal{S}$ such that $(t,u) \in S^C \in \mathcal{S}$. Set $S' = \{(t,u)\}$ and $\mathcal{S}' = \mathcal{S} \cup \{S'\}$.

To show $\mathcal{S}'$ is also a symbolic bisimulation we only need to check the case $(t,u) \in S'$. Let $t \xrightarrow{M,\alpha} t'$ with $bn(\alpha) \cap fn(t,u,D) = \emptyset$. Because $D \Rightarrow C$, for each $D' \in MCE_{fn(t,u)}(D \cup M)$ we also have $D' \in MCE_{fn(t,u)}(C \cup M)$. Using the fact that $(t,u) \in S^C$ and $\mathcal{S}$ is a symbolic bisimulation the rest of the proof is immediate.

($\Longleftarrow$) Suppose for every $D \in MCE_{fn(t,u)}(C)$ we have $t \approx_L^D u$, *i.e.* there is a late weak symbolic bisimulation $\mathcal{S_D}$ such that $(t,u) \in S^D$ for some $S^D \in \mathcal{S_D}$. Set $S'' = \{(t,u)\}$ and $\mathcal{S} = (\bigcup\{\, \mathcal{S}_D \mid D \in MCE_{fn(t,u)}(C)\,\}) \cup \{S''\}$.

To show $\mathcal{S}$ is a symbolic bisimulation we only need to check the case $(t,u) \in S''$. Let $t \xrightarrow{M,\alpha} t'$ with $bn(\alpha) \cap fn(t,u,C) = \emptyset$. For any $D \in MCE_{fn(t,u)}(C \cup M)$ we also have $D \in MCE_{fn(t,u)}(D \cup M)$. The rest of the proof is straightforward using the fact that $(t,u) \in S^D \in \mathcal{S}_D$ and $\mathcal{S}_D$ is a symbolic bisimulation. $\square$

Proposition 2.15 *Suppose $C \in MC_{fn(t,u)}$. Then $t \approx_L^C u$ if and only if $t\sigma \dot{\approx}_l u\sigma$ for any $\sigma \models C$.*

Proof: ($\Longrightarrow$) Define

$$R = \{\, (t\sigma, u\sigma) \mid t \approx_L^C u \text{ for some } C \in MC_{fn(t,u)}, \text{ and } \sigma \models C \,\}$$

We show R is a weak ground bisimulation.

Suppose $(t\sigma, u\sigma) \in R$, *i.e.* there is a $C \in MC_{fn(t,u)}$ such that $\sigma \models C$ and $t \approx_L^C u$. Let $t\sigma \xrightarrow{\alpha} p$ with $bn(\alpha) \cap fn(t\sigma, u\sigma) = \emptyset$. The proof proceeds by case analysis on the type of α.

- $\alpha \equiv a(x)$. Choose a fresh name w s.t. $w \notin fn(t,u) \cup n(\sigma) \cup n(C)$ (hence $w \notin fn(t\sigma, u\sigma)$). By Lemma 2.4, $t\sigma \xrightarrow{a(w)} p_1$ with $p \equiv p_1[x/w]$. Note that $x \notin fn(p_1)$. By Lemma 2.6 there exist M, a', t' such that $\sigma \models M$, $a \equiv a'\sigma$, $p \equiv t'\sigma$ and $t \xrightarrow{M,a'(w)} t'$. By Lemma 2.5, $n(M) \subseteq fn(t,u)$. Since $\sigma \models C$, $\sigma \models M$ and $C \in MC_{fn(t,u)}$, by Lemma 2.3 $C \Rightarrow M$. So $C \cup M = C$. Hence, up to logical equivalence, C is the only element in $MCE_{fn(t,u)}(C \cup M)$. By Definition 2.13 there is $u \overset{N,b'(w)}{\Longrightarrow}_L u'$ with $C \Rightarrow N$, $a' =^C b'$ and the other properties guaranteed by the definition. From $\sigma \models C$ and $C \Rightarrow N$ follows $\sigma \models N$, hence by Lemma 2.12

$$u\sigma \overset{b(w)}{\Longrightarrow}_l q_1 \equiv u'\sigma \text{ with } b \equiv b'\sigma \tag{1}$$

Since $a' =^C b'$ and $\sigma \models C$, $\sigma \models a' = b'$. Hence $a = a'\sigma = b'\sigma = b$. Apply Lemma 2.8 we have

$$u\sigma \overset{b(x)}{\Longrightarrow}_l q \text{ with } q[w/x] \equiv q_1 \tag{2}$$

Now we show for any y there is a q' such that $q[y/x] \Longrightarrow_l q'$ and $(p[y/x], q') \in R$. From $\sigma \models C$ and $w \notin n(C) \cup n(\sigma)$ we know $\sigma[y/w] \models C$. Let $D = C \cup \{ e \mid e \text{ has the form } w = z \text{ or } w \neq z,\ z \in fn(t,u),\ \sigma[y/w] \models e \}$. Then, since $C \in MC_{fn(t,u)}$,

$$D \in MCE_{fn(t,u) \cup \{w\}}(C) \text{ and } \sigma[y/w] \models D \tag{3}$$

Since $t \approx_L^C u$, there is $u' \overset{N'}{\Longrightarrow}_L u''$ such that $D \Rightarrow N'$ (hence $\sigma[y/w] \models N'$) and $t' \approx_L^D u''$. By Lemma 2.12

$$u'\sigma[y/w] \Longrightarrow_l q' \text{ for some } q' \equiv u''\sigma[y/w] \tag{4}$$

From (1) and (2), $u'\sigma \equiv q[w/x]$, so $u'\sigma[y/w] \equiv q[w/x][y/w]$. If $w \equiv x$ then $q[w/x][y/w] \equiv q[y/x]$ holds trivially. If $w \not\equiv x$ then from $fn(q) \subseteq fn(u\sigma) \cup \{x\}$ we know $w \notin fn(q)$, so $q[w/x][y/w] \equiv q[y/x]$ also. Therefore $u'\sigma[y/w] \equiv q[y/x]$. From (4) we have $q[y/x] \Longrightarrow_l q' \equiv u''\sigma[y/w]$. Now from (3) and $fn(t', u'') \subseteq fn(t,u) \cup \{w\}$, we have $D \in MC_{fn(t',u'')}$. Hence by the construction of R, $(t'\sigma[y/w], u''\sigma[y/w]) \in R$. Since $x \notin fn(p_1)$, $p[y/x] \equiv p_1[x/w][y/x] \equiv p_1[y/w] \equiv t'\sigma[y/w]$. Moreover $q' \equiv u''\sigma[y/w]$. Therefore $(p[y/x], q') \in R$.

- $\alpha \equiv \overline{a}(x)$. We proceed in the same way as in the previous case until (2), with a, a', b', b replaced by $\overline{a}, \overline{a'}, \overline{b'}, \overline{b}$, respectively, and $t' \approx_L^D u'$ where $D = C \cup \{ w \neq y \mid y \in fn(\overline{a'}(w).t', \overline{b'}(w).u') \}$. Now we have $D \in MC_{fn(t',u')}$. From $\sigma \models C$ and $w \notin n(C) \cup n(\sigma)$ follows $\sigma[x/w] \models C$. Furthermore, since $x \notin fn(t\sigma, u\sigma)$ and $fn(\overline{a'}(w).t', \overline{b'}(w).u') \subseteq fn(t\sigma, u\sigma)$, $\sigma[x/w] \models w \neq y$ for every $y \in fn(\overline{a'}(w).t', \overline{b'}(w).u')$. Hence $\sigma[x/w] \models D$. Therefore, by the construction of R, $(t'\sigma[x/w], u'\sigma[x/w]) \in R$. Since $w \notin fn(q)$, $q \equiv q[w/x][x/w] \equiv q_1[x/w] \equiv u'\sigma[x/w]$. Moreover $p \equiv p_1[x/w] \equiv t'\sigma[x/w]$. Therefore $(p,q) \in R$.

The other cases are similar and easier.

($\Longleftarrow$) Define

$$S^C = \{ (t,u) \mid C \in MC_{fn(t,u)},\ t\sigma \dot{\approx}_l u\sigma \text{ for some } \sigma \models C \}$$

and $\mathcal{S} = \{S^C\}$. We show $\mathcal{S}$ is a weak symbolic bisimulation.

Suppose $(t,u) \in S^C$, *i.e.* $C \in MC_{fn(t,u)}$, $t\sigma \dot{\approx}_l u\sigma$ for some $\sigma \models C$. Let $t \overset{M,\alpha}{\longrightarrow} t'$ with $bn(\alpha) \cap fn(t,u,C) = \emptyset$. We may assume $bn(\alpha) \cap n(\sigma) = \emptyset$. By Lemma 2.5, $n(M) \subseteq fn(t,u)$. For each condition in $MCE_{fn(t,u)}(C \cup M)$ we need to find a matching symbolic transition from u. If M is inconsistent with C then $MCE_{fn(t,u)}(C \cup M)$ is empty and we are done; Otherwise, since $C \in MC_{fn(t,u)}$ and $n(M) \subseteq fn(t,u)$, we have $C \Rightarrow M$ by Lemma 2.3. Hence $MCE_{fn(t,u)}(C \cup M) = \{C\}$ and $\sigma \models M$. By Lemma 2.6 $t\sigma \overset{\alpha\sigma}{\longrightarrow} p \equiv t'\sigma$. Again there are four cases to consider according to the type of α.

- $\alpha \equiv a(x)$. Since $t\sigma \dot{\approx}_l u\sigma$, there is $u\sigma \stackrel{\alpha\sigma}{\Longrightarrow}_l q'$ such that for all y there exists $q'[y/x] \Longrightarrow_l q$ and $p[y/x] \dot{\approx}_l q$. Apply Lemma 2.12 we get N, β, u' such that $\sigma \models N$, $\beta\sigma = \alpha\sigma$, $u'\sigma \equiv q'$ and $u \stackrel{N}{\Longrightarrow}_L u'$. Since $fn(\alpha, \beta) \subseteq fn(t, u)$ and C is maximally consistent on $fn(t, u)$, by Lemma 2.3 $\alpha =^C \beta$.

  Let $D \in MCE_{fn(t,u)\cup\{x\}}(C)$ we must show there is $u' \stackrel{N'}{\Longrightarrow}_L u''$ such that $D \Rightarrow N'$ and $(t', u') \in S^D$. Let

  $$y = \begin{cases} z\sigma & \text{if } D \Rightarrow x = z \text{ for some } z \in fn(t,u) \\ x & \text{otherwise} \end{cases}$$

  The well-definedness of y can be argued as follows: if $D \Rightarrow x = z'$ and $D \Rightarrow x = z''$ for z', $z'' \in fn(t, u)$ then $D \Rightarrow z' = z''$, i.e. $z' = z''$ is consistent with C. Since $z', z'' \in fn(t, u)$ and C is maximally consistent on $fn(t, u)$, $C \Rightarrow z' = z''$. Therefore $z'\sigma = z''\sigma$.

  We claim $\sigma[y/x] \models D$: for any $e \in D$, if $e \in C$ then $\sigma \models e$, so $\sigma[y/x] \models e$ because $x \notin n(e)$. Otherwise e must be of the form $x = z$ or $x \neq z$ with $z \in fn(t, u)$. In the former case $x(\sigma[y/x]) = z\sigma$ hence $\sigma[y/x] \models x = z$; in the latter case $x(\sigma[y/x]) = x$ hence $\sigma[y/x] \models x \neq z$. Therefore $\sigma[y/x] \models e$ holds in any case.

  Now $u'\sigma[y/x] \equiv q'[y/x] \Longrightarrow_l q$, so by Lemma 2.12 there exist N', u'' such that $\sigma[y/x] \models N'$, $u''\sigma[y/x] \equiv q$ and $u' \stackrel{N'}{\Longrightarrow}_L u''$. Since $D \in MC_{fn(t,u)\cup\{x\}}$ and $n(N') \subseteq fn(t, u) \cup \{x\}$, $D \Rightarrow N'$. Since $t'\sigma[y/x] \equiv p[y/x]$ and $p[y/x] \dot{\approx}_l q$, $t'\sigma[y/x] \dot{\approx}_l u''\sigma[y/x]$ as well. Furthermore $D \in MC_{fn(t',u'')}$ since $fn(t', u'') \subseteq fn(t, u) \cup \{x\}$. Therefore $(t', u'') \in S^D$.

- $\alpha \equiv \overline{a}(x)$. Since $t\sigma \dot{\approx}_l u\sigma$, there is $u\sigma \stackrel{\alpha\sigma}{\Longrightarrow}_l q$ such that $p \dot{\approx}_l q$. Apply Lemma 2.12 we get N, b, u' such that $\sigma \models N$, $\sigma \models a = b$, $u'\sigma \equiv q'$ and $u \stackrel{N,\overline{b}(x)}{\Longrightarrow}_L u'$. Since $n(N) \cup \{a, b\} \subseteq fn(t, u)$, $\sigma \models C$ and C is maximally consistent on $fn(t, u)$, we have $C \Rightarrow N$ and $\overline{a}(x) =^C \overline{b}(x)$. Moreover, since $x \notin fn(t, u) \cup n(\sigma)$ and $fn(\overline{a}(x).t', \overline{b}(x).u') \subseteq fn(t, u) \cup \{x\}$, we know $C \cup \{\, x \neq y \mid y \in fn(\overline{a}(x).t', \overline{b}(x).u') \,\} \in MC_{fn(\overline{a}(x).t', \overline{b}(x).u')}$. Finally from $x \notin fn(t, u) \cup n(\sigma)$ it follows that $\sigma \models \{\, x \neq y \mid y \in fn(\overline{a}(x).t', \overline{b}(x).u') \,\}$, hence $\sigma \models C \cup \{\, x \neq y \mid y \in fn(\overline{a}(x).t', \overline{b}(x).u') \,\}$. Therefore $(t', u') \in S^{C\cup\{\, x\neq y \mid y\in fn(\overline{a}(x).t', \overline{b}(x).u') \,\}}$.

The other two cases are similar and easier, and we leave them to the reader. $\square$

Combining the above two propositions gives the main result of this section:

Theorem 2.16 *$t \approx_L^C u$ if and only if $t\sigma \dot{\approx}_l u\sigma$ for every $\sigma \models C$.*

Proof: By Proposition 2.14 $t \approx_L^C u$ if and only if $t \approx_L^D u$ for every $D \in MCE_{fn(t,u)}(C)$. By Lemma 2.2 $\sigma \models C$ if and only if $\sigma \models D$ for some $D \in MCE_{fn(t,u)}(C)$. By Proposition 2.15, for any $D \in MCE_{fn(t,u)}(C)$, $t \approx_L^D u$ if and only if $t\sigma \dot{\approx}_l u\sigma$ for any $\sigma \models D$.

Hence the conclusion. □

Late symbolic observation congruence is defined in terms of $\approx_L^C$ as usual:

Definition 2.17 Two terms t, u are *late symbolic observation congruent* with respect to C, written $t \simeq_L^C u$, if whenever $t \xrightarrow{M,\alpha} t'$ with $bn(\alpha) \cap fn(t,u,C) = \emptyset$, then for each $D \in MCE_{fn(t,u)}(C \cup M)$ there is a $u \overset{N,\beta}{\Longrightarrow}_L u'$ such that $D \Rightarrow N$, $\alpha =^D \beta$, and

- If $\alpha \equiv a(x)$ then for each $D' \in MCE_{fn(t,u) \cup \{x\}}(D)$ there is $u' \overset{M'}{\Longrightarrow}_L u''$ such that $D' \Rightarrow M'$ and $t' \approx_L^{D'} u''$.
- If $\alpha \equiv \overline{a}(x)$ then $t' \approx_L^{D \cup \{\, x \neq y | y \in fn(\alpha.t', \beta.u') \,\}} u'$.
- Otherwise $t' \approx_L^D u'$.

And similarly for u. □

Theorem 2.18 *$t \simeq_L^C u$ if and only if $t\sigma \stackrel{.}{\simeq}_l u\sigma$ for every $\sigma \models C$.*

Proof: Similar to that of Theorem 2.16. □

3 The Inference System

This section is devoted to formulating a proof system for late symbolic observation congruence and proving its soundness and completeness. By Theorem 2.18 it is also sound and complete for late observation congruence. It is well-known that such a proof system does not exist for the full calculus, so we shall restrict ourselves to the finite fragment, *i.e.*, leave out recursive definitions. We will first concentrate on a sublanguage without the parallel composition operator $|$. Later we will see that this operator can be easily axiomatised.

The inference system is reported in Figure 3. It inherits many features from the proof systems for general message-passing processes presented in [6]. The judgments are of the form

$$C \triangleright t = u$$

where C is a condition. An important difference is that the inference system of [6] relies on some "oracle" to answer questions concerning data, while in the present setting we have a *decidable* theory for the message domain which is simply the plain set of channel names. Therefore in the present inference system questions about names are treated as side conditions to inference rules.

$$\text{EQUIV} \quad \frac{}{\mathit{true} \rhd t = t} \quad \frac{C \rhd t = u}{C \rhd u = t} \quad \frac{C \rhd t = u \;\; C \rhd u = v}{C \rhd t = v}$$

$$\text{AXIOM} \quad \frac{}{\mathit{true} \rhd t = u}\; t = u \text{ is an axiom instance}$$

$$\text{CHOICE} \quad \frac{C \rhd t_i = u_i \quad i \in \{1,2\}}{C \rhd t_1 + t_2 = u_1 + u_2}$$

$$\text{L-INPUT} \quad \frac{C \rhd t = u}{C \rhd a(x).t = b(x).u}\; C \Rightarrow a = b,\; x \notin n(C)$$

$$\text{OUTPUT} \quad \frac{C \rhd t = u}{C \rhd \bar{a}x.t = \bar{b}y.u}\; C \Rightarrow a = b,\; C \Rightarrow x = y$$

$$\text{TAU} \quad \frac{C \rhd t = u}{C \rhd \tau.t = \tau.u}$$

$$\text{MATCH} \quad \frac{C \cup \{x = y\} \rhd t = u \quad C \cup \{x \neq y\} \rhd \mathbf{0} = u}{C \rhd [x = y]t = u}$$

$$\text{RES} \quad \frac{C \cup \{\, x \neq y \mid y \in \mathit{fn}((x)t,(x)u) \,\} \rhd t = u}{C \rhd (x)t = (x)u}\; x \notin n(C)$$

$$\text{PARTITION} \quad \frac{C \cup \{x = y\} \rhd t = u \quad C \cup \{x \neq y\} \rhd t = u}{C \rhd t = u}$$

$$\text{CONSEQ} \quad \frac{C \rhd t = u}{C' \rhd t = u}\; C' \Rightarrow C$$

$$\text{ABSURD} \quad \frac{}{\mathit{false} \rhd t = u}$$

Figure 3: The Inference Rules for Late Symbolic Bisimulation

As we are working modulo α-equivalence, we also assume the following rule

$$\text{ALPHA} \; \frac{}{\mathit{true} \rhd t = u}\; t \text{ and } u \text{ are } \alpha\text{-equivalent}$$

In order to derive interesting equations from this inference system we need some standard axioms. These are listed in Figure 4. They include the axioms for the choice operator (S1 – S4), the axioms for the restriction operator (R1 – R5), and the τ-laws (T1 – T3). Let $\mathcal{A}_s = \{S1, S2, S3, S4, R1, R2, R3, R4, R5\}$ and $\mathcal{A}_w = \mathcal{A}_s \cup \{T1, T2, T3\}$.

We write $\vdash C \rhd t = u$ and $\vdash_w C \rhd t = u$ to mean that $C \rhd t = u$ can be derived from $\mathcal{A}_s$ and $\mathcal{A}_w$, respectively, using the above inference rules. It is immediate from the definition that $\vdash C \rhd t = u$ implies $\vdash_w C \rhd t = u$. It is shown in [7] that $\vdash$ is sound and complete for strong bisimulation. In the rest of this paper we shall establish the soundness and completeness of $\vdash_w$ for late observation congruence.

S1	$X + \mathbf{0} = X$	R1	$(x)\mathbf{0} = \mathbf{0}$
S2	$X + X = X$	R2	$(x)\alpha.X = \alpha.(x)X$ if $x \notin n(\alpha)$
S3	$X + Y = Y + X$	R3	$(x)\alpha.X = \mathbf{0}$ if x is the subject of α
S4	$(X + Y) + Z = X + (Y + Z)$	R4	$(x)(y)X = (y)(x)X$
		R5	$(x)(X + Y) = (x)X + (x)Y$

T1	$\alpha.\tau.X = \alpha.X$
T2	$X + \tau.X = \tau.X$
T3	$\alpha.(X + \tau.Y) + \alpha.Y = \alpha.(X + \tau.Y)$

Figure 4: The Axioms for Choice, Restriction and τ

Some useful properties of these proof systems, namely Propositions 3.1, 3.2, 3.3, are listed below.

Proposition 3.1 *1. If $C \Rightarrow M$ and $\vdash C \rhd t = u$ then $\vdash C \rhd Mt = u$.*

2. If $C \cup M \Rightarrow$ false then $\vdash C \rhd Mt = \mathbf{0}$.

3. $\vdash t = t + Mt$.

4. $\vdash M(t + u) = Mt + Mu$.

Proof: As examples we prove *1* and *3*.
1. By induction on the length of M. If M is empty then $M = \mathit{true}$ and the result is immediate. Otherwise let $M = [x = y]N$. Using MATCH we need to show $\vdash C \cup \{x = y\} \rhd Nt = u$ and $\vdash C \cup \{x \neq y\} \rhd \mathbf{0} = u$. From $C \Rightarrow M$ we know $C \Rightarrow N$ and $C \Rightarrow x = y$. So by the induction hypothesis, $\vdash C \rhd Nt = u$. By CONSEQ, $\vdash C \cup \{x = y\} \rhd Nt = u$. For the remaining proof obligation we know $C \Rightarrow x = y$, hence $C \cup \{x \neq y\} \Rightarrow \mathit{false}$. Now applications of CONSEQ and ABSURD give $\vdash C \cup \{x \neq y\} \rhd \mathbf{0} = u$.
3. Apply induction on the length of M. When M is empty the result defaults to S2. For the induction step let $M = [x = y]N$. Using PARTITION we need to show

$$\vdash x = y \rhd t = t + [x = y]Nt \quad \text{and} \quad \vdash x \neq y \rhd t = t + [x = y]Nt$$

By *1* we have $\vdash x = y \rhd Nt = [x = y]Nt$, so using the induction hypothesis we derive

$$\begin{aligned} \vdash x = y \rhd t &= t + Nt \\ &= t + [x = y]Nt \end{aligned}$$

For the second subgoal we have $\vdash x \neq y \rhd [x = y]Nt = \mathbf{0}$ by *2*, hence $\vdash x \neq y \rhd t = t + [x = y]Nt$ by S1. □

The rule PARTITION permits a case analysis on the name space represented by a condition: To see if $t = u$ holds over C, we can decompose C into $C \cup \{x = y\}$ and $C \cup \{x \neq y\}$, and examine each separately. We have seen an example of its application in the proof of the above proposition. This rule can be generalised to allow arbitrary decompositions. The following proposition gives the case for an extremal decomposition, *i.e.* the decomposition of a condition into its maximally consistent extensions (Lemma 2.2):

Proposition 3.2 *If* $\vdash D \rhd t = u$ *for each* $D \in MCE_V(C)$ *where* $V = fn(t, u)$*, then* $\vdash C \rhd t = u$*.*

Proof: Suppose $\vdash D \rhd t = u$ for each $D \in MCE_V(C)$. Let $E_V(C) = \{\, x = y \mid x, y \in V,\ C \not\Rightarrow x = y \text{ and } C \not\Rightarrow x \neq y \,\}$. Apply induction on the cardinality of $E_V(C)$.

If $E_V(C)$ is empty then C is the only maximally consistent extension of itself, and the result holds trivially.

Otherwise assume $E_V(C)$ has $n+1$ elements and let $x = y \in E_V(C)$. Each maximally consistent extension of C contains either $x = y$ or $x \neq y$, so we have

$$MCE_V(C) = MCE_V(C \cup \{x = y\}) \cup MCE_V(C \cup \{x \neq y\})$$

Since

$$E_V(C \cup \{x = y\}) \subseteq E_V(C) - \{x = y\}$$
$$E_V(C \cup \{x \neq y\}) \subseteq E_V(C) - \{x \neq y\}$$

we can invoke induction to obtain $\vdash C \cup \{x = y\} \rhd t = u$ and $\vdash C \cup \{x \neq y\} \rhd t = u$. From these an application of PARTITION gives the required $\vdash C \rhd t = u$. $\square$

The following proposition summarises the interaction between the restriction and match operators.

Proposition 3.3 *1.* $\vdash (x)[x = x]t = (x)t$*.*

2. $\vdash (x)[x = y]t = \mathbf{0}$*.*

3. If $x \notin \{y,\ z\}$ *then* $\vdash (x)[y = z]t = [y = z](x)t$*.*

Proof: We only show *2* here. The proofs of the others are similar.

Let $C = \{\, x \neq z \mid z \in fn((x)[x = y]t) \,\}$. Then $x \neq y \in C$. By R1 and RES we need to show $\vdash C \rhd [x = y]t = \mathbf{0}$. Using MATCH this reduces to $\vdash C \cup \{x = y\} \rhd t = \mathbf{0}$ and $\vdash C \cup \{x \neq y\} \rhd \mathbf{0} = \mathbf{0}$. The later is trivial while the former follows from ABSURD. $\square$

Theorem 3.4 *(Soundness of* $\vdash_w$*) If* $\vdash_w C \rhd t = u$ *then* $t \simeq_L^C u$*.*

Proof: By a standard induction on the length of the derivation. As an example we show the case for the L-INPUT rule.

Assuming $t \simeq_L^C u$ and $x \notin n(C)$, we show $a(x).t \simeq_L^C a(x).u$ as follows. Since $x \notin fn(a(x).t, a(x).u, C)$, we can just let $a(x).t \stackrel{true,a(x)}{\longrightarrow} t$. For any $D \in MCE_{fn(a(x).t,a(x).u)}(C)$, we have a matching move $a(x).u \stackrel{true,a(x)}{\longrightarrow} u$. For any $D' \in MCE_{fn(a(x).t,a(x).u)\cup\{x\}}(D)$, we have $u \Longrightarrow_L u$. Since $fn(t,u) \subseteq fn(a(x).t, a(x).u) \cup \{x\}$, $D' \in MCE_{fn(t,u)}(C)$. Hence $t \approx_L^{D'} u$ by Proposition 2.14 and the fact that $t \simeq_L^C u$ implies $t \approx_L^C u$. □

The rest of this section is devoted to the proof of the completeness of $\vdash_w$.

The *height* of a term t is defined inductively thus

- $\mid \mathbf{0} \mid = 0$
- $\mid t + u \mid = \ max\{\mid t \mid, \mid u \mid\}$
- $\mid [x = y]t \mid = \mid t \mid$
- $\mid (x)t \mid = \mid t \mid$
- $\mid \alpha.t \mid = 1 + \mid t \mid$

If $a \neq x$ then we abbreviate $(x)\overline{a}x.t$ as $\overline{a}(x).t$. $\overline{a}(x)$ is a derived action and is called *bound output*. Using RES, the following two equations can be easily derived from T1 and T3:

$$\overline{a}(x).\tau.X = \overline{a}(x).X$$

$$\overline{a}(x).(X + \tau.Y) + \overline{a}(x).Y = \overline{a}(x).(X + \tau.Y)$$

So from now on we will allow the action α in T1 and T3 to range over bound output as well.

We also have the following derived inference rule for bound output:

Proposition 3.5 *Suppose $C \Rightarrow a = b$ and $x \notin n(C)$. If $\vdash C \cup \{\, x \neq y \mid y \in fn(\overline{a}(x).t, \overline{b}(x).u) \,\} \rhd t = u$ then $\vdash C \rhd \overline{a}(x).t = \overline{b}(x).u$.*

Proof: Since $\vdash C \cup \{\, x \neq y \mid y \in fn(\overline{a}(x).t, \overline{b}(x).u) \,\} \rhd t = u$ and $C \Rightarrow a = b$, by OUTPUT we obtain $\vdash C \cup \{\, x \neq y \mid y \in fn(\overline{a}(x).t, \overline{b}(x).u) \,\} \rhd \overline{a}x.t = \overline{b}x.u$. From this and $x \notin n(C)$ an application of RES gives the required $\vdash C \rhd \overline{a}(x).t = \overline{b}(x).u$. □

A term is *restriction-free* if, using the above abbreviation, it does not explicitly contain any occurrences of the restriction operator.

Lemma 3.6 *For any term t there is a restriction-free term t' such that $\vdash t = t'$ and $\mid t' \mid \leq \mid t \mid$.*

Proof: Using Axioms R1 – R5 and Proposition 3.3, we can push each restriction inwards, until it either disappears or gives rise to a bound output. This process does not increase the height of t. $\square$

A restriction-free term is a *standard form* if it has the form $\sum_{i \in I} M_i \alpha_i.t_i$ and each t_i is a standard form.

Lemma 3.7 *For any term t there is a standard form t' such that $\vdash t = t'$ and $\mid t' \mid \leq \mid t \mid$.*

Proof: By Lemma 3.6 we may assume t is restriction-free. The proof is by straightforward induction on the structure of t. $\square$

We will need a generalised form of T3:

Lemma 3.8 *If $n(M) \cap bn(\alpha) = \emptyset$ then $\vdash_w \alpha.(t + M\tau.u) = \alpha.(t + M\tau.u) + M\alpha.u$.*

Proof: By Proposition 3.1, $\vdash_w \alpha.(t+M\tau.u) = \alpha.(t+M\tau.u)+M\alpha.(t+M\tau.u)$. Moreover

$$\begin{array}{rcl}
\vdash_w M\alpha.(t + M\tau.u) & = & M\alpha.M(t + M\tau.u) \\
& = & M\alpha.M(t + \tau.u) \\
& = & M\alpha.(t + \tau.u) \\
& \stackrel{T3}{=} & M(\alpha.(t + \tau.u) + \alpha.u) \\
& = & M\alpha.(t + \tau.u) + M\alpha.u \\
& = & M\alpha.(t + M\tau.u) + M\alpha.u
\end{array}$$

Hence $\vdash_w \alpha.(t + M\tau.u) = \alpha.(t + M\tau.u) + M\alpha.u$. $\square$

Definition 3.9 A standard form $\sum_{i \in I} M_i \alpha_i.t_i$ is a *full standard form* if

- $t \stackrel{M,\alpha}{\Longrightarrow}_L t'$, where $bn(\alpha) \cap n(M) = \emptyset$, implies $t \stackrel{M,\alpha}{\longrightarrow} t'$;
- Each t_i is in full standard form.

$\square$

Lemma 3.10 *(Absorption) If $t \stackrel{M,\alpha}{\Longrightarrow}_L t'$ with $bn(\alpha) \cap n(M) = \emptyset$ then $\vdash_w t = t + M\alpha.t'$.*

Proof: By induction on why $t \stackrel{M,\alpha}{\Longrightarrow}_L t'$.

- $t \stackrel{M,\alpha}{\longrightarrow} t'$. Apply induction on why $t \stackrel{M,\alpha}{\longrightarrow} t'$.

– $\alpha.t' \stackrel{true,\alpha}{\longrightarrow} t'$. Then $\vdash_w t = t + \alpha.t'$ by S2.

– $[x = y]t'' \stackrel{[x=y]N,\alpha}{\longrightarrow} t'$ because $t'' \stackrel{N,\alpha}{\longrightarrow} t'$. By induction $\vdash_w t'' = t'' + N\alpha.t'$. Hence

$$\begin{array}{rcl} \vdash_w [x = y]t'' & = & [x = y](t'' + N\alpha.t') \\ & = & [x = y]t'' + [x = y]N\alpha.t' \end{array}$$

– The other cases are similar.

- $t \stackrel{M_1,\alpha}{\longrightarrow} t'' \stackrel{M_2,\tau}{\Longrightarrow}_L t'$ with $M \equiv M_1M_2$. By induction $\vdash_w t = t + M_1\alpha.t''$ and $\vdash_w t'' = t'' + M_2\tau.t'$. So

$$\begin{array}{rcl} \vdash_w t & = & t + M_1\alpha.t'' \\ & = & t + M_1\alpha.(t'' + M_2\tau.t') \\ & \stackrel{3.8}{=} & t + M_1(\alpha.(t'' + M_2\tau.t') + M_2\alpha.t') \\ & = & t + M_1\alpha.(t'' + M_2\tau.t') + M_1M_2\alpha.t' \\ & = & t + M_1M_2\alpha.t' \end{array}$$

- $t \stackrel{M',\tau}{\Longrightarrow}_L t'' \stackrel{M_2,\alpha}{\longrightarrow} t'$ with $M \equiv M_1M_2$. By induction $\vdash_w t = t + M_1\tau.t''$ and $\vdash_w t'' = t'' + M_2\alpha.t'$. So

$$\begin{array}{rcl} \vdash_w t & = & t + M_1\tau.(t'' + M_2\alpha.t') \\ & \stackrel{T2}{=} & t + M_1(\tau.(t'' + M_2\alpha.t') + t'' + M_2\alpha.t') \\ & \stackrel{S2}{=} & t + M_1(\tau.(t'' + M_2\alpha.t') + t'' + M_2\alpha.t' + M_2\alpha.t') \\ & \stackrel{T2}{=} & t + M_1(\tau.(t'' + M_2\alpha.t') + M_2\alpha.t') \\ & = & t + M_1\tau.(t'' + M_2\alpha.t') + M_1M_2\alpha.t' \\ & = & t + M\alpha.t' \end{array}$$

□

Lemma 3.11 *For any term t there is a full standard term t' such that $\vdash_w t = t'$ and $\mid t' \mid \leq \mid t \mid$.*

Proof: By Lemma 3.7 we may assume t is in standard form. Apply induction on the height of t. For the non-trivial case when t is not $\mathbf{0}$, let $t \equiv \sum_{i\in I} M_it_i$ and consider the term $t' \equiv \sum\{\, M\alpha.u \mid t \stackrel{M,\alpha}{\Longrightarrow}_L u \,\}$. t' has height no greater than t, and by Lemma 3.10 $\vdash_w t = t'$. □

The following proposition relates observation congruence to weak bisimulation. It will play an important rôle in the proof of the completeness result.

Proposition 3.12 *Suppose $D \in MC_{fn(t,u)}$. Then $t \approx_L^D u$ if and only if either $t \simeq_L^D u$, or $\tau.t \simeq_L^D u$, or $t \simeq_L^D \tau.u$.*

Proof: The "if" part is trivial. For the "only if" part, assuming $t \approx_L^D u$ then

- If $t \xrightarrow{M,\tau} t'$ with $D \Rightarrow M$, then there exists $u \overset{N,\hat{\tau}}{\Longrightarrow}_L u'$ s.t. $D \Rightarrow N$ and $t' \approx_L^D u'$.
- If $u \xrightarrow{N,\tau} u'$ with $D \Rightarrow N$, then there exists $t \overset{M,\hat{\tau}}{\Longrightarrow}_L t'$ s.t. $D \Rightarrow M$ and $t' \approx_L^D u'$.

If for some t', $t \xrightarrow{M,\tau} t' \approx_L^D u$, then $t \simeq_L^D \tau.u$. If for some u', $u \xrightarrow{N,\tau} u' \approx_L^D t$, then $\tau.t \simeq_L u$. Otherwise, assuming both cases do not hold, then we can show $t \simeq_L^D u$ as follows:

Let $t \xrightarrow{M,\alpha} t'$ with $D \Rightarrow M$, then there exists $u \overset{N,\hat{\alpha}}{\Longrightarrow}_L u'$ s.t. $D \Rightarrow N$, $\alpha =^D \beta$ and $t' \approx_L^D u'$. Now by the assumption u' can not be u when α is τ, so $u \overset{N,\alpha}{\Longrightarrow}_L u'$ as required. By symmetry $t \simeq_L^D u$. □

The following technical lemma will be useful for the completeness proof:

Lemma 3.13 *If $t \overset{M,\hat{\tau}}{\Longrightarrow}_L t'$, $C \Rightarrow M$ and $\vdash_w C \rhd \tau.t' = \tau.u$, then $\vdash_w C \rhd \tau.t = \tau.t + \tau.u$.*

Proof: If $t \equiv t'$ then $\vdash_w C \rhd \tau.t = \tau.u$, hence $\vdash_w C \rhd \tau.t = \tau.t + \tau.u$ by S2.

Otherwise, by Lemma 3.10 $\vdash_w t = t + M\tau.t'$. Hence

$$\begin{array}{rcl} \vdash_w C \rhd \tau.t & = & \tau.(t + M\tau.t') \\ & \overset{3.8}{=} & \tau.(t + M\tau.t') + M\tau.t' \\ & = & \tau.t + M\tau.t' \\ & = & \tau.t + M\tau.u \\ & \overset{3.1}{=} & \tau.t + \tau.u \end{array}$$

□

We now embark on proving the main result of this paper.

Theorem 3.14 *(Completeness of $\vdash_w$) If $t \simeq_L^C u$ then $\vdash_w C \rhd t = u$.*

Proof: By Lemma 3.7 we may assume t, u are standard forms: $t \equiv \sum_i M_i\alpha_i.t_i$, $u \equiv \sum_j N_j\beta_j.u_j$. We may further assume that bound actions in α_i, β_j use the same bound name $z \notin fn(t,u)$. The proof is by induction on the joint height of t and u. By Proposition 3.2 we need only to show $\vdash_w D \rhd t = u$ for each $D \in MCE_{fn(t,u)}(C)$, and for this it is sufficient to show $\vdash_w D \rhd t = t + N_j\beta_j.u_j$ for each j.

If $D \not\Rightarrow N_j$ then, since D is maximally consistent, $D \cup N_j \Rightarrow \mathit{false}$. So by Proposition 3.1, $\vdash_w D \rhd N_j\beta_j.u_j = \mathbf{0}$. Hence $\vdash_w D \rhd t = t + N_j\beta_j.u_j$ by S1.

Now assume $D \Rightarrow N_j$ (hence $D \in MCE_{fn(t,u)}(C \cup N_j)$), and consider four cases according to the type of β_j.

- $\beta_j \equiv \tau$. Then $u \xrightarrow{N_j,\tau} u_j$. Since $t \simeq_L^C u$ and t is a full normal from, $t \xRightarrow{M_i,\tau}_L t_i$ for some i s.t. $D \Rightarrow M_i$ and $t_i \approx_L^D u_j$. We can not invoke the induction hypothesis here because we only have $t_i \approx_L^D u_j$. But by Proposition 3.12 we have either $t_i \simeq_L^D u_j$, or $\tau.t_i \simeq_L^D u_j$, or $t_i \simeq_L^D \tau.u_j$. In the first case we can apply induction to get $\vdash_w D \triangleright t_i = u_j$, and by TAU, $\vdash_w D \triangleright \tau.t_i = \tau.u_j$. In the second case, by Lemma 3.11 there is a full normal form t' with height no greater than t s.t. $\vdash_w \tau.t_i = t'$. By the soundness of $\vdash_w$, $t' \simeq_L^{true} \tau.t_i$, hence $t' \simeq_L^D u_j$. By induction $\vdash_w D \triangleright t' = u_j$, so $\vdash_w D \triangleright \tau.t_i = u_j$. Apply TAU and T1 we get $\vdash_w D \triangleright \tau.t_i = \tau.u_j$. In the third case we can similarly get $\vdash_w D \triangleright \tau.t_i = \tau.u_j$. Since $D \Rightarrow M_i$ and $D \Rightarrow N_j$, by Proposition 3.1 $\vdash_w D \triangleright M_i\tau.t_i = N_j\tau.u_j$. Finally by S2, $\vdash_w D \triangleright t = t + N_j\tau.u_j$.

- $\beta_j \equiv b(z)$, $z \notin fn(t,u,C)$. Then $u \xrightarrow{N_j,b(z)} u_j$. Since $t \simeq_L^C u$ and t is a full normal from, $t \xrightarrow{M_i,a_i(z)} t_i$ for some i s.t. $D \Rightarrow M_i$, $D \Rightarrow a_i = b$, and for each $D' \in MCE_{fn(t,u)\cup\{z\}}(D)$ there is $t_i \xRightarrow{M',\hat{\tau}}_L t'$ s.t. $D' \Rightarrow M'$ and $t' \approx_L^{D'} u_j$. Using Proposition 3.12, by the same argument as in the τ case above, we can show $\vdash_w D' \triangleright \tau.t' = \tau.u_j$. Note that here t' depends on D'. But by Lemma 3.13, $\vdash_w D' \triangleright \tau.t_i = \tau.t_i + \tau.u_j$. Now we can use Proposition 3.2 to obtain $\vdash_w D \triangleright \tau.t_i = \tau.t_i + \tau.u_j$. By L-INPUT, $\vdash_w D \triangleright a_i(z).\tau.t_i = a_i(z).(\tau.t_i + \tau.u_j)$. So

$$\begin{aligned}
\vdash_w D \triangleright a_i(z).t_i &\overset{T1}{=} a_i(z).\tau.t_i \\
&= a_i(z).(\tau.t_i + \tau.u_j) \\
&\overset{T3}{=} a_i(z).(\tau.t_i + \tau.u_j) + a_i(z).u_j \\
&= a_i(z).t_i + a_i(z).u_j \\
&= a_i(z).t_i + b(z).u_j
\end{aligned}$$

  Since $D \Rightarrow M_i$ and $D \Rightarrow N_j$, by Proposition 3.1, $\vdash_w D \triangleright M_i a_i(z).t_i = M_i a_i(z).t_i + N_j b(z).u_j$. Hence $\vdash_w D \triangleright t = t + N_j b(z).u_j$ by S2.

- $\beta_j \equiv \overline{b}(z)$. Then $u \xrightarrow{N_j,\overline{b}(z)} u_j$, so $t \xRightarrow{M_i,\overline{a}_i(z)}_L t_i \xRightarrow{M',\hat{\tau}}_L t'$ for some i s.t. $D \Rightarrow M_iM'$, $D \Rightarrow a_i = b$ and $t' \approx_L^{D'} u_j$, where $D' \equiv D \cup \{\, z \neq y \mid y \in fn(\overline{a}_i(z).t_i, \overline{b}(z).u_j) \,\}$. Note that $D' \in MC_{fn(t_i,u_j)}$, so by the same argument as in the τ case, $\vdash_w D' \triangleright \tau.t' = \tau.u_j$. By Lemma 3.13, $\vdash_w D' \triangleright \tau.t_i = \tau.t_i + \tau.u_j$. By Proposition 3.5, $\vdash_w D \triangleright \overline{a}_i(z).\tau.t_i = \overline{a}_i(z).(\tau.t_i + \tau.u_j)$. The rest is the same as the previous case.

- $\beta_j \equiv \overline{b}y$. Then $u \xrightarrow{N_j,\overline{b}y} u_j$, so $t \xRightarrow{M_i,\overline{a}_i x}_L t_i$ for some i s.t. $D \Rightarrow M_i$, $D \Rightarrow \overline{a}_i = \overline{b}, D \Rightarrow x = y$ and $t_i \approx_L^D u_j$. By the same argument as in the τ case, $\vdash_w D \triangleright \tau.t_i = \tau.u_j$. By OUTPUT and T1, $\vdash_w D \triangleright \overline{a}_i x.t_i = \overline{b}y.u_j$. By Proposition 3.1, $\vdash_w D \triangleright M_i\overline{a}_i x.t_i = N_j\overline{b}y.u_j$. Hence $\vdash_w D \triangleright t = t + N_j\overline{b}y.u_j$ by S2.

This completes the proof. $\square$

4 The Early Case

To define (concrete) early bisimulation, instead of bound input action, we shall use *free input* action of the form ax ([10, 14]). So we add the following rule to Figure 1:

$$\text{free-input}\ \frac{}{a(x).t \xrightarrow{ay} t[y/x]}$$

The definition of (concrete) weak transition relations, $\Longrightarrow_e$, remain the same as Definition 2.7. Note that the condition "If α does not have the form $a(x)$" in the last clause can be removed because no action has such form now.

Free input is not needed for early symbolic bisimulation. To define early symbolic weak transition relations, $\Longrightarrow_E$, we simply remove the "non-input action" restriction to the last rule in Definition 2.11, with the other rules unchanged. This is because early bisimulation allows a single input move from one process to be matched by *several* such moves from the other, so there is no danger for input moves to absorb τ moves *after* them. It is possible to use the same weak transition relations as in the late case, but removing this restriction will make the definition of early weak symbolic bisimulation simpler.

The following lemma modifies Lemma 2.6 in the case of input moves.

Lemma 4.1 *1. If $t\sigma \xrightarrow{ay} p$ then there exist M, t', b, x s.t. $x \notin n(\sigma)$, $a = b\sigma$, $\sigma \models M$, $t'\sigma[y/x] \equiv p$ and $t \xrightarrow{M,b(x)} t'$.*

2. If $t \xrightarrow{M,a(x)} t'$ then for any $\sigma \models M$ with $x \notin n(\sigma)$, for any y, there exist $p \equiv t'\sigma[y/x]$ and $b = a\sigma$ s.t. $t\sigma \xrightarrow{by} p$.

Proof: *1.* By induction on why $t\sigma \xrightarrow{ay} p$.

- $(b(z).t'')\sigma \xrightarrow{ay} p$. Then $b\sigma = a$. Take $x \notin n(\sigma) \cup fn(b(z).t'')$ and let $t' = t''[x/z]$. We have $b(z).t'' \equiv b(x).t'$ and $(b(z).t'')\sigma \equiv a(x).t'\sigma$, so $p \equiv t'\sigma[y/x]$. Since $b(x).t' \xrightarrow{true,b(x)} t'$ and symbolic transitions are defined up-to α-equivalence, we are done.

- $([z = z']t'')\sigma \xrightarrow{ay} p$ because $z\sigma = z'\sigma$ and $t''\sigma \xrightarrow{ay} p$. By induction there exist M, t', b, x s.t. $x \notin n(\sigma)$, $a = b\sigma$, $\sigma \models M$, $t'\sigma[y/x] \equiv p$ and $t'' \xrightarrow{M,b(x)} t'$. So $[z = z']t'' \xrightarrow{M[z=z'],b(x)} t'$ and we have $\sigma \models M[z = z']$.

- The other cases are similar.

2. By induction on why $t \xrightarrow{M,a(x)} t'$.

- $a(x).t' \xrightarrow{true,a(x)} t'$. Then for any σ with $x \notin n(\sigma)$ and any y, $(a(x).t')\sigma \equiv b(x), t'\sigma \xrightarrow{by} t'\sigma[y/x]$ where $b = a\sigma$.

- $[z=z']t \overset{N[z=z'],a(x)}{\longrightarrow} t'$ because $t \overset{N,a(x)}{\longrightarrow} t'$. Let $\sigma \models N[z=z']$ with $x \notin n(\sigma)$. Then $\sigma \models N$, so by induction for any y there exist $p \equiv t'\sigma[y/x]$ and $b = a\sigma$ s.t. $t\sigma \xrightarrow{by} p$. Since $\sigma \models z = z'$ we have $z\sigma = z'\sigma$. Hence $([z=z']t)\sigma \equiv [z\sigma = z'\sigma]\sigma t\sigma \xrightarrow{by} p$.
- The other cases are similar.

□

Definition 4.2 A symmetric relation R is an early weak ground bisimulation if $(p,q) \in R$ implies whenever $p \xrightarrow{\alpha} p'$ then $q \overset{\hat{\alpha}}{\Longrightarrow}_e q'$ for some q' and $(p',q') \in R$.
Write $p \dot{\approx}_e q$ if there exists an early weak ground bisimulation R s.t. $(p,q) \in R$.
t is early weak bisimular to u, written $t \approx_e u$, if $t\sigma \dot{\approx}_e u\sigma$ for every substitution σ. □

Definition 4.3 Two terms t and u are early ground observation congruent, written $p \dot{\simeq}_e q$, if whenever $p \xrightarrow{\alpha} p'$ then $q \overset{\alpha}{\Longrightarrow}_e q'$ for some q' and $p' \approx_e q'$. And similarly for u.
t is early observation congruent to u, written $t \simeq_e u$, if $t\sigma \dot{\simeq}_e u\sigma$ for every substitution σ. □

Lemma 4.4 *1. If $t \overset{M,\alpha}{\Longrightarrow}_E u$ then $n(M) \subseteq fn(t) \cup bn(\alpha)$.*

2. If $t \overset{M,a(x)}{\Longrightarrow}_E t'$ then there exist $M_1,\ M_2$ with $x \notin n(M_1),\ M = M_1M_2$ s.t. $t \overset{M_1,a(x)}{\Longrightarrow}_E \overset{M_2}{\Longrightarrow}_E t'$.

Proof: By transition induction. □

Lemma 4.5 *1. (a) If $t\sigma \overset{ay}{\Longrightarrow}_e p$ then there exist $b,\ x,\ M,\ N,\ t'$ with $a = b\sigma,\ x \notin n(\sigma),\ \sigma \models M,\ \sigma[y/x] \models N,\ t'\sigma[y/x] \equiv p$ s.t. $t \overset{M,b(x)}{\Longrightarrow}_E \overset{N}{\Longrightarrow}_E t'$.*

(b) If $t \overset{M,a(x)}{\Longrightarrow}_E \overset{N}{\Longrightarrow}_E t'$ then for any $\sigma \models M$ with $x \notin n(\sigma)$ there exist $y,\ b,\ p$ with $b = a\sigma,\ p \equiv t'\sigma[y/x],\ \sigma[y/x] \models N$ s.t. $t\sigma \overset{by}{\Longrightarrow}_e p$.

2. For the other actions

(a) If $t\sigma \overset{\alpha}{\Longrightarrow}_e p$ where $bn(\alpha) \cap (fn(t) \cup n(\sigma)) = \emptyset$, then there exist $M,\ \beta,\ t'$ s.t. $\sigma \models M,\ \alpha = \beta\sigma,\ p \equiv t'\sigma$ and $t \overset{M,\beta}{\Longrightarrow}_E t'$.

(b) If $t \overset{M,\alpha}{\Longrightarrow}_E t'$ then for any $\sigma \models M$ with $bn(\alpha) \cap (fn(t) \cup n(\sigma)) = \emptyset$, $t\sigma \overset{\alpha\sigma}{\Longrightarrow}_e p \equiv t'\sigma$.

Proof: Similar to that of Lemma 2.12. □

Definition 4.6 A condition indexed family of symmetric relations $\mathcal{S} = \{S^C\}$ is an early weak symbolic bisimulation if $(t, u) \in S^C$, where $n(C) \subseteq fn(t, u)$, implies

whenever $t \xrightarrow{M,\alpha} t'$ with $bn(\alpha) \cap fn(t, u) = \emptyset$, then for each $D \in MCE_V(C \cup M)$ there is a $u \overset{N,\hat{\beta}}{\Longrightarrow}_E u'$ such that $D \Rightarrow N$, $\alpha =^D \beta$, and $(t', u') \in S^{D'}$, where

$$V = \begin{cases} fn(t,u) \cup \{x\} & \textit{if } \alpha \equiv a(x) \\ fn(t,u) & \textit{otherwise} \end{cases}$$
$$D' = \begin{cases} D \cup \{\, x \neq y \mid y \in fn(\alpha.t', \beta.u') \,\} & \textit{if } \alpha \equiv \overline{a}(x) \\ D & \textit{otherwise} \end{cases}$$

Let $\approx_E$ be the largest early weak symbolic bisimulation. □

Compared with the definition of late symbolic bisimulation, the only difference is concerning input move: now the maximally consistent decomposition of the indexing condition C is allowed to be over the name space including the input name. As a consequence, the two maximally consistent decompositions used in Definition 2.13 are combined into one.

Theorem 4.7 *$t \approx_E^C u$ if and only if $t\sigma \approx_e u\sigma$ for every $\sigma \models C$.*

Proof: The proof differs from that of Theorem 2.16 only in the case of input action in the proof of Proposition 2.15, which we show below.

($\Longrightarrow$) Define

$$R = \{\, (t\sigma, u\sigma) \mid t \approx_L^C u \text{ for some } C \in MC_{fn(t,u)}, \text{ and } \sigma \models C \,\}$$

We show R is a weak ground bisimulation.

Let $(t\sigma, u\sigma) \in R$ and $t\sigma \xrightarrow{ay} p$. We need to find a matching transition from $u\sigma$. By Lemma 4.1 there exist a', x, M, t' with $x \notin n(\sigma)$, $\sigma \models M$, $a = a'\sigma$, $t'\sigma[y/x] \equiv p$ s.t. $t \xrightarrow{M,a'(x)} t'$. By Lemma 2.5 we may assume $x \notin fn(t, u)$. Since $\sigma \models C$, $\sigma \models M$, $n(M) \subseteq fn(t, u)$ and C is maximally consistent on $fn(t, u)$, $C \models M$ by Lemma 2.3. Let

$$D = C \cup \{\, x = z \mid z \in fn(t,u), y\sigma = z\sigma \,\} \cup \{\, x \neq z \mid z \in fn(t,u), y\sigma \neq z\sigma \,\}$$

Then $D \in MCE_{fn(t,u)\cup\{x\}}(C)$ and $\sigma[y/x] \models D$. So by Definition 4.6 and Lemma 4.4 there exist $u \overset{N_1,b'(x)}{\Longrightarrow}_E \overset{N_2}{\Longrightarrow}_E u'$ s.t. $x \notin n(N_1)$, $D \Rightarrow N_1N_2$, $a' =^D b'$ and $t' \approx_L^D u'$.

Now $\sigma[y/x] \models N_1N_2$ and $x \notin n(N_1)$, hence $\sigma \models N_1$, $\sigma[y/x] \models N_2$. Apply Lemma 4.5 we get $u\sigma \overset{by}{\Longrightarrow}_e q \equiv u'\sigma[y/x]$ with $b = b'\sigma$. From $a = a'\sigma$, $a' =^D b'$, $b = b'\sigma, \sigma[y/x] \models D$, $\{a, b\} \subseteq fn(t, u)$ and $x \notin fn(t, u)$ we infer $a = a'\sigma[y/x] = b'\sigma[y/x] = b$.

By the definition of R, $(t'\sigma[y/x], u'\sigma[y/x]) \in R$. Since $p \equiv t'\sigma[y/x]$, $q \equiv u'\sigma[y/x]$, we conclude $(p, q) \in R$.

($\Longleftarrow$) Define

$$S^C = \{\, (t, u) \mid C \in MC_{fn(t,u)},\ t\sigma \dot{\approx}_l u\sigma \text{ for some } \ \sigma \models C \,\}$$

and $\mathcal{S} = \{S^C\}$. We show $\mathcal{S}$ is a weak symbolic bisimulation.

Assume $(t, u) \in S^C$, *i.e.* $C \in MC_{fn(t,u)}$, $t\sigma \dot{\approx}_l u\sigma$ for some $\sigma \models C$. Let $t \xrightarrow{M,a(x)} t'$ with $C \Rightarrow M$ and $x \notin fn(t, u) \cup n(\sigma)$. Let $\mathcal{E}$ be the set of equivalence classes on $fn(t, u)$ determined by σ, *i.e.* x, y are in the same class if and only if $\sigma \models x = y$. Let also $D \in MCE_{fn(t,u)\cup\{x\}}(C)$. Since $C \in MC_{fn(t,u)}$ and $\sigma \models C$, D is equivalent to

$$C \cup \{\, x = z \mid z \in [w] \,\} \cup \{\, x \neq z \mid z \in fn(t, u) - [w] \,\}$$

for some $[w] \in \mathcal{E}$. Pick a y from $[w]$. Then $\sigma[y/x] \models D$. By Lemma 4.1, $t\sigma \xrightarrow{a'y} p \equiv t'\sigma[y/x]$ with $a' = a\sigma$. So there exists $u\sigma \overset{a'y}{\Longrightarrow}_e q$ and $p \dot{\approx}_l q$. Apply Lemma 4.5 we get N_1, N_2, u', b s.t. $x \notin n(N_1)$, $\sigma \models N_1$, $\sigma[y/x] \models N_2$, $u'\sigma[y/x] \equiv q$, $b\sigma = a'$ and $u \overset{N_1,b(x)}{\Longrightarrow}_E \overset{N_2}{\Longrightarrow}_E u'$.

From $b\sigma = a' = a\sigma$, $\{a, b\} \subseteq fn(t, u)$ and $x \notin fn(t, u)$ it follows $\sigma[y/x] \models a = b$. Since $\sigma[y/x] \models D$ and D is maximally consistent on $fn(t, u) \cup \{x\}$, $a =^D b$ by Lemma 2.3. Finally, since $t'\sigma[y/x] \dot{\approx}_l u'\sigma[y/x]$ and $\sigma[y/x] \models D$, we obtain $(t', u') \in S^D$. □

Early symbolic observation congruence is defined on top of $\approx_E$ as usual:

Definition 4.8 Two terms t, u are *early symbolic observation congruent* with respect to C, written $t \simeq_E^C u$, if

whenever $t \xrightarrow{M,\alpha} t'$ with $bn(\alpha) \cap fn(t, u) = \emptyset$, then for each $D \in MCE_V(C \cup M)$ there is a $u \overset{N,\beta}{\Longrightarrow}_E u'$ such that $D \Rightarrow N$, $\alpha =^D \beta$, and $t' \approx_E^{D'} u'$, where

$$V = \begin{cases} fn(t, u) \cup \{x\} & \textit{if } \ \alpha \equiv a(x) \\ fn(t, u) & \textit{otherwise} \end{cases}$$

$$D' = \begin{cases} D \cup \{\, x \neq y \mid y \in fn(\alpha.t', \beta.u') \,\} & \textit{if } \ \alpha \equiv \overline{a}(x) \\ D & \textit{otherwise} \end{cases}$$

And similarly for u. □

Theorem 4.9 *$t \simeq_E^C u$ if and only if $t\sigma \simeq_e u\sigma$ for every $\sigma \models C$.*

Proof: Similar to that of Proposition 4.7. □

The proof systems for early weak congruence can be obtained by replacing the L-INPUT rule in Figure 3 with the following

$$\text{E-INPUT}\ \frac{C \rhd \sum_{i\in I} \tau.t_i = \sum_{j\in J} \tau.u_j}{C \rhd \sum_{i\in I} a_i(x).t_i = \sum_{j\in J} b_j(x).u_j} \quad \begin{array}{l} C \Rightarrow a_i = b_j,\ i \in I,\ j \in J \\ x \notin n(C) \end{array}$$

It is worth pointing out that with this new rule axiom T3 can be weakened to let α range over output actions only, as the cases for τ and input actions can now be derived easily.

Let us write $\vdash_{wE} C \rhd t = u$ to mean $C \rhd t = u$ can be derived from this new proof system.

The E-INPUT rule has a generalised form:

Proposition 4.10 *Suppose $x \notin fn(C)$, $C \Rightarrow a_i = b_j,\ i \in I,\ j \in J$. If*

$$\vdash_{wE} C \rhd \sum_{i\in I} M_i\tau.t_i = \sum_{j\in J} N_j\tau.u_j$$

then

$$\vdash_{wE} C \rhd \sum_{i\in I} M_i a_i(x).t = \sum_{j\in J} N_j b_j(x).u$$

Proof: By Proposition 3.2, we only need to show $\vdash_{wE} D \rhd \sum_{i\in I} M_i a_i(x).t = \sum_{j\in J} N_j b_j(x).u$ for each $D \in MCE_V(C)$, where V is the set of free names appearing in the conclusion. Since D is maximally consistent on V, for each i either $C \Rightarrow M_i$ or $C \cup M_i = false$. Hence by Proposition 3.1

$$\vdash_{wE} D \rhd \sum_{i\in I} M_i\tau.t_i = \sum_{i\in I'} M_i\tau.t_i$$

$$\vdash_{wE} D \rhd \sum_{i\in I} M_i a_i(x).t_i = \sum_{i\in I'} M_i a_i(x).t_i$$

where $I' = \{\, i \in I \mid D \Rightarrow M_i \,\}$. Similarly,

$$\vdash_{wE} D \rhd \sum_{j\in J} N_j\tau.u_j = \sum_{j\in J'} N_j\tau.u_j$$

$$\vdash_{wE} D \rhd \sum_{j\in J} N_j b_j(x).u_j = \sum_{j\in J'} N_j b_j(x).u_j$$

where $J' = \{\, j \in J \mid D \Rightarrow N_j \,\}$. Hence, by Proposition 3.1 from the assumption we can derive

$$\vdash_{wE} D \rhd \sum_{i\in I'} M_i\tau.t_i = \sum_{j\in J'} N_j\tau.u_j$$

By E-INPUT

$$\vdash_{wE} D \rhd \sum_{i\in I'} M_i a_i(x).t = \sum_{j\in J'} N_j b_j(x).u$$

Therefore

$$\vdash_{wE} D \triangleright \sum_{i \in I} M_i a_i(x).t = \sum_{j \in J} N_j b_j(x).u$$

$\square$

All the lemmas in the previous section hold for $\vdash_{wE}$ as well. And we have the following counterpart of Theorem 3.14:

Theorem 4.11 *(Completeness of $\vdash_{wE}$) If $t \simeq_E^C u$ then $\vdash_{wE} C \triangleright t = u$.*

Proof: The proof of this theorem differs from that of Theorem 3.14 only in the case for input action, and we examine it below.

Case $\beta_j \equiv b(z)$, $z \notin fn(t, u, C)$. Let $I' = \{\, i \in I \mid \alpha_i \equiv a_i(z),\ D \Rightarrow a_i = b \,\}$, $t' = \sum_{i \in I'} M_i a_i(z).t_i$, and $t^\tau = \sum_{i \in I'} M_i \tau.t_i$. We only need to show $\vdash_{wE} D \triangleright t' = t' + N_j b(z).u_j$, and, since $z \notin n(D)$, by Proposition 4.10 this can be reduced to $\vdash_{wE} D \triangleright t^\tau = t^\tau + N_j \tau.u_j$.

Now $u \stackrel{N_j, b(z)}{\longrightarrow} u_j$ and t is a full standard form, therefore for each $D' \in MCE_{fn(t,u,C) \cup \{z\}}(D)$, $t \stackrel{M_i, a_i(z)}{\longrightarrow} t_i \stackrel{M', \hat{\tau}}{\Longrightarrow}_E t'$ for some i s.t. $z \notin n(M_i)$, $D' \Rightarrow M_i M'$ and $t' \approx_E^{D'} u_j$. By the same argument as in the τ case, we can show $\vdash_{wE} D' \triangleright \tau.t' = \tau.u_j$. By Lemma 3.13, $\vdash_{wE} D' \triangleright \tau.t_i = \tau.t_i + \tau.u_j$. By Proposition 3.2, $\vdash_{wE} D \triangleright \tau.t_i = \tau.t_i + \tau.u_j$. Since $D \Rightarrow M_i$ and $D \Rightarrow N_j$, by Proposition 3.1, $\vdash_{wE} D \triangleright M_i \tau.t_i = M_i \tau.t_i + N_j \tau.u_j$. Hence $\vdash_{wE} D \triangleright t^\tau = t^\tau + N_j \tau.u_j$. $\square$

5 Other Operators

5.1 Parallel Composition

To deal with parallel composition all we need is a suitable form of expansion law which is presented in Figure 5 (taken from [12]). With this law it is standard that any finite term containing parallel operator can be equated to a finite term without it, hence the normal form lemma still holds, as does the completeness theorem.

5.2 Mismatch

Mismatch, *i.e.* testing inequality between names, is not included in the original π-calculus. Some later publications, notably [4], [1], [12] and [2], extended the calculus with mismatch in order to give axiomatisation for testing or bisimulation equivalences.

To include mismatch into the language we first extend the operational semantics by including the following two rules: "mismatch" and "Mismatch" in Figure 1 and Figure 2, Let $t \equiv \sum_i M_i\alpha_i.t_i$ and $u \equiv \sum_j N_j\beta_j.u_j$ with $bn(\alpha_i) \cap fn(u) = bn(\beta_j) \cap fn(t) = \emptyset$. Then

$$t \mid u = \sum_i M_i\alpha_i.(t_i \mid u) \ + \ \sum_j N_j\beta_j.(t \mid u_j) \ + \sum_{\alpha_i \ opp \ \beta_j} M_iN_j[a_i = b_j]\tau.v_{ij}$$

where α_i *opp* β_j and v_{ij} are defined as follows

1. $\alpha_i \equiv a(x), \beta_j \equiv \overline{b}y$; then $v_{ij} \equiv t_i[y/x] \mid u_j$;
2. The converse of the above clause;
3. $\alpha_i \equiv a(x), \beta_j \equiv \overline{b}(y)$; then $v_{ij} \equiv (z)(t_i[z/x] \mid u_j[z/y])$ with $z \notin fn(t, u)$;
4. The converse of the above clause.

Figure 5: The Expansion Law

respectively (now M ranges over conditions)

$$\text{mismatch}\ \frac{t \stackrel{\alpha}{\longrightarrow} t'}{[x \neq y]t \stackrel{\alpha}{\longrightarrow} t'}\ x \neq y \qquad \text{Mismatch}\ \frac{t \stackrel{M,\alpha}{\longrightarrow} t'}{[x \neq y]t \stackrel{M[x\neq y],\alpha}{\longrightarrow} t'}$$

For the symbolic operational semantics we also need to modify the rules Res and Open:

$$\text{Res}\ \frac{P \stackrel{M,\alpha}{\longrightarrow} P'}{(x)P \stackrel{\nu_x M,\alpha}{\longrightarrow} (x)P'}\ x \notin n(\alpha) \qquad \text{Open}\ \frac{P \stackrel{M,\overline{a}x}{\longrightarrow} P'}{(x)P \stackrel{\nu_x M,\overline{a}(x)}{\longrightarrow} P'}\ x \neq a$$

where ν_x is an operation on conditions defined thus ([2]):

$$\begin{array}{lll} \nu_x true = true & \nu_x[x = x] = true & \nu_x[x = y] = false \\ \nu_x[y = z] = [y = z] & \nu_x\neg M = \neg\nu_x M & \nu_x(M \wedge N) = \nu_x M \wedge \nu_x N \end{array}$$

The inference rule for mismatch is dual to that for match:

$$\text{MISMATCH}\ \frac{C \cup \{x \neq y\} \rhd t = u \quad C \cup \{x = y\} \rhd \mathbf{0} = u}{C \rhd [x \neq y]t = u}$$

With this rule Proposition 3.1 can be generalised to allow M to be an arbitrary condition, not just a match.

Now, in a normal form $\sum_i M_i\alpha_i.t_i$, M_i may contain inequality tests as well as equality tests. This does not affect the proofs of the normal form lemma and the completeness theorem.

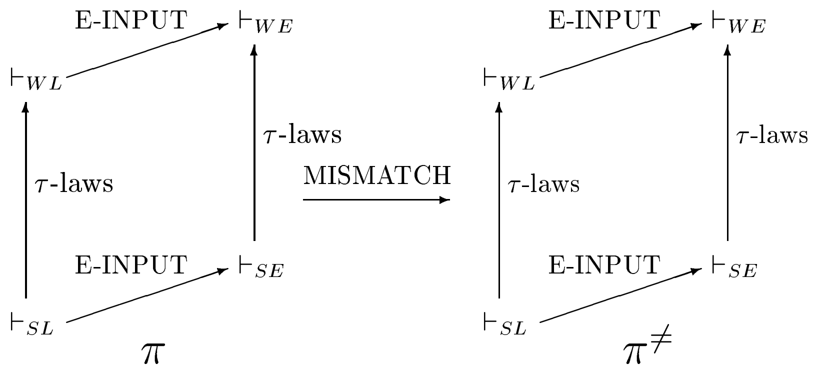

Figure 6: The Relationship between Proof Systems

6 Conclusions And Related Work

We have presented complete proof systems for late and early weak observation equivalences for the π-calculus. The inference system for the early equivalence is obtained by adding the E-INPUT rule to the inference system for the late equivalence. To enrich the language with the *mismatch* construction we only need to extend the inference systems with a single rule, namely MISMATCH. In each case the proof system for weak equivalence is obtained by adding the standard τ-laws to the one for the strong equivalence. Thus in all we have eight proof systems. The relationship between them are pictured in Figure 6, where arrows stand for "extension" and $\pi^{\neq}$ is the version of π-calculus enriched with mismatch.

These results verify a conjecture posed in [10, 12, 2]: the three τ-laws are sufficient for the transition from strong to weak bisimulation equivalences for the π-calculus, as in the case of *CCS*.

The present work relies heavily on the *symbolic* technique, inherited from our previous work on general message-passing process calculi [5, 6]. As the process calculi considered there take the languages for data and boolean expressions as parameters, the completeness results in those papers are relative to data reasoning. Here, by exploiting the specific features presented in the π-calculus, there is no need to refer to an "oracle" for the data domain and the completeness results are no longer relative to reasoning about data.

We have succeeded in formulating complete proof systems for the π-calculus without the mismatch operator, by carefully employing it only in the meta-language, i.e. in the condition part of judgments. In Parrow and Sangiorgi's equational axiomatisation for late and early strong bisimulations ([12]), mismatch plays an essential rôle in three places: the definition of normal form, the axiom for early equivalence, and the equational characterisation of the restriction operator. [2] also proposed symbolic style proof systems for both early and late strong bisimulations in π-calculus with mismatch. It is instructive to compare the notions of normal form used in [12], [2] and the present paper.

They all have the general form:

$$\sum_{i\in I} \phi_i \alpha_i.t_i$$

In [12] it is required that ϕ_i be *complete* (i.e. maximally consistent). In [2] this is weakened to only require that ϕ_i:s be disjoint, namely $\phi_i \wedge \phi_j = \mathit{false}$ for $i \neq j$. Note that both necessitate the inclusion of the mismatch operator in the calculus. In the present paper no requirement is imposed on these ϕ_i:s: they can be either matches (in the calculus without mismatch) or conditions (when mismatch is included). Here the maximal consistence is achieved at meta-level, in the condition part of judgments. If mismatch is included in the calculus, it is not difficult to show that in our proof systems

$$\vdash C \rhd t = u \quad \text{if and only if} \quad \vdash Ct = Cu$$

So the proof system can be turned into equational axiomatisation by *internalising* conditions in the judgments as parts of process terms.

Another interesting point in [12] is the axiomatisation of early bisimulation, which is obtained by adding the following axiom to those for late:

$$\text{EA} \quad a(x).t + a(x).u = a(x).t + a(x).u + a(x).([x = y]t + [x \neq y]u)$$

where y is an arbitrary name. In the present paper we use the more general rule E-INPUT, adapted from our earlier work on message-passing process algebras. The main reason is that E-INPUT does not rely on mismatch, and one objective of this paper is to demonstrate that complete inference systems can be formulated for the π-calculus without mismatch. If mismatch is allowed then E-INPUT can be replaced by EA. A proof of this fact is provided in the Appendix.

[3] proposed an axiomatization for weak late observation equivalence for the π-calculus. In their work the calculus was extended with *environments* recording the information about which names are equal and which are private. In contrast, in the present paper such information is kept in *conditions* which have been carefully separated from the calculus itself.

[8] gives a symbolic version of distinction-indexed bisimulation for a sub-language of the π-calculus (without the *match* operator) and shows that this notion of bisimulation captures the conventional bisimulation equivalences. But no proof system is considered there.

In [13] another notion of bisimulation, called *open bisimulation*, for the π-calculus is proposed, along with an equational axiomatisation. A “symbolic” characterisation of open bisimulation is also given. To deal with the restriction operator, distinctions have to be employed as indices in the definition of open bisimulation as well as in the axiomatisation. Open bisimulation has been further studied in [14] where (concrete) early weak bisimulation is also discussed.

Acknowledgments Thanks to Matthew Hennessy for reading a draft of this paper. Detailed comments from an anonymous referee have led to several improvements.

References

[1] Michele Boreale and Rocco De Nicola (1995), Testing equivalences for mobile processes. *Information and Computation*, **120**, 279–303.

[2] Michele Boreale and Rocco De Nicola (1996), A symbolic semantics for the π-calculus. *Information and Computation*, **126**, 34–52.

[3] G. Ferrari, U. Montanari and P. Quaglia (1995), The weak late π-calculus semantics as observation equivalence. In *Proc. CONCUR'95*, Vol 962, Lecture Notes in Computer Science, Springer–Verlog.

[4] M. Hennessy (1991), A model for the π-calculus. Technical Report 8/91, CSAI, University of Sussex.

[5] M. Hennessy and H. Lin (1995), Symbolic bisimulations. *Theoretical Computer Science*, **138**, 353-389.

[6] M. Hennessy and H. Lin (1996), Proof systems for message-passing process algebras. *Formal Aspects of Computing*, **8**, 379-407.

[7] H. Lin (1994), Symbolic bisimulations and proof systems for the π-calculus. Report ISCAS-LCS-94-06, Laboratory for Computer Science, Institute of Software, Chinese Academy of Sciences.

[8] X. Liu (1994), Characterizing bisimulation congruence in the π-calculus. In *CONCUR'94*, Vol 836, Lecture Notes in Computer Science. Springer–Verlag.

[9] R. Milner (1989), "Communication and Concurrency," Prentice-Hall.

[10] R. Milner, J. Parrow and D. Walker (1992), A calculus of mobile processes, part I,II. *Information and Computation*, **100**, 1–77.

[11] R. Milner, J. Parrow and D. Walker (1993), Modal logics for mobile processes. *Theoretical Computer Science*, **114**, 149–171.

[12] J. Parrow and D. Sangiorgi (1995), Algebraic theories for name-passing calculi. *Information and Computation*, **120**, 174–197.

[13] Davide Sangiorgi (1996), A theory of bisimulation for the π-calculus. *Acta Informatica*, **33**, 69–97.

[14] D. Walker (1994), On bisimulation in the π-calculus. In *CONCUR'94*, Vol 836, Lecture Notes in Computer Science. Springer–Verlag.

7 Appendix

We mentioned in the Conclusion that if mismatch is included in the language then E-INPUT can be replaced by EA. This Appendix is devoted to proving the claim. In the sequel $\vdash_{wE}$ will refer to derivability in the proof system consisting of the inference rules in Figure 3, the equational axioms in Figure 4 and EA.

We first generalise EA.

Lemma 7.1 *Suppose $n(D) \subseteq V$, D is maximally consistent on V, $x \notin V$, $MCE_{V\cup\{x\}}(D) = \{ D_i \mid i \in I \}$ and $fn(t_i) \subseteq V$ for each $i \in I$. Then*

$$\vdash_{wE} D \triangleright \sum_{i\in I} a(x).t_i = \sum_{i\in I} a(x).t_i + \sum_{i\in I} a(x).D_i t_i$$

Proof: Let $\{[y_1], \dots, [y_k]\}$ be the set of equivalence classes on V determined by D, i.e., for any $y, y' \in V$, $D \Rightarrow y = y'$ if y, y' are in the same equivalence class and $D \Rightarrow y \neq y'$ otherwise. Then, up to logical equivalence, $MCE_{V\cup\{x\}}(D)$ contains the following elements:

$$\begin{array}{lcl}
D_1 & \equiv & D \wedge x = y_1 \\
D_2 & \equiv & D \wedge x \neq y_1 \wedge x = y_2 \\
 & & \dots \\
D_k & \equiv & D \wedge x \neq y_1 \ \dots \ \wedge x \neq y_{k-1} \wedge x = y_k \\
D_{k+1} & \equiv & D \wedge x \neq y_1 \ \dots \ \wedge x \neq y_{k-1} \wedge x \neq y_k
\end{array}$$

By repeatedly applying S2 and EA, we have (using $A =\!+\ B$ to abbreviate $A = A + B$)

$$\begin{array}{ll}
\vdash_{wE} & D \triangleright \sum_{i\in I} a(x).t_i \\
=+ & a(x).t_k + a(x).t_{k+1} \\
=+ & a(x).t_k + a(x).t_{k+1} + a(x).([x = y_k]t_k + [x \neq y_k]t_{k+1}) \\
=+ & a(x).t_{k-1} + a(x).([x = y_k]t_k + [x \neq y_k]t_{k+1}) \\
=+ & a(x).([x = y_{k-1}]t_{k-1} + [x \neq y_{y-1}]([x = y_k]t_k + [x \neq y_k]t_{k+1}) \\
 & \dots \\
=+ & a(x).([x = y_1]t_1 + [x \neq y_1][x = y_2]t_2 \ \dots \ + [x \neq y_1] \dots [x \neq y_k]t_{k+1}) \\
\stackrel{3.1}{=} & \sum_{i\in I} a(x).t_i + a(x).(D_1 t_1 + D_2 t_2 \ \dots \ + D_{k+1} t_{k+1}) \\
= & \sum_{i\in I} a(x).t_i + \sum_{i\in I} a(x).D_i t_i
\end{array}$$

$\square$

T1 also has a generalised form:

Lemma 7.2 *Suppose $n(D) \subseteq V$, $x \notin n(D)$, $MCE_{V\cup\{x\}}(D) = \{ D_i \mid i \in I \}$ and $fn(t_i) \subseteq V$ for each $i \in I$. Then*

$$\vdash_{wE} D \triangleright a(x).\sum_{i\in I} D_i t_i = a(x).\sum_{i\in I} D_i \tau.t_i$$

Proof: Since the elements of $MCE_{V\cup\{x\}}(D)$ are mutually disjoint, using Proposition 3.1 we have, for each $i \in I$,

$$\begin{array}{rcl} \vdash_{wE} D_i & \rhd & \tau.\sum_{i\in I} D_i t_i \\ & = & \tau.t_i \\ & = & D_i\tau.t_i \\ & = & \sum_{i\in I} D_i\tau.t_i \end{array}$$

By PARTITION we obtain $\vdash_{wE} D \rhd \tau.\sum_{i\in I} D_i t_i = \sum_{i\in I} D_i\tau.t_i$. By L-INPUT,

$$\vdash_{wE} D \rhd a(x).\tau.\sum_{i\in I} D_i t_i = a(x).\sum_{i\in I} D_i\tau.t_i$$

From this an application of T1 gives the result. □

Now we prove Theorem 4.11 for $\vdash_{wE}$.

Proof: Recall that $t \equiv \sum_{i\in I} M_i\alpha_i.t_i$, $u \equiv \sum_{j\in J} N_j\beta_j.u_j$ and $t \simeq_E^C u$. We need to show, for every $D \in MCE_{fn(t,u)}(C)$, $\vdash_{wE} D \rhd t = t + N_j\beta_j.u_j$ for each j.

Case $\beta_j \equiv b(z)$, $z \notin fn(t,u,C)$. Let $I_D = \{\, i \in I \mid \alpha_i \equiv a_i(z),\ D \Rightarrow a_i = b \wedge M_i \wedge N_j \,\}$ and $t_D \equiv \sum_{i\in I_D} a_i(z).t_i$. Using Proposition 3.1 we only need to show

$$\vdash_{wE} D \rhd t_D = t_D + b(z).u_j$$

Since $u \xrightarrow{N_j, b(z)} u_j$, for each $D' \in MCE_{fn(t,u,C)\cup\{z\}}(D)$, $t \xrightarrow{M_i, a_i(z)} t_{i_{D'}} \stackrel{M',\hat{\tau}}{\Longrightarrow}_E t'$ for some $i_{D'} \in I_D$ and $t' \approx_E^{D'} u_j$. By the same argument as in the τ case in the proof of Theorem 3.14 we can obtain $\vdash_{wE} D' \rhd \tau.t' = \tau.u_j$. By Lemma 3.13,

$$\vdash_{wE} D' \rhd \tau.t_{i_{D'}} = \tau.t_{i_{D'}} + \tau.u_j \tag{5}$$

Write MCE for $MCE_{fn(t,u,C)\cup\{z\}}(D)$. Since $\{\, i_{D'} \mid D' \in MCE \,\} \subseteq I_D \subseteq I$, by S2 and Lemma 7.1

$$\vdash_{wE} D \rhd t_D = t_D + b(z).\sum_{D'\in MCE} D' t_{i_{D'}}$$

So we are done if we can show

$$\vdash_{wE} D \rhd b(z).\sum_{D'\in MCE} D' t_{i_{D'}} = b(z).\sum_{D'\in MCE} D' t_{i_{D'}} + b(z).u_j$$

which can be derived as follows:

$$
\begin{array}{rcl}
\vdash_{wE} D & \rhd & b(z).\sum_{D' \in MCE} D' t_{i_{D'}} \\
& \stackrel{7.2}{=} & b(z).\sum_{D' \in MCE} D' \tau.t_{i_{D'}} \\
& \stackrel{(5)}{=} & b(z).\sum_{D' \in MCE} D'(\tau.t_{i_{D'}} + \tau.u_j) \\
& \stackrel{3.1}{=} & b(z).(\sum_{D' \in MCE} D' \tau.t_{i_{D'}} + \sum_{D' \in MCE} D' \tau.u_j) \\
& = & b(z).(\sum_{D' \in MCE} D' \tau.t_{i_{D'}} + D\tau.u_j) \\
& \stackrel{3.1}{=} & b(z).(\sum_{D' \in MCE} D' \tau.t_{i_{D'}} + \tau.u_j) \\
& \stackrel{T3}{=} & b(z).(\sum_{D' \in MCE} D' \tau.t_{i_{D'}} + \tau.u_j) + b(z).u_j \\
& = & b(z).\sum_{D' \in MCE} D' t_{i_{D'}} + b(z).u_j
\end{array}
$$

This completes the proof. $\square$