Fundamenta Informaticae XX (2011) 1–26
IOS Press

An Operational Petri Net Semantics for A^{2}CCS

Roberto Gorrieri **Cristian Versari**
Dipartimento di Scienze dell'Informazione, Università di Bologna,
Mura A. Zamboni, 7, 40127 Bologna, Italy
email: {gorrieri, versari}@cs.unibo.it

Abstract. A^{2}CCS is a conservative extension of CCS, enriched with an operator of strong prefixing, enabling the modeling of atomic sequences and multi-party synchronization (realized as an atomic sequence of binary synchronizations); the classic dining philosophers problem is used to illustrate the approach. A step semantics for A^{2}CCS is also presented directly as a labeled transition system. A safe Petri net semantics for this language is presented, following the approach of Degano, De Nicola, Montanari and Olderog. We prove that a process p and its associated net $Net(p)$ are interleaving bisimilar (Theorem 5.1). Moreover, to support the claim that the intended concurrency is well-represented in the net, we also prove that a process p and its associated net $Net(p)$ are step bisimilar (Theorem 5.2).

1. Introduction

CCS [19], even if largely used in academia, is not expressive enough to model some useful behaviors. For instance, there is no means to express that certain sequences of actions are to be executed atomically; similarly, it is not possible to express multi-party synchronization where more than two processes communicate. A conservative extension of CCS to model atomic actions and multiway synchronization was presented in [16, 15, 13] under the name of A^{2}CCS. It extends CCS with an additional operator of *strong* prefixing (in opposition to *normal* prefixing $\mu.P$), denoted $\underline{\mu}.P$, that is used to create atomic sequences, that can synchronize according to some commit-based discipline on sequences. Classical problems where a transactional behavior is required can be now modeled faithfully. As a typical example of this class of problems, we will consider Dijkstra's dining philosophers problem, which is solved by imposing that the acquisition of the two forks is an atomic action.

Petri net semantics for process algebras have received a lot of attention in the past (see, e.g., [7, 10, 11, 21, 2], just to mention a few). The main aim of these semantics is to offer a better understanding of the causality structure of process algebraic specifications of distributed systems (useful in some cases, e.g., for error recovery) and possibly also to offer some further verification techniques developed for

Petri nets (e.g., net invariants). Here we intend to provide also A^{2}CCS with a net semantics, as a further step towards a possible fruitful cross-fertilization between the areas of process algebra and Petri nets.

The approach we follow was initiated by Degano, De Nicola and Montanari (hereafter named DDM approach) and made popular by Olderog [7, 21]. It exploits the syntactic structure of the process terms (notably, the parallel operator) to define their associated sets of places in a safe Place/Transition Petri net. This approach has the merit of being very general and has been successfully applied to several process algebras (e.g., [8, 3]). However, the resulting safe Petri net is finite if and only if the original interleaving semantics of the process generates a finite-state labeled transition system; moreover, it usually produces rather huge nets, even in case of finite ones.

Another approach – initiated by Goltz [11, 12] and developed also in [14] – ignores the syntactic structure of the parallel operator and exploits only the information on the label of the transitions to produce unsafe P/T Petri nets. This approach has the merit of producing small(er) net representations, but has shown limited applicability so far, because of problems with certain operators, notably CCS restriction and unguarded sum. This approach has been conservatively extended, in order to cope with these operators, to a technique – initiated by Busi and Gorrieri (hereafter named BG approach) in [4, 1] – which generates contextual nets (i.e., unsafe P/T nets with inhibitor arcs and, possibly, also read arcs). The applicability of the BG approach seems less wide, as some process algebra operators cannot be dealt with very easily (e.g., sequential composition); nonetheless, it has been applied also to π-calculus [5, 6], a process algebra for which the DDM approach would offer unpractically large nets. Unfortunately, we cannot apply the BG approach to A^{2}CCS because its parallel composition operator is *not* associative, a necessary prerequisite for BG applicability. On the contrary, the DDM approach is so general that can be applied to any form of parallel composition, be it associative or not.

In order to avoid useless intricacies, the sum operator is always guarded by prefix (normal or strong): an assumption that is usually done in modern process algebras [20]. This makes the DDM approach for A^{2}CCS (and so also for CCS) very simple and intuitive, avoiding to deal with the problem of distributed choice (see [7, 21] for a discussion of this issue). The main results of the paper are the following:

- **Soundness.** The interleaving marking graph associated to the Petri net $Net(p)$ for any A^{2}CCS term p tightly corresponds to its interleaving transition system. To be more precise, p and $Net(p)$ are bisimilar (Theorem 5.1).

- **Non-interleaving Semantics.** A step semantics for A^{2}CCS is induced from the corresponding one for P/T nets. In order to strengthen the trust in our net semantics, we define directly a step transition system for A^{2}CCS and prove that, for any p, the step transition system of p is (step) bisimilar to the step marking graph of $Net(p)$ (Theorem 5.2).

- **Finiteness** We prove that $Net(p)$ is finite if and only if p is finite-state (Proposition 5.5); a sufficient condition for finiteness is that neither parallel composition nor restriction can occur inside a recursion.

The paper is organized as follows. Section 2 contains the background on labeled transition systems, bisimulation semantics, and P/T Petri nets. Section 3 contains an overview of A^{2}CCS, its interleaving semantics, bisimulation equivalence, a new step semantics for the calculus and also presents the case study of the dining philosophers [9]. Section 4 introduces the operational net semantics for A^{2}CCS and the safe net for the dining philosophers. Section 5 discusses the soundness of the DDM semantics

w.r.t. interleaving and step semantics, and also the finiteness result. Finally, Section 6 reports some conclusions.

2. Background

Here we recall the basic definition about transition systems and Petri net systems we will use in the following.

2.1. Labeled transition systems

Definition 2.1. A *labeled transition system* (or *lts*, for short) is a triple $TS = (St, A, \rightarrow)$ where

- St is the set of states;
- A is the set of labels;
- $\rightarrow \subseteq St \times A \times St$ is the transition relation.

In the following we use $s \xrightarrow{a} s'$ to denote $(s, a, s') \in \rightarrow$. A *rooted* lts is a pair (TS, s_0) where $TS = (St, A, \rightarrow)$ is a lts and $s_0 \in St$ is the *initial state*. A *path* from s_1 to s_{n+1} is a sequence of transitions $s_1 \xrightarrow{a_1} \ldots s_n \xrightarrow{a_n} s_{n+1}$. We say that s' is *reachable* from s if there exists a path from s to s'. We denote with $Reach(s)$ the set of the states reachable from s. An lts $TS = (St, A, \rightarrow)$ is *finite-state* if St is finite and $\rightarrow$ is a finite relation. □

Definition 2.2. A *bisimulation* between TS_1 and TS_2 is a relation $R \subseteq (St_1 \times St_2)$ such that if $(s_1, s_2) \in R$ then for all $a \in (A_1 \cup A_2)$

- $\forall s_1'$ such that $s_1 \xrightarrow{a} s_1'$, $\exists s_2'$ such that $s_2 \xrightarrow{a} s_2'$ and $(s_1', s_2') \in R$
- $\forall s_2'$ such that $s_2 \xrightarrow{a} s_2'$, $\exists s_1'$ such that $s_1 \xrightarrow{a} s_1'$ and $(s_1', s_2') \in R$.

If $TS_1 = TS_2$ we say that R is a bisimulation on TS_1. Two states s and s' are bisimilar, $s \sim s'$, if there exists a bisimulation R such that $(s, s') \in R$. □

2.2. P/T nets

We report some basic notions on P/T Petri nets (see, e.g., [26] for more details). We use here a non standard notation that better suits our needs.

Notation. Let $\mathbb{N}$ be the set of natural numbers. Given a set S, a *finite multiset* over S is a function $m : S \rightarrow \mathbb{N}$ such that the set $dom(m) = \{s \in S \mid m(s) \neq 0\}$ is finite. The *multiplicity* of an element s in m is given by the natural number $m(s)$. The set of all finite multisets over S, denoted by $\mathcal{M}_{fin}(S)$, is ranged over by m. The set of all finite sets over S is denoted by $\wp_{fin}(S)$. We write $m \subseteq m'$ if $m(s) \leq m'(s)$ for all $s \in S$. The operator $\oplus$ denotes *multiset union*: $(m \oplus m')(s) = m(s) + m'(s)$. The operator $\setminus$ denotes *multiset difference*: $(m \setminus m')(s) = m(s) - m'(s)$ if $m(s) > m'(s)$, 0 otherwise. The *scalar product* of a number j with multiset m is $(j \cdot m)(s) = j \cdot (m(s))$. A finite multiset m can be equivalently represented as $k_1 s_{i_1} \oplus k_2 s_{i_2} \oplus \ldots \oplus k_n s_{i_n}$, where $dom(m) = \{s_{i_1}, \ldots, s_{i_n}\}$ and $k_j = m(s_{i_j})$ for $j = 1, \ldots, n$.

Definition 2.3. A *labeled P/T net* is a tuple $N = (S, A, T)$, where

- S is the set of *places*;
- A is a set of labels;
- $T \subseteq \wp_{fin}(S) \times A \times \wp_{fin}(S)$ is the set of transitions.

A P/T net is *finite* if both S and T are finite. Given a transition $t = (m, a, m')$, we use the notation ${}^\bullet t$ to denote its *preset* m, $t^\bullet$ for its *postset* m' and $l(t)$ for its label a. Hence, transition t can be also represented as ${}^\bullet t \xrightarrow{l(t)} t^\bullet$. A finite multiset over the set S of places is called a *marking*. Given a marking m and a place s, we say that the place s contains $m(s)$ *tokens*. □

Note that our definition of T as a set of triples ensures that the net is *transition simple*, i.e., for any $t_1, t_2 \in T$, if ${}^\bullet t_1 = {}^\bullet t_2$ and $t_1^\bullet = t_2^\bullet$ and $l(t_1) = l(t_2)$, then $t_1 = t_2$.

Definition 2.4. Given a net $N = (S, A, T)$, a transition t is *enabled* at marking m, written as $m[t\rangle$, if ${}^\bullet t \subseteq m$. The execution of a transition t enabled at m produces the marking $m' = (m \setminus {}^\bullet t) \oplus t^\bullet$. This is usually written as $m[t\rangle m'$.

A finite, non empty multiset over the set T is called a *step*. A step G is enabled at m if $m_1 \subseteq m$, where $m_1 = \bigoplus_t G(t) \cdot {}^\bullet t$. The execution of a step G enabled at m produces the marking $m' = (m \setminus m_1) \oplus m_2$, where $m_2 = \bigoplus_t G(t) \cdot t^\bullet$. This is written as $m[G\rangle m'$.

A *P/T system* is a tuple $N(m_0) = (S, A, T, m_0)$, where (S, A, T) is a P/T net and m_0 is a finite multiset over S, called the *initial marking*. The set of markings *reachable* from m, denoted $[m\rangle$, is defined as the least set such that

- $m \in [m\rangle$ and
- if $m_1 \in [m\rangle$ and, for some transition $t \in T$, $m_1[t\rangle m_2$, then $m_2 \in [m\rangle$.

We say that m is reachable if m is reachable from the initial marking m_0.

A P/T system is said to be *safe* if any place contains at most one token in any reachable marking, i.e. $m(s) \leq 1$ for all $s \in S$ and for all $m \in [m_0\rangle$. □

Definition 2.5. The *interleaving marking graph* of $N(m_0)$ is the rooted lts

$IMG(N(m_0)) = ([m_0\rangle, A, \rightarrow, m_0)$

where m_0 is the initial state and the transition relation $\rightarrow \subseteq \mathcal{M}_{fin}(S) \times A \times \mathcal{M}_{fin}(S)$ is defined by $m \xrightarrow{a} m'$ iff there exists a transition $t \in T$ such that $m[t\rangle m'$ and $a = l(t)$. The P/T systems $N_1(m_1)$ and $N_2(m_2)$ are *interleaving bisimilar* ($N_1 \sim N_2$) iff there exists a strong bisimulation relating the initial states of $IMG(N_1(m_1))$ and $IMG(N_2(m_2))$.

The *step marking graph* of $N(m_0)$ is the rooted lts

$SMG(N(m_0)) = ([m_0\rangle, \mathcal{M}_{fin}(A), \rightarrow, m_0)$

where $\rightarrow \subseteq \mathcal{M}_{fin}(S) \times \mathcal{M}_{fin}(A) \times \mathcal{M}_{fin}(S)$ is defined by $m \xrightarrow{A} m'$ iff there exists a step G such that $m[G\rangle m'$ and $A = l(G)$.[1] The P/T systems $N_1(m_1)$ and $N_2(m_2)$ are *step bisimilar* ($N_1 \sim_{step} N_2$) iff there exists a strong bisimulation relating the initial states of $SMG(N_1(m_1))$ and $SMG(N_2(m_2))$ □

[1] The labeling function l is extended to multisets of transitions in the obvious way: $l(G)(a) = \sum_{t_i \in dom(G).l(t_i)=a} G(t_i)$.

A useful observation is that the state space of a finite safe P/T net is finite.

Proposition 2.1. The interleaving marking graph $IMG(N)$ of a finite, safe P/T net N is a finite-state lts.

The DDM-operational P/T net semantics we will define for A^{2}CCS generates always safe P/T nets, as we will see in Section 4.

3. Adding Atomic Actions to CCS

In this section we present the language A^{2}CCS, originally introduced in [16, 15], then the case study and a novel step semantics for this language.

3.1. A^{2}CCS

Let $\mathcal{L}$ be a denumerable set of channel names, ranged over by $a, b, \ldots$. Let $\overline{\mathcal{L}}$ the disjoint set of co-names, ranged over by $\overline{a}, \overline{b}, \ldots$. The set $\mathcal{L} \cup \overline{\mathcal{L}}$, ranged over by $\alpha, \beta, \ldots$, is the set of visible actions. With $\overline{\alpha}$ we mean the complement of α, assuming that $\overline{\overline{\alpha}} = \alpha$. Let $Act = \mathcal{L} \cup \overline{\mathcal{L}} \cup \{\tau\}$, such that $\tau \notin \mathcal{L} \cup \overline{\mathcal{L}}$, be the set of actions, ranged over by $\mu, \ldots$. Action τ denotes an invisible activity. Let $\mathcal{X}$ be a denumerable set of process variables, disjoint from Act, ranged over by $X, Y, \ldots$. The process terms are generated by the following grammar, where we are using two syntactic categories: p, to range over sequential processes (i.e., processes that start sequentially), and q, to range over any kind of processes.

$$p ::= \mathbf{0} \;\mid\; \mu.q \;\mid\; \underline{\mu}.q \;\mid\; p + p \quad \textit{sequential processes}$$

$$q ::= p \;\mid\; q \,|\, q \;\mid\; (\nu x)q \;\mid\; q[f] \;\mid\; X \;\mid\; recX.q \quad \textit{processes}$$

Term $\mathbf{0}$ is the terminated process, $\mu.q$ is a normally prefixed process where action μ (that can be either an input a, an output $\bar{a}$ or a silent move τ) is first performed and then q is ready, $\underline{\mu}.q$ is a strongly prefixed process where μ is the first action of a transaction that continues with q (provided that q can complete the transaction). Note that $p + p'$ is the sequential process obtained by the alternative composition of *sequential* processes p and p'; hence we are restricting the use of $+$ to so-called *guarded sum*. For instance, a term of the form $(a.\mathbf{0} \,|\, b.\mathbf{0}) + c.\mathbf{0}$ is not legal because the first summand is not sequential. This limitation is rather inessential for expressiveness, and moreover makes easier the development of the theory in Section 4. Term $q \,|\, q'$ is the parallel composition of q and q', $(\nu a)q$ is process q where the name a is made private (restriction), $q[f]$ is process q where its actions are renamed according to function $f : Act \to Act$ (such that $f(\overline{\alpha}) = \overline{f(\alpha)}$ and $f(\tau) = \tau$), X is a process variable and $recX.q$ is the usual recursion construct.

The set $\mathcal{P}$ of *processes* contains those terms which are, w.r.t. process variables, *closed* (all the variables occur in a rec binder) and *guarded* (all the recursion variables occurring in the body q of a recursion occurs inside a *normally prefixed* subprocess $\mu.q'$ of q). With abuse of notation, $\mathcal{P}$ will be ranged over by $p, q, \ldots$, i.e., p may denote any kind of process terms, not only sequential ones.

The operational semantics for A^{2}CCS is given by the labeled transition system $(\mathcal{P}, \mathcal{A}, \longrightarrow)$, where the states are the processes in $\mathcal{P}$, $\mathcal{A} = Act^*$ is the set of labels (ranged over by σ), and $\longrightarrow \subseteq \mathcal{P} \times \mathcal{A} \times \mathcal{P}$ is the minimal transition relation generated by the SOS rules [24, 25] listed in Table 1.

(Pref)	$\mu.p \xrightarrow{\mu} p$	(S-pref)	$\dfrac{p \xrightarrow{\sigma} p'}{\underline{\mu}.p \xrightarrow{\mu\sigma} p'}$
(Sum)	$\dfrac{p \xrightarrow{\sigma} p'}{p+q \xrightarrow{\sigma} p'}$	(Com)	$\dfrac{p \xrightarrow{\sigma_1} p' \qquad q \xrightarrow{\sigma_2} q'}{p \mid q \xrightarrow{\sigma} p' \mid q'} \quad Sync(\sigma_1, \sigma_2, \sigma)$
(Par)	$\dfrac{p \xrightarrow{\sigma} p'}{p \mid q \xrightarrow{\sigma} p' \mid q}$	(Res)	$\dfrac{p \xrightarrow{\sigma} p'}{(\nu a)p \xrightarrow{\sigma} (\nu a)p'} \quad a, \bar{a} \notin n(\sigma)$
(Rel)	$\dfrac{p \xrightarrow{\sigma} p'}{p[f] \xrightarrow{f(\sigma)} p'[f]}$	(Rec)	$\dfrac{p\{\text{rec } X.p/X\} \xrightarrow{\sigma} p'}{\text{rec } X.p \xrightarrow{\sigma} p'}$

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

$Sync(\alpha, \bar{\alpha}\sigma, \tau\sigma)$	$\dfrac{Sync(\sigma_1, \sigma_2, \sigma)}{Sync(\mu\sigma_1, \sigma_2, \mu\sigma)}$	$\dfrac{Sync(\sigma_1, \sigma_2, \sigma)}{Sync(\alpha\sigma_1, \bar{\alpha}\sigma_2, \tau\sigma)}$
$Sync(\alpha\sigma, \bar{\alpha}, \tau\sigma)$	$\dfrac{Sync(\sigma_1, \sigma_2, \sigma)}{Sync(\sigma_1, \mu\sigma_2, \mu\sigma)}$	

Table 2. Synchronization relation

Let us briefly comment the rules that are less standard. Rule (S-pref) allows for the creation of transitions labeled by non-empty sequences of actions. In order for $\underline{\mu}.q$ to make a move, it is necessary that q can perform a transition, i.e., the rest of the transaction. Hence, $\underline{\mu}.\mathbf{0}$ cannot perform any action. If a transition is labeled by $\sigma = \mu_1 \dots \mu_n$, then all the actions $\mu_1 \dots \mu_{n-1}$ are due to strong prefixes, while μ_n to a normal prefix. Rule (Com) has a side-condition on the possible synchronizability of sequences σ_1 and σ_2. $Sync(\sigma_1, \sigma_2, \sigma)$ holds if σ is obtained from an interleaving (possibly with synchronizations) of σ_1 and σ_2, where the last action of one of the two sequences is to be synchronized, hence reflecting that the subtransaction that ends first signals this fact (i.e., *commits*) to the other subtransaction. Relation $Sync$ is formally defined by the inductive rules reported in Table 2. Note that the two rules on the left of the Table, when $\sigma = \epsilon$, reduces to the standard synchronization rule of CCS. These two rules are the axioms defining relation $Sync$, hence the merging of the two sequences must end with a synchronization between the last action of one of the two sequences and one action (not necessarily the last one) of the other sequence. Rule (Res) requires that no action in σ can be a or $\bar{a}$. With $n(\sigma)$ we denote the set of all actions occurring in σ. In rule (Rel), the relabeling function f is applied to sequences homomorphically.

Remark 3.1. (Guarded recursion) We assume that any recursion variable in the body q of a recursion occurs inside a *normally prefixed* subprocess $\mu.q'$ of q. This will prevent infinitely branching sequential processes. For instance, consider the non legal process $p = \text{rec}\, X.(\underline{a}.X + b.\mathbf{0})$. According to the operational rules, p has infinitely many transitions leading to $\mathbf{0}$, each of the form $a^n b$, for $n = 0, 1, \ldots$.

Example 3.1. (Multi-party synchronization) Assume that we have three processes that want to synchronize. This can be easily expressed in A^{2}CCS. For instance, consider processes $p = \underline{a}.a.p'$ – that acts as the leader of the multiple synchronization – $q = \bar{a}.q'$ and $r = \bar{a}.r'$ and the whole system $P = (\nu a)((p \,|\, q) \,|\, r)$. It is easy to observe that $P \xrightarrow{\tau\tau} (\nu a)((p' \,|\, q') \,|\, r')$, so the three processes have synchronized in one single atomic transition. □

Two terms p and q are *interleaving bisimilar*, $p \sim q$, if there exists a bisimulation R such that $(p, q) \in R$. As expected, for bisimulation equivalence the sum operator is associative, commutative and $\mathbf{0}$ absorbent. As $(\nu a)(\nu b)p \sim (\nu b)(\nu a)p$, a simplification in the notation can be adopted, namely restriction on a set of names, e.g., $(\nu a, b)p$. On the other hand, the parallel operator is commutative and $\mathbf{0}$ absorbent, but not associative, as the following example shows.

Example 3.2. (Parallel operator is not associative) Consider the processes of Example 3.1. It is easy to see that $P' = (\nu a)(p \,|\, (q \,|\, r))$ cannot perform the multiway synchronization. As another, more complex example, take processes $p = \underline{a}.\underline{\bar{a}}.a.b.\mathbf{0}$, $q = \bar{a}.c.\mathbf{0}$ and $r = \underline{a}.\bar{a}.d.\mathbf{0}$. Consider $P = (\nu a)((p \,|\, q) \,|\, r)$ and $Q = (\nu a)(p \,|\, (q \,|\, r))$. It is easy to observe that $P \xrightarrow{\tau\tau\tau} (\nu a)((b.\mathbf{0} \,|\, c.\mathbf{0}) \,|\, d.\mathbf{0})$, while there is no Q' such that $Q \xrightarrow{\tau\tau\tau} Q'$. □

We assume the parallel composition operator to be left-associative by convention, so that, e.g., $p \,|\, q \,|\, r \,|\, s$ reads unambiguously as $(((p \,|\, q) \,|\, r) \,|\, s)$.

3.2. The dining philosophers

Example 3.3. This famous problem, defined by Dijkstra in [9], can be solved in A^{2}CCS, but not in CCS. Five philosophers sit at a round table, with a private plate and where each of the five forks is shared by two neighbor philosophers. Philosophers can think and eat; in order to eat, a philosopher has to acquire both forks that he shares with his/her neighbors, starting from the fork at his left and then the one at his right. All philosophers behave the same, so the problem is intrinsically symmetric. Clearly a naïve solution would cause deadlock exactly when all five philosophers take the fork at their left at the same time and are waiting for the fork at their right. A simple solution is to enforce atomicity on the acquisition of the two forks so that either both are taken or none. In order to have a small net model, we consider the case of two philosophers only. The two forks can be defined by the two CCS processes $fork_i$:

$$recX.\overline{up_i}.\overline{down_i}.X \text{ for } i = 0, 1$$

The two philosophers can be described by the processes $phil_i$:

$$recY.(think.Y + \underline{up_i}.up_{i+1(mod2)}.eat.\underline{down_i}.down_{i+1(mod2)}.Y) \text{ for } i = 0, 1$$

where the atomic sequence $up_i up_{i+1(mod2)}$ ensures the atomic acquisition of the two forks. For simplicity, we assume also that the release of the two forks is atomic, but this is not necessary for correctness.

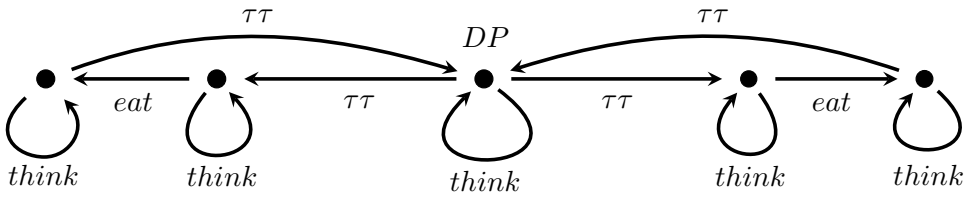

Figure 1. The finite (interleaving) lts for DP.

The whole system is

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

where $L = \{up_0, up_1, down_0, down_1\}$. Note that the operational semantics generates a finite-state lts for DP, depicted in Figure 1. The system for the case of 5 philosophers is

$$DP_5 \stackrel{def}{=} (\nu up_i, down_i, i = 1, \ldots, 4)(PHIL \,|\, fork_0 \,|\, fork_1 \,|\, fork_2 \,|\, fork_3 \,|\, fork_4)$$

where $PHIL$ is the composition of the five philosophers. □

3.3. Step Semantics

A^{2}CCS can be given a step semantics, i.e., a semantics where each transition is labeled by a (multi-)set of sequences that concurrent subprocesses can perform at the same time.

The step operational semantics for A^{2}CCS is given by the lts $(\mathcal{P}, \mathcal{B}, \longrightarrow_s)$, where the states are the processes in $\mathcal{P}$, $\mathcal{B} = \mathcal{M}_{fin}(Act^+)$ is the set of labels (ranged over by M), and $\longrightarrow_s \subseteq \mathcal{P} \times \mathcal{B} \times \mathcal{P}$ is the minimal transition relation generated by the rules listed in Table 3. Note that rule (S-pref$_s$) assumes that the transition in the premise is sequential. Note also that rule (Com$_s$) uses an additional auxiliary relation $MSync$, defined in Table 4.

Bisimulation on the step transition system of A^{2}CCS, called *step equivalence* and denoted $\sim_{step}$, is more discriminating than ordinary interleaving bisimulation $\sim$. For instance, $(a.\mathbf{0} \,|\, b.\mathbf{0}) \sim a.b.\mathbf{0} + b.a.\mathbf{0}$ but the two are not step bisimilar as only the former can perform a transition labeled by $\{a, b\}$.

Example 3.4. (Proving mutual exclusion) Let us consider the system DP of Example 3.3. A proof that DP acts correctly, i.e., it never allows both philosophers to eat at the same time, can be given by inspecting its step transition system (see Figure 2). As a matter of fact, the step $\{eat, eat\}$ is not present. □

4. Operational Net Semantics for A^{2}CCS

In this section we first apply the DDM technique [7, 21] for building a safe P/T system for the whole A^{2}CCS starting from a description of its places and of its net transitions. Then we describe how to construct the subnet $Net(p)$, with initial marking $dec(p)$, associated to a specific A^{2}CCS process p,

(Pref$_s$)	$\mu.p \xrightarrow{\{\mu\}}_s p$		(S-pref$_s$)	$\dfrac{p \xrightarrow{\{\sigma\}}_s p'}{\underline{\mu}.p \xrightarrow{\{\mu\sigma\}}_s p'}$
(Sum$_s$)	$\dfrac{p \xrightarrow{M}_s p'}{p+q \xrightarrow{M}_s p'}$		(Par$_s$)	$\dfrac{p \xrightarrow{M}_s p'}{p \mid q \xrightarrow{M}_s p' \mid q}$
(Com$_s$)	$\dfrac{p \xrightarrow{M_1}_s p' \qquad q \xrightarrow{M_2}_s q'}{p \mid q \xrightarrow{M} p' \mid q'}$	$MSync(M_1, M_2, M)$	(Rel$_s$)	$\dfrac{p \xrightarrow{M}_s p'}{p[f] \xrightarrow{f(M)}_s p'[f]}$
(Res$_s$)	$\dfrac{p \xrightarrow{M}_s p'}{(\nu a)p \xrightarrow{M}_s (\nu a)p'}$	$\forall \sigma \in M \;\; a, \bar{a} \notin n(\sigma)$	(Rec$_s$)	$\dfrac{p\{\text{rec } X.p/X\} \xrightarrow{M}_s p'}{\text{rec } X.p \xrightarrow{M}_s p'}$

Table 3. Step operational semantics (symmetric rules omitted)

$MSync(M_1, M_2, M_1 \oplus M_2)$	$\dfrac{Sync(\sigma_1, \sigma_2, \sigma) \;\; MSync(M_1, M_2, M)}{MSync(M_1 \oplus \{\sigma_1\}, M_2 \oplus \{\sigma_2\}, M \oplus \{\sigma\})}$

Table 4. Step synchronization relation

as there is no need to build the whole infinite net for the whole language if one is interested in the subnet reachable from the initial (finite) marking $dec(p)$. The case study of the dining philosophers is re-considered here and the resulting P/T net is constructed, as an illustration of the technique.

4.1. Building the P/T system for A^{2}CCS

We first need to define syntactic names for places. As a matter of fact, places are decorated sequential processes, where the decoration has to do with the way we handle the operators of parallel composition, restriction and relabeling.

4.1.1. Places and Markings

The infinite set S_{A^2CCS} of places, ranged over by s (possibly indexed), is generated by the following syntax:

$$s \;::=\; p \mid s|id \mid id|s \mid (\nu a)s \mid s[f]$$

where p is a sequential A^{2}CCS process term, place $s|id$ ($id|s$) are used to model that s is one of the left (right) subcomponents of a parallel term, while $(\nu a)s$ describes that s is one of the components under the scope of the restriction on a, and similarly for $s[f]$. These four operations on places can

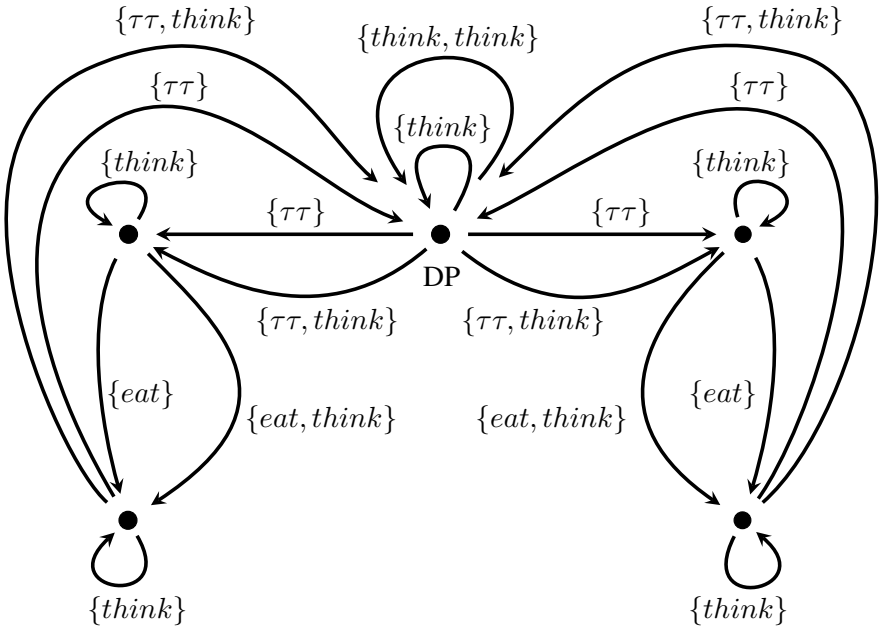

Figure 2. The step labeled transition system for DP.

be extended to markings in the obvious way, i.e., by (multiset) elementwise application. For instance, $id|(m_1 \oplus m_2) = id|(m_1) \oplus id|(m_2)$ and $(\nu a)(m_1 \oplus m_2) = (\nu a)(m_1) \oplus (\nu a)(m_2)$.

$dec(\mathbf{0}) = \{\mathbf{0}\}$	$dec(\mu.q) = \{\mu.q\}$	$dec(\underline{\mu}.q) = \{\underline{\mu}.q\}$
$dec(p + p') = \{p + p'\}$	$dec(q \mid q') = dec(q)\|id \ \oplus \ id\|dec(q')$	$dec(q[f]) = dec(q)[f]$
$dec((\nu a)q) = (\nu a)dec(q)$	$dec(\text{rec } X.q) = dec(q\{\text{rec } X.q/X\})$	

Table 5. Decomposition function

Function $dec : \mathcal{P} \to \wp_{fin}(S_{A^2CCS})$, which defines the decomposition of processes into markings, is reported in Table 5. Each sequential process p generates one single place with the same name. This is indeed the case for $\mathbf{0}$, $\mu.p$, $\underline{\mu}.p$ and for the guarded choice $p + p'$. Parallel composition is interpreted as disjoint set union, by means of the auxiliary decorations $|id$ and $id|$ imposed on places. Hence, in this context, the multiset union operator $\oplus$ actually reduces to set union $\cup$. The decomposition of a restricted process $(\nu a)q$ generates the set of places $dec(q)$ where each place is decorated with the restriction (νa) to record the syntactic position in the term. Similarly for $q[f]$. Finally, a recursive process is first unwound once and then decomposed.

It is possible to prove that the decomposition function dec is well-defined by induction on a suitably

defined notion of complexity of terms (following [21]). The complexity $\gamma(p)$ of closed process terms p is defined as follows:

$$\begin{array}{llll} \gamma(\mathbf{0}) & = 0 & \gamma(\mu.p) & = 0 \\ \gamma(\underline{\mu}.p) & = 0 & \gamma(p_1 + p_2) & = 0 \\ \gamma(p_1 \mid p_2) & = 1 + max\{\gamma(p_1), \gamma(p_2)\} & \gamma((\nu a)p) & = 1 + \gamma(p) \\ \gamma(p[f]) & = 1 + \gamma(p) & \gamma(\text{rec } X.q) & = 1 + \gamma(q\{\text{rec } X.q/X\}) \end{array}$$

Note that guardedness (even w.r.t. any kind of prefix) on recursion variables ensures that $\gamma(\text{rec } X.q) = n$ for some $n \in \mathbb{N}$. Guardedness is also essential to prove the following obvious facts.

Proposition 4.1.

1. Function dec is well-defined.
2. For any A^{2}CCS process p, $dec(p)$ is a finite set of places.

Proof:
By induction on the complexity of closed process terms. □

Let H, K, possibly indexed, range over $\wp_{fin}(S_{A^2CCS})$.

Definition 4.1. A set of places $H \subseteq S_{A^2CCS}$ is *complete* if there exists a process q such that $H = dec(q)$. □

The notion of a complete set of places is useful because we will prove that, starting from a complete set of places, the firing of a transition will produce another complete set of places.

Note that function dec is not injective because some recursive terms and their unwound version are mapped to the same marking, e.g., rec $X.a.X$ and a.rec $X.a.X$.

4.1.2. Net transitions

The P/T net for A^{2}CCS is the triple $N_{A^2CCS} = (S_{A^2CCS}, \mathcal{A}, T_{A^2CCS})$, where the infinite set T_{A^2CCS} of net transitions is the least set generated by the axiom schemata and inference rules reported in Table 6.

Axiom (pref) is trivial: if one token is present in place $\mu.q$, then the transition labeled μ produces one token in each place of $dec(q)$. Rule (sum) is also trivial, due to the guardedness of the choice operator: as both p and p' are sequential, the decomposition of p is the singleton set $\{p\}$, which is used in the premise. Rule (par), and its omitted symmetrical rule, are interesting as they explain the role of operators $|id$ and $id|$: if a set of places H can move, then the same can do the set of places $H|id$ or $id|H$, hence the move can be done in any parallel context. Rule (res) ensures that a transition σ can be performed by $(\nu a)H$, provided that H can do it and no actions in σ is either a or $\bar{a}$. Rule (s-pref) states that, if a token is present in place $\underline{\mu}.q$ and if a subset H of places of $dec(q)$ can move with σ, then a transition labeled $\mu\sigma$ is derivable that puts one token not only in H', but also in the idle places K because, intuitively, they have been activated by the execution of the initial μ. Rule (com) explains how a synchronization between two parallel sets of places can take place; note that we require that H is on the left and K is on the right, meaning that they originate from the left, resp. right, part of a parallel term.

$$\text{(pref)}\quad \{\mu.q\} \xrightarrow{\mu} dec(q) \qquad \text{(sum)}\quad \frac{\{p\} \xrightarrow{\sigma} H}{\{p+p'\} \xrightarrow{\sigma} H} \qquad \text{(rel)}\quad \frac{H \xrightarrow{\sigma} H'}{H[f] \xrightarrow{f(\sigma)} H'[f]}$$

$$\text{(par)}\quad \frac{H \xrightarrow{\sigma} H'}{H|id \xrightarrow{\sigma} H'|id} \qquad \text{(res)}\quad \frac{H \xrightarrow{\sigma} H'}{(\nu a)H \xrightarrow{\sigma} (\nu a)H'}\ a, \bar{a} \notin n(\sigma)$$

$$\text{(s-pref)}\quad \frac{H \xrightarrow{\sigma} H'}{\{\underline{\mu}.q\} \xrightarrow{\mu\sigma} H' \cup K}\ H \cup K = dec(q)$$

$$\text{(com)}\quad \frac{H \xrightarrow{\sigma_1} H'\ K \xrightarrow{\sigma_2} K'}{H|id \cup id|K \xrightarrow{\sigma} H'|id \cup id|K'}\ Sync(\sigma_1, \sigma_2, \sigma)$$

Table 6. Rules for net transitions (symmetric rules for (sum) and (par) omitted).

Proposition 4.2. Let $H \oplus K$ be a complete set of places. Let $H \xrightarrow{\sigma} H'$ be a transition in T_{A^2CCS}. Then $H' \oplus K$ is a complete set of places.

Proof:
The proof is by induction on the proof of transition $H \xrightarrow{\sigma} H'$.

Case $\{\mu.q\}$ In this case, the thesis holds because the only derivable transition is $\{\mu.q\} \xrightarrow{\mu} dec(q)$ that reaches a complete set of place.

Case $\{\underline{\mu}.q\}$ In this case, we know that $H \oplus K = dec(q)$. By inductive hypothesis, we have that transition $H \xrightarrow{\sigma} H'$ is such that $H' \oplus K$ is complete. Hence the thesis follows.

Case $\{p + p'\}$ In this case, by inductive hypothesis, we know that in transition $\{p\} \xrightarrow{\sigma} H$ set H is complete. Hence the thesis follows.

Case $H|id$ In this case, let $H|id \ \oplus \ K|id \ \oplus \ id|dec(q)$ be a complete set. Hence $H \oplus K$ is also complete. By inductive hypothesis, we know that $H \xrightarrow{\sigma} H'$ ensures that $H' \oplus K$ is complete. Hence the thesis follows because $(H' \oplus K)|id \oplus id|dec(q)$ is a complete set of places.

Case $H|id \oplus id|K$ In this case, let $H|id \oplus H''|id \oplus id|K \oplus id|K''$ be a complete set. Hence $H \oplus H''$ and $K \oplus K''$ are also complete. By inductive hypothesis, we know that $H \xrightarrow{\sigma_1} H'$ and $K \xrightarrow{\sigma_2} K'$ ensure that $H' \oplus H''$ and $K' \oplus K''$ are complete. Hence the thesis follows because $(H' \oplus H'')|id \oplus id|(K' \oplus K'')$ is a complete set of places.

Case $(\nu a)H$ In this case, let $(\nu a)H \ \oplus \ (\nu a)K$ be complete. Hence, $H \oplus K$ is also complete. By inductive hypothesis, we know that if $H \xrightarrow{\sigma} H'$ then $H' \oplus K$ is complete. Then the thesis follows as $(\nu a)(H' \oplus K)$ is complete.

Case $H[f]$ In this case, let $H[f] \ \oplus \ K[f]$ be complete. Hence, $H \oplus K$ is also complete. By inductive hypothesis, we know that if $H \xrightarrow{\sigma} H'$ then $H' \oplus K$ is complete. Then the thesis follows as $(H' \oplus K)[f]$ is complete. □

This result is useful when proving that there is a strong correspondence between the interleaving

marking graph and the original labeled transition system, as the states of the marking graph are essentially processes (via dec).

4.1.3. The P/T subnet associated to a process

Given a process p, the P/T system associated to p, denoted with $Net(p)$, is the subnet of N_{A^2CCS} reachable from the initial marking $dec(p)$.

Definition 4.2. Let p be a process. The P/T system associated to p is $Net(p) = (S_p, A_p, T_p, m_0)$, where $m_0 = dec(p)$ and

$$\begin{array}{rcl} S_p & = & \{s \in S_{A^2CCS} \mid \exists m \in [m_0\rangle\ (m(s) > 0)\} \\ T_p & = & \{t \in T_{A^2CCS} \mid \exists m \in [m_0\rangle\ (m[t\rangle)\} \\ A_p & = & \{a \in \mathcal{A} \mid \exists t \in T_p, a = l(t)\} \end{array}$$

□

The definition above suggests a way of generating $Net(p)$ by means of an algorithm in least-fixpoint style. Start by the initial marking $dec(p)$, which is a finite multiset, and then try to apply the rules in Table 6 in order to produce the set of transitions enabled at $dec(p)$ as well as the set of markings reachable in one step; this will also produce possible new places to be added to the current set of places. Then repeat (from the markings reached so far) until no new places are added and no new transitions are derivable; hence, this algorithm ends only for finite nets. In order to formalize this algorithm, we need some auxiliary notations. With $m \models t$, or $m \models {}^\bullet t \xrightarrow{l(t)} t^\bullet$, we denote the fact that transition $t = {}^\bullet t \xrightarrow{l(t)} t^\bullet$ is derivable by the rules of Table 6 and that marking m enables the transition, i.e., ${}^\bullet t \subseteq m$. Formally, we can define a sequence of structures $N(q)_i = (S_i, A_i, T_i, M_i)$, for $i = 0, 1, \ldots$, as follows:

- $N(q)_0 = (S_0, A_0, T_0, M_0)$, where $S_0 = dom(dec(q))$, $A_0 = \emptyset$, $T_0 = \emptyset$ and $M_0 = \{dec(q)\}$.
- $N(q)_{i+1} = (S_{i+1}, A_{i+1}, T_{i+1}, M_{i+1})$, where
  - $S_{i+1} = S_i \cup \{t^\bullet \mid \exists m \in M_i . m \models t\}$
  - $A_{i+1} = A_i \cup \{l(t) \mid \exists m \in M_i . m \models t\}$
  - $T_{i+1} = T_i \cup \{t \mid \exists m \in M_i . m \models t\}$
  - $M_{i+1} = M_i \cup \{m' \mid \exists m \in M_i, t \in T_{i+1} . m[t\rangle m'\}$.

We may define an obvious ordering $\sqsubseteq$ on such structures as follows: $N(q)_i \sqsubseteq N(q)_{i+1}$ if $S_i \subseteq S_{i+1}$ and $T_i \subseteq T_{i+1}$. This is indeed the case, hence the family $\{N(q)_i\}_i$ is a chain w.r.t. $\sqsubseteq$. Note that any component of $N(q)_i$ is finite, for any $i \in \mathbb{N}$. This chain may have a finite top element if, for some i, $S_i = S_{i+1}$ and $T_i = T_{i+1}$. In such a case, we conclude that $Net(q) = (S_i, \mathcal{A}_i, T_i, m_0)$ and $Net(q)$ is a finite net.

Example 4.1. As an example, let us consider the process

$$p = \underline{a}.(recX.b.b.X | recY.c.c.Y)$$

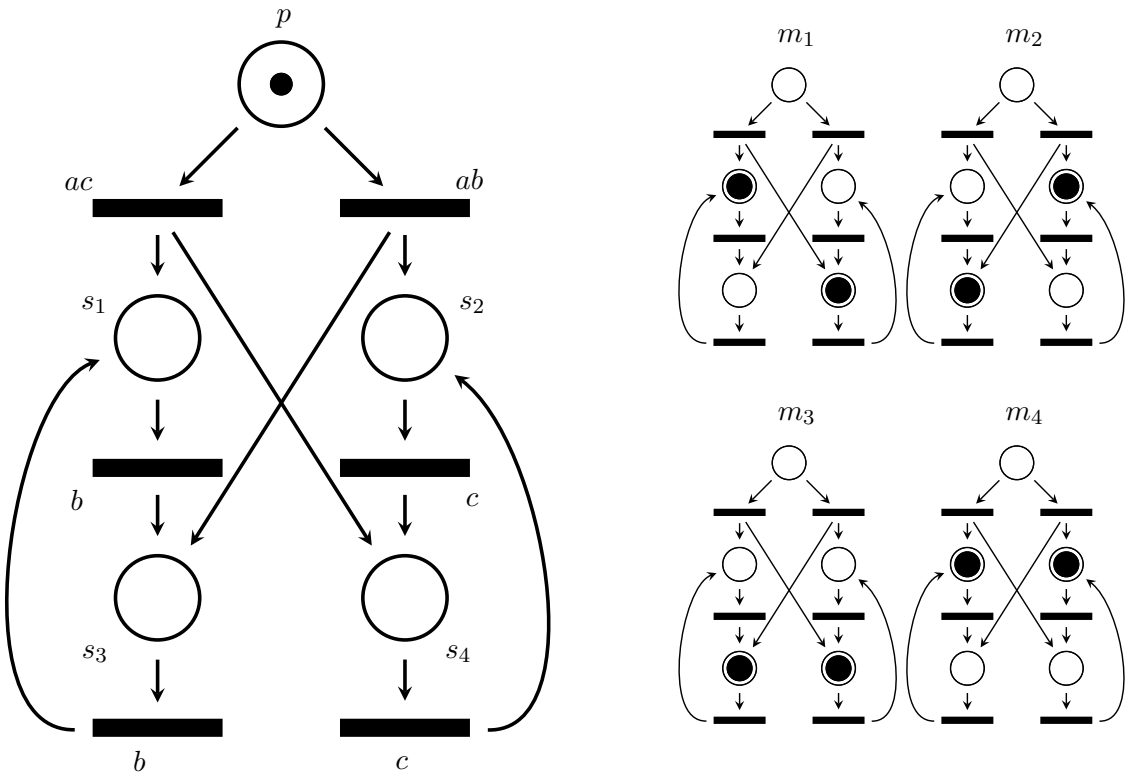

Figure 3. The P/T net $Net(p)$ associated with the process p of Example 4.1: on the left, with the initial marking m_0, on the right with any further reachable marking.

In order to build the associated P/T net $Net(p)$, reported in Figure 3, we apply the function $dec(\cdot)$ to p to get the initial set of places S_0 as well as the initial marking m_0, which constitutes the unique element of M_0. We obtain that $m_0 = dec(p) = \{p\} = S_0$, since p is sequential and is the only process we are considering. The strong prefix of the action a implies that p can produce two possible initial transitions, one for each of the two parallel processes guarded by a, so $T_1 = \{t_1, t_2\}$ where $t_1 = \{p\} \xrightarrow{ac} \{s_1, s_4\}$ (because in the premise of rule (s-pref) we have $t = s_2 \xrightarrow{c} s_4$), and $t_2 = \{p\} \xrightarrow{ab} \{s_2, s_3\}$ (because in the premise of rule (s-pref) we have $t' = s_1 \xrightarrow{b} s_3$), with

$$s_1 = b.b.recX.b.b.X|id \qquad s_2 = id|c.c.recY.c.c.Y$$
$$s_3 = b.recX.b.b.X|id \qquad s_4 = id|c.recY.c.c.Y$$

Consequently, $S_1 = S_0 \cup \{s_1, s_2, s_3, s_4\}$, and $A_1 = \{ab, ac\}$. Since in M_0 there is only one marking with one place and one token, we obtain exactly two further markings $m_1 = \{s_1, s_4\}, m_2 = \{s_2, s_3\}$ to be added to $M_1 = M_0 \cup \{m_1, m_2\}$. Now, for each of the two new markings we have to consider the available transitions. It turns out that each of them enables two transitions, one for each parallel process:

- m_1 enables $t_3 = \{s_1\} \xrightarrow{b} \{s_3\}$, and $t_4 = \{s_4\} \xrightarrow{c} \{s_2\}$;
- m_2 enables $t_5 = \{s_2\} \xrightarrow{c} \{s_4\}$, and $t_6 = \{s_3\} \xrightarrow{b} \{s_1\}$.

So we have four new transitions in $T_2 = T_1 \cup \{t_3, t_4, t_5, t_6\}$ with two new labels in $A_2 = A_1 \cup \{b, c\}$ and two new markings in $M_2 = M_1 \cup \{m_3, m_4\}$, each of them obtained twice:

- $m_3 = \{s_3, s_4\}$ obtained by applying t_3 to m_1 or t_5 to m_2;
- $m_4 = \{s_1, s_2\}$ obtained by applying t_4 to m_1 or t_6 to m_2.

The set of places S_2 is instead unchanged with respect to S_1, because no new places have been added. Finally, when we consider the possible transitions enabled by markings m_3 and m_4 we realize that all of them have been already added to M_2. In fact:

- m_3 enables t_4 and t_6, the first leading to marking m_2, the second to m_1;
- m_4 enables t_3 and t_5, the first leading to marking m_2, the second to m_1.

Since $M_3 = M_2$, also $T_3 = T_2$, $S_3 = S_2$ and $A_3 = A_2$, so that the generation of the P/T net $Net(p)$ is complete. □

Remark 4.1. According to the rules in Table 6, in order to find the net transitions originating from any given state $dec(p)$, we need to compute in advance the transitions generated by the subprocesses in p which are guarded by a strong prefix. For example, if $p = \underline{\mu}.q$, then rule (s-pref) of Table 6 states that we first need to compute $dec(q)$ (and corresponding transitions) in order to find any transition originating from $dec(p)$. In Example 4.1, $p = \underline{\mu}.q = \underline{a}.(recX.b.b.X|recY.c.c.Y)$, so that we need to compute the transitions of $dec(q) = \{b.b.recX.b.b.X|id, id|c.c.recY.c.c.Y\}$ to find the transitions of $dec(p)$. This correspond exactly to the look-ahead operation that is performed in rule (S-pref) of Table 1, where the transactions of q are computed in order to find each transaction of p: in both cases, the maximal number of look-ahead operations corresponds to the maximal length of the sequences of strongly prefixed actions originating from p. Thanks to the assumption of guarded recursion, such length is always finite, and the number of different transitions of any p is finite as well. This ensures that each step of the least-fixpoint style algorithm for calculating $Net(p)$ can be performed in a finite amount of time, independently of the (in)finiteness of the network associated with any subprocess q or of the (in)finiteness of $Net(p)$ itself. Since the generation of $Net(p)$ implies the enumeration of all the transitions originating from each state reachable from $dec(p)$, a clever implementation of the cited algorithm would obviously enumerate once for all the full set of transitions of every subprocess q before calculating those of $\underline{\mu}.q$.

Proposition 4.3. $Net(p)$ is a safe P/T net for any process $p \in \mathcal{P}$.

Proof:
By Proposition 4.1, $dec(p)$ is a complete set of places, for any $p \in \mathcal{P}$. The fact that for any marking m reachable from $dec(p)$ we have that $m(s) \leq 1$ for any $s \in S_p$, derives by (repeated applications of) Proposition 4.2, which states that from a complete set of places we can only reach a complete set of places. □

4.2. Dining Philosophers

Let us consider the system DP of Example 3.3. Marking $dec(DP)$ consists of the four places

$$s_1 = (\nu L)(((phil_0|id)|id)|id) \qquad s_2 = (\nu L)(((id|phil_1)|id)|id)$$
$$s_3 = (\nu L)((id|fork_0)|id) \qquad s_4 = (\nu L)(id|fork_1)$$

where $L = \{up_0, up_1, down_0, down_1\}$, $fork_i = \overline{up_i}.\overline{down_i}.(recX.\overline{up_i}.\overline{down_i}.X)$ and

$$phil_i = (think.Y + \underline{up_i}.up_{i+1}.eat.\underline{down_i}.down_{i+1}.Y) \\ \{recY(think.Y + \underline{up_i}.up_{i+1}.eat.\underline{down_i}.down_{i+1}.Y)/Y\}$$

Each place corresponds to a sequential subsystem of DP, suitably decorated with its context information. Initially, the two philosophers can think on their own (possibly in parallel):

$$s_1 \xrightarrow{think} s_1 \qquad s_2 \xrightarrow{think} s_2$$

or can compete for the acquisition of the two forks:

$$s_1 \oplus s_3 \oplus s_4 \xrightarrow{\tau\tau} s'_1 \oplus s'_3 \oplus s'_4 \qquad s_2 \oplus s_3 \oplus s_4 \xrightarrow{\tau\tau} s'_2 \oplus s'_3 \oplus s'_4$$

where

$$s'_1 = (\nu L)(((phil'_0|id)|id)|id) \qquad s'_2 = (\nu L)(((id|phil'_1)|id)|id)$$
$$s'_3 = (\nu L)((id|fork'_0)|id) \qquad s'_4 = (\nu L)(id|fork'_1)$$

with, for $i = 0, 1$, $fork'_i = \overline{down_i}.(recX.\overline{up_i}.\overline{down_i}.X)$ and

$$phil'_i = eat.\underline{down_i}.down_{i+1}.(recY(think.Y + \underline{up_i}.up_{i+1}.eat.\underline{down_i}.down_{i+1}.Y)$$

Now two further alternative transitions are derivable, namely:

$$s'_1 \xrightarrow{eat} s''_1 \qquad s'_2 \xrightarrow{eat} s''_2$$

where

$$s''_1 = (\nu L)(((phil''_0|id)|id)|id) \qquad s''_2 = (\nu L)(((id|phil''_1)|id)|id)$$

with, for $i = 0, 1$,

$$phil''_i = \underline{down_i}.down_{i+1}.(recY(think.Y + \underline{up_i}.up_{i+1}.eat.\underline{down_i}.down_{i+1}.Y)$$

Finally,

$$s''_1 \oplus s'_3 \oplus s'_4 \xrightarrow{\tau\tau} s_1 \oplus s_3 \oplus s_4 \qquad s''_2 \oplus s'_3 \oplus s'_4 \xrightarrow{\tau\tau} s_2 \oplus s_3 \oplus s_4$$

and we are back to the initial marking $dec(DP)$. The resulting $Net(DP)$ is reported in Figure 4. Note that the two philosophers can never eat at the same time, i.e., in no reachable marking m we have that $m(s'_1) = 1 = m(s'_2)$.

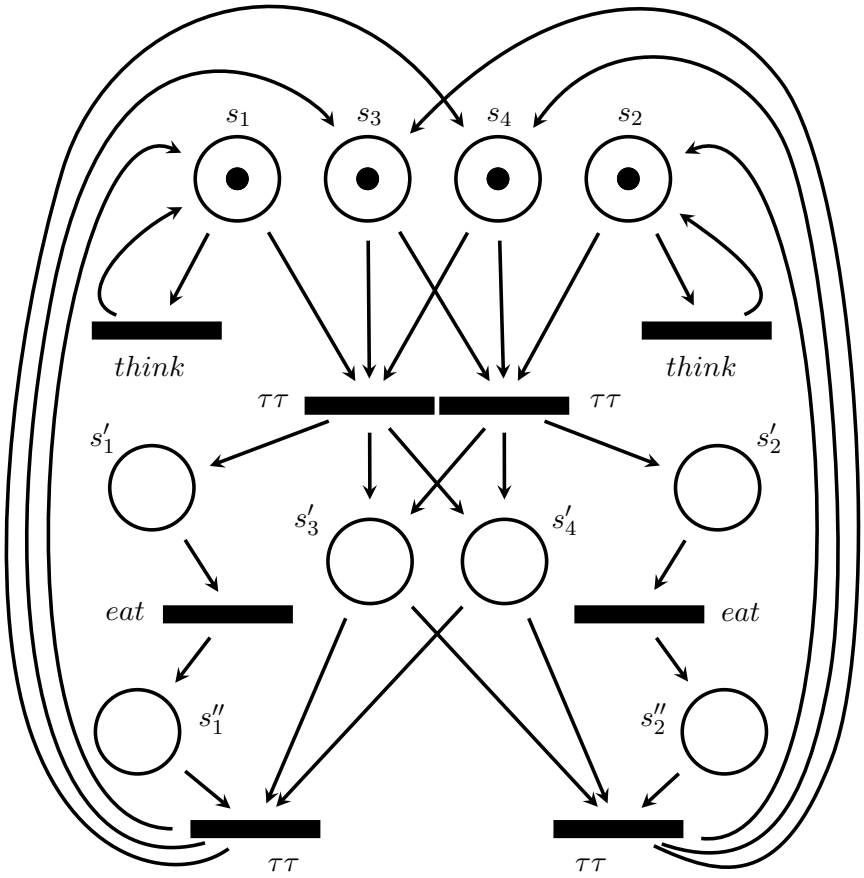

Figure 4. The net for the two dining philosophers DP.

5. Properties of the DDM semantics

Here a soundness result is provided, namely that the interleaving marking graph associated to $dec(p)$, $IMG(Net(p))$, for any A^{2}CCS term p tightly corresponds to its interleaving transition system.

Proposition 5.1. For any process $p \in \mathcal{P}$, if $p \xrightarrow{\sigma} p'$ then there exists a transition $t \in T_p$ such that $dec(p)[t\rangle dec(p')$ with $l(t) = \sigma$.

Proof:
The proof is by induction on the proof of transition $p \xrightarrow{\sigma} p'$. For details, see the appendix. □

Proposition 5.2. For any process $p \in \mathcal{P}$, if $dec(p)[t\rangle K$ for some $t \in T_p$ such that $l(t) = \sigma$ then there exists p' such that $p \xrightarrow{\sigma} p'$ and $K = dec(p')$.

Proof:
By induction on (the definition of) $dec(p)$ and then by induction on the proof of the transition $t \in T_p$. For details, see the appendix. □

Theorem 5.1. For any process $p \in \mathcal{P}$, $p \sim dec(p)$, i.e., $p \sim Net(p)$.

Proof:
Take relation $R = \{(p, dec(p)) \quad | \quad p \in \mathcal{P}\}$. Thanks to Proposition 5.1 and 5.2, it is immediate to see that R is a bisimulation. □

The same soundness result is proved for step semantics. Also the proofs are very similar, but need a bit more attention on the way steps are synchronized.

Proposition 5.3. For any process $p \in \mathcal{P}$, if $p \xrightarrow{M}_s p'$ then there exists a step G such that $dec(p)[G\rangle dec(p')$ with $l(G) = M$.

Proof:
The proof is by induction on the proof of transition $p \xrightarrow{M}_s p'$. For details, see the appendix. □

Proposition 5.4. For any $p \in \mathcal{P}$, if there exists a step G such that $dec(p)[G\rangle K$, with $l(G) = M$, then there exists p' such that $p \xrightarrow{M}_s p'$ and $K = dec(p')$.

Proof:
By induction on (the definition of) $dec(p)$ and then by induction on the proof of the transitions in G. For details, see the appendix. □

An immediate consequence of the two propositions above is the following.

Theorem 5.2. For any process $p \in \mathcal{P}$, $p \sim_{step} dec(p)$, i.e., $p \sim_{step} Net(p)$.

Finally, we prove that the subnet $Net(p)$ associated to a specific A^{2}CCS process p is finite if and only if the interleaving operational semantics generates a finite-state transition system. Hence, a sufficient condition for finiteness is that parallel composition, restriction and relabeling do not occur inside recursion.

Proposition 5.5. For any process $p \in \mathcal{P}$, $Net(p)$ is finite if and only if the transition system reachable from p is finite-state.

Proof:
By Proposition 5.1, the cardinality of the set $[dec(p)\rangle$ is less than (or equal to) the cardinality of set $Reach(p)$. Therefore, if $Reach(p)$ is a finite set, then $IMG(Net(p))$ is finite-state, hence $Net(p)$ has finitely many places and transitions, by construction. Conversely, if $Reach(p)$ is an infinite set, then p must be a term where occurrences of the operators of parallel composition, restriction or relabeling are inside the scope of a recursion binder. In either case, $Net(p)$ is an infinite net by construction. □

6. Conclusion

This paper can be seen as another proof that the DDM technique is so general that it can be used for any kind of process algebra with an associated operational semantics in Plotkin' SOS style [24, 25]. The challenges raised by A^{2}CCS are the operation of strong prefixing and the more elaborated CCS-based transactional synchronization. Observe that the DDM technique applies also to a non-associative parallel composition, while most known techniques would require associativity.

The resulting Petri net for A^{2}CCS offers two main advantages:

- it is now possible to investigate non-interleaving semantics for A^{2}CCS, based on notions of causality and conflict that are primitive in Petri nets while completely absent in ordinary transition systems;
- it is possible to apply well-known Petri nets analysis techniques also to A^{2}CCS processes (e.g., net invariants).

On the other hand, also the operator of strong prefixing is important for Petri nets. In a recent paper [17], we present the Multi-CCS process algebra as a variant of A^{2}CCS with an associative parallel composition operator and with a different synchronization relation. This calculus is given a Petri net semantics following the BG approach and we prove that the so-called finite-net processes (i.e., those processes for which restriction cannot occur inside recursion) are as expressive as finite P/T Petri nets: the semantics of a finite-net process p is a finite P/T net $Net(p)$ and, on the other hand, for any finite P/T system $N(m_0)$ there exists a finite-net process p_N such that $Net(p_N)$ is isomorphic to $N(m_0)$. This result provides a clear bridge between the areas of Petri nets and process algebra: the development of a similar correspondence between finite safe P/T Petri nets and a proper, weakened variant of A^{2}CCS is being considered for future work.

In addition, since the DDM technique cannot be applied directly to process algebras like Multi-CCS because of its SOS semantics defined modulo structural congruence, a generalization of the DDM approach is being considered in order to allow its application even in the case of such a kind of semantics.

From an expressiveness point of view, it may seem unnecessary to use the transactional approach of A^{2}CCS for expressing multiway synchronization. Indeed, CSP [18, 27] offers multiway synchronization in a natural way. For instance, a formalization of the dining two-philosophers problem with a CSP-like parallel composition would be as follows. Each of the two forks is $fork_i$:

$$recX.\overline{up}_i.\overline{down}_i.X + \overline{up}_{i+1(mod2)}.\overline{down}_{i+1(mod2)}.X \text{ for } i = 0, 1$$

where now up_i means that the fork is to be taken by the i-th philosopher. The two philosophers can be described by the processes $phil_i$:

$$recY.(think.Y + up_i.eat.down_i.Y) \text{ for } i = 0, 1$$

The whole system is

$$DP_{CSP} \stackrel{def}{=} \tau_L((phil_0 \,||_\emptyset phil_1) \,||_L (fork_0 \,||_L fork_1))$$

where $L = \{up_0, up_1, down_0, down_1\}$, τ_L is the hiding operator on the action in set L, and $||_L$ represents the CSP parallel operator, where synchronization is forced on actions in L. The only difference between the A^{2}CCS specification of Example 3.3 and the CSP specification above is that the former is built around the philosopher who has two acquire the two different forks, while the latter is built around the forks that offer their service to the two neighbour philosophers.

However, we feel that the transactional approach of A^{2}CCS is, in general, more flexible. For instance, consider the following example.

Example 6.1. (Cigarette smokers' problem) This famous problem, proposed by Patil [22], is described in [23] as follows. The problem consists of four processes: an agent and three smokers. Each smoker continuously makes a cigarette and smokes it. But to make a cigarette, three ingredients are needed: tobacco, paper and matches. One of the smokers has paper, another tobacco and the third has matches. The agent has an infinite supply of all three. The agent places two of the ingredients on the table. The smokers who has the remaining ingredient can then make and smoke a cigarette, signalling the agent upon completion. The agent then puts out another two of the three ingredients and the cycle repeats. The problem is to define the code for the smoker processes to determine which of the three processes should proceed (the agent is fixed and cannot be changed), avoiding deadlock. Clearly, a solution to the problem is to have an atomic transaction in which the agent synchronize only with the smoker that has the missing ingredient.

$$
\begin{array}{lcl}
Agent & \stackrel{def}{=} & \text{rec}\, X.\tau.(\underline{\overline{tob}}.\overline{mat}.end.X) + \tau.(\underline{\overline{mat}}.\overline{pap}.end.X) + \tau.(\underline{\overline{pap}}.\overline{tob}.end.X) \\
S_{pap} & \stackrel{def}{=} & recX.\underline{tob}.mat.smoke.\overline{end}.X \\
S_{tob} & \stackrel{def}{=} & recX.\underline{mat}.pap.smoke.\overline{end}.X \\
S_{mat} & \stackrel{def}{=} & recX.\underline{pap}.tob.smoke.\overline{end}.X \\
Patil & \stackrel{def}{=} & (\nu\, tob, pap, mat, end)(Agent \mid S_{tob} \mid S_{mat} \mid S_{pap})
\end{array}
$$

This solution, based on transactional synchronization, determines a finite-state labeled transition system and a finite P/T net. Apparently, this transactional solution cannot be provided in CSP. If we want to solve this problem with multiway synchronization, we should treat the ingredients as resources that are consumed and then reproduced. An A^{2}CCS representation of the problem in this multiway style is as follows:

$$
\begin{array}{lcl}
Agent' & \stackrel{def}{=} & \text{rec}\, X.\tau.(\overline{tob}.\mathbf{0} \mid \overline{mat}.\mathbf{0} \mid end.X) + \tau.(\overline{mat}.\mathbf{0} \mid \overline{pap}.\mathbf{0} \mid end.X) \\
 & & \qquad + \tau.(\overline{pap}.\mathbf{0} \mid \overline{tob}.\mathbf{0} \mid end.X) \\
Patil' & \stackrel{def}{=} & (\nu\, tob, pap, mat, end)(Agent' \mid S_{tob} \mid S_{mat} \mid S_{pap})
\end{array}
$$

where a multi-party synchronization takes place among a smoker and the two needed ingredients. This solution, however, generates an infinite-state transition system as well as an infinite P/T net. An attempt to formulate in CSP this multiway style specification is a bit unnatural, as we are forced to loose identity of each single ingredient: as multiway synchronization happens on a common name, we should use the same name for, e.g., tobacco and matches in the first branch of $Agent'$, as well as a different common name for matches and paper in the second branch, etc.. □

From a modeling perspective, A^{2}CCS transactional approach allows the expression of multiway synchronization in a fully compositional way with respect to the parallel operator, that is without need to specify any information about the synchronization capabilities of the processes on (or externally to) the

parallel operator itself. As a consequence, the parallel composition of sequential A^2CCS processes may generate an exponential number of different transactions as a function of the number and length of the transactions available to each of the processes. This constitutes a desirable feature from a modelling perspective, since A^2CCS may provide an extremely concise and clear formalization of systems despite of their complex behavior. However, the exponential number of transactions may quickly render unfeasible the analysis of seemingly small models.

Future work shall be also devoted to an inductive definition (i.e., denotational in style) of the net semantics: it seems that a definition of net operators corresponding to those of A^2CCS can be done without too much efforts in the style of [21].

Future research should be also devoted to address the important problem of finding a general theory of SOS for Petri nets, as a generalization of the DDM technique. The theory of transition systems specification via SOS has a large body of results (e.g., formats that ensure that bisimulation is a congruence) that could conceivably be approached also for a to-be-developed theory of Petri nets specification via SOS. For instance, it could be interesting to study which formats the inference rules should have to ensure that a Petri net is safe, or of a special class (elementary, rather than Place/Transitions or else), or such that history preserving bisimulation is a congruence.

References

[1] M. Bernardo, N. Busi, R. Gorrieri, "A Distributed Semantics for EMPA Based on Stochastic Contextual Nets", *The Comp. Jour.* 38(7): 492-509, 1995.

[2] E. Best, R. Devillers, M. Koutny, "The Box Algebra = Petri Nets + Process Expressions.", *Inf. Comput.*, 178(1):44-100, 2002.

[3] A. Brogi, R. Gorrieri:, "A Distributed, Net Oriented Semantics for Delta Prolog", in *Theory and Practice of Software Development - Colloquium on Trees in Algebra and Programming* (TAPSOFT 89), LNCS 351, 162-177, Springer-Verlag, 1989.

[4] N. Busi, R. Gorrieri, "Distributed Conflicts in Communicating Systems", in *Object-Based Models and Languages for Concurrent Systems*, LNCS 924, 49-65, Springer-Verlag, 1994.

[5] N. Busi, R. Gorrieri, "A Petri Net Semantics for π- calculus", In Proc. *Concur'95*, LNCS 962, Springer-Verlag, 145-159, 1995.

[6] N. Busi, R. Gorrieri, "Distributed semantics for the π-calculus based on Petri nets with inhibitor arcs", *Journal of Logic and Alg. Prog.* 78(3):138-162, 2009.

[7] P. Degano, R. De Nicola, U. Montanari, "A Distributed Operational Semantics for CCS based on C/E Systems", *Acta Informatica* 26(1-2):59-91, 1988.

[8] P. Degano, R. Gorrieri, S. Marchetti, "An Exercise in Concurrency: A CSP Process as a Condition/Event System", *Advances in Petri Nets 1988*, LNCS 340, 85-105, Springer-Verlag, 1988.

[9] E.W.Dijkstra, "Hierarchical ordering of sequential processes", *Acta Informatica* 1(2):115-138, 1971.

[10] R. van Glabbeek, F. Vaandrager, "Petri Net Models for Algebraic Theories of Concurrency", in Proc. *PARLE'87*, LNCS 259, 224-242, Springer-Verlag, 1987.

[11] U. Goltz, "On Representing CCS Programs by Finite Petri Nets", in Proc. *MFCS'88*, LNCS 324, 339-350, Springer-Verlag, 1988.

[12] U. Goltz, "CCS and Petri Nets", LNCS 469, 334-357, Springer-Verlag, 1990.

[13] R. Gorrieri, "A Hierarchy of System Descriptions via Atomic Linear Refinement", *Fundamenta Informaticae*, 16 (3/4):289-336, 1992.

[14] R. Gorrieri, U. Montanari, "SCONE: A Simple Calculus of Nets", in Proc. *CONCUR'90*, LNCS 458, 2-30, Springer-Verlag, 1990.

[15] R. Gorrieri, U. Montanari, "Towards Hierarchical Specification of Systems: A Proof System for Strong Prefixing", *Int. Journal of Foundations of Computer Science*, 1(3):277-293, 1990.

[16] R. Gorrieri, S. Marchetti, U. Montanari, "A^2CCS: Atomic Actions for CCS", *Theoretical Computer Science* 72(2-3): 203-223, 1990.

[17] R. Gorrieri, C. Versari, "A Process Calculus for expressing finite Place/Transition Petri Nets", in Procs. Express'10, EPTCS, 2010.

[18] C.A.R. Hoare, *Communicating Sequential Processes* Prentice-Hall International Series in Computer Science, 1985.

[19] R. Milner, *Communication and Concurrency*, Prentice-Hall, 1989.

[20] R. Milner. *Communicating and mobile systems: the π-calculus*, Cambridge University Press, 1999.

[21] E. R. Olderog, *Nets, Terms and Formulas*, Cambridge Tracts in Theoretical Computer Science 23, Cambridge University Press, 1991.

[22] S. Patil "Limitations and Capabilities of Dijkstra's Semaphore Primitives for Coordination Among Processes", Computation Structures Group Memo 57, Project MAC, MIT, 1971.

[23] J.L. Peterson, *Petri Net Theory and the Modeling of Systems*, Prentice-Hall, 1981.

[24] G. D. Plotkin "A structural approach to operational semantics", Technical Report DAIMI FN-19, Aarhus University (1981)

[25] G. D. Plotkin "A structural approach to operational semantics", *J. Logic and. Algebraic Programming* 60-61: 17-139 (2004).

[26] W. Reisig, *Petri Nets: An Introduction*, EATCS Monographs on Theoretical Computer Science, Springer-Verlag, 1985.

[27] A.W. Roscoe, *The Theory and Practice of Concurrency* Prentice-Hall, 1998.

A. Proofs of Section 5

Proposition 5 For any process $p \in \mathcal{P}$, if $p \xrightarrow{\sigma} p'$ then there exists a transition t such that $dec(p)[t\rangle K$ with $l(t) = \sigma$ and $K = dec(p')$.

Proof:
The proof is by induction on the proof of transition $p \xrightarrow{\sigma} p'$.

If axiom (Pref) has been used, then $p = \mu.q$, $\sigma = \mu$ and $p' = q$. The thesis follows by observing that axiom (pref) ensures that $dec(p) = \{\mu.q\} \xrightarrow{\mu} dec(q) = dec(p')$.

If rule (S-pref) is the last rule used to derive $p \xrightarrow{\sigma} p'$, then $p = \underline{\mu}.q$ and $\sigma = \mu\sigma'$. The inductive hypothesis on the premise of rule (S-pref) ensures that $q \xrightarrow{\sigma'} p'$ implies that there exists a transition $t = (H, \sigma', H')$ such that $dec(q)[t\rangle dec(p')$ with $l(t) = \sigma'$. Hence the thesis follows by rule (s-pref): $t = (H, \sigma', H')$ implies $dec(p) = \{\underline{\mu}.q\} \xrightarrow{\mu\sigma'} dec(p')$.

If rule (Sum) is the last rule applied to derive transition $p \xrightarrow{\sigma} p'$, then $p = p_1 + p_2$. The inductive hypothesis on the premise of rule (Sum) ensures that $p_1 \xrightarrow{\sigma} p'$ implies that there exists a transition $t = (\{p_1\}, \sigma, dec(p'))$ such that $dec(p_1)[t\rangle dec(p')$ with $l(t) = \sigma$. Hence, the thesis follows by rule (sum): t implies $dec(p_1 + p_2) \xrightarrow{\sigma} dec(p')$.

If rule (Par) is the last rule used to derive $p \xrightarrow{\sigma} p'$, then $p = p_1 \,|\, p_2$. The inductive hypothesis on the premise of rule (Par) ensures that $p_1 \xrightarrow{\sigma} p_1'$ implies that there exists a transition $t = (H, \sigma, H')$ such that $dec(p_1)[t\rangle dec(p_1')$ with $l(t) = \sigma$. Hence, by rule (par), t implies transition $t|id = (H|id, \sigma, H'|id)$, and $t|id$ is such that $dec(p_1)|id[t|id\rangle dec(p_1')|id$ with $l(t|id) = \sigma$. The thesis then follows by additivity:
$dec(p_1 \,|\, p_2) = dec(p_1)|id \oplus id | dec(p_2)[t|id\rangle dec(p_1')|id \oplus id|dec(p_2) = dec(p_1' \,|\, p_2)$.

If rule (Com) is the last rule used to derive $p \xrightarrow{\sigma} p'$, then $p = p_1 \,|\, p_2$. The inductive hypothesis on the premise of rule (Com) ensures that $p_1 \xrightarrow{\sigma_1} p_1'$ and $p_2 \xrightarrow{\sigma_2} p_2'$ imply that there exist transitions t_1 and t_2 such that $dec(p_1)[t_1\rangle dec(p_1')$ with $l(t_1) = \sigma_1$, and $dec(p_2)[t_2\rangle dec(p_2')$ with $l(t_2) = \sigma_2$, with $Sync(\sigma_1, \sigma_2, \sigma)$. Hence, by rule (com), t_1 and t_2 imply transition $t_1|t_2 = ({}^\bullet t_1|id \oplus id|{}^\bullet t_2, \sigma, t_1^\bullet|id \oplus id|t_2^\bullet)$, and $t_1|t_2$ is such that $dec(p_1)|id \oplus id|dec(p_2)[t_1|t_2\rangle dec(p_1')|id \oplus id|dec(p_2')$. Hence: $dec(p_1 \,|\, p_2)[t_1|t_2\rangle dec(p_1' \,|\, p_2')$.

If rule (Res) is the last rule used to derive $p \xrightarrow{\sigma} p'$, then $p = (\nu a)p_1$. The inductive hypothesis on the premise of rule (Res) ensures that $p_1 \xrightarrow{\sigma} p_1'$ implies that there exists a t such that $dec(p_1)[t\rangle dec(p_1')$ with $l(t) = \sigma$. Hence, by rule (res), t implies transition $(\nu a)t = ((\nu a)^\bullet t, \sigma, (\nu a)t^\bullet)$, and $(\nu a)t$ is such that $(\nu a)dec(p_1)[(\nu a)t\rangle(\nu a)dec(p_1')$. Hence the thesis: $dec(p)[(\nu a)t\rangle dec(p')$.

Similarly as above if rule (Rel) is the last rule used to derive $p \xrightarrow{\sigma} p'$.

If rule (Rec) is the last rule used to derive $p \xrightarrow{\sigma} p'$, then $p = \text{rec}\, X.p_1$. The inductive hypothesis on the premise of (Rec) ensures that $p_1\{\text{rec}\, X.p_1/X\} \xrightarrow{\sigma} p'$ implies that there exists a transition t such that $dec(p_1\{\text{rec}\, X.p_1/X\})[t\rangle dec(p')$. Since $dec(\text{rec}\, X.p_1) = dec(p_1\{\text{rec}\, X.p_1/X\})$, the thesis follows. □

Proposition 6 For any process $p \in \mathcal{P}$, if $dec(p)[t\rangle K$ for some t such that $l(t) = \sigma$ then there exists p' such that $p \xrightarrow{\sigma} p'$ and $K = dec(p')$.

Proof:
By induction on (the definition of) $dec(p)$ and then by induction on the proof of the transition $t \in T_p$.

$dec(\mu.q) = \{\mu.q\}$, then a transition is derivable by axiom (pref), where ${}^\bullet t = \{\mu.q\}$, $l(t) = \mu$ and $t^\bullet = dec(q)$. The thesis follows by axiom (Pref).

$dec(\underline{\mu}.q) = \{\underline{\mu}.q\}$, then a transition t is derivable by rule (s-pref), where ${}^\bullet t = \{\underline{\mu}.q\}$, $l(t) = \mu\sigma'$ and $t^\bullet = H' \oplus K$. The inductive hypothesis on the premise of the rule (s-pref) ensures that $dec(q)[t'\rangle H' \oplus K$, with $t' = (H, \sigma', H')$, implies $q \xrightarrow{\sigma'} q'$ and $H' \oplus K = dec(q')$. The thesis follows by rule (S-pref).

$dec(p_1 + p_2) = \{p_1 + p_2\}$, then a transition t is derivable by rule (sum), where ${}^\bullet t = \{p_1 + p_2\}$, $l(t) = \sigma$ and $t^\bullet = H$. The inductive hypothesis on the premise of the rule (sum) ensures that $\{p_1\}[t'\rangle H$, with $t' = (\{p_1\}, \sigma, H)$, implies $p_1 \xrightarrow{\sigma} p'$ and $H = dec(p')$. The thesis follows by rule (Sum).

$dec(p_1 \,|\, p_2) = dec(p_1)|id \oplus id|dec(p_2)$, then three cases are possible. If a transition $t|id = (H|id, \sigma, H'|id)$ is enabled at $dec(p_1)|id$ and so $dec(p_1)|id[t|id\rangle K|id$, then the transition on the premise of the rule (par) $t = (H, \sigma, H')$ is such that $dec(p_1)[t\rangle K$. By induction, we know that $p_1 \xrightarrow{\sigma} p_1'$ with $K = dec(p_1')$. Hence the thesis follows by rule (Par). Symmetrically, if $id|t$ is enabled at $id|dec(p_2)$. The third case is when transition t has been derived by rule (com), i.e., $t = (H|id \oplus id|K, \sigma, H'|id \oplus id|K')$ with $H|id \subseteq dec(p_1)|id$ and $id|K \subseteq id|dec(p_2)$. The transitions on the premise of rule (com) are $t_1 = (H, \sigma_1, H')$ and $t_2 = (K, \sigma_2, K')$ for which $dec(p_1)[t_1\rangle K_1$ and $dec(p_2)[t_2\rangle K_2$. The inductive hypothesis ensures that $p_1 \xrightarrow{\sigma_1} p_1'$ and $p_2 \xrightarrow{\sigma_2} p_2'$ such that $K_i = dec(p_i')$ for $i = 1, 2$. The thesis then follows by rule (Com).

$dec((\nu a)p_1) = (\nu a)dec(p_1)$. If transition $(\nu a)t = ((\nu a)H, \sigma, (\nu a)H')$ is enabled at $(\nu a)dec(p_1)$, then transition $t = (H, \sigma, H')$ is enabled at $dec(p_1)$ and $dec(p_1)[t'\rangle K$. By inductive hypothesis, we know that $p_1 \xrightarrow{\sigma} p_1'$ and $K = dec(p_1')$. The thesis then follows by rule (Res).

$dec(p_1[f]) = dec(p_1)[f]$. Similar to the above.

$dec(\text{rec}\, X.p) = dec(p\{\text{rec}\, X.p/X\})$, then the thesis follows by induction as, by guardedness, $dec(p\{\text{rec}\, X.p/X\})$ is in one of the previous format. □

Proposition 7 For any process $p \in \mathcal{P}$, if $p \xrightarrow{M}_s p'$ then there exists a step G such that $dec(p)[G\rangle dec(p')$ with $l(G) = M$.

Proof:
The proof is by induction on the proof of transition $p \xrightarrow{M}_s p'$.

If axiom (Pref$_s$) has been used, then $p = \mu.q$, $M = \{\mu\}$ and $p' = q$. The thesis follows by observing that axiom (pref) ensures that transition $t = \{\mu.q\} \xrightarrow{\mu} dec(q)$ is derivable, so $dec(p)[\{t\}\rangle dec(p')$.

If rule (S-pref$_s$) is the last rule used to derive $p \xrightarrow{M}_s p'$, then $p = \underline{\mu}.q$ and $M = \{\mu\sigma\}$. The inductive hypothesis on the premise of rule (S-pref$_s$) ensures that $q \xrightarrow{\{\sigma\}}_s p'$ implies that there exists a singleton step G such that $dec(q)[G\rangle dec(p')$ with $l(G) = \{\sigma\}$. Hence, the unique transition of G is $t = (H, \sigma, H')$, and rule (s-pref) ensures that there is a derived transition $t' = \{\underline{\mu}.q\} \xrightarrow{\mu\sigma} dec(p')$ such that $dec(p)[\{t'\}\rangle dec(p')$.

If rule (Sum$_s$) is the last rule applied to derive transition $p \xrightarrow{M}_s p'$, then $p = p_1 + p_2$. The inductive hypothesis on the premise of rule (Sum$_s$) ensures that $p_1 \xrightarrow{M}_s p'$ implies that there exists a step G such that $(\{p_1\}[G\rangle dec(p')$ with $l(G) = M$. Actually G must be a singleton $\{t\}$, because no parallel transition can be performed by a sequential process such as p_1. Hence, the thesis follows by rule (sum): $t = (dec(p_1), M, dec(p'))$ implies $t' = dec(p_1 + p_2) \xrightarrow{M} dec(p')$.

$$GSync(G_1, G_2, G_1|id \oplus id|G_2) \quad \frac{Sync(\sigma_1, \sigma_2, \sigma) \;\; l(t_i) = \sigma_i, l(t_1|t_2) = \sigma \;\; GSync(G_1, G_2, G)}{GSync(G_1 \oplus \{t_1\}, G_2 \oplus \{t_2\}, G \oplus \{t_1|t_2\})}$$

Table 7. Synchronization of steps

If rule (Par$_s$) is the last rule used to derive $p \xrightarrow{M}_s p'$, then $p = p_1 \,|\, p_2$. The inductive hypothesis on the premise of rule (Par$_s$) ensures that $p_1 \xrightarrow{M}_s p_1'$ implies that there exists a step G such that $dec(p_1)[G\rangle dec(p_1')$ with $l(G) = M$. Hence, for each transition $t = (H, \sigma, H') \in G$, rule (par) ensures the existence of transition $t|id = (H|id, \sigma, H'|id)$; so, the step $G|id = \{t|id \;\; \big| \;\; t \in G\}$ is also enabled at $dec(p_1)|id$, i.e., $dec(p_1)|id[G|id\rangle dec(p_1')|id$ with $l(G|id) = l(G) = M$. The thesis then follows by additivity: $dec(p_1 \,|\, p_2) = dec(p_1)|id \oplus id|dec(p_2)[G|id\rangle dec(p_1')|id \oplus id|dec(p_2) = dec(p_1' \,|\, p_2)$.

If rule (Com$_s$) is the last rule used to derive $p \xrightarrow{M}_s p'$, then $p = p_1 \,|\, p_2$. The inductive hypothesis on the premise of rule (Com$_s$) ensures that $p_1 \xrightarrow{M_1}_s p_1'$ and $p_2 \xrightarrow{M_2}_s p_2'$ imply that there exist steps G_1 and G_2 such that $dec(p_1)[G_1\rangle dec(p_1')$ with $l(G_1) = M_1$, and $dec(p_2)[G_2\rangle dec(p_2')$ with $l(G_2) = M_2$, and that M_1 and M_2 are related such: $MSync(M_1, M_2, M)$. Now, by following the proof of $MSync(M_1, M_2, M)$ (according to the rules in Table 4), one can derive a step G as the result of step synchronization $GSync(G_1, G_2, G)$ (according to the rules in Table 7), as follows: if $MSync(M_1, M_2, M_1 \oplus M_2)$, then $GSync(G_1, G_2, G_1|id \oplus id|G_2)$; if instead $MSync(M_1 \oplus \{\sigma_1\}, M_2 \oplus \{\sigma_2\}, M \oplus \{\sigma\})$ is due to $Sync(\sigma_1, \sigma_2, \sigma)$ and $MSync(M_1, M_2, M)$, then, by induction we can assume that $GSync(G_1, G_2, G)$, and given t_1 and t_2 such that $l(t_i) = \sigma_i$, we can conclude $GSync(G_1 \oplus t_1, G_2 \oplus t_2, G \oplus t_1|t_2)$, where $t_1|t_2 = ({}^\bullet t_1|id \oplus id|{}^\bullet t_2, \sigma, t_1^\bullet|id \oplus id|t_2^\bullet)$. It is clear that the so derived step G is such that $dec(p_1)|id \oplus id|dec(p_2)[G\rangle \; dec(p_1')|id \oplus id|dec(p_2')$. Hence: $dec(p_1 \,|\, p_2)[G\rangle dec(p_1' \,|\, p_2')$ with $l(G) = M$.

If rule (Res$_s$) is the last rule used to derive $p \xrightarrow{M}_s p'$, then $p = (\nu a)p_1$. The inductive hypothesis on the premise of rule (Res$_s$) ensures that $p_1 \xrightarrow{M}_s p_1'$ implies that there exists a step G such that $dec(p_1)[G\rangle dec(p_1')$ with $l(G) = M$. Hence, by rule (res), for every $t \in G$ we get a transition $(\nu a)t = ((\nu a)^\bullet t, \sigma, (\nu a)t^\bullet)$, and step $(\nu a)G$ is such that $(\nu a)dec(p_1)[(\nu a)G\rangle(\nu a)dec(p_1')$. Hence the thesis: $dec(p)[(\nu a)G\rangle dec(p')$.

Similar to the above is the case when rule (Rel$_s$) is the last rule used.

If rule (Rec$_s$) is the last rule used to derive $p \xrightarrow{M}_s p'$, then $p = \text{rec}\, X.p_1$. The inductive hypothesis on the premise of (Rec$_s$) ensures that $p_1\{\text{rec}\, X.p_1/X\} \xrightarrow{M}_s p'$ implies that there exists a step G such that $dec(p_1\{\text{rec}\, X.p_1/X\})[G\rangle dec(p')$ with $l(G) = M$. Since $dec(\text{rec}\, X.p_1) = dec(p_1\{\text{rec}\, X.p_1/X\})$, the thesis follows. □

Proposition 8 For any process $p \in \mathcal{P}$, if there exists a step G such that $dec(p)[G\rangle K$, with $l(G) = M$, then there exists p' such that $p \xrightarrow{M}_s p'$ and $K = dec(p')$.

Proof:

By induction on (the definition of) $dec(p)$ and then by induction on the proof of the transitions in G.

$dec(\mu.q) = \{\mu.q\}$, then only one transition is derivable by axiom (pref), where ${}^\bullet t = \{\mu.q\}$, $l(t) = \mu$

and $t^\bullet = dec(q)$. The thesis follows by axiom (Pref$_s$).

$dec(\underline{\mu}.q) = \{\underline{\mu}.q\}$, then a transition t is derivable by rule (s-pref), where ${}^\bullet t = \{\underline{\mu}.q\}$, $l(t) = \mu\sigma'$ and $t^\bullet = H' \oplus K$. The inductive hypothesis on the premise of the rule (s-pref) ensures that $dec(q)[t'\rangle H' \oplus K$, with $t' = (H, \sigma', H')$, implies $q \xrightarrow{\{\sigma'\}}_s q'$ and $H' \oplus K = dec(q')$. The thesis follows by rule (S-pref$_s$).

$dec(p_1 + p_2) = \{p_1 + p_2\}$, then only singleton steps can be derivable, each of the form $G = \{t\}$, where t is derivable by rule (sum), with ${}^\bullet t = \{p_1 + p_2\}$, $l(t) = \sigma$ and $t^\bullet = H$. The inductive hypothesis on the premise of the rule (sum) ensures that $\{p_1\}[\{t'\}\rangle H$, with $t' = (\{p_1\}, \sigma, H)$, implies $p_1 \xrightarrow{\{\sigma\}}_s p'$ and $H = dec(p')$. The thesis follows by rule (Sum$_s$).

$dec(p_1 \mid p_2) = dec(p_1)|id \oplus id|dec(p_2)$, then three cases are possible. If a step $G|id$, composed of transitions $t|id$ of the form $(H|id, \sigma, H'|id)$, is such that $dec(p_1)|id[G|id\rangle K|id$, then the transitions t on the premise of the rule (par) of the form (H, σ, H') are such that $dec(p_1)[G\rangle K$. By induction, we know that $p_1 \xrightarrow{M}_s p_1'$ with $K = dec(p_1')$ and $M = l(G)$. Hence the thesis follows by rule (Par$_s$). Symmetrically, if $id|G$ is enabled at $id|dec(p_2)$. The third case is when step G is obtained as $GSync(G_1, G_2, G)$ (see Table 7); in such a case, we have to mimic the proof of $GSync(G_1, G_2, G)$ to build the proof of $MSync(M_1, M_2, M)$. If $GSync(G_1, G_2, G_1|id \oplus id|G_2)$, then $MSync(M_1, M_2, M_1 \oplus M_2)$; if instead $GSync(G_1 \oplus \{t_1\}, G_2 \oplus \{t_2\}, G \oplus \{t_1|t_2\})$ where $t_1|t_2 = ({}^\bullet t_1|id \oplus id|{}^\bullet t_2, \sigma, t_1^\bullet|id \oplus id|t_2^\bullet)$, $l(t_1) = \sigma_1$, $l(t_2) = \sigma_2$ with $Sync(\sigma_1, \sigma_2, \sigma)$ is derived by $GSync(G_1, G_2, G)$, then, by induction we can assume that $MSync(M_1, M_2, M)$ and conclude that $MSync(M_1 \oplus \{\sigma_1\}, M_2 \oplus \{\sigma_2\}, M \oplus \{\sigma\})$. It is clear that if the so derived step G is such that $dec(p_1)|id \oplus id|dec(p_2)[G\rangle dec(p_1')|id \oplus id|dec(p_2')$, then also the following $dec(p_1)[G_1\rangle dec(p_1')$ and $dec(p_2)[G_2\rangle dec(p_2')$ are true. By induction, we have that $p_1 \xrightarrow{M_1}_s p_1'$ and $p_2 \xrightarrow{M_2}_s p_2'$. Hence the thesis follows by (Com$_s$).

$dec((\nu a)p_1) = (\nu a)dec(p_1)$. If a step $(\nu a)G$ of transitions of the form $(\nu a)t = ((\nu a)H, \sigma, (\nu a)H')$ is enabled at $(\nu a)dec(p_1)$, then step G of transitions of the form $t = (H, \sigma, H')$ is enabled at $dec(p_1)$ and $dec(p_1)[G\rangle K$. By inductive hypothesis, we know that $p_1 \xrightarrow{M}_s p_1'$ and $K = dec(p_1')$. The thesis then follows by rule (Res$_s$).

$dec(p_1[f]) = dec(p_1)[f]$. Similar to the above.

$dec(\text{rec}\, X.p) = dec(p\{\text{rec}\, X.p/X\})$, then the thesis follows by induction as, by guardedness, $dec(p\{\text{rec}\, X.p/X\})$ is in one of the previous format. □