Compositionality of some behavioral properties for free-choice Nested Petri Nets [⋆]

Leonid V. Dvoryansky[1] and Irina A. Lomazova[2]

[1] ISP RAS, Moscow, Russia
leo@mathtech.ru
[2] Higher School of Economics, Moscow, Russia
i_lomazova@mail.ru

Abstract. Nested Petri nets are Petri nets with net tokens. The liveness and boundedness problems are undecidable for two-level Nested Petri nets (NP-nets) [1]. Boundedness is EXPSPACE and liveness is EXPSPACE or worse for plain Petri nets [5]. For the restricted class of free-choice Petri nets some problems become more amenable to analysis. There is the polynomial time algorithm to check if a free-choice Petri net is live and bounded [3]. In this paper we show how these results can be applied to NP-nets with free-choice components. We prove, that for NP-nets boundedness can be checked in a compositional way and define restrictions, under which liveness is also compositional. The motivating example of such restricted NP-nets is provided.

1 Introduction

Nested Petri nets (NP-nets) [1] is an extension of high-level Petri nets according to the "nets-within-nets" approach. A NP-net is a high-level Petri net with net-tokens being Petri nets themselves. NP-nets is a convenient formalism for modeling systems of dynamic interacting agents: an agent is represented by a net token, and all agents are distributed in a system net. Levels in NP-nets are coordinated via synchronized transitions (simultaneous firing of transitions in adjacent levels of the model). Being less than Turing power NP-nets are stronger more expressive than Petri nets. Thus the liveness and boundedness problems are undecidable for two-level NP-nets [1].

Though boundedness and liveness is decidable for plain Petri nets, these problems are not tractable in general. Boundedness is EXPSPACE and liveness is EXPSPACE or worse for Petri nets [5]. So, analysis of Petri nets subclasses, for which these problems can be decided effectively, is of great importance. Free-choice Petri nets is such a popular subclass. Well-formedness (liveness and boundedness) can be checked for free-choice Petri nets in polynomial time [3].

In this paper we study how these results can be applied to NP-nets with free-choice components. These restrictions are not something artificial: we can

[⋆] This research is partly supported by Russian Fund for Basic Research (projects 09-01-00277, 11-01-00737) and Federal Program "Kadry" (project 02.740.11.0207).

meet systems with free-choice components in different application areas. A small example of such a system is provided.

For a NP-net with free-choice components we can effectively check well-formedness of its separate components. Then we define conditions for NP-nets, under which boundedness and liveness of an "entire" NP-net can be deduced from the corresponding properties of its components (compositionality).

2 Preliminary definitions

A Petri net (PN) is a bipartite graph $N = \langle P, T, F\rangle$ with a finite set of nodes $P \cup T$, where $P \cap T = \emptyset$, and an incidence relation $F \subseteq (P \times T) \cup (T \times P)$. An incidence relation F induces the incidence function $F : ((P \times T) \cup (T \times P)) \to \mathbb{N}$ such that $f(x, y) = 1$ iff $(x, y) \in F$, and $F(x, y) = 0$ otherwise. Nodes from P are called *places*, they correspond to local states of a system. Nodes from T are called *transitions* and correspond to actions or events. A *marking* in a PN is a function $m : P \to \mathbb{N}$, mapping each place to some natural number. P-elements are represented by circles, T-elements – by boxes, and a flow relation – by directed arcs. Places may carry tokens represented by filled circles. A current marking m is designated by putting $m(p)$ tokens into each place $p \in P$.

For a transition $t \in T$ an arc (x, t) is called an *input arc*, and an arc (t, x) — an *output arc*; the *preset* ${}^\bullet x$ and the *postset* $x^\bullet$ are subsets of $P \cup T$ such that ${}^\bullet x = \{y|(y, x) \in F\}$ and $x^\bullet = \{y|(x, y) \in F\}$. A transition $t \in T$ is *enabled* in a marking m iff $\forall p \in {}^\bullet t : m(p) \geq 1$. An enabled transition t may *fire* yielding a new marking $m' =_{\text{def}} m - {}^\bullet t + t^\bullet$, i.e. $m'(p) = m(p) - F(p, t) + F(t, p)$ for each $p \in P$ (denoted $m \xrightarrow{t} m'$). The set of all markings reachable from a marking m (via a sequence of firings) is denoted by $R(m)$.

Definition 1. *A net $N = \langle P, T, F\rangle$ is called free-choice, iff $\forall p \in P : \forall t \in T : \langle p, t\rangle \in F$ implies ${}^\bullet t \times p^\bullet \subseteq F$.*

Further we use the term "free-choice" instead of "extended free-choice" following the convention from [3]. Informally, for a free-choice Petri net it holds that, for any position p if any of its outgoing transitions is active, then each of its other outgoing transitions is also active, so the system can "freely choose" between them.

In *two-level nested Petri nets (NP-nets)* [1, 2] tokens may be PNs themselves. A NP-net consists of a *system net* and *element nets*. Marked element nets are called *net tokens*.

A system net in a NP-net is a high level PN with expressions on arcs. In arc expressions we use *variables* and *constants*, which we call *atoms*. An expression is a sum (multiset) of atoms. Such expressions form the simple additive language $Expr$.

Definition 2. *A (two-level) NP-net is a tuple $NP = \langle Lab, SN, \mathbb{E}\rangle$. Here Lab is a set of labels for synchronization of transitions, $\mathbb{E}$ – a finite set of element nets. Some transitions in element nets may be labeled with labels from Lab.*

A system net $SN = \langle N, \mathcal{L}, \mathcal{U}, W, m_0 \rangle$ is a high-level Petri net, where $N = \langle P_{SN}, T_{SN}, F_{SN} \rangle$ is a net with P_{SN} – a set of typed places (a type is either an atomic type, or an element net from $\mathbb{E}$); $\mathcal{L} = Expr$ – an arc expression language (sums of constants and variables); $\mathcal{U} = \langle A, \mathcal{I} \rangle$ – a model of $\mathcal{L}$ with a domain $A = A_{net} \cup A_{atom}$, where A_{net} – a set of marked element nets (net tokens), A_{atom} — a set of plain colored tokens, $\mathcal{I} : Con \to A$ – an interpretation function; $W : F_{SN} \to \mathcal{L}$ – an arc expression function; $\Lambda : T_{SN} \to Lab$ — a transition labeling function.

According to NP-nets definition constants or several instances of the same variable are not allowed in input arc expressions. In a transition firing each variable in the transition input and output arc expressions is assigned to some colored or net token.

For further details see [1, 2]. Note, however, that here we consider a typed variant of NP-nets, when a type is instantiated to each place, and due to syntactical limitations this place may contain only tokens of this type.

Let NP be a NP-net. Then a system net of NP and its element nets are called *components* of NP.

3 Motivating example

In this section we give an example of a NP-net NP, modeling a conceptual part of P2P protocol. Here the system net SN (fig.2) models the interaction protocol itself and the element nets (fig.1) model behaviour of seeds E_1 providing datum, peers E_2 seeking for datum and pipes E_3 – channels for a secure data transfer.

Seeds and peers are modeled roughly as we are interested in their interactions. A seed E_1 can be in one of “ready to upload” p_1, “uploading” p_2 and “reinitialization” p_3 states. Seed events are “upload started” t_1, “upload complete” t_2 and “reinitialization complete” t_3. A peer E_2 can be in “ready to download” p_4, or in “downloading” p_5 states. Peer events are “download started” t_4 and “download complete” t_5.

A pipe is a connection which can be used to change/improve such aspects of data transfer as security, reliability etc. A pipe E_3 has a seed (p_6,p_7) and a peer (p_{12},p_{13}) interfaces and a 2-cells buffer (p_8,p_9,p_{10},p_{11}). Initially the pipe is waiting for the seed (p_7). The seed can start (t_6) to transfer data (p_6) to the buffer and the peer starts waiting for the datum portion. When a transmission is finished (t_7), the seed returns to the wait state (p_7) and the first cell of buffer is filled (p_8). Then a data portion propagates (t_8) to the second cell of the buffer, thus the first cell becomes empty (p_9) and the second cell becomes filled (p_{10}). When the peer has downloaded (t_9) the data portion from the second cell (p_{10}) the second buffer cell becomes empty (p_{11}) and the peer waits for another portion of data (p_{13}). When the seed has sent all datum, the pipe waits for a data portion to be propagated through the buffer to the peer (p_{13}) and then moves (t_{12}) to reinitialization state ($p14$). When reinitialization is finished (t_{10}) the pipe completes (t_{10}) a transfer cycle and becomes ready for another cycle (p_{15}). Now the next transfer cycle can start (t_{11}).

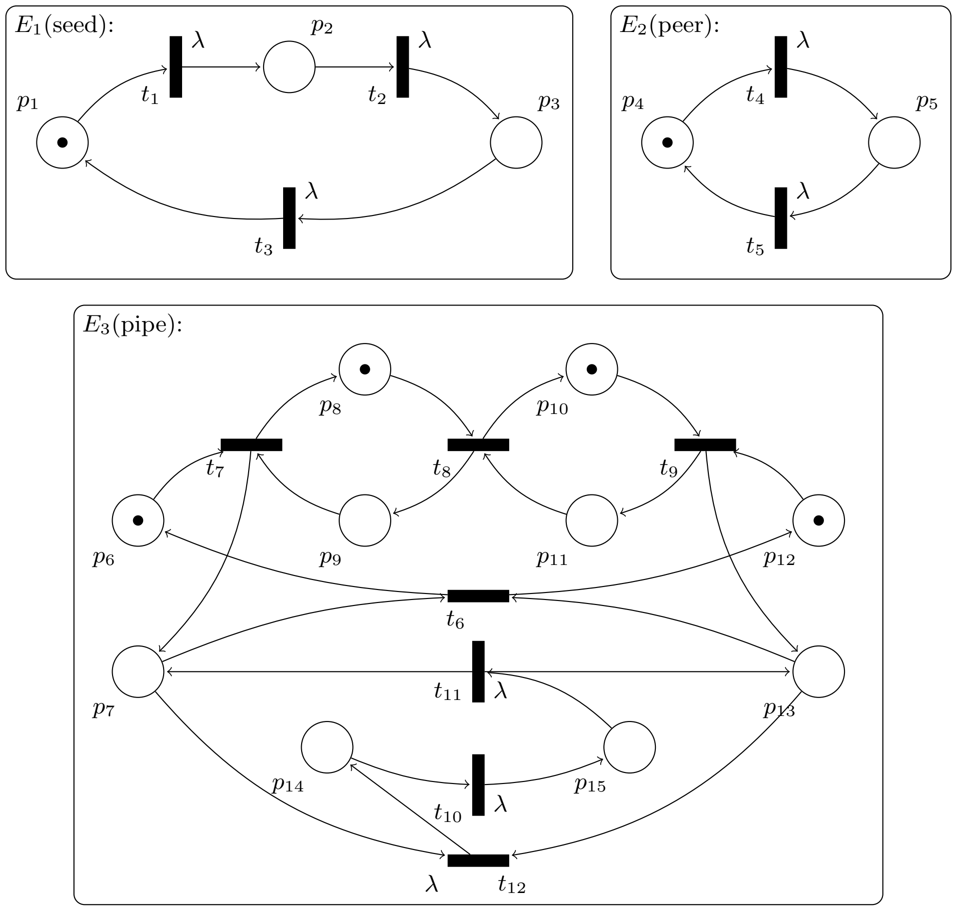

Fig. 1. The element nets of NP.

The system consists of initial and final pools for seeds, peers and pipes. In the initial pool an agent is ready for interaction, while in the final pool an agent reinitializes its inner structures for the next interaction activity. There is no final pool for peers, since we do not model peer initialization routines. The initial pools for seeds (p_{16}), peers (p_{17}) and available pipes (p_{18}) are filled with some agents in the initial state of the system. If there are a seed and a peer which are ready to transfer data and there is an available session, then they can start (t_{14}) interaction (p_{19},p_{20},p_{21}) controlled by the pipe net (E_3). After the seed and the peer have finished (t_{16}) their data exchange, the peer returns to the initial pool (p_{17}), the seed and the session get to their final pools (p_{22}, p_{23}). After reinitializing a pipe returns (t_{17}) to the initial pool of pipes (p_{18}) and a peer returns (t_{15}) to its initial pool (p_{16}).

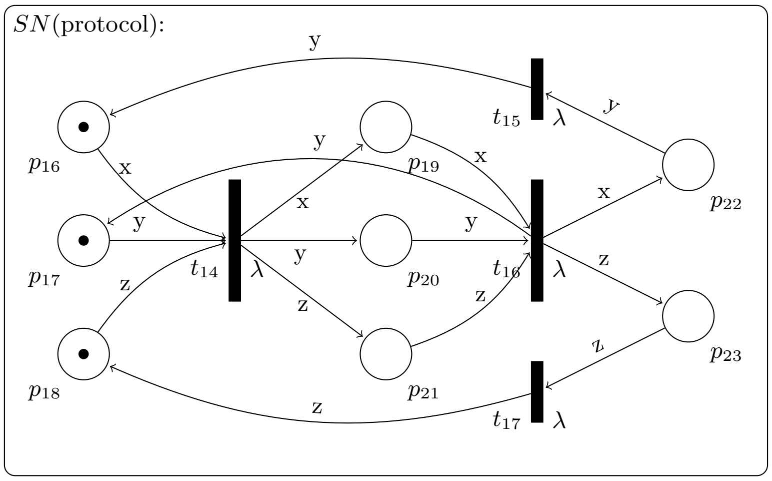

Fig. 2. The system net of NP.

This is certainly only the core of the protocol, since such aspects as system termination, or exceptions (transmission failures) are not specified.

4 Boundedness of NP-nets

A plain Petri net is bounded iff there exists an upper bound for number of tokens in all reachable place markings. It is also well known, that the boundedness of a given PN is equivalent to finiteness of the PN reachability set. It is easy to extend both variants of this definition to NP-nets.

Definition 3. *A marked NP-net NP with an initial marking m_0 is k-bounded ($k \in \mathbb{N}$), iff for any reachable marking $m \in R(m_0)$ the number of tokens in any place in any component of NP is not greater than k, i.e. $(\forall p \in P_{SN} : |m(p)| \leq k) \wedge \big(\forall \alpha \in m(p) : \forall p' \in P_{EN(\alpha)} : m_\alpha(p') \leq k\big)$. A marked NP-net NP is bounded, iff NP is k-bounded for some k.*

It is easy to note, that a marked NP-net NP is bounded iff its reachability set $R_{NP}(m_0)$ is finite.

In what follows, when we speak of separate components we do not take into account transition synchronization labels. So, we can consider element nets as ordinary Petri nets and a system net as a high-level Petri net with a finite number of colors (each color corresponds to either an atomic type, or an element net type).

Theorem 1. *Let NP be a marked NP-net. If the system net, all net tokens in the initial marking, as well as all net constants in arc expressions in NP are bounded, then NP is bounded.*

The proof of this theorem follows immediately from the definitions, since vertical synchronizations in a system and element nets can only reduce reachability set of NP. Then the maximum of all bounds for all net components can be chosen as a bound for NP.

Note, that the converse of the Theorem 1 is not valid.

Definition 4. *Let (P,T,F) be a Petri net. A mapping $I : P \to \mathbb{Q}$ is an S-invariant iff for every $t \in T$ we have $\sum_{p \in {}^\bullet t} I(p) = \sum_{p \in t^\bullet} I(p)$.*

An S-invariant is called *positive* iff $I(p) > 0$ for every place $p \in P$. The following theorem was proved in [3].

Theorem 2. *If a marked PN has a positive S-invariant, then it is bounded.*

Theorem 2 can be directly used for checking boundedness of net tokens in a NP-net. As for a system net, it is a high-level PN, and in general case we cannot speak of integer valued invariants for high-level PNs.

However, system nets form a special, rather restricted subclass of high-level Petri nets. Net tokens are identical in terms of separate system net behavior, since each place contains net tokens of only one type. Then a transition firing depends only on count of net tokens in its input places. When a system net is considered as a separate high-level PN, arc expressions can be replaced by integers: a number n corresponds to an expression with n variables and constants (taking into account their multiplicity). Thus a system net is behaviorally equivalent to some plain PN with weighted arcs. Moreover, we can compute its S-invariants with algorithms for plain PNs.

Thus from Theorems 1 and 2 we obtain

Theorem 3. *Let NP be a marked NP-net. If the system net, all net tokens in the initial marking, and all net constants in the system net arc expressions in NP have positive invariants, then NP is bounded.*

5 Liveness of NP-nets

Liveness can be defined in several ways [6]. We will use "L4-live" property, which we redefine for NP-nets as follows.

Definition 5. *Let t be a transition in a component of a marked NP-net NP with an initial marking m_0. A transition t is called* live *iff for every reachable marking m there exists a sequence of firings starting from m, which includes t, i.e. $\forall m \in R(m_0) : \exists \sigma \in T^* : m \xrightarrow{\sigma} m' \xrightarrow{t}$.*

Checking liveness for NP-nets is a complicate problem. Transition firings in different NP-net components may be related by synchronization mechanism. Also net tokens may be consumed by system net transitions. So, for some applications we may consider liveness only for system net transitions.

Definition 6. *Let NP be a marked two-level NP-net. NP is called 0L-live iff every transition in its system net is live.*

0L-liveness is important when we are more interested in analyzing a system dynamics, rather than behavior of certain agents. The following theorem states some sufficient conditions for compositionality of 0L-liveness.

Theorem 4. *Let NP be a marked two-level NP-net. Let also NP satisfy the following conditions:*

1. *the system net in NP is live (considered as a separate Petri net);*
2. *all net tokens in the initial marking and all net constants in every arc expression are live (considered as separate Petri nets)*
3. *NP has only one label of vertical synchronization λ;*
4. *if t is a system net transition in NP labeled with λ, then for any $p \in {}^\bullet t$ the type of p is an element net, containing a transition labeled with λ.*

Then NP is 0L-live.

The proof of this theorem is based on the following. A synchronization transition in a system net can fire only simultaneously with synchronization transitions in net tokens involved in this firing. Liveness of net tokens then guarantee, that each net token can reach a state, when a transition needed for synchronization is enabled.

Note, that conditions 3 and 4 here are syntactical restrictions, that can be easily checked.

Definition 7. *Let NP be a marked NP-net. NP is said to be 1L-live iff every transition in its system net and every transition in each net token from the initial marking in NP are live.*

Definition 8. *Let NP be a NP-net. NP is called conservative iff for each transition t in the system net in NP the set of all variables in input arc expressions for t is a subset of all variables in its output arc expressions.*

It is easy to see, that if a conservative NP-net contains a net token of type α in its initial marking, then it contains at least one net token of type α in every reachable marking.

Theorem 5. *Let NP be a marked NP-net. Let also NP satisfy conditions 1-4 in the Theorem 4, as well as the following two conditions:*

1. *NP is conservative*
2. *each strongly connected component of the system net contains at least one transition labeled with λ.*

Then NP is 1L-live.

The proof of this theorem is inherently similar to the proof of the Theorem 4. Additionally note, that liveness of a net token may be violated, when this token is trapped in a system subnet without synchronization transitions. In such a case a system net is still live, but synchronization transitions in the net token may become dead. The last condition excludes this case.

Theorems 4 and 5 give us an approach for checking liveness of NP-nets with free-choice components compositionally.

Free-choice nets is an important subclass of Petri nets. They allow modeling a conflict and a synchronization, but not “confusion” (which is an interference of both).

It is well known that “checkability” frontier separates free-choice nets [3]. For free-choice nets there are effective (polynomial) algorithms for several important behavioral properties such as well-formedness (boundedness + liveness) [4, 3]. Actually, for bounded free-choice nets liveness can be checked in polynomial time.

Let us return to our example in section 3. Here the element nets and the system net in the P2P protocol model are free-choice nets. It can be checked (in polynomial time), that they are structurally live and bounded (well-formed). And that they are live and bounded in the given marking.

Under compositionality of liveness and boundedness for NP-nets the 0L-liveness and boundedness of the entire NP-net can be proved. Moreover, since the conditions of the theorem 5 are satisfied the modeled protocol is 1L-live.

6 Conclusion

In this paper we have characterized conditions under which nested Petri nets boundedness and liveness can be deduced from the same properties for net components. Compositional approach allows to reduce crucially Petri net state space, and makes checking these properties a tractable problem.

References

1. Lomazova I. A., *Nested Petri nets: modeling and analysis of distributed systems with object structure.* 208 p. Nauchny Mir, Moscow (2004) (in Russian).
2. Lomazova I. A., *Nested Petri nets — a Formalism for Specification and Verification of Multi-Agent Distributed Systems., Fundam. Inform.*, 43(1-4):195–214 , 2000.
3. Desel J., Esparza J., *Free Choice Petri Nets.* Cambridge University Press, New York, NY, USA, 1995.
4. Esparza J., *A Polynomial-Time Algorithm for Checking Consistency of Free-Choice Signal Transition Graphs.* Fundam. Inform. 62(2): 197–220, 2004.
5. Esparza J., Nielsen M., *Decibility issues for Petri nets — a survey.* Journal of Informatik Processing and Cybernetics, 30(3):143–160, 1994.
6. Murata T., *Petri Nets: Properties, Analysis and Applications, an invited survey paper.* Proceedings of the IEEE, Vol.77, No.4 pp.541–580, April, 1989.