A Fully-Abstract Semantics for Atomicity

Roberto Gorrieri
gorrieri@cs.unibo.it

Dipartimento di Scienze dell'Informazione, Università di Bologna,
Mura A. Zamboni, 7, 40127 Bologna, Italy

Abstract. Multi-CCS extends conservatively CCS with an operator of strong prefixing able to model atomic sequences of actions as well as multiparty synchronization. We equip this calculus with an interleaving semantics, which makes use of a limited structural congruence in order to ensure that parallel composition be associative. Unfortunately, this semantics is not compositional, as it is not a congruence for parallel composition: in $p \,|\, q$, an atomic sequence performed by a process p can sense, to some extent, the degree of concurrency of the parallel component q. Step bisimilarity, however, turns out to be too discriminating: there are interleaving bisimilar processes, that no parallel context can tell apart, that are not step bisimilar. Then, linear-step bisimilarity is introduced and proved to be the coarsest congruence contained in interleaving bisimilarity. A finite sound and complete axiomatization of linear-step bisimilarity is presented for finite Multi-CCS processes.

1 Introduction

CCS [12], even if Turing-complete, is not expressive enough to model some useful behaviour. For instance, it does not support multiway synchronization, nor atomic transactions. In this line, it can be proved [9, 10] that a classic solution to the famous dining philosophers problem [3] that assumes atomicity in the acquisition of the forks (or equivalently, that requires a three-way synchronization among one philosopher and the two forks), cannot be provided in CCS.

Multi-CCS is a conservative extension to CCS to allow for the description of transactional behaviour, as well as multiparty synchronization expressed by means of an atomic sequence of binary CCS synchronizations. Syntactically, this is obtained by adding one prefixing operator $\underline{\mu}.p$, called *strong prefixing* (in opposition to normal prefixing $\mu.p$): in a process $\underline{\mu}.p$, action μ is the first one of a transaction that continues with p, provided that p can complete the transaction. Therefore, for instance, $\underline{a}.\underline{b}.p \xrightarrow{ab} p$, i.e., the resulting transition system is labeled on sequences of actions. There is also a generalization of the operational rule for communication that takes care of the synchronization of two sequences: if $p \xrightarrow{\sigma_1} p'$ and $q \xrightarrow{\sigma_2} q'$, then $p \,|\, q \xrightarrow{\sigma} p' \,|\, q'$, where sequence σ has been obtained by an interleaving (with possible synchronizations) of σ_1 and σ_2 where at least one synchronization has taken place. Sequence σ can be used for a possible synchronization with an additional process r: since $p \,|\, q \xrightarrow{\sigma} p' \,|\, q'$, if $r \xrightarrow{\sigma'} r'$, then

$(p \mid q) \mid r \xrightarrow{\sigma''} (p' \mid q') \mid r'$, hence representing a form of ternary synchronization, realized by means of an atomic sequence of binary synchronizations. In general, any form of multiway synchronization can be represented.

Strong prefixing was introduced in [6, 5]. The calculus presented there was named A^2CCS. It differs from Multi-CCS for two important features. First, parallel composition is associative for Multi-CCS thanks to the introduction of a structural congruence $\equiv$ induced by a minimal set of equations. Second, the synchronization discipline on pairs of sequences is more liberal in Multi-CCS (it simply requires that at least one synchronization takes place) and less verbose (as intermediate occurrences of τ are dissolved, so that labels can be only of two types: either τ or a non-empty sequence of visibile actions).

Multi-CCS is very expressive and many well-known synchronization problems are easily solvable, e.g., dining philosophers [3], concurrent readers and writers [2], Patil's cigarettes' smokers [16], etc.. In [7], it is also shown that such a calculus is expressive enough to represent all finite Place/Transition Petri nets.

In this paper, we investigate the behavioural semantics for Multi-CCS. Interleaving bisimilarity enjoys some expected algebraic laws but it is not compositional, as it is not a congruence for parallel composition. The problem is due to the fact that interleaving bisimilar processes may be distinguished by a parallel component able to perform an atomic sequence. For instance, consider process $p = \overline{a}.\overline{a}.\mathbf{0}$ and process $q = \overline{a}.\mathbf{0} \mid \overline{a}.\mathbf{0}$. Clearly, p is bisimilar to q, written $p \sim q$. However, if we consider a context $\mathcal{C}[-] = - \mid \underline{a}.\underline{a}.c.\mathbf{0}$, we get that $\mathcal{C}[p] \not\sim \mathcal{C}[q]$, because the latter can perform c, i.e., $\mathcal{C}[q] \xrightarrow{c} (\mathbf{0} \mid \mathbf{0}) \mid \mathbf{0}$, while $\mathcal{C}[p]$ cannot. The reason for this difference is that the process $\underline{a}.\underline{a}.c.\mathbf{0}$ can react with a number of concurrently active components equal to the length of the trace it can perform. Hence, a congruence semantics for parallel composition may need to distinguish p and q on the basis of their different degree of parallelism.

Therefore, one may define a step transition system where transitions are labeled by multisets of concurrently executable sequences. Step bisimilarity would distinguish $p = \overline{a}.\overline{a}.\mathbf{0}$ and $q = \overline{a}.\mathbf{0} \mid \overline{a}.\mathbf{0}$, as only q can perform a step $\{\overline{a}, \overline{a}\}$. However, step bisimilarity is not fully abstract: it is possible to find processes p and q such that for all r, $p \mid r \sim q \mid r$, but $p \not\sim_{step} q$. Simply take $p = (a \mid a) + \underline{a}.a.\mathbf{0}$ and $q = a.a.\mathbf{0} + \underline{a}.a.\mathbf{0}$. In other words, step bisimilarity is not the coarsest congruence contained in interleaving bisimilarity.

This example suggests that the contents of a step $\{a, a\}$ which is testable by a parallel component is its possible atomic linearisation aa. Hence, it seems that the coarsest congruence should not observe steps, rather linearisation of steps. Following this intuition, we define a novel semantics, called *linear-step* semantics and prove to be the coarsest congruence contained in interleaving bisimilarity. Moreover, contrary to non-interleaving equivalences, linear-step bisimilarity can be axiomatized. For finite Multi-CCS we present a finite sound and complete axiomatization.

The paper is organized as follows. Section 2 presents the syntax of Multi-CCS. Section 3 reports the interleaving semantics and its algebraic properties. Section 4 introduces the step semantics for Multi-CCS and discusses that, albeit

a congruence, it is not fully asbtract. Section 5 proposes the novel linear-step semantics and proves that it is the coarsest congruence. Section 6 is devoted to the finite axiomatization. Some concluding remarks are reported in Section 7.

2 Multi-CCS: Syntax

We assume the reader familiar with CCS [12, 14]. We assume to have a denumerable set $\mathcal{L}$ of channel names, its complementary set $\overline{\mathcal{L}}$ of co-names, the set $\mathcal{L} \cup \overline{\mathcal{L}}$ (ranged over by $\alpha, \beta, \ldots$) of visible actions and the set of all actions $Act = \mathcal{L} \cup \overline{\mathcal{L}} \cup \{\tau\}$, such that $\tau \notin \mathcal{L} \cup \overline{\mathcal{L}}$, ranged over by μ.

The process terms are generated by the following grammar:

$$p ::= \mathbf{0} \mid \mu.p \mid \underline{\mu}.p \mid p + p \mid p \,|\, p \mid (\nu a)p \mid C$$

Term $\mathbf{0}$ is the terminated process, $\mu.p$ is a normally prefixed process where action μ (that can be either an input a, an output $\bar{a}$ or a silent move τ) is first performed and then p is ready, $\underline{\mu}.p$ is a strongly prefixed process where μ is the first action of a transaction that continues with p (provided that p can complete the transaction). With $p + p'$ we denote the process obtained by the alternative composition of p and p'. Term $p \,|\, p'$ is the parallel composition of p and p', $(\nu a)p$ is process p where the (input) name a is made private (restriction), and C is a process constant, equipped with a defining equation $C \stackrel{def}{=} p$.

We denote with $\mathcal{P}$ the set of *processes*, containing those terms which are, w.r.t. process constants, closed and *guarded* (for any defining equation $C \stackrel{def}{=} p$, any occurrence of a constant in p is within a *normally prefixed* subprocess $\mu.p'$ of p). $\mathcal{P}$ will be ranged over by p, q, r, possibly indexed.

3 Interleaving Semantics

The operational semantics for Multi-CCS is given by the labelled transition system $(\mathcal{P}, \mathcal{A}, \longrightarrow)$, where the states are the processes in $\mathcal{P}$, $\mathcal{A} = (\mathcal{L} \cup \overline{\mathcal{L}})^+ \cup \{\tau\}$ is the set of labels (ranged over by σ, possibly indexed), and $\longrightarrow \subseteq \mathcal{P} \times \mathcal{A} \times \mathcal{P}$ is the minimal transition relation generated by the rules listed in Table 1.

Let us comment only the non-standard rules. Rule (S-Pref$_3$) is responsible for the creation of transitions labeled by non-empty sequences of actions: in order for $\underline{\alpha}.p$ to make a move, it is necessary that p can perform a transition, i.e., the rest of the transaction. Hence, if $p \xrightarrow{\sigma} p'$ then $\underline{\alpha}.p \xrightarrow{\alpha\sigma} p'$. Note that $\underline{\mu}.\mathbf{0}$ cannot perform any action, as $\mathbf{0}$ is stuck. Usually, if a transition is labeled by $\sigma = \alpha_1 \ldots \alpha_{n-1}\alpha_n$, then all the actions $\alpha_1 \ldots \alpha_{n-1}$ are due to strong prefixes, while α_n to a normal prefix (or α_n is the last strong prefix before a τ). Rules (S-Pref$_1$) and (S-Pref$_2$) ensures that τ's are never added in a sequence σ, hence ensuring that in a transition $p \xrightarrow{\sigma} p'$ either $\sigma = \tau$ or σ is composed only of visible actions, i.e., σ ranges over $\mathcal{A} = (\mathcal{L} \cup \overline{\mathcal{L}})^+ \cup \{\tau\}$.

Rule (S-Com) has a side-condition on the possible synchronizability of sequences σ_1 and σ_2, whose result may be σ. $Sync(\sigma_1, \sigma_2, \sigma)$ holds if σ is obtained

$$\text{(Pref)}\quad \mu.p \xrightarrow{\mu} p \qquad \text{(S-Pref}_1\text{)}\quad \frac{p \xrightarrow{\sigma} p'}{\underline{\tau}.p \xrightarrow{\sigma} p'}$$

$$\text{(S-Pref}_2\text{)}\quad \frac{p \xrightarrow{\tau} p'}{\underline{\alpha}.p \xrightarrow{\alpha} p'} \qquad \text{(S-Pref}_3\text{)}\quad \frac{p \xrightarrow{\sigma} p' \quad \sigma \neq \tau}{\underline{\alpha}.p \xrightarrow{\alpha\sigma} p'}$$

$$\text{(Cong)}\quad \frac{p \equiv p' \xrightarrow{\sigma} q' \equiv q}{p \xrightarrow{\sigma} q} \qquad \text{(Sum}_1\text{)}\quad \frac{p \xrightarrow{\sigma} p'}{p + q \xrightarrow{\sigma} p'}$$

$$\text{(Par}_1\text{)}\quad \frac{p \xrightarrow{\sigma} p'}{p \,|\, q \xrightarrow{\sigma} p' \,|\, q} \qquad \text{(S-Res)}\quad \frac{p \xrightarrow{\sigma} p'}{(\nu a)p \xrightarrow{\sigma} (\nu a)p'}\ a, \bar{a} \notin n(\sigma)$$

$$\text{(S-Com)}\ \frac{p \xrightarrow{\sigma_1} p' \qquad q \xrightarrow{\sigma_2} q'}{p \,|\, q \xrightarrow{\sigma} p' \,|\, q'}\ Sync(\sigma_1, \sigma_2, \sigma)$$

Table 1. Operational semantics (symmetric rules for (Sum$_1$) and (Par$_1$) omitted)

S1	$(p \,\|\, q) \,\|\, r = p \,\|\, (q \,\|\, r)$	
S2	$p \,\|\, q = q \,\|\, p$	
S3	$A = q$	if $A \stackrel{def}{=} q$
S4	$(\nu a)(p \,\|\, q) = p \,\|\, (\nu a)q$	if a not free in p
S5	$(\nu a)p = (\nu b)(p\{b/a\})$	if b does not occur in p

Table 2. Axioms generating the structural congruence $\equiv$.

from an interleaving (possibly with synchronizations) of σ_1 and σ_2, where at least one synchronization has taken place. Relation *Sync* is defined by the inductive rules of Table 3, which make use of the auxiliary relation *Int*, which is just as *Sync* without requiring that at least one synchronization occurs.[1] Note that when the resulting σ is not τ, then it can be used for further synchronization with some additional parallel components, hence allowing for multiparty synchronization.

Rule (S-Res) is slightly different, as it requires that no action in σ can be a or $\bar{a}$. With $n(\sigma)$ we denote the set of all actions occurring in σ.

There is one further new rule, called (Cong), which makes use of a structural congruence $\equiv$, that is needed to overcome a shortcoming of parallel composi-

[1] In the definition of *Sync* and *Int*, with abuse of notation, we let σ range over $\mathcal{A} \cup \{\epsilon\}$, the empty sequence.

$$Sync(\alpha, \overline{\alpha}, \tau) \qquad \frac{Int(\sigma_1, \sigma_2, \sigma)}{Sync(\alpha\sigma_1, \overline{\alpha}\sigma_2, \sigma)} \qquad \frac{Sync(\sigma_1, \sigma_2, \tau)}{Sync(\alpha\sigma_1, \sigma_2, \alpha)}$$

$$\frac{Sync(\sigma_1, \sigma_2, \tau)}{Sync(\sigma_1, \alpha\sigma_2, \alpha)} \quad \frac{Sync(\sigma_1, \sigma_2, \sigma) \quad \sigma \neq \tau}{Sync(\alpha\sigma_1, \sigma_2, \alpha\sigma)} \quad \frac{Sync(\sigma_1, \sigma_2, \sigma) \quad \sigma \neq \tau}{Sync(\sigma_1, \alpha\sigma_2, \alpha\sigma)}$$

$$Int(\alpha, \overline{\alpha}, \tau) \qquad Int(\alpha, \epsilon, \alpha) \qquad Int(\epsilon, \alpha, \alpha) \qquad \frac{Int(\sigma_1, \sigma_2, \sigma)}{Int(\alpha\sigma_1, \overline{\alpha}\sigma_2, \sigma)}$$

$$\frac{Int(\sigma_1, \sigma_2, \tau)}{Int(\alpha\sigma_1, \sigma_2, \alpha)} \quad \frac{Int(\sigma_1, \sigma_2, \tau)}{Int(\sigma_1, \alpha\sigma_2, \alpha)} \quad \frac{Int(\sigma_1, \sigma_2, \sigma) \quad \sigma \neq \tau}{Int(\alpha\sigma_1, \sigma_2, \alpha\sigma)} \quad \frac{Int(\sigma_1, \sigma_2, \sigma) \quad \sigma \neq \tau}{Int(\sigma_1, \alpha\sigma_2, \alpha\sigma)}$$

Table 3. Synchronization relation *Sync* and interleaving relation *Int*.

tion: without rule (Cong), parallel composition is not associative. The structural congruence $\equiv$ on process terms is induced by the five axioms (S1-S5) in Table 2. The first equation is for associativity of the parallel operator; the second law is for commutativity of parallel composition. The third equation explains why we have no explicit operational rule for handling constants: the transitions derivable from A are those transitions derivable from the structurally congruent term p, if $A \stackrel{def}{=} p$. The fourth law allows for enlargement of the scope of restriction; the last equation is the so-called law of *alpha-conversion*, which makes use of syntactic substitution. The intuition is that, with this structural congruence, all the parallel components are to be considered as in a bunch where they can freely interact independently of their actual association or ordering. Indeed, rule (Cong) enlarges the set of transitions derivable from p, as the following four examples show.

Example 1. **(Multi-party synchronization)** Assume that three processes want to synchronize. This can be easily expressed in Multi-CCS by means of a transaction. E.g., consider $p = \underline{a}.a.p'$, $q = \bar{a}.q'$ and $r = \bar{a}.r'$ and the whole system $P = (\nu a)((p \,|\, q) \,|\, r)$. It is easy to see that $P \stackrel{\tau}{\longrightarrow} (\nu a)((p' \,|\, q') \,|\, r')$ (and this can be proved in two ways), so the three processes have synchronized in one single atomic transition. It is interesting to observe that $P' = (\nu a)(p \,|\, (q \,|\, r))$ could not perform the multiway synchronization if rule (Cong) were not allowed. □

Example 2. In order to see that also the commutativity axiom may be useful, consider process $p = (\underline{a}.c.0|b.0)|(\overline{a}.0|\underline{\overline{b}}.\overline{c}.0)$. Such a process can do a four-way synchronization τ to $q = (\mathbf{0} \,|\, \mathbf{0}) \,|\, (\mathbf{0} \,|\, \mathbf{0})$, because $p' = (\underline{a}.c.0|\overline{a}.0)|(b.0|\underline{\overline{b}}.\overline{c}.0)$, which is structurally congruent to p, can perform τ reaching q. Without rule (Cong), process p could not perform such a multiway synchronization. □

Example 3. Consider $Q = p_1 \,|\, (\nu a)(p_2 \,|\, p_3)$, where $p_1 = \underline{b}.c.p_1'$, $p_2 = \overline{b}.p_2'$ and $p_3 = \overline{c}.p_3'$. Without rule (Cong), the three-way synchronization $Q \stackrel{\tau}{\longrightarrow} Q''$, where

$Q'' = p'_1 \,|\, (\nu a)(p'_2 \,|\, p'_3)$, could not take place. However, by taking a new name d not occurring free in Q, we can prove that Q is structurally congruent to $Q' = (\nu d)((p_1 \,|\, p_2\{d/a\})p_3\{d/a\})$, because of scope enlargement and alpha-conversion, and $Q' \xrightarrow{\tau} Q''' \equiv Q''$, where $Q''' = (\nu d)((p'_1 \,|\, p'_2\{d/a\})p'_3\{d/a\})$. □

Example 4. Consider $R = \underline{a}.c.0|A$, where $A \stackrel{def}{=} \overline{a}.0 \,|\, \overline{c}.0$. Without rule (Cong) (and axiom (S3) of Table 2), it is not possible to derive $R \xrightarrow{\tau} 0 \,|\, (0 \,|\, 0)$. □

Example 5. **(Guardedness)** We assume that each process constant in a defining equation occurs inside a normally prefixed subprocess $\mu.q$. This will prevent infinitely branching sequential processes. E.g, consider the non legal process $A \stackrel{def}{=} \underline{a}.A + b.\mathbf{0}$. According to the operational rules, A has infinitely many transitions leading to $\mathbf{0}$, each of the form $a^n b$, for $n = 0, 1, ...$ In fact, under guardedness, the set of terms generated by

$$p ::= \mathbf{0} \;\mid\; \mu.p \;\mid\; \underline{\mu}.p \;\mid\; p + p \;\mid\; C$$

defines, up to isomorphism, the set of transition systems labeled on $\mathcal{A} = (\mathcal{L} \cup \overline{\mathcal{L}})^+ \cup \{\tau\}$ with finitely many states and transitions. □

Example 6. **(Dining Philosophers)** This famous problem, defined by Dijkstra in [3], can be solved in Multi-CCS. Five philosophers sit at a round table, with a private plate and where each of the five forks is shared by two neighbors. Philosophers can think and eat; in order to eat, a philosopher has to acquire both forks that he shares with his neighbors, starting from the fork at his left and then the one at his right. All philosophers behave the same, so the problem is intrinsically symmetric. Clearly a naïve solution would cause deadlock exactly when all five philosophers take the fork at their left at the same time and are waiting for the fork at their right. A simple solution is to force atomicity on the acquisition of the two forks so that either both are taken or none. In order to have a small model, we consider the case of two philosophers only. The forks can be defined by the constants $fork_i$:

$$fork_i \stackrel{def}{=} \overline{up_i}.\overline{dn_i}.fork_i \quad \text{for } i = 0, 1$$

The two philosophers can be described as

$$phil_i \stackrel{def}{=} think.phil_i + \underline{up_i}.up_{i+1}.eat.\underline{dn_i}.dn_{i+1}.phil_i \quad \text{for } i = 0, 1$$

where $i + 1$ is computed modulo 2 and the atomic sequence $up_i up_{i+1}$ ensures the atomic acquisition of the two forks. For simplicity, we assume also that the release of the two forks is atomic, but this is not necessary for correctness. The whole system is

$$DP \stackrel{def}{=} (\nu L)(phil_0 \,|\, phil_1 \,|\, fork_0 \,|\, fork_1)$$

where $L = \{up_0, up_1, dn_0, dn_1\}$. Note that the operational semantics generates a finite-state lts for DP, depicted in Figure 1. □

Fig. 1. The labeled transition system for DP.

3.1 Properties of the semantics

Two terms p and q are *interleaving bisimilar*, written $p \sim q$, if there exists a strong bisimulation [12] R such that $(p, q) \in R$. Interleaving bisimulation equivalence enjoys some expected algebraic properties.

Proposition 1. *Let $p, q, r \in \mathcal{P}$ be processes. Then the following holds:*

(1) $(p + q) + r \sim p + (q + r)$ (2) $p + q \sim q + p$
(3) $p + \mathbf{0} \sim p$ (4) $p + p \sim p$
(5) $p \,|\, (q \,|\, r) \sim (p \,|\, q) \,|\, r$ (6) $p \,|\, q \sim q \,|\, p$
(7) $p \,|\, \mathbf{0} \sim p$ (8) $(\nu x)(p \,|\, q) \sim p \,|\, (\nu x)q$ *if* $x \notin fn(p)$
(9) $(\nu x)(\nu y)p \sim (\nu y)(\nu x)p$ (10) $(\nu x)p \sim (\nu y)(p\{y/x\})$ *if* $y \notin fn(p)$
(11) $(\nu x)\mathbf{0} \sim \mathbf{0}$

Proof. The proof is standard. □

Corollary 1. *$p \equiv q$ implies $p \sim q$.*

Proof. Because of (5), (6), (8) and (10) of Proposition 1. □

A few properties of strong prefixing are as follows:

Proposition 2. *Let $p, q \in \mathcal{P}$ be processes. Then the following hold:*

(1) $\underline{\mu}.(p + q) \sim \underline{\mu}.p + \underline{\mu}.q$ (2) $\underline{\mu}.\mathbf{0} \sim \mathbf{0}$
(3) $\underline{\tau}.p \sim p$ (4) $\underline{\mu}.\tau.p \sim \mu.p$

□

Interleaving bisimulation is a congruence for almost all the operators of Multi-CCS, in particular for strong prefixing.

Proposition 3. *If $p \sim q$, then the following hold:*

1. $\mu.p \sim \mu.q$ *for all* $\mu \in Act$,
2. $\underline{\mu}.p \sim \underline{\mu}.q$ *for all* $\mu \in Act$
3. $p + r \sim q + r$ *for all* $r \in \mathcal{P}$,
4. $(\nu a)p \sim (\nu a)q$ *for all* $a \in \mathcal{L}$.

Proof. Standard. □

Unfortunately, $\sim$ is not a congruence for parallel composition, as the following example shows.

Example 7. **(No congruence for parallel composition)** Consider process $p = \overline{a}.\overline{a}.\mathbf{0}$ and process $q = \overline{a}.\mathbf{0} \,|\, \overline{a}.\mathbf{0}$. Clearly, $p \sim q$. However, if we consider a context $\mathcal{C}[-] = - \,|\, \underline{a}.\underline{a}.c.\mathbf{0}$, we get that $\mathcal{C}[p] \not\sim \mathcal{C}[q]$, because the latter can perform c, i.e., $\mathcal{C}[q] \xrightarrow{c} (\mathbf{0} \,|\, \mathbf{0}) \,|\, \mathbf{0}$, while $\mathcal{C}[p]$ cannot. The reason for this difference is that the process $\underline{a}.\underline{a}.c.\mathbf{0}$ can react with a number of concurrently active components equal to the length of the trace it can perform. Hence, a congruence semantics for parallel composition must distinguish p and q on the basis of their different degree of parallelism. □

$$(\text{Pref}^s)\quad \mu.p \xrightarrow{\{\mu\}}_s p \qquad (\text{S-Pref}_1^s)\ \frac{p \xrightarrow{\{\sigma\}}_s p'}{\underline{\tau}.p \xrightarrow{\{\sigma\}}_s p'}$$

$$(\text{S-pref}_2^s)\ \frac{p \xrightarrow{\{\tau\}}_s p'}{\underline{\alpha}.p \xrightarrow{\{\alpha\}}_s p'} \qquad (\text{S-Pref}_3^s)\ \frac{p \xrightarrow{\{\sigma\}}_s p' \quad \sigma \neq \tau}{\underline{\alpha}.p \xrightarrow{\{\alpha\sigma\}}_s p'}$$

$$(\text{Sum}_1^s)\ \frac{p \xrightarrow{M}_s p'}{p + q \xrightarrow{M}_s p'} \qquad (\text{Con}^s)\ \frac{p \xrightarrow{M}_s p'}{C \xrightarrow{M}_s p'} \quad C \stackrel{def}{=} p$$

$$(\text{Par}_1^s)\ \frac{p \xrightarrow{M}_s p'}{p \,|\, q \xrightarrow{M}_s p' \,|\, q} \qquad (\text{Res}^s)\ \frac{p \xrightarrow{M}_s p'}{(\nu a)p \xrightarrow{M}_s (\nu a)p'}\ \forall \sigma \in M\ \ a, \bar{a} \notin n(\sigma)$$

$$(\text{S-Com}^s)\ \frac{p \xrightarrow{M_1}_s p' \qquad q \xrightarrow{M_2}_s q'}{p \,|\, q \xrightarrow{M} p' \,|\, q'}\ MSync(M_1 \oplus M_2, M)$$

Table 4. Step operational semantics (symmetric rules for (Sum_1^s) and (Par_1^s) omitted).

4 Step Semantics

Multi-CCS can be equipped with a step semantics, i.e., a semantics where each transition is labeled by a (multi-)set of sequences that concurrent subprocesses can perform at the same time. This equivalence was originally introduced over Petri nets [15] and for a process algebra in [11].

The step operational semantics for Multi-CCS is given by the lts $(\mathcal{P}, \mathcal{B}, \longrightarrow_s)$, where the states are the processes in $\mathcal{P}$, $\mathcal{B} = \mathcal{M}_{fin}(\mathcal{A})$ is the set of labels (ranged over by M), and $\longrightarrow_s \subseteq \mathcal{P} \times \mathcal{B} \times \mathcal{P}$ is the minimal transition relation generated by the rules listed in Table 4. Note that rules (S-Pref_1^s), (S-Pref_2^s) and (S-Pref_3^s) assume that the transition in the premise is sequential, i.e., composed of one single sequence. Moreover, it can be proved that in rule (Sum_1^s), the derivable label M is always a singleton. Note also that rule (S-Com^s) uses an additional auxiliary relation *MSync*, defined in Table 5, where $\oplus$ denotes multiset union. The intuition behind the definition of rule (S-Com^s) and *MSync* is that, whenever two parallel processes p and q perform steps M_1 and M_2, then we can put all the sequences together – $M_1 \oplus M_2$ – and see if $MSync(M_1 \oplus M_2, \overline{M})$ holds. The resulting $\overline{M}$ may be just $M_1 \oplus M_2$ (hence no synchronization takes place), according to axiom $MSync(M, M)$, or the M' we obtain from the application of the rule: select two sequences σ_1 and σ_2 from $M_1 \oplus M_2$, synchronize them producing σ, then recursively apply *MSync* to $M_1 \oplus M_2 \setminus \{\sigma_1, \sigma_2\} \cup \{\sigma\}$ to obtain M'. This procedure of synchronizing sequences may go on until pairs of synchronizable sequences can be found, but may also stop at any moment due to the axiom $MSync(M, M)$.

It is interesting to observe that these step operational rules do not make use of structural congruence $\equiv$. The same operational effect of rule (Cong) is here en-

$$MSync(M, M) \qquad \frac{Sync(\sigma_1, \sigma_2, \sigma) \quad MSync(M \oplus \{\sigma\}, M')}{MSync(M \oplus \{\sigma_1, \sigma_2\}, M')}$$

Table 5. Step synchronization relation

sured by relation *MSync* that allows for multiple synchronization of concurrently active subprocesses.

Lemma 1. *Let $p, q \in \mathcal{P}$ be processes. Then the following hold:*

1. *If $p \xrightarrow{\{\sigma\}}_s q$, then $p \xrightarrow{\sigma} q$.*
2. *If $p \xrightarrow{\sigma} q$, then $\exists q' \equiv q$ such that $p \xrightarrow{\{\sigma\}}_s q'$.*

Proof. The proof of (1) is by induction on the proof of $p \xrightarrow{\{\sigma\}}_s q$. In the case of rule (S-Com^s), the actual proof of relation MSync tells in which order the parallel subcomponents are to be arranged by means of the structural congruence.

The proof of (2) is by induction on the proof of $p \xrightarrow{\sigma} q$. We cannot prove the stronger result $p \xrightarrow{\{\sigma\}}_s q$, because of the free use of structural congruence; e.g., $\mu.(p \,|\, (q \,|\, r)) \xrightarrow{\mu} ((p \,|\, q) \,|\, r)$ (due to (Cong)), while $\mu.(p \,|\, (q \,|\, r))$ cannot reach $((p \,|\, q) \,|\, r)$ in the step transition system. □

We call *step equivalence*, denoted $\sim_{step}$, the bisimulation equivalence on the step transition system of Multi-CCS. The following obvious fact follows.

Lemma 2. *Let $p, q \in \mathcal{P}$ be processes. If $p \equiv q$ then $p \sim_{step} q$.*

Proof. One has to show that for each equation $p = q$ generating $\equiv$, we have that $p \sim_{step} q$. The only non-trivial case is for associativity $(p \,|\, q) \,|\, r \;=\; p \,|\, (q \,|\, r)$ where one has to prove the following auxiliary lemma: if $p \xrightarrow{M_1}_s p'$, $q \xrightarrow{M_2}_s q'$ and $r \xrightarrow{M_3}_s r'$, then for all M, M' such that $Sync(M_1 \oplus M_2, M')$ and $Sync(M' \oplus M_2, M)$, there exists N such that $Sync(N_2 \oplus M_3, N)$ and $Sync(M_1 \oplus N, M)$. The thesis of this lemma follows by observing that such M can be obtained as $Sync(M_1 \oplus M_2 \oplus M_3, M)$. □

Proposition 4. *For any pair of processes $p, q \in \mathcal{P}$, if $p \sim_{step} q$ then $p \sim q$.*

Proof. Let R be a step bisimulation (i.e., a bisimulation over the step lts) such that $(p, q) \in R$. Then, $\equiv \circ R \circ \equiv$ is an interleaving bisimulation containing the pair (p, q) (where $\circ$ is the operation of relation composition). To prove this, we can resort to Lemma 1 and Lemma 2. □

Theorem 1. (Congruence) *If $p \sim_{step} q$, then the following hold:*

1. *$\mu.p \sim_{step} \mu.q \quad$ for all $\mu \in Act$,*

2. $\underline{\mu}.p \sim_{step} \underline{\mu}.q \quad$ *for all* $\mu \in Act$
3. $p + r \sim_{step} q + r \quad$ *for all* $r \in \mathcal{P}$,
4. $p \,|\, r \sim_{step} q \,|\, r \quad$ *for all* $r \in \mathcal{P}$,
5. $(\nu a)p \sim_{step} (\nu a)q \quad$ *for all* $a \in \mathcal{L}$.

Proof. Standard. □

Theorem 1(4) and Proposition 4 ensure that for any pair of processes $p, q \in \mathcal{P}$, if $p \sim_{step} q$ then, for all $r \in \mathcal{P}$, $p \,|\, r \sim q \,|\, r$. One may wonder if the reverse hold, i.e., if for all $r \in \mathcal{P}$, $p \,|\, r \sim q \,|\, r$ can we conclude that $p \sim_{step} q$? If this is the case, we can say that step equivalence is the coarsest congruence contained in interleaving bisimulation. The answer to this question is negative, as the following examples show.

Example 8. Take processes $p = \tau.\tau.\mathbf{0}$ and $q = \tau \,|\, \tau$. It is not difficult to see that for all $r \in \mathcal{P}$, $p \,|\, r \sim q \,|\, r$; however, $p \not\sim_{step} q$ as only the latter can perform the step $\{\tau, \tau\}$. □

Example 9. Take $p = (a \,|\, a) + \underline{a}.a.\mathbf{0}$ and $q = a.a.\mathbf{0} + \underline{a}.a.\mathbf{0}$. It is easy to see that $p \not\sim_{step} q$, even if for all $r \in \mathcal{P}$, $p \,|\, r \sim q \,|\, r$. □

5 Coarsest Congruence: Linear-step bisimilarity

We may wonder which is the coarsest (i.e., as abstract as possible) congruence contained in interleaving bisimulation. We observed above that step bisimilarity, albeit a congruence, is not the coarsest one, as it distinguishes processes that no parallel context can tell apart. Exemple 9 suggests that the contents of a step $\{a, a\}$ which is testable by a parallel component is its possible atomic linearisation aa. Hence, it seems that the coarsest congruence should not observe steps, rather linearisation of steps. Example 8 also suggests that the coarsest congruence should not be able to observe multiple occurrences of concurrent τ's. This requirement is rather intuitive: if some activity is unobservable, we cannot say how many concurrent internal activities it is composed of.

We define a third operational semantics for Multi-CCS. The linear-step operational semantics is given by the lts $(\mathcal{P}, \mathcal{A}, \longrightarrow_{ls})$, where and $\longrightarrow_{ls} \subseteq \mathcal{P} \times \mathcal{A} \times \mathcal{P}$ is the minimal transition relation generated by the rules listed in Table 6. Note also that rule (S-Com^{ls}) uses an additional auxiliary relation $AInt$, defined in Table 7, which is an extension of relation Int of Table 3 to consider also the case in which the arguments may be τ. Note that if $AInt(\sigma_1, \sigma_2, \sigma)$ holds, then σ is either τ or a sequence composed only of observable actions. Observe that the only real difference between the interleaving semantics and the linear-step semantics is that $AInt(\sigma_1, \sigma_2, \sigma)$ allows for the free interleaving of σ_1 and σ_2, while $Sync(\sigma_1, \sigma_2, \sigma)$ holds only if at least one synchronization takes place. Therefore, $a \,|\, a \xrightarrow{aa}_{ls} \mathbf{0} \,|\, \mathbf{0}$, while $a \,|\, a \not\xrightarrow{aa}$. This extra possibility is also compensating the omission of rule (Cong), as already observed for the step semantics.

Lemma 3. *Let* $p, q \in \mathcal{P}$. *If* $p \xrightarrow{\sigma} q$, *then* $\exists q' \equiv q$ *such that* $p \xrightarrow{\sigma}_{ls} q'$. □

$$(\text{Pref}^{ls}) \quad \mu.p \xrightarrow{\mu}_{ls} p \qquad (\text{S-Pref}_1^{ls}) \quad \frac{p \xrightarrow{\sigma}_{ls} p'}{\underline{\tau}.p \xrightarrow{\sigma}_{ls} p'}$$

$$(\text{S-Pref}_2^{ls}) \quad \frac{p \xrightarrow{\tau}_{ls} p'}{\underline{\alpha}.p \xrightarrow{\alpha}_{ls} p'} \qquad (\text{S-Pref}_3^{ls}) \quad \frac{p \xrightarrow{\sigma}_{ls} p' \quad \sigma \neq \tau}{\underline{\alpha}.p \xrightarrow{\alpha\sigma}_{ls} p'}$$

$$(\text{Sum}_1^{ls}) \quad \frac{p \xrightarrow{\sigma}_{ls} p'}{p + q \xrightarrow{\sigma}_{ls} p'} \qquad (\text{Cons}^{ls}) \quad \frac{p \xrightarrow{\sigma}_{ls} p'}{C \xrightarrow{\sigma}_{ls} p'} \quad C \stackrel{def}{=} p$$

$$(\text{Par}_1^{ls}) \quad \frac{p \xrightarrow{\sigma}_{ls} p'}{p \,|\, q \xrightarrow{\sigma}_{ls} p' \,|\, q} \qquad (\text{S-Res}^{ls}) \quad \frac{p \xrightarrow{\sigma}_{ls} p'}{(\nu a)p \xrightarrow{\sigma}_{ls} (\nu a)p'} \; a, \bar{a} \notin n(\sigma)$$

$$(\text{S-Com}^{ls}) \quad \frac{p \xrightarrow{\sigma_1}_{ls} p' \qquad q \xrightarrow{\sigma_2}_{ls} q'}{p \,|\, q \xrightarrow{\sigma}_{ls} p' \,|\, q'} \; AInt(\sigma_1, \sigma_2, \sigma)$$

Table 6. Operational semantics (symmetric rules for (Sum_1^{ls}) and (Par_1^{ls}) omitted)

$$AInt(\tau, \sigma, \sigma) \quad AInt(\sigma, \tau, \sigma) \quad \frac{\sigma_1 \neq \tau \quad \sigma_2 \neq \tau \quad Int(\sigma_1, \sigma_2, \sigma)}{AInt(\sigma_1, \sigma_2, \sigma)}$$

Table 7. Abstract interleaving relation

Ordinary bisimulation equivalence on the linear-step transition system is called *linear step equivalence* and denoted with $\sim_{ls}$. Of course, the structural congruence $\equiv$ is a finer equivalence relation.

Lemma 4. *Let $p, q \in \mathcal{P}$ be processes. If $p \equiv q$ then $p \sim_{ls} q$.* □

Linear step bisimilarity $\sim_{ls}$ is indeed finer than interleaving bisimilarity $\sim$ because of Lemma 3 and Lemma 4. The reverse implication does not hold: for instance, $(a.\mathbf{0} \,|\, b.\mathbf{0}) \sim a.b.\mathbf{0} + b.a.\mathbf{0}$ but the two are not linear-step bisimilar as only $(a.\mathbf{0} \,|\, b.\mathbf{0})$ can perform the atomic sequence ab.

Moreover, linear-step bisimilarity $\sim_{ls}$ is coarser than step bisimilarity $\sim_{step}$. This can be proved by showing that if $p \xrightarrow{\sigma}_{ls} p'$ then $p \xrightarrow{M}_s p'$, where σ is the result of the interleaving (according to relation $AInt$) of all the sequences in M. Hence, If $p \not\sim_{ls} q$ then there is a pair of reachable states p' and q' and a sequence σ such that $p' \xrightarrow{\sigma}_{ls} p''$ while q' cannot perform σ; as a consequence, this means that $p' \xrightarrow{M}_s p''$ with σ as the result of the linearisation of the sequences in M, while q' cannot perform M, i.e., $p \not\sim_{step} q$. The reverse implication does not hold: for instance, the two processes p and q discussed in Exercise 9 are linear-step

bisimilar, but not step bisimilar, as only p can perform a transition labeled by $\{a, a\}$. These observations justify the following:

Proposition 5. $p \equiv q \Rightarrow p \sim_{step} q \Rightarrow p \sim_{ls} q \Rightarrow p \sim q$. □

One can prove that $\sim_{ls}$ is a congruence for Multi-CCS.

Theorem 2. *If $p \sim_{ls} q$, then the following hold:*

1. $\mu.p \sim_{ls} \mu.q$ *for all* $\mu \in Act$,
2. $\underline{\mu}.p \sim_{ls} \underline{\mu}.q$ *for all* $\mu \in Act$
3. $p + r \sim_{ls} q + r$ *for all* $r \in \mathcal{P}$,
4. $p \,|\, r \sim_{ls} q \,|\, r$ *for all* $r \in \mathcal{P}$,
5. $(\nu a)p \sim_{ls} (\nu a)q$ *for all* $a \in \mathcal{L}$.

Proof. Standard. □

We are now ready to state the coarsest congruence theorem. We first need an auxiliary lemma.

Lemma 5. *If $p \stackrel{\tau}{\longrightarrow}_{ls} p'$, then $p \stackrel{\tau}{\Longrightarrow} p'$.*

Proof. By induction on the proof of $p \stackrel{\tau}{\longrightarrow}_{ls} p'$. The only non-trivial case is when $p = p_1 \,|\, p_2$ and $p_1 \,|\, p_2 \stackrel{\tau}{\longrightarrow}_{ls} p_1' \,|\, p_2'$ because $p_1 \stackrel{\sigma_1}{\longrightarrow}_{ls} p_1'$, $p_2 \stackrel{\sigma_2}{\longrightarrow}_{ls} p_2'$ and $AInt(\sigma_1, \sigma_2, \tau)$. By definition of relation $AInt$, the result may be τ only if either $\sigma_1 = \sigma_2 = \tau$ or $\sigma_1 \neq \tau \neq \sigma_2$ and $Int(\sigma_1, \sigma_2, \tau)$. In the former case, by induction we know that $p_1 \stackrel{\tau}{\Longrightarrow} p_1'$ and $p_2 \stackrel{\tau}{\Longrightarrow} p_2'$, hence by rule ($Par_1$) and ($Par_2$) we have $p_1 \,|\, p_2 \stackrel{\tau}{\Longrightarrow} p_1' \,|\, p_2 \stackrel{\tau}{\Longrightarrow} p_1' \,|\, p_2'$, as required. In the latter case, $Int(\sigma_1, \sigma_2, \tau)$ can be derived only if at least one synchronization takes place, hence also $Sync(\sigma_1, \sigma_2, \tau)$ holds. Now, by rule (S-Com) we have $p_1 \,|\, p_2 \stackrel{\tau}{\longrightarrow} p_1' \,|\, p_2'$, as required. □

Theorem 3. (Coarsest congruence) *Assume that $fn(p) \cup fn(q) \neq \mathcal{L} \cup \overline{\mathcal{L}}$. Then, $p \sim_{ls} q$ if and only if, for all $r \in \mathcal{P}$, $p \,|\, r \sim q \,|\, r$.*

Proof. The implication from left to right is due to the fact that $\sim_{ls}$ is a congruence for parallel composition (by Theorem 2(4)) and is also finer than interleaving bisimilarity (by Proposition 5). The implication from right to left is proved by contradiction: assuming that $p \not\sim_{ls} q$, we prove that there exists r such that $p \,|\, r \not\sim q \,|\, r$. As $p \not\sim_{ls} q$, there exist $p_i, q_i, i = 1, \ldots n (n \geq 0)$ such that $p \stackrel{\sigma_1}{\longrightarrow}_{ls} p_1 \stackrel{\sigma_2}{\longrightarrow}_{ls} p_2 \ldots \stackrel{\sigma_n}{\longrightarrow}_{ls} p_n$ and $q \stackrel{\sigma_1}{\longrightarrow}_{ls} q_1 \stackrel{\sigma_2}{\longrightarrow}_{ls} q_2 \ldots \stackrel{\sigma_n}{\longrightarrow}_{ls} q_n$, but $p_n \stackrel{\gamma}{\longrightarrow}_{ls} p_{n+1}$ and $q_n \stackrel{\gamma}{\not\longrightarrow}_{ls}$. If $\gamma = \tau$, then by Lemma 5, we have that $p_n \stackrel{\tau}{\Longrightarrow} p_{n+1}$, while by Lemma 3 we have that $q_n \stackrel{\tau}{\not\longrightarrow}$. As a consequence, $p \not\sim q$ and so for, e.g., $r = \mathbf{0}$, we have that $p \,|\, r \not\sim q \,|\, r$. If instead $\gamma = \alpha_1 \ldots \alpha_n$ for $n > 0$, let $r = \overline{\alpha_1}. \ldots . \overline{\alpha_n}.d.\mathbf{0}$, where d is an action not occurring in $fn(p) \cup fn(q)$. It is the case that $p_n \,|\, r \stackrel{d}{\longrightarrow} p_{n+1} \,|\, \mathbf{0}$, while $q_n \,|\, r \stackrel{d}{\not\longrightarrow}$. Hence, $p \,|\, r \not\sim q \,|\, r$. □

A1	$x+(y+z)=(x+y)+z$	**A2**	$x+y=y+x$
A3	$x+\mathbf{0}=x$	**A4**	$x+x=x$
P1	$\underline{\mu}.\mathbf{0}=\mathbf{0}$	**P2**	$\underline{\tau}.x=x$
P3	$\underline{\mu}.\tau.x=\mu.x$	**P4**	$\underline{\mu}.(x+y)=\underline{\mu}.x+\underline{\mu}.y$
R1		$(\nu a)\mathbf{0}=\mathbf{0}$	
R2	if $\mu \notin \{a,\bar{a}\}$	$(\nu a)\mu.x=\mu.(\nu a)x$	
R3	if $\mu \in \{a,\bar{a}\}$	$(\nu a)\mu.x=\mathbf{0}$	
R4	if $\mu \notin \{a,\bar{a}\}$	$(\nu a)\underline{\mu}.x=\underline{\mu}.(\nu a)x$	
R5	if $\mu \in \{a,\bar{a}\}$	$(\nu a)\underline{\mu}.x=\mathbf{0}$	
R6		$(\nu a)(x+y)=(\nu a)x+(\nu a)y$	

Table 8. Axioms for choice, strong prefixing and restriction.

6 Axiomatization

With the help of the auxiliary operators of left-merge $-\lfloor -$ and communication-merge $-\|-$ of ACP [1], we can provide a finite, sound and complete axiomatization to linear-step bisimilarity over finite Multi-CCS processes. The operational SOS rules for these two operators are as follows:

$$\text{(Left)}\ \frac{p \xrightarrow{\sigma}_{ls} p'}{p\lfloor q \xrightarrow{\sigma}_{ls} p' \mid q} \qquad \text{(S-Merge)}\ \frac{p \xrightarrow{\sigma_1}_{ls} p' \qquad q \xrightarrow{\sigma_2}_{ls} q'}{p\|q \xrightarrow{\sigma}_{ls} p' \mid q'}\ AInt(\sigma_1,\sigma_2,\sigma)$$

It is possible to prove that $\sim_{ls}$ is preserved by the left merge and the communication merge. So, it is a congruence not only for the operators of finite Multi-CCS, but also for these auxiliary operators. Hence, it can be axiomatized.

The axioms for the operators of choice, strong prefixing and restriction are reported in Table 8. These reflect the intuition that the normal forms for finite Multi-CCS are terms of the form $\Sigma_{i\in I}\sigma_i.p_i$, where each p_i is a normal form and σ_i is either τ or a sequence of visible actions; to be more precise, with $\sigma_i.p_i$ we actually mean $\tau.p_i$ if $\sigma=\tau$, or $\underline{\alpha}_1.\ldots.\underline{\alpha}_{n-1}.\alpha_n.p_i$ if $\sigma=\alpha_1\ldots\alpha_n$.

The axiomatization of the parallel composition operator is given by axiom **Par** of Table 9, together with the axioms **L1-L4** for left merge and **C1-C9** for communication merge. Let $\mathcal{MSB}$ be the set of axioms reported in Table 8 and Table 9. It is possible to prove that the equational theory $\mathcal{MSB}$ is a sound and complete *finite* axiomatization of $\sim_{ls}$.

Theorem 4. (Soundness) *If $\mathcal{MSB}\vdash p=q$, then $p\sim_{ls} q$.*

Proof. Because all the axioms in Tables 8 and 9 are sound for $\sim_{ls}$. □

About completeness, we first prove the statement for normal forms.

		Par $x \mid y = x\lfloor y + y\lfloor x + x \| y$		
	L1	$\mathbf{0}\lfloor y = \mathbf{0}$	**L2**	$(\mu.x)\lfloor y = \mu.(x \mid y)$
	L3	$(\underline{\mu}.x)\lfloor y = \underline{\mu}.(x\lfloor y)$	**L4**	$(x+y)\lfloor z = x\lfloor z + y\lfloor z$
C1	$x \| y = y \| x$	**C2**	$\mathbf{0} \| y = \mathbf{0}$	**C3** $(x+y) \| z = x \| z + y \| z$
C4	if $\mu_1 = \overline{\mu_2}$	$(\mu_1.x) \| (\mu_2.y) = \tau.(x \mid y) + \underline{\mu}_1.(\mu_2.y\lfloor x) + \underline{\mu}_2.(\mu_1.x\lfloor y)$		
C5	if $\mu_1 \neq \overline{\mu_2}$	$(\mu_1.x) \| (\mu_2.y) = \underline{\mu}_1.(\mu_2.y\lfloor x) + \underline{\mu}_2.(\mu_1.x\lfloor y)$		
C6	if $\mu_1 = \overline{\mu_2}$	$(\underline{\mu}_1.x) \| (\mu_2.y) = (x\lfloor y) + \underline{\mu}_1.(x \| \mu_2.y) + \underline{\mu}_2.(\underline{\mu}_1.x\lfloor y)$		
C7	if $\mu_1 \neq \overline{\mu_2}$	$(\underline{\mu}_1.x) \| (\mu_2.y) = \underline{\mu}_1.(x \| \mu_2.y) + \underline{\mu}_2.(\underline{\mu}_1.x\lfloor y)$		
C8	if $\mu_1 = \overline{\mu_2}$	$(\underline{\mu}_1.x) \| (\underline{\mu}_2.y) = (x \| y) + \underline{\mu}_1.(x \| \underline{\mu}_2.y) + \underline{\mu}_2.(\underline{\mu}_1.x \| y)$		
C9	if $\mu_1 \neq \overline{\mu_2}$	$(\underline{\mu}_1.x) \| (\underline{\mu}_2.y) = \underline{\mu}_1.(x \| \underline{\mu}_2.y) + \underline{\mu}_2.(\underline{\mu}_1.x \| y)$		

Table 9. Axioms for the auxiliary operators

Proposition 6. *Let p, q be normal forms. If $p \sim_{ls} q$, then $\mathcal{MSB} \vdash p = q$.*

Proof. *By induction on the sum of depths*[2] *of p and q.* □

In order to prove completeness for any pair of equivalent processes, we first need to show that any process can be reduced to normal form.

Lemma 6. *If p and q are normal forms, then the following hold:*

- *there exists a normal form r such that $\mathcal{MSB} \vdash \underline{\mu}.p = r$.*
- *there exists a normal form r such that $\mathcal{MSB} \vdash (\overline{\nu}a)p = r$.*
- *there exists a normal form r such that $\mathcal{MSB} \vdash p \mid q = r$.*

Proof. *By induction on the sum of depths of p and q.* □

Proposition 7. *For any process p there exists a normal form q such that $\mathcal{MSB} \vdash p = q$.*

Proof. *By structural induction on p, using Lemma 6.* □

Theorem 5. (Completeness) *If $p \sim_{ls} q$, then $\mathcal{MSB} \vdash p = q$.*

Proof. *First, by Proposition 7, we have normal forms p' and q' such that $\mathcal{MSB} \vdash p = p'$ and $\mathcal{MSB} \vdash q = q'$. By Theorem 4, $p \sim_{ls} p'$ and $q \sim_{ls} q'$, hence, by transitivity, also $p' \sim_{ls} q'$. By Proposition 6, we get that $\mathcal{MSB} \vdash p' = q'$ and so, by transitivity, also $\mathcal{MSB} \vdash p = q$.* □

[2] The depth of p is the maximal number of nested prefixes in its syntax.

7 Conclusion

We have studied the problem of finding the correct semantics for atomicity. To this aim, we have chosen a simple operator for atomicity (strong prefixing) and a discipline of interaction among atomic behaviors (relation *Sync*), yielding Multi-CCS. Then, we have identified the intuitively correct model for Multi-CCS: the interleaving model identifies all the actual (i.e., physical) atomic transitions of processes. However, we have seen that interleaving bisimulation is not a congruence for parallel composition. Hence, it turns out to be necessary to enrich the model with additional transitions. The first attempt has been to model steps of interleaving transitions; the resulting step bisimilarity is a congruence, but not the coarsest congruence. To this aim, a linear-step transitions system has been defined whose transitions are labeled with linearisation of steps. Linear-step bisimilarity, which is a new behavioural semantics, turns out to be the coarsest congruence contained in interleaving bisimilarity.

Linear-step bisimilarity is interesting also because, although finer than interleaving bisimilarity, it is still equipped with an axiomatization in interleaving style, i .e., with an expansion theorem. This is not the case, for instance, for step bisimilarity, for which no axiomatization is available.

We conjecture that linear step bisimilarity can be a very appropriate bisimulation-based behavioural equivalence also for calculi with joint input, such as those in [4, 10], as Multi-CCS subsumes such a kind of synchronization pattern. For instance, the joint input construct $[a_1, \ldots a_{n-1}, a_n].p$ of [10] can be easily encoded in Multi-CCS as $\underline{a_1}. \ldots \underline{a_{n-1}}.a_n.p$.

Acknowledgment

The first author would like to thank Massimo Morara for useful comments and suggestions.

References

1. J.A. Bergstra, J.W. Klop, "Process Algebra for Synchronous Communication", *Information and Control* 60:109-137, 1984.
2. P. Courtois, F. Heymans, D. Parnas, "Concurrent control with Readers and Writers", *Communications of the ACM* 14(10):667-668, 1971.
3. E.W. Dijkstra, "Hierarchical ordering of sequential processes", *Acta Informatica* 1(2):115-138, 1971.
4. C. Fournet, G. Gonthier, 'The reflexive CHAM and the join-calculus", in Procs POPL'96, pages 372 385, ACM Press, 1996.
5. R. Gorrieri, U. Montanari, "Towards Hierarchical Specification of Systems: A Proof System for Strong Prefixing", *Int. Journal of Foundations of Computer Science*, 1(3):277-293, 1990.
6. R. Gorrieri, S. Marchetti, U. Montanari, "A^2CCS: Atomic Actions for CCS", *Theoretical Computer Science* 72(2-3): 203-223, 1990.

7. R. Gorrieri, C. Versari, "A Process Calculus for Expressing Finite Place/Transition Petri Nets", in Procs. EXPRESS'10, EPTCS, 2010. DOI 10.4204/EPTCS.41.6, available at arXiv:1011.6433v1.
8. C.A.R. Hoare, *Communicating Sequential Processes* Prentice-Hall, 1985.
9. D.J. Lehmann. M.l O. Rabin., "On the advantages of free choice: A symmetric and fully distributed solution to the dining philosophers problem", In Procs. POPL'81, pages 133-138, ACM Press, 1981.
10. C. Laneve, A. Vitale, "The Expressive Power of Synchronizations", in Procs LICS'10, pages 382-391, IEEE-CS Press, 2010.
11. G. J. Milne, "Circal and the Representation of Communication, Concurrency, and Time", *ACM Trans. Program. Lang. Syst.* 7(2): 270-298, 1985.
12. R. Milner. *Communication and Concurrency*, Prentice-Hall, 1989.
13. R. Milner. A Complete Axiomatisation for Observational Congruence of Finite-State Behaviors. *Inf. Comput.* 81(2): 227-247, 1989.
14. R. Milner. *Communicating and mobile systems: the π-calculus*, Cambridge University Press, 1999.
15. M. Nielsen and P. S. Thiagarajan, "Degrees of Non-Determinism and Concurrency: A Petri Net View", in Procs. of the Fourth Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'84), Lecture Notes in Computer Science vol. 181, Springer, Berlin, pp. 89-117, 1984.
16. S. Patil "Limitations and Capabilities of Dijkstra's Semaphore Primitives for Coordination Among Processes", Computation Structures Group Memo 57, Project MAC, MIT, 1971.
17. J.L. Peterson, *Petri Net Theory and the Modeling of Systems*, Prentice-Hall, 1981.
18. C. Versari, N. Busi, R. Gorrieri. An expressiveness study of priority in process calculi. *Mathematical Structures in Computer Science* 19(6): 1161-1189, 2009.