A Multiset Semantics for the pi-Calculus with Replication

Joost Engelfriet*

Department of Computer Science, Leiden University
P.O.Box 9512, 2300 RA Leiden, The Netherlands

Abstract. A multiset (or Petri net) semantics is defined for the π-calculus with replication. The semantic mapping is a strong bisimulation, and structurally congruent processes have the same semantics. This paper is readable without knowledge of Petri nets.

1 Introduction

The π-calculus has recently been introduced as an extension of CCS to mobile concurrent processes (see [16, 13, 14, 15]). As for CCS [12], the (interleaving) semantics of the π-calculus is given by a transition system of which the states are process terms. In this paper, and in its sequel paper [9], we provide a Petri net semantics for the "small π-calculus", i.e., the subset of the π-calculus defined by Milner in [13]. The main features of this subset are that it has no choice operator and that recursion is replaced by the more basic operation of replication, denoted by an exclamation mark: if P is a process, then $!P$ stands for a countably infinite number of concurrent copies of P. It is shown in [13] that this subset suffices to simulate important aspects of the λ-calculus.

Petri net semantics of process algebras has been studied in, e.g., [1, 6, 10, 17, 18, 21]. In such a semantics, a Petri net is associated with each process; the idea is that this Petri net expresses the concurrency present in the process in a more direct way than the interleaving transition system. Here we wish to stress that a Petri net (and in particular a P/T net) is just a particular kind of transition system, viz. one of which the states are multisets (also called markings) and the transition relation $\rightarrow$ satisfies the following "chemical law" (where S_1, S_2, and S are states and $\cup$ is multiset union): if $S_1 \rightarrow S_2$, then $S_1 \cup S \rightarrow S_2 \cup S$. For this reason, we propose the term "multiset transition system" as an alternative to "Petri net" (just as a "transition system" used to be called "automaton"). It has been the early insight of Petri that the (multi)set is exactly the datastructure that fits to the notion of concurrency: a (multi)set can be viewed as the parallel composition of all its elements, and the communication between these elements can be modeled by (multi)set replacement, as formalized by the chemical law. The suggestive "chemical" terminology is from [2, 4] where a multiset is viewed as

* The research of the author was supported by the Esprit Basic Working Group No.6067 CALIBAN.

a chemical soup of molecules; but we will use Petri nets rather than the recent CHemical Abstract Machines (CHAM's) of [4], which have some unnecessary features and have been less well studied.

The semantics of the "small π-calculus" is presented in [13] (and in [14]) in a novel way, inspired by the CHAM. First a so-called structural congruence is defined on the process terms that is meant to capture the fact that two processes are structurally, i.e., statically, the same. In other words, the processes have the same flow graph (see [12, 16]), which roughly means that they can be decomposed into the same concurrent subprocesses. Then, an interleaving transition system is given in which structurally congruent processes are given the same behaviour, by definition. This separation of "physical" structure and behaviour is intuitively clear, and simplifies the transition system to a large extent. In particular, the commutativity and associativity of parallel composition are handled on the structural level, and replication is even handled completely on the structural level (reducing it to parallel composition).

In this paper, and its sequel [9], we wish to put forward the general idea that the multiset (or Petri net) semantics of a process algebra should also be used to express the structure of the processes: we would like two processes to have the same structure if and only if they have the same multiset semantics. Intuitively, the syntax of process terms that is needed to describe a multiset of concurrent subprocesses, should not be present in that multiset; the syntactic laws needed to describe multisets should in fact be sound, and preferably even complete.

We define one "large" multiset transition system (or Petri net), called Mπ, and we define a (compositional) semantic mapping that associates a state of Mπ with each process of the small π-calculus. Thus, the meaning of a process is a multiset (or marking of the net Mπ); intuitively, it is the multiset of all its concurrent subprocesses. The Petri net Mπ has one type of transition only, which corresponds to the basic action in the small π-calculus: a communication between two subprocesses. In this way Mπ is similar to the "object-oriented" interleaving transition system of the small π-calculus. Our main results on this semantics are:

(A) the semantic mapping is a strong bisimulation between the interleaving transition system of the small π-calculus and the multiset transition system Mπ, and

(B) if two processes of the small π-calculus are structurally congruent, then they have the same semantics in Mπ.

Result (A) ensures that a process and its corresponding multiset in Mπ have the same (interleaving) behaviour. Result (B) means that two processes that have the same structure also have the same multiset semantics. The converse of (B) does not hold and thus the laws of structural congruence of the small π-calculus are sound, but not complete relative to the multiset semantics. We suggest that the structural congruence should be extended in such a way that (B) does hold in both directions. In fact, we present four natural laws for structural congruence that are not valid in the small π-calculus. After extending structural congruence with these new laws, we show that results (A) and (B) still hold, and we show in

[9] that result (B) now even holds in both directions (i.e., the laws of structural congruence are now sound and complete relative to the multiset semantics). Moreover, it is shown in [9] that structural congruence is decidable (which does not seem to be known without the extension); since structural congruence is a static, "physical", property, its decidability is desirable.

(A′) The semantic mapping is a strong bisimulation between the interleaving transition system of the extended small π-calculus and the multiset transition system Mπ,

(B′) two processes of the extended small π-calculus are structurally congruent if and only if they have the same semantics in Mπ, and

(C′) it is decidable, for two processes of the extended small π-calculus, whether or not they are structurally congruent (i.e., have the same semantics in Mπ).

We note that results (A) and (A′) imply that each process of the small π-calculus is strongly bisimilar with itself in the extended small π-calculus. Thus the proposed extension of structural congruence does not change the behaviour of the processes (modulo strong bisimilarity).

Our semantics satisfies, in a certain sense, the two requirements for a Petri net semantics to be a "good" semantics as formulated by Olderog in [18]. The first requirement is that the interleaving semantics should be "retrievable" from the Petri net semantics in the sense that they should be strongly bisimilar; this is exactly result (A). The second requirement is that the Petri net semantics should reflect the "intended concurrency". Although its formalization in [18] is not applicable here, we believe intuitively that it is fulfilled, i.e., we claim that the Petri net semantics that we provide for the small π-calculus, models the concurrency that is meant to be present in the processes of the π-calculus. We note here that the two CHAM's proposed in [4] for the small π-calculus both fail to satisfy the first requirement, due to their heating rules. In our opinion, they also fail to satisfy the second requirement, the first CHAM because it uses cooling rules to implement α-conversion, and the second CHAM because it has a non-distributed name server. The second CHAM is strongly related to our multiset semantics; in fact, our semantic mapping may be viewed as a one-stroke implementation of its heating rules.

The semantic mapping associates a multiset S in Mπ with each process term P. Intuitively, S is the decomposition of P into its concurrent subprocesses. Thus, the semantic mapping is similar to the decomposition mappings of [6, 18]; however, as opposed to [6, 18], it also decomposes all (guarded) subterms of P. Another difference is that it decomposes into multisets rather than sets; in fact, the replication operation forces us to consider non-safe Petri nets. The advantages of non-safe nets have been pointed out in [10]; runs (or "processes") of such nets have been studied in [11] (see also [7]). A third, essential, difference with [6, 18] is that it is impossible to reconstruct P from S; in fact, such a reconstruction would contradict the desired result (B). A final difference with [6, 18] is discussed in general in what follows.

One may ask what type of semantics is given to the π-calculus in this paper. Is it compositional, denotational, operational, etc.? Although these terms are necessarily vague, we believe that we have given a new type (or subtype) of semantics. Let us be more precise. On the one hand, any semantics that associates an initialized transition system (i.e., a transition system together with one of its states) with each process, is operational by nature, because transition systems are operational. On the other hand, different types of such operational semantics can be distinguished, e.g., the following: system-compositional operational semantics, structured operational semantics (or SOS), and state-compositional operational semantics. In [18] the first two types are just called compositional and operational semantics, respectively. The third type is introduced here and is the type of our π-calculus semantics. Note that our structural requirement (that two processes have the same structure iff they have the same semantics) is orthogonal to this classification.

System-compositional semantics. The mapping that associates an initialized transition system with each process is compositional, i.e., the syntactical operations on processes are interpreted as semantical operations on initialized transition systems.

Structured operational semantics (SOS). The semantics is defined by specifying one "large" transition system and a mapping that associates a state of that transition system with each process, thus initializing it. The set of transitions of the transition system is defined recursively, by SOS rules and axioms that follow the syntax of processes. The mapping is defined in some "natural" way.

State-compositional semantics. As in the previous case, the semantics is defined by specifying one "large" transition system and a mapping that associates a state of that transition system with each process. The transition system is defined in some "natural" way. The mapping is compositional, i.e., the syntactical operations on processes are interpreted as semantical operations on states of the transition system.

Clearly, the first and third type of semantics are very similar, because they are both compositional. However, they are essentially different: in one case the operations are on transition systems, and in the other case on states of a fixed transition system.

It should also be clear from the above description that the second and third type of semantics are very similar, and, in fact, the distinction between these two types is not clear cut. For the usual SOS interleaving semantics of process algebra's the mapping that associates a state with each process, is trivial; in fact, it is the identity (because the processes themselves are the states of the transition system). For the usual SOS Petri net semantics of process algebra's that mapping is more complicated, as it associates a multiset with each process. It is the decomposition mapping of [6, 18] mentioned above. This mapping is compositional, but usually on part of the syntax only (typically on parallel composition, restriction, and relabeling). Thus, the usual SOS Petri net semantics are in fact inbetween the second and third type of semantics. The semantics of the π-calculus in this paper is of the third type. The mapping is compositional

on the whole syntax, and the transition system is defined without SOS rules.

The structure of this paper is as follows. Section 2 contains the definition of the small π-calculus. In Section 3 we discuss multisets and multiset transition systems. In Section 4 we define the multiset π-calculus Mπ and discuss its basic properties. Section 5 contains the definition of the multiset semantics of the small π-calculus, i.e., the relation between process terms and the states of Mπ. In Section 6 we state the main results. Section 7 contains the proofs of results (A), (B), (A$'$), and the only-if part of (B$'$). The if part of result (B$'$), and result (C$'$), are proved in the sequel paper [9].

A previous version of this paper was published in the Proceedings of CONCUR'93 [8].

2 The Small π-Calculus

We briefly recall the definition of the small π-calculus from [13].

Let N be an infinite set of names. The context-free syntax for process terms is as follows (where we use a comma rather than | to separate alternatives):

$$P \quad ::= \quad \overline{x}y.P \;\;,\;\; x(y).P \;\;,\;\; \mathbf{0} \;\;,\;\; P \mid P \;\;,\;\; !P \;\;,\;\; (\nu y)P$$

with $x, y \in \mathrm{N}$. The strings $\overline{x}y$ and $x(y)$ are called *guards*. The y in $x(y).P$ and in $(\nu y)P$ binds all free occurrences of y in P. We denote by $\mathrm{fn}(P)$ the set of names that occur free in process P; thus, $\mathrm{fn}(P) \subseteq \mathrm{N}$.

Informally, process $\overline{x}y.P$ sends the name y along the link x and then continues as process P, and process $x(y).P$ receives any name z along the link x and then continues as process $P[z/y]$, where $P[z/y]$ denotes the result of substituting z for all free occurrences of y in P (renaming bound names where necessary, as usual). Parallel composition of processes P and Q is denoted $P \mid Q$ as usual in CCS, and $\mathbf{0}$ is the inactive process; $(\nu y)P$ is the restriction of y to P, denoted $P\backslash y$ in CCS. Finally, the process $!P$ is the replication of process P and abbreviates $P \mid P \mid P \mid \cdots$.

Structural congruence, denoted $\equiv$, is the smallest congruence over the set of all process terms such that

(α) $\quad P \equiv Q$ whenever P and Q are α-convertible,

(1.1) $\quad P \mid \mathbf{0} \equiv P$,

(1.2) $\quad P \mid Q \equiv Q \mid P$,

(1.3) $\quad P \mid (Q \mid R) \equiv (P \mid Q) \mid R$,

(2.1) $\quad (\nu x)(\nu y)P \equiv (\nu y)(\nu x)P$,

(2.2) $\quad (\nu x)P \equiv P$
provided $x \notin \mathrm{fn}(P)$,

(2.3) $(\nu x)(P \mid Q) \equiv P \mid (\nu x)Q$
provided $x \notin \mathrm{fn}(P)$, and

(3.1) $!P \equiv P \mid !P$.

As usual, two terms are α-convertible if they are the same modulo a renaming of bound names. In [13], structural law (2.2) is stated as its special case $(\nu x)\mathbf{0} \equiv \mathbf{0}$. However, as shown in [13], the general case can easily be proved from this, together with structural laws (1.1) and (2.3): if $x \notin \mathrm{fn}(P)$, then $(\nu x)P \equiv (\nu x)(P \mid \mathbf{0}) \equiv P \mid (\nu x)\mathbf{0} \equiv P \mid \mathbf{0} \equiv P$.

The states of the transition system of the small π-calculus are all process terms, and its transition relation $\rightarrow$ (also called *reduction*) is the smallest relation satisfying the following (SOS) axiom and rules:

COM: $x(y).P \mid \overline{x}z.Q \rightarrow P[z/y] \mid Q$

PAR: If $P \rightarrow P'$, then $P \mid Q \rightarrow P' \mid Q$

RES: If $P \rightarrow P'$, then $(\nu y)P \rightarrow (\nu y)P'$

STRUCT: If $Q \equiv P$, $P \rightarrow P'$, and $P' \equiv Q'$, then $Q \rightarrow Q'$.

The COM rule formalizes the synchronous communication between two processes along link x, during which the name z is sent from Q to P (where it replaces y). The STRUCT rule embodies the idea that structurally congruent processes have the same behaviour.

In [13] the replication $!P$ of process P is said to "stand for $P \mid P \mid \cdots$, as many concurrent instances of P as you like". This is a rather vague statement, which might seem to refer to the structural law $!P \equiv P \mid !P$ as allowing one to take as many instances of P as one likes off $!P$. However, "taking off" is something dynamic rather than static. The only way in which the above law can be true for the static structure of $!P$ is by viewing $!P$ as infinitely many instances of P in parallel. Thus, we conclude that "as many as you like" means "infinitely many".

3 Multiset Transition Systems

Since $!P$ represents a (countably) infinite number of concurrent copies of P, i.e., $!P = P \mid P \mid P \mid \cdots$, we need multisets in which elements may occur infinitely many times. We will consider countable multisets only.

A *multiset* S is a countable set D_S together with a mapping $\phi_S : D_S \rightarrow \mathbf{N} \cup \{\omega\}$, where $\mathbf{N} = \{1, 2, 3, \ldots\}$ is the set of all positive integers. For $d \in D_S$, $\phi_S(d)$ is the *multiplicity* of d in S. By convention, we also define $\phi_S(x) = 0$ for any object x that is not in D_S. Note that every countable set A can be viewed as the multiset S with $D_S = A$ and $\phi_S(a) = 1$ for every $a \in A$. For a set D,

S is said to be a *multiset over* D if $D_S \subseteq D$. Intuitively, a multiset S is a bag that contains $\phi_S(d)$ copies of each element d of D_S (where "ω copies" means: a countably infinite number of copies). The *union* of multisets S and S', denoted $S \cup S'$, is defined by: $D_{S \cup S'} = D_S \cup D_{S'}$ and $\phi_{S \cup S'}(d) = \phi_S(d) + \phi_{S'}(d)$, where, for $n \in \mathbf{N} \cup \{0, \omega\}$, $n + \omega = \omega + n = \omega$. A similar definition holds for arbitrary (countable) union: if S_i is a multiset for every $i \in I$, where I is a countable index set, then their union $S = \bigcup_{i \in I} S_i$ is defined by: $D_S = \bigcup_{i \in I} D_{S_i}$ and $\phi_S(d) = \sum_{i \in I} \phi_{S_i}(d)$, where, for $n_i \in \mathbf{N} \cup \{0, \omega\}$, $\sum_{i \in I} n_i$ is defined as follows: let $\mathrm{Pos} = \{i \in I \mid n_i \neq 0\}$; if Pos is infinite, then $\sum_{i \in I} n_i = \omega$, and otherwise $\sum_{i \in I} n_i = \sum_{i \in \mathrm{Pos}} n_i$. Note that if $n_i = \omega$ for some $i \in I$, then $\sum_{i \in I} n_i = \omega$. In fact, $\sum_{i \in I} n_i$ is defined in such a way that if the cardinality of a set A_i is n_i (where ω stands for cardinality $\aleph_0$) and the A_i are mutually disjoint, then the cardinality of $\bigcup_{i \in I} A_i$ is $\sum_{i \in I} n_i$ (where, intuitively, A_i is a set of n_i copies of some element of a multiset). From this it can easily be seen that multiset union is commutative and associative.

A *transition system* is a tuple $(Q, \rightarrow)$ where Q is a set of states and $\rightarrow$ is a binary relation on Q, called the transition relation. Note that we restrict ourselves here to unlabeled transition systems.

A *multiset transition system* (or multiset rewriting system) is a tuple (D, T) where D is a set and T is a set of basic transitions, which are pairs (S_1, S_2) where S_1 and S_2 are multisets over D. Such a multiset transition system (D, T) is viewed as a transition system $(Q, \rightarrow)$ where Q is the set of all multisets over D, and $\rightarrow$ is the smallest relation on Q that contains the relation T and is closed under multiset union. More precisely, the transition relation $\rightarrow$ is defined by the following axiom and rule (where S_1, S_2, S are multisets over D):

(1) if $(S_1, S_2) \in T$, then $S_1 \rightarrow S_2$, and

(2) if $S_1 \rightarrow S_2$, then $S_1 \cup S \rightarrow S_2 \cup S$.

Axiom (1) is the "reaction law" and rule (2) is the "chemical law" of the CHemical Abstract Machine of Berry and Boudol ([4]). Intuitively, a basic transition $(S_1, S_2) \in T$ models a local communication between the elements of S_1, as a result of which they turn into the elements of S_2. The above two laws express the locality or "context-freeness" of such communications: if there is a basic transition from S_1 to S_2 then that transition can take place in any "context" S. In fact, it follows from associativity of multiset union that, for states S' and S'', $S' \rightarrow S''$ if and only if there exist S, S_1, S_2 such that $S' = S_1 \cup S$, $S'' = S_2 \cup S$, and $(S_1, S_2) \in T$.

This last fact means that a multiset transition system (D, T) is the same as a (possibly infinite) Place/Transition net, see [19]. D is the set of places of the net and T is its set of transitions. Multisets of places are the markings of the net, and the chemical law defines the firing of a (basic) transition in a marking. Hence, the transition system $(Q, \rightarrow)$ is the case graph of the Place/Transition net.

For the reader familiar with Petri nets, we note that our multiset transition system is in fact slightly more general than the P/T net because it allows mul-

tiplicity ω. Since the multiplicities of the places in the multisets of a (basic) transition correspond to the weights on the arcs of the net, it corresponds to a P/T net that allows infinitely many tokens on a place, and infinite weights on the arcs. In such a net the firing of a transition $t = (S_1, S_2)$ in a given marking for which it is enabled, need not lead to a unique new marking, because $\omega - \omega$ may have any value. However, if no element of S_1 has multiplicity ω (i.e., the weights of the input arcs of t are not ω), then the new marking is unique. In this paper we will only consider such transitions (but elements of S_2 may well have multiplicity ω).

Apart from union we need the following operation on multisets. For a multiset S over D and a mapping $h : D \to E$, the *image of* S *under* h, denoted $h(S)$, is the multiset over E defined by: $D_{h(S)} = h(D_S)$ and $\phi_{h(S)}(e) = \sum_{d \in h^{-1}(e)} \phi_S(d)$.

Note that $h(D_S)$ is the usual image of the set D_S under h. In fact, for sets S and T, the notations $h(S)$ and $S \cup T$ are ambiguous, because they may be interpreted as the usual operations on sets or as the operations on multisets defined above (because every set is also viewed as a multiset). And in general the results will be different. In what follows it should always be clear from the context which of the two interpretations is meant.

We will use two basic properties of the image operation. First, if S and T are multisets over D, and $h : D \to E$, then $h(S \cup T) = h(S) \cup h(T)$. A similar result holds for countable unions: if S_i is a multiset over D for every $i \in I$, then $h(\bigcup_{i \in I} S_i) = \bigcup_{i \in I} h(S_i)$. Second, if S is a multiset over D, $h_1 : D \to E$, and $h_2 : E \to F$, then $(h_2 \circ h_1)(S) = h_2(h_1(S))$.

4 The Multiset π-Calculus

We now introduce a specific multiset transition system, that we call the multiset π-calculus, denoted Mπ.

Let N be the set of names of the π-calculus. Let New be an uncountably infinite set of *new names*, disjoint with N. These new names will be used to cope with restriction. The notion of guard is extended accordingly: from now on, a *guard* is a string of the form $x(y)$ or $\overline{x}z$ with $x, z \in \mathrm{N} \cup \mathrm{New}$ and $y \in \mathrm{N}$. To get rid of α-conversion we will employ a variant of the idea of De Bruijn ([5], see also [3, Appendix C]) to use numbers instead of bound names, in a systematic way to be explained. For this reason we define a *schematic guard* to be a string of the form $x(-)$ or $\overline{x}z$ with $x, z \in \mathrm{N} \cup \mathrm{New} \cup \mathbf{N}$.

Taking over the chemical terminology of [2, 4], the states of Mπ are called *solutions*, which are multisets of *molecules*, defined in a mutually recursive way as follows:

(1) A solution is a multiset over the set of molecules.

(2) A molecule is a pair $g.S$, where g is a schematic guard and S is a solution.

Note that the recursion starts with the empty solution $\emptyset$. Note also that the ordered pair (g, S) is written as $g.S$ in order to remain closer to the syntax of the small π-calculus.

The operation of forming a multiset stands for (possibly infinite) parallel composition, which is commutative and associative. The schematic guards should be interpreted in the same way as in the small π-calculus, where it is understood that each occurrence of a number i in a solution or molecule is bound by the ith schematic guard $x(-)$ that has i in its scope, counting from the outside in (as opposed to [5] where one counts from the inside out). As an example, $S = \{x(-).\{1(-).\{\overline{y}2.\emptyset, \overline{1}z.\emptyset\}\}, y(-).\{\overline{1}z.\emptyset\}, \overline{x}y.\emptyset\}$ is a solution consisting of the three molecules $m_1 = x(-).\{1(-).\{\overline{y}2.\emptyset, \overline{1}z.\emptyset\}\}$, $m_2 = y(-).\{\overline{1}z.\emptyset\}$, and $m_3 = \overline{x}y.\emptyset$. Also, $m_1 = x(-).S_1$ where $S_1 = \{m_4\}$ with $m_4 = 1(-).S_4$ and S_4 is the solution consisting of the two molecules $\overline{y}2.\emptyset$ and $\overline{1}z.\emptyset$. A picture of S is given in Fig.1. The multiset S will be the semantics of (among others) the process term $x(u).u(v).(\overline{y}v.\mathbf{0} \mid \overline{u}z.\mathbf{0}) \mid (y(w).\overline{w}z.\mathbf{0} \mid \overline{x}y.\mathbf{0})$ which also shows how to interpret the binding of the numbers. Note that there exist solutions in which numbers occur that are not bound to any schematic guard; such solutions will not be the semantics of process terms. A molecule $g.S$ is similar to a molecule with a "subsolution" S in the CHemical Abstract Machine of [4]. However we stress here that we will not adopt the "membrane law" of the CHAM, i.e., there cannot be any activity within a subsolution.

More formally, the sets Sol and Mol of solutions and molecules, respectively, are the smallest sets such that (1) $\mathrm{Sol} \subseteq \mathcal{P}_m(\mathrm{Mol})$, and (2) $\mathrm{Mol} \subseteq \mathrm{G} \times \mathrm{Sol}$, where $\mathcal{P}_m(\mathrm{Mol})$ is the set of all multisets over Mol, and G is the set of all schematic guards. More informally, one may also view a molecule as an unordered rooted directed tree that has no infinite directed path starting at the root, and of which the, countably many, nodes are labeled with schematic guards. A solution can be viewed in the same way, except that the root is unlabeled. The operation $g.S$ consists of labeling the root of S with g, and the operation of forming a multiset S of molecules consists of taking a new root for S and connecting it with the roots of all (mutually disjoint) molecules.

To define the transitions of Mπ we need to define the notion of substitution of names for names in solutions. Since solutions do not contain bound variables, such a substitution amounts to a straightforward change of names. We also need to increase and decrease the numbers that occur in a solution; this is a straightforward change of numbers. From the point of view of labeled trees these operations consist of a simple relabeling of the nodes. With the recursive definition of solutions and molecules, they can be defined recursively, as follows. Let, in general, f be an arbitrary mapping of $\mathrm{N} \cup \mathrm{New} \cup \mathbf{N}$ into itself. Then f is recursively extended to a mapping f^σ on solutions and a mapping f^μ on molecules as follows:

$f^\sigma(S) = f^\mu(S)$, the image of S under f^μ (restricted to D_S),

$f^\mu(x(-).S) = f(x)(-).f^\sigma(S)$ and $f^\mu(\overline{x}z.S) = \overline{f(x)}f(z).f^\sigma(S)$.

For the (usual) definition of the image of a multiset under a mapping, see Section 3. Whenever there will be no danger for confusion, we will denote both f^σ and f^μ also by f. Note that, by the first basic property of multisets mentioned at the end of Section 3, for solutions S_i, $f(S_1 \cup S_2) = f(S_1) \cup f(S_2)$ and $f(\bigcup_{i \in I} S_i) = \bigcup_{i \in I} f(S_i)$.

Fig. 1. Solutions S and S' consisting of three and two molecules, respectively.

We now define the mappings inc, dec, $[z/y]$: $\mathrm{N} \cup \mathrm{New} \cup \mathbf{N} \to \mathrm{N} \cup \mathrm{New} \cup \mathbf{N}$, for $z, y \in \mathrm{N} \cup \mathrm{New} \cup \mathbf{N}$, as follows (where $[z/y]$ is written postfix):

$\mathrm{inc}(x) = x + 1$ for $x \in \mathbf{N}$, $\mathrm{inc}(x) = x$ for $x \in \mathrm{N} \cup \mathrm{New}$,

$\mathrm{dec}(x) = x - 1$ for $x \in \mathbf{N} - \{1\}$, $\mathrm{dec}(x) = x$ for $x \in \mathrm{N} \cup \mathrm{New} \cup \{1\}$,

$x[z/y] = z$ for $x = y$, $x[z/y] = x$ for $x \neq y$.

Thus, for a solution S, $S[z/y]$ is the result of substituting z for all "occurrences" of y in S. As mentioned before, since S does not have bound names, this just means that every y is replaced by z. Also, $\mathrm{inc}(S)$ is the result of increasing all numbers that "occur" in S by one, and, similarly, $\mathrm{dec}(S)$ is the result of decreasing all numbers that "occur" in S (except 1) by one.

Consider two mappings $f, h : \mathrm{N} \cup \mathrm{New} \cup \mathbf{N} \to \mathrm{N} \cup \mathrm{New} \cup \mathbf{N}$. It is straightforward to prove that $h(f(S)) = (h \circ f)(S)$ for every solution S (and similarly for every molecule). More formally, the extension of the composition of f and h is the composition of the extensions of f and h, i.e., $(h \circ f)^\sigma = h^\sigma \circ f^\sigma$ and $(h \circ f)^\mu =$

$h^\mu \circ f^\mu$. The proof is by induction on the recursive definition of solutions and molecules. First, for a solution S, assuming that $(h \circ f)^\mu(m) = h^\mu(f^\mu(m))$ for every molecule $m \in D_S$, one shows that $(h \circ f)^\sigma(S) = h^\sigma(f^\sigma(S))$. This follows from the second basic property of multisets mentioned at the end of Section 3. Then, for a schematic guard g and a solution S, assuming that $(h \circ f)^\sigma(S) = h^\sigma(f^\sigma(S))$, one shows that $(h \circ f)^\mu(g.S) = h^\mu(f^\mu(g.S))$.

This fact can be used to prove a number of elementary properties of the functions inc, dec, and $[z/y]$ in a straightforward way. Thus, for instance, for every solution S, $\mathrm{dec}(\mathrm{inc}(S)) = (\mathrm{dec} \circ \mathrm{inc})(S) = S$, because dec $\circ$ inc is the identity function id on $\mathrm{N} \cup \mathrm{New} \cup \mathbf{N}$, and, clearly, $\mathrm{id}(S) = S$. Note also that $S[y/y] = S$, because $[y/y] = \mathrm{id}$. As another example, $S[y/x][z/y] = S[z/y][z/x]$ because $u[y/x][z/y] = u[z/y][z/x]$ for every $u \in \mathrm{N} \cup \mathrm{New} \cup \mathbf{N}$. Similarly, if $v \neq x$ and $y \neq x$, then $S[u/x][v/y] = S[v/y][u[v/y]/x]$, and if additionally $u \neq y$ then $S[u/x][v/y] = S[v/y][u/x]$. In this way, all the usual substitution laws can easily be shown. Since there are no bound names in solutions, there is also no need for α-conversion; this makes the proofs straightforward.

The *multiset π-calculus* Mπ is now defined to be the multiset transition system (Mol,T), where Mol is the set of all molecules and T consists of all the following basic transitions (that model communication between molecules):

$$\{x(-).S, \overline{x}z.S'\} \rightarrow \mathrm{dec}(S[z/1]) \cup S'$$

where $x, z \in \mathrm{N} \cup \mathrm{New}$ and S, S' are solutions. Note that the left-hand side multiset of a basic transition is always a set of two molecules. Note that, by the chemical law, the transition relation of Mπ consists of all transitions

$$\{x(-).S, \overline{x}z.S'\} \cup S'' \rightarrow \mathrm{dec}(S[z/1]) \cup S' \cup S''.$$

As an example, $S \rightarrow S'$ where S is the above example solution and $S' = \{y(-).\{\overline{y}1.\emptyset, \overline{y}z.\emptyset\}, y(-).\{\overline{1}z.\emptyset\}\}$, see Fig.1.

For $x \in \mathrm{N} \cup \mathrm{New}$, $y \in \mathrm{N}$, and a solution S, the *guarded molecule* $x(y).S$ is defined by $x(y).S = x(-).\mathrm{inc}(S)[1/y]$. The example solution S can now be written as $\{x(u).\{u(v).\{\overline{y}v.\emptyset, \overline{u}z.\emptyset\}\}, y(w).\{\overline{w}z.\emptyset\}, \overline{x}y.\emptyset\}$ which is closer to the process term $x(u).u(v).(\overline{y}v.\mathbf{0} \mid \overline{u}z.\mathbf{0}) \mid (y(w).\overline{w}z.\mathbf{0} \mid \overline{x}y.\mathbf{0})$ of which it is the meaning.

Since, for $y \in \mathrm{N}$ and $z \in \mathrm{N} \cup \mathrm{New}$, $\mathrm{dec}(\mathrm{inc}(S)[1/y][z/1]) = \mathrm{dec}(\mathrm{inc}(S)[z/y])$ $= \mathrm{dec}(\mathrm{inc}(S[z/y])) = S[z/y]$, it follows that for all $y \in \mathrm{N}$, $x, z \in \mathrm{N} \cup \mathrm{New}$, and solutions S, S',

$$\{x(y).S, \overline{x}z.S'\} \rightarrow S[z/y] \cup S'$$

is a basic transition of Mπ. Consequently, the transition relation of Mπ contains all transitions

$$\{x(y).S, \overline{x}z.S'\} \cup S'' \rightarrow S[z/y] \cup S' \cup S''.$$

These transitions are actually the ones that will simulate those of the small π-calculus. Note that the effect of such a transition is that the solutions S and S' that are hidden (by guards) in the molecules $x(y).S$ and $\overline{x}z.S'$, are added to

the soup S'' of molecules that is the current state of $\mathrm{M}\pi$, after changing S into $S[z/y]$.

The next lemma shows the relation between substitution and the behaviour of solutions in $\mathrm{M}\pi$.

Lemma 1. *Let S, T be solutions, and let $y, z \in \mathrm{N} \cup \mathrm{New}$. If $S \to T$ in $M\pi$, then $S[z/y] \to T[z/y]$ in $M\pi$.*

Proof. The transition $S \to T$ is of the form

$$\{x(-).S_1, \overline{x}w.S_2\} \cup S_3 \to \mathrm{dec}(S_1[w/1]) \cup S_2 \cup S_3.$$

We have to show that there is a transition

$$\{x'(-).S_1[z/y], \overline{x'}w'.S_2[z/y]\} \cup S_3[z/y] \to \mathrm{dec}(S_1[w/1])[z/y] \cup S_2[z/y] \cup S_3[z/y],$$

where $x' = x[z/y]$ and $w' = w[z/y]$. This follows from the fact that there is a basic transition

$$\{x'(-).S_1[z/y], \overline{x'}w'.S_2[z/y]\} \to \mathrm{dec}(S_1[z/y][w'/1]) \cup S_2[z/y]$$

and that $\mathrm{dec}(S_1[w/1])[z/y] = \mathrm{dec}(S_1[w/1][z/y]) = \mathrm{dec}(S_1[z/y][w[z/y]/1])$. □

We define the set $\mathrm{occ}(S) \subseteq \mathrm{N} \cup \mathrm{New} \cup \mathbf{N}$ of *occurrences* in a solution S, and similarly $\mathrm{occ}(m)$ for a molecule m, recursively as follows:

$\mathrm{occ}(S) = \bigcup\{\mathrm{occ}(m) \mid m \in D_S\}$,

$\mathrm{occ}(x(-).S) = \mathrm{occ}(S) \cup \{x\}$, and $\mathrm{occ}(\overline{x}z.S) = \mathrm{occ}(S) \cup \{x, z\}$.

We denote $\mathrm{occ}(S) \cap (\mathrm{N} \cup \mathrm{New})$ by $\mathrm{fn}(S)$, the set of (free) *names of* S. Note that in fact all names that occur in S are free. We denote $\mathrm{fn}(S) \cap \mathrm{New}$ by $\mathrm{new}(S)$, the set of *new names of* S. Note that $\mathrm{occ}(S)$ is countable, and hence $\mathrm{new}(S)$ is a proper subset of New (because we have assumed New to be uncountable). For the example solution $S = \{x(-).\{1(-).\{\overline{y}2.\emptyset, \overline{1}z.\emptyset\}\}, y(-).\{\overline{1}z.\emptyset\}, \overline{x}y.\emptyset\}$, we have $\mathrm{occ}(S) = \{x, y, z, 1, 2\}$, $\mathrm{fn}(S) = \{x, y, z\}$, and $\mathrm{new}(S) = \emptyset$.

Obviously, $\mathrm{occ}(S_1 \cup S_2) = \mathrm{occ}(S_1) \cup \mathrm{occ}(S_2)$ and similarly for countable unions. For a function f from $\mathrm{N} \cup \mathrm{New} \cup \mathbf{N}$ to itself and a solution S, it can easily be shown, by induction on the recursive definition of S, that $\mathrm{occ}(f(S)) = f(\mathrm{occ}(S))$. This implies, e.g., that $\mathrm{occ}(S[z/y]) = (\mathrm{occ}(S) - \{y\}) \cup \{z \mid y \in \mathrm{occ}(S)\}$. Similar properties hold for fn and new.

It is now straightforward to show that, during the behaviour of $\mathrm{M}\pi$, the set of free names in the solution cannot increase: for solutions S and T, if $S \to T$ in $\mathrm{M}\pi$, then $\mathrm{fn}(T) \subseteq \mathrm{fn}(S)$. For instance, for the example transition $S \to S'$ (see Fig.1), $\mathrm{fn}(S') = \{y, z\} \subseteq \mathrm{fn}(S) = \{x, y, z\}$.

Let f, h be functions from $\mathrm{N} \cup \mathrm{New} \cup \mathbf{N}$ to itself. Obviously, the value of $f(S)$ is determined by the values of f on $\mathrm{occ}(S)$ only. In other words, if the restrictions of f and h to $\mathrm{occ}(S)$ are equal, then $f(S) = h(S)$. This can easily be shown by induction on S. It implies, e.g., that $S[z/y] = S$ if $y \notin \mathrm{occ}(S)$ (because $[z/y]$ is the identity on $\mathrm{occ}(S)$). As another example, if $y \notin \mathrm{occ}(S)$ then $S[y/x][z/y] = S[z/x]$, and in particular $S[y/x][x/y] = S$.

We observe now that almost all basic transitions of Mπ are of the form

$$\{x(y).S, \overline{x}z.S'\} \rightarrow S[z/y] \cup S'.$$

To prove this, consider a basic transition

$$\{x(-).S, \overline{x}z.S'\} \rightarrow \mathrm{dec}(S[z/1]) \cup S'$$

and assume that $\mathrm{occ}(S) \cap \mathrm{N} \subset \mathrm{N}$, i.e., there exists $y \in \mathrm{N}$ such that $y \notin \mathrm{occ}(S)$. For the semantics of the small π-calculus it actually suffices to consider such solutions. We now show that the above basic transition equals

$$\{x(y).T, \overline{x}z.S'\} \rightarrow T[z/y] \cup S'$$

for $T = \mathrm{dec}(S[y/1])$. Note first that $\mathrm{inc}(T) = S[y/1]$ because $1 \notin \mathrm{occ}(S[y/1])$. Now $x(y).T = x(-).\mathrm{inc}(T)[1/y] = x(-).S[y/1][1/y] = x(-).S$ because $y \notin \mathrm{occ}(S)$. Also, $T[z/y] = \mathrm{dec}(S[y/1][z/y]) = \mathrm{dec}(S[z/1])$ because $y \notin \mathrm{occ}(S)$.

Finally we list some properties of guarded molecules $u(x).S$ that are easy to prove. Let $u, y, z \in \mathrm{N} \cup \mathrm{New}$ and $x \in \mathrm{N}$. Then

$\mathrm{fn}(u(x).S) = (\mathrm{fn}(S) - \{x\}) \cup \{u\}$,
if $y \notin \mathrm{fn}(S) - \{x\}$, then $(u(x).S)[z/y] = u[z/y](x).S$,
if $y \neq x$ and $z \neq x$, then $(u(x).S)[z/y] = u[z/y](x).S[z/y]$,
if $w \in \mathrm{N} - \mathrm{fn}(S)$, then $u(x).S = u(w).S[w/x]$.

As an example we prove the last property: $u(w).S[w/x] = u(-).\mathrm{inc}(S[w/x])[1/w] = u(-).\mathrm{inc}(S)[w/x][1/w] = u(-).\mathrm{inc}(S)[1/x] = u(x).S$.

5 Semantics of the π-Calculus

We have cheated a little in the introduction by saying that we will associate one solution S of Mπ with every process term P. In fact, in order to treat restriction properly, we are forced to associate infinitely many solutions S with P. However, all these solutions can be obtained from each other by a bijective renaming of the new names occurring in them.

We will write $P \Rightarrow S$ to indicate that the solution S of Mπ is associated to the process term P. Thus, we will define a relation $\Rightarrow$ between the process terms of the small π-calculus and the solutions of the multiset π-calculus.

The *semantic relation* $\Rightarrow$ is defined to be the smallest relation that satisfies the following compositional requirements:

(S0) $\mathbf{0} \Rightarrow \emptyset$

(S1) If $P_1 \Rightarrow S_1$ and $P_2 \Rightarrow S_2$, then $P_1 \mid P_2 \Rightarrow S_1 \cup S_2$
provided $\mathrm{new}(S_1) \cap \mathrm{new}(S_2) = \emptyset$

(S2) If $P \Rightarrow S$, then $(\nu x)P \Rightarrow S[n/x]$
provided $n \in \mathrm{New} - \mathrm{new}(S)$

(S3) If $P \Rightarrow S$ and g is a guard, then $g.P \Rightarrow \{g.S\}$

(S4) If $P \Rightarrow S_i$ for all $i \in \mathbf{N}$, then $!P \Rightarrow \bigcup_{i\in\mathbf{N}} S_i$
provided $\mathrm{new}(S_i) \cap \mathrm{new}(S_j) = \emptyset$ for all $i \neq j$.

In (S3), g is of course an "ordinary" guard, i.e., one that contains no new names. Note that by rule (S3) $\overline{x}z.P$ is translated into $\{\overline{x}z.S\}$, and $x(y).P$ is translated into $\{x(y).S\}$ which abbreviates $\{x(-).\mathrm{inc}(S)[1/y]\}$. Rules (S1) and (S4) translate parallel composition and replication into multiset union, and rule (S0) translates the inactive process into the empty multiset.

It is straightforward to prove, by induction on the structure of P, that if $P \Rightarrow S$, then $\mathrm{fn}(P) = \mathrm{fn}(S) \cap \mathrm{N}$. The names in $\mathrm{new}(S) = \mathrm{fn}(S) \cap \mathrm{New}$ are all introduced to get rid of the restrictions in P, through rule (S2). Intuitively, the name x of $(\nu x)P$, of which the scope is restricted to P, is replaced by a completely new name n with a global scope.

Process terms P and Q are *multiset congruent*, denoted $P \equiv_m Q$, if $\{S \mid P \Rightarrow S\} = \{S \mid Q \Rightarrow S\}$, i.e., if P and Q have the same multiset semantics in Mπ. It is immediate from the compositional definition of the semantic relation $\Rightarrow$ that $\equiv_m$ is a congruence.

Let us consider some examples of the semantic relation $P \Rightarrow S$. As a first example, let $P =$

$$(x(w).w(y).w(z).\overline{y}z.\mathbf{0} \mid x(w).\mathbf{0}) \mid ((\nu u)(\overline{x}u.\overline{u}y_1.\overline{u}z_1.\mathbf{0}) \mid (\nu u)(\overline{x}u.\overline{u}y_2.\overline{u}z_2.\mathbf{0})).$$

Then $P \Rightarrow S$, where $S =$

$$\{x(w).\{w(y).\{w(z).\{\overline{y}z.\emptyset\}\}\}, x(w).\emptyset, \overline{x}n.\{\overline{n}y_1.\{\overline{n}z_1.\emptyset\}\}, \overline{x}m.\{\overline{m}y_2.\{\overline{m}z_2.\emptyset\}\}\},$$

and n and m are distinct new names. Note that, in more detail, $S =$

$$\{x(-).\{1(-).\{1(-).\{\overline{2}3.\emptyset\}\}\}, x(-).\emptyset, \overline{x}n.\{\overline{n}y_1.\{\overline{n}z_1.\emptyset\}\}, \overline{x}m.\{\overline{n}y_2.\{\overline{m}z_2.\emptyset\}\}\}.$$

As a second example, we observe the big difference between $!(\nu x)Q$ and $(\nu x)(!Q)$. Let $Q = \overline{y}x.x(z).\mathbf{0}$. Then

$$!(\nu x)Q = !(\nu x)(\overline{y}x.x(z).\mathbf{0}) \Rightarrow \{\overline{y}n_1.\{n_1(z).\emptyset\}, \overline{y}n_2.\{n_2(z).\emptyset\}, \overline{y}n_3.\{n_3(z).\emptyset\}, \ldots\}$$

where the n_i are distinct new names, $i \in \mathrm{N}$. But

$$(\nu x)(!Q) = (\nu x)(!\overline{y}x.x(z).\mathbf{0}) \Rightarrow \{\overline{y}n.\{n(z).\emptyset\}, \overline{y}n.\{n(z).\emptyset\}, \overline{y}n.\{n(z).\emptyset\}, \ldots\}$$

where n is a new name. Thus, the domain of the first solution contains infinitely many molecules, each with multiplicity 1, and the domain of the second solution contains one molecule, with multiplicity ω.

The following two lemma's show that the semantic relation is in fact a total function modulo a bijective renaming of the new names. We start with totality.

Lemma 2. *For every P there exists S such that $P \Rightarrow S$.*

Proof. See Section 7. □

It is easy to show that for process terms P that do not contain any restrictions, this S is unique (and $\mathrm{new}(S) = \emptyset$). In general, functionality is modulo bijective renamings of new names. If S is a solution and $f : \mathrm{new}(S) \rightarrow \mathrm{New}$, then we tacitly assume that f is extended to a function from $\mathrm{N} \cup \mathrm{New} \cup \mathbf{N}$ into itself by defining $f(x) = x$ for every x not in $\mathrm{new}(S)$. Then, as usual, $f(S)$ denotes the value for S of the extension of this f as defined in Section 4.

Lemma 3. *If $P \Rightarrow S$, then $P \Rightarrow S'$ if and only if there exists a bijection $f : \mathrm{new}(S) \rightarrow \mathrm{new}(S')$ such that $f(S) = S'$.*

Proof. See Section 7. □

Clearly, the relation between solutions S and S' that holds when $f(S) = S'$ for some bijection $f : \mathrm{new}(S) \rightarrow \mathrm{new}(S')$, is an equivalence relation. The above lemma's show that the semantic relation associates an equivalence class of solutions with each process term. Intuitively, the solutions in one such equivalence class are just "isomorphic copies" of each other. Thus, rule (S1) can be understood as follows: if S_1 is the meaning of P_1 and S_2 is the meaning of P_2, then $S_1' \cup S_2'$ is the meaning of $P_1 \mid P_2$, where S_1' and S_2' are "disjoint copies" of S_1 and S_2, respectively. Similarly for (S4): if S is the meaning of P, then $\bigcup_{i \in \mathbf{N}} S_i$ is the meaning of $!P$, where the S_i are mutually disjoint copies of S.

The next, important, lemma shows that the semantic relation is compositional with respect to substitution: $\{S \mid P[z/y] \Rightarrow S\} = \{S[z/y] \mid P \Rightarrow S\}$. This implies that the multiset congruence $\equiv_m$ is also a congruence with respect to substitution.

Lemma 4. *Let $y, z \in \mathrm{N}$.*
(1) If $P \Rightarrow S$, then $P[z/y] \Rightarrow S[z/y]$.
(2) If $P[z/y] \Rightarrow S'$, then there exists S such that $P \Rightarrow S$ and $S' = S[z/y]$.

Proof. See Section 7. □

6 Main Results

(A) The semantic relation $\Rightarrow$ is a strong bisimulation between the transition systems of the small π-calculus and the multiset π-calculus Mπ.

(B) For process terms P and Q of the small π-calculus, if $P \equiv Q$ (i.e., P and Q are structurally congruent), then $P \equiv_m Q$ (i.e., P and Q are multiset congruent).

The proofs of these results are given in Section 7. In Section 7 we also prove (in Lemma 7) that $!(P \mid Q) \equiv_m !P \mid !Q$, $!!P \equiv_m !P$, $!\mathbf{0} \equiv_m \mathbf{0}$, and $(\nu x)g.P \equiv_m g.(\nu x)P$ provided x does not occur in g. It should be clear from Lemma's 2 and 3 that $\equiv_m = (\Rightarrow)^{-1} \circ (\Rightarrow)$. Thus, it follows from (A) and the well-known closure of strong bisimulations under composition and inverse, that the multiset congruence $\equiv_m$ is a strong bisimulation: if $P \equiv_m Q$, then P and Q are strongly bisimilar in the transition system of the small π-calculus. From that point of view

one may then safely add $P \equiv Q$ to the laws of structural congruence (in the sense that the STRUCT rule will not change their behaviour). Let the *extended* small π-calculus be defined by adding the following laws to the definition of structural congruence:

(3.2) $\quad !(P \mid Q) \equiv !P \mid !Q,$

(3.3) $\quad !!P \equiv !P,$

(3.4) $\quad !\mathbf{0} \equiv \mathbf{0}$, and

(2.4) $\quad (\nu x)g.P \equiv g.(\nu x)P$
provided x does not occur in g.

Note that, altogether, structural laws (3.1)-(3.4) deal with replication, and structural laws (2.1)-(2.4) deal with restriction. The remaining laws deal with α-conversion (law (α)) and parallel composition (laws (1.1)-(1.3)).

(A$'$) The semantic relation $\Rightarrow$ is a strong bisimulation between the transition systems of the extended small π-calculus and the multiset π-calculus Mπ.

(B$'$) For process terms P and Q of the extended small π-calculus, $P \equiv Q$ (i.e., P and Q are structurally congruent) if and only if $P \equiv_m Q$ (i.e., P and Q are multiset congruent).

(C$'$) For process terms P and Q of the extended small π-calculus, it is decidable whether or not $P \equiv Q$ (i.e., whether or not $P \equiv_m Q$).

The proofs of (A$'$) and the only-if part of (B$'$) are given in Section 7. The if-part of (B$'$), and (C$'$), are the main results of [9].

Note that the three new replication laws (3.2)-(3.4) do not hold in the (ordinary) small π-calculus, because structural congruence preserves the number of replications. If one adds the first law only, then structural congruence preserves the nesting depth of replication, and so the other two laws still do not hold. However, intuitively all three structural laws should hold, if one recalls that $!P$ stands for ω (i.e., infinitely many) copies of P. Clearly, ω copies of P and Q is the same as ω copies of P and ω copies of Q, ω copies of ω copies of P is the same as ω copies of P (because $\omega \cdot \omega = \omega$, viewing ω as aleph zero), and ω copies of nothing is nothing. The original structural law (3.1) for replication is based on the fact that $1+\omega = \omega$. It is also easy to see that the new restriction law (2.4) does not hold in the small π-calculus; it should hold for the same reasons that structural law (2.3) should hold.

Extending structural congruence implies, by the STRUCT rule, that also the transition relation $\rightarrow$ of the small π-calculus is extended. Thus, in the transition system of the extended small π-calculus there are more transitions possible than in the transition system of the ordinary small π-calculus. However, by results (A) and (A$'$), the relation $\equiv_m = (\Rightarrow)^{-1} \circ (\Rightarrow)$ is a strong bisimulation between these two transition systems. Hence, since $P \equiv_m P$, every process term P is

strongly bisimilar to itself in the two transition systems. Thus, modulo strong bisimilarity, the behaviour of all processes is the same in both transition systems.

We now illustrate results (A) and (B) with an example. P and Q are two friends that want to go fishing. P knows a nice spot where there are many fish, and he sends Q the address. Then they pack their bags, meet at the address, and throw out there fishing rods. There is also a supply B of two (empty) bags. Finally, there is a jealous friend R who would like to go fishing with P in the place of Q, but does not know the address. In the following formalization, we abbreviate, for any name a, the guards $a(d)$ and $\overline{a}d$ by a and $\overline{a}$, respectively, where d is a dummy name: a and $\overline{a}$ are communications in which the name that is sent, is irrelevant. Now $P = \overline{s}.\mathbf{0} \mid s.(\nu y)(b.y.\overline{r}.\mathbf{0} \mid \overline{x}y.\mathbf{0})$, $Q = x(y).b.\overline{y}.\overline{r}.\mathbf{0}$, $B = \overline{b}.\mathbf{0} \mid \overline{b}.\mathbf{0}$, and $R = \overline{y}.\overline{r}.\mathbf{0}$.

Consider the process term $P_1 = P \mid Q \mid B \mid R$. Then $P_1 \Rightarrow S_1$ with $S_1 = \{\overline{s}.\emptyset, s.\{b.\{n.\{\overline{r}.\emptyset\}\}, \overline{x}n.\emptyset\}, x(-).\{b.\{\overline{1}.\{\overline{r}.\emptyset\}\}\}, \overline{b}.\emptyset, \overline{b}.\emptyset, \overline{y}.\overline{r}.\emptyset\}$. Let us now consider the behaviour of P_1 and S_1 in their respective transition systems. First, there is just one possible transition: P sends the address to Q. This is the transition $P_1 \rightarrow P_2$, where $P_2 = \mathbf{0} \mid (\nu y)(b.y.\overline{r}.\mathbf{0} \mid \overline{x}y.\mathbf{0}) \mid Q \mid B \mid R$. Now $P_2 \equiv P_2' \equiv P_2''$, where $P_2' = (\nu y)(b.y.\overline{r}.\mathbf{0} \mid \overline{x}y.\mathbf{0}) \mid x(z).b.\overline{z}.\overline{r}.\mathbf{0} \mid B \mid R$ by the structural laws (1.1) and (α), removing $\mathbf{0}$ and α-converting Q, and $P_2'' = (\nu y)(b.y.\overline{r}.\mathbf{0} \mid \overline{x}y.\mathbf{0} \mid x(z).b.\overline{z}.\overline{r}.\mathbf{0} \mid B) \mid R$ by the structural law (3.1), extending the scope of (νy). Hence (by STRUCT) also $P_1 \rightarrow P_2''$. In Mπ the corresponding transition is $S_1 \rightarrow S_2$, where $S_2 = \{b.\{n.\{\overline{r}.\emptyset\}\}, \overline{x}n.\emptyset, x(-).\{b.\{\overline{1}.\{\overline{r}.\emptyset\}\}\}, \overline{b}.\emptyset, \overline{b}.\emptyset, \overline{y}.\overline{r}.\emptyset\}$. Clearly, $P_2 \Rightarrow S_2$, $P_2' \Rightarrow S_2$, and $P_2'' \Rightarrow S_2$. This illustrates both results (A) and (B).

Next, several things can happen in parallel: P packs his bags, and Q first receives the address and then also starts packing. The transition in which Q receives the address is in the small π-calculus $P_2'' \rightarrow P_3$, where $P_3 = (\nu y)(b.y.\overline{r}.\mathbf{0} \mid \mathbf{0} \mid b.\overline{y}.\overline{r}.\mathbf{0} \mid B) \mid R$, and $P_3 \equiv P_3' = (\nu y)(b.y.\overline{r}.\mathbf{0} \mid b.\overline{y}.\overline{r}.\mathbf{0} \mid \overline{b}.\mathbf{0} \mid \overline{b}.\mathbf{0}) \mid R$ by structural law (1.1). In Mπ the transition is $S_2 \rightarrow S_3$, where $S_3 = \{b.\{n.\{\overline{r}.\emptyset\}\}, b.\{\overline{n}.\{\overline{r}.\emptyset\}\}, \overline{b}.\emptyset, \overline{b}.\emptyset, \overline{y}.\overline{r}.\emptyset\}$. Then there are two transitions in which P and Q pack their bags. In the small π-calculus they are $P_3' \rightarrow^2 P_5$, where $P_5 = (\nu y)(y.\overline{r}.\mathbf{0} \mid \overline{y}.\overline{r}.\mathbf{0}) \mid R$ and in Mπ they are $S_3 \rightarrow^2 S_5$, where $S_5 = \{n.\{\overline{r}.\emptyset\}, \overline{n}.\{\overline{r}.\emptyset\}, \overline{y}.\overline{r}.\emptyset\}$. Finally, P and Q meet at the address of P and throw out their fishing rods. In the small π-calculus this is the transition $P_5 \rightarrow P_6$, where $P_6 = (\nu y)(\overline{r}.\mathbf{0} \mid \overline{r}.\mathbf{0}) \mid R \equiv \overline{r}.\mathbf{0} \mid \overline{r}.\mathbf{0} \mid R$ and in Mπ it is $S_5 \rightarrow S_6$, where $S_6 = \{\overline{r}.\emptyset, \overline{r}.\emptyset, \overline{y}.\overline{r}.\emptyset\}$.

One of the nice aspects of multiset transition systems (i.e., Petri nets) is that there is a natural notion of parallel computation of the system (unfortunately often called a "process" in Petri net terminology), see, e.g., [19, 11, 7]. A picture of such a parallel computation neatly shows the concurrencies and dependencies between the basic transitions that occur during a run of the system. In Fig.2 a picture is shown of the parallel computation of the example solution S_1, described above. The ovals represent molecules, and the rectangles represent the basic transitions. For a basic transition (S, S') represented by a rectangle, there are directed edges from the ovals representing the molecules in S to the rectangle,

and from the rectangle to the ovals representing the molecules in S'. The parallel computation naturally induces a partial order on the five basic transitions that occur during the computation: the address is sent by P before it is received by Q and before P packs his bags, Q receives the address before he packs his bags, and P and Q pack their bags before they meet at the address. There is no order between P packing bags and Q receiving the address, and there is no order between P and Q packing bags. Thus, the parallel computation gives a much better insight in what happens "in real life" than the above sequence of transitions.

7 Proofs

Lemma 2. *For every P there exists S such that $P \Rightarrow S$.*

Proof. We prove that for every process term P and every countably infinite set $W \subseteq \mathrm{New}$ there exists a solution S such that $P \Rightarrow S$ and $\mathrm{new}(S) \subseteq W$. This is done by induction on the structure of P. It is obvious for $P = \mathbf{0}$ by (S0).

$P = P_1 \mid P_2$. Partition W into two countably infinite sets W_1 and W_2. By induction there exist S_1 and S_2 such that $P_1 \Rightarrow S_1$ and $P_2 \Rightarrow S_2$ with $\mathrm{new}(S_1) \subseteq W_1$ and $\mathrm{new}(S_2) \subseteq W_2$. Since $\mathrm{new}(S_1) \cap \mathrm{new}(S_2) = \emptyset$, we obtain, by (S1), that $P_1 \mid P_2 \Rightarrow S_1 \cup S_2$ with $\mathrm{new}(S_1 \cup S_2) = \mathrm{new}(S_1) \cup \mathrm{new}(S_2) \subseteq W$.

$P = (\nu x)Q$. Take $n \in W$ arbitrarily. By induction there exists S such that $Q \Rightarrow S$ and $\mathrm{new}(S) \subseteq W - \{n\}$. Hence, by (S2), $(\nu x)Q \Rightarrow S[n/x]$ and $\mathrm{new}(S[n/x]) \subseteq \mathrm{new}(S) \cup \{n\} \subseteq W$.

$P = g.Q$. By induction $Q \Rightarrow S$ with $\mathrm{new}(S) \subseteq W$. Then, by (S3), $g.P \Rightarrow \{g.S\}$ with $\mathrm{new}(\{g.S\}) = \mathrm{new}(S) \subseteq W$.

$P = !Q$. Partition W into a countably infinite number of countably infinite sets W_i, $i \in \mathbf{N}$. By induction there exist S_i such that $Q \Rightarrow S_i$ and $\mathrm{new}(S_i) \subseteq W_i$. By (S4), $!Q \Rightarrow \bigcup_{i \in \mathbf{N}} S_i$ and $\mathrm{new}(\bigcup_{i \in \mathbf{N}} S_i) \subseteq \bigcup_i W_i = W$. □

Lemma 3. *If $P \Rightarrow S$, then $P \Rightarrow S'$ if and only if there exists a bijection $f : \mathrm{new}(S) \rightarrow \mathrm{new}(S')$ such that $f(S) = S'$.*

Proof. The proof is by induction on the definition of the semantic relation $\Rightarrow$, i.e., by induction on the structure of P. The case $\mathbf{0} \Rightarrow \emptyset$ is obvious.

$P_1 \mid P_2 \Rightarrow S_1 \cup S_2$ with $P_1 \Rightarrow S_1$, $P_2 \Rightarrow S_2$, and $\mathrm{new}(S_1) \cap \mathrm{new}(S_2) = \emptyset$. Note that $\mathrm{new}(S_1 \cup S_2) = \mathrm{new}(S_1) \cup \mathrm{new}(S_2)$.
(Only if). Assume that $P \Rightarrow S'$. By (S1), $S' = S'_1 \cup S'_2$ with $P_1 \Rightarrow S'_1$, $P_2 \Rightarrow S'_2$, and $\mathrm{new}(S'_1) \cap \mathrm{new}(S'_2) = \emptyset$. By induction there are bijections $f_i : \mathrm{new}(S_i) \rightarrow \mathrm{new}(S'_i)$ such that $f_i(S_i) = S'_i$ for $i = 1, 2$. Let $f = f_1 \cup f_2$. Then f is a bijection from $\mathrm{new}(S)$ to $\mathrm{new}(S')$. Since f_i is the restriction of f to $\mathrm{new}(S_i)$, $f_i(S_i) = f(S_i)$ and so $f(S) = f(S_1 \cup S_2) = f(S_1) \cup f(S_2) = f_1(S_1) \cup f_2(S_2) = S'_1 \cup S'_2 = S'$.
(If) Assume that $f : \mathrm{new}(S_1) \cup \mathrm{new}(S_2) \rightarrow \mathrm{new}(S')$ is a bijection with $f(S_1 \cup S_2) = S'$. Let f_i be the restriction of f to $\mathrm{new}(S_i)$ and let $S'_i = f_i(S_i)$. As above, $f_i(S_i) = f(S_i)$ and so $S' = f(S_1) \cup f(S_2) = S'_1 \cup S'_2$. Clearly, f_i is a bijection from

$\mathrm{new}(S_i)$ to $f_i(\mathrm{new}(S_i)) = \mathrm{new}(f_i(S_i)) = \mathrm{new}(S_i')$. Hence, by induction, $P_i \Rightarrow S_i'$. Since $f_1(\mathrm{new}(S_1)) \cap f_2(\mathrm{new}(S_2)) = \emptyset$, this implies by (S1) that $P \Rightarrow S_1' \cup S_2' = S'$.

$(\nu x)P_1 \Rightarrow S_1[n/x]$ with $P_1 \Rightarrow S_1$ and $n \in \mathrm{New} - \mathrm{new}(S_1)$. This case is in fact quite similar to the previous one.

(Only if). Assume that $P \Rightarrow S'$. By (S2), $S' = S_1'[m/x]$ with $P_1 \Rightarrow S_1'$ and $m \in \mathrm{New} - \mathrm{new}(S_1')$. By induction there is a bijection $f_1 : \mathrm{new}(S_1) \rightarrow \mathrm{new}(S_1')$ such that $f_1(S_1) = S_1'$. We first consider the case that $x \notin \mathrm{fn}(P_1)$. Then also $x \notin \mathrm{fn}(S_1)$, because $P_1 \Rightarrow S_1$ implies that $\mathrm{fn}(P_1) = \mathrm{fn}(S_1) \cap \mathrm{N}$. Similarly $x \notin \mathrm{fn}(S_1')$. Hence, in this case, $S' = S_1'$ and $S = S_1$ which finishes the proof. Assume now that $x \in \mathrm{fn}(P_1)$. Then $\mathrm{new}(S) = \mathrm{new}(S_1[n/x]) = \mathrm{new}(S_1) \cup \{n\}$ and $\mathrm{new}(S') = \mathrm{new}(S_1'[m/x]) = \mathrm{new}(S_1') \cup \{m\}$. Let $f = f_1 \cup \{(n, m)\}$. Then f is a bijection from $\mathrm{new}(S)$ to $\mathrm{new}(S')$. Moreover $f(S) = f(S_1[n/x]) = f(S_1)[m/x] = f_1(S_1)[m/x] = S_1'[m/x] = S'$.

(If) Assume that $f : \mathrm{new}(S_1[n/x]) \rightarrow \mathrm{new}(S')$ is a bijection with $f(S_1[n/x]) = S'$. If $x \notin \mathrm{fn}(P_1)$, then $S_1[n/x] = S_1$. Hence, by induction, $P_1 \Rightarrow S'$ and so, by (S2), $P \Rightarrow S'[m/x]$ for any $m \in \mathrm{New} - \mathrm{new}(S')$ (note that $\mathrm{new}(S')$ is a proper subset of New). Since $x \notin \mathrm{fn}(S')$, $P \Rightarrow S'$. Assume now that $x \in \mathrm{fn}(P_1)$. Then $f : \mathrm{new}(S_1) \cup \{n\} \rightarrow \mathrm{new}(S')$. Let f_1 be the restriction of f to $\mathrm{new}(S_1)$, let $m = f(n)$, and let $S_1' = f_1(S_1)$. Then $S' = f(S_1[n/x]) = f_1(S_1)[m/x] = S_1'[m/x]$. Clearly, f_1 is a bijection from $\mathrm{new}(S_1)$ to $f_1(\mathrm{new}(S_1)) = \mathrm{new}(f_1(S_1)) = \mathrm{new}(S_1')$. Hence, by induction, $P_1 \Rightarrow S_1'$. Since $m \notin f_1(\mathrm{new}(S_1))$, we obtain from (S2) that $(\nu x)P_1 \Rightarrow S_1'[m/x]$ and so $P \Rightarrow S'$.

The case that $P = g.P_1$ is straightforward (because the guard g does not contain new names), and the case that $P = !P_1$ is very similar to the case that $P = P_1 \mid P_2$. They are left to the reader. □

Lemma 4. *Let $y, z \in \mathrm{N}$.*
(1) If $P \Rightarrow S$, then $P[z/y] \Rightarrow S[z/y]$.
(2) If $P[z/y] \Rightarrow S'$, then there exists S such that $P \Rightarrow S$ and $S' = S[z/y]$.

Proof. After proving (1), it is easy to show (2) with the previous lemma's, as follows. Assume that $P[z/y] \Rightarrow S'$. By Lemma 2 there exists S_1 such that $P \Rightarrow S_1$. Then, by (1), $P[z/y] \Rightarrow S_1[z/y]$. Lemma 3 then implies that $f(S_1[z/y]) = S'$ for some bijection $f : \mathrm{new}(S_1[z/y]) \rightarrow \mathrm{new}(S')$. Note that $\mathrm{new}(S_1[z/y]) = \mathrm{new}(S_1)$ because $z, y \in \mathrm{N}$. Let $S = f(S_1)$. Then $\mathrm{new}(S) = f(\mathrm{new}(S_1))$. Hence $f : \mathrm{new}(S_1) \rightarrow \mathrm{new}(S)$. By Lemma 3 (in the other direction) $P \Rightarrow S$. Moreover $S[z/y] = f(S_1)[z/y] = f(S_1[z/y]) = S'$.

It remains to show (1). The proof is based on the fact that the substitution operation of Mπ satisfies all the usual laws of substitution, as shown in Section 4.

The proof is by induction on the length (i.e., the number of symbols) of P. We consider the usual cases of the syntactical form of P. Let $P \Rightarrow S$. For $P = \mathbf{0}$ we obtain from (S0) that $S = \emptyset$ and so $P[z/y] = \mathbf{0} \Rightarrow \emptyset = S[z/y]$.

$P = P_1 \mid P_2$. From (S1) follows that $S = S_1 \cup S_2$ with $P_1 \Rightarrow S_1$, $P_2 \Rightarrow S_2$, and $\mathrm{new}(S_1) \cap \mathrm{new}(S_2) = \emptyset$. By induction (because P_1 and P_2 are shorter than P), $P_1[z/y] \Rightarrow S_1[z/y]$ and $P_2[z/y] \Rightarrow S_2[z/y]$. Since $\mathrm{new}(S_i[z/y]) = \mathrm{new}(S_i)$, we obtain from (S1) that $P[z/y] = P_1[z/y] \mid P_2[z/y] \Rightarrow S_1[z/y] \cup S_2[z/y] = (S_1 \cup S_2)[z/y] = S[z/y]$.

$P = (\nu x)P'$. By (S2), $S = S'[n/x]$ with $P' \Rightarrow S'$ and $n \in \mathrm{New} - \mathrm{new}(S')$. We consider three cases.
(1) $y \notin \mathrm{fn}(P) = \mathrm{fn}((\nu x)P')$. Since $P \Rightarrow S$, also $y \notin \mathrm{fn}(S)$. Then $P[z/y] = P \Rightarrow S = S[z/y]$.
(2) $y \in \mathrm{fn}((\nu x)P')$ and $z \neq x$. Note that $y \neq x$. By induction $P'[z/y] \Rightarrow S'[z/y]$. Then, by (S2), $((\nu x)P')[z/y] = (\nu x)P'[z/y] \Rightarrow S'[z/y][n/x] = S'[n/x][z/y] = S[z/y]$.
(3) $y \in \mathrm{fn}((\nu x)P')$ and $z = x$. Then $((\nu x)P')[z/y] = (\nu w)P'[w/x][z/y]$ for some $w \in \mathrm{N}$ with $w \notin \mathrm{fn}(P')$ and $w \neq x$. Note that $w \neq y$ and $w \notin \mathrm{fn}(S')$. By induction $P'[w/x] \Rightarrow S'[w/x]$. Again by induction (because $P'[w/x]$ has the same length as P'), $P'[w/x][z/y] \Rightarrow S'[w/x][z/y]$. Then, by (S2), $P[z/y] = (\nu w)P'[w/x][z/y] \Rightarrow S'[w/x][z/y][n/w] = S'[w/x][n/w][z/y] = S'[n/x][z/y] = S[z/y]$.

$P = \overline{u}x.P'$. By (S3), $S = \{\overline{u}x.S'\}$ with $P' \Rightarrow S'$. Then, by induction and (S3), $(\overline{u}x.P')[z/y] = \overline{u[z/y]}x[z/y].P'[z/y] \Rightarrow \{\overline{u[z/y]}x[z/y].S'[z/y]\} = \{\overline{u}x.S'\}[z/y] = S[z/y]$.

$P = u(x).P'$. By (S3), $S = \{u(x).S'\}$ with $P' \Rightarrow S'$. We consider three cases. We will use the substitution laws for guarded molecules that are listed at the end of Section 4.
(1) $y \notin \mathrm{fn}((\nu x)P')$, i.e., $y \notin \mathrm{fn}(P') - \{x\}$. Hence also $y \notin \mathrm{fn}(S') - \{x\}$. Then $P[z/y] = (u(x).P')[z/y] = u[z/y](x).P' \Rightarrow \{u[z/y](x).S'\} = \{u(x).S'\}[z/y] = S[z/y]$.
(2) $y \in \mathrm{fn}((\nu x)P')$ and $z \neq x$. Note that $y \neq x$. By induction and (S3), $(u(x).P')[z/y] = u[z/y](x).P'[z/y] \Rightarrow \{u[z/y](x).S'[z/y]\} = \{u(x).S'\}[z/y] = S[z/y]$.
(3) $y \in \mathrm{fn}((\nu x)P')$ and $z = x$. Then $(u(x).P')[z/y] = u[z/y](w).P'[w/x][z/y]$ for some $w \in \mathrm{N}$ with $w \notin \mathrm{fn}(P')$ and $w \neq x$. Note that $w \neq y$ and $w \notin \mathrm{fn}(S')$. By induction (twice) $P'[w/x][z/y] \Rightarrow S'[w/x][z/y]$. Then, by (S3), $P[z/y] = u[z/y](w).P'[w/x][z/y] \Rightarrow \{u[z/y](w).S'[w/x][z/y]\} = \{u(w).S'[w/x]\}[z/y] = \{u(x).S'\}[z/y] = S[z/y]$.

$P = !P'$. By (S4), $S = \bigcup_{i \in \mathbf{N}} S_i$ with $P' \Rightarrow S_i$ and $\mathrm{new}(S_i) \cap \mathrm{new}(S_j) = \emptyset$. Then, by induction and (S4), $(!P')[z/y] = !(P'[z/y]) \Rightarrow \bigcup_{i \in \mathbf{N}} S_i[z/y] = (\bigcup_{i \in \mathbf{N}} S_i)[z/y] = S[z/y]$. □

The proof of (A) is split into two parts: the left and the right part of the bisimulation.

(AL) If $P \Rightarrow S$ and $P \rightarrow P'$, then there exists S' such that $S \rightarrow S'$ and $P' \Rightarrow S'$.

(AR) If $P \Rightarrow S$ and $S \rightarrow S'$, then there exists P' such that $P \rightarrow P'$ and $P' \Rightarrow S'$.

In the proof of (AL) we need Lemma's 1 and 4, and we need (B) to handle the STRUCT rule. For this reason we start with the proof of (B).

Theorem B. *If $P \equiv Q$, then $P \equiv_m Q$.*

Proof. Since $\equiv_m$ is a congruence, to prove that $\equiv \subseteq \equiv_m$ it suffices to show that $\equiv_m$ satisfies the laws (1)-(8) of structural congruence.

(1) We have to show that $P \equiv_\alpha Q$ implies $P \equiv_m Q$, where $\equiv_\alpha$ denotes α-conversion of process terms. Since $\equiv_m$ is a congruence, it follows from the properties of $\equiv_\alpha$ that it suffices to prove the following two special cases (a) and (b).

(a) $x(y).P \equiv_m x(z).P[z/y]$ with $z \notin \mathrm{fn}(P)$ and $z \neq y$. Assume first that $x(y).P \Rightarrow S$. By (S3), $S = \{x(y).T\}$ with $P \Rightarrow T$. By Lemma 4(1), $P[z/y] \Rightarrow T[z/y]$. Hence, by (S3), $x(z).P[z/y] \Rightarrow \{x(z).T[z/y]\}$. Since $z \notin \mathrm{fn}(T)$, we have $\{x(z).T[z/y]\} = \{x(y).T\} = S$. Hence $x(z).P[z/y] \Rightarrow S$. Assume now that $x(z).P[z/y] \Rightarrow S$. By (S3), $S = \{x(z).T'\}$ with $P[z/y] \Rightarrow T'$. By Lemma 4(2), there exists a solution T such that $P \Rightarrow T$ and $T[z/y] = T'$. By (S3), $x(y).P \Rightarrow \{x(y).T\}$ and, since $z \notin \mathrm{fn}(T)$, $\{x(y).T\} = \{x(z).T[z/y]\} = \{x(z).T'\} = S$.

(b) $(\nu y)P \equiv_m (\nu z)P[z/y]$ with $z \notin \mathrm{fn}(P)$ and $z \neq y$. The proof is similar to the one of (a). Assume first that $(\nu y)P \Rightarrow S$. By (S2), $S = T[n/y]$ with $P \Rightarrow T$ and $n \in \mathrm{New} - \mathrm{new}(T)$. By Lemma 4(1), $P[z/y] \Rightarrow T[z/y]$. Hence, by (S2), $(\nu z)P[z/y] \Rightarrow T[z/y][n/z] = T[n/y] = S$. Assume now that $(\nu z)P[z/y] \Rightarrow S$. By (S2), $S = T'[n/z]$ with $P[z/y] \Rightarrow T'$ and $n \in \mathrm{New} - \mathrm{new}(T')$. By Lemma 4(2), there exists T such that $P \Rightarrow T$ and $T[z/y] = T'$. Note that $\mathrm{new}(T) = \mathrm{new}(T')$. Hence, by (S2), $(\nu y)P \Rightarrow T[n/y]$. Also $T[n/y] = T[z/y][n/z] = T'[n/z] = S$.

(2) We show that $P \mid \mathbf{0} \equiv_m P$. If $P \Rightarrow S$ then, by (S0) and (S1), $P \mid \mathbf{0} \Rightarrow S \cup \emptyset = S$. If $P \mid \mathbf{0} \Rightarrow S$ then $S = S_1 \cup S_2$ with $P \Rightarrow S_1$ and $\mathbf{0} \Rightarrow S_2$. Hence $S_2 = \emptyset$ and so $S = S_1$ and $P \Rightarrow S$.

(3) Next we show that $P \mid Q \equiv_m Q \mid P$. Let $P \mid Q \Rightarrow S$. By (S1), $S = S_1 \cup S_2$ with $P \Rightarrow S_1$, $Q \Rightarrow S_2$, and $\mathrm{new}(S_1) \cap \mathrm{new}(S_2) = \emptyset$. Then, by (S1), $Q \mid P \Rightarrow S_2 \cup S_1 = S_1 \cup S_2 = S$ by the commutativity of multiset union. The other direction is symmetric.

(4) The proof of $P \mid (Q \mid R) \equiv_m (P \mid Q) \mid R$ is similarly based on the associativity of multiset union.

(5) To show that $!P \equiv_m P \mid !P$, consider first $!P \Rightarrow S$. By (S4), $S = \bigcup_{i \in \mathbf{N}} S_i$ with $P \Rightarrow S_i$ and all $\mathrm{new}(S_i)$ are disjoint. Then $!P \Rightarrow \bigcup_{i \in \mathbf{N}} S_{i+1}$ by (S4). Hence $P \mid !P \Rightarrow S_1 \cup \bigcup_{i \in \mathbf{N}} S_{i+1} = \bigcup_{i \in \mathbf{N}} S_i = S$ by (S1). Next consider $P \mid !P \Rightarrow S$. Then $S = S_1 \cup T$ with $P \Rightarrow S_1$, $!P \Rightarrow T$, and $\mathrm{new}(S_1) \cap \mathrm{new}(T) = \emptyset$. Hence $T = \bigcup_{i \in \mathbf{N}} S_{i+1}$ with $P \Rightarrow S_{i+1}$ and the $\mathrm{new}(S_{i+1})$ are mutually disjoint. This implies that the $\mathrm{new}(S_i)$, $i \in \mathbf{N}$, are mutually disjoint, and so $!P \Rightarrow \bigcup_{i \in \mathbf{N}} S_i = S_1 \cup \bigcup_{i \in \mathbf{N}} S_{i+1} = S_1 \cup T = S$.

(7) We first show that $(\nu x)P \equiv_m P$ if $x \notin \mathrm{fn}(P)$. If $P \Rightarrow S$, then take any $n \in \mathrm{New} - \mathrm{new}(S)$ (note that $\mathrm{new}(S)$ is a proper subset of New). Then $(\nu x)P \Rightarrow S[n/x]$, but since $x \notin \mathrm{fn}(S)$, $S[n/x] = S$. On the other hand, if $(\nu x)P \Rightarrow S$, then $S = T[n/x]$ with $P \Rightarrow T$ and $n \in \mathrm{New} - \mathrm{new}(T)$. Since $x \notin \mathrm{fn}(T)$, $T[n/x] = T$ and so $S = T$ and $P \Rightarrow S$.

(6) To show that $(\nu x)(\nu y)P \equiv_m (\nu y)(\nu x)P$, with $x \neq y$, we may now assume, by the proof of case (7), that $x, y \in \mathrm{fn}(P)$. Consider $(\nu x)(\nu y)P \Rightarrow S$. Then $S = T[n/x]$ with $(\nu y)P \Rightarrow T$ and $n \in \mathrm{New} - \mathrm{new}(T)$. Hence $T = U[m/y]$ with $P \Rightarrow U$ and $m \in \mathrm{New} - \mathrm{new}(U)$. Since $y \in \mathrm{fn}(P)$, also $y \in \mathrm{fn}(U)$ and hence $m \in \mathrm{new}(T)$. Thus, $m \neq n$. Now $(\nu x)P \Rightarrow U[n/x]$ because $\mathrm{new}(U) \subseteq \mathrm{new}(T)$. And $(\nu y)(\nu x)P \Rightarrow U[n/x][m/y]$ because $\mathrm{new}(U[n/x]) \subseteq \mathrm{new}(U) \cup \{n\}$. Since

$U[n/x][m/y] = U[m/y][n/x] = T[n/x] = S$, we have $(\nu y)(\nu x)P \Rightarrow S$. The other part follows by symmetry.

(8) We have to show that $(\nu x)(P \mid Q) \equiv_m P \mid (\nu x)Q$ if $x \notin \mathrm{fn}(P)$. Again, by the proof of case (7), we may assume that $x \in \mathrm{fn}(Q)$. Consider $(\nu x)(P \mid Q) \Rightarrow S$. Then $S = T[n/x]$ with $P \mid Q \Rightarrow T$ and $n \in \mathrm{New} - \mathrm{new}(T)$. Then $T = S_1 \cup S_2$ with $P \Rightarrow S_1$, $Q \Rightarrow S_2$, and $\mathrm{new}(S_1) \cap \mathrm{new}(S_2) = \emptyset$. Now $(\nu x)Q \Rightarrow S_2[n/x]$ and $P \mid (\nu x)Q \Rightarrow S_1 \cup S_2[n/x]$. Since $x \notin \mathrm{fn}(S_1)$, $S_1[n/x] = S_1$ and so $S_1 \cup S_2[n/x] = (S_1 \cup S_2)[n/x] = T[n/x] = S$. Hence $P \mid (\nu x)Q \Rightarrow S$. The other part is similar, using the fact that $x \in \mathrm{fn}(Q)$. Let $P \mid (\nu x)Q \Rightarrow S$. Then $S = S_1 \cup T$ with $P \Rightarrow S_1$, $(\nu x)Q \Rightarrow T$, and $\mathrm{new}(S_1) \cap \mathrm{new}(T) = \emptyset$. Then $T = S_2[n/x]$ with $Q \Rightarrow S_2$ and $n \in \mathrm{New} - \mathrm{new}(S_2)$. Now $P \mid Q \Rightarrow S_1 \cup S_2$ and $(\nu x)(P \mid Q) \Rightarrow (S_1 \cup S_2)[n/x]$; note that since $x \in \mathrm{fn}(S_2)$, $n \in \mathrm{new}(T)$ and so $n \notin \mathrm{new}(S_1)$. As above, $(S_1 \cup S_2)[n/x] = S_1 \cup S_2[n/x] = S_1 \cup T = S$, and so $(\nu x)(P \mid Q) \Rightarrow S$. □

Theorem AL. *If $P \Rightarrow S$ and $P \rightarrow P'$, then there exists S' such that $S \rightarrow S'$ and $P' \Rightarrow S'$.*

Proof. Induction on the definition of $P \rightarrow P'$.
(COM) $P \rightarrow P'$ is $x(y).P_1 \mid \overline{x}z.P_2 \rightarrow P_1[z/y] \mid P_2$. By (S1), $S = S_1 \cup S_2$ with $x(y).P_1 \Rightarrow S_1$, $\overline{x}z.P_2 \Rightarrow S_2$ and $\mathrm{new}(S_1) \cap \mathrm{new}(S_2) = \emptyset$. Then, by (S3), $S_1 = \{x(y).S_1'\}$ and $S_2 = \{\overline{x}z.S_2'\}$ with $P_1 \Rightarrow S_1'$ and $P_2 \Rightarrow S_2'$. Clearly $\mathrm{new}(S_i') = \mathrm{new}(S_i)$, and hence $\mathrm{new}(S_1') \cap \mathrm{new}(S_2') = \emptyset$. Thus, we have that
$x(y).P_1 \mid \overline{x}z.P_2 \Rightarrow S = \{x(y).S_1', \overline{x}z.S_2'\}$
with $P_1 \Rightarrow S_1'$, $P_2 \Rightarrow S_2'$, and $\mathrm{new}(S_1') \cap \mathrm{new}(S_2') = \emptyset$.
Now $\{x(y).S_1', \overline{x}z.S_2'\} \rightarrow S_1'[z/y] \cup S_2' = S'$. It remains to show that $P' \Rightarrow S'$. Since $P_1 \Rightarrow S_1'$, Lemma 4(1) implies that $P_1[z/y] \Rightarrow S_1'[z/y]$. Together with $P_2 \Rightarrow S_2'$, we obtain from (S1) that $P' = P_1[z/y] \mid P_2 \Rightarrow S_1'[z/y] \cup S_2' = S'$.
(PAR) $P \rightarrow P'$ is $P_1 \mid P_2 \rightarrow P_1' \mid P_2$ with $P_1 \rightarrow P_1'$, and the result holds for $P_1 \rightarrow P_1'$. Then $S = S_1 \cup S_2$ with $P_1 \Rightarrow S_1$ and $P_2 \Rightarrow S_2$ and $\mathrm{new}(S_1) \cap \mathrm{new}(S_2) = \emptyset$. By induction there exists S_1' such that $S_1 \rightarrow S_1'$ and $P_1' \Rightarrow S_1'$. By the chemical law, $S_1 \cup S_2 \rightarrow S_1' \cup S_2 = S'$. Since $\mathrm{new}(S_1') \subseteq \mathrm{new}(S_1)$, $\mathrm{new}(S_1') \cap \mathrm{new}(S_2) = \emptyset$. Hence $P' = P_1' \mid P_2 \Rightarrow S_1' \cup S_2 = S'$.
(RES) $P \rightarrow P'$ is $(\nu x)Q \rightarrow (\nu x)Q'$ with $Q \rightarrow Q'$ and the result holds for $Q \rightarrow Q'$. By (S2), $S = T[n/x]$ with $Q \Rightarrow T$ and $n \in \mathrm{New} - \mathrm{new}(T)$. By induction there exists T' such that $T \rightarrow T'$ and $Q' \Rightarrow T'$. By Lemma 1 it follows from $T \rightarrow T'$ that $T[n/x] \rightarrow T'[n/x] = S'$. From $Q' \Rightarrow T'$ (and the fact that $\mathrm{new}(T') \subseteq \mathrm{new}(T)$ and hence $n \notin \mathrm{new}(T')$), it follows with (S2) that $P' = (\nu x)Q' \Rightarrow T'[n/x] = S'$.
(STRUCT) $P \rightarrow P'$ with $Q \rightarrow Q'$, $P \equiv Q$, and $P' \equiv Q'$, and the result holds for $Q \rightarrow Q'$. By Theorem B, $P \Rightarrow S$ and $P \equiv Q$ imply that $Q \Rightarrow S$. By induction there exists S' such that $S \rightarrow S'$ and $Q' \Rightarrow S'$. Again by Theorem B, it follows from $Q' \Rightarrow S'$ and $P' \equiv Q'$ that $P' \Rightarrow S'$. □

To prove (AR) we need two key lemma's. In their proofs we use all the laws of structural congruence. We abbreviate $[n_1/x_1] \cdots [n_m/x_m]$ by $[n_i/x_i]$. For a guard g we denote by $g[n/x]$ the guard obtained by replacing the free ocurrences of x by n, i.e., $(u(v))[n/x] = u[n/x](v)$ and $(\overline{u}v)[n/x] = \overline{u[n/x]}v[n/x]$.

Lemma 5. *Let $F \subset \mathrm{N}$ be finite. If $P \Rightarrow S$ and $S = \{g.S_1\} \cup S_2$, where g is a guard, then there exist $x_1, \ldots, x_m \in \mathrm{N} - F$ ($m \geq 0$), $n_1, \ldots, n_m \in \mathrm{new}(S)$, a guard g' over N, process terms P_1, P_2, and solutions S_1', S_2' such that*

$$P \equiv (\nu x_1) \cdots (\nu x_m)(g'.P_1 \mid P_2),$$

$g'[n_i/x_i] = g$, $P_1 \Rightarrow S_1'$, $S_1'[n_i/x_i] = S_1$, $P_2 \Rightarrow S_2'$, $S_2'[n_i/x_i] = S_2$, $\mathrm{new}(S_1') \cap \mathrm{new}(S_2') = \emptyset$, and $n_1, \ldots, n_m \notin \mathrm{new}(S_1') \cup \mathrm{new}(S_2')$.

Proof. The proof is by induction on the length of P. We consider the usual cases of $P \Rightarrow S$. It is trivial for $\mathbf{0} \Rightarrow \emptyset$.

(S1) $P \Rightarrow S$ is $Q_1 \mid Q_2 \Rightarrow T_1 \cup T_2$ with $Q_1 \Rightarrow T_1$, $Q_2 \Rightarrow T_2$, and $\mathrm{new}(T_1) \cap \mathrm{new}(T_2) = \emptyset$. Assume $T_1 = \{g.S_1\} \cup U$ and $S_2 = U \cup T_2$ (the case that $g.S_1$ is in T_2 is symmetric). By induction, for $Q_1 \Rightarrow T_1$ with the finite set $F_1 = F \cup \mathrm{fn}(Q_2)$, we obtain that $Q_1 \equiv (\nu x_1) \cdots (\nu x_m)(g'.P_1 \mid R)$, $n_i \in \mathrm{new}(T_1)$, $g'[n_i/x_i] = g$, $P_1 \Rightarrow S_1'$, $S_1'[n_i/x_i] = S_1$, $R \Rightarrow U'$, $U'[n_i/x_i] = U$, $\mathrm{new}(S_1') \cap \mathrm{new}(U') = \emptyset$, and $n_i \notin \mathrm{new}(S_1') \cup \mathrm{new}(U')$. Since $x_i \notin F_1$ and hence $x_i \notin \mathrm{fn}(Q_2)$, it follows that $P = Q_1 \mid Q_2 \equiv (\nu x_1) \cdots (\nu x_m)(g'.P_1 \mid R) \mid Q_2 \equiv (\nu x_1) \cdots (\nu x_m)(g'.P_1 \mid R \mid Q_2)$ by structural laws (1.2), (1.3), and (2.3). Let $P_2 = R \mid Q_2$ and $S_2' = U' \cup T_2$. Then $P_2 \Rightarrow S_2'$ because $\mathrm{new}(U') \subseteq \mathrm{new}(U) \subseteq \mathrm{new}(T_1)$, and $S_2'[n_i/x_i] = U'[n_i/x_i] \cup T_2[n_i/x_i] = U \cup T_2 = S_2$ because $x_i \notin \mathrm{fn}(T_2)$. Furthermore $\mathrm{new}(S_1') \cap \mathrm{new}(S_2') = \emptyset$ because $\mathrm{new}(S_1') \cap \mathrm{new}(U') = \emptyset$ and $\mathrm{new}(S_1') \cap \mathrm{new}(T_2) = \emptyset$, where the latter holds because $\mathrm{new}(S_1') \subseteq \mathrm{new}(S_1) \subseteq \mathrm{new}(T_1)$ and $\mathrm{new}(T_1) \cap \mathrm{new}(T_2) = \emptyset$. Finally, $n_i \notin \mathrm{new}(S_2')$ because $n_i \notin \mathrm{new}(U')$ and $n_i \notin \mathrm{new}(T_2)$, where the latter holds because $n_i \in \mathrm{new}(T_1)$.

(S2) $P \Rightarrow S$ is $(\nu x)Q \Rightarrow T[n/x]$ with $Q \Rightarrow T$ and $n \notin \mathrm{new}(T)$. We first observe that we may assume that x is not bound in g and that $x \notin F$. Otherwise we could consider a suitable α-equivalent $P' = (\nu w)Q[w/x]$ instead of P. Then $P' \equiv P$ by structural law (α), and hence $P' \Rightarrow S$ by Theorem B. Also, the induction hypothesis holds for $Q[w/x]$ because it is shorter than P.

In the case that $x \notin \mathrm{fn}(Q)$ we are ready by induction, because then $P = (\nu x)Q \equiv Q$ by structural law (2.2), and $S = T[n/x] = T$.

Now assume that $x \in \mathrm{fn}(Q)$. Consider $T[n/x] = \{g.S_1\} \cup S_2$. Since, as discussed above, x is not bound in g, $T = T[n/x][x/n] = \{g[x/n].S_1[x/n]\} \cup S_2[x/n]$. Hence $T = \{h.T_1\} \cup T_2$, with $h[n/x] = g$, $T_1[n/x] = S_1$, and $T_2[n/x] = S_2$. By induction, for $Q \Rightarrow T$ with the same F, it follows that $Q \equiv (\nu x_1) \cdots (\nu x_m)(h'.P_1 \mid P_2)$ with $n_i \in \mathrm{new}(T)$, $h'[n_i/x_i] = h$, $P_1 \Rightarrow S_1'$, $S_1'[n_i/x_i] = T_1$, $P_2 \Rightarrow S_2'$, and $S_2'[n_i/x_i] = T_2$. Then $P = (\nu x)Q \equiv (\nu x_1) \cdots (\nu x_m)(\nu x)(h'.P_1 \mid P_2)$ by structural law (2.1), with $h'[n_i/x_i][n/x] = h[n/x] = g$ and $S_j'[n_i/x_i][n/x] = T_j[n/x] = S_j$. Furthermore $n \notin \mathrm{new}(S_j')$ because $\mathrm{new}(S_j') \subseteq \mathrm{new}(T_j) \subseteq \mathrm{new}(T)$. Finally, $n \in \mathrm{new}(T[n/x])$ because $x \in \mathrm{fn}(Q) = \mathrm{fn}(T) \cap \mathrm{N}$.

(S3) $P \Rightarrow S$ is $g.P_1 \Rightarrow \{g.S_1\}$ with $P_1 \Rightarrow S_1$. Let $S_2 = \emptyset$ and $P_2 = \mathbf{0}$. Then $P \equiv g.P_1 \mid \mathbf{0}$ by structural law (1.1), with $m = 0$, $g' = g$, $S_1' = S_1$, and $S_2' = S_2$.

(S4) $P \Rightarrow S$ is $!Q \Rightarrow \bigcup_{i \in \mathbf{N}} T_i$ with $Q \Rightarrow T_i$. Since this case is very similar to case (S1), we do not consider all details. Assume that $T_1 = \{g.S_1\} \cup U$ and $S_2 = U \cup \bigcup_{i \in \mathbf{N}} T_{i+1}$. By induction, for $Q \Rightarrow T_1$, we obtain that $Q \equiv (\nu x_1) \cdots (\nu x_m)(g'.P_1 \mid R)$. It follows that $P = !Q \equiv Q \mid !Q \equiv (\nu x_1) \cdots (\nu x_m)(g'.P_1 \mid$

$R \mid !Q$) by structural law (3.1). Now let $P_2 = R \mid !Q$ and $S_2' = U' \cup \bigcup_{i \in \mathbf{N}} T_{i+1}$. □

By applying Lemma 5 twice, in a rather straightforward way, we obtain the following lemma, to be used in the proof of (AR).

Lemma 6. *If $P \Rightarrow \{g_1.S_1, g_2.S_2\} \cup S_3$, where g_1 and g_2 are guards, then there exist $x_1, \ldots, x_m \in \mathrm{N}$ ($m \geq 0$), $n_1, \ldots, n_m \in \mathrm{New}$, guards g_1', g_2' over N, terms P_1, P_2, P_3, and solutions S_1', S_2', S_3' such that*

$$P \equiv (\nu x_1) \cdots (\nu x_m)(g_1'.P_1 \mid g_2'.P_2 \mid P_3),$$

$g_1'[n_i/x_i] = g_1$, $g_2'[n_i/x_i] = g_2$, $P_j \Rightarrow S_j'$ and $S_j'[n_i/x_i] = S_j$ for $j = 1, 2, 3$, the sets $\mathrm{new}(S_j')$ are mutually disjoint, and $n_1, \ldots, n_m \notin \mathrm{new}(S_1') \cup \mathrm{new}(S_2') \cup \mathrm{new}(S_3')$.

Proof. Let $T = \{g_2.S_2\} \cup S_3$. Since $P \Rightarrow \{g_1.S_1\} \cup T$, we obtain from Lemma 5, with F consisting of $\mathrm{fn}(P)$ together with the bound name in g_2 if it has one, that $P \equiv (\nu x_1) \cdots (\nu x_k)(g_1'.P_1 \mid Q)$ and $g_1'[n_i/x_i] = g_1$, $P_1 \Rightarrow S_1'$, $S_1'[n_i/x_i] = S_1$, $Q \Rightarrow T'$, $T'[n_i/x_i] = T$, $\mathrm{new}(S_1') \cap \mathrm{new}(T') = \emptyset$, and $n_i \notin \mathrm{new}(S_1') \cup \mathrm{new}(T')$. Now $T[x_i/n_i] = T'[n_i/x_i][x_i/n_i] = T'$ because $n_i \notin \mathrm{new}(T')$. Hence $Q \Rightarrow T[x_i/n_i]$ and, since x_i is not bound in g_2 (because x_i is not in F), $T' = T[x_i/n_i] = \{h_2.T_2\} \cup T_3$ with $h_2 = g_2[x_i/n_i]$, $T_2 = S_2[x_i/n_i]$, and $T_3 = S_3[x_i/n_i]$. Applying Lemma 5 to $Q \Rightarrow \{h_2.T_2\} \cup T_3$, with $F' = \{x_1, \ldots, x_k\} \cup \mathrm{fn}(g_1'.P_1)$, we obtain that $Q \equiv (\nu y_1) \cdots (\nu y_p)(g_2'.P_2 \mid P_3)$ and $m_i \in \mathrm{new}(T')$, $g_2'[m_i/y_i] = h_2$, $P_2 \Rightarrow S_2'$, $S_2'[m_i/y_i] = T_2$, $P_3 \Rightarrow S_3'$, $S_3'[m_i/y_i] = T_3$, $\mathrm{new}(S_2') \cap \mathrm{new}(S_3') = \emptyset$, and $m_i \notin \mathrm{new}(S_2') \cup \mathrm{new}(S_3')$. Since the y's are different from the x's and $y_i \notin \mathrm{fn}(g_1'.P_1)$, we get

$$\begin{aligned} P &\equiv (\nu x_1) \cdots (\nu x_k)(g_1'.P_1 \mid (\nu y_1) \cdots (\nu y_p)(g_2'.P_2 \mid P_3)) \\ &\equiv (\nu y_1) \cdots (\nu y_p)(\nu x_1) \cdots (\nu x_k)(g_1'.P_1 \mid g_2'.P_2 \mid P_3). \end{aligned}$$

It can now be checked that all requirements are fulfilled. First, $g_1'[m_i/y_i][n_i/x_i] = g_1'[n_i/x_i] = g_1$ because y_i does not occur free in g_1'; also $g_2'[m_i/y_i][n_i/x_i] = h_2[n_i/x_i] = g_2[x_i/n_i][n_i/x_i] = g_2$ because $x_i \notin \mathrm{fn}(P)$ and hence x_i does not occur free in g_2. For similar reasons, $S_1'[m_i/y_i][n_i/x_i] = S_1'[n_i/x_i] = S_1$ and, for $j = 2, 3$, $S_j'[m_i/y_i][n_i/x_i] = T_j[n_i/x_i] = S_j[x_i/n_i][n_i/x_i] = S_j$. Next we observe that, for $j = 2, 3$, $\mathrm{new}(S_j') \subseteq \mathrm{new}(T_j) \subseteq \mathrm{new}(T')$. This implies that the sets $\mathrm{new}(S_j')$ are mutually disjoint for $j = 1, 2, 3$ and that $n_i \notin \mathrm{new}(S_1') \cup \mathrm{new}(S_2') \cup \mathrm{new}(S_3')$. Also, since $m_i \in \mathrm{new}(T')$, $m_i \notin \mathrm{new}(S_1')$. □

Theorem AR. *If $P \Rightarrow S$ and $S \rightarrow S'$, then there exists P' such that $P \rightarrow P'$ and $P' \Rightarrow S'$.*

Proof. First we observe that if $P \Rightarrow S$, then $\mathrm{fn}(S) \cap \mathrm{N} = \mathrm{fn}(P)$ and hence $\mathrm{fn}(S) \cap \mathrm{N} \subset \mathrm{N}$. Consequently, as argued in Section 4, the transition $S \rightarrow S'$ is the result of communication of two guarded molecules, i.e.,

$$S = \{x(y).S_1, \overline{x}z.S_2\} \cup S_3 \text{ and } S' = S_1[z/y] \cup S_2 \cup S_3.$$

By Lemma 6, $P \equiv (\nu x_1)\cdots(\nu x_m)(x'(y).P_1 \mid \overline{x}'z'.P_2 \mid P_3)$ with $x_i \neq y$ and with the properties mentioned in that lemma. Hence (using STRUCT) $P \rightarrow P'$ where $P' = (\nu x_1)\cdots(\nu x_m)(P_1[z'/y] \mid P_2 \mid P_3)$. It remains to show that $P' \Rightarrow S'$. Since $P_1 \Rightarrow S_1'$, $P_1[z'/y] \Rightarrow S_1'[z'/y]$ by Lemma 4(1). Hence, using the properties of new(S_j'), $P' \Rightarrow (S_1'[z'/y] \cup S_2' \cup S_3')[n_i/x_i] = S_1'[z'/y][n_i/x_i] \cup S_2 \cup S_3$. Since $z'[n_i/x_i] = z$, $S_1'[z'/y][n_i/x_i] = S_1'[n_i/x_i][z/y] = S_1[z/y]$. Hence $P' \Rightarrow S_1[z/y] \cup S_2 \cup S_3 = S'$. □

Next we show that the new structural laws (3.2)-(3.4) and (2.4) are valid for multiset congruence.

Lemma 7. *$!(P \mid Q) \equiv_m !P \mid !Q$, $!!P \equiv_m !P$, $!\mathbf{0} \equiv_m \mathbf{0}$, and $(\nu x)g.P \equiv_m g.(\nu x)P$ provided x does not occur in g.*

Proof. We prove the equivalences one by one.

$!(P \mid Q) \equiv_m !P \mid !Q$. Consider $!(P \mid Q) \Rightarrow S$. Then $S = \bigcup_{i\in\mathbf{N}} S_i$ with $P \mid Q \Rightarrow S_i$ for every $i \in \mathbf{N}$, and all new(S_i) are disjoint. Then $S_i = T_i \cup U_i$ with $P \Rightarrow T_i$ and $Q \Rightarrow U_i$ and new$(T_i) \cap$ new$(U_i) = \emptyset$. By (S4), $!P \Rightarrow \bigcup_i T_i$ and $!Q \Rightarrow \bigcup_i U_i$. Hence, by (S1), $!P \mid !Q \Rightarrow \bigcup_i T_i \cup \bigcup_i U_i = \bigcup_i (T_i \cup U_i) = \bigcup_i S_i = S$, and so $!P \mid !Q \Rightarrow S$. Note that the equality $\bigcup_i T_i \cup \bigcup_i U_i = \bigcup_i (T_i \cup U_i)$ follows from the (general) associativity of multiset union. In the other direction, if $!P \mid !Q \Rightarrow S$ then $S = T \cup U$ with $!P \Rightarrow T$ and $!Q \Rightarrow U$. Then $T = \bigcup_i T_i$ with $P \Rightarrow T_i$ and $U = \bigcup_i U_i$ with $Q \Rightarrow U_i$. By (S1), $P \mid Q \Rightarrow T_i \cup U_i$ and so, by (S4), $!(P \mid Q) \Rightarrow \bigcup_i (T_i \cup U_i) = \bigcup_i T_i \cup \bigcup_i U_i = T \cup U = S$. Hence $!(P \mid Q) \Rightarrow S$.

$!!P \equiv_m !P$. Consider $!!P \Rightarrow S$. Then $S = \bigcup_{i\in\mathbf{N}} S_i$ with $!P \Rightarrow S_i$ for all i, and the new(S_i) are mutually disjoint. Hence, for every i, $S_i = \bigcup_{j\in\mathbf{N}} S_{i,j}$ with $P \Rightarrow S_{i,j}$ for all j, and the new$(S_{i,j})$ are mutually disjoint. Let $c : \mathbf{N} \times \mathbf{N} \rightarrow \mathbf{N}$ be a bijection. Define, for $k \in \mathbf{N}$, $T_k = S_{c^{-1}(k)}$, i.e., $T_{c(i,j)} = S_{i,j}$. Since $P \Rightarrow T_k$ for all k, and the new(T_k) are mutually disjoint, $!P \Rightarrow \bigcup_{k\in\mathbf{N}} T_k$. But $S = \bigcup_{i\in\mathbf{N}} S_i = \bigcup_{i\in\mathbf{N}} \bigcup_{j\in\mathbf{N}} S_{i,j} = \bigcup_{k\in\mathbf{N}} T_k$ by (general) associativity of multiset union. And so $!P \Rightarrow S$. In the other direction, consider $!P \Rightarrow S$. Then $S = \bigcup_{k\in\mathbf{N}} T_k$ with $P \Rightarrow T_k$ for every k. Define $S_{i,j} = T_{c(i,j)}$, and, for every i, $S_i = \bigcup_{j\in\mathbf{N}} S_{i,j}$. Then, for every i, $!P \Rightarrow S_i$. Hence $!!P \Rightarrow \bigcup_{i\in\mathbf{N}} S_i = S$.

$!\mathbf{0} \equiv_m \mathbf{0}$. This follows from the obvious fact that if $S_i = \emptyset$ for all i, then $\bigcup_{i\in\mathbf{N}} S_i = \emptyset$.

If x does not occur in g, then $(\nu x)g.P \equiv_m g.(\nu x)P$. Clearly $(\nu x)g.P \Rightarrow \{g.S\}[n/x]$ for all S and n such that $P \Rightarrow S$ and $n \notin$ new$(g.S)$. Also, $g.(\nu x)P \Rightarrow \{g.S[n/x]\}$ for all S and n such that $P \Rightarrow S$ and $n \notin$ new(S). Now new$(g.S) =$ new(S). Moreover, $\{g.S\}[n/x] = \{g[n/x].S[n/x]\}$ because x does not occur bound in g, and $\{g[n/x].S[n/x]\} = \{g.S[n/x]\}$ because x does not occur free in g. □

It is easy to see from the proofs in this section that after adding the laws (9)-(12) to structural congruence, results (A) and (B) are still valid. This proves result (A′) and the only-if part of result (B′).

8 Conclusion

It would be nice to extend result (A) to the labeled transition system of the small π-calculus, with labels τ, $x(y)$, $\overline{x}z$, and $(\nu z)\overline{x}z$. This raises the problem of finding a suitable notion of strong bisimulation between the labeled transition system of the small π-calculus and a correspondingly labeled transition system of the multiset π-calculus Mπ. Lemma 4(1) suggests that the semantic mapping could be an open bisimulation, in the sense of [20]. Also, one would like to extend the whole approach (a multiset semantics for which structural congruence is sound and complete) to CCS and the full π-calculus. The addition of choice (+) to the small π-calculus in its full generality would ruin our approach, due to the possible presence of parallel composition in a choice context (see [6, 18] for quite complicated solutions to this well-known problem). However, this paper can be extended in a straightforward way to guarded choice, in particular to the (monadic) π-calculus presented in [14].

Acknowledgments. I am grateful to P.S.Thiagarajan for the stimulating discussions that led to this paper. I thank my student Tjalling Gelsema for checking many of the details.

References

1. J.C.M.Baeten, J.A.Bergstra, J.W.Klop; An operational semantics for process algebra, Mathematical Problems in Computation Theory, Banach Center Publications, Vol.21, PWN, Warsaw, 1988, pp.47-81
2. J.-P.Bânatre, D. Le Métayer; Programming by multiset transformation, Comm. of the ACM 36 (1993), 98-111
3. H.P.Barendregt; *The Lambda Calculus*, North-Holland, Amsterdam, 1984
4. G.Berry, G.Boudol; The chemical abstract machine, Theor.Comput.Sci. 96 (1992), 217-248
5. N.G.De Bruijn; Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, Indag. Math. 34 (1972), 381-392
6. P.Degano, R.De Nicola, U.Montanari; A distributed operational semantics for CCS based on Condition/Event systems, Acta Informatica 26 (1988), 59-91
7. J.Engelfriet; Branching processes of Petri nets, Acta Informatica 28 (1991), 575-591
8. J.Engelfriet; A multiset semantics of the pi-calculus with replication, in Proc.CONCUR'93, Lecture Notes in Computer Science 715, Springer-Verlag, 1993, 7-21
9. J.Engelfriet, T.E.Gelsema; Multisets and structural congruence of the pi-calculus with replication, Technical Report, Department of Computer Science, Leiden University, August 1994.
10. U.Goltz; On representing CCS programs by finite Petri nets, in Proc. MFCS'88, Lecture Notes in Computer Science 324, Springer-Verlag, 1988, pp.339-350
11. U.Goltz, W.Reisig; The non-sequential behaviour of Petri nets, Inform. Control 57 (1983), 125-147
12. R.Milner; *Communication and Concurrency*, Prentice-Hall, Englewood Cliffs, NJ., 1989
13. R.Milner; Functions as processes, Math. Struct. in Comp. Science 2 (1992), 119-141

14. R.Milner; The polyadic π-calculus: a tutorial, Report ECS-LFCS-91-180, University of Edinburgh, 1991
15. R.Milner; Elements of interaction, Comm. of the ACM 36 (1993), 78-89, Turing Award Lecture
16. R.Milner, J.Parrow, D.Walker; A calculus of mobile processes, Inform. Comput. 100 (1992), 1-77
17. M.Nielsen; CCS and its relationship to net theory, in *Petri nets: applications and relationships to other models of concurrency*, Lecture Notes in Computer Science 255, Springer-Verlag, 1987, pp.393-415
18. E.-R.Olderog; *Nets, Terms and Formulas*, Cambridge University Press, Cambridge, 1991
19. W.Reisig; *Petri Nets*, EATCS Monographs in Theoretical Computer Science, Springer-Verlag, 1982
20. D.Sangiorgi; A theory of bisimulation for the π-calculus, in Proc.CONCUR'93, Lecture Notes in Computer Science 715, Springer-Verlag, 1993, 127-142. Also, Report ECS-LFCS-93-270, University of Edinburgh
21. D.Taubner; *Finite representations of CCS and TCSP programs by automata and Petri nets*, Lecture Notes in Computer Science 369, Springer-Verlag, 1989

This article was processed using the LaTeX macro package with LLNCS style

Fig. 2. Picture of a parallel computation in Mπ.