# Some Decidability Results for Nested Petri Nets *

Irina A. Lomazova[1] and Philippe Schnoebelen[2]

[1] Program Systems Institute of the Russian Academy of Science
Pereslavl-Zalessky, 152140, Russia
email: irina@univ.botik.ru

[2] Lab. Specification & Verification, ENS de Cachan & CNRS UMR 8643
61, av. Pdt. Wilson, 94235 Cachan Cedex, France
email: phs@lsv.ens-cachan.fr

**Abstract.** *Nested Petri nets* are Petri nets using other Petri nets as tokens, thereby allowing easy description of hierarchical systems. Their nested structure make some important verification problems undecidable (reachability, boundedness, ...) while some other problems remain decidable (termination, inevitability, ...).

## 1 Introduction

For modelling and analysis distributed concurrent systems, there exists a large variety of formalisms based on Petri nets [Rei85,Jen92,Smi96,Lom97]. Among them, several approaches extend the Petri nets formalism by notions and structures inspired from object oriented programming [Sib94,Lak95,MW97,Val98]. Such extensions are helpful for modelling *hierarchical* multi-agent distributed systems.

While Sibertin-Blanc [Sib94], Lakos [Lak95], Moldt and Wienberg [MW97] consider systems with communicating coloured Petri nets, Valk [Val98] in his *object Petri nets* considers tokens as objects with a net structure. In his approach, the system net and object nets are elementary net systems, but an object is in some sense not located in one place (since Valk uses object Petri nets for solving specific fork-join situations in task planning systems), and this leads to a rather complex definition of the notion of states for object Petri nets.

*Nested Petri nets.* Here we study another Petri net model where tokens may be nets themselves: *nested* [1] *Petri nets* [Lom98]. Nested Petri nets are a convenient

---

* This work was mainly prepared during the stay of the first author at Lab. Specification & Verification in June-July 1998, and was partly supported by INTAS-RFBR (Grant 95-0378) and the Russian Fund for Basic Research (Project No. 96-01-01717)

[1] The word "nested" points to the analogy with nested sets, containing sets as their elements, which in turn may contain sets and so on. There may be any fixed number of levels in nested Petri nets. It is also possible to consider nested nets with unbounded depth, but we do not do this here.

tool for modelling hierarchical multi-agent dynamic systems. The object nets in a nested Petri net have their own structure and behaviour, they may evolve and disappear during the system lifetime, and their number is unlimited. A nested Petri net has four kinds of steps. A *transfer step* is a step in a system net, which can "move", "generate", or "remove" objects, but does not change their inner states. An *object-autonomous* step changes only an inner state in one object. There are also two kinds of synchronisation steps. *Horizontal synchronisation* means simultaneous firing of two object nets, situated in the same place of a system net. *Vertical synchronisation* means simultaneous firing of a system net together with some of its objects involved in this firing.

In this paper we show how some crucial verification problems remain decidable for nested Petri nets and some become undecidable. This shows that nested Petri nets are in some weaker than Turing machines and stronger than ordinary, "flat" Petri nets. The decidability results are mostly based on the theory of Well-Structured Transition Systems [Fin90,AČJY96,FS98].

The paper is organised as follows. In section 2 we start with a simple example of a two-level nested Petri net with ordinary Petri nets as tokens. Section 3 contains definitions of nested Petri nets. In Section 4 the expressive power of nested Petri nets and some other Petri nets models is compared. In Section 5 we prove that nested Petri nets are well-structured transition systems and deduce some decidability and undecidability properties. Section 6 gives some conclusions and directions for further research.

## 2 An Introductory Example

To give the reader an intuitive idea of nested Petri nets we start with a small example of a two-level nested Petri net *NPN* represented in Fig. 1. It models a set of workers receiving some tasks from time to time. A worker's behaviour is described by an object (element) net *EN*. *EN* is an elementary Petri net. When a task comes, a worker is to borrow a tool from the buffer of tools. A buffer of tools is represented by a system net *SN*. It is a high-level Petri net with tokens of three types: black dots, tools (unstructured dots of some color) and workers (represented by nets).

The number of workers involved in this system is unlimited. The set $A$ of tools is fixed and is initially represented in the place $S_5$. In our example $A$ is finite (with $N$ elements).

Arcs in the system net *SN* are further labeled by expressions (variables and constants in our example), as in high-level Petri nets. If no expression is ascribed, the arc is supposed to transfer a black dot. In our *NPN* example, the arc expressions $x$ is a variable for a worker (having a marked net *EN* as its value), $y$ is a variable for a tool (a corresponding arc transfers a coloured dot for a tool), $\mathbf{W}_2$ is a constant for an element net *EN* with the marking $\{W_2\}$, i.e., having only one token in place $W_2$.

Some transitions are marked by labels $t_1, t_2, t_3, t_4$ in *EN* and $\overline{t_1}, \overline{t_2}, \overline{t_3}, \overline{t_4}$ in *SN*. They are used for synchronisation of transition firings in system and element

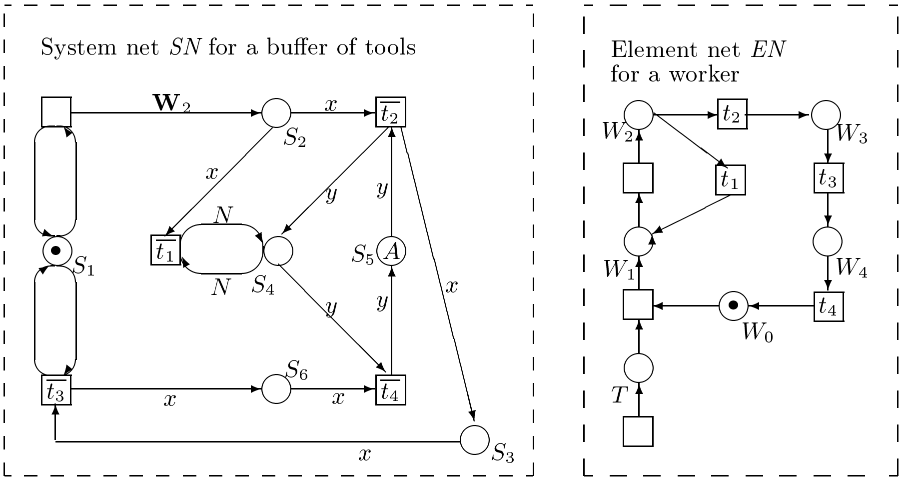


**Fig. 1.** *NPN*, a nested Petri net

nets. Thus transition marked by $\overline{t_2}$ in *SN* may only fire simultaneously with the firing of a transition marked by $t_2$ in the element net *EN* which is involved in firing $\overline{t_2}$ (i.e, which is transfered by it).

The substantive meaning of places in the element net *EN* is as follows: $T$ — there is a task for the worker; $W_0$ — the worker is idle; $W_1$ — the worker has got a task; $W_2$ — the worker is applying for a tool; $W_3$ — the worker is busy with a task; $W_3$ — the worker finished a task.

In the system net *SN* places are: $S_1$ — a buffer of tools is open; $S_2$ — workers, applying for a tool; $S_3$ — workers with tools; $S_4$ — the number of borrowed tools; $S_5$ — tools available; $S_6$ — workers, returning tools.

In the initial marking represented in Fig. 1 the system net *SN* (modelling a buffer of tools) contains a black dot token in the place $S_1$ (meaning the buffer is open for workers) and the set $A$ of $N$ tools in the place $S_5$, a net *EN* for a worker contains a token in the place $W_0$ (a worker is idle) and $T$ is empty, meaning there are no tasks for a worker. Note that initially there are no net tokens (workers) in *SN*, so the element net *EN* for a worker plays a role of type description.

To illustrate the behaviour of a nested Petri net we follow several possible steps of *NPN*. In the initial marking the unlabelled transition in *SN* may fire, putting a net token $\mathbf{W}_2$ (*EN* with marking $\{W_2\}$) into place $S_2$. This step creates an instance of *EN* in $S_2$. After that the transition marked by $\overline{t_2}$ in *SN* may fire synchronously with the transition marked by $t_2$ in the element net lying in $S_2$. After that the net *EN* with the marking $\{W_3\}$ will be situated in the place $S_3$, the set $A$ in $S_5$ will be diminished by one token and the place $S_4$ gets one token. Then the transition marked by $\overline{t_4}$ in *SN* may fire synchronously with transition marked by $t_4$ in the element net lying in $S_3$. Continuation of this process may lead to a marking shown in Fig. 2, where there is one worker applying for a

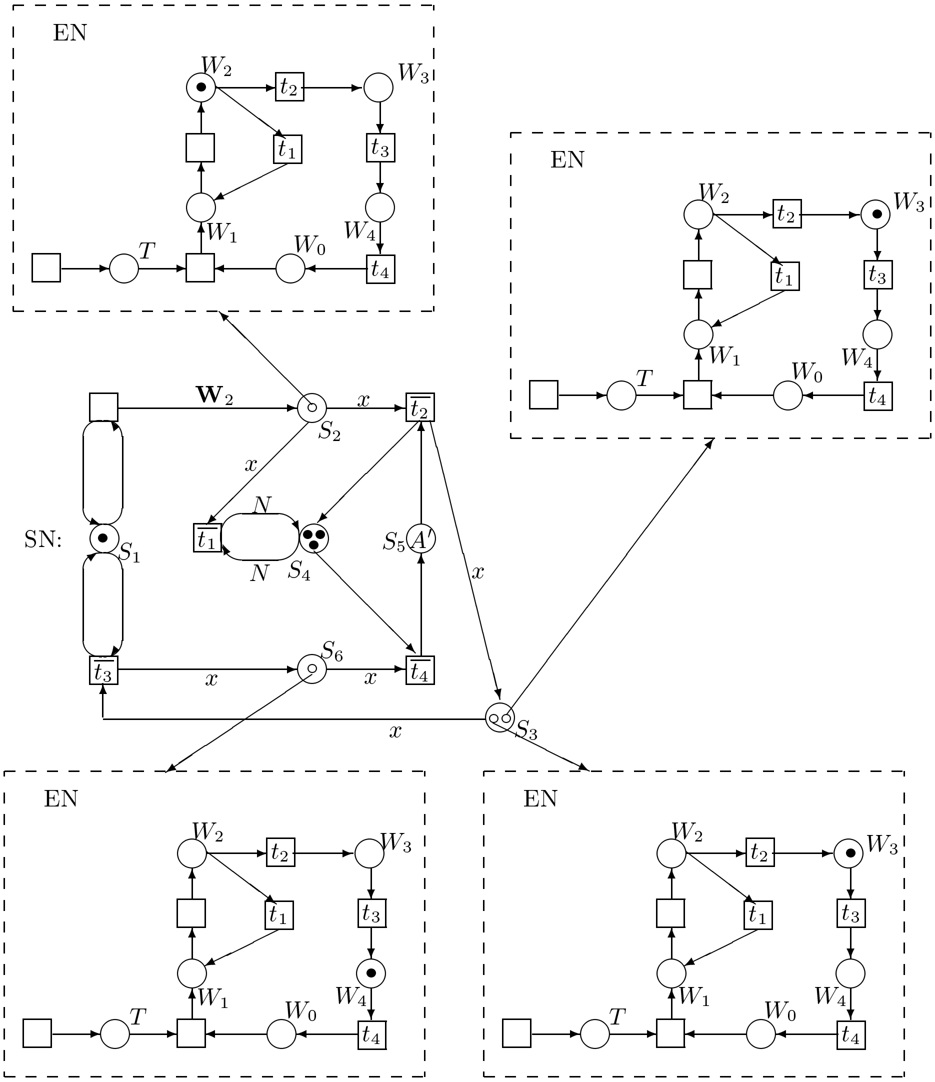


**Fig. 2.** An example of a reachable state for *NPN*

tool, two workers with tools and one worker has come to return a tool. Here $A'$ designates the set $A$ diminished by three tokens.

## 3 Nested Petri Nets

**Definition 3.1.** *Let $P$ and $T$ be disjoint sets and let $F \subseteq (P \times T) \cup (T \times P)$. Then $\mathcal{N} = (P, T, F)$ is a* net*. The elements of $P, T$ and $F$ are called* places, transitions *and* arcs *respectively.*

Pictorially, $P$-elements are represented by circles, $T$-elements by boxes, and the flow relation $F$ by directed arcs. For $x \in P \cup T$ we write ${}^\bullet x$ for the pre-set $\{y \mid yFx\}$ of $x$, and $x^\bullet$ for its post-set $\{y \mid xFy\}$. The *input arcs* of a transition $t$ are those in $\{(x,t) \mid x \in {}^\bullet t\}$, its *output arcs* are those in $\{(t,x) \mid x \in t^\bullet\}$.

*Markings.* In the Coloured Petri nets formalism [Jen92], places carry marked multisets of coloured tokens. Recall that a *multiset* $m$ over a set $S$ is a mapping $m : S \to \mathbb{N}$, where $\mathbb{N}$ is the set of natural numbers. $m$ is *finite* iff $\{s \in S \mid m(s) > 0\}$ is. We let $m \leq m'$ (resp. $m + m'$) denote multiset inclusion (resp. sum). By $S_{MS}$ we denote the set of all finite multisets over $S$.

**Definition 3.2.** *Let $\mathcal{N} = (P, T, F)$ be a net and $S$ an arbitrary set. A* marking *of $\mathcal{N}$ over $S$, also called an $S$-*marking*, is a function $M$ from $P$ to $S_{MS}$ mapping every place to a multiset over $S$. A* marked net *is a net together with some marking, called the* initial marking *of this net.*

In the above definition tokens may be arbitrarily complex objects (as in Coloured Petri nets). In nested Petri nets, tokens may be nets.

*Transitions.* As with Coloured Petri nets, we want to keep track of moving tokens. For this we label arcs with variables and other expressions.

Let $V = \{v_1, \ldots\}$ be a set of *variable* names, and $C = \{c_1, \ldots\}$ a set of *constant* names. Write $A$ for the set $V \cup C$ of *atoms*. An *expression* is a finite multiset of atoms (usually written with the binary symbol $+$: e.g., $v_1 + (c_2 + v_1)$ is an expression). $Expr(A)$ is another way of denoting $A_{MS} = \{e, \ldots\}$, the set of expressions. For $e \in Expr(A)$, $Var(e)$ is the set of variables occurring in expression $e$.

Assume any constant $c$ denotes a fixed element $c_S$ in $S$. Assume $b$ maps any variable $v$ to an element $b(v) \in S$. Then $b(e)$ denotes a multiset over $S$ in the obvious way.

Let $Lab = \{l_1, l_2, \ldots\}$ and $Lab' = \{l^1, l^2, \ldots\}$ be two disjoint sets of labels. For each label $l \in Lab \cup Lab'$ we define an *adjacent* label $\bar{l}$, such that the sets $Lab$, $Lab'$, $\overline{Lab} =_{\text{def}} \{\bar{l} \mid l \in Lab\}$ and $\overline{Lab'} =_{\text{def}} \{\overline{l'} \mid l' \in Lab'\}$ are pairwise disjoint. Let $\bar{\bar{l}} =_{\text{def}} l$ and $\mathcal{L} =_{\text{def}} Lab \cup Lab' \cup \overline{Lab} \cup \overline{Lab'}$.

Now we come to the definition of a nested Petri net structure, consisting of a system net, several element nets, labels on arcs, and labels on transitions.

**Definition 3.3.** *A* nested Petri net structure $\Sigma$ *is an array of* $k \geq 1$ *nets* $\mathcal{N}_1$, ..., $\mathcal{N}_k$, *where* $\mathcal{N}_1$ *is a distinguished net, called a* system net, *and the* $\mathcal{N}_i$*'s, for* $i = 2, \ldots, k$, *are called* element nets.

*In any* $\mathcal{N}_i = (P_i, T_i, F_i)$ *the input (resp. output) arcs from* $F_i$ *are labeled by expressions* $\mathcal{E}(p,t)$ *(resp.* $\mathcal{E}(t,p)$*) from* $Expr(A)$*. We require that there are no constants in input arc labels and no variable occurs twice in an input label* $\mathcal{E}(p,t)$*, or in two input labels for a same transition. (There is no restriction on the output labels.) Examples of forbidden arc inscriptions are shown in Fig. 3.*

*In any* $\mathcal{N}_i$*, the transitions may carry labels from* $\mathcal{L}$ *(possibly several labels).*

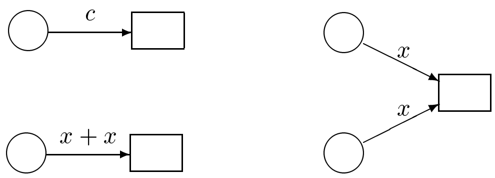


**Fig. 3.** Forbidden input arc inscriptions

Assume a given nested Petri net structure $\Sigma$ and let markings in element nets $\mathcal{N}_2, \ldots, \mathcal{N}_k$ be considered over some *finite* sets $S_2, \ldots, S_k$ correspondingly and let $\mathcal{M}$ denote the set of all marked element nets of $\Sigma$.

**Definition 3.4.** *A* nested Petri net *(NP-net) is a nested Petri net structure* $\Sigma$ *with each constant* $c \in C$ *interpreted as some marked element net from* $\mathcal{M}$.

*By a* marking *of a NP-net, we mean a marking of its system net over the set* $\mathcal{M}$.

*A* marked NP-net *is an NP-net together with some (initial) marking.*

Note that the definition of an NP-net depends on the sets $S_2, \ldots, S_k$ as parameters. If the $S_i$'s are one-element sets, then the object nets are ordinary Petri nets with black dots as tokens, and a nested Petri net is just a system net with ordinary nets as tokens. If the $S_i$'s are sets of coloured tokens, then a nested Petri net has Coloured Petri nets as tokens. If the $S_i$'s are sets of marked nets, we get a three-levels (or more) structure, in which element nets are system nets with respect to the next level. It's clear that we can have as many levels as we like. And at last, if some of sets $S_2, S_3, \ldots, S_k$ contain the system net $\mathcal{N}_1$, as its element we get recursion, which is not considered here.

In NP-nets, firing a transition requires instantiating the variables in arc labels:

**Definition 3.5.** *Let* $\mathcal{N}_i = (P_i, T_i, F_i)$ *be a net in a nested net NPN.*

1. *A* binding *of a transition* $t \in T_i$ *is a function* $b$ *mapping each variable* $v \in V$ *to a value* $b(v)$ *from the set* $\mathcal{M} \cup S_2 \cup S_3 \cup \ldots \cup S_k$.

2. *A* binded transition *is a pair* $Y = (t, b)$*, where* $t$ *is a transition and* $b$ *is a binding of* $t$*.*
3. *A binded transition* $Y = (t, b)$ *is* enabled *in a marking* $M$ *of* $\mathcal{N}_i$ *iff* $\forall p \in {}^\bullet t$ : $b(\mathcal{E}(p,t)) \subseteq M(p)$.
4. *An enabled binded transition* $Y = (t, b)$ *may fire in a marking* $M$ *and yield a new marking* $M'$*, written* $M[Y\rangle M'$*. For any* $p \in P_i$, $M'(p) =_{def} M(p) - b(\mathcal{E}(p,t)) + b(\mathcal{E}(t,p))$.
5. *For marked element nets (except black dot tokens), which serve as variable values in input arc expressions from* $\mathcal{E}(t)$*, we say, that they are* involved *in firing of* $t$*. (They are removed from input places and may be brought to output places of* $t$*).*

Now we come to defining a step in a NP-net.

**Definition 3.6.** *Let NPN be an NP-net. A step of NPN is either*

**a transport step:** *firing (through some appropriate binding) an unlabeled transition in the system net* $\mathcal{N}_1$*, not changing markings of element nets;*

**an object-autonomous step:** *firing an unlabeled transition in one of the element nets, while all element nets remain in the same places of the system net;*

**an horizontal synchronisation step:** *simultaneous firing of two transitions of two element nets lying in the same place w.r.t. the same binding, provided these two transitions are marked by two adjacent labels* $l$ *and* $\bar{l}$ *from* $Lab' \cup \overline{Lab'}$*;*

**a vertical synchronisation step:** *simultaneous firing of a transition* $t$ *marked by a label* $l \in Lab \cup \overline{Lab}$ *in the system net and transitions marked by the adjacent label* $\bar{l}$ *in element nets involved in firing of* $t$*.*

We say a marking $M'$ is (directly) reachable from a marking $M$ and write $M \to M'$, if there is a step in *NPN* leading from $M$ to $M'$.

An execution of NP-net *NPN* is a sequence of markings $M_0 \to M_1 \to M_2 \ldots$ successively reachable from the initial marking $M_0$.

## 4 Nested Petri nets and other Petri net models

In this section we compare expressive power of nested Petri nets with some other Petri net models. First of all, since tokens in a system net may be just black dots, we immediately get

**Proposition 4.1.** *Ordinary Petri nets form a special case of nested Petri nets.*

Then we compare nested Petri nets with some extensions of ordinary Petri net model.

*Petri nets with reset arcs* [Cia94] extend the basic model with special "reset" arcs, which denote that firing of some transitions resets (empties) the corresponding places.

**Theorem 4.2.** *Petri nets with reset arcs can be simulated by nested Petri nets with ordinary Petri nets as object nets.*

*Proof.* The idea is to simulate the presence of $n$ tokens in a place by one simple element net having $n$ tokens. Then it is possible simulate the effect of a reset arc by removing this net token in one step, replacing it with $E0$, a constant net with zero tokens. Incrementations and decrementations of tokens in a place are simulated by incrementations and decrementations of tokens in the corresponding element nets. They are enforced by the synchronisation mechanism.

Fig. 4(a) shows a fragment of a Petri net with $n$ tokens in a place $p$, incrementing (for $p$) arc $(t_+, p)$ and decrementing arc $(t_-, p)$. Fig. 4(b) represents a fragment of a NP-net, simulating it. Here $n$ tokens in $p$ are replaced by one net token $EN$, which has one place with $n$ black dot tokens and two transitions marked by $\overline{l_+}$ and $\overline{l_-}$, which add or remove a token to/from $p$. These transitions fire synchronously with transitions $t_+$, or $t_-$ respectively in a system net. Transition $t_r$ in a system net removes a et token from $p$, thus emptying it. $\square$

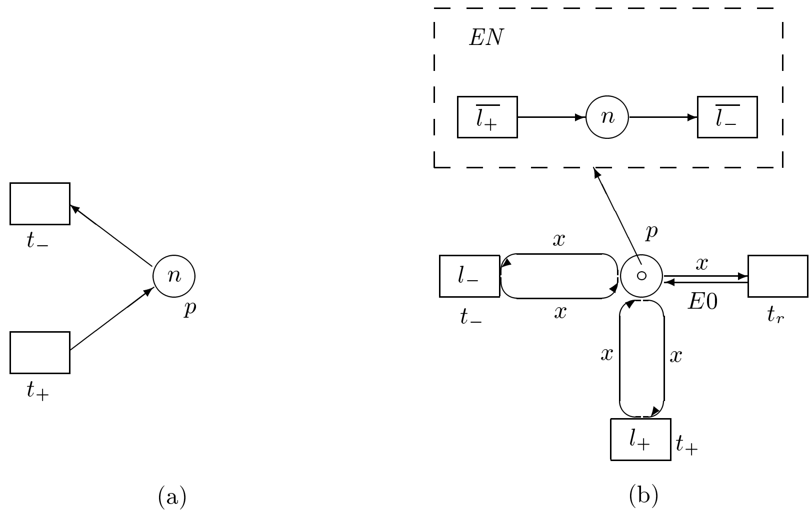


**Fig. 4.** Simulation of a reset arc

Since it is known [DFS98,DJS99] that Petri nets with reset arcs are more expressive than ordinary Petri nets, we immediately get the following

**Theorem 4.3.** *Nested Petri nets with ordinary Petri nets as object nets are more expressive than "flat" ordinary Petri nets.*

## 5 Decidability for Nested Petri Nets

In this section we discuss some issues of decidability for nested Petri nets. First, we briefly formulate some problems crucial for verification of Petri nets.

A net terminates if there exists no infinite execution (*Termination Problem*). A marking $M'$ is reachable from $M$, if there exists a sequence of steps leading from $M$ to $M'$ (*Reachability Problem*). The reachability set of a net is the set of all markings reachable from the initial marking. A net is bounded if its reachability set is finite (*Boundedness Problem*). The *Control-State Maintainability Problem* is to decide, given an initial marking $M$ and a finite set $Q = \{q_1, q_2, \ldots, q_m\}$ of markings, whether there exists a computation starting from $M$ where all markings cover (are not less than w.r.t. some ordering) one of the $q_i$'s. The dual problem, called the *Inevitability Problem*, is to decide whether all computations starting from $M$ eventually visit a state not covering one of the $q_i$'s, e.g. for Petri nets we can ask whether a given place will eventually be emptied.

Since NP-nets simulate Petri nets with reset arcs, problems undecidable for Petri nets with reset arcs are also undecidable for NP-nets.

**Theorem 5.1.** *1. Reachability is undecidable for nested Petri nets.*
*2. Boundedness is undecidable for nested Petri nets.*

*Proof.* Due to Theorem 4.2 nested Petri nets can simulate Petri nets with reset arcs, hence, validity of this two statements follows from undecidability of reachability [AK77] and boundedness [DFS98,DJS99] for Petri nets with reset arcs. □

To obtain decidability results we use the notion of well-structured transition system introduced in [Fin90,AČJY96]. Recall that a transition system is a pair $\mathcal{S} = \langle S, \rightarrow \rangle$ where $S$ is an abstract set of states (or configurations) and $\rightarrow \subseteq S \times S$ is any transition relation. For a transition system $\mathcal{S} = \langle S, \rightarrow \rangle$ we write $Succ(s)$ for the set $\{s' \in S \mid s' \rightarrow s\}$ of immediate successors of $s$. $\mathcal{S}$ is finitely branching if all $Succ(s)$ are finite.

A well-structured transition system is a transition system with a compatible wqo: recall that a quasi-ordering (a qo) is any reflexive and transitive relation $\leq$ (over some set $X$).

**Definition 5.2.** *A* well-quasi-ordering *(a wqo) is any quasi-ordering $\leq$ such that, for any infinite sequence $x_0, x_1, x_2, \ldots,$ in $X$, there exist indexes $i < j$ with $x_i \leq x_j$.*

Note, that if $\leq$ is a wqo, then any infinite sequence contains an infinite increasing subsequence: $x_{i_0} \leq x_{i_1} \leq x_{i_2} \ldots$

**Definition 5.3.** *A well-structured transition system (a WSTS) is a transition system $\Sigma = \langle S, \rightarrow, \leq \rangle$ equipped with an ordering $\leq \subseteq S \times S$ between states such that*
*– $\leq$ is a wqo, and*
*– $\leq$ is "compatible" with $\rightarrow$,*
*where "compatible" means that for all $s_1 \leq t_1$, and transition $s_1 \rightarrow s_2$, there exists a transition $t_1 \rightarrow t_2$, such that $s_2 \leq t_2$.*

[FS98,FS97] introduce more liberal notions of compatibility:

A WSTS $\Sigma$ has *transitive compatibility* if for all $s_1 \leq t_1$, and transition $s_1 \to s_2$, there exists a nonempty sequence $t_1 \to t_2 \to \ldots \to t_n$ with $s_2 \leq t_n$.

A WSTS $\Sigma$ has *stuttering compatibility* if for all $s_1 \leq t_1$, and transition $s_1 \to s_2$, there exists a nonempty sequence $t_1 \to t_2 \to \ldots \to t_n$ with $s_2 \leq t_n$ and $s_1 \leq t_i$ for all $i < n$.

Now we define a wqo on the set of states of our NP-nets and show that they are WSTS.

**Definition 5.4.** *Let NPN be a nested Petri net, $\mathcal{M}_{MS}$ — the set of all its states.*

*A quasi-ordering $\preceq$ on $\mathcal{M}_{MS}$ is defined as follows:*
*for $M_1, M_2 \in \mathcal{M}_{MS}$: $M_1 \preceq M_2$ iff for all $p \in P_{\mathcal{N}_1}$ there exists an injective function $\jmath_p : M_1(p) \to M_2(p)$, such that $\forall \langle \mathcal{N}_i, m\rangle \in M_1(p)$, for $s \in M_1(p)$: either $\jmath_p(s) = s$ or $s = \langle \mathcal{N}_i, m\rangle$ and $\jmath_p(\langle \mathcal{N}_i, m\rangle) = \langle \mathcal{N}_j, m'\rangle$ implies $m \preceq m'$.*

Fig. 5 shows an example of two markings $M_1, M_2$ of some NP-net, ordered w.r.t. $\preceq$. Here the system net has three places $p_1, p_2, p_3$. The only element net has places $q_1, q_2$. In both markings the place $p_1$ is empty and the place $p_2$ contains one net token, but the marking of this net token in $M_1$ is included in the corresponding marking in $M_2$. The place $p_3$ in $M_2$ contains the same net token as in in $M_1$ plus one more net token. Thus, the relation $\preceq$ is a kind of a nested set inclusion.

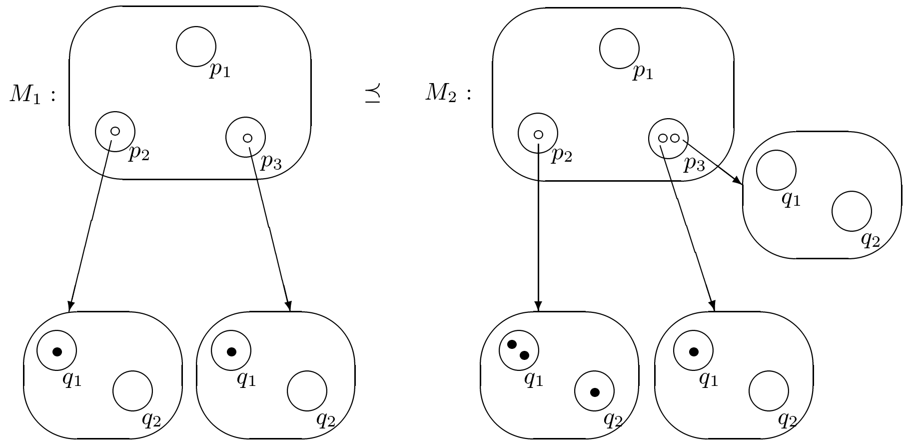


**Fig. 5.** An example of markings $M_1 \preceq M_2$

**Proposition 5.5.** *Let NPN be a nested Petri net, with $\mathcal{M}_{MS}$ the set of all its states, $\to$ the step relation on $\mathcal{M}_{MS}$, and $\preceq$ the quasi-ordering on $\mathcal{M}_{MS}$, defined above. Then $\langle \mathcal{M}_{MS}, \to, \preceq\rangle$ is a well-structured transition system.*

*Proof.* $\preceq$ is clearly a well quasi-ordering but we must show that it is compatible with the transition relation $\to$. We have four cases.

1. Let $M_1 \xrightarrow{t} M_1'$ be a transport step in a nested net via a transition $t$ and let $M_1 \preceq M_2$. Then for every token $s \in M_1(p)$ transferred by $t$ (with $p \in {}^\bullet t$) there exists an object $\jmath_p(s) \in M_2(p)$. Since due to the restriction on input expressions all objects are transferred independently and firing of $t$ doesn't depend actually on object markings, the transition $t$ is enabled also in $M_2$. It is easy to see, that if $M_2 \xrightarrow{t} M_2'$, then $M_1' \preceq M_2'$.
2. For an object-autonomous step compatibility is obvious.
3. A horizontal synchronisation step is a simultaneous execution of several object-autonomous steps. Its compatibility can be proved analogously to the previous case.
4. A vertical synchronisation step is a simultaneous execution of a transport and several object-autonomous steps. Its compatibility is not a direct implication from the two first cases, but it can be proved by combining previous proofs. $\square$

Note that if we would not restrict multiple occurrences of variables in input arc expressions, we would not have WSTS, as well as Object Petri nets of Valk are not WSTS.

It was proved in [FS97] that

- Termination is decidable for WSTS's with (1) transitive compatibility, (2) decidable $\leq$, and (3) effective $Succ(s)$. (Theorem 4.6.)
- The control-state maintainability problem and the inevitability problem are decidable for WSTS's with (1) stuttering compatibility, (2) decidable $\leq$, and (3) effective $Succ(s)$. (Theorem 4.8.)

It turns out that for NP-nets

**Lemma 5.6.** *(1). The qo $\preceq$ is decidable.*
*(2). Succ is effective.*

With the help of these statements we can obtain the following decidability results for NP-nets:

**Theorem 5.7.** *Termination is decidable for nested Petri nets.*

*Proof.* Follows from Proposition 5.5, Lemma 5.6 and Theorem 4.6 in [FS97]. $\square$

**Corollary 5.8.** *Nested Petri nets are expressively strictly weaker than Turing machines.*

*Proof.* Since termination is not decidable for Turing machines. $\square$

**Theorem 5.9.** *The control-state maintainability problem and the inevitability problem (w.r.t. $\preceq$) are decidable for nested Petri nets.*

*Proof.* Follows from Proposition 5.5, Lemma 5.6 and Theorem 4.8 in [FS97]. $\square$

## 6 Concluding remarks

Nested Petri nets are an extension of the Petri nets formalism which gives visual and clear dynamic hierarchical and modular structure of the system. The synchronization of hierarchical components is natural and powerful.

The structure of nested Petri nets gives a good intuition of its distributed behaviour. Though we have only defined here an interleaving semantics, it can be naturally generalised to simultaneous or independent firings. With two kinds of synchronisation: horizontal for cooperation of elements and vertical for coordination of system and its elements nested Petri nets formalism can be considered as a kind of generalisation of module Petri nets (see, e.g., [CP92,Lom97a]) and hierarchical Petri nets (e.g., [Jen92]) models.

Thus, nested Petri nets turns out to be a visual and expressive tool for modelling multi-agent distributed systems. At the same time, decidability of such important properties as termination gives ground for solving some verification problems for them. Being still less expressive than Turing machines, nested Petri nets preserve merits of Petri nets model.

Further research on nested Petri nets supposes investigation of recursive nested Petri nets, when a system net contains its own copy as its element (directly or via other elements) and decidability questions for them.

## References


[AČJY96] P. A. Abdulla, K. Čerāns, B. Jonsson, and T. Yih-Kuen. General decidability theorems for infinite-state systems. In *Proc. 11th IEEE Symp. Logic in Computer Science (LICS'96), New Brunswick, NJ, USA, July 1996*, pages 313–321, 1996.

[AK77] T. Araki and T. Kasami. Some decision problems related to the reachability problem for Petri nets. *Theoretical Computer Science*, 3(1):85–104, 1977.

[CP92] S. Christensen and L. Petrucci. Towards a modular analysis of coloured petri nets. In *Proc. 13th Int. Conf. Application and Theory of Petri Nets, Sheffield, UK, June 1992*, volume 616 of *Lecture Notes in Computer Science*, pages 113–133. Springer, 1992.

[Cia94] G. Ciardo. Petri nets with marking-dependent arc cardinality: Properties and analysis. In *Proc. 15th Int. Conf. Application and Theory of Petri Nets, Zaragoza, Spain, June 1994*, volume 815 of *Lecture Notes in Computer Science*, pages 179–198. Springer, 1994.

[DFS98] C. Dufourd, A. Finkel, and Ph. Schnoebelen. Reset nets between decidability and undecidability. In *Proc. 25th Int. Coll. Automata, Languages, and Programming (ICALP'98), Aalborg, Denmark, July 1998*, volume 1443 of *Lecture Notes in Computer Science*, pages 103–115. Springer, 1998.

[DJS99] C. Dufourd, P. Jančar, and Ph. Schnoebelen. Boundedness of Reset P/T nets. In *Proc. 26th Int. Coll. Automata, Languages, and Programming (ICALP'99), Prague, Czech Republic, July 1999*, volume 1644 of *Lecture Notes in Computer Science*, pages 301–310. Springer, 1999.

[Fin90] A. Finkel. Reduction and covering of infinite reachability trees. *Information and Computation*, 89(2):144–179, 1990.

[FS97] A. Finkel and Ph. Schnoebelen. Well-structured transition systems everywhere ! Accepted for publication in Theor. Comp. Sci., October 1997.

[FS98] A. Finkel and Ph. Schnoebelen. Fundamental structures in well-structured infinite transition systems. In *Proc. 3rd Latin American Theoretical Informatics Symposium (LATIN'98), Campinas, Brazil, Apr. 1998*, volume 1380 of *Lecture Notes in Computer Science*, pages 102–118. Springer, 1998.

[Jen92] K. Jensen. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use. Vol. 1, Basic Concepts.* EATCS Monographs on Theoretical Computer Science. Springer, 1992.

[Lak95] C. Lakos. From coloured Petri nets to object Petri nets. In *Proc. 16th Int. Conf. Application and Theory of Petri Nets, Turin, Italy, June 1995*, volume 935 of *Lecture Notes in Computer Science*, pages 278–297. Springer, 1995.

[Lom97] I. A. Lomazova. Multi-agent systems and In *Proc. Int. Workshop on Distributed Artificial Intelligence and Multi-Agent Systems (DAIMAS'97), St. Petersburg, Russia, June 1997*, pages 147–152, 1997.

[Lom97a] I. A. Lomazova On Proving Large Distributed Systems: Petri Net Modules Verification. In *Proc. 4th Int. Conf. Parallel Computing Technologies (PaCT'97), Yaroslavl, Russia, Sep. 1997*, volume 1277 of *Lecture Notes in Computer Science*, pages 70–75. Springer, 1997.

[Lom98] I. A. Lomazova. Modelling of multi-agent dynamic systems by nested Petri nets (in Russian), August 1998. In *Programmnye Sistemy: Teoreticheskie Osnovy i Prilozheniya,* pages 143–156, Moscow: Nauka, 1999.

[MW97] D. Moldt and F. Wienberg. Multi-agent-systems based on coloured Petri nets. In *Proc. 18th Int. Conf. Application and Theory of Petri Nets, Toulouse, France, June 1997*, volume 1248 of *Lecture Notes in Computer Science*, pages 82–101. Springer, 1997.

[Rei85] W. Reisig. *Petri Nets. An Introduction*, volume 4 of *EATCS Monographs on Theoretical Computer Science*. Springer, 1985.

[Sib94] C. Sibertin-Blanc. Cooperative nets. In *Proc. 15th Int. Conf. Application and Theory of Petri Nets, Zaragoza, Spain, June 1994*, volume 815 of *Lecture Notes in Computer Science*, pages 471–490. Springer, 1994.

[Smi96] E. Smith. A survey on high-level Petri-net theory. *EATCS Bull.*, 59:267–293, June 1996.

[Val98] R. Valk. Petri nets as token objects: An introduction to elementary object nets. In *Proc. 19th Int. Conf. Application and Theory of Petri Nets, Lisbon, Portugal, June 1998*, volume 1420 of *Lecture Notes in Computer Science*, pages 1–25. Springer, 1998.