Technische Universität Berlin
Fachbereich Informatik
Fachgebiet Theoretische Informatik/Formale Spezifikation

Comparison of Mobile Elementary Object Systems with Algebraic Higher Order Nets

Bachelorarbeit
Geprüft von: Prof. Dr. Hartmut Ehrig
und Prof Dr. Julia Padberg

Angefertigt von
Lydia Mattick
Matrikelnummer: 312278

Eidesstattliche Erklärung

Die selbständige und eigenhändige Ausfertigung versichert an Eides statt
Berlin, den 17. Juni 2010.

Unterschrift

Abstract

Algebraic higher order (AHO) nets and mobile elementary object systems (mobile EOS) can be used to model processes for many applications. They are especially suitable for two layered environments with interaction between independent layers.

Both follow the nets as tokens paradigm but utilize different formalisms to describe the system nets. Since they apply different approaches, they vary in their firing behavior. The name spaces, used in mobile EOS, introduce a new concept for object nets to interact with the system net. By granting access to synchronization in some and denying it in other places, mobile EOS cover a new field of applications.

With the help of an example, we recreate a mobile EOS using an AHO net with Petri nets as tokens. We will furthermore demonstrate a way for the created AHO net to imitate mobile EOS behavior including an examination of the consequences introduced by name spaces.

Even though this work does not deliver proper formalisms to convert mobile EOS to AHO nets in general, it gives an insight on how to introduce name spaces into AHO nets.

Abstract

Algebraic higher order (AHO) Netze und mobile elementary object systems (mobile EOS) können zur Modellierung von Prozessen für viele Anwendungen genutzt werden. Sie eignen sich besonders für zweischichtige Umgebungen mit Interaktion zwischen unabhängigen Schichten.

Beide folgen dem ”Netze als Token”-Paradigma, nutzen aber verschiedene Formalismen, um das Systemnetz zu beschreiben. Da sie unterschiedliche Ansätze verwenden, variieren sie in ihrem Schaltverhalten. Die Namensbereiche, die in mobilen EOS verwendet werden, stellen ein neues Konzept für Interaktion von Objektnetzen mit dem Systemnetz dar. Dadurch, dass der Zugang zur Synchronisation in einigen Orten gewährt und in anderen Orten verhindern werden kann, erobern mobile EOS ein neues Feld von Anwendungen.

Mit Hilfe eines Beispiels erstellen wir für ein mobiles EOS ein AHO Netz mit Petri-Netzen als Token. Außerdem zeigen wir eine Möglichkeit, wie dieses geschaffene AHO Netz das Verhalten des mobilen EOS einschließlich der Bewältigung der Folgen von Namensräumen nachahmen kann.

Auch wenn diese Arbeit keine ordnungsgemäßen Formalismen liefert, um mobile EOS im Allgemeinen zu AHO Netzen zu konvertieren, gibt sie einen Einblick darüber, wie Namensräume in AHO Netzen eingeführt werden können und dass es möglich ist, das Verhalten von mobilen EOS zu imitieren.

Contents

Chapter 1

Introduction

Elementary object systems (EOS) as introduced by Valk in [Val98] are used to model systems using place/transition nets (P/T nets). The main innovation of this approach is the establishment of a two-layered modeling technique, allowing one system net and several object nets. In [Val98] Valk describes two different semantics for this concept, value and reference semantics. Both of the semantics show advantages as well as, in some cases, undesired properties.

Value semantics, when firing a forking system net transition, create copies of object nets. The tokens of the original object net are distributed between the created nets. This means each place constitutes a name space of its own and the location of tokens is unmistakably defined after distribution. When firing a transition of the object net, changes only affect one copy of the net.

Reference semantics do not copy nets, but use references to the same object net. By applying these semantics, the whole system net represents one name space, which means that by firing an object net transition, the effect is visible in all places where references are present.

Mobile elementary object systems (mobile EOS) combine all properties of both semantics into one, while introducing name spaces. That way object nets can be referred to, within one name space, and copied, while being transitioned to another. The introduction of name spaces allows for new environments to be modeled. A mobile agent moving between buildings while having access to different services inside them can be mentioned as one example. [KF07]

EOS and mobile EOS both use P/T nets as their framework, and therefore use set theoretical expressions to model courses of events. Algebraic higher-order nets are described in a formal way (as in [Hof05]). The use of algebraic modeling of the system net offers flexibility and does not restrict it to the methods provided by P/T nets. Simulating EOS, using algebraic higher order (AHO) nets, not only provides a common ground for comparison purposes, but allows enhancements which cannot be accomplished by remaining in the P/T net framework.

While AHO nets with P/T nets as tokens have been studied (for example) in [HEM05] and [HERP08] and the translation of EOS to AHO nets has been shown in [Hof05],

mobile EOS have not been related to AHO nets yet. In this thesis we imitate a mobile EOS with an AHO net using an easily comprehended example. By means of this example, we analyze the structure and firing behavior of both systems, showing exemplarily their equivalence.

We start by giving the definition for P/T nets and defining the object nets used by both, AHO nets and mobile EOS 2. In the following chapter 3 we define the EOS needed to assemble the mobile EOS while citing the definitions. The mobile EOS example used for the comparison is created similarly in the next chapter 4. We assemble the AHO net while giving the extensions needed to cover mobile EOS behavior in chapter 6 after recalling the definitions in chapter 5. In chapter 7 we compare the created examples.

Chapter 2

Place/Transition Nets

Since both the mobile EOS as well as the AHO nets with place/transition nets (P/T nets) as tokens instrumentalize P/T nets, we start by recalling the according notations as given by Köhler in [KF07]. In this chapter we will construct the object nets used by our EOS and AHO net example.

2.1 Definition

A P/T net is a tuple $N = (P, T, \mathbf{pre}, \mathbf{post})$, such that:

1. P is a set of places.
2. T is a set of transitions, with $P \cap T = \emptyset$.
3. $\mathbf{pre}, \mathbf{post} : T \to MS(P)$ are the pre- and post domain functions.

A marking[1] of N is a multiset of places: $\mathbf{m} \in MS(P)$.
A P/T net with initial marking $\mathbf{m}$ is denoted $N = (P, T, \mathbf{pre}, \mathbf{post}, \mathbf{m})$.
A transition $t \in T$ is enabled in $\mathbf{m}$ if and only if: $\mathbf{m} \geq \mathbf{pre}(t)(p)$ holds.
The follower marking $\mathbf{m}$ ' is given by $\mathbf{m}' = \mathbf{m} - \mathbf{pre}(t)(p) + \mathbf{post}(t)(p)$.

The net N_H is a P/T net with one token on the place $h.3.eos$.

$$
\begin{aligned}
N_H = \quad & (\{h.1.eos, h.2.eos, h.3.eos\}, \\
& \{h.pet.eos, h.fill.eos, h.out.eos, h.stf.eos, h.treat.eos\}, \\
& \{(h.pet.eos, h.1.eos), (h.fill.eos, h.2.eos), (h.out.eos, h.3.eos), \\
& (h.stf.eos, h.3.eos), (h.treat.eos, h.1.eos)\}, \\
& \{(h.pet.eos, h.2.eos), (h.fill.eos, h.3.eos), (h.out.eos, h.1.eos), \\
& (h.stf.eos, h.3.eos), (h.treat.eos, h.1.eos)\}, \\
& 1h.3.eos)
\end{aligned}
$$

[1] For the definition of EOS and mobile EOS markings and pre- and post domain functions use multisets [KF07] while the P/T nets used in the AHO net definition in [HEM05] use a free commutative monoid over the set of places. We assume here that these two notations are equivalent.

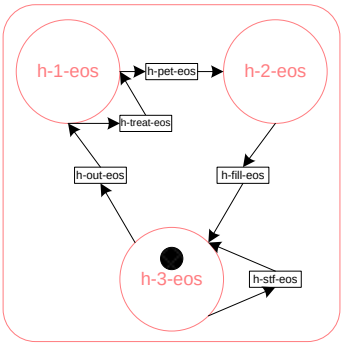

Figure 2.1: *Human object net N_H*

The transition $h.stf.eos$ is enabled because

$$\mathbf{m} = 1h.3.eos \geq 1h.3.eos = \mathbf{pre}\ (t)(p).$$

The follower marking results in

$$\mathbf{m}' = \mathbf{m} - \mathbf{pre}\ (t)(p) + \mathbf{post}\ (t)(p) = 1h.3.eos - 1h.3.eos + 1h.3.eos = 1h.3.eos.$$

The net N_P is a P/T net with one token on the place $p.1.eos$.

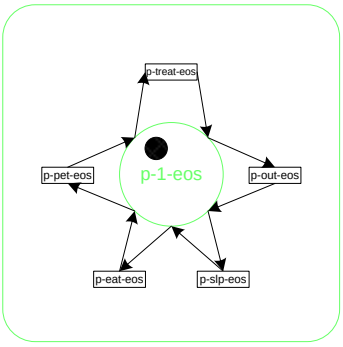

Figure 2.2: *Pet object net N_P*

$$\begin{aligned} N_P = \quad & (\{(p.1.eos\}, \\ & \{p.slp.eos, p.out.eos, p.eat.eos, p.pet.eos, p.treat.eos\}, \\ & \{(p.slp.eos, p.1.eos), (p.eat.eos, p.1.eos), (p.pet.eos, p.1.eos), \\ & (p.out.eos, p.1.eos), (p.treat.eos, p.1.eos)\}, \\ & \{(p.slp.eos, p.1.eos), (p.eat.eos, p.1.eos), (p.pet.eos, p.1.eos), \\ & (p.out.eos, p.1.eos), (p.treat.eos, p.1.eos)\}, \\ & 1p.1.eos) \end{aligned}$$

For the P/T net N_P all transitions are enabled with this initial marking.

Chapter 3

Elementary Object Systems

Elementary object systems (EOS) portray a way to model two-layered environments. The object nets can be viewed as a dynamic refinement of places or in a more application based view, they can be seen as agents acting in the system net environment. Valk [Val98] proposed two kinds of semantics for EOS, as mentioned in the introduction. In [KF07] Köhler introduced mobile EOS using the EOS definition cited here, which allows object nets to access tokens of other object nets throughout the whole system net. After firing a system net transition, tokens are distributed between object nets of the same type. Therefore the semantics allow a reference like acces but with tokens distributed on different copies of the object nets.

3.1 Definition

An EOS is a tuple $OS = (\hat{N}, \mathscr{N}, d, \theta, \mu_0)$,such that:

1. $\hat{N}$ is a P/T net, called *system net*.
2. $\mathscr{N}$ is a finite set of P/T nets, called *object nets*.
3. $d : \hat{P} \to \{\bullet\} \cup \mathscr{N}$ is a monotonous *typing* of the system net places.
4. $\theta \subseteq \mathscr{T}$ is a finite *synchronisation structure*.
5. $\mu_0 \in \mathscr{M}_{\mathscr{N}}$ is the *initial marking*.

3.1.1 Typing

While $d(\hat{p})$ returns the token type of a system net place, $d^{-1}(N)$ with $N \in \mathscr{N}$ returns the set of all the system net places typed for object nets of type N.

$d^{-1}(\bullet)$ returns the set of all system net places typed for black tokens.

To ensure monotony of the firing rule, monotonous typing is required. This means that for each object net token in the pre-domain of a system net transition there is a place typed accordingly in the post domain of that transition. The same has to hold for black tokens.

3.1.2 Marking

$\mathscr{M}_{\mathscr{N}} = MS((d^{-1}(\bullet) \times 0) \bigcup \cup_{N \in \mathscr{N}} (d^{-1}(N) \times MS(P_N)))$ is the set of all possible markings of an EOS.
A marking $\mu \in \mathscr{M}_{\mathscr{N}}$ is denoted by $\mu = \sum_{k=1}^{|\mu|} (\hat{p}_k, M_k)$.

3.1.3 Events

$\varepsilon_{\hat{p}}$ is the idle system net transition for the place $\hat{p} \in \hat{P}$. Therefore in each place of the system net object nets can fire transitions, if the synchronization structure allows it.
ε_N is defined as the idle object net transition for object nets of the type $N \in \mathscr{N}$. With the idle object net transition, it is possible to determine object nets which do not take part in events. Both types of idle transitions are needed in order to give the events a universal structure.

$\mathscr{C} = \{C | C : \mathscr{N} \to \bigcup_{N \in \mathscr{N}} (T_N \cup \{\varepsilon_N\})\}$ is the set of all possible mappings C, where each object net is mapped to one of its transitions or its idle transition.
$\mathscr{T} = \{(\hat{\tau}, C) | \hat{\tau} \in \hat{T} \cup \{\varepsilon_{\hat{p}} | \hat{p} \in \hat{P}\} \wedge C \in \mathscr{C}\} \setminus \{(\varepsilon_{\hat{p}}, \varepsilon_C) | \hat{p} \in \hat{P}\}$ is the set of all possible events. An event is a pair $(\hat{\tau}, C)$, providing one transition or the idle transition for the system net and each object net type.

3.1.4 Projections

$\prod^1(\sum_{k=1}^{|\mu|}(\hat{p}_k, M_k)) := \sum_{k=1}^{|\mu|} \hat{p}_k$ is the first projection, which returns the system net marking exclusively.
$\prod_N^2(\sum_{k=1}^{|\mu|}(\hat{p}, M_k)) := \sum_{k=1}^{|\mu|} \pi_N(\hat{p}_k) * M_k$ is the second projection, where $\pi_N : \hat{P} \to \{0,1\}$ is an indicator function with $\pi_N(\hat{p}) = 1$ if and only if $d(\hat{p}) = N$. It returns the marking of one object net type throughout the whole system net.

3.1.5 Enabling Predicate

$$\begin{aligned} \Phi \quad & ((\hat{\tau}, C), \lambda, \rho) \Leftrightarrow \prod^1(\lambda) \geq \mathbf{pre}\,(\hat{\tau}) \wedge \prod^1(\rho) \geq \mathbf{post}\,(\hat{\tau}) \\ & \wedge \forall N \in \mathscr{N} : \prod_N^2(\lambda) \geq \mathbf{pre}_N(C(N)) \\ & \wedge \forall N \in \mathscr{N} : \prod_N^2(\rho) = \prod_N^2(\lambda) - \mathbf{pre}_N(C(N)) + \mathbf{post}_N(C(N)) \end{aligned}$$

is the enabling predicate where $\lambda, \rho \in \mathscr{M}_{\mathscr{N}}$ are markings of the whole net.

For system autonomous events, C equals ε_C which means $\forall N \in \mathscr{N} : C(N) = \varepsilon_N$. In this case only a system net transition fires and only the first projection determines whether the transition is enabled or not.

The enabling predicate Φ does not distinguish between markings that coincide in their projections. This means that for enabling a transition, as long as the system net places contain tokens, it does not matter where in the system net the object net tokens are. This can be expressed by defining the equivalence $\cong \subseteq \mathscr{M}_{\mathscr{N}}{}^2$:
$\alpha \cong \beta :\Leftrightarrow \prod^1(\alpha) = \prod^1(\beta) \wedge \forall N \in \mathscr{N} : \prod_N^2(\alpha) = \prod_N^2(\beta)$.

According to this equivalence object net tokens may change their locations at the moment of firing a transition.

3.1.6 Enabling of a Transition

A transition t is enabled in mode $(\lambda, \rho) \in \mathscr{M}_{\mathscr{N}}{}^2$ if the following holds:

$$\lambda \leq \mu \wedge \exists \lambda', \rho' : (\lambda' \cong \lambda) \wedge (\rho' \cong \rho) \wedge \Phi((\hat{\tau}, C), \lambda', \rho')$$

while μ is a marking of an EOS. The successor marking is defined as $\mu' := \mu - \lambda + \rho$.

3.2 Example

3.2.1 System Net

Figure 3.1 shows the system net $\hat{N}$. Colors represent the typing of the system places. Red colored places allow human object nets, green ones allow pet object nets and blue ones contain black tokens.

Monotonous typing is guaranteed by the transitions, shown for black tokens:

$$\exists \hat{p}_1 \in \textbf{pre}\ (eat.eos) : d(\hat{p}_1) = \bullet \Rightarrow \exists \hat{p}_2 \in \textbf{post}\ (eat.eos) : d(\hat{p}_2) = \bullet$$
$$\exists \hat{p}_2 \in \textbf{pre}\ (fill.eos) : d(\hat{p}_2) = \bullet \Rightarrow \exists \hat{p}_1 \in \textbf{post}\ (fill.eos) : d(\hat{p}_1) = \bullet$$

This holds for $\hat{p}_1 = blacktoken.full.eos$ and $\hat{p}_2 = blacktoken.empty.eos$.

$$
\begin{aligned}
\hat{N} \;=\; & \\
\{\;\; & human.outofreach.eos, human.kitchen.eos, human.livingroom.eos, \\
& human.couch.eos, human.outside.eos, pet.kitchen.eos, pet.bowl.eos, \\
& pet.couch.eos, pet.blanket.eos, pet.livingroom.eos, pet.outside.eos, \\
& blacktoken.full.eos, blacktoken.empty.eos \qquad \} \\
\{\;\; & s1.eos, s2.eos, s3.eos, s4.eos, s5.eos, s6.eos, s7.eos, s8.eos, s9.eos, \\
& s10.eos, s11.eos, sa.eos, sb.eos, sc.eos, sd.eos, se.eos, sf.eos, \\
& sg.eos, sh.eos, out.eos, slp.eos, eat.eos, pet.eos, fill.eos, new.eos \qquad \} \\
\{\;\; & (s1.eos, human.couch.eos), (s2.eos, human.livingroom.eos), \\
& (s3.eos, pet.couch.eos), (s4.eos, pet.livingroom.eos), \\
& (s5.eos, pet.blanket.eos), (s6.eos, pet.couch.eos), \\
& (s7.eos, pet.blanket.eos), (s8.eos, pet.livingroom.eos), \\
& (s9.eos, pet.livingroom.eos), (s10.eos, pet.bowl.eos), \\
& (s11.eos, pet.kitchen.eos), (sa.eos, human.outofreach.eos), \\
& (sb.eos, human.livingroom.eos), (sc.eos, human.kitchen.eos), \\
& (sd.eos, human.livingroom.eos), (se.eos, pet.kitchen.eos), \\
& (sf.eos, pet.livingroom.eos), \\
& (sg.eos, human.outside.eos + pet.outside.eos), \\
& (sh.eos, human.outside.eos + pet.outside.eos), \\
& (out.eos, human.outside.eos + pet.outside.eos), \\
& (slp.eos, pet.blanket.eos), \\
& (eat.eos, pet.bowl.eos + blacktoken.full.eos), \\
& (pet.eos, human.couch.eos + pet.couch.eos), \\
& (fill.eos, human.kitchen.eos + blacktoken.empty.eos), \\
& (new.eos, human.outofreach) \qquad \} \\
\{\;\; & (s1.eos, human.livingroom.eos), (s2.eos, human.couch.eos), \\
& (s3.eos, pet.livingroom.eos), (s4.eos, pet.couch.eos), \\
& (s5.eos, pet.couch.eos), (s6.eos, pet.blanket.eos), \\
& (s7.eos, pet.livingroom.eos), (s8.eos, pet.blanket.eos), \\
& (s9.eos, pet.blanket.eos), (s10.eos, pet.kitchen.eos), \\
& (s11.eos, pet.bowl.eos), (sa.eos, human.livingroom.eos), \\
& (sb.eos, human.outofreach.eos), (sc.eos, human.livingroom.eos), \\
& (sd.eos, human.kitchen.eos), (se.eos, pet.livingroom.eos), \\
& (sf.eos, pet.kitchen.eos), \\
& (sg.eos, human.livingroom.eos + pet.livingroom.eospet.livingroom.eos), \\
& (sh.eos, human.outside.eos + pet.outside.eos), \\
& (out.eos, human.outside.eos + pet.outside.eos), \\
& (slp.eos, pet.blanket.eos), \\
& (eat.eos, pet.bowl.eos + blacktoken.empty.eos), \\
& (pet.eos, human.couch.eos + pet.couch.eos), \\
& (fill.eos, human.kitchen.eos + blacktoken.full.eos), \\
& ((new.eos, human.outofreach + human.outofreach)) \qquad \}
\end{aligned}
$$

3.2.2 Object Nets

The set $\mathcal{N}$ consists of N_H and N_P as portrayed in the previous chapter 2.

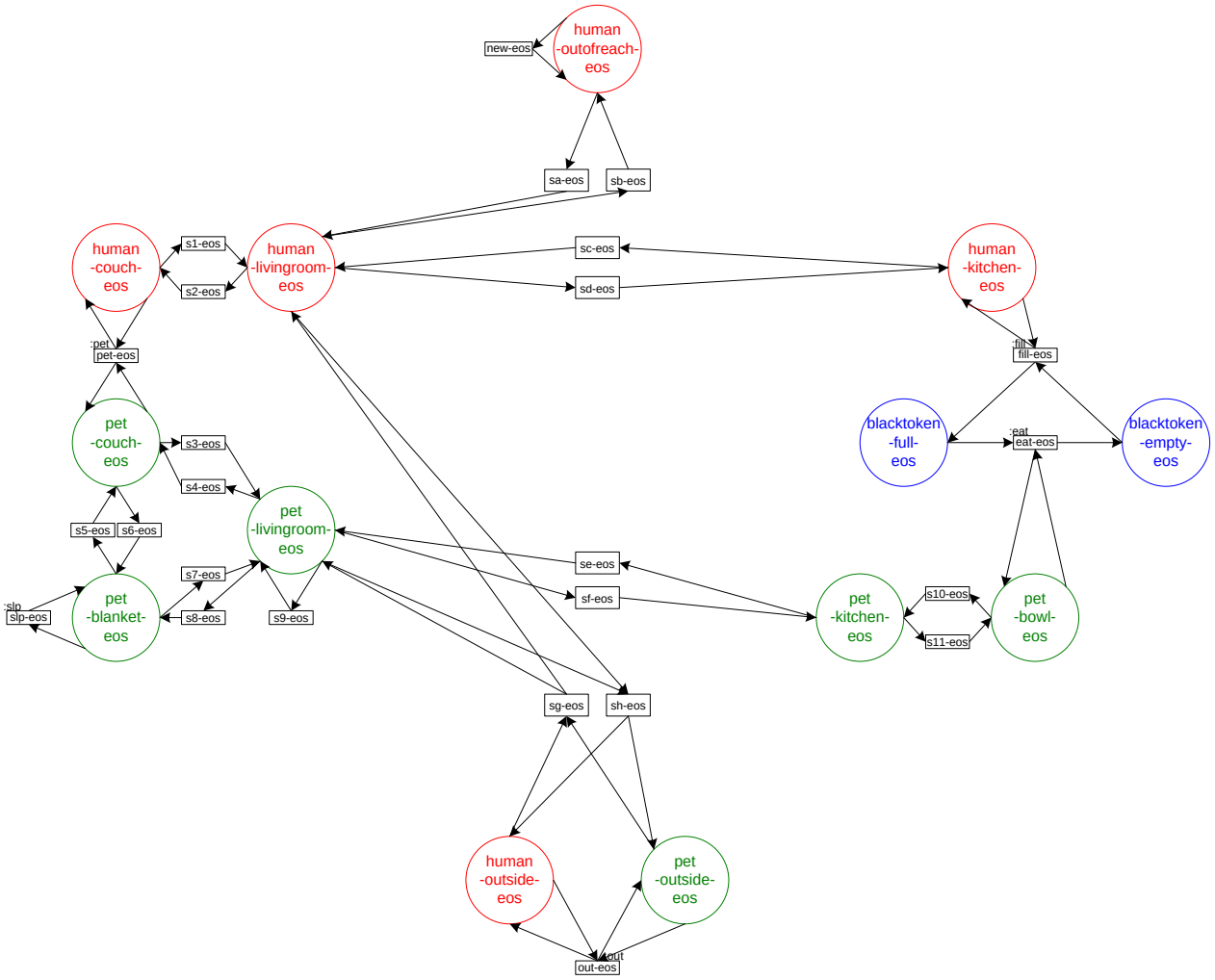

Figure 3.1: *EOS system net $\hat{N}$*

3.2.3 Typing

The typing is graphically represented by colors but can be formalized:

$$\begin{aligned}
N_H &= d(human.kitchen.eos) = d(human.livingroom.eos) \\
&= d(human.outofreach.eos) = d(human.outside.eos) \\
&= d(human.couch.eos) \\
N_P &= d(pet.bowl.eos) = d(pet.couch.eos) \\
&= d(pet.kitchen.eos) = d(pet.livingroom.eos) \\
&= d(pet.outside.eos) = d(pet.blanket.eos) \\
\bullet &= d(blacktoken.full.eos) = d(blacktoken.empty.eos)
\end{aligned}$$

3.2.4 Synchronization Structure

The synchronization structure defines all possible events for the EOS. The events can be categorized into four subsets:

The set of all synchronized events consists of all events, where at least one object net fires synchronously with the system net. In our example the mentioned transitions have distinct names in both the system net and the object nets:

$$\begin{aligned}
SysSyncEvents &= \{(eat.eos, (\varepsilon_{N_H}, p.eat.eos)\} \\
&\cup \{(fill.eos, (h.fill.eos, \varepsilon_{N_P}))\} \\
&\cup \{(out.eos, (h.out.eos, p.out.eos)\} \\
&\cup \{(pet.eos, (h.pet.eos, p.pet.eos)\} \\
&\cup \{(slp.eos, (\varepsilon_{N_H}, p.slp.eos)\}
\end{aligned}$$

The set of all system autonomous events consists of all events, where only the system net fires, or in other words, object nets are transported:

$$SysAutoEvents = \{(\hat{t}, \varepsilon_C) | \hat{t} \in \hat{T} \setminus \{eat.eos, fill.eos, out.eos, pet.eos, slp.eos\}\}$$

The set of all object autonomous events consists of all events, where only one object net fires. In our example this is only the case for the transition $h.stf.eos$ of the human object net. This transition can be fired in every system net place:

$$ObjAutoEvents = \{\varepsilon_{\hat{p}}, h.stf.eos, \varepsilon_{N_P}) | \hat{p} \in \hat{P}\}$$

The set of all object synchronous events consists of all events, where the two object net types fire synchronously but without the system net. In our example this is the case for the transitions $h.treat.eos$ and $p.treat.eos$. This synchronized event can be fired whenever the transitions are enabled in both object nets.

$$ObjSyncEvents = \{\varepsilon_{\hat{p}}, h.treat.eos, p.treat.eos) | \hat{p} \in \hat{P}\}$$

These four sets combined build the synchronization structure:

$$\theta = SysSyncEvents \cup SysAutoEvents \cup ObjAutoEvents \cup ObjSyncEvents$$

3.2.5 Initial Marking

The initial marking of the system net consists of two human object nets, one on the place *human.outside.eos* and the other one on the place *human.outofreach.eos*, and one pet object net on the place *pet.outside.eos*. One of the human object nets has one token on the place *h*.3.*eos*, the other has no token and the pet object net has one token on its place.

$$\begin{aligned}\mu_0 = \quad & (human.outofreach.eos, h.3.eos) + (human.outside.eos, 0) \\ & + (pet.outside.eos, p.1.eos)\end{aligned}$$

With this initial marking, the object autonomous transition *h.stf.eos* is enabled. Also the synchronous transition *out.eos* is enabled, even though this does not seem apparent. By using the enabling predicate we can confirm this enabling:

$$\begin{aligned}\Phi \quad & ((out.eos, (h.out.eos, p.out.eos)), (human.outofreach.eos, h.3.eos) \\ & + (human.outside.eos, 0) + (pet.outside.eos, p.1.eos), \\ & (human.outofreach.eos, 0) + (human.outside.eos, h.1.eos) \\ & + (pet.outside.eos, p.1.eos)) \\ \Leftrightarrow \quad & \prod^1((human.outofreach.eos, h.3.eos) + (human.outside.eos, 0) + \\ & (pet.outside.eos, p.1.eos)) \\ & = human.outofreach.eos + human.outside.eos + pet.outside.eos \\ & \geq human.outside.eos + pet.outside.eos = \mathbf{pre}\ (out.eos) \\ \wedge \quad & \prod^1((human.outofreach.eos, 0) + (human.outside.eos, h.1.eos) \\ & + (pet.outside.eos, p.1.eos)) \\ & = human.outofreach.eos + human.outside.eos + pet.outside.eos \\ & \geq human.outside.eos + pet.outside.eos = \mathbf{post}\ (out.eos) \\ \wedge \quad & \prod^2_{N_H}((human.outofreach.eos, h.3.eos) + (human.outside.eos, 0) \\ & + (pet.outside.eos, p.1.eos)) \\ & = h.3.eos \\ & \geq \mathbf{pre}_{N_H}(h.out.eos) \\ \wedge \quad & \prod^2_{N_H}((human.outofreach.eos, 0) + (human.outside.eos, h.1.eos) \\ & + (pet.outside.eos, p.1.eos)) \\ & = h.1.eos = h.3.eos - h.3.eos + h.1.eos \\ & = \prod^2_{N_H}((human.outofreach.eos, h.3.eos) + (human.outside.eos, 0) \\ & + (pet.outside.eos, p.1.eos)) - \mathbf{pre}_{N_H}(h.out.eos) + \mathbf{post}_{N_H}(h.out.eos) \\ \wedge \quad & \prod^2_{N_P}((human.outofreach.eos, h.3.eos) + (human.outside.eos, 0) \\ & + (pet.outside.eos, p.1.eos)) \\ & = p.1.eos \geq \mathbf{pre}_{N_P}(p.out.eos) \\ \wedge \quad & \prod^2_{N_P}((human.outofreach.eos, 0) + (human.outside.eos, h.1.eos) \\ & + (pet.outside.eos, p.1.eos)) \\ & = p.1.eos = p.1.eos - p.1.eos + p.1.eos \\ & = \prod^2_{N_P}((human.outofreach.eos, h.3.eos) + (human.outside.eos, 0) \\ & + (pet.outside.eos, p.1.eos)) - \mathbf{pre}_{N_P}(p.out.eos) + \mathbf{post}_{N_P}(p.out.eos)\end{aligned}$$

Since the enabling predicate is verified, the event, which fires the according transitions in the system net and both object nets, is enabled.

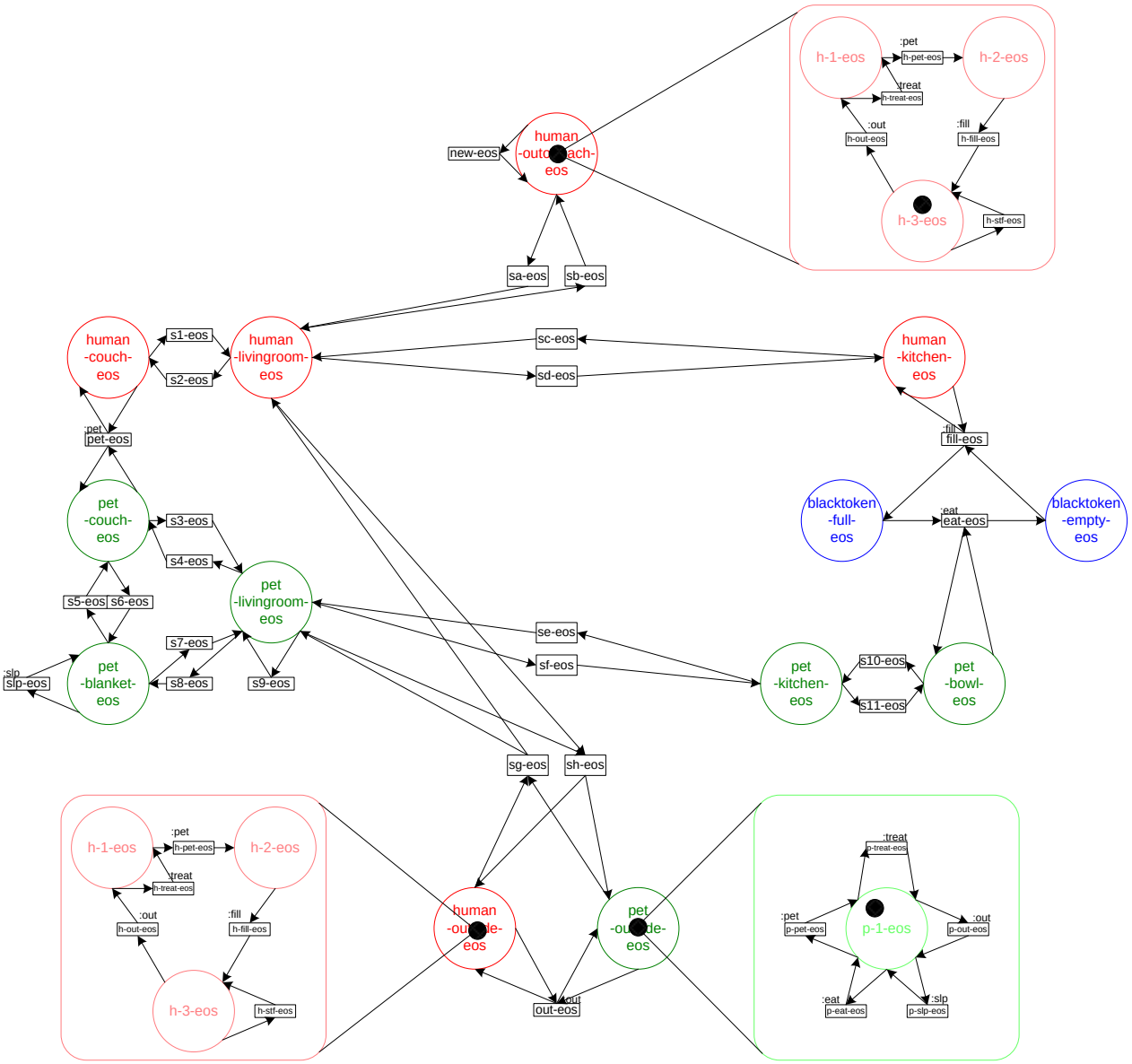

Figure 3.2: *EOS system net $\hat{N}$ with initial marking μ_0*

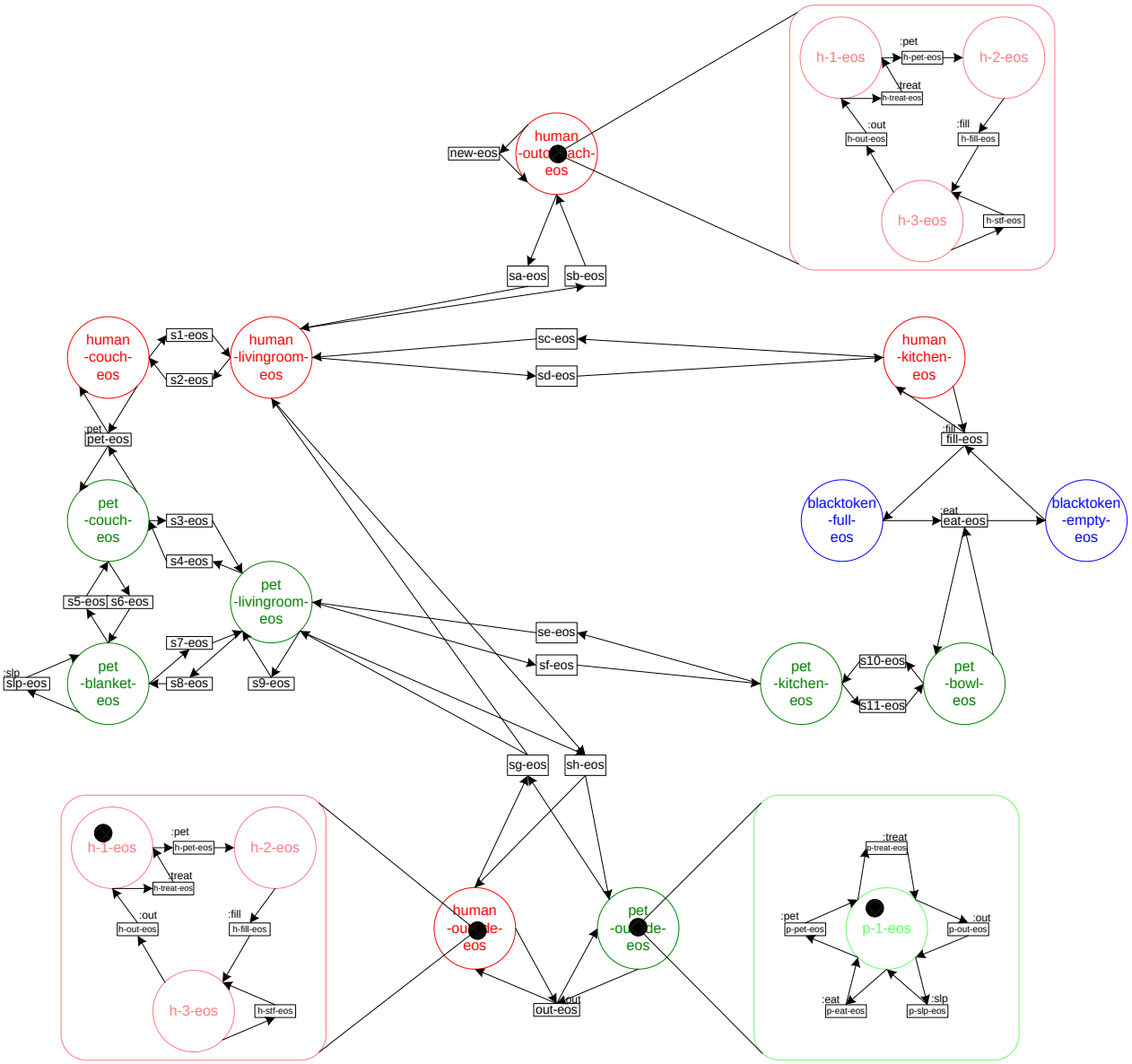

Figure 3.3: *EOS system net $\hat{N}$ with marking after firing the event $(out.eos, (h.out.eos, p.out.eos))$*

Chapter 4

Mobile EOS

After citing the definitions for mobile EOS given by Köhler in [KF07], we can assemble the mobile EOS which implements name spaces based on the EOS constructed in the previous chapter. For our example we chose name spaces to represent rooms in the appartment. The example modeled with mobile EOS changes the net structure. Name spaces become places for themselves, while EOS object nets are integrated into the EOS system net, forming new object nets. These very object nets cannot move; only their markings are transferred between name space places. The former EOS object nets' places are found in each of those new object nets, even though some of them are not connected to the former EOS system net's places via transitions. In our example it is not possible for the pet to feed from the bowl while being in the living room. This expresses the unavailability of services in certain name spaces. The connection only exists, if the EOS object net's transition is synchronized with the EOS system net's transition. For example for the pet, sleeping is only accessible on the blanket in the living room. Transitions which are object autonomous can be fired at any time, thus available at any location. In mobile EOS, the initial marking has to be adjusted as well. Each of the new system net's places (consisting of name spaces) features one object net token which cannot be removed. The object net present at one name space includes all the former EOS system net's places within one name space as displayed in 4.1.

Whenever a system net transition is fired, only the object net's marking is transferred, the system net remains the same. Places like $human.outside.eos, pet.livingroom.eos$ or $human.outofreach.eos$, which were able to hold object net tokens, now hold black tokens symbolizing the whereabouts of the EOS object nets.

4.1 Definition

4.1.1 Locality Infrastructure

A tuple $LS = (\mathscr{L}, N_m)$ is called locality infrastrucure, where

1. $\mathscr{L}$ is a finite set of disjoint nets, called locations, where $L = (P_L, T_L, \mathbf{pre}_L, \mathbf{post}_L)$.

2. $N_m = (P_m, T_m, \mathbf{pre}_m, \mathbf{post}_m)$ is the mobility infrastructure disjoint from all $L \in \mathscr{L}$ where $P_m = \emptyset$ and for all $t_m \in T_m$ exist two location nets $L, L' \in \mathscr{L}$ with $L \neq L'$ that are connected by t_m:

   $$\{p \in P | \mathbf{pre}_m(t_m)(p) > 0\} \subseteq P_L \wedge \{p \in P | \mathbf{post}_m(t_m)(p) > 0\} \subseteq P_{L'}$$

   The net $N(LS) := (P, T, \mathbf{pre}, \mathbf{post})$ generated by an infrastructure LS is given by $P = \bigcup_{L \in \mathscr{L}} P_L$, $T = T_m \cup \bigcup_{L \in \mathscr{L}} T_L$, $\mathbf{pre} = \mathbf{pre}_m \cup \bigcup_{L \in \mathscr{L}} \mathbf{pre}_L$ and $\mathbf{post} = \mathbf{post}_m \cup \bigcup_{L \in \mathscr{L}} \mathbf{post}_L$.

A mobility system is the pair (OS, LS) where $OS = (\hat{N}, \mathscr{N}, d, \theta, \mu_0)$ is an EOS and $LS = (\mathscr{L}, N_m)$ is a locality infrastrucure generating the system net: $N(LS) = \hat{N}$.

4.1.2 Conversion Equivalence

A mobile EOS is a pair $(OS, \leftrightarrow)$ where OS is an EOS and $\leftrightarrow$ is a conversion equivalence on the object nets' places $P = \bigcup_{N \in \mathscr{N}} P_N$, which is called conversion. Its equivalence classes are conversion sets. $P/_{\leftrightarrow}$ denotes the equivalence classes of $\leftrightarrow$ and $g_{\leftrightarrow} : P \to P/_{\leftrightarrow}$ is the natural surjection of $P/_{\leftrightarrow}$, i.e. the function that maps each place $p \in P$ to the conversion set that p belongs to:

$g_{\leftrightarrow}(p) = \{p' | p' \leftrightarrow p\}$.

The conversion $\leftrightarrow$ is extended to an equivalence on nested multisets. The equivalence identifies $\alpha, \beta \in \mathscr{M}_{\mathscr{N}}$ whenever the sum of tokens in all net tokens of type N (which is $\prod_N^2(\alpha)$ and $\prod_N^2(\beta)$) is equal in both nested multisets modulo $\leftrightarrow$:

$\alpha \leftrightarrow \beta \Leftrightarrow \prod^1(\alpha) \wedge \prod^1(\beta) \wedge \forall N \in \mathscr{N} : g_{\leftrightarrow}^{\#}(\prod_N^2(\alpha)) = g_{\leftrightarrow}^{\#}(\prod_N^2(\beta))$

The relation $\alpha \leftrightarrow \beta$ abstracts from the concrete location $\hat{p}$ in the system net in which an object net place p is marked, and additionally allows token conversions via $\leftrightarrow$.

4.1.3 Enabling of a Transition

Let $(OS, \leftrightarrow)$ be a mobile EOS and let $\mu, \mu' \in \mathscr{M}_{\mathscr{N}}$ be markings. The transition $(\hat{\tau}, C) \in \mathscr{T}$ is $\leftrightarrow$-enabled in mode $(\lambda, \rho) \in \mathscr{M}_{\mathscr{N}}{}^2$ if the following holds:

$\lambda \leq \mu \wedge \exists \lambda', \rho' : (\lambda' \leftrightarrow \lambda) \wedge (\rho' \leftrightarrow \rho) \wedge \phi((\hat{\tau}, C), \lambda', \rho')$

The successor marking is defined as $\mu' := \mu - \lambda + \rho$

4.1.4 Mobile EOS

If (OS, LS) is a mobility system where $OS = (\hat{N}, \mathscr{N}, d, \theta, \mu_0)$ is an EOS and $LS = (\mathscr{L}, N_m)$ is a locality infrastructure, a mobile EOS is defined as:

$$Mob(OS) := ((\hat{N}', \mathscr{N}', d', \theta', \mu_0'), \leftrightarrow)$$

where:

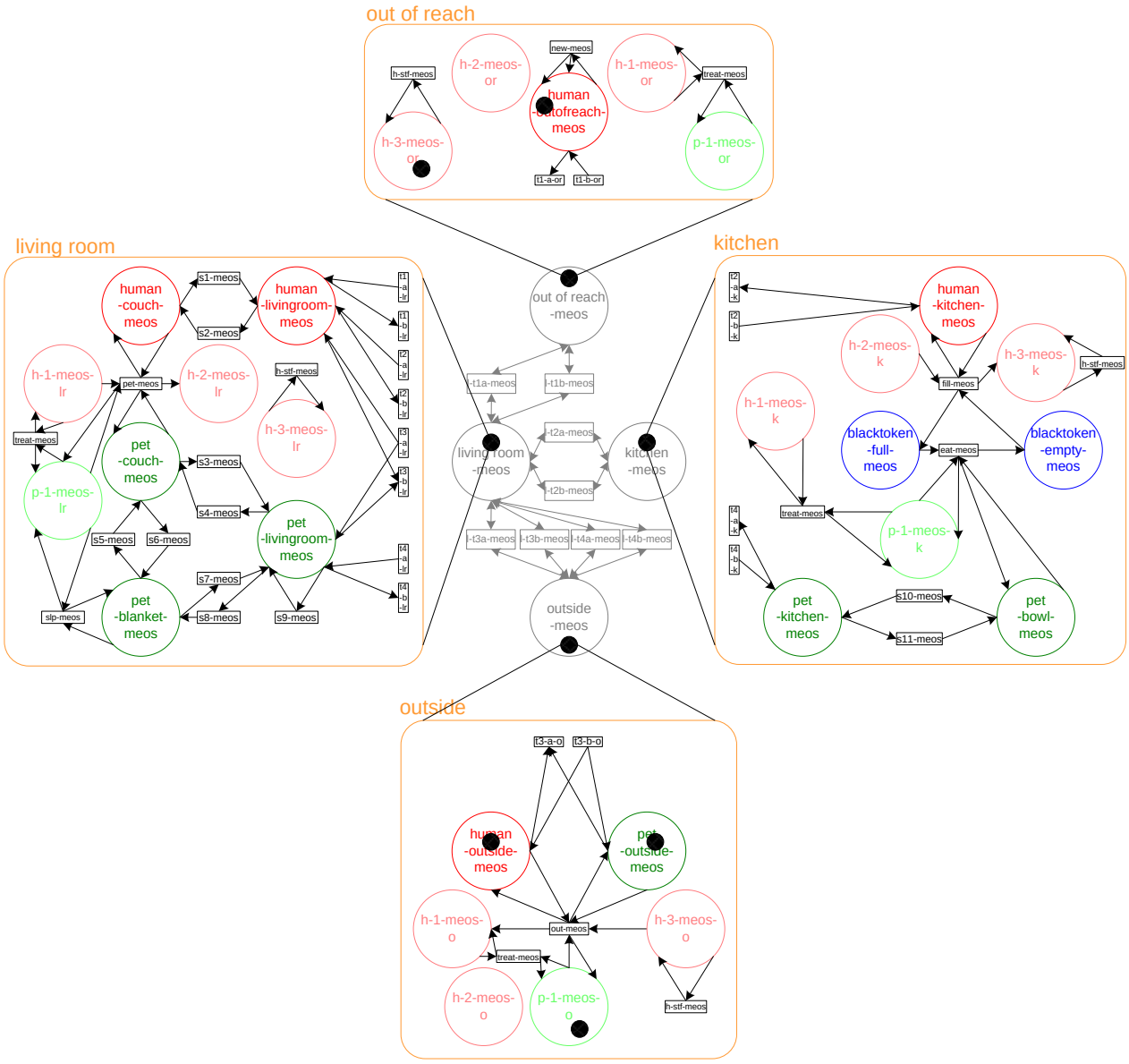

Figure 4.1: *Mobile EOS example net with initial marking* μ_0

1. The system net is $\hat{N}' = (\hat{P}', \hat{T}', \hat{\mathbf{pre}}\,' \hat{\mathbf{post}}\,')$ where $\hat{P}' = \{p_L | L \in \mathscr{L}\}$, $\hat{T}' = T_m$, and $\mathbf{pre}\ (t_m)(p_L) = \mathbf{post}\ (t_m)(p_L)$
$$= \begin{cases} 1 & if\ \bullet t_m \subseteq P_L \vee t_m \bullet \subseteq P_L \\ 0 & else \end{cases}$$
2. The set of object nets is $\mathscr{N}' = \{N_{\mathscr{N},L} | L \in \mathscr{L}\}$ where $N_{\mathscr{N},L} = (P, T, \mathbf{pre}\ , \mathbf{post}\ )$ is defined for each location $L = (P_L, T_L, \mathbf{pre}\ _L, \mathbf{post}\ _L) \in \mathscr{L}$ by:
$$P := P_L \cup \bigcup_{N \in \mathscr{N}} (P_N \times L)$$
$$T := \{(\hat{\tau}, C, L) | (\hat{\tau}, C) \in \theta \wedge \hat{\tau} \in T_L\}$$
$$\cup \{(t_m, L) | t_m \in T_m \wedge (t_m \bullet \subseteq P_L \vee \bullet t_m \subseteq P_L)\}$$
$$\mathbf{pre}\ (\hat{\tau}, C, L) = \mathbf{pre}\ (\hat{\tau}) + \sum_{\tau \in C} \mathbf{pre}\ (\tau)$$
$$\mathbf{pre}\ (t_m, L), (p) = \begin{cases} \mathbf{pre}\ _L(t_m)(p) & if\ p \in P_L \\ 0 & else \end{cases}$$
Analogously for **post** .
3. Each system net place p_L carries tokens of type $N_{\mathscr{N},L}$, i.e. $d(p_L) = N_{\mathscr{N},L}$.
4. $\theta = \{(t_m, C) | t_m \in T_m \wedge C = \{(t_m, L) | (t_m \bullet \subseteq P_L \vee \bullet t_m \subseteq P_L)\}\}$
$$\cup \{\varepsilon_{N_{\mathscr{N},L}} | N_{\mathscr{N},L} \in \mathscr{N}' \wedge \neg(t_m \bullet \subseteq P_L \vee \bullet t_m \subseteq P_L)\}$$
5. The initial marking is $\mu_0' = (\sum_{L \in \mathscr{L}} (p_L), \sum_{i=1}^{|\mu_L|} p_{L,i} + (M_{L,i} \times \{L\}))$, where $\mu_0 = \sum_{L \in \mathscr{L}} \mu_L$ and $\mu_L = \sum_{i=1}^{|\mu_L|} (p_{L,i}, M_{L,i})$, such that $p_{L,i} \in P_L$ for all L and $1 \leq i \leq |\mu_L|$.
6. The conversion $\leftrightarrow$ is defined by the family of conversion sets $\mathscr{E} = \{E_{N,p} | N \in \mathscr{N}, p \in P_N\}$ with $E_{N,p} := \{(p, L) | L \in \mathscr{L}\}$.

4.2 Example

1. Figure 4.2 shows the system net used in our example. Places represent name spaces and transitions allow object nets on the name space places to exchange tokens while object nets do not move.

2. The object nets for each system net place are shown in figures 4.3, 4.4, 4.5, and 4.6. These object nets can only reside on their according name space place and not move between system net places.

   The set of places of each object net consists of all EOS system net's places (their names beginning with either "human" or "pet") present at the particular name space which the object net is supposed to emulate, combined with all EOS object nets' places (beginning with "h" or "p" with extended names to include name space abbreviation).

   The set of transitions of each object net consists of all system autonomous transitions (beginning with the letter "s") present at the particular name space combined with system synchronized transitions (named pet-meos, out-meos, fill-meos and eat-meos) which gain new connections to the according object net places, and all object autonomous and object synchronized transitions (h-stf-meos and treat-meos). Additionally transitions (beginning with the letter "t"

in our example) which allow black tokens (which represent EOS object nets' whereabouts) to migrate between name spaces are included. These migrating transitions either only have a place in the pre- or post condition but not both and are synchronized with a system net's transition between the according name spaces and another migrating transition in another name space.

3. Each of the object nets mentioned above is placed on the according system net place.

4. The synchronisation structure includes all system net transitions coupled with the according migrating transitions from the object nets. This means if a migrating transition fires in one object net, the according migrating transition in another object net on a different name space fires, aswell as the according system net transition.

   The events containing migrating transitions are the only events involving the system net. All other events use the idle transition for the system net.

   All (EOS) system autonomous transitions are included in events in the synchronisation structure but now work like object autonomous transitions because the system net structure is contained in the object nets.

   All (EOS) system synchronized transitions are included as events but could also be considered object autonomous transitions. Since all EOS object nets are included in the object nets with the EOS system net's structure, the according transitions in EOS object and EOS system nets are combined into one.

   EOS object autonomous transitions are included without any changes and object synchronized transitions are included by merging the according object nets' transitions into one. The according events are adapted accordingly and are both considered object autonomous

5. The initial marking consits of one object net on each name space place (it has to be the according object net which includes the EOS system net's structure present at the particular name space).

   On the EOS system net's places (inside of the mobile EOS object nets) marking is changed to black tokens instead of object net tokens. EOS object net tokens are included only on EOS object nets' places in the name space's object net where the EOS object net was present in the underlying EOS.

6. The conversion sets each consist of all copies of one particular EOS object net place present at the different name spaces. With this conversion, migrating transitions can move object nets' tokens accordingly.

With the initial marking from the EOS example, the event firing the transition $out.meos$ is not enabled, since there is no token on place $h3.meos.o$. To enable this event, we need to fire $t1a.meos$, $t3b.meos$ and then $h3b.meos$ to let both humans meet in the living room exchange tokens and one of them to take the pet outside again. After those transitions, the event is enabled.

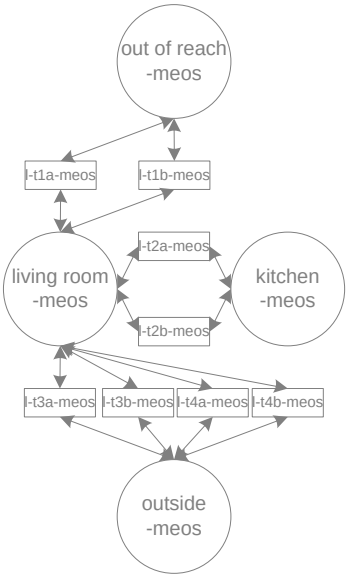

Figure 4.2: *Mobile EOS system net*

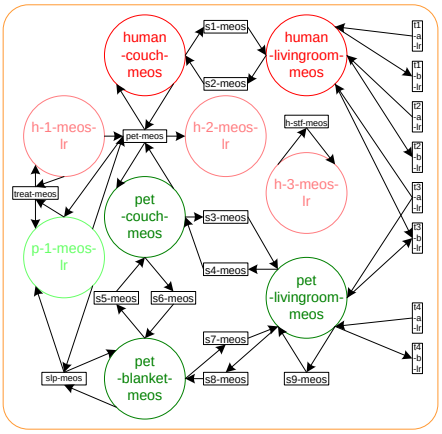

Figure 4.3: *Mobile EOS object net for the place livingroom.meos*

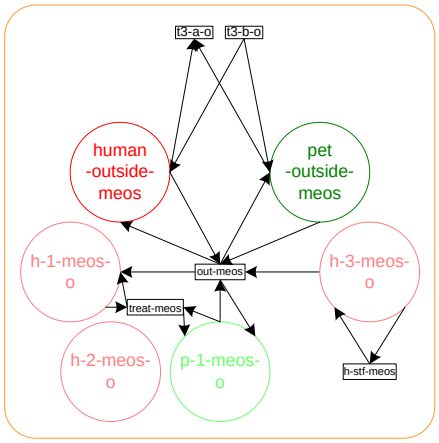

Figure 4.4: *Mobile EOS object net for the place outside.meos*

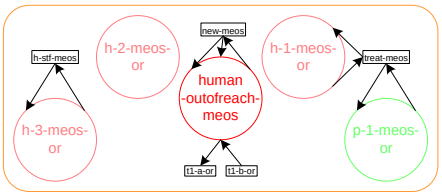

Figure 4.5: *Mobile EOS object net for the place outofreach.meos*

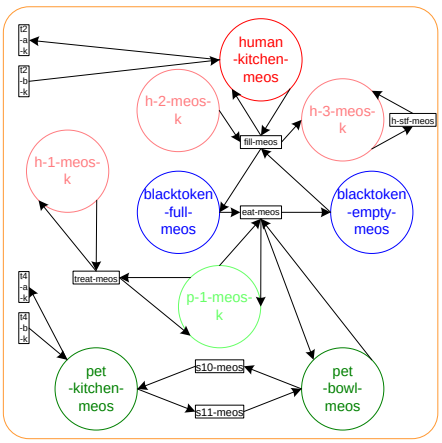

Figure 4.6: *Mobile EOS object net for the place kitchen.meos*

Chapter 5

Algebraic Higher Order Nets

In this chapter we cite the definitions of algebraic higher order (AHO) nets given in [HEM05], based on which we construct our AHO net example in the next chapter.

5.1 Definition

5.1.1 Algebraic High-Level Net

An alebraic high-level (AHL) net

$AN = (SP, P_{AN}, T_{AN}, pre_{AN}, post_{AN}, cond_{AN}, type_{AN}, A)$ consists of

1. an algebraic signature $\Sigma = (S, OP; X)$ with variables X;
2. a set of places P_{AN} and a set of transitions T_{AN};
3. pre- and post conditions $pre_{AN}, post_{AN} : T_{AN} \to (T_\Sigma(X) \otimes P)^\oplus$;
4. firing conditions $cond_{AN} : T_{AN} \to \mathscr{P}_{fin}(Eqns(\Sigma; X))$;
5. a type of places $type_{AN} : P_{AN} \to S$ and
6. a Σ-algebra A

where the signature $\Sigma = (S, OP)$ consists of sorts S and operation symbols OP, $T_\Sigma(X)$ is the set of terms with variables over X, $(T_\Sigma(X) \otimes P_{AN}) = \{(term, p) | term \in T_\Sigma(X)_{type_{AN}(p)}, p \in P\}$ and $Eqns(\Sigma; X)$ are all equations over the signature Σ with variables X.

($\mathscr{P}_{fin}(Y)$ returns the set of all finite subsets of Y.)

5.1.2 Firing Behavior of AHL Nets

A marking of an AHL net is given by $M_{AN} \in CP^\oplus$ where $CP = (A \otimes P_{AN}) = \{(a, p) | a \in A_{type_{AN}(p)}, p \in P_{AN}\}$.

The set of variables $Var(t) \subseteq X$ of a transition $t \in T_{AN}$ are the variables of the net inscriptions in $pre_{AN}(t)$, $post_{AN}(t)$ and $cond_{AN}(t)$. Let

$v : Var(t) \to A$

be a variable assignment with term evaluation $v^{\#} : T_\Sigma(Var(t)) \to A$, then (t,v) is a consistent transition assignment if and only if $cond_{AN}(t)$ is validated in A under v. The set CT of consistent transition assignments is defined by $CT = \{(t,v) | (t,v)\ \textit{consistent transition assignment}\}$.

A transition $t \in T_{AN}$ is enabled in M_{AN} under v if and only if $(t,v) \in CT$ and

$pre_A(t,v) \leq M_{AN}$, where $pre_A : CT \to CP^{\oplus}$ defined by

$pre_A(t,v) = \hat{v}(pre(t)) \in (A \otimes P_{AN})^{\oplus}$ and $\hat{v} : (T_\Sigma(Var(t)) \otimes P_{AN})^{\oplus} \to (A \otimes P_{AN})^{\oplus}$ is the obvious extension of $v^{\#}$ to terms and places (similar $post_A : CT \to CP^{\oplus}$). Then the follower marking is computed by $M'_{AN} = M_{AN} \ominus pre_A(t,v) \oplus post_A(t,v)$.

In order to allow P/T nets and rules as token of an AHL net AN, a specific signature HLNR-system-SIG together with a suitable HLNR-system-SIG algebra A are provided, where HLNR system refers to high-level net and rule systems (HLNR standing for high-level net and rule).

5.1.3 HLNR-System-SIG Signature

The signature HLNR-system-SIG is given by

$$
\begin{array}{lll}
AHON_{SIG} = & & \\
\quad sorts: & Transitions & ,Places,Bool,System,Mor,Rules \\
\quad opns: & tt,ff & :\to Bool \\
 & enabled & : System \times Transitions \to Bool \\
 & fire & : System \times Transitions \to System \\
 & applicable & : Rules \times Mor \to Bool \\
 & transform & : Rules \times Mor \to System \\
 & coproduct & : System \times System \to System \\
 & isomorphic & : System \times System \to Bool \\
 & cod & : Mor \to System
\end{array}
$$

5.1.4 HLNR-System-SIG Algebra

Given vocabularies T_0 and P_0, the HLNR-system-SIG algebra A for P/T systems and rules as tokens is given by

$A_{Transitions} = T_0, A_{Places} = P_0, A_{Bool} = \{true, false\}$,

A_{System} is the set of all P/T systems over T_0 and P_0, i.e.

$A_{System} = \{PN | PN = (P,T,pre,post,M) P/T\ system, P \subseteq P_0, T \subseteq T_0\}$
$\quad \cup \{undef\}$

A_{Mor} is the set of all rules of P/T morphisms for A_{System}, i.e.

$A_{Mor} = \{f | f : PN \to PN' \text{ P/T morphism with } PN, PN' \in A_{System}\}$.

A_{Rules} is the set of all rules of P/T systems, i.e.

$A_{Rules} = \{r | r = (L \overset{i_1}{\leftarrow} I \overset{i_2}{\rightarrow} R)$ *rule of P/T systems with strict inclusions* $i_1, i_2\}$,

$tt_A = true$, $ff_A = false$,

$enabled_A : A_{System} \times T_0 \to \{true, false\}$for $PN = (P, T, pre, post, M)$ with

$$enabled_A(PN,t) = \begin{cases} true & if\ t \in T, pre(t) \leq M \\ false & else \end{cases}$$

$fire_A : A_{System} \times T_0 \to A_{System}$ for $PN = (P, T, pre, post, M)$ with

$$fire_A(PN,t) = \begin{cases} (P, T, pre, post, M \ominus pre(t) \oplus post(t)) \\ \qquad\qquad if\ enabled_A(PN,t) = tt \\ undef \qquad\quad else \end{cases}$$

$applicable_A : A_{Rules} \times A_{Mor} \to \{true, false\}$ with

$$applicable_A(r,m) = \begin{cases} true & if\ r\ is\ applicable\ at\ match\ m \\ false & else \end{cases}$$

$transform_A : A_{Rules} \times A_{Mor} \to A_{System}$ with

$$transform_A(r,m) = \begin{cases} H & if\ applicable_A(r,m) \\ undef & else \end{cases}$$

where for $L \overset{m}{\rightarrow} G$ and $applicale_A(r,m) = true$ we have a direct transormation $G \overset{r}{\Rightarrow} h$

$coproduct_A : A_{System} \times A_{System} \to A_{System}$ is the disjoint union (i.e. the two P/T systems are combined without interaction) with

$$coproduct_A(PN_1, PN_2) = \underline{if}\ (PN_1 = undef \vee PN_2 = undef)\ \underline{then}\ undef$$
$$\underline{else}((P_1 \uplus P_2), (T_1 \uplus T_2), pre_3, post_3, M_1 \oplus M_2)$$

where $pre_3, post_3 : (T_1 \uplus T_2) \to (P_1 \uplus P_2)^{\oplus}$ are defined by

$pre_3(t) = \underline{if}\ t \in T_1 \underline{then}\ pre_1(t) \underline{else}\ pre_2(t)$

$post_3(t) = \underline{if}\ t \in T_1 \underline{then}\ post_1(t) \underline{else}\ post_2(t)$

$isomorphic_A : A_{System} \times A_{System} \to \{true, false\}$ with

$$isomorphic_A(PN_1, PN_2) = \begin{cases} true & PN_1 \cong PN_2 \\ false & else \end{cases}$$

where $PN_1 \cong PN_2$ means that there is a strict P/T morphism $f = (f_P, f_T) : PN_1 \to PN_2$ so that f_P and f_T are bijective functions,

$cod_A : A_{Morph} \to A_{System}$ with $cod_A(f : PN_1 \to PN_2) = PN_2$.

5.1.5 High-Level Net and Rule System

Given the signature HLNR-system-SIG and the HLNR-system-SIG algebra A as above, a high-level net and rule system $HLNR = (AN, INIT)$

consists of an AHL net AN with $SP = (HLNR - system - SIG; X)$ where X are variables over HLNR-system-SIG, and initial marking $INIT$ of AN such that

1. all places $p \in P_{AN}$ are either
   system net places i.e. $p \in P_{Systems} = \{p \in P_{AN} | type_{AN}(p) = System\}$
   or
   rule places i.e. $p \in P_{Rules} = \{p \in P_{AN} | type_{AN}(p) = Rules\}$,
2. all rule places $p \in P_{Rules}$ are contextual, i.e. for all transitions $t \in T_{AN}$ connected with p there exists a variable $r \in X$ such that $pre_{AN}(t)_{|p} = post_{AN}(t)_{|p} = r$, i.e. in the net structure of AN the connection between p and t is given by a double arrow labeled with the variable r.

Chapter 6

AHO Nets Imitating Mobile EOS

In this section we use the definitions from the previous chapter, change them to suit our purpose, and simultaneously construct the net we use for comparison. The reasons why we build it as we do are, to some extent, explained in the comparison chapter.

The AHO net framework, as defined in the previous chapter, provides a framework which allows P/T object nets and net transforming rules as tokens in an AHL net. Since the mobile EOS framework does not allow object nets to be transformed, we can reduce the HLNR-system-SIG signature and algebra to exclude rules and rule based transformations.

6.1 mEOS-System-SIG Signature

The adapted signature HLNR-system-SIG to cover mobile EOS behavior is given by

$$
\begin{array}{lll}
AHON_{SIG} = & & \\
\quad sorts: & Transitions & ,Places, Bool, System, Markings, \\
 & & ObjSyncTrans, ObjAutTrans, SyncTrans \\
\quad opns: & tt, ff & :\rightarrow Bool \\
 & enabled & : System \times Transitions \rightarrow Bool \\
 & fire & : System \times Transitions \rightarrow System \\
 & typePet & :\rightarrow System \\
 & typeHuman & :\rightarrow System \\
 & hPet & :\rightarrow SyncTrans \\
 & hOut & :\rightarrow SyncTrans \\
 & hFill & :\rightarrow SyncTrans \\
 & pPet & :\rightarrow SyncTrans \\
 & pOut & :\rightarrow SyncTrans \\
 & pSlp & :\rightarrow SyncTrans \\
 & pEat & :\rightarrow SyncTrans \\
 & combM & : Markings \times Markings \rightarrow Markings \\
 & getM & : Net \rightarrow Markings
\end{array}
$$

6.2 mEOS-System-SIG Algebra

The set T_0 contains all object nets' transitions present in the mobile EOS example:

$$T_0 = \{p.slp.ahon, p.eat.ahon, p.pet.ahon, p.treat.ahon, p.out.ahon, h.pet.ahon, h.fill.ahon, h.stf.ahon, h.out.ahon, h.treat.ahon\}$$

The set P_0 contains all object nets' places present in the mobile EOS example:

$$P_0 = \{p.1.ahon, h.1.ahon, h.2.ahon, h.3.ahon\}$$

The algebra A for P/T systems as tokens, able to imitate the mobile EOS given in chapter 4 is given by:

$$A_{Transitions} = T_0, A_{Places} = P_0, A_{Bool} = \{true, false\},$$

A_{System} is the set of all P/T systems over T_0 and P_0 (including undefined):

$$A_{System} = \{PN | PN = (P, T, pre, post, M) P/T - system, P \subseteq P_0, T \subseteq T_0\} \cup \{undef\}$$

We need to include markings in our algebra to imitate migrating transitions used in mobile EOS:

$$A_{Markings} = \{M | M \in P_0^{\oplus}\}$$

To access object synchronous transitions exclusively at the appropriate system net transitions, we include these transitions by defining the sort $A_{ObjSyncTrans}$:

$$A_{ObjSyncTrans} = \{(h.treat.ahon, p.treat.ahon)\} \subseteq T_0 \times T_0$$

To access object synchronous transitions exclusively at the appropriate system net transitions, we include these transitions by defining the sort $A_{ObjAutTrans}$:

$$A_{ObjAutTrans} = \{h.stf\} \subseteq T_0$$

Synchronous transitions are used as inscriptions in pre_A and $post_A$ and therefore need to be included in the signature:

$$A_{SyncTrans} = \{h.pet.ahon, h.out.ahon, h.fill.ahon, p.pet.ahon, p.out.ahon, p.slp.ahon, p.eat.ahon\}$$
$$tt_A = true, ff_A = false,$$

The operation $enabled_A$ not only verifies that a transition is enabled in an object net, but verifies that the chosen transition belongs to the object net.

$enabled_A : A_{System} \times A_{Transitions} \to \{true, false\}$ for $PN = (P, T, pre, post, M)$ with

$$enabled_A(PN,t) = \begin{cases} true & if\ t \in T, pre(t) \leq M \\ false & else \end{cases}$$

$fire_A : A_{System} \times A_{Transitions} \to A_{System}$ for $PN = (P, T, pre, post, M)$ with

$$fire_A(PN,t) = \begin{cases} (P, T, pre, post, M \ominus pre(t) \oplus post(t)) & \\ \qquad if\ enabled_A(PN,t) = tt & \\ undef \qquad else & \end{cases}$$

We do not use the constants $typePet_A$, $typeHuman_A$, and $typeBlackToken_A$ anywhere in the system net, but we need these constants to describe our conditions for the initial marking:

$typePet_A =$
$(\{p.1.ahon\},$
$\{p.slp.ahon, p.eat.ahon, p.pet.ahon, p.treat.ahon, p.out.ahon\},$
$\{(p.slp.ahon, p.1.ahon), (p.eat.ahon, p.1.ahon),$
$(p.pet.ahon, p.1.ahon), (p.treat.ahon, p.1.ahon),$
$(p.out.ahon, p.1.ahon)\},$
$\{(p.slp.ahon, p.1.ahon), (p.eat.ahon, p.1.ahon),$
$(p.pet.ahon, p.1.ahon), (p.treat.ahon, p.1.ahon),$
$(p.out.ahon, p.1.ahon)\},$
$\{\})$
$typeHuman_A =$
$(\{h.1.ahon, h.2.ahon, h.3.ahon\},$

$\{h.pet.ahon, h.fill.ahon, h.stf.ahon, h.out.ahon, h.treat.ahon\},$
$\{(h.pet.ahon, h.1.ahon), (h.fill.ahon, h.2.ahon),$
$(h.stf.ahon, h.3.ahon), (h.out.ahon, h.3.ahon),$
$(h.treat.ahon, h.1.ahon)\},$
$\{(h.pet.ahon, h.2.ahon), (h.fill.ahon, h.3.ahon),$
$(h.stf.ahon, h.3.ahon), (h.out.ahon, h.1.ahon),$
$(h.treat.ahon, h.1.ahon)\},$
$\{\})$
$typeBlackToken_A = \{\}$

As mentioned earlier synchronous transitions are used as inscriptions in pre_A and $post_A$ and therefore need to be included in the signature. Since we intend to synchronize object nets' transitions with specific system net's transitions, we require the following constants:

$hPet_A = h.pet.ahon,$
$hOut_A = h.out.ahon,$
$hFill_A = h.fill.ahon,$
$pPet_A = p.pet.ahon,$
$pOut_A = p.Out.ahon,$
$pSlp_A = p.slp.ahon,$
$pEat_A = p.eat.ahon,$

In the $cond_A$ inscriptions in migrating transitions, we need to make sure that object nets may take markings with them to the target name space. At the same time we demand the total of all object nets' markings in both the source and target name space not to change. To include these conditions, we added the following operations working witch object nets' markings:

$getM_A : A_{System} \to A_{Marking}$
$getM_A(PN, M) = M$
$combM_A : A_{Markings} \times A_{Markings} \to A_{Markings}$
$combM_A((PN_1, M_1), (PN_2, M_2)) = M_1 \oplus M_2$

6.3 $AHON_{mEOS}$

Given the signature and algebra statet above, we can compile the AHO net example which implements our example mobile EOS behavior:

$$AHON_{mEOS} = ((mEOS - system - SIG; X), P, T, pre, post, cond, type, A, M)$$

with our adapted signature mEOS-system-SIG and our specified mEOS-system-SIG algebra A.

P consists of all the places from the EOS used in the mobile EOS example and one additional place for each name space.

T consists of all the transitions from the EOS used in the mobile EOS example and additional transitions which allow object autonomous and object synchronous transitions. As introduced in [Hof05], in AHO net we need a dedicated system net transition to let object nets fire quasi autonomously.

pre and *post* are almost identical to the *pre* and *post* functions used in the EOS used in the mobile EOS example but including the name space places (including tuples for object autonomous and object synchronized transitions) and terms for accessing and manipulating object nets. Name space places are needed in the *pre* and *post* condition of all transitions for which interaction with object nets is required.

cond euals the empty set for all system autonomous (transport) transitions. For all system synchronous transitions, it contains a term

$$enabled(system, syncTrans) = true$$

where *syncTrans* is one of the constants defined in the algebra *A* and *system* is a P/T net from one of the name space places. For object autonomous and object synchronized transitions, it contains a term

$$enabled(system, trans) = true$$

(for object synchronized transitions it contains two of those terms) where *system* is a P/T net from one of the name space places and *trans* is either an element from the set $ObjAutTrans$ or one part of a tuple from $ObjSyncTrans$.

type types all places for P/T nets. Other than in the mobile EOS framework, we do not distinguish between human and pet typed places but allow all kinds of net tokens on places in general. We restrict this by phrasing conditions for the initial marking.

We assume disjunct naming of transitions of all object nets. Therefore we avoid checking for affiliation of transitions and object nets, with other than the *enabled* condition. To imitate mobile EOS behavior correctly, we have to include conditions for the initial marking of the AHO net. We need to postulate the initial marking only places nets without markings on all places other than the name space places. Name space places require one object net of each kind (except for the black token kind) which holds all object nets' tokens present at that particular name space.

Every initial marking not implementing these conditions would not destroy the net's behavior but result in transitions not beeing enabled even though they were supposed to be.

Since AHO nets' system net places do not distinguish between object net's types, we have to provide distribution of object net tokens according to the given mobile EOS. This condition could be excluded, if every system net transition included a condition term checking the object net's type.

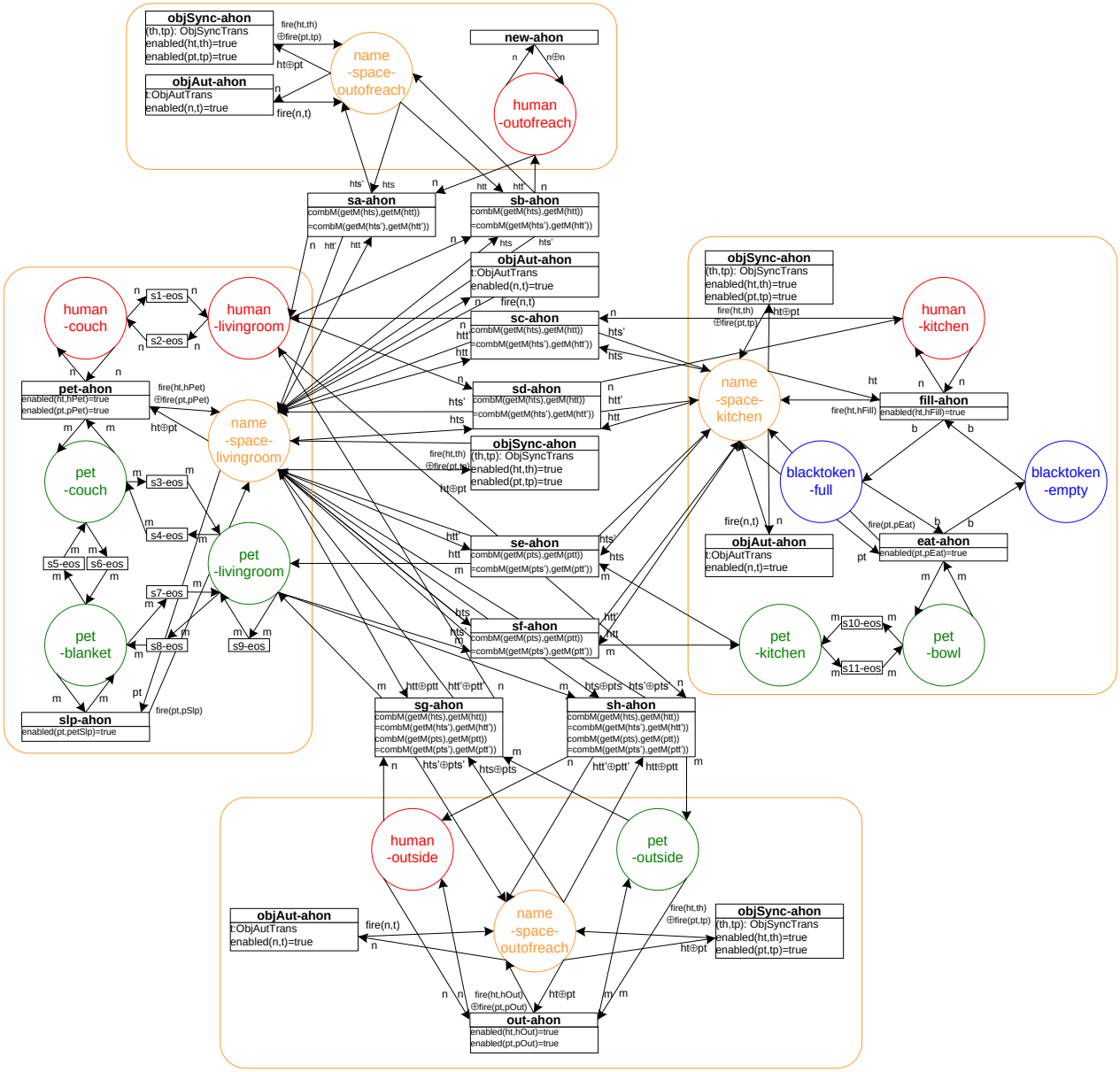

Figure 6.1: *$AHON_{mEOS}$ example net*

Chapter 7

Comparison

There is no algorithm for converting a given mobile EOS (including the underlying EOS and conversion equivalence) to the AHO net framework yet. The aim of this work is not to formalize the converting process but to construct one particular AHO net based on a given mobile EOS. The main features which the mobile EOS framework offers are the following:

1. It has a system net with various object nets which can interact with the system net and each other or fire autonomously (all of this happens in the name spaces' object nets)
2. EOS object nets may access each other's tokens while being in the same name space
3. EOS object nets may migrate between name spaces while taking some of their tokens (or all of them) with them.
4. It is possible to introduce new object nets without a marking.
5. Synchronized transitions (which are represented by an event consisting of a system net transition and at most one transition of each object net type) can be fired if a synchronized system net's transition as well as the according synchronized object nets' transitions are enabled in the same name space.

Our example constructed in chapter 4 covers all of those features. The aim of the construction of the AHO net example was to integrate all of those features into the AHO net framework. In the following sections we analyze the constructed AHO net example with focus on how the net sructures compare (7.1), how the main features of the mobile EOS are integrated (7.3) and we show exemplarily that both constructed nets perform the same behavior (7.2).

7.1 Structure

7.1.1 Name Spaces

Name spaces are used to combine some places of the system net where object nets are allowed to communicate with each other. Name spaces become obsolete, if object nets consist of one place only or do not have any synchronized transitions. If only one name space exists, it does not affect the behavior either. In those cases EOS and mobile EOS show the same firing behavior. In our example we have four name spaces and object nets do have synchronized transitions, so the name spaces impact the behavior. The mobile EOS object nets share their tokens inside of a name space. This behavior is realized by merging all object nets with the according system net places by creating an object net token on each name space place. Since set theoretical operations to collect tokens from different object nets are not given in the AHO net framework without adding the according transitions, we collect all object nets' tokens on one name space place to imitate this behavior. When migrating between name spaces in mobile EOS, object nets' tokens can be transferred between name spaces. So when a migrating transition fires, either none, some or all of the tokens belonging to the associated type are transferred (see scenario A 8). In the AHO net example, we added migrating transitions which allow exactly this behavior (see scenario B 8).

7.1.2 System net

The system net offers an environment in which object nets may move and interact with the system net. In our EOS example the system net represents the apartment in which object nets, i.e. people and pets, may move. The places represent areas in the apartment, while transitions offer a way to move between or interact with those places. The EOS framework does not allow different types of object nets on one place. For that reason we have to include a couch place for pets and a couch place for humans, even though these two places represent the same area in the apartment.

The mobile EOS represents the same environment as the EOS example, but changes the net structure to implement name spaces. In mobile EOS places in the system net symbolize name spaces which are more general areas, i.e. rooms. In mobile EOS we only have one type of system net transitions, the ones which allow migrating between name spaces. Object nets are bind to explicit places because they contain the EOS system net's structure available in that particular name space. Therefore the migrating transitions only exchange the EOS object nets' tokens (in a nondeterministic way), which is explained in 7.1.3.

In the AHO net example we use a system net structure similar to the one used in our EOS example. The signature does not distinguish between different types of object nets and we therefore could combine EOS places like *pet.couch.ahon* and *human.couch.ahon* into one place *couch.ahon*. We do not do this combination to keep the similarity to the EOS and mobile EOS structure, even though it would make sense for the meaning of our example. The ability for combining places is one major advantage of the AHO net framework, which allows nets to be modeled realisticly.

Because we adopt the EOS structure, we have to include conditions to position object nets on the designated places for the initial marking. Consistency thereof is guaranteed by the transitions of our example. Since we keep the EOS structure, we have different types of transitions: transport, system object synchronized, object autonomous, object synchronized and migration. We gave detailed descriptions of those transitions in 6.3.

7.1.3 Object nets

Object nets act as agents in the system net. They can move, synchronize with the system net, synchronize with each other or fire autonomously. In EOS object nets are P/T nets, which can do all the things mentioned above. Object nets' types stand for humans and pets respectively while one instance thereof represents one particular human or pet. Human object nets may only reside on human net typed places, pet object nets on pet typed places respectively. Object nets may interact with the system net (for instance $fill$ the bowl) or interact with each other (give $treat$ transition) everywhere in the system net, if the according transitions are enabled. Object nets can also interact with each other and the system net at the same time (take pet *out*side transition). Object nets' markings, in our example, could be interpreted as chores or tasks. If a human object net has a token on the place *h*3 the transition *out* is enabled and therefore taking the pet outside is the next task to be performed. For pets the tasks are not ordered and can all be performed from the initial state (we assume that a pet does not plan its day). In EOS two humans can exchange tokens for performing a task. For instance, if one human object net with a token on place h3, which enables the transition *out*, is on the place $human.outofreach.eos$ and another human object net without any tokens is on the place $h.ouside.eos$, the synchronized transition $out.eos$ is enabled if a pet with one token is on the place $pet.outside.eos$, as shown in 3.2.5.

This behavior is not wanted in mobile EOS since the two places with human object nets are not in the same name space. Therefore this transition with according marking is not enabled in mobile EOS. Object nets in mobile EOS contain the EOS system places available in one name space, all object nets' places and those object nets' transitions available in that particular name space. Therefore we get new object nets, for each name space one. With the initial marking (one human object net token on $human.outofreach.meos$ with a token on $h.3.meos$, one human object net token on $human.out.meos$ and one pet object net on $pet.out.meos$ with a token on $p1.meos$) in mobile EOS, the two human object nets and the pet object net which were object net tokens in EOS but are now black tokens on EOS system net's places in the object nets on each name space place, would have to meet in the living room name space. When both of the human object net tokens are present in the same name space, their markings on the EOS object nets' places present in the name space are combined and any human token leaving the name space could take all or some tokens with it. Therefore one human and pet token could migrate to the outside name space and take the tokens with them. This enables the synchronized transition $out - meos$. This event could be interpreted as one person asking the other to perform his task.

In the AHO net example we use the same object nets as in EOS. But the AHO net framework does not allow object nets to fire autonomously (without the system nets). Therefore even for object autonomous transitions we need a system net transition to fire it. Object synchronized transitions also need to be initiated by the system net. We

integrated those two types of transitions by including object autonomous and object synchronized transitions for each name space place. We included the condition that all object nets on places other than name space places are not supposed to hold any tokens (empty marking) and all object nets' tokens are collected in one object net for each type on the name space place. Therefore, whenever a transition synchronized with another object net is fired, the according name space object net token fires and is put back on the name space place.

7.1.4 Black Tokens

Black tokens are a way to implement P/T net behavior into the system net structure. Since they do not allow being transferred between system net places other than from places in the pre-domain to places in the post domain of a system net transition, they let the system net act as a P/T net.

In mobile EOS object nets from the underlying EOS are represented as black tokens on EOS system net places in object nets on name space places. The same holds for black tokens. Black tokens do not possess any internal structure and therefore do not bring any additional places to the object nets on the name space places, as EOS object nets do with their object net places.

There are different ways to integrate EOS black tokens into the AHO net framework. Instead of creating a new sort for black tokens, we included them by defining black tokens as empty object nets (a P/T net consisting of no places, transition pre and post condition functions or markings). Since in mobile EOS black token places are integrated into the infrastructure, there is no need to distinguish them from ordinary object net places. In our example the system net transitions assure that they stay on their designated places and cannot be transported to any places for regular object nets.

7.2 Behavior

In mobile EOS we have two different kinds of firing scenarios: Object nets' firing and exchange of tokens. While object net's firing includes firing of the EOS system net which is now integrated into the object net, this scenario covers four kinds of events: object autonomous, object synchronized, system synchronized, and system autonomous events. The only kind of transition not executed in the object nets is the migrating kind.

Migrating transition, in mobile EOS, exchange object nets' tokens between name spaces without removing the object nets from their name space places. When firing an object net's transition, which is synchronized within a system net's transition, all, some, or none of the tokens of EOS object nets' places are moved to the according places in the target name space object net.

For our adapted AHO net definition we did not replicate the merging of the system net and object nets which occurs by transforming an EOS to its according mobile EOS. Therefore the different events in mobile EOS do not structurally resemble the events of our AHO net framework. But since we have shown the integration of the four types

of events and migrating transitions in the previous section, we have shown exemplarily the behavioral similarity of both example nets.

7.3 Main Features

The main features of mobile EOS are achieved by our AHO net example as follows:

1. *It has a system net with various object nets which can interact with the system net and each other or fire autonomously (all of this happens in the name spaces' object nets)*:

   Even though we used the structure given by the underlying EOS, our AHO net example provides a system net and allows for various object nets of different types to be transported, fire synchronously with the system net, fire quasi autonomously or fire synchronized with an object net of another type.

   The two latter are made available through the system net transitions $objAut.ahon$ and $objSync.ahon$. Transporting transitions of the system net are characterized by empty conditions for object nets and system synchronous transitions are defined by provided constants in the algebra which are used in according system net transitions to fire the object nets' synchronized transitions respectively.

2. *EOS Object nets may access each other's tokens while being in the same name space*

   Since we included a condition which assures all object nets' tokens to be collected in a designated object net on each name space place, this feature is implemented. The system net transitions conserve this state by keeping those object nets' markings seperated from all other object nets' markings continually.

3. *EOS Object nets may migrate between name spaces while taking some of their tokens (or all of them) with them.*

   This feature is realized by including migrating transitions in the system net. These transitions take a human object net from a system net place and transport it to the system net place in another name space while exchanging the markings of the object net tokens on the name space places. For example the transition $sa.ahon$ takes a human net n from the place $human.outofreach$ and puts it on the place $human.livingroom$. At the same time the human object net hts (which should be the only one on that place) from $name.space.outofreach$ and htt from $name.space.livingroom$ are removed and their markings are redistributed using the condition $combM(getM(hts), getM(htt)) = combM(getM(hts'), getM(htt'))$. The updated tokens

   hts' and htt' are returned to the according name space places.

   With this condition we can imitate the nondeterministic behavior of mobile EOS' token distribution by chosing the according variable assignment to satisfy this condition. It also allows for object nets' tokens to be transported from the target to the source name space, but this behavior is not prevented in mobile EOS either. This could be excluded in the AHO net example by adding a condition for assuring that the target marking is larger than or equal to the origin marking.

4. *It is possible to introduce new object nets without a marking.*

   In our AHO net example it is possible to introduce new object nets with empty marking. The system net transitions *new.ahon* replicates the object net which enables *new.ahon* and therefore introduces a new object net into the system.

5. *Synchronized transitions (which are represented by an event consisting of a system net transition and at most one transition of each object net type) can be fired if a synchronized system net's transition as well as the according synchronized object nets' transitions are enabled in the same name space.*

   The firing of a synchronized transition is shown in a scenario in 8.

Chapter 8

Conclusion/Future Work

In chapter 2 through 4 we constructed a mobile EOS which implements all the main features given in chapter 7.

Based on this example, we started defining our AHO net in chapter 6 by extending the known signature and algebra to cover mobile EOS behavior. We chose an architecture similar to the underlying EOS to keep the structure lucid and included conditions (6.3) to assure the net behaves as intended.

In chapter 7 we analyzed both the mobile EOS and AHO net example with focus on their structure and their behavior. We showed exemplarily that our AHO net example achieves the same behavior as our mobile EOS example. Within the AHO net framework, we were capable to implement the different kinds of events provided by the mobile EOS framework (object autonomous, object synchronized, system autonomous, system synchronized, migration) and discovered some advantages of the AHO net framework:

1. Places which represent the same area (in reality) can be combined into one place while allowing different object net types on them. This retains lucidity for the observer and compresses the structure of the net.
2. The AHO net framework grants extensions including new sorts, operations and equations. Therefore new aspects could be implemented without changing the framework.
3. The original AHO net definition provides for rule tokens to be integrated. These rules are able to transform object nets, which allows for even more application fields to be explored

The AHO net framework offers a variety of extension and to be able to integrate mobile EOS behavior into the framework means that the offered extensions could be applied to the embedded mobile EOS aswell. We can conclude that for the chosen example the conversion to the AHO net framework was successful. For future work we recommend to formalize the conversion of a given mobile EOS to the AHO net framework. Based on the findings of this work it should be possible to describe this process with a formal

algorithm. There might even be more suitable approaches, for example by using a three layered architecture.

Appendix

A Appendix: Mobile EOS Scenario

This scenario shows how the transition *out*.*meos* can be fired starting from the inital marking. We need to fire $t1$*a*.*meos*, $t3$*a*.*meos* and $t3$*b*.*meos* to achieve the needed marking which enables *out*.*meos*.

B Appendix: AHO Net Scenario

This scenario shows how the transition *out*.*ahon* can be fired starting from the inital marking. We need to fire *sa*.*ahon*, *sg*.*ahon* and *sh*.*ahon* to achieve the needed marking which enables *out*.*ahon*.

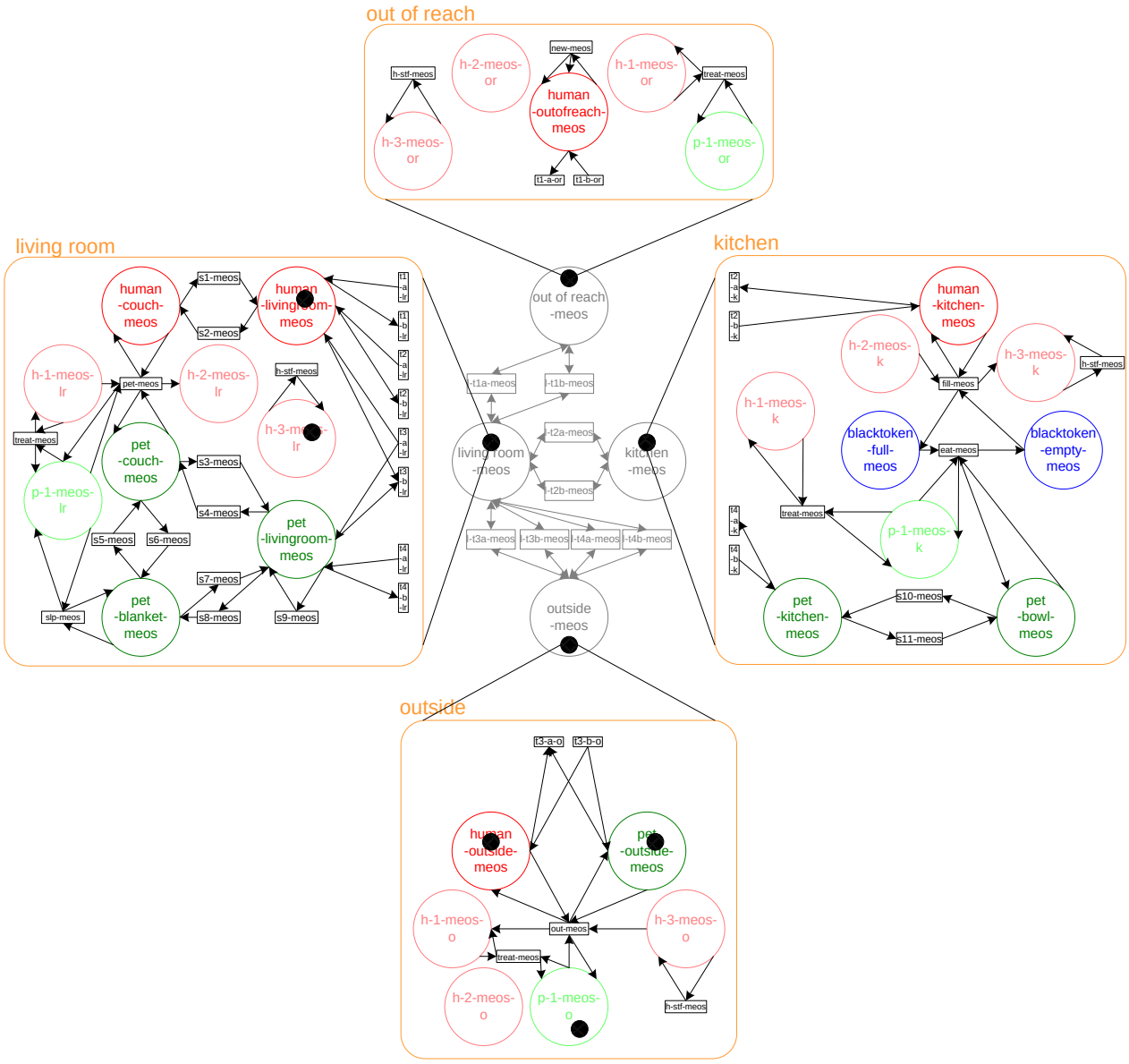

Figure 8.1: *Mobile EOS example after firing t1a.meos*

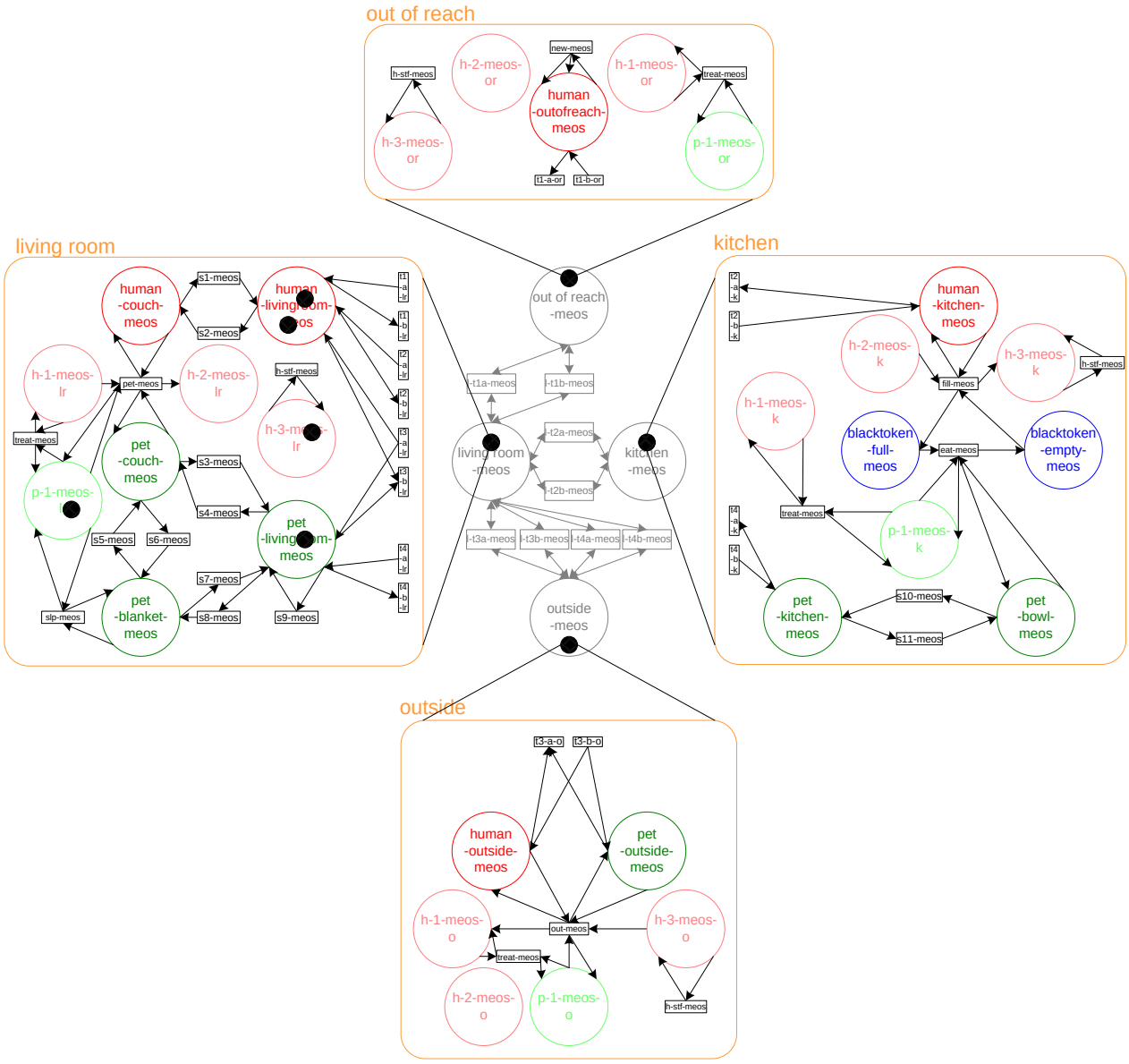

Figure 8.2: *Mobile EOS example after firing t3a.meos*

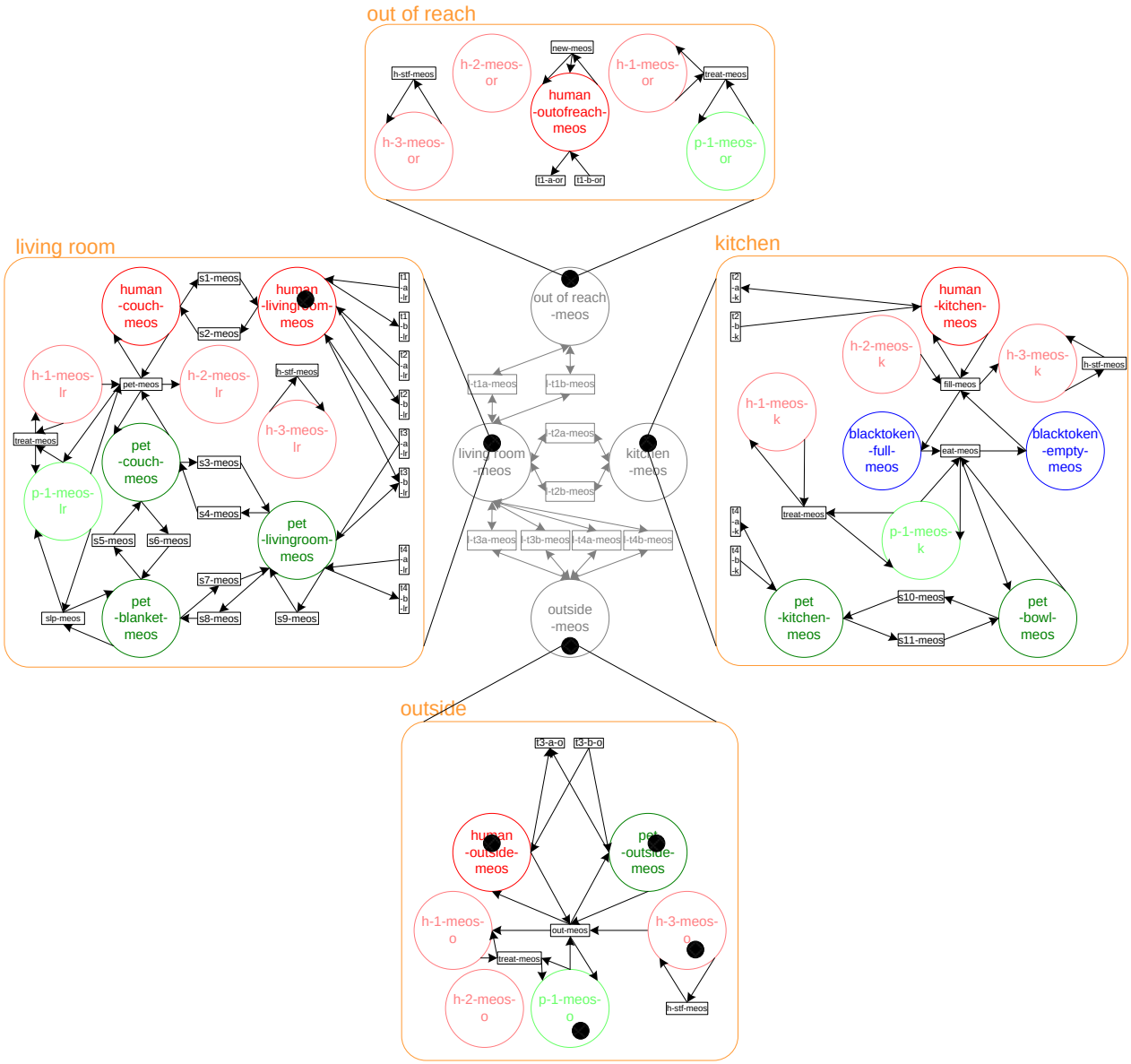

Figure 8.3: *Mobile EOS example after firing t3b.meos*

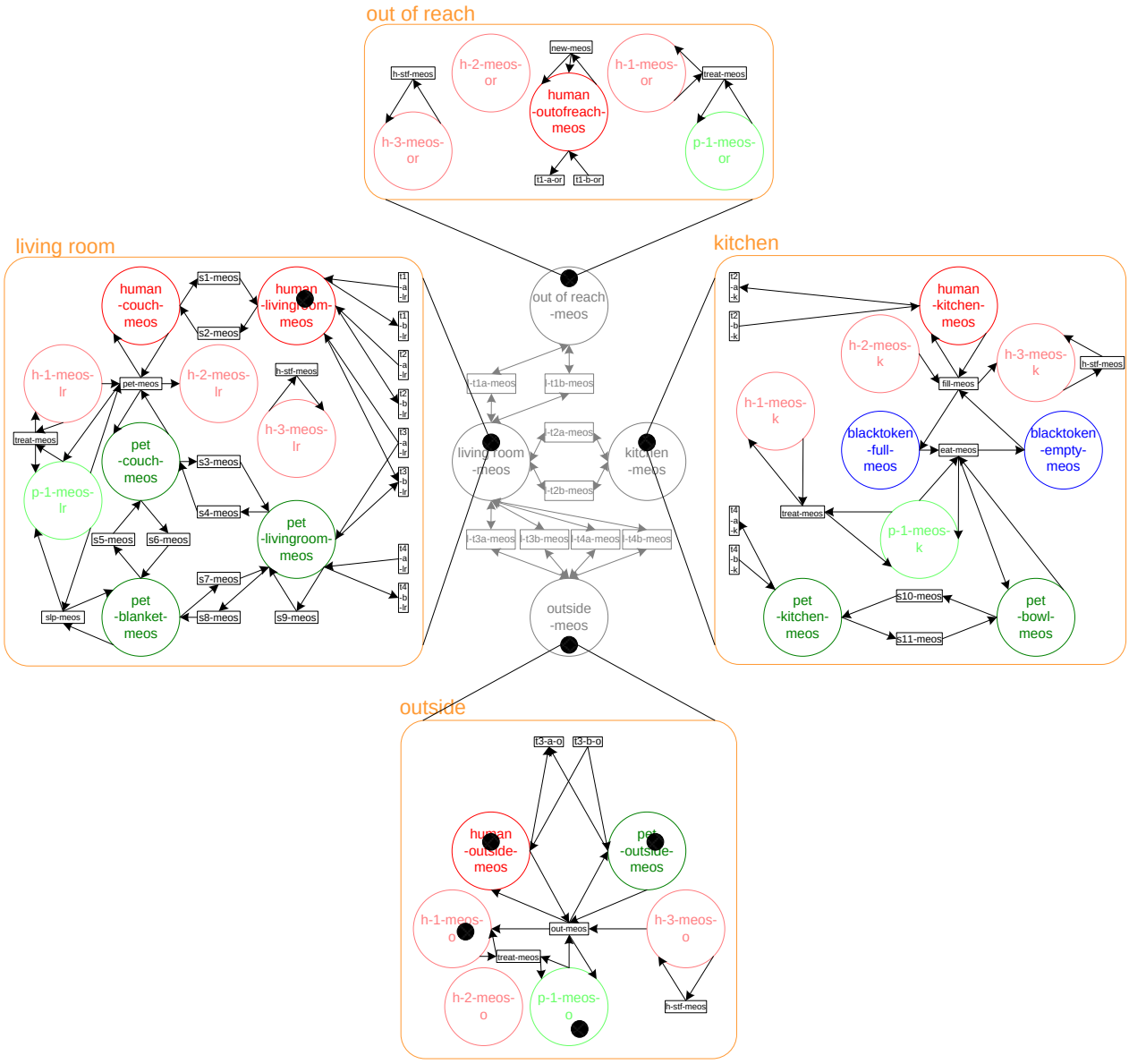

Figure 8.4: *Mobile EOS example after firing out.meos*

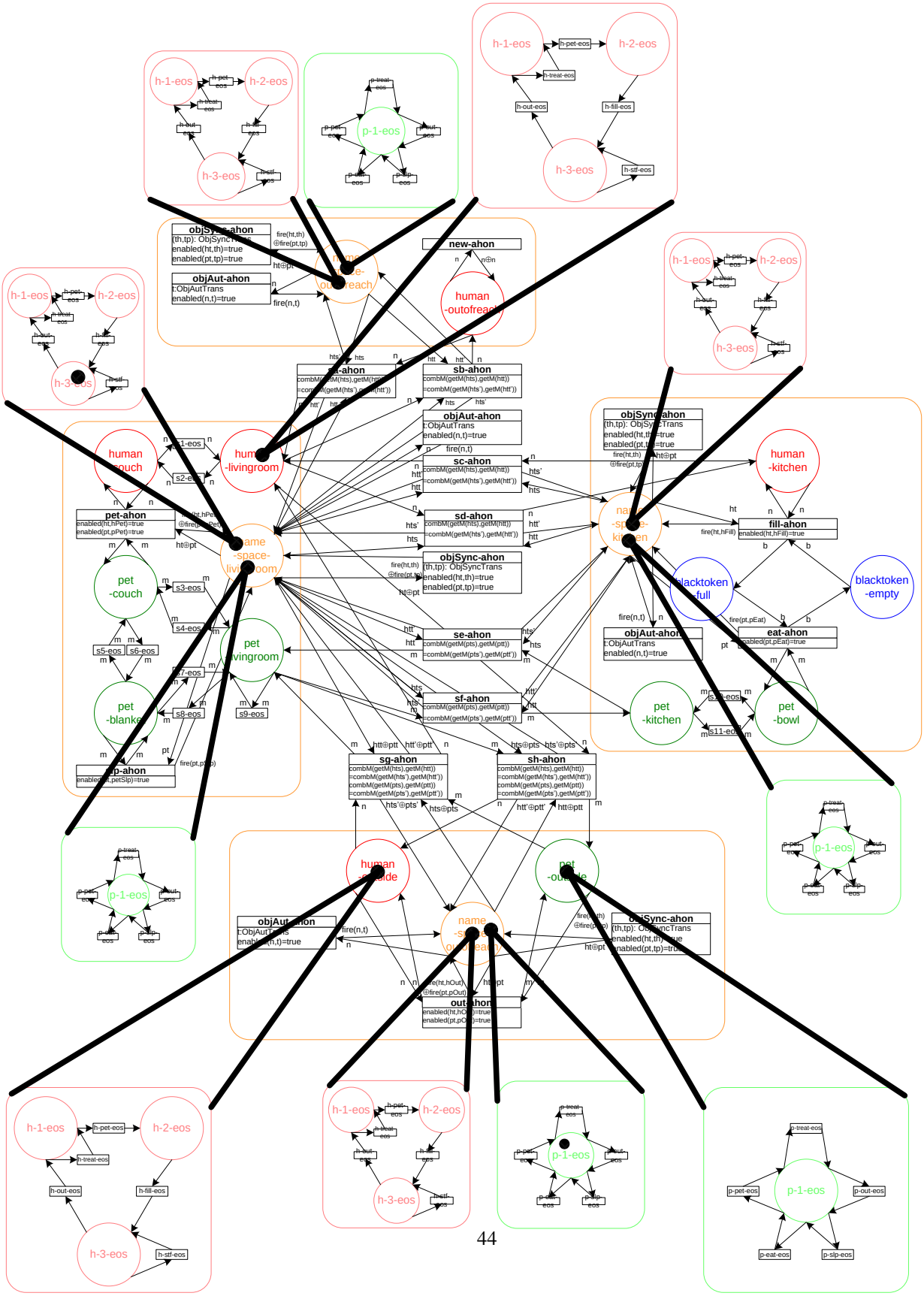

Figure 8.5: *AHO net example after firing sa.ahon*

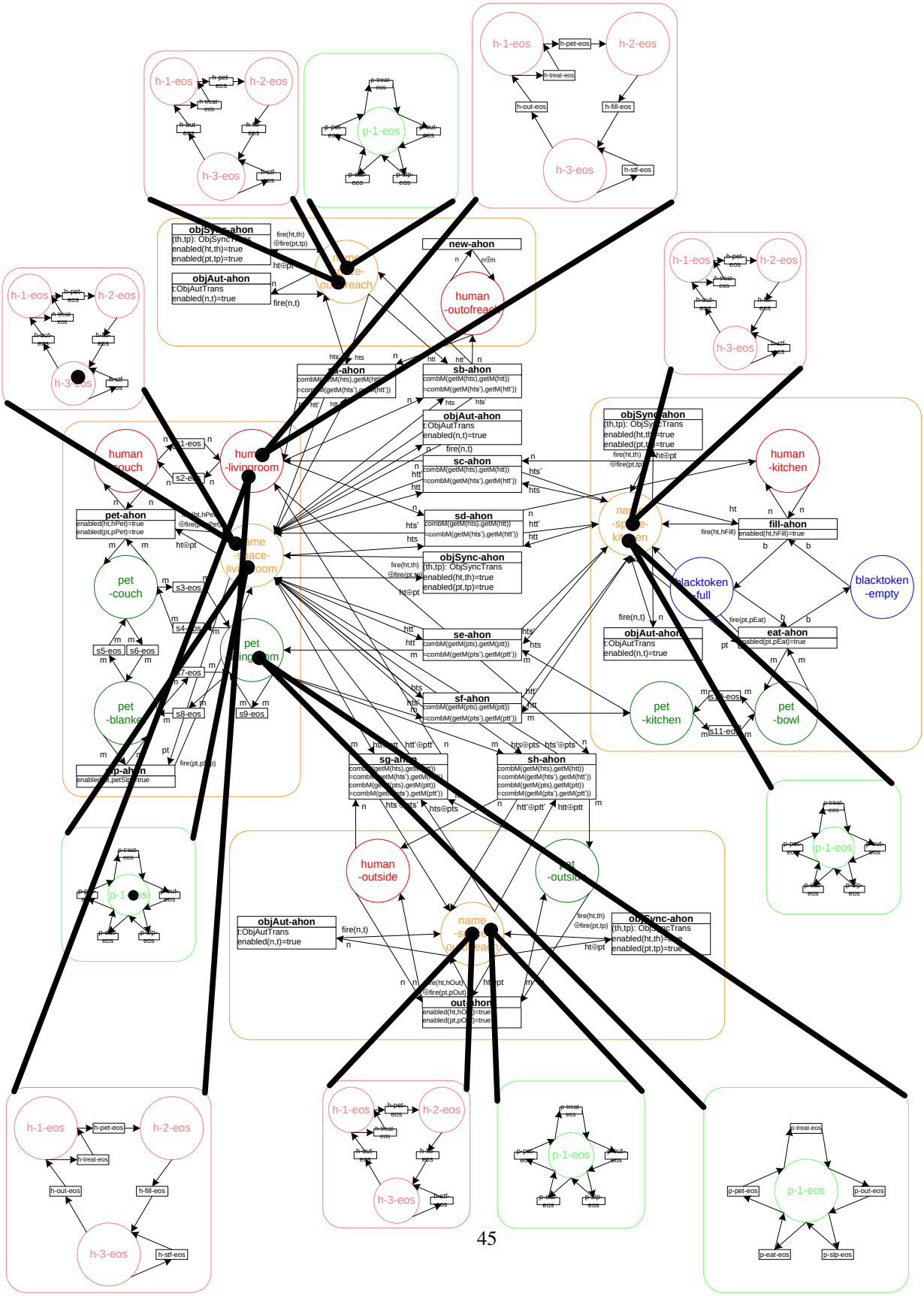

Figure 8.6: *AHO net example after firing sg.ahon*

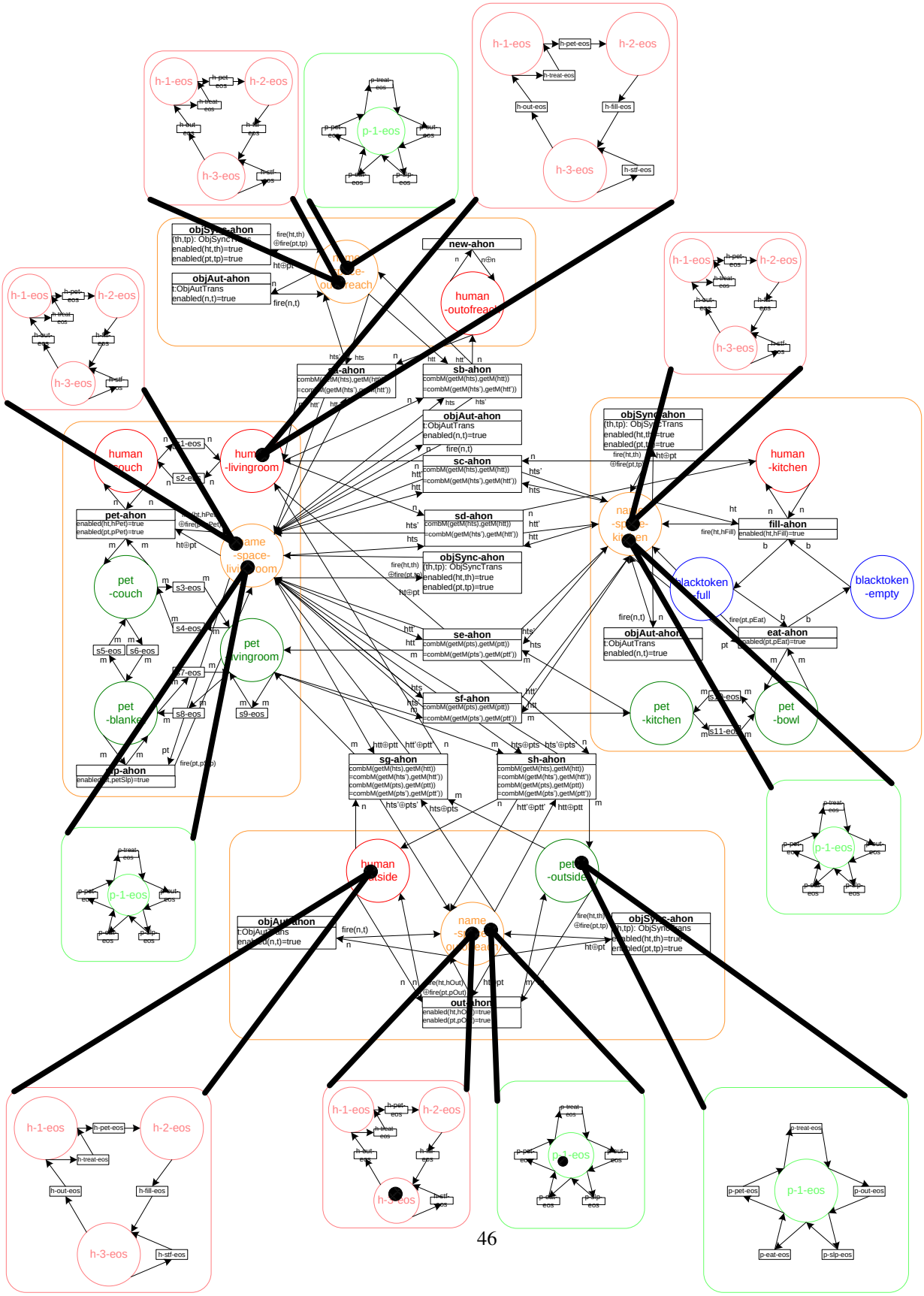

Figure 8.7: *AHO net example after firing sh.ahon*

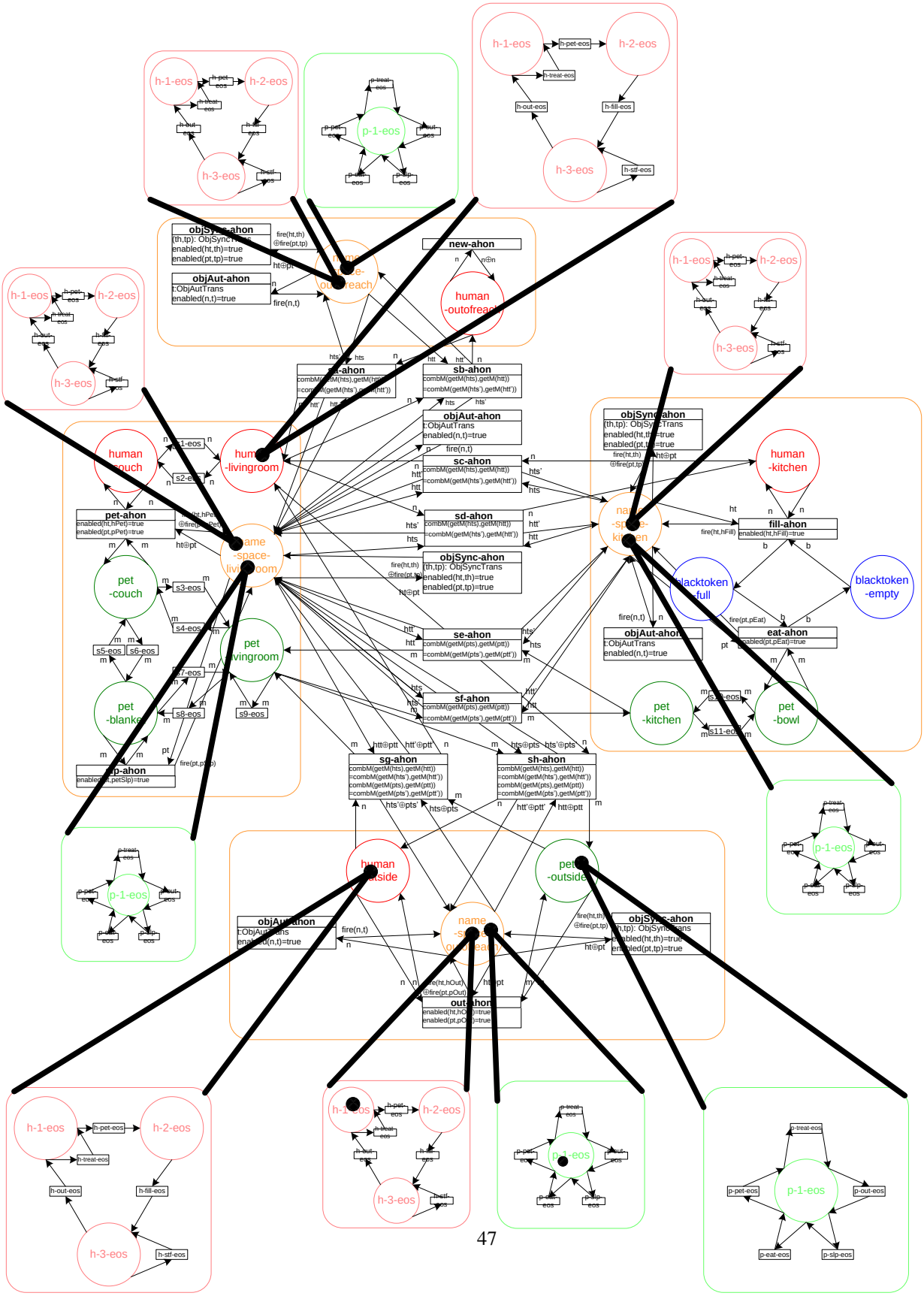

Figure 8.8: *AHO net example after firing out.ahon*

Bibliography

[HEM05] Kathrin Hoffmann, Hartmut Ehrig, and Till Mossakowski. High-level nets with nets and rules as tokens. In Gianfranco Ciardo and Philippe Darondeau, editors, *ICATPN*, volume 3536 of *Lecture Notes in Computer Science*, pages 268–288. Springer, 2005.

[HERP08] Kathrin Hoffmann, Hartmut Ehrig, Alexander Rein, and Julia Padberg. Modeling data dependnent workflows in mobile ad/hoc networks. In *WADT*, 2008.

[Hof05] Kathrin Hoffmann. *Formal Approach and Applictions of Algebraic Higher Order Nets*. PhD thesis, Technische Universität Berlin, 2005.

[KF07] Michael Köhler and Berndt Farwer. Object nets for mobility. In J. Kleijn and A. Yakovlev, editors, *ICATPN 2007*, volume 4546 of *Lecture Notes in Computer Science*, pages 244–262. Springer-Verlag, 2007.

[Val98] Rüdiger Valk. Petri nets as token objects: An introduction to elementary object nets. In Jörg Desel and Manuel Silva, editors, *ICATPN*, volume 1420 of *Lecture Notes in Computer Science*, pages 1–25. Springer, 1998.