ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume **17**, Number 3, 2014, 265–288

A Nominal Approach of Fusion Calculus

Andrei ALEXANDRU, Gabriel CIOBANU

Romanian Academy, Institute of Computer Science, Iaşi

E-mail: andrei.alexandru@iit.academiaromana-is.ro
gabriel@info.uaic.ro

Abstract. We provide a nominal semantics of the monadic version of the fusion calculus. A set of compact transition rules is presented in the Fraenkel-Mostowski framework by using a specific nominal quantifier. Using several nominal techniques, it is proved the equivalence between the new nominal semantics and the original semantics of the monadic fusion calculus.

1. Introduction

The Fraenkel-Mostowski (FM) set theory over an infinite set of atoms represents the mathematical model for names in syntax. The FM axioms are precisely the Zermelo-Fraenkel with atoms (ZFA) axioms over an infinite set A of atoms [6], together with the special axiom of finite support claiming that for each element x in an arbitrary set we can find a finite set of atoms supporting x. Atoms have the same properties as names in computer science. The finite support axiom is motivated by the fact that syntax can only involve finitely many names. The approach based on the FM set theory became successful because it provides a balance between a rigorous formalism and an informal reasoning. This is discussed in [10], where principles of structural recursion and induction are explained within nominal sets.

The fusion calculus [8] has emerged as a good model for distributed computation paradigms like web-services and service oriented architectures. The monadic fusion calculus was presented for the first time in [9]. The monadic fusion calculus contains the monadic π-calculus as a proper subcalculus (as it is proved in Section 2.3 from [9]), and so it inherits all its expressive power. Therefore, everything that can be done in the monadic π-calculus can be done in the monadic fusion calculus without added complications. However, the fusion calculus has only one binding operator, whereas the π-calculus has two (binding of input variables and restriction of communication channels). Furthermore, there is a complete symmetry between input and output

actions in the fusion calculus, which does not occur in the π-calculus. This is due to the fact that, in the π-calculus, input entails a binding while output does not.

In this paper we employ nominal techniques to obtain a new semantics of the monadic fusion calculus, techniques which are similar to those involved in describing the nominal semantics of the πI-calculus in [1]. We use a special nominal quantifier N [6] in order to present a set of compact transition rules for the monadic version of the fusion calculus. The central idea is to use *atoms* to represent variable symbols, and *nominal abstraction* to represent the binding operator in the fusion calculus. A mixing of $\forall$ and N quantifiers is used to replace the side conditions in the transition rules of the fusion calculus. Finally, we prove that there is an equivalence between the expressive power of nominal semantics and the usual semantics of the (monadic) fusion calculus presented in [9].

2. Nominal Techniques

The notions presented in this section are slightly modified versions of those introduced in [6]. They were also presented in Section 2 from [1]. We remind them in order to keep the paper self-contained. Let A be a fixed infinite ZF-set. The following results make also sense if A is considered to be the set of atoms in the ZFA framework (characterized by the axiom "$y \in x \Rightarrow x \notin A$"), and if 'ZF' is replaced by 'ZFA' in their statement. This is because the theory of nominal sets makes sense in both ZF and ZFA.

Definition 2.1. A *transposition* is a function $(a\,b) : A \to A$ defined by $(a\,b)(a) = b$, $(a\,b)(b) = a$ and $(a\,b)(n) = n$ for $n \neq a, b$. A *permutation* of A is generated by composing finitely many transpositions.

Let S_A be the set of all permutations.

Definition 2.2.

1. Let X be a ZF set. An S_A*-action* on X is a function $\cdot : S_A \times X \to X$ having the properties that $Id \cdot x = x$ and $\pi \cdot (\pi^{'} \cdot x) = (\pi \circ \pi^{'}) \cdot x$ for all $\pi, \pi' \in S_A$ and $x \in X$. An S_A*-set* is a pair $(X, \cdot)$ where X is a ZF set, and $\cdot : S_A \times X \to X$ is an S_A-action on X.

2. Let $(X, \cdot)$ be an S_A-set. We say that $S \subset A$ *supports* x whenever for each $\pi \in Fix(S)$ we have $\pi \cdot x = x$, where $Fix(S) = \{\pi \mid \pi(a) = a, \forall a \in S\}$.

3. Let $(X, \cdot)$ be an S_A-set. We say that X is a *nominal set* if for each $x \in X$ there exists a finite set $S_x \subset A$ which supports x.

4. Let X be an S_A-set and let $x \in X$. If there exists a finite set supporting x, then there exists a least finite set supporting x [6] which is called *the support of* x, and is denoted by $supp(x)$.

Proposition 2.3. *Let $(X, \cdot)$ be an S_A-set and let $\pi \in S_A$. If $x \in X$ is finitely supported, then $\pi \cdot x$ is finitely supported and $supp(\pi \cdot x) = \pi(supp(x))$.*

Example 2.4.

1. *The set A of atoms is an S_A-set with the S_A-action $\cdot : S_A \times A \to A$ defined by $\pi \cdot a := \pi(a)$ for all $\pi \in S_A$ and $a \in A$. $(A, \cdot)$ is a nominal set because for each $a \in A$ we have that $\{a\}$ supports a. Moreover, $supp(a) = \{a\}$ for each $a \in A$.*

2. *The set A of atoms is an S_A-set with the S_A-action $\cdot : S_A \times A \to A$ defined by $\pi \cdot a := a$ for all $\pi \in S_A$ and $a \in A$. $(A, \cdot)$ is a nominal set because for each $a \in A$ we have that $\emptyset$ supports a. Moreover, $supp(a) = \emptyset$ for each $a \in A$.*

3. *The set S_A is an S_A-set with the S_A-action $\cdot : S_A \times S_A \to S_A$ defined by $\pi \cdot \sigma := \pi \circ \sigma \circ \pi^{-1}$ for all $\pi, \sigma \in S_A$. $(S_A, \cdot)$ is a nominal set because for each $\sigma \in S_A$ the finite set $\{a \in A \,|\, \sigma(a) \neq a\}$ supports σ. Moreover, $supp(\sigma) = \{a \in A \,|\, \sigma(a) \neq a\}$ for each $\sigma \in S_A$.*

4. *If $(X, \cdot)$ is an S_A-set, then $\wp(X) = \{Y \,|\, Y \subseteq X\}$ is also an S_A-set with the S_A-action $\star : S_A \times \wp(X) \to \wp(X)$ defined by $\pi \star Y := \{\pi \cdot y \,|\, y \in Y\}$ for all permutations π of A and subsets Y of X. Note that $\wp(X)$ is not necessarily a nominal set even if X is. For each nominal set $(X, \cdot)$ we denote by $\wp_{fs}(X)$ the set of subsets of X which are finitely supported according to the action $\star$. According to Proposition **??**, $(\wp_{fs}(X), \star|_{\wp_{fs}(X)})$ is a nominal set, where $\star|_{\wp_{fs}(X)}$ represents the action $\star$ restricted to $\wp_{fs}(X)$.*

5. *Let $(X, \cdot)$ and $(Y, \diamond)$ be S_A-sets. The Cartesian product $X \times Y$ is also an S_A-set with the S_A-action $\star : S_A \times (X \times Y) \to (X \times Y)$ defined by $\pi \star (x, y) = (\pi \cdot x, \pi \diamond y)$ for all $\pi \in S_A$ and $x \in X$, $y \in Y$. If $(X, \cdot)$ and $(Y, \diamond)$ are nominal sets, then $(X \times Y, \star)$ is also a nominal set.*

If A is the set of atoms in ZFA, consider the FM cumulative hierarchy $FM(A)$ described in [6]. $FM(A)$ is a model of the FM set theory. A detailed construction of $FM(A)$ is also presented in Section 2 from [1]. We know that $FM(A)$ is a nominal set with the S_A-action $\cdot : S_A \times FM(A) \to FM(A)$ defined inductively by $\pi \cdot a := \pi(a)$ for all atoms $a \in A$ and $\pi \cdot x := \{\pi \cdot y \,|\, y \in x\}$ for all $x \in FM(A)$.

In this paper we are interest especially on those elements in $FM(A)$ which are closed under the S_A-action on $FM(A)$. We define an *FM-set* as an element in $FM(A)$. We define an *NFM-set* as a nominal set with a special S_A-action induced by the S_A action on $FM(A)$. Precisely, an NFM-set is a set from ZFA which is closed under the S_A-action on $FM(A)$ and whose all elements are finitely supported. The S_A-action on an NFM-set is called an *interchange function*.

Definition 2.5.

i) An element from the FM universe $FM(A)$ that is not an atom is called a *Fraenkel-Mostowski set (FM-set)*.

ii) An *interchange function* on a set X defined by the axioms of ZFA set theory is a function $\cdot : S_A \times X \to X$ defined inductively by $\pi \cdot a := \pi(a)$ for all atoms $a \in A$ and $\pi \cdot x := \{\pi \cdot y \,|\, y \in x\}$ which satisfies the axiom that for all $x \in X$ there is a finite subset $S \subset A$ such that $(a\, b) \cdot x = x$ for all $a, b \notin S$.

iii) An *NFM-set* is a pair $(X, \cdot)$, where X is a set defined by the axioms of ZFA set theory and $\cdot : S_A \times X \to X$ is an interchange function on X.

Clearly, $FM(A)$ is an NFM-set. Note also that an FM-set is not a nominal set unless it is empty supported as an element in $FM(A)$. Details are presented in Section 2 from [1]. Also, every NFM-set is also a nominal set; the converse is not valid.

Example 2.6. *The set of atoms A with the S_A-action defined as in Example 2.4 (1) is an NFM-set, while the set of atoms A with the S_A-action defined as in Example 2.4 (2) is a nominal set but not an NFM-set.*

Since every NFM-set is also a nominal set, we obtain that every element in an NFM set have a (finite) support.

Example 2.7. *Consider the NFM set A of atoms.*

- *If $B \subset A$ and B is finite, then $supp(B) = B$.*
- *If $C \subset A$ and C is cofinite, then $supp(C) = A \setminus C$.*

The following Example (considered also in [6]) shows how we can express the λ-calculus in FM. In a similar way the processes in fusion calculus can be represented in FM.

Example 2.8.

1. *If X' is the set of λ-terms t, we inductively define an action $\star$ of S_A on X' by:*

   - *variable: $\pi \star a = \pi(a)$ whenever a is a variable (corresponding to atoms) and π is a permutation of atoms;*
   - *application: $\pi \star (tt') = (\pi \star t)(\pi \star t')$ for all λ-terms t and t' and for all $\pi \in S_A$;*
   - *abstraction: $\pi \star (\lambda a.t) = \lambda(\pi(a)).(\pi \star t)$ for all variables a, all λ-terms t and for all $\pi \in S_A$.*

   It is easy to check that $(X', \star)$ is a nominal set (it is also an NFM-set by the definition of the S_A-action), and the support of a λ-term t is the finite set of atoms occurring in t, whether as free bound or binding occurrences.

2. *Let X be the set of the α-equivalences classes of the λ-calculus terms t. We define an action $\cdot$ of S_A on X by $\pi \cdot [t]_\alpha = [\pi \star t]_\alpha$ for all λ-terms t and $\pi \in S_A$, where $[t]_\alpha$represents the α-equivalence class of the λ-term t. If two λ-terms t and t' are α-equivalent, it is clear that $\pi \star t = \pi \star t'$, and so the action $\cdot$ is well defined. It is easy to check that $(X, \cdot)$ is a nominal set (it is also an NFM-set). Moreover, X is a set which is in a bijection with an inductively defined FM-set (according to [6]). If t is chosen to be a representative of its α-equivalence class, then $supp(t)$ coincides with $fn(t)$, where $fn(t)$ is the set of free variables of t defined by λ-calculus rules. An α-equivalence class of terms does not contain*

bound names (in the sense of the quotient of the equivalence class over them). According to [6], we cannot define a function $bn : X \to \wp_{fin}(A)$ able to extract exactly the bound names for each FM element t. The α-equivalent terms are identified in the nominal framework because two α-equivalent terms have the same set of free variables.

Definition 2.9. If $x \in A$ and $y \in Y$ where Y is a nominal set, we say that x *is fresh for* y (and denote this by $x\#y$) if $supp(x) \cap supp(y) = \emptyset$.

Since A is in finite, for an arbitrary name we can always find a name outside its support, *i.e.*, a fresh name.

Proposition 2.10. ([6]) *Suppose that $p((x_i)_i)$ is a formula in the logic of ZFA or FM, where the free variables of p are listed in the set $(x_i)_i$, and $(x_i)_i$ are arbitrary distinct variables. Then $\forall a, b \in A.(p((x_i)_i) \Leftrightarrow p((a\,b) \cdot (x_i)_i))$, where $p((a\,b) \cdot (x_i)_i)$ denotes the result of substituting $(a\,b) \cdot x_j$ for all free occurrences x_j from the set $(x_i)_i$ in p.*

Definition 2.11 Let P be a predicate on A. We say that $И a.P(a)$ whenever $P(a)$ is true for all but finitely many elements of A. $И$ is called the *nominal quantifier*.

Proposition 2.12. ([6])

1. *Let X and Y be nominal sets. For each $x \in X$ and $y \in Y$ we have $supp((x, y)) = supp(x) \cup supp(y)$.*
2. *Let X be a finite FM-set. Then $supp(X) = \cup\{supp(x) \mid x \in X\}$.*

Proposition 2.13. *Let $(x_i)_i$ be a set of distinct variables and p a formula in the logic of FM. We have the following implications:*

$$[\forall a \in A.(a\#(x_i)_i \Rightarrow p)] \Rightarrow [И a.p] \Rightarrow [\exists a \in A.(a\#(x_i)_i \wedge p)].$$

Proof: We know that the set $\{a \in A \mid a\#(x_i)_i\}$ is cofinite. If we suppose $\forall a.(a\#(x_i)_i \Rightarrow p)$, we have $\{a \in A \mid a\#(x_i)_i\} \subset \{a \in A \mid p\}$ and so $И a.p$. Now, if we suppose $И a.p$, then $\{a \in A \mid p\}$ is cofinite, and so $\{a \in A \mid a\#(x_i)_i\} \cap \{a \in A \mid p\}$ is the intersection of two cofinite subsets of the infinite set A which cannot be empty. Indeed, if we suppose that the intersection $B \cap C$ of two cofinite subsets of A is empty, we have $C_{B \cap C} = A$, and so $C_B \cup C_C = A$. Since C_B and C_C are finite, then A is finite which contradicts the infiniteness of A. □

Proposition 2.14. *If the free variables of the formula p are contained in the set of distinct variables $\{a, (x_i)_i\}$, we also get the converse implication:*

$$[\exists a \in A.(a\#(x_i)_i \wedge p)] \text{ implies } [\forall a \in A.(a\#(x_i)_i \Rightarrow p)].$$

Proof: Let us assume that for some $a \in A$ we have $a\#(x_i)_i \wedge p$. Then, according to Proposition 2.10, we obtain that for every atom b we have $p(b, (a\,b) \cdot (x_i)_i)$, where $p(b, (a\,b) \cdot (x_i)_i)$ denotes the formula obtained when we substitute b for all free occurrences of a in p, and $(a\,b) \cdot x_j$ for all free occurrences of x_j in p. If we suppose $b\#(x_i)_i$, we get $(a\,b) \cdot (x_i)_i = (x_i)_i$ because we have $a\#(x_i)_i$), and so we obtain $p(b, (x_i)_i)$. □

By now we consider that the set A is equipped with the interchange function $\cdot : S_A \times X \to X$ defined by $\pi \cdot a = \pi(a)$ for all $\pi \in S_A$ and $a \in A$. This means that A is an NFM-set.

Definition 2.15. Let X be a nominal set.

1. For $a \in A$ and $x \in X$ we define an abstractive element $[a]x$ as $[a]x = \cap\{V \subset A \times X \mid (a, x) \in V \wedge supp(V) \subset supp(x) \setminus \{a\}\}$.
2. We define the abstraction function $abs : A \times X \to [A]X$ by $(a, x) \mapsto [a]x$, where $[A]X = \{[a]x \mid a \in A \wedge x \in X\}$.

Theorem 2.16. ([6]) *Let X be a nominal set, $a \in A$ and $x \in X$. Then we have*

a) $[a]x = \{(y, (y, a)x) \mid y \in A, y \neq a \text{ and } y\#x\} \cup \{(a, x)\}$;

b) $supp([a]x) = supp(x) \setminus \{a\}$.

Example 2.17.

- $[a](A\setminus\{a\}) = \{(a, A\setminus\{a\}), (b, A\setminus\{b\}), (c, A\setminus\{c\}), ...\}$
- $[a]\{a, b\} = \{(a, \{a, b\}), (c, \{c, b\}), (d, \{d, b\}), (e, \{e, b\}), ...$

According to [6], we have the following result.

Proposition 2.18. *Let X be a nominal set, $a, b \in A$ and $x, y \in X$.*
$[a]x = [b]y$ if and only if one of the following statements is true:

- *$a = b$ and $x = y$;*
- *$a \neq b$, $b\#x$ and $y = (b\,a) \cdot x$.*

Corollary 2.19. *Let X be a nominal set, $a, b \in A$ and $x, y \in X$.*
If $c\#(a, b, x, y)$ and $[a]x = [b]y$, then $(a\,c) \cdot x = (b\,c) \cdot y$.

Remark 2.20. *For a λ-term x, we denote by $fn(x)$ the free names of x. According to Theorem 2.16 and Example 2.8, $fn([a]x) = fn(x) \setminus \{a\}$. This means that* [ ] can be seen as a binding of a in x. *For more details about free names and support, see [6]. The bound names are not defined in the FM approach; thus we define the abstraction (function) to be similar to the λ-calculus binding. It is important to note that $[a]x$ is an equivalence class on renamings.*

In the previous results about abstraction we can consider X to be the nominal set $FM(A)$, and the elements of X to be FM-sets. Whenever x is an FM-set, the element $[a]x$ can be defined as in Definition 2.15, *i.e.*, $[a]x = \{(y, (y, a)x) \mid y \in A, y \neq a \text{ and } y\#x\} \cup \{(a, x)\}$ which is an FM-set because of axiom 6 of the FM set theory. Whenever $a \in A$ and x is an FM-set, $supp([a]x) = supp(x) \setminus \{a\}$. Proposition 2.18 and Corollary 2.19 remain valid when X is $FM(A)$, and x, y are FM-sets.

3. Fusion Calculus

The monadic version of the fusion calculus (also called update calculus) is introduced in [9]. Fusion calculus contains the monadic π-calculus as a proper subcalculus. However, there are some differences between the monadic fusion calculus and the monadic π-calculus. The most important is that in fusion calculus the input action does not bind names; the effect of an interaction is that the output and the input names become identified by an *update*. There are two differences between updates and π-calculus interactions: the effect of an update is not necessarily local but regulated by the use of a *scope* operator, and updates are symmetric with respect to the input-output polarities. The only binding operator in the fusion calculus is called *scope*; it is denoted by $(x)P$, meaning that the x is local in P. In a sense, it is a common denominator of input binding and restriction in π-calculus. Restriction implies that no name will ever replace bound names, while input binding requires that the bound name is immediately replaced by something received along the communication channel. Scope neither forbids nor forces a replace; it merely states the extent of the name.

We assume an infinite set of names ranged over $u, v, ..., x, y, ...$. As in π-calculus, names are used to represent communication channels, and placeholders for channels and values. Processes are defined by

$$P, Q ::= 0 \mid z\lfloor x\rfloor.P \mid z\lceil x\rceil.P \mid P + Q \mid P|Q \mid (x)P\,.$$

0 is an empty process. The output guarded process $z\lfloor x\rfloor.P$ sends an object x along a channel z and then, after the output is completed, continues as P. The input guarded process $z\lceil x\rceil.P$ receives an object along the channel z, and replace x with that object; after the input is completed, it continues as P. Contrary to the π-calculus, x is not bound in $z\lceil x\rceil.P$. As in the π-calculus, $P + Q$ represents nondeterministic choice and $P \mid Q$ represents parallel composition (that is, components P and Q can proceed independently and can interact via shared channels). The scope $(x)P$ limits the scope of x to P; scopes can be used to delimit the extent of updates (that is, update effects with respect to x are limited to P). The name x is said to be bound in $(x)P$. The free names $fn(P)$ are the names with a non-bound occurrence in P.

Definition 3.1. (α-convertibility)

1. If a name y does not occur in the process P, then $P\{y|z\}$ is the process obtained by replacing each free occurrence of z in P by y.

2. A change of bound name in a process P is the replacement of a subterm $(x)Q$ of P by $(y)Q\{y|x\}$, where y does not occur in Q.

3. Two processes P and Q are *α-convertible* (and this is denoted by $P \equiv_\alpha Q$) if Q can be obtained from P by a finite number of changes of bound names.

It is easy to note that α-convertibility in fusion calculus is the same as the α-equivalence in λ-calculus. In [9], the processes which are α-convertible are identified

by convention. In the nominal approach, we do not use such a convention, but *prove* that two processes are α-convertible if and only if they are equal. Therefore, in the FM framework we are able to prove that α-convertible processes are identical, whilst in the ZF framework this fact is assumed by an informal agreement.

The actions of the monadic fusion calculus are defined by

$$\gamma ::= z\lfloor x\rfloor \mid z\lceil x\rceil \mid [x/z] \mid 1 \mid (x)z\lfloor x\rfloor \mid (x)\, z\lceil x\rceil.$$

An output action $z\lfloor x\rfloor$ is similar to the free output in the π-calculus; $z\lfloor x\rfloor$ means "send the object x along the channel z". An input action $z\lceil x\rceil$ means "receive an object along the channel z and replace x with that object"; this action does not bind x. The update action $[x/z]$ (which does not appear in the π-calculus) indicates the replacement of all z by x. An action $[x/x]$ is called "inert" and is denoted by 1 (because the choice of x in $[x/x]$ is not important).

The actions $z\lfloor x\rfloor$, $z\lceil x\rceil$ and $[x/z]$ are called free actions, and are generically denoted by α. The object x of all these actions is free (in these actions); free actions have no bound names. A transition labelled with a free action is denoted by $P \xrightarrow[uc]{\alpha} Q$.

The actions $(x)z\lfloor x\rfloor$ and $(x)\, z\lceil x\rceil$ are called bound actions, and the name x is bound in these actions. The meaning of a bound action is simply that the object is emitted out of its scope, and must therefore not be confused with any name from the environment. A transition labelled with an action (either free or bound) is denoted by $P \xrightarrow[uc]{\gamma} Q$.

We denote by $n(\gamma)$, $fn(\gamma)$ and $bn(\gamma)$ all the names, free names and bound names occurring in γ, respectively. By definition, $n(1) = \emptyset$.

γ	$n(\gamma)$	$fn(\gamma)$	$bn(\gamma)$
$z\lfloor x\rfloor$	$\{x, z\}$	$\{x, z\}$	$\emptyset$
$z\lceil x\rceil$	$\{x, z\}$	$\{x, z\}$	$\emptyset$
$[x/z]$	$\{x, z\}$	$\{x, z\}$	$\emptyset$
1	$\emptyset$	$\emptyset$	$\emptyset$
$(x)z\lfloor x\rfloor$	$\{x, z\}$	$\{z\}$	$\{x\}$
$(x)\, z\lceil x\rceil$	$\{x, z\}$	$\{z\}$	$\{x\}$

A structural congruence relation can be defined over the set of processes. To define a structural congruence, we need the notions of context and congruence. A minor auxiliary definition is required for sums: an occurrence 0 in a process is *degenerate* if it is either the left or right term in a sum $P + Q$, and *non-degenerate* otherwise. Informally, a context differs from a process only in having a *hole* $[\cdot]$ in it. A context is regarded as a syntactic entity that transforms processes into processes. A *context* is obtained when the hole $[\cdot]$ replaces a non-degenerate occurrence of 0 in a process. If C is a context and P a process, we write $C[P]$ for the process obtained by replacing the hole $[\cdot]$ in C by P. This replacement is literal, and so names free in P may become bound in $C[P]$.

An equivalence relation $\Re$ on processes is a *congruence* if $(P, Q) \in \Re$ implies $(C[P], C[Q]) \in \Re$ for every context C.

Definition 3.2. The relation $\equiv$ over the set of processes is called a *structural congruence* and it is defined as the smallest congruence which satisfies:

- $P \equiv Q$ if $P \equiv_\alpha Q$
- $P + 0 \equiv P$, $P + Q \equiv Q + P$
- $(P + Q) + R \equiv P + (Q + R)$
- $P \mid 0 \equiv P$, $P \mid Q \equiv Q \mid P$
- $(P \mid Q) \mid R \equiv P \mid (Q \mid R)$
- $(x)0 \equiv 0$, $(x)(y)P \equiv (y)(x)P$
- $(x)(P + Q) \equiv (x)P + (x)Q$
- $(x)(P \mid Q) \equiv P \mid (x)Q$ if $x \notin fn(P)$.

Definition 3.3. The following labeled transition rules define *the Parrow-Victor (PV) semantics* of the monadic fusion calculus:

$$OUT: \quad \frac{}{z\lfloor x\rfloor.P \xrightarrow[uc]{z\lfloor x\rfloor} P} \qquad INP: \quad \frac{}{z\lceil x\rceil.P \xrightarrow[uc]{z\lceil x\rceil} P}$$

$$SUM: \quad \frac{P \xrightarrow[uc]{\alpha} P'}{P + Q \xrightarrow[uc]{\alpha} P'} \qquad CONG: \quad \frac{P \equiv P', Q \equiv Q', P \xrightarrow[uc]{\gamma} Q}{P' \xrightarrow[uc]{\gamma} Q'}$$

$$PAR: \quad \frac{P \xrightarrow[uc]{\alpha} P'}{P \mid Q \xrightarrow[uc]{\alpha} P' \mid Q} \qquad COM: \quad \frac{P \xrightarrow[uc]{z\lceil x\rceil} P', Q \xrightarrow[uc]{z\lfloor y\rfloor} Q'}{P \mid Q \xrightarrow[uc]{[y/x]} P' \mid Q'}$$

$$O\text{-}PASS: \quad \frac{P \xrightarrow[uc]{x\lfloor y\rfloor} P'}{(z)P \xrightarrow[uc]{x\lfloor y\rfloor} (z)P'}, z \neq x, y$$

$$I\text{-}PASS: \quad \frac{P \xrightarrow[uc]{x\lceil y\rceil} P'}{(z)P \xrightarrow[uc]{x\lceil y\rceil} (z)P'}, z \neq x, y$$

$$U\text{-}PASS: \quad \frac{P \xrightarrow[uc]{[x/y]} P'}{(z)P \xrightarrow[uc]{[x/y]} (z)P'}, z \neq x, y\,;\, x \neq y$$

$$1\text{-}PASS: \quad \frac{P \xrightarrow[uc]{1} P'}{(z)P \xrightarrow[uc]{1} (z)P'} \qquad SCOPE: \quad \frac{P \xrightarrow[uc]{[y/z]} P'}{(z)P \xrightarrow[uc]{1} P'\{y|z\}}, y \neq z$$

$$O\text{-}OPEN: \quad \frac{P \xrightarrow[uc]{z\lfloor x\rfloor} P'}{(x)P \xrightarrow[uc]{(x)z\lfloor x\rfloor} P'} z \neq x \qquad I\text{-}OPEN: \quad \frac{P \xrightarrow[uc]{z\lceil x\rceil} P'}{(x)P \xrightarrow[uc]{(x)z\lceil x\rceil} P'} z \neq x$$

There is no premise in any transition rule of this PV semantics, containing bound actions, except the premise of $CONG$. In this operational semantics, the only rules for generating bound actions are $CONG$, $O\text{-}OPEN$ and $I\text{-}OPEN$. In [9], the rules $O\text{-}OPEN$ and $I\text{-}OPEN$ were presented in a compressed way as a single general rule for *output* and *input* named $OPEN$. Also in [9], instead of the rules $O\text{-}PASS$, $I\text{-}PASS$, $U\text{-}PASS$ and $1\text{-}PASS$ there is a single rule $PASS$: $\dfrac{P \xrightarrow[uc]{\alpha} P'}{(z)P \xrightarrow[uc]{\alpha} (z)P'}, z \notin n(\alpha)$ which includes all these subcases. In fact, we use a corresponding rule $PASS$ for every possible form of α. The rules $U\text{-}PASS$ and $1\text{-}PASS$ have the form presented in Definition 3.3 because when α is the inert action $1 = [x/x]$, the names of α are $\emptyset$ and not x. However, the presentation here is actually the same as the one given in [9]; it only helps us later in presenting the nominal semantics of the monadic fusion calculus.

The operational semantics of monadic π-calculus [11] differs from that of the monadic fusion calculus in the following way: there are not (free) input prefixes in the π-calculus; instead, there is a bound input prefix written as $x(y).P$, where $x(y).P$ waits until a name is received along x, substitutes it for the bound variable y and continues as P. In $x(y).P$, the name y is bound; the occurrences of y in P indicate the places where the name received via x will replace y when the interaction acts. Thus, the input transition in the π-calculus is given by the rule INP_π : $\dfrac{}{x(y).P \xrightarrow[\pi]{(y)x\lceil y\rceil} P}$.

In the π-calculus there is no scoping operator similar to that of fusion calculus, but a related restriction operator $new\, x$ inherits the rules $PASS$ and $OPEN$. There is no $SCOPE$ rule because the update actions are not present in the π-calculus. The interaction rule is COM_π : $\dfrac{P \xrightarrow[\pi]{(y)x\lceil y\rceil} P', Q \xrightarrow[\pi]{x\lfloor z\rfloor} Q'}{P \mid Q \xrightarrow[\pi]{1} P'\{z|y\} \mid Q'}$. In the π-calculus, the action $x\lfloor y\rfloor$ is written as $\overline{x}\langle y\rangle$, the action $(y)x\lceil y\rceil$ is written as $x(y)$, and the action $(y)x\lfloor y\rfloor$ is written as $\overline{x}(y)$. The action 1 of fusion calculus is denoted by τ in the π-calculus. The π-calculus is isomorphic to a subcalculus of the fusion calculus formed by requiring that all input prefixes $z\lceil x\rceil.P$ occur immediately under a scope (x) of the object in the input prefix. This subcalculus is called up_π-calculus. The isomorphism between the up_π-calculus and the π-calculus is denoted by $\leftrightarrow$, and it is given by the homomorphism extension of the following two equivalences: $(x)z\lceil x\rceil.P \leftrightarrow z(x).P$ and $(z)P \leftrightarrow new\, zP$ whenever P is not a free input prefix. The connection between the up_π and π-calculus is presented in [9].

Theorem 3.4. *Let P be a process in the π-calculus, Q a process in the up_π-calculus, and $\circ$ the composition of relations.*

1. *If $P \leftrightarrow Q$ and $P \xrightarrow[\pi]{\gamma} P'$, then $Q \xrightarrow[uc]{\gamma} Q'$ and $P' \equiv \circ \leftrightarrow \circ \equiv Q'$;*
2. *If $P \leftrightarrow Q$ and $Q \xrightarrow[uc]{\gamma} Q'$, then $P \xrightarrow[\pi]{\gamma} P'$ and $P' \equiv \circ \leftrightarrow \circ \equiv Q'$.*

4. Nominal Fusion Calculus

In this section we (re)write the transition rules from Definition 3.3 in a more compact form by using specific nominal techniques. The transition rules in the nominal semantics of the fusion calculus are presented using both the universal quantifier $\forall$ and the nominal quantifier N. Using the same arguments as in Example 2.8, we can say that the set of the fusion calculus terms form an NFM-set, and the set of the fusion calculus terms modulo α-conversion form also an NFM-set and an FM-set. If we organize the set of variables as an NFM-set (variables in fusion calculus are atoms, and the set A of atoms can be organized as an NFM-set with the S_A-action given in Example 2.4 item 1), we can define the abstraction between a variable and a fusion calculus term in the sense of Definition 2.15. The variable x is bound in the expression $(x)P$; in the nominal approach we use the notion of abstraction, and write this expression as $[x]P$. Thus we can see bindings represented by the scope operator as α-abstractions defined in the FM framework. When applying $(z)P$, we eliminate z from $supp(P)$ (that is represented by the set of free names of P) which intuitively is a "binding of $supp(z)$ in P" in the sense of Definition 2.15.

Definition 4.1. *Processes in the nominal fusion calculus* are defined by

$$P, Q ::= 0 \mid z\lfloor x\rfloor.P \mid z\lceil x\rceil.P \mid P + Q \mid P|Q \mid [x]P.$$

Definition 4.2. *Actions of the nominal fusion calculus* are given by

$$\gamma ::= z\lfloor x\rfloor \mid z\lceil x\rceil \mid [x/z] \mid 1 \mid [x]z\lfloor x\rfloor \mid [x]\, z\lceil x\rceil.$$

As in the PV semantics of the fusion calculus, $z\lfloor x\rfloor$, $z\lceil x\rceil$ and $[x/z]$ are called free actions, and generically denoted by α. A transition labeled with a free action is denoted by $P \xrightarrow[nuc]{\alpha} Q$ in the nominal semantics of the fusion calculus.

The actions $[x]z\lfloor x\rfloor$ and $[x]z\lceil x\rceil$ are called bound actions. In fact the bound actions $[x]z\lfloor x\rfloor$ and $[x]z\lceil x\rceil$ are actually the same as $(x)z\lfloor x\rfloor$ and $(x)z\lceil x\rceil$; however, they are denoted slightly different just because to emphasize that we work in the nominal framework. The name x in the actions $[x]z\lfloor x\rfloor$ and $[x]z\lceil x\rceil$ also binds into the derivates (whenever we present a bound action of form $[x]z\lfloor x\rfloor$, we actually have an abstraction in the sense of Definition 2.15 between the atom x and the Cartesian pair $(z\lfloor x\rfloor, derivate)$). Similar for $[x]z\lceil x\rceil$. A transition labeled with an action (either free or bound) is denoted by $P \xrightarrow[nuc]{\gamma} Q$ in the nominal semantics of the fusion calculus.

We use $n(\gamma)$, $fn(\gamma)$, $bn(\gamma)$ to denote all the names occurring in γ, free names and bound names, respectively. By definition, $n(1) = \emptyset$. Since a function as bn is not defined in the nominal framework, we have:

γ	$n(\gamma)$	$fn(\gamma)$	$bn(\gamma)$
$z\lfloor x \rfloor$	$\{x, z\}$	$\{x, z\}$	$\emptyset$
$z\lceil x \rceil$	$\{x, z\}$	$\{x, z\}$	$\emptyset$
$[x/z]$	$\{x, z\}$	$\{x, z\}$	$\emptyset$
1	$\emptyset$	$\emptyset$	$\emptyset$
$[x]z\lfloor x \rfloor$	$\{z\}$	$\{z\}$	$\emptyset$
$[x]z\lceil x \rceil$	$\{z\}$	$\{z\}$	$\emptyset$

A possible nominal semantics for the fusion calculus can be given by highlighting each time the action in the transition rules of the usual (PV) semantics of the fusion calculus; however, a notation as bn cannot be used. Using Proposition 2.18, we obtain a property of the binding operator in fusion calculus.

Proposition 4.3. *If $[x]P = [y]Q$, then one of the following statements is true:*

- $x = y$ and $P = Q$;
- $x \neq y$ and $x\#Q$ and $P = (x\,y) \cdot Q$.

However, whenever $x\#Q$ we have $(x\,y) \cdot Q = Q\{x|y\}$ [6]. From Proposition 4.3 it is clear that two processes coincide in fusion calculus if and only if they are α-convertible. This is not assumed by convention; we can prove this by induction on the structure of processes by using Proposition 4.3.

Definition 4.4. The *nominal structural congruence* (denoted by $\equiv_n$) *in fusion calculus* is the smallest equivalence relation on the set of processes closed under the rules:

1. Processes which are obtained from one another by an α-conversion are congruent (in fact they are identified in the nominal framework);
2. $\forall P, Q, R.\ P + 0 \equiv_n P,\ P + Q \equiv_n Q + P,\ (P + Q) + R \equiv_n P + (Q + R)$;
3. $\forall P, Q, R.\ P \mid 0 \equiv_n P,\ P \mid Q \equiv_n Q \mid P,\ (P \mid Q) \mid R \equiv_n P \mid (Q \mid R)$;
4. $\forall x, y, P, Q.\ [x]0 \equiv_n 0,\ [x][y]P \equiv_n [y][x]P,\ [x](P + Q) \equiv_n [x]P + [x]Q$;
5. $\forall P.\ И x.\forall Q.\ [x](P \mid Q) \equiv_n P \mid [x]Q$.

The nominal structural congruence is not the equality relation in the nominal framework. Clearly, equal processes are congruent; however, the converse is not valid. For example, $[x][y]P = [y][x]P$ if and only if x, y are fresh for P (according to Proposition 2.18). However, we assume $[x][y]P \equiv_n [y][x]P$ for all x, y and P. In fact, a rule of form "$[x][y]P \equiv_n [y][x]P$ only if x, y are fresh for P" would be trivial, because such a rule would be implied by the first rule which states that α-equivalent processes are congruent. The forth rule for nominal structural congruence does not contradict Proposition 2.18, because the nominal structural congruence and the equality are different equivalence relations.

We make the distinction between global binding (realized by quantifiers) and local binding (realized by the scope operator as an α-abstraction) in a rule of the nominal

fusion calculus. If a variable appears bound once locally and another time globally in the same transition rule, this does not represent a logical contradiction. The binding provided by the scope operator is seen as a local binding only for the following process (for example in the expression $[x]P$, x binds the free occurrences of x only in P), and the binding made by the quantifiers $\forall$ and $\mathit{И}$ is seen as a global binding for the entire rule (formula). So, it is possible to have expressions of form $\forall x, ...f_1([x]P, ...)$ and $\mathit{И}x, ...f_2([x]P, ...)$ in some rules. This does not mean that x is bound twice in the same expression. We make the distinction between local binding (which is the binding only for the following process indicated by the scope operator) and global binding (which is the binding for the entire expression indicated by quantifiers). $\forall x, ...f_1([x]P, ...)$ expresses that $f_1([x]P, ...)$ is valid, and its validity does not depend on x. Also, $\mathit{И}x, ...f_2([x]P, ...)$ means that $f_2([x]P, ...)$ is valid for all but finitely many x (eventually these are valid iff x is fresh for an expression or for a process). For example, the expression $\forall z.\mathit{И}x.\forall P, P'.\dfrac{P \xrightarrow[nuc]{z\lfloor x\rfloor} P'}{[x]P \xrightarrow[nuc]{[x]z\lfloor x\rfloor} P'}$, where by $\cdot \xrightarrow[nuc]{} \cdot$ we denote a transition in the nominal semantics of the fusion calculus, is correctly written even though we might be tempted to say that x is bound twice. The variables x and z are globally bound once. The second binding given by the scope operator is a local one, and it is restricted to the process P (or respectively to the action $[x]z\lfloor x\rfloor$). Thus, to say that $\forall z.\mathit{И}x.\forall P, P'.\dfrac{P \xrightarrow[nuc]{z\lfloor x\rfloor} P'}{[x]P \xrightarrow[nuc]{[x]z\lfloor x\rfloor} P'}$ is the same as saying that $P \xrightarrow[nuc]{z\lfloor x\rfloor} P'$ implies $[x]P \xrightarrow[nuc]{[x]z\lfloor x\rfloor} P'$. if x satisfies a cofiniteness condition (proved later that is actually "x is fresh for z").

Definition 4.5. The following labeled transition rules define the **nominal semantics** of the (monadic version of) the fusion calculus. A transition of the form $P \xrightarrow{\gamma} Q$ (where γ is the action) in the nominal semantics of the mondaic fusion calculus is denoted by $P \xrightarrow[nuc]{\gamma} Q$.

1. $\forall x, z, P.z\lfloor x\rfloor.P \xrightarrow[nuc]{z\lfloor x\rfloor} P$
2. $\forall x, z, P.z\lceil x\rceil.P \xrightarrow[nuc]{z\lceil x\rceil} P$
3. $\forall \alpha, P, Q, P'.\dfrac{P \xrightarrow[nuc]{\alpha} P'}{P + Q \xrightarrow[nuc]{\alpha} P'}$
4. $\forall P, Q, P', Q', \gamma.\dfrac{P \equiv_n P' , Q \equiv_n Q',\ P \xrightarrow[nuc]{\gamma} Q}{P' \xrightarrow[nuc]{\gamma} Q'}$

5. $\forall P, P', Q, \alpha. \dfrac{P \xrightarrow[nuc]{\alpha} P'}{P \mid Q \xrightarrow[nuc]{\alpha} P' \mid Q}$

6. $\forall x, y, z, P, P', Q, Q' \dfrac{P \xrightarrow[nuc]{z\lceil x\rceil} P' \ , \ Q \xrightarrow[nuc]{z\lfloor y\rfloor} Q'}{P \mid Q \xrightarrow[nuc]{[y/x]} P' \mid Q'}$

7. $\forall x, y. \text{И} z. \forall P, P'. \dfrac{P \xrightarrow[nuc]{x\lfloor y\rfloor} P'}{[z]P \xrightarrow[nuc]{x\lfloor y\rfloor} [z]P'}$

8. $\forall x, y. \text{И} z. \forall P, P'. \dfrac{P \xrightarrow[nuc]{x\lceil y\rceil} P'}{[z]P \xrightarrow[nuc]{x\lceil y\rceil} [z]P'}$

9. $\forall x. \text{И} y. \text{И} z. \forall P, P'. \dfrac{P \xrightarrow[nuc]{[x/y]} P'}{[z]P \xrightarrow[nuc]{[x/y]} [z]P'}$

10. $\forall z, P, P'. \dfrac{P \xrightarrow[nuc]{1} P'}{[z]P \xrightarrow[nuc]{1} [z]P'}$

11. $\forall y. \text{И} z. \forall P, P'. \dfrac{P \xrightarrow[nuc]{[y/z]} P'}{[z]P \xrightarrow[nuc]{1} P'\{y|z\}}$

12. $\forall z. \text{И} x. \forall P, P'. \dfrac{P \xrightarrow[nuc]{z\lfloor x\rfloor} P'}{[x]P \xrightarrow[nuc]{[x]z\lfloor x\rfloor} P'}$

13. $\forall z. \text{И} x. \forall P, P' \dfrac{P \xrightarrow[nuc]{z\lceil x\rceil} P'}{[x]P \xrightarrow[nuc]{[x]z\lceil x\rceil} P'}$

We are able to provide a mathematical proof of the equivalence of the PV and of the nominal semantics of the fusion calculus. Firstly, we define a set of transition rules in the form given by Parrow and Victor (assuming some cofiniteness conditions in the rules hypotheses). Next, we prove that these rules are in fact identical with those in the nominal semantics of the fusion calculus.

Definition 4.6. A basic transition rule in the nominal semantics of the fusion calculus is in a *PV-nominal form* if it coincides with one of the following rules:

$$OUT_{PVn}: \ \frac{}{z\lfloor x\rfloor.P \xrightarrow[nuc]{z\lfloor x\rfloor} P} \qquad INP_{PVn}: \ \frac{}{z\lceil x\rceil.P \xrightarrow[nuc]{z\lceil x\rceil} P}$$

$$SUM_{PVn}: \ \frac{P \xrightarrow[nuc]{\alpha} P'}{P+Q \xrightarrow[nuc]{\alpha} P'} \qquad CONG_{PVn}: \ \frac{P \equiv_n P', Q \equiv_n Q', P \xrightarrow[nuc]{\gamma} Q}{P' \xrightarrow[nuc]{\gamma} Q'}$$

$$PAR_{PVn}: \ \frac{P \xrightarrow[nuc]{\alpha} P'}{P \mid Q \xrightarrow[nuc]{\alpha} P' \mid Q} \qquad COM_{PVn}: \ \frac{P \xrightarrow[nuc]{z\lceil x\rceil} P', Q \xrightarrow[nuc]{z\lfloor y\rfloor} Q'}{P \mid Q \xrightarrow[nuc]{[y/x]} P' \mid Q'}$$

$$O\text{-}PASS_{PVn}: \ \frac{P \xrightarrow[nuc]{x\lfloor y\rfloor} P'}{[z]P \xrightarrow[nuc]{x\lfloor y\rfloor} [z]P'}, z\#x\,;\, z\#y$$

$$I\text{-}PASS_{PVn}: \ \frac{P \xrightarrow[nuc]{x\lceil y\rceil} P'}{[z]P \xrightarrow[nuc]{x\lceil y\rceil} [z]P'}, z\#x\,;\, z\#y$$

$$U\text{-}PASS_{PVn}: \ \frac{P \xrightarrow[nuc]{[x/y]} P'}{[z]P \xrightarrow[nuc]{[x/y]} [z]P'}, z\#x\,;\, z\#y\,;\, y\#x$$

$$1\text{-}PASS_{PVn}: \ \frac{P \xrightarrow[nuc]{1} P'}{[z]P \xrightarrow[nuc]{1} [z]P'} \qquad SCOPE_{PVn}: \ \frac{P \xrightarrow[nuc]{[y/z]} P'}{[z]P \xrightarrow[nuc]{1} P'\{y|z\}}, z\#y$$

$$O\text{-}OPEN_{PVn}: \frac{P \xrightarrow[nuc]{z\lfloor x\rfloor} P'}{[x]P \xrightarrow[nuc]{[x]z\lfloor x\rfloor} P'}, x\#z \qquad I\text{-}OPEN_{PVn}: \frac{P \xrightarrow[nuc]{z\lceil x\rceil} P'}{[x]P \xrightarrow[nuc]{[x]z\lceil x\rceil} P'}, x\#z$$

We provide some results (lemmas) allowing to prove that *each transition rule in the nominal semantics of the fusion calculus can be presented in a PV-nominal form.* Later, we prove a result showing that the PV and the nominal semantics of the fusion calculus provide the same transitions.

Lemma 4.7.

1. *The rule number 1 in the nominal semantics of the fusion calculus is identical with the rule* OUT_{PVn}.

2. *The rule number 2 in the nominal semantics of the fusion calculus is identical with the rule* INP_{PVn}.

3. *The rule number 3 in the nominal semantics of the fusion calculus is identical with the rule* SUM_{PVn}.

4. *The rule number 4 in the nominal semantics of the fusion calculus is identical with the rule* $CONG_{PVn}$.

5. *The rule number 5 in the nominal semantics of the fusion calculus is identical with the rule* PAR_{PVn}.

6. *The rule number 6 in the nominal semantics of the fusion calculus is identical with the rule* COM_{PVn}.

7. *The rule number 10 in the nominal semantics of the fusion calculus is identical with the rule* 1-$PASS_{PVn}$.

In view of Proposition 2.13 and Proposition 2.14, we provide the following results.

Lemma 4.8. *Rule O-*$PASS_{PVn}$ *of Definition 4.6 is identical with the rule number 7 in the nominal semantics of the fusion calculus.*

Proof: Let p be the formula "$\forall P, P'.(P \xrightarrow[nuc]{x\lfloor y\rfloor} P'$ implies $[z]P \xrightarrow[nuc]{x\lfloor y\rfloor} [z]P')$". The free variables of p are contained in the set $\{x, y, z\}$. Indeed, the set of free variables of p is formed only by x, y because P, P' are bound by the quantifier $\forall$, and z is bound by the scope operator. If we assume the free variables of p to be only the variables of p which are globally unbound, then these free variables of p would be x, y, z and again these are contained in the set $\{x, y, z\}$. Now, according to Proposition 2.13 and Proposition 2.14, we have the equivalence result "$\forall z.(z\#\{x, y\}$ implies $p)$" if and only if "$И z.p$". Since x and y are arbitrarily chosen, we can say that rule O-$PASS_{PVn}$ of Def. 4.6 is identical with rule 7 in the nominal semantics of the fusion calculus. □

Lemma 4.9. *Rule I-*$PASS_{PVn}$ *of Definition 4.6 is identical with rule number 8 in the nominal semantics of the fusion calculus.*

Proof: Let p be the formula "$\forall P, P'.(P \xrightarrow[nuc]{x\lceil y\rceil} P'$ implies $[z]P \xrightarrow[nuc]{x\lceil y\rceil} [z]P')$". The free variables of p are contained in the set $\{x, y, z\}$. Indeed, the set of free variables of p is formed only by x, y because P and P' are bound by the quantifier $\forall$, and z is bound by the scope operator. If we assume the free variables of p to be only the variables of p which are globally unbound, then these free variables of p would be x, y, z, and again these are contained in the set $\{x, y, z\}$. According to Proposition 2.13 and Proposition 2.14, we have the equivalence result "$\forall z.(z\#\{x, y\}$ implies $p)$" if and only if "$И z.p$". Since x and y are arbitrarily chosen, we can say that rule I-$PASS_{PVn}$ of Definition 4.6 is identical with rule number 8 in Definition 4.5. □

Lemma 4.10. *Rule U-*$PASS_{PVn}$ *of Definition 4.6 is identical with rule number 9 in the nominal semantics of the fusion calculus.*

Proof: Let p be the formula "$\forall P, P'.(P \xrightarrow[nuc]{[x/y]} P'$ implies $[z]P \xrightarrow[nuc]{[x/y]} [z]P')$", and q be the formula $И z.\forall P, P'.(P \xrightarrow[nuc]{[x/y]} P'$ implies $[z]P \xrightarrow[nuc]{[x/y]} [z]P')$ which is precisely $И z.p$. The free variables of q are contained in the set $\{x, y\}$. Indeed, the set of free variables of q is formed only by x, y because P and P' are bound by the quantifier $\forall$, and z is bound by the nominal quantifier. If we assume the free variables of q to be only the variables of q which are globally unbound, then these free variables of q would be x, y,

and again these are contained in the set $\{x, y\}$. According to Proposition 2.13 and Proposition 2.14, we obtain the equivalence result "$\forall y.(y\#x$ implies $q)$" if and only if "$И y.q$". Now, the free variables of p are contained in the set $\{x, y, z\}$. Indeed, the set of free variables of p is formed only by x, y because P, P' are bound by the quantifier $\forall$, and z is bound by the scope operator. If we assume the free variables of p to be only the variables of p which are globally unbound, then the free variables of p would be x, y, z and again these are contained in the set $\{x, y, z\}$. According to Proposition 2.13 and Proposition 2.14, we obtain the equivalence result "$\forall z.(z\#\{x, y\}$ implies $p)$" if and only if "$И z.p$".

Let us assume that rule $U\text{-}PASS_{PVn}$ of Definition 4.6 is valid. Let $y\#x$ be arbitrarily chosen (fixed). From $U\text{-}PASS_{PVn}$ we know that "$\forall z.(z\#\{x, y\}$ implies $p)$" is valid; moreover "$\forall z.(z\#\{x, y\}$ implies $p)$" is valid even when $x = y$, because of rule $1\text{-}PASS_{PVn}$. This means that $И z.p$ is valid, and hence q is valid. Thus the implication "$\forall y.(y\#x$ implies $q)$" is valid, and hence $И y.q$ is valid. Since x was arbitrarily chosen, we can say that rule $U\text{-}PASS_{PVn}$ of Definition 4.6 implies rule number 9 in the nominal semantics of the fusion calculus.

Conversely, let us consider an arbitrary atom x, and assume that $И y.И z.p$ is valid, which means $И y.q$ is valid. Let $y\#x$ and $z\#x, y$. We must prove the validity of the proposition p: "$\forall P, P'.(P \xrightarrow[nuc]{[x/y]} P'$ implies $[z]P \xrightarrow[nuc]{[x/y]} [z]P')$". Since $y\#x$, we have that q is valid, which means $И z.p$ is valid. Since $z\#\{x, y\}$, it follows that p is valid, and hence $U\text{-}PASS_{PVn}$ is a valid rule. □

Lemma 4.11. *Rule $SCOPE_{PVn}$ of Definition 4.6 is identical with rule number 11 in the nominal semantics of the fusion calculus.*

Proof: Let p be the formula "$\forall P, P'.(P \xrightarrow[nuc]{[y/z]} P'$ implies $[z]P \xrightarrow[nuc]{[y/z]} P'\{y|z\})$" where $n([y/z]) = fn([y/z])$. The free variables of p are contained in the set $\{y, z\}$. Indeed the set of free variables of p is formed only by y because P, P' are bound by the quantifier $\forall$, and z is bound by the scope operator; if we assume the free variables of p to be only the variables of p which are globally unbound, then these free variables of p would be y, z and again these are contained in the set $\{y, z\}$. Now, because of Proposition 2.13 and Proposition 2.14, we obtain the equivalence result: "$\forall z.(z\#y$ implies $p)$" if and only if "$И z.p$". Since y was arbitrarily chosen we can say that rule $SCOPE_{PVn}$ of Definition 4.6 is identical with rule number 11 in the nominal semantics of the fusion calculus. □

Lemma 4.12. *Rule $O\text{-}OPEN_{PVn}$ of Definition 4.6 is identical with rule number 12 in the nominal semantics of the fusion calculus.*

Proof: Let p be the formula "$\forall P, P'.(P \xrightarrow[nuc]{z\lfloor x\rfloor} P'$ implies $[x]P \xrightarrow[nuc]{[x]z\lfloor x\rfloor} P')$" where $n(z\lfloor x\rfloor) = fn(z\lfloor x\rfloor)$ and $n([x]z\lfloor x\rfloor) = \{z\}$. The free variables of p are contained in the set $\{x, z\}$. Indeed the set of free variables of p is formed only by z because P, P' are bound by the quantifier $\forall$, and x is bound by the scope operator; if we assume the free variables of p to be only the variables of p which are globally unbound, then these free variables of p would be x, z and again these are contained in the set $\{x, z\}$.

Now, because of Proposition 2.13 and Proposition 2.14, we obtain the equivalence result: "$\forall x.(x\#z$ implies $p)$" if and only if "$И x.p$". Since z was arbitrarily chosen we can say that rule $O\text{-}OPEN_{PVn}$ of Definition 4.6 is identical with rule number 12 in the nominal semantics of the fusion calculus. □

Lemma 4.13. *Rule $I\text{-}OPEN_{PVn}$ of Definition 4.6 is identical with rule number 13 in the nominal semantics of the fusion calculus.*

Proof: Let p be the formula "$\forall P, P'.(P \xrightarrow[nuc]{z\lceil x\rceil} P'$ implies $[x]P \xrightarrow[nuc]{[x]z\lceil x\rceil} P')$" where $n(z\lceil x\rceil) = fn(z\lceil x\rceil)$ and $n([x]z\lceil x\rceil) = \{z\}$. The free variables of p are contained in the set $\{x, z\}$. Indeed the set of free variables of p is formed only by z because P, P' are bound by the quantifier $\forall$, and x is bound by the scope operator; if we assume the free variables of p to be only the variables of p unbound globally, then these free variables of p would be x, z and again these are contained in the set $\{x, z\}$. Now, according to Proposition 2.13 and Proposition 2.14, we obtain the equivalence result: "$\forall x.(x\#z$ implies $p)$" if and only if "$И x.p$". Since z was arbitrarily chosen we can say that rule $I\text{-}OPEN_{PVn}$ of Definition 4.6 is identical with rule number 13 in the nominal semantics of the fusion calculus. □

From Lemmas 4.7, 4.8, 4.9, 4.10, 4.11, 4.12, 4.13 we are able to say that each transition rule in Definition 4.5 can be expressed in a PV-nominal form. This leads to the following corollary:

Corollary 4.14. *Up to nominal structural congruence, the PV-nominal transition rules of Definition 4.6 also represent the* ***nominal semantics*** *of the (monadic version of) the fusion calculus.*

Definition 4.15. A basic structural congruence rule in the nominal fusion calculus is in a *PV-nominal form* if it coincides with one of the following rules:

1. Processes which are obtained from one another by an α-conversion are congruent; in fact they are identified in the nominal framework;
2. $P + 0 \equiv_n P$, $P + Q \equiv_n Q + P$, $(P + Q) + R \equiv_n P + (Q + R)$;
3. $P \mid 0 \equiv_n P$, $P \mid Q \equiv_n Q \mid P$, $(P \mid Q) \mid R \equiv_n P \mid (Q \mid R)$;
4. $[x]0 \equiv_n 0$, $[x][y]P \equiv_n [y][x]P$, $[x](P + Q) \equiv_n [x]P + [x]Q$;
5. $[x](P \mid Q) \equiv_n P \mid [x]Q$ if $x\#P$.

Lemma 4.16. *Rule number 5 of Definition 4.4 is identical with rule number 5 of Definition 4.5.*

Proof: Let p be the formula "$\forall Q.[x](P \mid Q) \equiv_n P \mid [x]Q$". The free variables of p are contained in the set $\{x, P\}$. Indeed the set of free variables of p is formed only by P (precisely only by the set of free names of P, which is included in P) because Q is bound by the quantifier $\forall$, and x is bound by the scope operator; if we assume the free variables of p to be only the variables of p which are globally unbound, then these free

variables of p would be x, P and again these are contained in the set $\{x, P\}$. Now, because of Proposition 2.13 and Proposition 2.14, we obtain the equivalence result: "$\forall x.(x\#P$ implies $p)$" if and only if "$И x.p$". Since P was arbitrarily chosen we can say that rule number 5 of Definition 4.4 is identical with rule number 5 of Definition 4.15. □

From Lemma 4.16 we are able to say that each structural congruence rule in Definition 4.4 can be expressed in a PV-nominal form. This leads to the following corollary:

Corollary 4.17. *The PV-nominal structural congruence rules of Definition 4.15 are identical with the nominal structural congruence rules in Definition 4.4.*

Now we can make a comparison between the PV semantics of the fusion calculus and the nominal semantics of the fusion calculus. First we remind the reader how we can prove, in the general theory, the equivalence of two semantics of the fusion calculus. The method presented here is similar with the method presented in [1] where the equivalence between several semantics of πI-calculus is proved. For proving that two semantics of the fusion calculus, namely u_1 (where a transition is denoted by $\rightarrow_1$, an action is denoted by γ_1 and a prefix is denoted by p_1) and u_2 (where a transition is denoted by $\rightarrow_2$, an action is denoted by γ_2 and a prefix is denoted by p_2), are *"equivalent"* or *"have the same expressive power"*, we should be able to define an encoding (morphism) $\varphi : u_1 \rightarrow u_2$ with a special property between the syntactic constructions, and to find a way to prove that everything that can be expressed with the transition rules in u_1 can also be expressed with the transition rules in u_2 as an image under the morphism φ and vice versa.

A practical way to do this is to define the morphism φ inductively by the following rules:

1. $\varphi(0) = 0$.
2. $\varphi(p_1^i.P_1) = p_2^i.\varphi(P_1)$, for each prefix p_1^i and process P_1, where $\{p_1^i\}$ are the prefixes in u_1, and each of the prefixes p_1^i has a corespondent p_2^i in u_2.
3. $\varphi(P_1|Q_1) = \varphi(P_1)|\varphi(Q_1)$, for each P_1, Q_1.
4. $\varphi((x)P_1) = (x)\varphi(P_1)$, for each x, P_1, where $(x)P$ is a general notation for the binding of x in P in various semantics ($(x)P$ could be the binding realized by the scope operator in the PV semantics or, respectively, the α-abstraction in the nominal semantics).

With φ defined in this way we should be able to prove that φ is surjective and its kernel is precisely the α-equivalence on u_1. To prove that u_1 and u_2 are equivalent semantics of the fusion calculus we must show the following implications (the classical procedure is by induction after the depth of the deduction tree):

- For each P_1, Q_1, γ_1 in u_1 we have that $P_1 \xrightarrow{\gamma_1}_1 Q_1$ implies $\varphi(P_1) \xrightarrow{\gamma_2}_2 \varphi(Q_1)$, where γ_2 is the related corespondent in u_2 of γ_1 (for example if u_1 is the PV semantics of the fusion calculus and u_2 is the nominal semantics of the fusion

calculus then: if $\gamma_1 = x\lfloor y\rfloor$ then $\gamma_2 = x\lfloor y\rfloor$, if $\gamma_1 = x\lceil y\rceil$ then $\gamma_2 = x\lceil y\rceil$, if $\gamma_1 = [x/y]$ then $\gamma_2 = [x/y]$, if $\gamma_1 = (x)x\lfloor y\rfloor$ then $\gamma_2 = [x]x\lfloor y\rfloor$, and if $\gamma_1 = (x)x\lceil y\rceil$ then $\gamma_2 = [x]x\lceil y\rceil$).

- For each P_2, Q_2, γ_2 in u_2, choosing fresh names for the bound atoms in P_2 and Q_2 such that $P_2 = \varphi(P_1)$ and $Q_2 = \varphi(Q_1)$, we have that $P_2 \xrightarrow{\gamma_2}_2 Q_2$ implies $P_1 \xrightarrow{\gamma_1}_1 Q_1$, where γ_2 is the related corespondent in u_2 of γ_1 (for example if u_2 is the nominal semantics of the fusion calculus and u_1 is the PV semantics of the fusion calculus then: if $\gamma_2 = x\lfloor y\rfloor$ then $\gamma_1 = x\lfloor y\rfloor$, if $\gamma_2 = x\lceil y\rceil$ then $\gamma_1 = x\lceil y\rceil$, if $\gamma_2 = [x/y]$ then $\gamma_1 = [x/y]$, if $\gamma_2 = [x]x\lfloor y\rfloor$ then $\gamma_1 = (x)x\lfloor y\rfloor$, if $\gamma_2 = [x]x\lceil y\rceil$ then $\gamma_1 = (x)x\lceil y\rceil$).

In our case, for the fusion calculus, things are very clear. Both in the PV semantics of the fusion calculus and in the nominal semantics of the fusion calculus, *the syntax of processes and the basic actions are actually the same* (see Definitions 4.1, 4.2). So, it is not necessary to define explicitly a morphism φ between the PV semantics of the fusion calculus and the nominal semantics of the fusion calculus as in the general theory. We can assume *φ to be a morphism which leaves the processes and the actions in both semantics unchanged (with the remark that the terms obtained after an α-conversion are identified in the the framework of nominal sets, and the bindings in the nominal framework are in sense of Definition 2.15) and whose kernel is the α-equivalence on the PV semantics of the (monadic version of the) fusion calculus.*

Such a φ is inductively induced by the nominal abstraction. The main property of φ is that for any process P in u_1 (when u_1 is the PV semantics of the fusion calculus) we have $z \notin fn(P)$ iff $z\#\varphi(P)$ (easy to prove by induction on the syntax of φ). These properties of φ allow us to make the convention of eliminating φ in the proof of the following theorem (φ is seen as an *inclusion* for easy writing).

A connection between the PV structural congruence and the nominal structural congruence is given in the following theorem:

Theorem 4.18. *For any two processes P and Q we have the following equivalence:*

$$P \equiv Q \text{ if and only if } P \equiv_n Q \ .$$

Proof: For the nominal structural congruence in the fusion calculus we consider the set of structural congruence rules presented in Definition 4.15 instead of the structural congruence rules presented in Definition 4.4. By Corollary 4.17, these two sets of rules coincide. Each structural congruence of the form $P \equiv_n Q$ is obtained by applying repeatedly the structural congruence rules in Definition 4.15. We have to prove a double implication. The first part will be proved by induction after the depth of the deduction tree of $\equiv$, the second by induction after the depth of the deduction tree of $\equiv_n$. To save space we do not rewrite out the induction hypotheses in complete formality. We analyze only one case. The rest of them are trivial. First we prove that $P \equiv Q$ implies $P \equiv_n Q$ (in fact $P \equiv Q$ implies $\varphi(P) \equiv_n \varphi(Q)$; however we can eliminate φ and write $\varphi(P)$ as P with the remark that the terms obtained after an α-conversion are identified, and the bindings in the in the nominal framework are in the sense of Definition 2.15).

We analyze the case of rule number 5 in Definition 3.2. Suppose $(x)(P \mid Q) \equiv P \mid (x)Q$ with $x \notin fn(P)$. Then, because $fn(P) = supp(P)$ (Example 2.8), we have $x\#P$ and hence $[x](P \mid Q) \equiv P \mid [x]Q$.

Conversely we prove that $P \equiv_n Q$ implies $P \equiv Q$. We analyze only the case of rule number 5 in Definition 4.15. Suppose $[x](P \mid Q) \equiv P \mid [x]Q$ with $x\#P$. Then, because $fn(P) = supp(P)$ (Example 2.8) we have $x \notin fn(P)$ and hence $(x)(P \mid Q) \equiv P \mid (x)Q$. □

Remark 4.19. *If we apply step by step the general theory, in the proof of the Theorem 4.18 we obtain (by induction) that $P \equiv_n Q$ implies $P' \equiv Q'$ where $P = \varphi(P')$ and $Q = \varphi(Q')$ for fresh choices of bound names in P and respectively in Q. Since* φ leaves the processes and the actions in both semantics unchanged, *we have that $P = \varphi(P)$ $Q = \varphi(Q)$. Now, because the kernel of φ is precisely the α-equivalence in PV, it follows that $P \equiv_\alpha P'$ and $Q \equiv_\alpha Q'$ and hence $P \equiv Q$.*

The main result in this paper is the following theorem which provides the connection between the PV and the nominal semantics of the fusion calculus.

Theorem 4.20. *For any two processes P and Q and any action γ we have:*

$$P \xrightarrow[uc]{\gamma} Q \text{ if and only if } P \xrightarrow[nuc]{\gamma} Q \ .$$

Proof: For the nominal semantics of the fusion calculus we consider the set of transition rules presented in Definition 4.6 instead of the transition rules presented in Definition 4.5. By Corollary 4.14, these two sets of transition rules coincide. Each transition of the form $P \xrightarrow[nuc]{\gamma} Q$ is obtained by applying repeatedly the basic transition rules in Definition 4.6. We have to prove a double implication. The first part will be proved by induction after the depth of the deduction tree of *uc*, the second by induction after the depth of the deduction tree of *nuc*. To save space we do not rewrite out the induction hypotheses in complete formality. We analyze only a few cases. The rest of them are very similar or trivial. First we prove that $P \xrightarrow[uc]{\gamma} Q$ implies $P \xrightarrow[nuc]{\gamma} Q$.

Case *U-PASS*. Suppose $P \xrightarrow[uc]{[x/y]} P'$ where $z \neq x, y\,; x \neq y$. Then $(z)P \xrightarrow[uc]{[x/y]} (z)P'$. By the inductive hypothesis we have $P \xrightarrow[nuc]{[x/y]} P'$. Since $supp(\{x, y\}) = \{x, y\}$ and $supp(\{x\}) = \{x\}$, we have that $z \neq x, y$ is equivalent with $z \notin supp(\{x, y\})$ which is $z\#x, y$, and $y \neq x$ is equivalent with $y \notin supp(\{x\})$ which is $y\#x$. We can apply rule $U\text{-}PASS_{PVn}$ and we get $[z]P \xrightarrow[nuc]{[x/y]} [z]P'$.

Case *SCOPE*. Suppose $P \xrightarrow[uc]{[y/z]} P'$ where $y \neq z$. Then $(z)P \xrightarrow[uc]{1} P'\{y|z\}$. By the inductive hypothesis we have $P \xrightarrow[nuc]{[y/z]} P'$. Since $supp(\{y\}) = \{y\}$ we have that $z \neq y$ is equivalent with $z \notin supp(\{y\})$ which is $z\#y$. We can apply rule $SCOPE_{PVn}$ and we get $[z]P \xrightarrow[nuc]{1} P'\{y|z\}$.

Case *O-OPEN*. Suppose $P \xrightarrow[uc]{z\lfloor x\rfloor} P'$ where $z \neq x$. Then $(x)P \xrightarrow[uc]{(x)z\lfloor x\rfloor} P'$. By the inductive hypothesis we have $P \xrightarrow[nuc]{z\lfloor x\rfloor} P'$. Since $supp(\{z\}) = \{z\}$ we have that $x \neq z$ is equivalent with $x \notin supp(\{z\})$ which is $x\#z$. We can apply rule $O\text{-}OPEN_{PVn}$ and we get $[x]P \xrightarrow[nuc]{[x]z\lfloor x\rfloor} P'$.

We prove $P \xrightarrow[nuc]{\gamma} Q$ implies $P \xrightarrow[uc]{\gamma} Q$.

Case $U\text{-}PASS_{PVn}$. Suppose $P \xrightarrow[nuc]{[x/y]} P'$ where $z\#x, y$; $y\#x$. Then $[z]P \xrightarrow[nuc]{[x/y]} [z]P'$. By the inductive hypothesis we have $P \xrightarrow[uc]{[x/y]} P'$. Since $supp(\{x, y\}) = \{x, y\}$ and $supp(\{x\}) = \{x\}$, we have that $z\#x, y$ is equivalent with $z \notin supp(\{x, y\})$ which is $z \neq x, y$, and $y\#x$ is equivalent with $y \notin supp(\{x\})$ which is $y \neq x$. We can apply rule *U-PASS* and we get $(z)P \xrightarrow[uc]{[x/y]} (z)P'$.

Case $SCOPE_{PVn}$. Suppose $P \xrightarrow[nuc]{[y/z]} P'$ where $z\#y$. Then $[z]P \xrightarrow[nuc]{1} P'\{y|z\}$. By the inductive hypothesis we have $P \xrightarrow[uc]{[y/z]} P'$. Since $supp(\{y\}) = \{y\}$ we have that $z\#y$ is equivalent with $z \notin supp(\{y\})$ which is $z \neq y$. We can apply rule *SCOPE* and we get $(z)P \xrightarrow[uc]{1} P'\{y|z\}$.

Case $O\text{-}OPEN_{PVn}$. Suppose $P \xrightarrow[nuc]{z\lfloor x\rfloor} P'$ where $x\#z$. Then $[x]P \xrightarrow[nuc]{[x]z\lfloor x\rfloor} P'$. By the inductive hypothesis we have $P \xrightarrow[uc]{z\lfloor x\rfloor} P'$. Since $supp(\{z\}) = \{z\}$ we have that $x\#z$ is equivalent with $x \notin supp(\{z\})$ which is $x \neq z$. We can apply rule *O-OPEN* and we get $(x)P \xrightarrow[uc]{(x)z\lfloor x\rfloor} P'$. □

Remark 4.21. *If we apply step by step the method of proving the equivalence of several semantics presented before, in the proof of the Theorem 4.20 we obtain (by induction) that* $P \xrightarrow[nuc]{\gamma} Q$ *implies* $P' \xrightarrow[uc]{\gamma} Q'$ *where* $P = \varphi(P')$ *and* $Q = \varphi(Q')$ *for fresh choices of bound names in* P *and respectively in* Q. *Since* φ leaves the processes and the actions in both semantics unchanged, *we have that* $P = \varphi(P)$ $Q = \varphi(Q)$. *Now, because the kernel of* φ *is precisely the* α*-equivalence in PV, it follows that* $P \equiv_\alpha P'$ *and* $Q \equiv_\alpha Q'$. *By applying rule CONG in the PV semantics of the fusion calculus, we also obtain that* $P \xrightarrow[uc]{\gamma} Q$. *The procedure described in this Remark is trivial. We just write:* $P \xrightarrow[nuc]{\gamma} Q$ *implies* $P \xrightarrow[uc]{\gamma} Q$.

5. Conclusions

Notions as renaming, binding and fresh names appear in several approaches; it became evident that they deserve to be studied in their own terms. Nominal sets are related to the binding operators appearing in computer science, and represent an alternative set theory, with a more relaxed notion of finiteness. They offer an elegant

formalism for describing λ-terms modulo α-conversion [6], automata on data words [5], or languages over infinite alphabets [4]. A survey on the applications of the theory of nominal sets in various areas of computer science is presented in Section 6 of [3].

Process calculi are used as a formal framework for the study of concurrent computation. Several versions of π-calculus [7] and other process algebras use one-to-one interactions, and it is difficult to describe complex systems where one-to-many and many-to-many interactions are emerging.

This paper is an extended version of [2]. In this paper we rewrite the basic transition rules from the monadic fusion calculus in a compact form, and provide the nominal semantics of this calculus. The transition rules of this semantics and the rules for structural congruence are expressed by using nominal techniques. We get a new presentation of the transition rules, and of the structural congruence rules which uses the quantifiers $\forall$ and the *nominal quantifier* N instead of side conditions. The transition rules of the fusion calculus are expressed without using supplementary freshness conditions in their hypotheses. This compact presentation is obtained by using the finite support property, Proposition 2.13, Proposition 2.14, and other technical results presented in the second part of the paper. Finally, we can prove that there is an equivalence between the nominal semantics and the usual (PV) semantics of the (monadic version of the) fusion calculus presented in [9] (see Theorem 4.20). A similar equivalence result for the structural congruence is presented in Theorem 4.18. Our paper establishes an agreement between different presentations which, potentially, may break down for more powerful systems.

A related nominal semantics of the πI-calculus is presented in [1]. Using similar techniques as in this paper, the authors proved a complete equivalence between the nominal semantics and the usual semantics of the πI-calculus [11].

Acknowledgements. The work was supported by a grant of the Romanian National Authority for Scientific Research, CNCS-UEFISCDI, project number PN-II-ID-PCE-2011-3-0919.

References

[1] ALEXANDRU A., CIOBANU G., *Nominal semantics for the πI-calculus*, Romanian Journal of Information, Science and Technology, **16**(4), pp. 261–286, 2013.

[2] ALEXANDRU A., CIOBANU G., *Nominal fusion calculus*, Proc. 14th SYNASC, IEEE Computer Society Press, pp. 376–384, 2013.

[3] ALEXANDRU A., CIOBANU G., *Nominal groups and their homomorphism theorems.*, Fundamenta Informaticae, **131**(3–4), pp. 279–298, 2014.

[4] BOJANCZYK M., *Nominal monoids*, Theory of Computing Systems, **53**(2), pp. 194–222, 2013.

[5] BOJANCZYK M., KLIN B., LASOTA S., *Automata with group actions*, Proc. 26th IEEE Symposium on Logic in Computer Science (LICS 2011), IEEE Computer Society Press, pp. 355–364, 2011.

[6] GABBAY M., PITTS A., *A new approach to abstract syntax with variable binding*, Formal Aspects of Computing, **13**, pp. 341–363, 2002.

[7] MILNER R., *Communicating and Mobile Systems: the π-Calculus*, Cambridge University Press, 1999.

[8] PARROW P., VICTOR B., *The fusion calculus: expressiveness and symmetry in mobile processes*, Proc. 13th IEEE Symposium on Logic in Computer Science (LICS 1998), IEEE Computer Society Press, pp. 176–185, 1998.

[9] PARROW P., VICTOR B., *The update calculus*, Proc. 6th Conference on Algebraic Methodology and Software Technology (AMAST'97), Lecture Notes in Computer Science, vol. **1349**, pp. 409–423, 1997.

[10] PITTS. A., *Alpha-structural recursion and induction*, Journal of the ACM, **53**, pp. 459–506, 2006.

[11] SANGIORGI D., WALKER D., *The π-Calculus: a Theory of Mobile Processes*, Cambridge University Press, 2001.