A Survey of Decidability Results for Elementary Object Systems

Michael Köhler-Bußmeier
koehler@informatik.uni-hamburg.de
University of Hamburg, Department for Informatics
Vogt-Kölln-Straße 30, D-22527 Hamburg

Abstract

This contribution presents the formalism of Elementary Object Systems (Eos). Object nets are Petri nets which have Petri nets as tokens – an approach known as the nets-within-nets paradigm.

Since object nets in general are immediately Turing complete, we introduce the restricted class of elementary object nets which restrict the nesting of nets to the depth of two.

One central aim of this contribution is present several (un)decidability properties of Eos. It turns out that Eos are more powerful than classical p/t nets which is demonstrated by the fact that e.g. reachability and liveness become undecidable problems for Eos. Despite these undecidability results other properties can be extended to Eos using a monotonicity argument similar to that for p/t nets.

Keywords: mobility, nets-within-nets, object nets, reachability, liveness

1 Introduction

Object Systems are Petri nets which have Petri nets as tokens – an approach which is called the *nets-within-nets* paradigm, proposed by Valk [1, 2] for a two levelled structure and generalised in [3, 4] for arbitrary nesting structures. The Petri nets that are used as tokens are called net-tokens. Net-tokens are tokens with internal structure and inner activity. This is different from place refinement, since tokens are transported while a place refinement is static. Net-tokens are some kind of *dynamic* refinement of states.

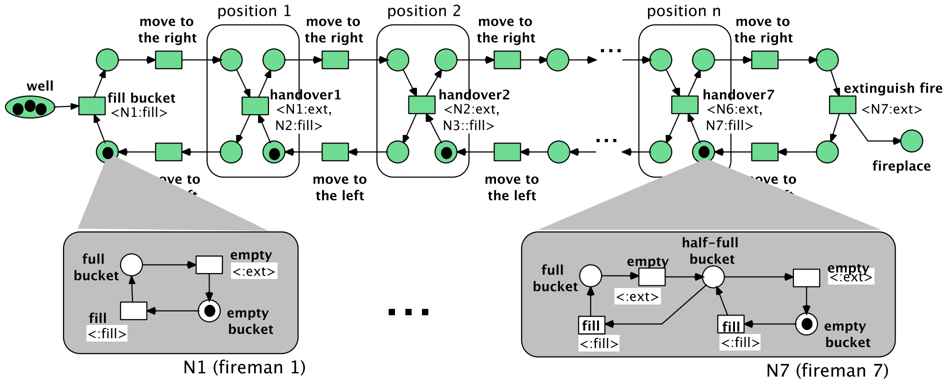

Figure 1: The Bucket Chain as a Nested System

It is quite natural to use object nets to model mobility and mobile agents (cf. [5] and [6]). Each place of the system net describes a location that hosts agents, which are net-tokens. Mobility can be modelled by moving the net-token from one place to another. This hierarchy forms a useful abstraction of the system: on a high level the agent system and on a lower level of the hierarchy the agent itself.

Without the viewpoint of nets as tokens, the modeller would have to encode the agent differently, e.g. as a data-type. This has the disadvantage, that the inner actions cannot be modelled directly, so, they have to be lifted to the system net, which seems quite unnatural. By using nets-within-nets we can investigate the concurrency of the system and the agent in one model without loosing the abstraction needed.

Example: Figure 1 shows an object systems which models Carl Adam Petri's bucket chain scenario [7] where the fireman are mobile. In the bucket chain-example n firemen are standing in a row, each equipped with a bucket. A well is available for the leftmost fireman and the fire is at the rightmost place. So the leftmost fireman fills his bucket, while the rightmost extinguishes the fire. Neighboured firemen can handover buckets, so full buckets are handed over to right (to extinguish the fire) and empty ones to the left for refilling.[1] The topology (i.e. each fireman can only interact with his intermediate neighbour) introduces an interesting causal dependency structure[2]: The effect of exchanging buckets at a location being k steps away can be observed only when the whole system has moved k steps ahead.

Figure 1 shows the Petri net modelling the bucket chain with $n = 7$ firemen. Each fireman initially carries one empty bucket. The fireman man are mobile: They are net-tokens that move around in the system net which models the chain itself. The agents have different capabilities. For example the object net N_7 has a bucket with double capacity. At the well the object net N_1 fires the transition fill bucket which synchronises over the communication channel $\langle$:fill$\rangle$ with the transition fill of the object net. Then the object net N_1 moves to the position 1 firing the transition move to the right. At the position 1 N_1 synchronises with N_1 firing the transition handover which has two invocations of communication channels: The object net N_1 is synchronised over the channel $\langle$:ext$\rangle$ which changes its state to empty bucket. Synchronously, the object net N_2 is synchronised over the channel $\langle$:fill$\rangle$ which changes its state to full bucket. Thereafter the object net N_1 moves back to the well via the transition move to the left and the object net N_2 moves in the direction of the fire via the transition move to the right. Similarly for the other positions. So, the firemen transport full buckets to the right and the fire will be finally extinguished.

Among the wealth of research on defining mobile systems, in recent years a variety of formalisms have been introduced or adopted to cover mobility: The approaches can be roughly separated into process calculi and Petri net based approaches. The π-calculus [9], the Ambient calculus [10] and the Seal calculus [11] are just three of the more popular calculi. Approaches dealing with mobility and Petri nets are elementary object net systems [12, 2], mobile nets [13], recursive nets [14], minimal object nets [15, 16], nested nets [17], mobile predicate/transition nets [18], Reference nets [19], PN^2 [20], hypernets [21], object net systems [4, 22, 23, 24], Mobile Systems [25], AHO systems [26], mobile object nets [23], adaptive workflow nets [27], ν-Abstract Petri nets [28], and Hornets [29].

One central aim of this contribution is to compile existing results on special aspects of Eos together with some unpublished properties within one self-contained presentation. As a byproduct most proofs have been rewritten and shortened in a way apropriate for the newer results.

The paper has the following structure: Section 2 recalls basic notations of Petri nets. Section 3 defines elementary object systems (Eos) and Section 4 compares

[1]This scenario has been introduced to study the causal dependencies of distributed cooperation. The scenario serves a similar purpose as the well known *Bankers-Problem* for deadlock-prevention in resource allocation systems (i.e. operating systems) or the *Dining Philosophers* for the study of fairness in distributed systems [8].

[2]In the general research of Petri this causal dependency structure is closely related to Einstein's physical theory of relativity. This topic is studied in Petri's research of general net theory.

EOS with a reference semantics based on p/t nets and introduces a sub class, called Generalised State Machines, which is of practical interest, because models of this class corresponds to scenarios related to physical entities. Section 5 provides a short overview of related nets-within-nets formalisms. Section 6 studies decidability problems for EOS, namely: reachability, liveness and boundedness.[3]

2 Preliminaries

The definition of Petri nets relies on the notion of multisets. A multiset $\mathbf{m}$ on the set D is a mapping $\mathbf{m} : D \to \mathbb{N}$. Multisets are generalisations of sets in the sense that every subset of D corresponds to a multiset $\mathbf{m}$ with $\mathbf{m}(d) \leq 1$ for all $d \in D$. The notation is used for sets as well as for multisets. The meaning will be apparent from its use. Multiset addition $\mathbf{m}_1, \mathbf{m}_2 : D \to \mathbb{N}$ is defined component-wise: $(\mathbf{m}_1 + \mathbf{m}_2)(d) := \mathbf{m}_1(d) + \mathbf{m}_2(d)$. The empty multiset $\mathbf{0}$ is defined as $\mathbf{0}(d) = 0$ for all $d \in D$. Multiset-difference $\mathbf{m}_1 - \mathbf{m}_2$ is defined by $(\mathbf{m}_1 - \mathbf{m}_2)(d) := \max(\mathbf{m}_1(d) - \mathbf{m}_2(d), 0)$. We use common notations for the cardinality of a multiset $|\mathbf{m}| := \sum_{d \in D} \mathbf{m}(d)$ and multiset ordering $\mathbf{m}_1 \leq \mathbf{m}_2$ where the partial order $\leq$ is defined by $\mathbf{m}_1 \leq \mathbf{m}_2 \iff \forall d \in D : \mathbf{m}_1(d) \leq \mathbf{m}_2(d)$. A multiset $\mathbf{m}$ is finite if $|\mathbf{m}| < \infty$. The set of all finite multisets over the set D is denoted $MS(D)$. The set $MS(D)$ naturally forms a monoid with multiset addition $+$ and the empty multiset $\mathbf{0}$. Multisets can be identified with the commutative monoid structure $(MS(D), +, 0)$. Multisets are the free commutative monoid over D since every multiset has the unique representation in the form $\mathbf{m} = \sum_{d \in D} \mathbf{m}(d) \cdot d$ where $\mathbf{m}(d)$ denotes the multiplicity of d. Multisets can be represented as a formal sum in the form $\mathbf{m} = \sum_{i=1}^{|\mathbf{m}|} x_i$ where $x_i \in D$.

Any mapping $f : D \to D'$ can be extended to a homomorphism $f^\sharp : MS(D) \to MS(D')$ on multisets: $f^\sharp\left(\sum_{i=1}^{n} x_i\right) = \sum_{i=1}^{n} f(x_i)$. This includes the special case $f^\sharp(\mathbf{0}) = \mathbf{0}$. We simply write f to denote the mapping $f^\sharp$. The notation is in accordance with the set-theoretic notation $f(A) = \{f(a) \mid a \in A\}$.

Definition 1. *A* p/t net *N is a tuple $N = (P, T, \mathbf{pre}, \mathbf{post})$, such that P is a set of places, T is a set of transitions, with $P \cap T = \emptyset$, and $\mathbf{pre}, \mathbf{post} : T \to MS(P)$ are the pre- and post-condition functions. A marking of N is a multiset of places: $\mathbf{m} \in MS(P)$. A p/t net with initial marking $\mathbf{m}$ is denoted $N = (P, T, \mathbf{pre}, \mathbf{post}, \mathbf{m})$.*

We use the usual notations for nets like ${}^\bullet x$ for the set of predecessors and $x^\bullet$ for the set of successors for a node $x \in (P \cup T)$.

A transition $t \in T$ of a p/t net N is enabled in marking $\mathbf{m}$ iff $\forall p \in P : \mathbf{m}(p) \geq \mathbf{pre}(t)(p)$ holds. The successor marking when firing t is $\mathbf{m}'(p) = \mathbf{m}(p) - \mathbf{pre}(t)(p) + \mathbf{post}(t)(p)$ for all $p \in P$. Using multiset notation enabling is expressed by $\mathbf{m} \geq \mathbf{pre}(t)$ and the successor marking is $\mathbf{m}' = \mathbf{m} - \mathbf{pre}(t) + \mathbf{post}(t)$. We denote the enabling of t in marking $\mathbf{m}$ by $\mathbf{m} \xrightarrow[N]{t}$. Firing of t is denoted by $\mathbf{m} \xrightarrow[N]{t} \mathbf{m}'$. The net N is omitted if it is clear from the context.

Firing is extended to sequences $w \in T^*$ in the obvious way:
(i) $\mathbf{m} \xrightarrow{\epsilon} \mathbf{m}$;
(ii) If $\mathbf{m} \xrightarrow{w} \mathbf{m}'$ and $\mathbf{m}' \xrightarrow{t} \mathbf{m}''$ hold, then we have $\mathbf{m} \xrightarrow{wt} \mathbf{m}''$.

We write $\mathbf{m} \xrightarrow{*} \mathbf{m}'$ whenever there is some $w \in T^*$ such that $\mathbf{m} \xrightarrow{w} \mathbf{m}'$ holds.

The set of reachable markings is $RS(\mathbf{m}_0) : \{\mathbf{m} \mid \exists w \in T^* : \mathbf{m}_0 \xrightarrow{w} \mathbf{m}\}$.

[3]Due to space restrictions, we omit proofs. An extended version of this paper can be obtained via the author's web page at `http://www.informatik.uni-hamburg.de/TGI`

3 Elementary Object Systems

An elementary object system (Eos) is composed of a system net, which is a p/t net $\widehat{N} = (\widehat{P}, \widehat{T}, \mathbf{pre}, \mathbf{post})$ and a set of object nets $\mathcal{N} = \{N_1, \ldots, N_n\}$, which are p/t nets given as $N = (P_N, T_N, \mathbf{pre}_N, \mathbf{post}_N)$. In extensionwe assume that all sets of nodes (places and transitions) are pairwise disjoint. Moreover we have $\widehat{N} \notin \mathcal{N}$. We assume the existence of the object net $\bullet \in \mathcal{N}$ which has no places and no transitions and is used to model anonymous, so called black tokens.

The system net places are typed by the mapping $d : \widehat{P} \to \mathcal{N}$ with the meaning, that the place $\widehat{p}$ of the system net contains net-tokens of the object net type N if $d(\widehat{p}) = N$.[4] No place of the system net is mapped to the system net itself since $\widehat{N} \notin \mathcal{N}$.

Since the tokens of an Eos are instances of object nets a *marking* $\mu \in \mathcal{M}$ of an Eos OS is a *nested* multiset.

A marking of an Eos OS is denoted $\mu = \sum_{k=1}^{|\mu|}(\widehat{p}_k, M_k)$ where $\widehat{p}_k$ is a place in the system net and M_k is the marking of the net-token of type $d(\widehat{p}_k)$. To emphasise the nesting, markings are also denoted as $\mu = \sum_{k=1}^{|\mu|} \widehat{p}_k[M_k]$. Tokens of the form $\widehat{p}[\mathbf{0}]$ and $d(\widehat{p}) = \bullet$ are abbreviated as $\widehat{p}[]$.

The set of all markings which are syntactically consistent with the typing d is denoted $\mathcal{M}$ (Here $d^{-1}(N) \subseteq \widehat{P}$ is the set of system net places of the type N):

$$\mathcal{M} := MS\left(\bigcup_{N \in \mathcal{N}} \left(d^{-1}(N) \times MS(P_N)\right)\right) \tag{1}$$

The transitions in an Eos are labelled with synchronisation channels. We assume a fixed set of channels C, including the channel ϵ which is used to describe the absence of any "real" channel. Each transition of the system net has one label for each object, defined by the labelling function function $\widehat{l} : \widehat{T} \to (N \to C)$. Each transition of an object net N has one single label, defined by the labelling function function $l_N : T_N \to C$. In the graphical representation the synchronisation labelling is defined by transition inscriptions in the form $\langle N_1 : \widehat{l}(\widehat{t})(N_1), \ldots, N_k : \widehat{l}(\widehat{t})(N_k)\rangle$ in the system net and in the form $\langle : l_(N(t)\rangle$ in the object nets (where ϵ is omitted).

A system event θ is generated by transitions with matching labels. The labelling introduces three cases of events:

1. System-autonomous firing: The transition $\widehat{t}$ of the system net fires autonomously, whenever $\widehat{l}(\widehat{t})(N) = \epsilon$ for all $N \in \mathcal{N}$.

2. Synchronised firing: There is at least one object net that has to be synchronised, i.e. there is a N such that $\widehat{l}(\widehat{t})(N) \neq \epsilon$.

3. Object-autonomous firing: An object net transition t fires autonomously whenever $l(t) = \epsilon$.

These three kinds of events can be reduced to the case of a synchronisation where a system net transition has exactly one synchronisation partner in each object net. This normal form is obtained by adding some idle transitions: For each object net $N \in \mathcal{N}$ we add the idle-transitions ϵ_N with $\mathbf{pre}_N(\epsilon_N) = \mathbf{post}_N(\epsilon_N) = \mathbf{0}$ to its transition set. For object-atonomous events we also add the set of idle transitions $\epsilon_{\widehat{P}} := \{\epsilon_{\widehat{p}} \mid \widehat{p} \in \widehat{P}\}$ with $\mathbf{pre}(\epsilon_{\widehat{p}}) = \mathbf{post}(\epsilon_{\widehat{p}}) = \widehat{p}$ to the set of system net transitions. We extend the labelling to idle transitions by $\widehat{l}(\epsilon_{\widehat{p}})(N) = l_N(\epsilon_N) = \epsilon$ for all $\widehat{p} \in \widehat{P}$ and $N \in \mathcal{N}$.

[4]In the following the terms *(marked) object net* and *net-token* are used almost interchangeable. We use the term *net-token* whenever we like to emphasise the aspect that the marked object net is a token of the system net.

With these idle transitions the channel ϵ (which means "no synchronisation") is modelled as a synchronisation with an idle transition that has no effect.

The synchronisation labelling generates the set of system events Θ. An event is a pair – denoted $\widehat{\tau}[\theta]$ in the following. Here, $\widehat{\tau}$ is either a real transition $\widehat{t}$ or $\epsilon_{\widehat{p}}$ for some $\widehat{p}$; θ maps each object net to one of its transitions. An event has the meaning that the system net transition $\widehat{\tau}$ fires synchronously with all the object net transitions $\theta(N), N \in \mathcal{N}$.

A special case for the mapping θ is the idle map $\epsilon_{\mathcal{N}}$ which is defined $\epsilon_{\mathcal{N}}(N) = \epsilon_N$ for all $N \in \mathcal{N}$.

All events are generated from the labels: $\widehat{l}(\widehat{\tau})(N) = l_N(\theta(N))$ must hold for all $N \in \mathcal{N}$. Whenever $\widehat{\tau}$ is an idle transition $\epsilon_{\widehat{p}} \in \epsilon_{\widehat{P}}$ we also demand that $\theta(N)$ is the idle event ϵ_N except for exactly one object net N (which is the object-autonomous event), i.e. $|\{N \in \mathcal{N} : \theta(N) \neq \epsilon_N\}| = 1$ holds.

$$\begin{aligned}\Theta_l \quad := \quad \Big\{\widehat{\tau}[\theta] \mid \quad & \forall N \in \mathcal{N} : \widehat{l}(\widehat{t})(N) = l_N(\theta(N)) \wedge \\ & \widehat{\tau} \in \epsilon_{\widehat{P}} \Longrightarrow |\{N \in \mathcal{N} : \theta(N) \neq \epsilon_N\}| = 1\Big\}\end{aligned} \tag{2}$$

Definition 2 (EOS)**.** *An elementary object system (* EOS*) is a tuple* $OS = (\widehat{N}, \mathcal{N}, d, \Theta_l)$ *such that:*

1. $\widehat{N}$ *is a p/t net, called the* system net*.*
2. $\mathcal{N}$ *is a finite set of disjoint p/t nets, called* object nets*.*
3. $d : \widehat{P} \to \mathcal{N}$ *is the typing of the system net places.*
4. Θ_l *is the set of events generated from the labelling* $l = (\widehat{l}, l_N)_{N \in \mathcal{N}}$*.*

An EOS *with initial marking is a tuple* $OS = (\widehat{N}, \mathcal{N}, d, \Theta_l, \mu_0)$ *where* $\mu_0 \in \mathcal{M}$ *is the initial marking.*

Example: Figure 2 shows an EOS with the system net $\widehat{N}$ and the object nets $\mathcal{N} = \{N_1, N_2\}$. The system has four net-tokens: two on place p_1 and one on p_2 and p_3 each. The net-tokens on p_1 and p_2 share the same net structure, but have independent markings.

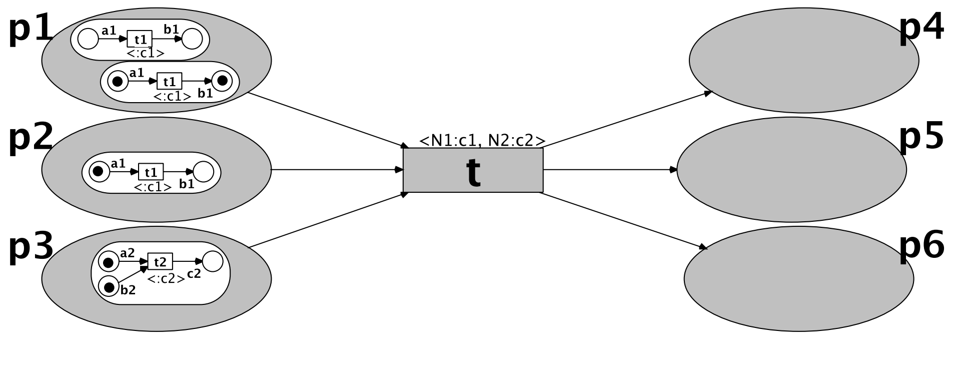

Figure 2: An Object Net

- The system net is given as $\widehat{N} = (\widehat{P}, \widehat{T}, \mathbf{pre}, \mathbf{post})$ with $\widehat{P} = \{p_1, \ldots, p_6\}$ and $\widehat{T} = \{t\}$.
- The first object net is $N_1 = (P_1, T_1, \mathbf{pre}_1, \mathbf{post}_1)$ with $P_1 = \{a_1, b_1\}$ and $T_1 = \{t_1\}$.

- The second object net is $N_2 = (P_2, T_2, \mathbf{pre}_2, \mathbf{post}_2)$ with $P_2 = \{a_2, b_2, c_2\}$ and $T_2 = \{t_2\}$.
- The typing is $d(p_1) = d(p_2) = d(p_4) = N_1$ and $d(p_3) = d(p_5) = d(p_6) = N_2$.
- The labelling function of the system net $\widehat{l}$ is defined by $\widehat{l}(t)(N_1) = c_1$ and $\widehat{l}(t)(N_2) = c_2$.

  The labelling l_{N_1} of the first object net is defined by setting $l_{N_1}(t_1) = c_1$. Similarily, l_{N_2} is defined by $l_{N_2}(t_2) = c_2$.

  There is only one synchronous event: $\Theta_l = \{t[N_1 \mapsto t_1, N_2 \mapsto t_2]\}$.
- The initial marking has two net-tokens on p_1, one on p_2, and one on p_3:

$$\mu = p_1[a_1 + b_1] + p_1[\mathbf{0}] + p_2[a_1] + p_3[a_2 + b_2]$$

Note, that for Figure 2 the structure is the same for the three net-tokens on p_1 and p_2 but the net-tokens' markings are different.

We name special properties of EOS:

- An EOS is *minimal* iff it has exactly one "real" object net: $|\mathcal{N} \setminus \{\bullet\}| = 1$.
- An EOS is *pure* iff it has no places for black tokens: $d^{-1}(\bullet) = \emptyset$.
- An EOS is *p/t-like* iff it has only places for black tokens: $d(\widehat{P}) = \{\bullet\}$.
- An EOS is *unary* iff it is pure and minimal.
- An EOS *OS* is a generalised state machine (GSM) iff for all $\widehat{t}$ there is either exactly one place in the preset and one in the postset typed with the object net N or there are no such places. (This class will be discussed in Section 4 in detail.)
- An EOS is *simple* iff so is its typing d. A typing is called *simple* iff for each place in the preset of a system net transition $\widehat{t}$ there is place in the postset being of equal type: $(d({}^{\bullet}\widehat{t}) \cap \mathcal{N}) \subseteq (d(\widehat{t}^{\bullet}) \cap \mathcal{N})$.

3.1 Projections and Firing Rule

Let $\mu = \sum_{k=1}^{|\mu|}(\widehat{p}_k, M_k)$ be a marking of an EOS. The projection Π^1 on the first component abstracts away the substructure of all net-tokens:

$$\Pi^1\left(\sum\nolimits_{k=1}^{|\mu|} \widehat{p}_k[M_k]\right) := \sum\nolimits_{k=1}^{|\mu|} \widehat{p}_k \tag{3}$$

The projection Π^2_N on the second component is the abstract marking of all net-tokens of the type $N \in \mathcal{N}$ ignoring their local distribution within the system net.

$$\Pi^2_N\left(\sum\nolimits_{k=1}^{|\mu|} \widehat{p}_k[M_k]\right) := \sum\nolimits_{k=1}^{|\mu|} \mathbf{1}_N(\widehat{p}_k) \cdot M_k \tag{4}$$

where the indicator function $\mathbf{1}_N : \widehat{P} \to \{0,1\}$ is $\mathbf{1}_N(\widehat{p}) = 1$ iff $d(\widehat{p}) = N$. Note, that $\Pi^2_N(\mu)$ results in a marking of the object net N.

A system event $\widehat{\tau}[\theta]$ removes net-tokens together with their individual internal markings. Firing the event replaces a nested multiset $\lambda \in \mathcal{M}$ that is part of the current marking μ, i.e. $\lambda \le \mu$, by the nested multiset ρ. The enabling condition is expressed by the *enabling predicate* ϕ_{OS} (or just ϕ whenever *OS* is clear from the context):

$$\begin{aligned}\phi(\widehat{\tau}[\theta], \lambda, \rho) \quad &\iff \Pi^1(\lambda) = \mathbf{pre}(\widehat{\tau}) \wedge \Pi^1(\rho) = \mathbf{post}(\widehat{\tau}) \wedge \\ &\forall N \in \mathcal{N} : \Pi^2_N(\lambda) \geq \mathbf{pre}_N(\theta(N)) \wedge \\ &\forall N \in \mathcal{N} : \Pi^2_N(\rho) = \Pi^2_N(\lambda) - \mathbf{pre}_N(\theta(N)) + \mathbf{post}_N(\theta(N))\end{aligned} \tag{5}$$

With $\widehat{M} = \Pi^1(\lambda)$ and $\widehat{M'} = \Pi^1(\rho)$ as well as $M_N = \Pi^2_N(\lambda)$ and $M'_N = \Pi^2_N(\rho)$ for all $N \in \mathcal{N}$ the predicate ϕ has the following meaning:

1. The first conjunct expresses that the system net multiset $\widehat{M}$ corresponds to the pre-condition of the system net transition $\widehat{t}$, i.e. $\widehat{M} = \mathbf{pre}(\widehat{t})$.

2. In turn, a multiset $\widehat{M'}$ is produced, that corresponds with the post-set of $\widehat{t}$.

3. An object net transition τ_N is enabled if the combination M_N of the markings net-tokens of type N enable it, i.e. $M_N \geq \mathbf{pre}_N(\theta(N))$.

4. The firing of $\widehat{\tau}[\theta]$ must also obey the *object marking distribution condition* which is essential for the formulation of linear invariants: $M'_N = M_N - \mathbf{pre}_N(\theta(N)) + \mathbf{post}_N(\theta(N))$ where $\mathbf{post}_N(\theta(N)) - \mathbf{pre}_N(\theta(N))$ is the effect of the object net's transition on the net-tokens.

Note, that (1) and (2) assures that only net-tokens relevant for the firing are included in λ and ρ. Conditions (3) and (4) allows for additonal tokens in the net-tokens.

For system-autonomous events $\widehat{t}[\epsilon_{\mathcal{N}}]$ the enabling predicate ϕ can be simplified further. We have $\mathbf{pre}_N(\epsilon_N) = \mathbf{post}_N(\epsilon_N) = \mathbf{0}$. This ensures $\Pi^2_N(\lambda) = \Pi^2_N(\rho)$, i.e. the sum of markings in the copies of a net-token is preserved w.r.t. each type N. This condition ensures the existence of linear invariance properties.

Analogously, for an object-autonomous event we have an idle-transition $\widehat{\tau} = \epsilon_{\widehat{p}}$ for the system net and the first and the second conjunct is: $\Pi^1(\lambda) = \mathbf{pre}(\widehat{t}) = \widehat{p} = \mathbf{post}(\widehat{t}) = \Pi^1(\rho)$. So, there is an addend $\lambda = \widehat{p}[M]$ in μ with $d(\widehat{p}) = N$ and M enables $t_N := \theta(N)$.

Definition 3 (Firing Rule)**.** *Let OS be an* Eos *and $\mu, \mu' \in \mathcal{M}$ markings. The event $\widehat{\tau}[\theta]$ is enabled in μ for the mode $(\lambda, \rho) \in \mathcal{M}^2$ iff $\lambda \leq \mu \wedge \phi(\widehat{\tau}[\theta], \lambda, \rho)$ holds.*

An event $\widehat{\tau}[\theta]$ that is enabled in μ for the mode (λ, ρ) can fire: $\mu \xrightarrow[OS]{\widehat{\tau}[\theta](\lambda,\rho)} \mu'$. The resulting successor marking is defined as $\mu' = \mu - \lambda + \rho$.

We write $\mu \xrightarrow[OS]{\widehat{\tau}[\theta]} \mu'$ whenever $\mu \xrightarrow[OS]{\widehat{\tau}[\theta](\lambda,\rho)} \mu'$ for some mode (λ, ρ).

Note, that the firing rule has no a-priori decision how to distribute the marking on the generated net-tokens. Therefore we need the mode (λ, ρ) to formulate the firing of $\widehat{\tau}[\theta]$ in a functional way.[5]

[5] Of course there are a lot different possible candidates for the firing rule. In fact there infinitely many, since there is a lot of freedom how to distribute the net-tokens' markings when there are several outgoing arcs in the system net. Valk [2] discusses three basic variants, called reference semantics, value semantics, and copy semantics. Reference semantics interprets net-tokens as pointers to object nets. This semantics can be equally expressed as a p/t net (cf. definition 10). Value semantics is the semantics presented in this paper. Copy semantics is a variant of value semantics where the net-tokens' markings are not distributed over the outgoing net-tokens but copied. From the modelling point of view each of the semantics has its own pro and cons. From a more theoretical point of value semantics is a special one since it allows to reinterpret every firing sequence also with respect to reference semantics – as formulated in Theorem 11. One can even show that value semantics is the only one with this propertiy (cf. [30] for details).

Example: Consider the Eos of Figure 2 again. The current marking μ of the Eos enables $t[N_1 \mapsto t_1, N_2 \mapsto t_2]$ in the mode (λ, ρ) where

$$\begin{aligned} \lambda &= p_1[a_1 + b_1] + p_2[a_1] + p_3[a_2 + b_2] \\ \rho &= p_4[a_1 + b_1 + b_1] + p_5[\mathbf{0}] + p_6[c_2] \end{aligned}$$

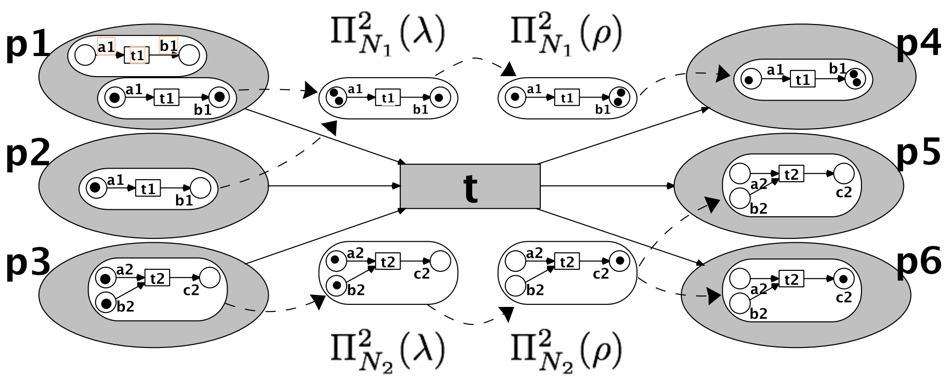

Figure 3: The EOS of Figure 2 illustrating the projections $\Pi^2_N(\lambda)$ and $\Pi^2_N(\rho)$

We have the current marking:

$$\mu \quad = \quad p_1[\mathbf{0}] + \underbrace{p_1[a_1 + b_1] + p_2[a_1] + p_3[a_2 + b_2]}_{\lambda} = p_1[\mathbf{0}] + \lambda$$

The net-tokens' markings are added by the projections Π^2_N resulting in the markings $\Pi^2_N(\lambda)$. The sub-synchronisation generate $\Pi^2_N(\rho)$. (The results are shown above and below the transition t.) After the synchronisation we obtain the successor marking on p_4, p_5, and p_6 as shown in the Figure 3:

$$\begin{aligned} \mu' &= (\mu - \lambda) + \rho = p_1[\mathbf{0}] + \rho \\ &= p_1[\mathbf{0}] + p_4[a_1 + b_1 + b_1] + p_5[\mathbf{0}] + p_6[c_2] \end{aligned}$$

Eos are a canonical extension of p/t nets in two ways: The behaviour of the system net in the Eos when ignoring the net-tokens structure cannot be distinguished from the system net as a p/t net (Lemma 4) and each p/t-like Eos is isomorphic to the system net as a p/t net (Lemma 5) .

Eos are a canonical extension of p/t nets, since the behaviour of an Eos when considering only the system net's perspective is in accordance with the behaviour of the system net considered as a p/t net, i.e. if an event $\widehat{t}$ is disabled in the p/t net then for all θ the event $\widehat{t}[\theta]$ is disabled in the Eos.

Lemma 4. *For $OS = (\widehat{N}, \mathcal{N}, d, \Theta, \mu_0)$ define $\Pi^1(OS) = \widehat{N}$. For each* Eos *OS we have:*

$$\mu \xrightarrow[OS]{\widehat{t}[\theta]} \mu' \Longrightarrow \Pi^1(\mu) \xrightarrow[\Pi^1(OS)]{\widehat{t}} \Pi^1(\mu')$$

For a p/t-like Eos we have no object nets: $\mathcal{N} = \emptyset$, synchronisation given as $\Theta = \{\widehat{t}[\emptyset] \mid \widehat{t} \in \widehat{T}\}$, and the typing is the constant function $d = \bullet$ with $\bullet(\widehat{p}) = \bullet$ for all $\widehat{p} \in \widehat{P}$. The initial marking contains no submarking: $\mu_0 \in \widehat{P} \times \{\mathbf{0}\} \subseteq \mathcal{M}$. So, p/t-like Eos have the form:

$$OS = (\widehat{N}, \emptyset, \bullet, \{\widehat{t}[\emptyset] \mid \widehat{t} \in \widehat{T}\}, \mu_0)$$

Lemma 5. *A p/t-like* Eos $OS = (\widehat{N}, \emptyset, \bullet, \Theta_l, \mu_0)$ *is isomorphic to the p/t net* $(\widehat{N}, \Pi^1(\mu_0))$ *in the following sense:*

$$\mu \xrightarrow[OS]{\widehat{\tau}[\emptyset](\lambda,\rho)} \mu' \iff \Pi^1(\mu) \xrightarrow[\widehat{N}]{\widehat{\tau}} \Pi^1(\mu')$$

3.2 Projection Invariance of the Firing Rule

We define the relation $\cong \subseteq \mathcal{M}^2$ on nested multisets, that relates nested markings which coincide in their projections. The *projection equivalence* $\cong$ is a relation on $\mathcal{M}$ defined by:

$$\alpha \cong \beta \quad :\iff \quad \Pi^1(\alpha) = \Pi^1(\beta) \wedge \forall N \in \mathcal{N} : \Pi^2_N(\alpha) = \Pi^2_N(\beta) \tag{6}$$

The relation $\alpha \cong \beta$ abstracts from the location, i.e. the concrete net-token, in which a object net's place p is marked as long as it is present in α and β. For example, for $d(\widehat{p}) = d(\widehat{p}')$ we have

$$\widehat{p}[p_1 + p_2] + \widehat{p}'[p_3] \cong \widehat{p}[p_3 + p_2] + \widehat{p}'[p_1]$$

which means that $\cong$ allows the tokens p_1 and p_3 to change their locations (i.e. between $\widehat{p}$ and $\widehat{p}'$).

Lemma 6. *The enabling predicate is invariant with respect to the relation* $\cong$*:*

$$\phi(\widehat{\tau}[\theta], \lambda, \rho) \iff (\forall \lambda', \rho' : \lambda' \cong \lambda \wedge \rho' \cong \rho \Longrightarrow \phi(\widehat{\tau}[\theta], \lambda', \rho'))$$

For the definition of firing we use the projection equivalence to express that on firing the system net collects all relevant object nets of the firing mode and combines them to one "virtual object net" that is only present at the moment of firing. Due to this collection the location of the object nets' tokens is irrelevant and can be ignored using the projection equivalence.

Proposition 7 (Invariance)**.** *Let OS be an* Eos *and* μ *a marking. The event* $\widehat{\tau}[\theta]$ *is enabled in the mode* (λ, ρ) *iff*

$$\lambda \leq \mu \wedge \forall \lambda', \rho' : \lambda' \cong \lambda \wedge \rho' \cong \rho \wedge \phi(\widehat{\tau}[\theta], \lambda', \rho')$$

3.3 Reversibility

A basic property of Petri nets is that their firing rule is symmetric in time, i.e. whenever all arcs are reversed then we can fire backwards: This is expressed by the reversed net $N^{rev} = (P, T, \mathbf{pre}^{rev}, \mathbf{post}^{rev})$ where $\mathbf{pre}^{rev} := \mathbf{post}$ and $\mathbf{post}^{rev} := \mathbf{pre}$ which is obtained from $N = (P, T, \mathbf{pre}, \mathbf{post})$ by dualising the effect. Symmetry in time is expressed as:

$$m_1 \xrightarrow[N]{t} m_2 \iff m_2 \xrightarrow[N^{rev}]{t} m_1$$

This property holds also for Eos. Given $OS = (\widehat{N}, \mathcal{N}, d, l)$ we define the *reverse* Eos as $OS^{rev} = (\widehat{N}^{rev}, \mathcal{N}^{rev}, d, l)$ where $\mathcal{N}^{rev} = \{N^{rev} \mid N \in \mathcal{N}\}$.

Lemma 8. *Let OS be an* Eos*. The enabling predicate is reversible:*

$$\phi_{OS}(\widehat{\tau}[\theta], \lambda, \rho) \iff \phi_{OS^{rev}}(\widehat{\tau}[\theta], \rho, \lambda)$$

This implies reversibility for Eos.

Proposition 9. *Let OS be an* Eos*. Firing is reversible:*

$$\mu \xrightarrow[OS]{\widehat{\tau}[\theta](\lambda,\rho)} \mu' \iff \mu' \xrightarrow[OS^{rev}]{\widehat{\tau}[\theta](\rho,\lambda)} \mu$$

4 Reference Semantics and Generalised State Machines

For each EOS there is an obvious construction of a p/t net, called the reference net, which is constructed by taking as the set of places the disjoint union of all places and as the set of transitions the synchronisations.

Definition 10. *Let $OS = (\widehat{N}, \mathcal{N}, d, l, \mu_0)$ be an* EOS. *The* reference net $\mathrm{RN}(OS)$ *is defined as the p/t net:*

$$\mathrm{RN}(OS) = \left(\left(\widehat{P} \cup \bigcup\nolimits_{N\in\mathcal{N}} P_N\right), \Theta, \mathbf{pre}^{\mathrm{RN}}, \mathbf{post}^{\mathrm{RN}}, \mathrm{RN}(\mu_0)\right)$$

where $\mathbf{pre}^{\mathrm{RN}}$ *(and analogously* $\mathbf{post}^{\mathrm{RN}}$*) is defined by:*

$$\mathbf{pre}^{\mathrm{RN}}(\widehat{\tau}[\theta]) = \mathbf{pre}(\widehat{\tau}) + \sum\nolimits_{N\in\mathcal{N}} \mathbf{pre}_N(\theta(N))$$

and for markings we define:

$$\mathrm{RN}(\mu) := \Pi^1(\mu) + \sum_{N\in\mathcal{N}} \Pi^2_N(\mu)$$

The net is called *reference net* because it behaves as if each object net would have been accessed via pointers and not like a value: A black token on a system net place $\widehat{p}$ is interpreted as a pointer to the object $\widehat{N} = d(\widehat{p})$ where each object net has exactly one instance but several pointers referring to it.

Theorem 11. *Let OS be an* EOS. *Every event $\widehat{\tau}[\theta]$ that is activated in OS for (λ, ρ) is so in* $\mathrm{RN}(OS)$*:*

$$\mu \xrightarrow[OS]{\widehat{\tau}[\theta](\lambda,\rho)} \mu' \quad\Longrightarrow\quad \mathrm{RN}(\mu) \xrightarrow[\mathrm{RN}(OS)]{\widehat{\tau}[\theta]} \mathrm{RN}(\mu')$$

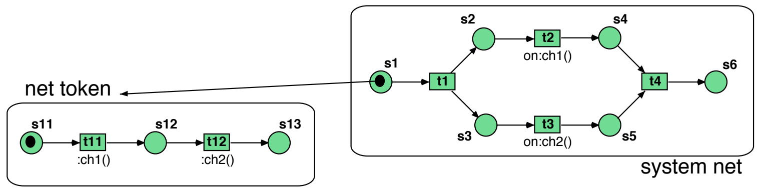

Figure 4: A sample EOS

The converse is not true in general, which can be demonstrated using the EOS in Fig. 4 known as the α-centauri example, cf. [12]. Initially we have $\mu_0 = \widehat{s}_1[s_{11}]$. In the reference net we have the initial marking $\mathrm{RN}(\mu_0) = \widehat{s}_1 + s_{11}$ which activates the firing sequence:

$$(\widehat{s}_1 + s_{11}) \xrightarrow{\widehat{t}_1[\epsilon]} (\widehat{s}_2 + \widehat{s}_3 + s_{11}) \xrightarrow{\widehat{t}_2[t_{11}]} (\widehat{s}_4 + \widehat{s}_3 + s_{12}) \xrightarrow{\widehat{t}_3[t_{12}]} (\widehat{s}_4 + \widehat{s}_5 + s_{13})$$

It is easy to see that in the EOS we can fire only a prefix, depending on the choice of the modes. The first mode assigns the token on s_{11} to the net-token on $\widehat{s}_3$:

$$\widehat{s}_1[s_{11}] \xrightarrow{\widehat{t}_1[\epsilon]} \widehat{s}_2[\mathbf{0}] + \widehat{s}_3[s_{11}]$$

The second mode assigns the token on s_{11} to the net-token on $\widehat{s}_2$:

$$\widehat{s}_1[s_{11}] \xrightarrow{\widehat{t}_1[\epsilon]} \widehat{s}_2[s_{11}] + \widehat{s}_3[\mathbf{0}] \xrightarrow{\widehat{t}_2[t_{11}]} \widehat{s}_4[s_{12}] + \widehat{s}_3[\mathbf{0}]$$

Since the effect in the object net is only local, $\widehat{t}_3[t_{12}]$ is not activated. So $w = \widehat{t}_1[\epsilon] \cdot \widehat{t}_2[t_{11}] \cdot \widehat{t}_3[t_{12}]$ is a possible firing sequence for the reference net, but not for the object net system.

From Theorem 11 and the above the following property follows.

Corollary 12. *Let OS be an* Eos. *If μ is reachable from μ_0, then* $\textsc{Rn}(\mu)$ *is reachable from* $\textsc{Rn}(\mu_0)$. *The reverse does not hold in general.*

So, we obtain only a sufficient condition for non-reachability: The marking μ is not reachable from μ_0 whenever $\textsc{Rn}(\mu)$ is not reachable from $\textsc{Rn}(\mu_0)$.

Fortunately, many practical models are *Generalised State Machines* and this sufficient condition can be strengthened to a necessary one for these. An Eos *OS* is a generalised state machine (GSM) iff for all $\widehat{t}$ there is either exactly one place in the preset and one in the postset typed with the object net N or there are no such places:

$$\forall N \in \mathcal{N} : \forall \widehat{t} \in \widehat{T} : \left|\{\widehat{p} \in {}^\bullet\widehat{t} \mid d(\widehat{p}) = N\}\right| = \left|\{\widehat{p} \in \widehat{t}^\bullet \mid d(\widehat{p}) = N\}\right| \leq 1 \tag{7}$$

and the inital marking has at most one net-token of each type:

$$\forall N \in \mathcal{N} : \sum_{\widehat{p} \in \widehat{P}, d(\widehat{p}) = N} \Pi^1(\mu)(\widehat{p}) \leq 1 \tag{8}$$

Obviously every p/t-like Eos is a generalised state machine since $d(\widehat{p}) = \bullet$ for all $\widehat{p}$. In addition generalised state machines are simple Eos.

For generalised state machines we can strengthen Theorem 11.

Theorem 13. *Let OS be an* Eos *with the generalised state machine property.*
A transition $\widehat{\tau}[\theta]$ is activated in OS for (λ, ρ) iff it is in $\textsc{Rn}(OS)$*:*

$$\mu \xrightarrow[OS]{\widehat{\tau}[\theta](\lambda,\rho)} \mu' \quad \iff \quad \textsc{Rn}(\mu) \xrightarrow[\textsc{Rn}(OS)]{\widehat{\tau}[\theta]} \textsc{Rn}(\mu')$$

A generalised state machine *OS* is therefor isomorphic with its reference net $\textsc{Rn}(OS)$.

From a modelling point of view this result is interesting since in many scenarios net-tokens model physical entities which are neither cloned, combined, created nor destroyed. These models therefore have the generalised state machine property. From a more theoretical point of view the correspondence of each generalised state machine *OS* with its reference net $\textsc{Rn}(OS)$ allows to simplify notations considerably – at the price of limiting the expressiveness. For these reasons some formalism, like e.g. [12], [21], or [27], are initially restricted to generalised state machines. For our analysis we have chosen to study the general case to obtain more insights in the models properties and their expressiveness.

5 A short Overview of Nets-within-Nets Formalisms

The idea to use Petri nets as tokens – also called the "nets-within-nets" approach – can be traced back to the early nineties: [1] studies systems where the net-tokens model the partial order of working plans that are executed within some external environment modelled as a Petri net again.

These nets are extended to elementary object net systems (EONS) in [12]. EONS are studied with respect to reference semantics, value semantics, and copy semantics (cf. [2] for a more recent overview). Reference semantics is equivalent to our construction of the *reference net* $\textsc{Rn}(OS)$. The value semantics of EONS defined in

[12] is defined for the special cases of unary Eos and the class of generalised state machines.

There is also some connection to recursive Petri nets (RPN) [14] where the firing of transitions can generate sub-net activity recursively. This nested threads of activity look somehow similar with a nesting of markings. The most obvious difference between RPN and Eos is the fact that the reachability problem is decidable for RPN (cf. Theorem 17 in [14]) but undecidable for Eos (cf. Theorem 19).

Another variant of nets-within-nets is the the formalism of PN^2 [20] which allows to have several object nets within one system net and is mainly the same as Eos with a copy semantics.

Mobile Systems [25] introduce another extension to object nets: modules that can interact via place and transition fusion. Modules describe locations and locations may be nested. Sub modules may shift from one module to another.

There are several formalism that extend the elementary case to systems with unbounded nesting. A first extension – called object nets – is defined in [4]. It can be shown that for an appropriate extension of the GSM property value and reference semantics are equivalent, too [22]. It is shown in [4] that object nets have the power to simulate Turing machines.

A very intersting restriction comes from the area of workflow nets: Adaptive workflow nets (AWFN) [27] restrict themselves to GSM and the net-tokens to workflow nets. The formalism is extended by the possibility to combine net-tokens at firing time with the usual workflow operations, like sequential composition, and-forks and or-decisions. Due to its restricted structure this formalism has some nice decidability properties.

Object nets are restricted in the sense that different levels of the system may synchronise, but cannot exchange markings directly. In [24] we defined the general case, i.e. object nets extended with communication channels, are defined. Of course, this extension cannot extend the expressibility any further, i.e. beyond Turing machines.

Similarly to object nets with communication channels, [21] defines object nets that allow to exchange object nets over communication channels, but they are restricted to the GSM case.

Reference nets [19] are the generalisation of Eos for the case of arbitrary pointer structures. The semantics is based on graph rewriting while Eos use term rewriting. Reference nets are supported by a very popular tool, called RENEW [31].

Another extension considers coloured tokens: Nested nets [17] can be seen as the extension of object nets in the direction of Coloured Petri Nets, i.e. we have tokens that are nets and tokens that are integers etc. Another example are mobile predicate/transition nets [18] which are object nets under reference semantics and predicates as tokens. AHO systems [26] allow very complex data types as tokens which can be used to encode net-tokens as data types. This encoding works nicely for two levels but it seems that this cannot extended without further some clever, indirect coding.

An interesting extension of objects nets – discussed in [29] – allows algebraic operations on the net-tokens. This formalism therefore subsumes Adaptive workflow nets [27] as well as object nets with communication channels [24].

The formalism of minimal object-based nets (MOB nets) of [15, 16] is related to Eos but with quite different basic assumptions: MOB nets do not make any assumptions about the structure of the tokens; tokens just have a unique identity which can be compared with other identities and MOB nets can generate new tokens having fresh identifiers. It is shown in [15] that these minimal assumptions are sufficient to show that reachability is undecidable for MOB nets – and therefore for every formalism dealing with name creation. Nevertheless it can be shown that boundedness remains decidable [16]. These results are the same for Eos (cf. Thm. 17).

The formalism of ν-abstract Petri nets [28] has the same ability to create fresh names and has therefore also an undecidable reachability problem.

6 Decidability Problems for EOS

The interesting part in the firing rule of Eos is the fact that moving an object net-token in the system net has the power to modify the state of an unbounded number of tokens, i.e. all the tokens of the object net-tokens (including the case of zero tokens). It is therefore a natural question whether this increases the expressiveness of Eos compared to p/t nets. Here we consider the most well known decidability problems for Petri nets: The reachability, the liveness and the boundedness problem.

For the reachability problem one has to decide whether $\mathbf{m}_1 \xrightarrow{*} \mathbf{m}_2$ for a given p/t net N and two markings $\mathbf{m}_1$ and $\mathbf{m}_2$. Reachability has been studied for p/t nets and for variants of object nets: The reachability problem for p/t nets is studied in [32].

For the liveness problem one has to decide whether all transitions of a given p/t net N and its initial marking $\mathbf{m}_0$ are live. A transition t is live if for all markings $\mathbf{m}$ reachable from $\mathbf{m}_0$ there exists one marking $\mathbf{m}'$ reachable from $\mathbf{m}$ that enables t:

$$\forall \mathbf{m} \in RS(\mathbf{m}_0) : \exists \mathbf{m}' \in RS(\mathbf{m}) : \mathbf{m}' \xrightarrow{t}$$

Since it is known for a long time that reachability and liveness are equivalent problems (cf. Theorem 5.5 and 5.6 in [33]) the question whether both are decidable or not was open for several years. Decidability of reachability for p/t nets is shown in [34], a different proof is given in [35].

In [15] it is shown that reachability is undecidable for Petri nets that can arbitrarily create fresh object identities. Note, that Eos do not have identities. [36] studies the relationship of elementary object nets with decidability of deduction in Linear Logic fragments. Decidability questions concerning object nets with coloured tokens, called nested nets, are studied in [37].

Boundedness is the problem to decide whether there are only finitely many reachable markings. The boundedness problem is decidable for p/t nets [38] which is due to the fact that p/t nets enjoy *monotonicity*: If transition t is enabled in marking m_1 then it is also enabled for each greater marking m_2. Formally:

$$\forall \mathbf{m}_1, \mathbf{m}_1', \mathbf{m}_2 : (\mathbf{m}_1 \xrightarrow{t} \mathbf{m}_1' \land \mathbf{m}_1 < \mathbf{m}_2) \Longrightarrow (\exists \mathbf{m}_2' : \mathbf{m}_2 \xrightarrow{t} \mathbf{m}_2' \land \mathbf{m}_1' < \mathbf{m}_2')$$

Here $<$ denotes the strict order on multisets. This fact leads to the construct of a coverability graph which is always finite and which is expressive enough to identify the unbounded places. The boundedness is decidable for some extensions of Petri nets – like Post-SM Nets and Transfer Nets – and undecidable for Reset Nets, Inhibitor Nets, and Self-Modifying Nets (cf. [39] for details).

It is well-known that it is undecidable whether a counter program with at least two counters will terminate. Eos can simulate counter programs, i.e. for each configuration C of the counter program there is a nested marking of the Eos $\mu(C)$ and program execution is equivalent to firing.

Lemma 14. *Each computation of a counter program CP is bisimulated by a firing sequence of OS(CP). Each command cmd_k is bisimulated by a sequence $\sigma(cmd_k)$:*

$$C_0 \xrightarrow[CP]{cmd_k} C \iff \mu(C_0) \xrightarrow[OS(CP)]{\sigma(cmd_k)} \mu(C)$$

Due to this strong simulation we obtain the following undecidabilty results.

Theorem 15. *Reachability, boundedness, and coverability are undecidable for* EOS.

6.1 Simple EOS and Boundedness

The expressiveness of EOS was due to non-simple typing. The typing d is called simple if for all t we have that each place $\widehat{p}$ in its preset there is a place $\widehat{p}'$ in its postset typed with the same net, i.e. $d(\widehat{p}) = d(\widehat{p}')$.

If this condition is violated, i.e. we have a system net transition $\widehat{t}$ such that $N \in d(^\bullet\widehat{t})$ and $N \notin d(\widehat{t}^\bullet)$ for some object net N, then firing of this transition enforces net-tokens of type N to be unmarked (emptiness constraint): The event $\widehat{\tau}[\theta]$ is enabled in mode (λ, ρ) only if all object nets in λ of this type N carry the empty marking: $\Pi^2_N(\lambda) = \mathbf{0}$ (cf. the definition of the enabling predicate ϕ in (5)). In other words: The system net cannot destroy the tokens within a net-token.[6]

The symmetric case is unproblematic: A transition $\widehat{t} \in \widehat{T}$ with an object net N that is present in the postset, but not in the preset, i.e. $N \notin d(^\bullet\widehat{t})$ and $N \in d(\widehat{t}^\bullet)$ generates net-tokens of type N. The firing rule ensures that these net-tokens carry the empty marking since in this case $\widehat{\tau}[\theta]$ is enabled in mode (λ, ρ) only if all object nets in ρ of this type N carry the empty marking.

We like to restrict EOS to simple ones since the emptiness constraint destroys monotonicity properties of the firing rule. We define the order $\preceq$ on nested markings by:

$$\alpha \preceq \beta \quad \iff \quad \begin{array}{l} \alpha = \sum_{i=1}^m \widehat{a}_i[A_i] \wedge \beta = \sum_{i=1}^n \widehat{b}_i[B_i] \wedge \\ \forall 1 \leq i \leq m : \widehat{a}_i = \widehat{b}_i \wedge A_i \leq B_i \end{array} \tag{9}$$

It is clear that for a transition $\widehat{t}$ such that $N \in d(^\bullet\widehat{t})$ and $N \notin d(\widehat{t}^\bullet)$ for some object net N, the enabling of $\widehat{\tau}[\theta]$ in the mode (λ, ρ) does *not* imply the enabling of $\widehat{\tau}[\theta]$ in all (λ', ρ') with $\lambda \preceq \lambda'$, i.e. the firing rule for general EOS is not monotous.

Therefore we have to forbid typings where $N \in d(^\bullet\widehat{t})$ and $N \notin d(\widehat{t}^\bullet)$ for some transition $\widehat{t}$, i.e. we restrict EOS to simple ones.

Given the representation above for $\alpha \preceq \beta$, then $\sum_{i=1}^m \widehat{a}_i[B_i]$ is called a α-*restriction of* β. In general there are many restrictions since the sum representation of α as $\sum_{i=1}^m \widehat{a}_i, [A_i]$ and β as $\sum_{i=1}^n \widehat{b}_i[B_i]$ are not unique. Let $(\beta \downarrow \alpha)$ denote the set of all α-restrictions of β. Let α and β be arbitrary nested multisets with $\alpha \preceq \beta$. Then we have:

$$\forall \alpha, \beta \in \mathcal{M} : \alpha \preceq \beta \Longrightarrow \forall \gamma \in (\beta \downarrow \alpha) : \alpha \preceq \gamma \preceq \beta \wedge \Pi^1(\alpha) = \Pi^1(\gamma) \leq \Pi^1(\beta) \tag{10}$$

Lemma 16. *For* EOS *with simple typing d the firing rule is monotonous, w.r.t. the order $\preceq$. If the event $\widehat{\tau}[\theta]$ is enabled then it is enabled for each greater marking:*

$$(\forall \mu_1, \mu_1', \mu_2 : \mu_1 \xrightarrow[OS]{\widehat{\tau}[\theta]} \mu_1' \wedge \mu_1 \prec \mu_2) \Longrightarrow (\exists \mu_2' : \mu_2 \xrightarrow[OS]{\widehat{\tau}[\theta]} \mu_2' \wedge \mu_1' \prec \mu_2')$$

From this monotonicity one can generalise the result of [38].

Theorem 17. *Boundedness and Coverability are decidable for simple* EOS.

The markings of an EOS have a bounded nesting depth. For general object nets there is no bound for the nesting depth and we know that the argument as given in Theorem 17 cannot be applied since we know that even simple object nets with

[6]Remark: If the system net would be able to destroy the net-token's tokens then is quite obvious that we can simulate reset nets i.e. Petri nets with reset arcs. It is known that reachability and boundedness is undecidable for reset nets while e.g. coverability of markings remains decidable showing that reset nets are weaker than inhibitor nets (cf. [39]). Since Nested Nets have the possibility to destroy tokens it is clear that Nested Nets can simulate reset nets (cf. [37]).

unbounded nesting depth are able to simulate counter programs [24]. The reasons for this lies in the fact that$\prec$ fails to be a wqo for unbounded nesting since we may have infinitly many incomparable markings.

Surprisingly, one can show that reachability remains undecidable even if we restrict Eos to simple typings.

Lemma 18. *Each computation of CP is simulated weakly by a firing sequence of the simple* Eos $OS_{weak}(CP)$ *in the following sense:*

$$C_0 \xrightarrow[CP]{*} C \iff \mu(C_0) \xrightarrow[OS_{weak}(CP)]{*} \mu(C)$$

The simulation is called *weak* since there are several firing sequences due to non-deterministic choices and only some of them correspond to counter program execution. The main difference of this simulation compared to that of Lemma14 is that $\mu(C_0)$ also enables firing sequences that do not correspond to an execution of the counter program, i.e. the simulation can make wrong "guesses". Fortunately, it is guarantued that no such "wrong" sequence will ever generate a marking of the form $\mu(C)$ for some C again. Since we can encode the test, whether for some given marking μ there is some configuration C such that $\mu = \mu(C)$ holds, directly as a property of the marking μ, this weak simulation suffices to establish the undecidability of the reachability problem.

Theorem 19. *Reachability is undecidable for simple* Eos*. It is even undecidable for pure* Eos *with simple typing and undecidable for minimal* Eos *with simple typing.*

It is an open question whether the reachability problem is decidable for simple, minimal and pure (i.e. unary) Eos.

Due to the weak form of simulation not everything becomes undecidable for Eos with simple typing. In fact, as shown in Theorem 17 boundedness is decidable for simple Eos.

6.2 The Liveness Problem for EOS

We define the liveness problem for Eos analogously to that of p/t nets: For the liveness problem one has to decide whether all events $\theta \in \Theta$ of a given Eos OS are live. An event θ is live if for all markings μ reachable from μ_0 there exists a marking μ' reachable from μ that enables θ.

Theorem 20. *Liveness is undecidable for* Eos *– even when restricted to simple* Eos*. It is undecidable even for pure* Eos *as well as for minimal* Eos*.*

The proof uses a modification of the Eos $OS_{weak}(CP)$ of Lemma 18. It is shown that if we can decide liveness for of a special event, then we can decide proper termination with empty counters which is an undecidable problem.

7 Conclusion

This papers studies the Petri net formalism of elementary object nets (Eos). Object nets are Petri nets which have Petri nets as tokens. The general formalism of objects nets allows arbitrarily nested nets (cf. [4, 22, 23, 24] for details).

Eos are called elementary since the nesting is restricted to two levels only. Interestingly enough, even for the restricted class of elementary object nets the reachability and the boundedness problems are undecidable. Even for the class of simple Eos where boundedness is decidable the reachability and the liveness problem remain undecidable.

Our view with respect to decidability questions is complemented to a comlexity perspective in [40] where we studied bounded Eos and the complexity of its reachability problem.

An interesting extension of objects nets – discussed in [29] – allows algebraic operations on the net-tokens, like sequential or parallel composition. This is a concise way to express the self-modification of net-tokens at run-time in an algebraic setting.

References

[1] Valk, R.: Modelling concurrency by task/flow EN systems. In: 3rd Workshop on Concurrency and Compositionality. Number 191 in GMD-Studien, St. Augustin, Bonn, Gesellschaft für Mathematik und Datenverarbeitung (1991)

[2] Valk, R.: Object Petri nets: Using the nets-within-nets paradigm. In Desel, J., Reisig, W., Rozenberg, G., eds.: Advanced Course on Petri Nets 2003. Volume 3098 of Lecture Notes in Computer Science., Springer-Verlag (2003) 819–848

[3] Köhler, M., Rölke, H.: Concurrency for mobile object-net systems. Fundamenta Informaticae **54** (2003)

[4] Köhler, M., Rölke, H.: Properties of Object Petri Nets. In Cortadella, J., Reisig, W., eds.: International Conference on Application and Theory of Petri Nets 2004. Volume 3099 of Lecture Notes in Computer Science., Springer-Verlag (2004) 278–297

[5] Köhler, M., Moldt, D., Rölke, H.: Modeling the behaviour of Petri net agents. In Colom, J.M., Koutny, M., eds.: International Conference on Application and Theory of Petri Nets. Volume 2075 of Lecture Notes in Computer Science., Springer-Verlag (2001) 224–241

[6] Köhler, M., Moldt, D., Rölke, H.: Modelling mobility and mobile agents using nets within nets. In v. d. Aalst, W., Best, E., eds.: International Conference on Application and Theory of Petri Nets 2003. Volume 2679 of Lecture Notes in Computer Science., Springer-Verlag (2003) 121–140

[7] Petri, C.A.: Introduction to general net theory. In Brauer, W., ed.: Net Theory and its applications. Proceedings of the Advanced course on general net theory of processes and systems. Volume 84 of Lecture Notes in Computer Science., Springer-Verlag (1979)

[8] Peterson, J.L., Silberschatz, A.: Operating System Concepts. Addison-Wesley Publishing Company, Reading, Massachusetts (1985) Second edition.

[9] Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes, parts 1-2. Information and computation **100** (1992) 1–77

[10] Cardelli, L., Gordon, A.D., Ghelli, G.: Mobility types for mobile ambients. In: Proceedings of the Conference on Automata, Languages, and Programming (ICALP'99). Volume 1644 of Lecture Notes in Computer Science., Springer-Verlag (1999) 230–239

[11] Vitek, J., Castagna, G.: Seal: A framework for secure mobile computations. In: ICCL Workshop: Internet Programming Languages. (1998) 47–77

[12] Valk, R.: Petri nets as token objects: An introduction to elementary object nets. In Desel, J., Silva, M., eds.: Application and Theory of Petri Nets. Volume 1420 of Lecture Notes in Computer Science. (1998) 1–25

[13] Busi, N.: Mobile nets. In Ciancarini, P., Fantechi, A., Gorrieri, R., eds.: Formal Methods for Open Object-Based Distributed Systems. Volume 139., Kluwer (1999) 51–66

[14] Haddad, S., Poitrenaud, D.: Theoretical aspects of recursive Petri nets. In Donatelli, S., Kleijn, J., eds.: Proceedings of the 20th International Conference on Application and Theory of Petri Nets. Volume 1639 of Lecture Notes in Computer Science., Springer-Verlag (1999) 228–247

[15] Kummer, O.: Undecidability in object-oriented Petri nets. Petri Net Newsletter **59** (2000) 18–23

[16] Kummer, O., Dietze, R., Kudlek, M.: Decidability problems of a basic class of object nets. Fundamenta Informaticae **79** (2008) 295–302

[17] Lomazova, I.A.: Nested Petri nets – a formalism for specification of multi-agent distributed systems. Fundamenta Informaticae **43** (2000) 195–214

[18] Xu, D., Deng, Y.: Modeling mobile agent systems with high level Petri nets. In: IEEE International Conference on Systems, Man, and Cybernetics'2000. (2000)

[19] Kummer, O.: Referenznetze. Logos Verlag (2002)

[20] Hiraishi, K.: PN^2: An elementary model for design and analysis of multi-agent systems. In Arbab, F., Talcott, C.L., eds.: Coordination Models and Languages, COORDINATION 2002. Volume 2315 of Lecture Notes in Computer Science., Springer-Verlag (2002) 220–235

[21] Bednarczyk, M.A., Bernardinello, L., Pawlowski, W., Pomello, L.: Modelling mobility with Petri hypernets. In Fiadeiro, J.L., Mosses, P.D., Orejas, F., eds.: Recent Trends in Algebraic Development Techniques (WADT 2004). Volume 3423 of Lecture Notes in Computer Science., Springer-Verlag (2004) 28–44

[22] Köhler, M., Rölke, H.: Reference and value semantics are equivalent for ordinary Object Petri Nets. In Darondeau, P., Ciardo, G., eds.: International Conference on Application and Theory of Petri Nets 2005. Volume 3536 of Lecture Notes in Computer Science., Springer-Verlag (2005) 309–328

[23] Köhler, M., Farwer, B.: Modelling global and local name spaces for mobile agents using object nets. Fundamenta Informaticae **72** (2006) 109–122

[24] Köhler-Bußmeier, M., Heitmann, F.: On the expressiveness of communication channels for object nets. Fundamenta Informaticae **93** (2009) 205–219

[25] Lakos, C.: A Petri net view of mobility. In: Formal Techniques for Networked and Distributed Systems (FORTE 2005). Volume 3731 of Lecture Notes in Computer Science., Springer-Verlag (2005) 174–188

[26] Hoffmann, K., Ehrig, H., Mossakowski, T.: High-level nets with nets and rules as tokens. In: Application and Theory of Petri Nets and Other Models of Concurrency. Volume 3536 of Lecture Notes in Computer Science., Springer-Verlag (2005) 268 – 288

[27] Lomazova, I.A., van Hee, K.M., Oanea, O., Serebrenik, A., Sidorova, N., Voorhoeve, M.: Nested nets for adaptive systems. In: Application and Theory of Petri Nets and Other Models of Concurrency. Lecture Notes in Computer Science, Springer-Verlag (2006) 241–260

[28] Velardo, F.R., de Frutos-Escrig, D.: Name creation vs. replication in petri net systems. Fundam. Inform. **88** (2008) 329–356

[29] Köhler-Bußmeier, M.: Hornets: Nets within nets combined with net algebra. In Wolf, K., Franceschinis, G., eds.: International Conference on Application and Theory of Petri Nets (ICATPN'2009). Volume 5606 of Lecture Notes in Computer Science., Springer-Verlag (2009) 243–262

[30] Köhler, M.: Objektnetze: Definition und Eigenschaften. Logos Verlag, Berlin (2004)

[31] Kummer, O., Wienberg, F., Duvigneau, M., Schumacher, J., Köhler, M., Moldt, D., Rölke, H., Valk, R.: An extensible editor and simulation engine for Petri nets: Renew. In Cortadella, J., Reisig, W., eds.: International Conference on Application and Theory of Petri Nets 2004. Volume 3099 of Lecture Notes in Computer Science., Springer-Verlag (2004) 484 – 493

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

[33] Peterson, J.: Petri Net Theory and the Modeling of Systems. Prentice Hall Inc., Englewood Cliffs NJ (1981)

[34] Mayr, E.W.: An algorithm for the general Petri net reachability problem. SIAM Journal Computation **13** (1984) 441–460

[35] Lambert, J.L.: A structure to decide the reachability in Petri nets. Theoretical

Computer Science **99** (1992) 79–104

[36] Farwer, B.: A linear logic view of object Petri nets. Fundamenta Informaticae **37** (1999) 225–246

[37] Lomazova, I.A., Schnoebelen, P.: Some decidability results for nested Petri nets. In: International Conference on Perspectives of System Informatics (PSI'99). Volume 1755 of Lecture Notes in Computer Science., Springer-Verlag (2000) 208–220

[38] Karp, R.M., Miller, R.E.: Parallel program schemata. Journal of Computer and System Sciences **3** (1969) 147–195

[39] Dufourd, C., Finkel, A., Schnoebelen, P.: Reset nets between decidability and undecidability. In Larsen, K., ed.: Automata, Languages, and Programming (ICALP'98). Volume 1443 of Lecture Notes in Computer Science., Springer-Verlag (1998) 103–115

[40] Köhler-Bußmeier, M., Heitmann, F.: Safeness for object nets. Fundamenta Informaticae (2010) To appear.