Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science 180 (2007) 77–94
Electronic Notes in Theoretical Computer Science
www.elsevier.com/locate/entcs

On the Expressiveness of Mobile Synchronizing Petri Nets

Fernando Rosa-Velardo David de Frutos-Escrig
Olga Marroquín-Alonso

Departamento de Sistemas Informáticos y Programación
Universidad Complutense de Madrid, Spain

Email: {fernandorosa,defrutos,alonso}@sip.ucm.es

Abstract

In recent papers we have introduced Mobile Synchronizing Petri Nets, a new model for mobility based on coloured Petri Nets. It allows the description of systems composed of a collection of (possibly mobile) hardware devices and mobile agents, both modelled in a homogenous way and abstracting from middleware details. Our basic model introduced a colour to describe localities, but still lacked appropriate primitives to deal with security, and in fact it was equivalent to P/T nets. Then, we introduced the primitives to cope with security: a new colour for identifiers, basically corresponding to the natural numbers, that are created by means of a special transition. This mechanism allows us to deal with authentication issues. In this paper we discuss the expressiveness of the extended model with the authentication primitives. More specifically, we study several instances of the classical reachability and coverability problems. Finally, we also study a more abstract version of the mechanism to create identifiers, using abstract names, close to those in the π-calculus or the *Ambient Calculus*. We have proved that both models are strictly in between P/T nets and Turing machines.

Keywords: Mobility, Petri nets, security, expressiveness

1 Introduction

The term *ubiquitous computing* [12,13] describes environments full of devices that compute and communicate with their surrounding context and, furthermore, interact with it in a highly distributed but pervasive way. In order to describe and analyze such environments, some features must be taken into account [8], such us distributed and mobile computing, security, privacy, boundaries and trust.

In [5,10] we have presented a new model for mobility, based on Petri Nets [3]. In [5] we introduced a basic version of the model called *Ubiquitous Nets*, in which

⋆ Work partially supported by the Spanish projects MIDAS TIC 2003-01000 and MASTER TIC 2003-07848-C02-01.

doi:10.1016/j.entcs.2005.05.048

processors provide services to processes that request them. Processors remain fixed in their locations, whereas processes move between processors in order to obtain the resources they need. Demand and supply of services is modelled by the synchronized firing of two transitions: a *service-demand* transition in the involved process and a *service-supply* transition in the corresponding processor. Mobility is formalized by the execution of *movement transitions*.

In [10] this simple model is enhanced by introducing colours, getting our *Mobile Synchronizing Petri Nets* (MSPN's). Now places can be occupied either by ordinary tokens or by *locality tokens*. The latter are used to specify the destination when firing a *go* transition. Next, in order to include a local identification mechanism, we add a new kind of coloured tokens. These are pairs of the form (i, n), where i is the index of some net and $n \in \mathbb{N}$.

As proved in [10], both Ubiquitous nets and Mobile Synchronizing nets without identifiers can be simulated by ordinary Petri nets. However, we did not know yet which was the expressiveness of MSPN systems when these identifiers are introduced. Since all along the paper we only consider this extension with identifiers, we will just use the term MSPN systems to refer to them. By means of labelled natural numbers that cannot be forged we get a mechanism to obtain authentication, that abstracts from the particular cryptographic techniques used to achieve it, typically proving possession of a secret key issued by some authority. In our setting this authority coincides with the principal asking for authentication, as usually happens in Ubiquitous Computing. In particular, it is enough to implement the most widely used authentication policy in Ubiquitous settings, the *resurrecting duckling policy* [11]. But one can argue that this mechanism based on natural numbers is too concrete, so that by examining the values of the corresponding tokens we can get information about when the identifiers were created. This is why we have also studied a more abstract version of the model, which we will call Abstract MSPN systems, where fresh identifiers are used instead of natural numbers.

We study the expressive power of both models by proving several decidability and undecidability results for several instances of the classical reachability and coverability problems. In particular, we prove that reachability and coverability are decidable for MSPN systems, but reachability of submarkings is not. Also, we prove that coverability is decidable in the case of Abstract MSPN systems, but reachability is not. These results show that their expressive powers lie somewhere between P/T nets and Turing machines.

The paper is structured as follows. Section 2 gives an overview of MSPN systems, together with an example. Section 3 considers reachability issues, whereas coverability results are presented in Section 4. Finally, conclusions and directions for further study are discussed in Section 5.

2 Mobile Synchronizing Petri Nets

In this section we will formally and intuitively describe Mobile Synchronizing Petri Nets (from now on, MSPN's). In the following, we will use the following notations:

$\mathcal{A}$ for autonomous transitions, with $go, succ \in \mathcal{A}$, S for services, $S! = \{s! \mid s \in S\}$ and $S? = \{s? \mid s \in S\}$ for offers and requests of those services respectively, $\bar{\cdot} : S! \cup S? \to S! \cup S?$ a conjugation function defined as $\overline{s?} = s!$ and $\overline{s!} = s?$, $\mathfrak{L}$ for locality names, $Var_{\mathfrak{L}}$ a set of locality variables, $Id = \bigcup_{i=1}^{\infty} \{(i,k) \mid k \in \mathbb{N}\}$, Var_{Id} a set of identity variables with $Var_{Aut} \subseteq Var_{Id}$, two special variables $\tau^+, \tau^- \in Var_{Id}$ and $Var_{\bullet} = \{\varepsilon\}$, and we take $Tokens = \mathfrak{L} \cup Id \cup \{\bullet\}$ and $Var = Var_{\mathfrak{L}} \cup Var_{Id} \cup Var_{\bullet}$.

Definition 2.1 A MSPN is a labelled Petri Net $N = (P, T, F, \lambda, C)$ where P is the set of places, T is the set of transitions, $F : (P \times T) \cup (T \times P) \to Var$ is a partial function from the set of arcs to variables, $C : P \to \{\mathfrak{L}, Id, \bullet\}$ is a function colouring places extended to arcs by $C((t,p)) = C((p,t)) = C(p)$, and $\lambda : T \to \mathcal{A} \cup S! \cup S?$ is a function labelling transitions, such that:

(i) For every $a \in Dom(F)$, $C(a) = \mathfrak{T} \Leftrightarrow F(a) \in Var_{\mathfrak{T}}$ for $\mathfrak{T} \in \{\mathfrak{L}, Id, \bullet\}$;

(ii) For every $t \in T$ such that $\lambda(t) \in \mathcal{A}$, $I(t) \subseteq O(t)$;

(iii) For every $t \in T$ with $\lambda(t) = go$, $|{}^{\bullet}t_{\mathfrak{L}}| = 1$;

(iv) $|\{t \mid \lambda(t) = succ\}| \leq 1$;

(v) If $\lambda(t) = succ$ there exists a place c such that:
- $\{{}^{\bullet}t_{Id}\} = \{{}^{\bullet}t_{Id} \cap t^{\bullet}_{Id}\} = \{c\}$ and $c \notin {}^{\bullet}s \cup s^{\bullet}$ for every $s \neq t$;
- $F(c,t) = \tau^-$, $F(t,p) = \tau^+$ for every $p \in t^{\bullet}_{Id}$ and τ^* and τ^- only appear in those arcs;

where ${}^{\bullet}t = \{p \in P \mid (p,t) \in F\}$, $t^{\bullet} = \{p \in P \mid (t,p) \in F\}$, $P_x = C^{-1}(x)$, ${}^{\bullet}t_x = {}^{\bullet}t \cap P_x$ with $x \in \{\bullet, \mathfrak{L}, Id\}$, $O(t) = \bigcup_{p \in {}^{\bullet}t} F(p,t) \backslash \{\varepsilon, \tau^+, \tau^-\}$ and $I(t) = \bigcup_{p \in t^{\bullet}} F(t,p) \backslash \{\varepsilon, \tau^+, \tau^-\}$.

A MSPN is a special kind of coloured Petri Net [6], with only three different colour types: one for localities, one for identifiers and a singleton colour type for ordinary black tokens. Arcs are labelled by variables that range over the set of values of the colour of its adjacent place, as stated in (i). Condition (ii) says that the variables in the outgoing arcs of a transition must appear in some incoming arcs, so that tokens can be consumed or moved by autonomous transitions, but not made up. Condition (iii) establishes the existence of a single locality precondition for movement transitions, those labelled by *go*, from where the net chooses its destination. According to the last two conditions, each net has at most one successor transition, in which case it has a single identifier precondition place that we will call *counter* of the net (denoted in the previous definition by c), that is also a postcondition of that transition, and that it is neither a precondition nor a postcondition of any other. These restrictions avoid the possibility to use these transitions as a mechanism to simulate ordinary counters, that is, natural variables. In particular, every time the *succ* transition is fired it produces a new value.

Definition 2.2 A marking M of a MSPN $N = (P, T, F, \lambda, C)$ is a function $M : P \to MS_f(Tokens)$ mapping each place to a finite multiset of tokens. We will say a marking is legal if $M(P_x) \subseteq MS_f(x)$ for $x \in \{\mathcal{L}, Id, \bullet\}$.

A MSPN system is just a collection of MSPN's and a marking of a MSPN system is a marking for each of its component nets together with a location function, mapping each net to a locality.

Definition 2.3 A MSPN system is a set $\mathcal{N} = \{N_i\}_{i=1}^n$ of disjoint MSPN's, $N_i = (P_i, T_i, F_i, \lambda_i, C_i)$. A marking of a MSPN system is a pair (M, loc) where $M = (M_1, \ldots, M_n)$ is such that each M_i is a marking of the MSPN N_i and $loc : \mathcal{N} \to \mathcal{L}$ is a function that maps each MSPN to a location.

Given such a marked MSPN system we will usually write $P = \bigcup_{i=1}^{n} P_i$, $T = \bigcup_{i=1}^{n} T_i$, $M(p)$ to denote $M_i(p)$ when $p \in P_i$, $F(a)$ to denote $F_i(a)$ if $a \in F_i$ and $\lambda(t)$ to denote $\lambda_i(t)$ when $t \in T_i$. With these notations we can take markings of MSPN systems as pairs (M, loc) with $M : P \to MS_f(Tokens)$. We use identifiers of the form (i, m) where i is the index of the net N_i where it was created, and $m \in \mathbb{N}$. For that purpose, we will assume that when the system is composed of n nets only identifier tokens with its first component in $\{1, \ldots, n\}$ will appear in any initial marking (and consequently, in any marking as we will see).

MSPN's may have by definition four types of transitions: autonomous transitions, synchronizing transitions, movement transitions and successor transitions. Autonomous transitions are just ordinary transitions in coloured nets [6], although we impose a constraint to the set of variables around any autonomous transition, so that locality and identifier tokens can be transferred or even replicated, but not made up (see Figure 1): The set of variables in outgoing arcs must be contained in the set of variables in incoming arcs. Since our nets belong to a special class of coloured nets their transitions fire relative to some *mode*, that chooses among the possible tokens that can be taken from the preconditions.

Definition 2.4 Using the previous notations, given an autonomous transition $t \in T_i$, a mode of t is any mapping $\sigma : Var(t) \to Tokens$ such that

(i) $\sigma(z) \in \mathcal{T} \Leftrightarrow z \in Var_{\mathcal{T}}$ for $\mathcal{T} \in \{\mathcal{L}, Id, \bullet\}$

(ii) If $\tau^+, \tau^- \in Var(t)$ then $\sigma(\tau^-) = (i, m)$ and $\sigma(\tau^+) = (i, m+1)$ for some $m \in \mathbb{N}$.

For homogeneity we are assuming in the definition that every arc is labelled with a variable. However, we only need variables to distinguish between the occurrence of different locality tokens and identity tokens. That is why we introduce the special variable ε, that labels every arc that is adjacent to an ordinary black-token place. Moreover, variables from $Var_{\mathcal{L}}$ will only be used for arcs that are adjacent to locality places and those from Var_{Id} only for arcs next to an identifier place.

Definition 2.5 [Autonomous transition] Let $\mathcal{N}$ be a MSPN system, $t \in T_i$ with $\lambda(t) \in \mathcal{A}$ and (M, loc) a marking of $\mathcal{N}$, using the previous notations. Transition t is enabled with mode σ if for all $p \in {}^\bullet t$, $\sigma(F(p,t)) \in M(p)$. The reached state of $\mathcal{N}$ after the firing of t with mode σ is the marking (M', loc') given by:

(i) For every $p \in P, M'(p) = (M(p) \backslash \sigma(F(p,t))) \cup \sigma(F(t,p))$.

(ii) If $\lambda(t) \neq go$ then $loc' = loc$. Otherwise, $loc'(N_j) = loc(N_j)$ for every $j \neq i$ and $loc'(N_i) = \sigma(F(p,t))$, where $p \in {}^{\bullet}t_{\mathfrak{L}}$.

Movement transitions are as autonomous ones, except that they change the location of the net firing that transition. For that purpose, every movement transition has a single distinguished locality-place to specify the destination of movements. Finally, successor transitions take an identifier (i, m) and yield its successor in the same domain, $(i, m+1)$, whenever the transition belongs to net N_i. The arc from this special precondition place to the successor transition is labelled by τ^- and every arc to an identifier postcondition is labelled by τ^+. In fact, the use of these new special variables τ^+ and τ^- is redundant, since the single precondition arc and all the postcondition arcs leading to identifier places must be labelled in this way. We just use them for simplicity in our definitions, so they are not shown in figures.

Definition 2.6 Given a pair of synchronizing transitions t_1 and t_2, we will say they are compatible if:

(i) $\lambda(t_1) = \overline{\lambda(t_2)}$

(ii) $I(t_1, t_2) \subseteq O(t_1, t_2)$

(iii) If $z \in O(t_i) \cap Var_{Aut}$ then $z \in O(t_j)$ for $i, j \in \{1, 2\}$, $i \neq j$.

where $I(t_1, t_2) = I(t_1) \cup I(t_2)$ and $O(t_1, t_2) = O(t_1) \cup O(t_2)$.

The firing of a synchronizing transition needs the presence of a compatible transition in the same location. The compatibility conditions are merely syntactical: On the one hand, their labels must be conjugate, which intuitively means that one offers the service and the other must be requesting it; On the other hand, the pair of transitions must meet together the same constraint as autonomous transitions, that is, every variable appearing in outgoing arcs must appear in some incoming arc of any of the transitions (see Figure 1). Only when both transitions are co-located and separately fireable according to the ordinary firing rule, they can be simultaneously fired. The last condition justifies the use of the variables in Var_{Aut}. Whenever such a variable labels a precondition arc of a synchronizing transition it must also appear as label of a precondition in its compatible transitions. In this way we can force the matching of two identifiers, each in one of the nets that synchronizes.

Definition 2.7 Let (t, t') be a pair of compatible transitions. We will call mode of (t, t') to any mapping $\sigma : Var(t, t') \to Tokens$ such that $\sigma(z) \in \mathfrak{T} \Leftrightarrow z \in Var_{\mathfrak{T}}$ for $\mathfrak{T} \in \{\mathfrak{L}, Id, \bullet\}$, where $Var(t_1, t_2) = I(t_1, t_2) \cup O(t_1, t_2)$.

Definition 2.8 [Synchronizing transition] Let $\mathcal{N}$ be a MSPN system, t_i and t_j two compatible transitions and (M, loc) a marking of $\mathcal{N}$. We say that the pair of transitions (t_i, t_j) is enabled with mode σ if for all $p \in {}^{\bullet}t_h$, $\sigma(F(t_h, p)) \in M(p)$ for $h \in \{i, j\}$. The reached state of $\mathcal{N}$ after the firing of (t_i, t_j) with mode σ is the marking (M', loc) where for every $p \in P$,

$$M'(p) = (M(p) \setminus \bigcup_{h \in \{i,j\}} \sigma(F(p, t_h))) \cup \bigcup_{h \in \{i,j\}} \sigma(F(t_h, p))$$

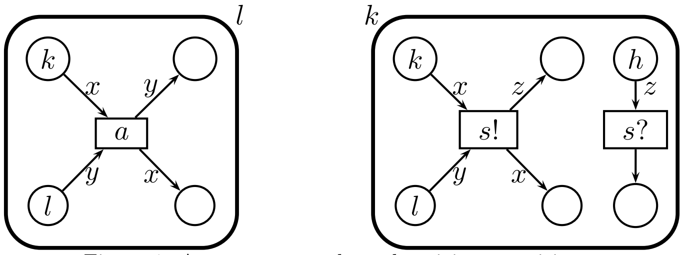

Figure 1. Autonomous and synchronizing transitions

An important characteristic of this class of coloured nets is that we do not have predicates limiting the firing of transitions (the set of consumed tokens), except for the matching obtained when using the same variable in different arcs. In particular, we cannot check the disequality or inequality of tokens.

2.1 Abstract names generation

We have restricted the use of identity naturals in the previous section so that they can only be used for authentication purposes. In particular, those restrictions avoid the use of identifiers to simulate counters, that is, natural variables, as could be done if no restriction had been imposed on successor transitions.

We have indeed obtained a mechanism to implement authentication by means of identifiers containing natural numbers. However, the use of natural numbers as identities has several undesirable consequences: First, when examining a marking from the outside, one can know if an identifier was generated before or after any other of the same net, which is clearly unrealistic. Besides, we can also know that some identifiers have been created during the computation even if they are not present in the current marking. To avoid this we would need some kind of "garbage collection", so that unused identifiers could be reused.

To avoid these problems we consider an alternative model, where the domain of identifiers *Id* is any countable set. The only difference between both models is that the successor transitions are changed by transitions *new*, that create fresh identifiers. The firing of the new transition changes accordingly:

Definition 2.9 [*New* transition] Let $\mathcal{N}$ be a MSPN system, $t \in T$ with $\lambda(t) = new$, Q the set of identifier postconditions of t and (M, loc) a marking of $\mathcal{N}$. Transition t is enabled with mode σ if for all $p \in {}^{\bullet}t$, $\sigma(F(p,t)) \in M(p)$. The reached state of $\mathcal{N}$ after the firing of t with mode σ is the marking (M', loc) given by:

(i) For every $p \in P \backslash Q, M'(p) = (M(p) \backslash \sigma(F(p,t))) \cup \sigma(F(t,p))$.

(ii) $M'(q) = (M(q) \backslash \sigma(F(q,t))) \cup \{\eta\}$ where $\eta \notin M(p) \forall q \in Q$.

Note that all the postcondition identifier tokens, even if associated to postcondition arcs labelled by different variables, are equal to the same value η.

In order to deal with really abstract names we need to introduce a notion of α-equivalence on markings, borrowed from renaming of bounded names in the π-calculus: We identify markings up to permutations of abstract names.

Definition 2.10 We say that two markings (M, loc) and (M', loc') of a MSPN

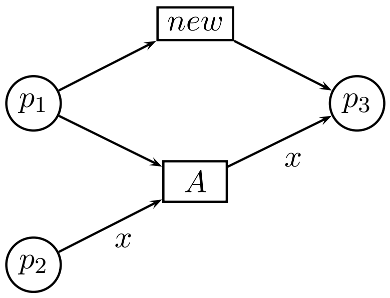

Figure 2. α-equivalent markings

system are α-equivalent if there exists a bijection $h : Id(M) \to Id(M')$ such that:

(i) $loc = loc'$

(ii) $M(p) = M'(p)$ for all $p \in P_\bullet \cup P_{\mathfrak{L}}$

(iii) $M(q)(\eta) = M'(q)(h(\eta))$ for all $q \in P_{Id}$ and $\eta \in Id(M)$

with $Id(M) = \{\eta \mid \exists p \in P_{Id}, \eta \in M(p)\}$. We will write $(M, loc) \equiv_h (M', loc')$ or just $M \equiv M'$ when there is no confusion.

As usually done, let us denote by $M = (A_1, \ldots, A_k)$ the marking of a MSPN system with places $P = \{p_1, \ldots, p_k\}$ such that $M(p_i) = A_i$. Then the marking $(\{\bullet, \bullet\}, \{k\}, \{\eta, \eta'\}, \{\eta'\}\}$ is α-equivalent to $(\{\bullet, \bullet\}, \{k\}, \{\eta, \eta'\}, \{\eta\}\}$, where $k \in \mathfrak{L}$ and $\eta, \eta' \in Id$, since the mapping $\eta \mapsto \eta'$, $\eta' \mapsto \eta$ is a bijection that fulfills the conditions.

Proposition 2.11 *The behaviour of abstract MSPN systems is invariant under α-conversion. More specifically, if $M_1 \equiv_h M_2$ and $(M_1, loc)[t(\sigma)\rangle(M_1', loc')$ then there is some M_2' with $M_1' \equiv M_2'$ such that $(M_2, loc)[t(h(\sigma))\rangle(M_2', loc')$, where $h(\sigma)(x) = h(\sigma(x))$, if $x \in Var_{Id}$, and $h(\sigma)(x) = \sigma(x)$, otherwise.*

Proof If $\lambda(t) \neq new$ then take $M_2' = h(M_2)$, where $h(M)(p)(h(\eta)) = M(p)(\eta)$ for $p \in P_{Id}$. It holds that $M_1' \equiv_h M_2'$ and $M_2[t(h(\sigma))\rangle M_2'$. If $\lambda(t) = new$ then $Id(M_2) = Id(M_1) \cup \{\eta\}$ for some new η. Then t is fireable also in M_2 with mode $h(\sigma)$, which yields M_2' with $Id(M_2') = Id(M_1') \cup \{\eta'\}$. Let us define h' by extending h with $h(\eta) = \eta'$, that verifies $M_1' \equiv_{h'} M_2'$. $\square$

For example, $M_1 = (\bullet, \eta, \emptyset)$ and $M_2 = (\bullet, \eta', \emptyset)$ are two α-equivalent markings of the MSPN system composed by a single net shown in Figure 2. M_1 can evolve to $(\emptyset, \emptyset, \eta)$ when firing A or to $(\emptyset, \eta, \eta'')$ for some new η'' when firing *new*. Analogously, M_2 can evolve to $(\emptyset, \emptyset, \eta')$ when firing A or to $(\emptyset, \eta', \eta''')$ for some new η''' when firing *new*. In both cases the reached markings are indeed α-equivalent for any η'' and η''' such that $\eta'' \neq \eta$ and $\eta''' \neq \eta'$. In particular, we can take $\eta'' = \eta'$ and $\eta''' = \eta$, which would yield $(\emptyset, \eta, \eta')$ and $(\emptyset, \eta', \eta)$, that are indeed equivalent.

2.2 A mutual exclusion protocol

The following example models a simple protocol of mutual exclusion in a distributed setting. The system is composed of four principals: the ticket server, the ticket

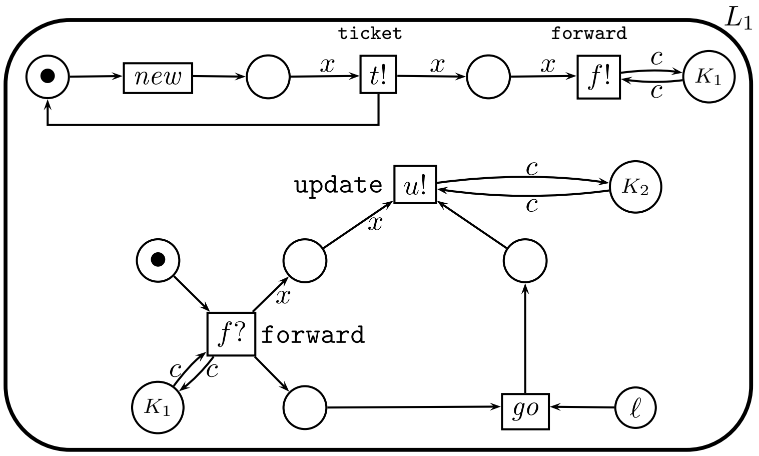

Figure 3. Ticket server and forwarder agent

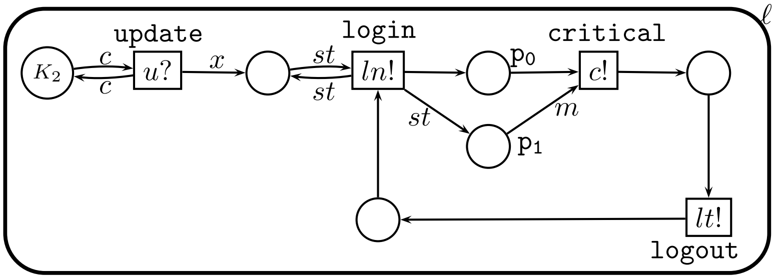

Figure 4. Critical server

forwarder, the critical server and the clients. The ticket server and the forwarder agent are shown in Figure 3. The ticket server (the one at the top of the figure) generates the tickets that clients must exhibit when accessing the critical section in the critical server. It outputs the tickets by synchronizing on transition `ticket` and tells the forwarder agent (bottom of Figure 3) to update the critical server (Figure 4) with that ticket as a valid one by synchronizing with it on transition `forward`. The ticket server and the forwarder agent communicate using a shared key K_1, while the critical server and the forwarder do it using key K_2. This is formalized simply by assuming that the variable c labelling some of the arcs is an authentication variable, that is, that it belongs to Var_{Aut}. At this point the forwarder agent can go to ℓ, the location where the critical server is, and update it, that is, synchronize with it on transition `update` and output the identifier that the ticker server gave to the client. The clients, that may be like the one in Figure 5, after receiving their ticket, can go to the location where the critical server is located and log in. Only at that moment, and showing its ticket once again, they can access the critical section, represented by transition `critical`. Only after they log out a new client may log in.

The critical system is safe if two different clients can never access the critical section at the same time, that is, if the following marking cannot be covered: $M(p_0) = 2$ and $M(q) = \emptyset$ for the rest of places. In Section 4 we will prove decidability of coverability for abstract MSPN systems, so that the safety of the system can be automatically proved.

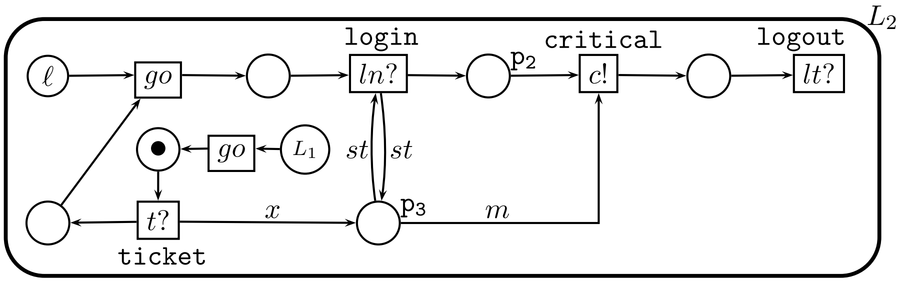

Figure 5. Clients

3 Reachability results

In this section we will consider the classical reachability problems. We will prove that reachability is decidable for MSPN systems, but undecidable for abstract ones. Instead, submarking reachability is also undecidable for MSPN systems, what will distinguish them from P/T nets, proving that they are more expressive.

3.1 Reachability is decidable for MSPN systems

We prove decidability of reachability for MSPN systems by reducing it to reachability for MSPN systems without identifiers. As we proved in [10] these class of MSPN systems is equivalent to ordinary P/T nets, for which reachability is decidable [9]. Given the MSPN system $\mathcal{N} = \{N_1, \ldots, N_n\}$ and the marking (M, loc) to reach from the initial marking (M_0, loc_0), we construct the associated system without identifiers $\mathcal{N}^* = \{N_1^*, \ldots, N_n^*\}$ with initial marking (M_0^*, loc_0) and a marking (M^*, loc) such that it is reachable in $\mathcal{N}^*$ from (M_0^*, loc_0) if and only if (M, loc) is reachable from (M_0, loc_0) in $\mathcal{N}$.

Without loss of generality, we can suppose that every net N_i has a counter place, marked in M with an identifier token (i, n_i). If that is not the case for some net N_i we can always add an isolated place, and mark it with the maximum identifer (relative to its second component) of the form (i, k) appearing in M_0. Then, let us denote by $\overline{Id} = \{(i, k) \mid 0 \le k \le n_i, 1 \le i \le n\}$, the set of identifer tokens that can appear on any sequence of markings reaching M. The key point is to unfold each $p \in P_{Id}$ to $\{p(i, k) \mid (i, k) \in \overline{Id}\}$, so that each token in a place $p(i, k)$ will simulate the occurrence of the identifer (i, k) in p. Of course, this simulation only works for a finite amount of time, as long as only identifiers in $\overline{Id}$ appear, which is enough to decide reachability of (M, loc).

We also need to unfold those $t \in T$ adjacent to identifier places. For it, we will consider partial modes.

Definition 3.1 Let t be a transition of a MSPN system. We call *partial mode* of t to any mapping $\sigma : Var_{Id}(t) \to \overline{Id}$ such that if $\lambda(t) = succ$ and $t \in T_i$ then $\sigma(\tau^-) = (i, m)$ and $\sigma(\tau^+) = (i, m + 1)$, for some $m \in \mathbb{N}$.

Partial modes are modes that are only defined in $Var_{Id}(t)$, the set of identifier variables, that is, modes that only decide which identifiers (not which localities) are taken from the preconditions. Now we unfold each $t \in T$ into $\{t_\sigma \mid \sigma : Var_{Id}(t) \to$

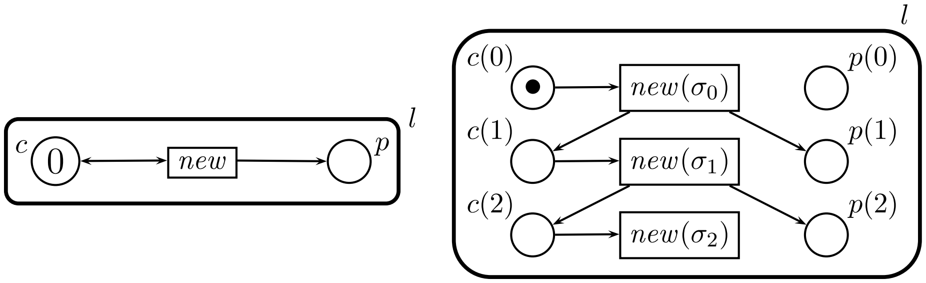

Figure 6. Unfolding of a MSPN system with respect to $M(c)=\{2\}$ and $M(p)=\emptyset$

$\overline{Id}$, σ partial mode $\}$. If (t,p) is an arc of some net in $\mathcal{N}$ with $C(p) = Id$ we add arcs $\{(t_\sigma, p(j,k)) \mid \sigma(F(t,p)) = (j,k)\}$, and analogously for (p,t).

Lemma 3.2 *Let t be a transition of a MSPN system and σ_I a partial mode of t. If σ is a mode for t then there is a mode $\sigma_{\mathfrak{L}}$ for t_{σ_I} such that $\sigma = \sigma_I \sqcup \sigma_{\mathfrak{L}}$. Reciprocally, if $\sigma_{\mathfrak{L}}$ is a mode for t_{σ_I} then $\sigma_I \sqcup \sigma_{\mathfrak{L}}$ is a mode for t.*

Thus, the firing of t_{σ_I} may simulate any firing of t in some mode that extends σ_I. In particular, by definition of partial mode, whenever $\lambda(t) = succ$ for $t \in T_i$, $q \in {}^\bullet t_{Id}$ we are adding arcs $(q(i,m), t_\sigma)$ and $(t_\sigma, p(i,m+1))$ for every $p \in t^\bullet_{Id}$ and any σ with $\sigma(\tau^-) = (i,m)$. Figure 6 shows an unfolding of a very simple net. In that case $\overline{Id} = \{(1,0),(1,1),(1,2)\} \approx \{0,1,2\}$.

For any marking M of $\mathcal{N}$ with $M(P_{Id}) \subseteq \overline{Id}$ we define M^* of $\mathcal{N}^*$ by

$$M^*(q) = \begin{cases} M(q) \text{ if } q \in P_\bullet \cup P_{\mathfrak{L}} \\ M(p)((i,m)) \text{ if } q = p(i,m) \text{ for } p \in P_{Id} \end{cases}$$

Then we have the following result.

Theorem 3.3 *Using the previous notation, if $M_i(p) \subseteq \overline{Id} \forall p \in P_{Id}$ and $i \in \{1,2\}$, then $(M_1, loc_1)[t(\sigma_{\mathfrak{L}} \sqcup \sigma_I)\rangle(M_2, loc_2) \Leftrightarrow (M_1^*, loc_1)[t_{\sigma_I}(\sigma_{\mathfrak{L}})\rangle(M_2^*, loc_2)$.*

Corollary 3.4 *Reachability is decidable for MSPN systems.*

Proof If (M, loc) is the marking whose reachability we want to decide, we define the MSPN system without identifiers shown above. By the previous theorem, that system faithfully simulates the original system as far as only identifer tokens in $\overline{Id}$ appear. Since M is reachable if and only if there is a sequence of states with identifiers within $\overline{Id}$ that reach M, we have that M is reachable in the original system if and only if M^* is reachable in the simulation, where reachability is decidable. □

Note that the key fact to obtain this last result is that the value of the counter of each net component is a bound of the identifiers generated by that net in the current marking of the system.

3.2 Reachability is undecidable for Abstract MSPN's

We show undecidability of reachability of abstract MSPN systems by reducing reachability of a state $\langle p, c_1 = 0, c_2 = 0\rangle$ of a two counter machine (TCM) to that of some marking of an abstract MSPN system, using the technique already used[1] in [4]. A TCM is a finite state machine operating on two counters, c_1 and c_2, that can perform the following kind of operations on them: check whether the counter is zero; increment the counter by one; decrement the counter by one (if it's already zero, this leaves it unchanged).

Formally, a TCM consists of a finite set of states $\mathcal{S} = \{s_0, \ldots, s_k\}$ and a set of instructions $\mathcal{I} = \mathit{Inc} \cup \mathit{Dec}$: $\mathit{Inc}(i, s, t) \in \mathit{Inc}$ increases counter c_i by one when at state s, and moves to state t; $\mathit{Dec}(i, s, t, u) \in \mathit{Dec}$ when in state s decreases the counter c_i by one and moves to state t, if $c_i > 0$, or just moves to u, otherwise. The initial state is $\langle s_0, c_1 = 0, c_2 = 0\rangle$.

We reduce reachability of TCM to reachability in abstract MSPN's by defining an abstract MSPN system, which will consist on a single net, without locality places or moving or synchronizing transitions.

At each marking there will be a single valid identifier value. To identify that value we use a special place *current*, where we storage the identity token with the current valid name. Then, we will simulate the value of the counter c_i with the amount of (valid) tokens in place c_i. We use ordinary places $\{s_0, \ldots, s_k\}$, one for each state of the TCM, so that a token at s_i means that the TCM is at the state s_i. Moreover, we will use some other auxiliary ordinary places to manipulate the control.

Instructions of the form $\mathit{Inc}(i, s, t)$ are simulated straightforwardly, by adding to c_i a copy of the token at *current* and passing the control (that is, one black token) from s to t (see Figure 7). However, those of the form $\mathit{Dec}(i, s, t, u)$ can only be weakly simulated, since we do not have the chance of checking for zero. The key idea is to decide non-deterministically whether c_i is zero or not and proceed. As a consequence, we can have both faithful and incorrect simulations of the TCM. In a correct run of the system, only valid tokens (those that coincide with the value stored at *current*) appear in places c_1 and c_2.

Figure 8 shows the case where we choose to decrement the counter, which can only be done when it is legal to do it, that is, when there is a (valid) token at the counter. In that case we can remove one of those valid tokens and move the black token from the old state to the new one. The case in which we guess that c_i is zero is more tricky, since this could be a bad guess. Figure 9 shows the simulation in that case. In order to identify whether we made a good guess, we will generate a new valid name and substitute (transition *swap*) every token at the other counter with the new valid name. Since we do not manipulate counter c_i, only if the guess was correct (that is, if counter c_i had no tokens) we will maintain the invariant that says that every token appearing at the counters is valid, since invalid tokens cannot be removed. Once we have swapped every old token by the new valid one we can fire *reset*. Once again, it could be possible to fire this transition even when not every invalid identifier in the counter has been replaced, which would also leave invalid garbage identifiers.

[1] Thanks to G. Delzanno for suggesting us to reuse here that technique we developed some time ago for the study of timed nets.

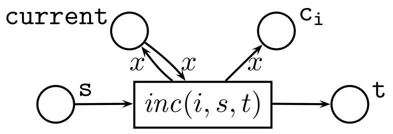

Figure 7. Simulation of $Inc(i, s, t)$

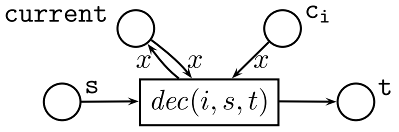

Figure 8. Simulation of $Dec(i, s, t, u)$ (guessing $c_i > 0$)

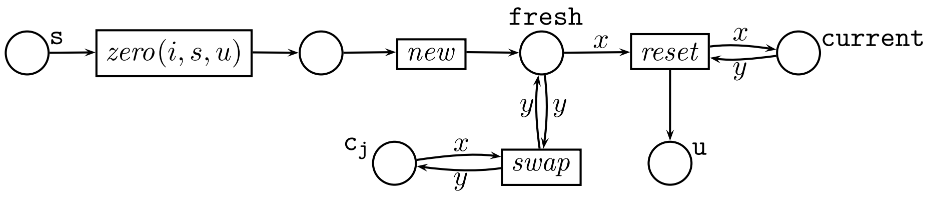

Figure 9. Simulation of $Dec(i, s, t, u)$ (guessing $c_i = 0$)

As initial marking we take M_0 with only one black token at place s_0 and an arbitrary identifier token at *current*. The weak simulation will be enough to prove undecidability of reachability.

Lemma 3.5 *The state $\langle s, c_1 = 0, c_2 = 0\rangle$ is reachable from $\langle s_0, c_1 = 0, c_2 = 0\rangle$ if and only if there exists a reachable marking M from M_0 such that $M(s) = 1$, $M(t) = 0$ for every $t \neq s$ in $P_\bullet$, $M(current) = \eta$ for some $\eta \in Id$, $M(q) = \emptyset$ for $q \in P_{Id} \setminus \{current\}$*

Proof The proof is based on the fact that any run of the constructed MSPN system that reaches a marking M in which $M(c_i) = \emptyset$ simulates a run of the TCM, which means that all the guesses have been correct, because otherwise there would be garbage tokens in some of the counters, that cannot be removed by construction. Besides, all the runs of the machine can be correctly simulated, by making the correct guess when deciding whether a place is empty or not. □

Corollary 3.6 *Reachability is undecidable for abstract MSPN systems.*

Proof It is well known that reachability of TCM is equivalent to reachability of the special states $\langle s, c_1 = 0, c_2 = 0\rangle$. Let us suppose that reachability for abstract MSPN systems is decidable, and use it to decide whether a given $\langle s, c_1 = 0, c_2 = 0\rangle$ is reachable. We take M such that $M(s) = 1$, $M(t) = 0$ for every $t \neq s$ in $P_\bullet$, $M(current) = \eta$ for some $\eta \in Id$, $M(q) = \emptyset$ for $q \in P_{Id} \setminus \{current\}$. By Lemma 3.5 we know it is reachable if and only if $\langle s, c_1 = 0, c_2 = 0\rangle$ is reachable. All we have to do is to choose any of those M and decide whether it is reachable or not. □

3.3 Submarking reachability is undecidable for MSPN's

The result that gives title to this section can be proven in a very similar manner to that of undecidability of reachability for abstract MSPN systems, but due to its importance we prefer to present it in a different section. Indeed, the fact that reachability has been proved to be decidable for MSPN systems could lead us to the wrong impression that they are equivalent to ordinary P/T-nets but, as we will see next, this is not the case.

Given $\langle p, c_1 = 0, c_2 = 0\rangle$, a marking of a TCM, we define a MSPN system that weakly simulates it. The simulation of *Inc* and *Dec* is as before. In the case in which we guess $c_i = 0$, the label *new* is changed by *succ* with its associated counter place *counter*. In order to control when the test for zero is correctly simulated, we proceed as we did in the previous section. We use a *succ* transition generating the identifiers for the net. Each time a $zero(i, s, u)$ transition is executed we increment the counter and then we refresh the tokens in the place c_j with the new value of the counter place. However, by definition of MSPN systems, there can only be one successor transition, so that all the counter places and *succ* transitions must be the same for any of such instructions. For that reason we must be careful and only allow the firing of the transition *reset* that corresponds to that instruction. That is why we need the extra control place as postcondition of $zero(i, s, u)$ (see Figure 10).

Now the relation between the TCM and its weak simulation is expressed in the following proposition.

Proposition 3.7 *A state $\langle p, c_1 = 0, c_2 = 0\rangle$ of the given counter machine is reachable if and only if there exists $N \in \mathbb{N}$ such that the marking M_N given by $M_N(counter) = M_N(current) = \{(1, N)\}$, $M_N(p) = 1$ and $M_N(q) = \emptyset$, for every $q \notin \{counter, current, p\}$, is reachable in the simulation.*

As an immediate consequence we have the following theorem.

Corollary 3.8 *Submarking reachability of MSPN systems is undecidable.*

Proof Given a state $\langle p, c_1 = 0, c_2 = 0\rangle$ of the given counter machine and the MSPN system constructed above, let M be the marking defined in every place but the counter place and place *current* by $M(p) = 1$ and $M(q) = \emptyset$, for every $q \notin \{counter, current, p\}$. This submarking is reachable if and only if some marking M_N is reachable (following the notations of the previous proposition), which happens if and only if the state $\langle p, c_1 = 0, c_2 = 0\rangle$ is reachable in the TCM, according to the previous result. □

It is well known that submarking reachability in P/T nets is equivalent to reachability, since given a partial marking M we can build a new net with transitions that remove every token from those places not in the domain of M. Then reachability of the partial marking M is equivalent to reachability of the global marking M' that extends M with $M(p) = \emptyset$ for every $p \notin Dom(M)$. But the restrictions on transitions *succ* avoid the possibility to remove the tokens in the counter, so that this reasoning cannot be applied to our nets.

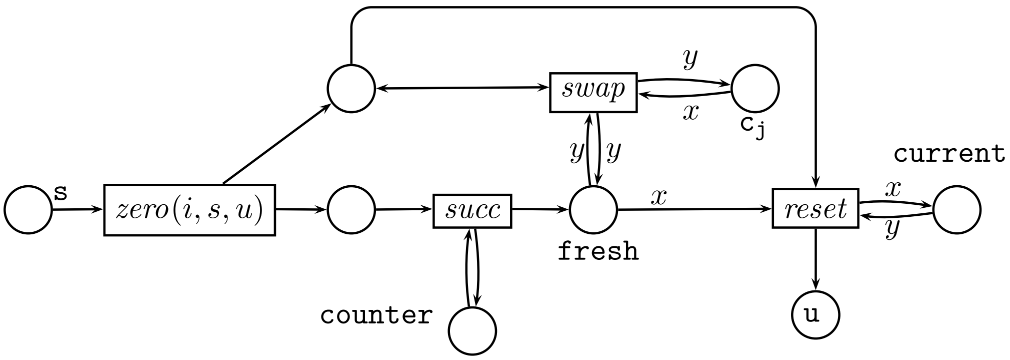

Figure 10. Simulation of $Dec(i,s,t,u)$ with naturals as identifiers (guessing $c_i = 0$)

Both simulations of the TCM's, the one using naturals as identifiers and the one using abstract names, use a place *current*, where the valid tokens are stored. In the case of natural identifiers, the token in this place coincides with the value in the *counter* place. However, the information that can be extracted from the value at that place is rather different in each case. If we use natural numbers, we know part of the history of the process: an upper bound on the number of created identifiers and the order in which they were created. However, if we use abstract names we do not know any of the two previous facts. In fact, we do not care for the concrete name that we find in *current*, since we are working under α-conversion and, therefore, it could be any concrete name.

4 Coverability results

In the previous section we considered reachability problems. In particular, reachability in abstract MSPN systems and submarking reachability in MSPN systems have been proved to be undecidable. Fortunately, safety properties of our systems are usually defined in terms of coverability. In this section we prove that coverability is indeed decidable both for MSPN systems and abstract MSPN systems.

4.1 Coverability is decidable for Abstract MSPN's

In order to prove decidability of coverability we use the technique based on well quasi-orders [1,2,7]. The technique consists on defining a suitable order (a well quasi-order) over the set of markings.

Definition 4.1 We say that a preorder $\sqsubseteq$ (a reflexive and transitive relation) is a well-quasi order if for every sequence $(a_i)_{i=1}^{\infty}$ there are indices $i < j$ such that $a_i \sqsubseteq a_j$.

To simplify our explanations we will focus on identifier places, since identifiers are our source of infinity (the order we will define is the identity when restricted to ordinary tokens and localities). Thus, regarding identifiers, a marking is a function $M : P \to Id \to \mathbb{N}$ that says given a place p and an identifier η how many tokens η can be found in p. However, we can currify them as $M : Id \to P \to \mathbb{N}$. Since identifiers are abstract names, the behaviour of a system does not depend on the particular names chosen, that is, it is invariant under α-conversion, as we proved in Proposition 2.11. Then we can take as (abstract) markings the multisets in $\mathcal{MS}(P \to \mathbb{N})$, and since $|P| = k < \infty$, also those in $\mathcal{MS}(\mathbb{N}^k)$.

In this way, a marking is represented by a multiset, with the cardinality of the set of different identifiers appearing in it. Consider for instance a net with two identifier places p_1 and p_2 and a marking M such that $M(p_1) = \{\eta, \eta, \eta', \nu\}$ and $M(p_2) = \{\eta', \nu\}$. That marking would be represented in this way by a multiset with cardinality 3, since only three different identifiers appear in it, namely $\{(2,0), (1,1), (1,1)\}$, where the tuple $(2,0)$ represents the identifier η, one of the $(1,1)$ represents η' and the other $(1,1)$ represents ν.

The order we are interested in for coverability is the following.

Definition 4.2 Let A and B be two elements in $\mathcal{MS}(\mathbb{N}^k)$. We will write $A \sqsubseteq B$ if there is an injective function $h : A \hookrightarrow B$ such that $a \leq h(a)$ for all $a \in A$, where $\leq$ is the punctual order in $\mathbb{N}^k$, that is, $(a_1, \ldots, a_k) \leq (b_1, \ldots, b_k)$ if and only if $a_i \leq b_i$ for every $i \in \{1, \ldots, k\}$.

Proposition 4.3 *The relation $\sqsubseteq$ on $\mathcal{MS}(\mathbb{N}^k)$ is a well quasi-order.*

Proof See [7] and [2]. □

Notice that the order itself takes into account α-conversion so that, for instance, in the case of a net with only two identifer places, the markings (α, β) and (β, α) are in the kernel of the order.

Lemma 4.4 *The α-equivalence relation $\equiv$ is the kernel of $\sqsubseteq$.*

Proposition 4.5 *The firing relation of Abstract MSPN systems is monotonic with respect to $\sqsubseteq$.*

Coverability of a marking M is just reachability of the ideal (upward closed set) generated by M, $\mathcal{C}(M) = \{M' \mid M \sqsubseteq M'\}$. Let us define the function *Pre* computing the predecessors of a set of states.

Definition 4.6 We define *Pre* as the function from markings to sets of markings defined by $Pre(M, loc) = \{(M', loc') \mid \exists t, \sigma (M, loc)[t(\sigma)\rangle(M', loc')\}$ and extend it trivially to sets of states.

In order to prove the following proposition, we need the following lemma stating that the transition relation is not only monotonic but injective when dealing with comparable markings.

Lemma 4.7 *If $M_1 \sqsubseteq M_2$, $M_1 \not\equiv M_2$ and $M_i[t(\sigma)\rangle M_i'$ for $i \in \{1,2\}$ then we have also $M_1' \not\equiv M_2'$ and $M_1' \sqsubseteq M_2'$.*

Proposition 4.8 *For each marking M the set $min(Pre(\mathcal{C}(M)))$ is computable, where min denotes the set of minimal elements of a set.*

Proof Let $Pre_{t,\sigma}(M) = \{M' \mid M'[t(\sigma)\rangle M\}$. By the previous lemma it holds that $min(Pre_{t,\sigma}(\mathcal{C}(M))) = Pre_{t,\sigma}(M)$, which is computable. Since $min(Pre(\mathcal{C}(M)))$

$= \bigcup min(Pre_{t,\sigma}(\mathcal{C}(M)))$ we only have to see that there are only finitely-many such t's and σ's. The only case which is not straightforward is when there exists t with $x \in I(t) \backslash O(t)$, that is, when the transition deletes a locality or an identifier token. If that variable is in a locality arc then we have to consider that any of the locality tokens appearing in the initial state may have been the one deleted. Regarding identifiers, the deleted token may be any of the ones appearing in M or any other, that we could arbitrarily choose, since we are working modulo α-conversion. In both cases, t can have been fired in a finite number of modes. □

By definition of well structured systems [1] we have the following

Corollary 4.9 *Abstract MSPN systems are well structured systems.*

As proved in [1], coverability (or control state reachability, as called there) is decidable for well structured systems. Indeed, without showing every detail, the infinite increasing chains of ideals generated by a well quasi order stabilize. That is, if $I_0 \subseteq I_1 \subseteq I_2 \subseteq \cdots$ is such a chain then there is k such that $I_i = I_k$ for every $k \geq i$. In particular, we can build the following chain of ideals: If M is the marking we want to cover, we take $I_0 = \mathcal{C}(M)$ and $I_{n+1} = I_n \cup Pre(I_n)$ for $n \geq 0$.

Since Pre is monotonic each I_i is an ideal and since $I_0 \subseteq I_1 \subseteq I_2 \subseteq \cdots$ and $\sqsubseteq$ is a well quasi-order it follows that there is an index k such that $I_l = I_k$ for all $l \geq k$. Then, by construction, I_0 is reachable (that is, M can be covered) if and only if the initial marking $M_0 \in I_k$.

Corollary 4.10 *Coverability is decidable for abstract MSPN systems.*

4.2 Coverability is decidable for MSPN systems

In this section we will prove that coverability is also decidable for non abstract MSPN systems. We cannot directly apply the well-quasi-order technique in this case, with numbers as identifiers, since a suitable order that does not depend on the particular marking to be covered, is not straightforward to obtain. But we can instead prove the decidability result by means of a reduction of the problem to the case of Abstract MSPN systems.

We could try to simulate the MSPN system by an abstract replica in which the *succ* transitions have been substituted by transitions labelled by *new*, and without counter places. However, abstract MSPN systems work modulo α-conversion of identifier tokens, so that a reachable abstract marking of the abstract net would correspond in this naive simulation to all the permutations of natural identifiers in the original MSPN system. Therefore, we need a way to track the order in which the abstract names are generated, to make them correspond to a single permutation. One can imagine that this would not be easy if we needed to remind the order during an arbitrary amount of time. Fortunately, given a marking M to cover, we will only need to remind the order for a finite time: exactly that corresponding to the highest identifier in the marking.

Let $\mathcal{N}$ be a MSPN system and M a marking of $\mathcal{N}$. Let us define an abstract MSPN system $\mathcal{N}^*$ in which coverability of M in $\mathcal{N}$ is reduced to coverability of some marking M^*. For every component net N_i of $\mathcal{N}$ we proceed as follows:

Let $n_i = max\{n \mid p \in P, (i,n) \in M(p)\}$, we define the component net N_i^* starting from N_i, removing its counter place, and changing label *succ* by *new*. Then we add a collection of places $p_1^i, \ldots, p_{n_i}^i$, that will be used to store the n_i first names created by N_i. To do so, we would have some auxiliary transitions that will fire in a row, after each one of the first n_i firings of transition *new*, filling each places p_j^i, with the j-th identifier created.

Then we define the marking M^* of the abstract system as follows: Locality and ordinary places are left as in M. Regarding identifier places, we consider a family of different new names $\{\eta_j^i \mid 1 \leq j \leq n_i\}$ and we define $M^*(p_j^i) = \{\eta_j^i\}$ and $M^*(q) = \{\eta_j^i \mid (i,j) \in M(q)\}$, where we are extending the usual notation for sets to multisets. Then we can prove the following

Proposition 4.11 *The marking M of $\mathcal{N}$ can be covered starting from M_0 if and only if the above defined marking M^* can be covered in $\mathcal{N}^*$ from the marking M_0^* that extends M_0 with $M_0^*(p_j^i) = \emptyset$, and the adequate values in the auxiliary places.*

Corollary 4.12 *Coverability is decidable for MSPN systems.*

5 Conclusions and Future Work

In this paper we have proved several results regarding expressiveness of MSPN systems with natural numbers as identifiers [10]. In particular, we have shown that reachability and coverability are decidable for MSPN systems, but submarking reachability is not, for the particular class of MSPN systems in which identifiers are basically unforgeable natural numbers generated by a counter mechanism. Although one could expect that submarking reachability should be easier to decide than reachability, the special restrictions constraining the use of the counter places, used to generate the identifiers, which cannot be either decreased or emptied, are responsible for this special feature of the model.

We have also considered a more abstract mechanism for the creation of identifiers, since if we use the successor function to generate the identifiers for authentication, then we have the possibility to check which identifiers were created before which, having also an upper bound on the number of identifers created. To avoid the possibility to distinguish two systems by means of these properties we now consider a *new* transition, that creates fresh names as in the π-calculus. Reachability in this case turns out to be undecidable, though coverability is still decidable, which is sufficient to check usual safety properties.

As future work, we plan to make our approach scalable. More exactly, we are investigating how, and under which hypothesis, we have to modify our name generation mechanism so that it also works in open environments. In addition, the coverability techniques seen so far can only be applied to closed systems. For instance, deciding coverability in the example shown in Section 2.2 only checks the system when the clients have that particular form, which is clearly insufficient to prove the safety of the system in an open setting, where the clients can have any unknown behaviour. We are currently developing an alternative version of our operational semantics that take into account any potential environment. Of course, in order to be able to prove properties, we have to assume that the environment does not know certain things about the system (secret keys, passwords,...), so that this semantics must be parametric with respect to that knowledge. In this new setting we can define the analogous problems of the original semantics, such as coverability. In order to make that semantics manageable we are defining an abstract version of that *open semantics*, that we can use to prove properties of the original semantics.

We also intend to study our model in the presence of an alternative primitive for interaction between components, namely broadcasting instead of synchronization, given that broadcasting is a widely used primitive, specially in the Ubiquitous Computing framework. Finally, we plan to extend our model with recursive or replication operators and study to what extent do the results obtained still hold.

References

[1] P. Aziz Abdulla, K. Cerans, and B. Jonsson. Algorithmic Analysis of Programs with Well Quasi-Ordered Domains. Inf. Comput., 160(1-2):109–127, 2000.

[2] P. Aziz Abdulla, and A. Nylén. Better is Better than Well. LICS'00: 132-140, 2000.

[3] J. Desel, and W. Reisig. *Place/transition petri nets.* Lectures on Petri Nets I: Basic Models, LNCS vol.1491, pp.122–173. Springer-Verlag, 1998.

[4] D. Frutos Escrig, F. Cuartero Gómez, and V. Valero Ruiz. On non-decidability of reachability for timed-arc Petri Nets. PNPM'99. 1999.

[5] D. Frutos Escrig, O. Marroquín Alonso and F. Rosa Velardo. *Ubiquitous Systems and Petri Nets.* Ubiquitous Web Systems and Intelligence, co-located with ICCSA 2005, LNCS vol.3841. Springer-Verlag, 2005.

[6] K. Jensen. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use..* Volume 1, Basic Concepts. Monographs in Theoretical Computer Science, Springer-Verlag, 2nd corrected printing 1997. ISBN: 3-540-60943-1.

[7] E. C. Milner. Basic wqo- and bqo-theory. In I. Rival, editor, *Graphs and Orders*, pages 487-502. 1985.

[8] R. Milner. *Theories for the Global Ubiquitous Computer.* Foundations of Software Science and Computation Structures-FoSSaCS 2004, LNCS vol.2987, pp.5-11. Springer-Verlag, 2004.

[9] C. Reutenauer. The Mathematics of Petri Nets. Masson and Prentice Hall, 1990.

[10] F. Rosa Velardo, D. Frutos Escrig and O. Marroquín Alonso. *Mobile Synchronizing Petri Nets: a choreographic approach for coordination in Ubiquitous Systems.* In 1st International Workshop on Methods and Tools for Coordinating Concurrent, Distributed and Mobile Systems, MTCoord'05. A Satellite Workshop of Coordination 2005. ENTCS (to appear).

[11] F. Stajano. Security for Ubiquitous Computing. John Wiley and Sons, 2002.

[12] M. Weiser. *Some Computer Science Issues in Ubiquitous Computing.* Comm. of the ACM vol.36(7), pp.74-84. ACM Press, 1993.

[13] M. Weiser. *The Computer for the 21st Century.* Proceedings of Human-computer Interaction: Toward the Year 2000, pp.933-940. Morgan Kaufmann Publishers Inc, 1995.