Transformations in Reconfigurable Place/Transition Systems*

Ulrike Prange, Hartmut Ehrig, Kathrin Hoffmann, and Julia Padberg

Technische Universität Berlin, Germany
{uprange,ehrig,hoffmann,padberg}@cs.tu-berlin.de

Abstract. Reconfigurable place/transition systems are Petri nets with initial markings and a set of rules which allow the modification of the net during runtime in order to adapt the net to new requirements. For the transformation of Petri nets in the double pushout approach, the categorical framework of adhesive high-level replacement systems has been instantiated to Petri nets. In this paper, we show that also place/transition systems form a weak adhesive high-level replacement category. This allows us to apply the developed theory also to tranformations within reconfigurable place/transition systems.

1 Introduction

Petri nets are an important modeling technique to describe discrete distributed systems. Their nondeterministic firing steps are well-suited for modeling the concurrent behavior of such systems. The formal treatment of Petri nets as monoids by Meseguer and Montanari in [1] has been an important step for a rigorous algebraic treatment and analysis of Petri nets which is also used in this paper.

As the adaptation of a system to a changing environment gets more and more important, Petri nets that can be transformed during runtime have become a significant topic in recent years. Application areas cover e.g. computer supported cooperative work, multi agent systems, dynamic process mining and mobile networks. Moreover, this approach increases the expressiveness of Petri nets and allows for a formal description of dynamic changes.

In [2], the concept of reconfigurable place/transition (P/T) systems was introduced for modeling changes of the net structure while the system is kept running. In detail, a reconfigurable P/T system consists of a P/T system and a set of rules, so that not only the follower marking can be computed but also the net structure can be changed by rule application. So, a new P/T system is obtained that is more appropriate with respect to some requirements of the environment. Moreover, these activities can be interleaved. In [3], the conflict situation of transformation and token firing has been dealt with. In this paper, we give the formal foundation for transformations of P/T systems.

For rule-based transformations of P/T systems we use the framework of adhesive high-level replacement (HLR) systems [4, 5] that is inspired by graph transformation systems [6]. Adhesive HLR systems have been recently introduced as a new categorical framework for graph transformation in the double pushout approach [4, 5]. They

* This work has been partly funded by the research project forMA$_\ell$NET of the German Research Council (see http://tfs.cs.tu-berlin.de/formalnet/).

P. Degano et al. (Eds.): Montanari Festschrift, LNCS 5065, pp. 96–113, 2008.

combine the well-known framework of HLR systems with the framework of adhesive categories introduced by Lack and Sobociński [7]. The main concept behind adhesive categories are the so-called van Kampen squares. These ensure that pushouts along monomorphisms are stable under pullbacks and, vice versa, that pullbacks are stable under combined pushouts and pullbacks. In the case of weak adhesive HLR categories, the class of all monomorphisms is replaced by a subclass $\mathcal{M}$ of monomorphisms closed under composition and decomposition, and for the van Kampen properties certain morphisms have to be additionally $\mathcal{M}$-morphisms.

In this paper, we present the formal foundations for transformations of nets with markings. We show that the category of P/T systems is a weak adhesive HLR category which allows the application of the developed theory also to tranformations within reconfigurable P/T systems. This theory comprises many results concerning the applicability of rules, the embedding and extension of transformations, parallel and sequential dependence and independence, and concurrency of rule applications, and hence gives precise notions for concurrent or conflicting situations in reconfigurable P/T systems. Our work is illustrated by an example in the area of mobile emergency scenarios.

This paper is organized as follows. In Section 2, we introduce weak adhesive HLR categories and adhesive HLR systems. The notion of reconfigurable P/T systems is presented in Section 3. In Section 4, we show that the category $\mathbf{PTSys}$ used for reconfigurable P/T systems is a weak adhesive HLR category. Finally, we give a conclusion and outline related and future work in Section 5.

2 Adhesive HLR Categories and Systems

In this section, we give a short introduction to weak adhesive HLR categories and summarize some important results for adhesive HLR systems (see [4]) which are based on adhesive categories introduced in [7].

The intuitive idea of an adhesive or (weak) adhesive HLR category is a category with suitable pushouts and pullbacks which are compatible with each other. More precisely, the definition is based on so-called van Kampen squares.

The idea of a van Kampen (VK) square is that of a pushout which is stable under pullbacks, and vice versa that pullbacks are stable under combined pushouts and pullbacks.

Definition 1 (van Kampen square). *A pushout (1) is a* van Kampen square *if for any commutative cube (2) with (1) in the bottom and the back faces being pullbacks it holds that: the top face is a pushout if and only if the front faces are pullbacks.*

$$\begin{tikzcd}
A \arrow[r, "m"] \arrow[d, "f"'] \arrow[dr, phantom, "(1)"] & B \arrow[d, "g"] \\
C \arrow[r, "n"'] & D
\end{tikzcd}
\qquad
\begin{tikzcd}
 & & A' \arrow[dll, "f'"'] \arrow[dr, "m'"] \arrow[ddd, "a"] & \\
C' \arrow[dr, "n'"] \arrow[ddd, "c"'] & & & B' \arrow[dll, "g'"'] \arrow[ddd, "b"] \\
 & D' \arrow[ddd, "d"] & & \\
 & & A \arrow[dll, "f"'] \arrow[dr, "m"] & \\
C \arrow[dr, "n"'] & & & B \arrow[dll, "g"] \\
 & D & &
\end{tikzcd}$$

(2)

Not even in the category $\mathbf{Sets}$ of sets and functions each pushout is a van Kampen square. Therefore, in (weak) adhesive HLR categories only those VK squares of Def. 1 are considered where m is in a class $\mathcal{M}$ of monomorphisms. A pushout (1) with $m \in \mathcal{M}$ and arbitrary f is called a pushout along $\mathcal{M}$.

The main difference between (weak) adhesive HLR categories as described in [4, 5] and adhesive categories introduced in [7] is that a distinguished class $\mathcal{M}$ of monomorphisms is considered instead of all monomorphisms, so that only pushouts along $\mathcal{M}$-morphisms have to be VK squares. In the weak case, only special cubes are considered for the VK square property.

Definition 2 ((weak) adhesive HLR category). *A category* $\mathbf{C}$ *with a morphism class* $\mathcal{M}$ *is a* (weak) adhesive HLR category*, if*

1. *$\mathcal{M}$ is a class of monomorphisms closed under isomorphisms, composition ($f : A \to B \in \mathcal{M}, g : B \to C \in \mathcal{M} \Rightarrow g \circ f \in \mathcal{M}$) and decomposition ($g \circ f \in \mathcal{M}, g \in \mathcal{M} \Rightarrow f \in \mathcal{M}$),*
2. *$\mathbf{C}$ has pushouts and pullbacks along $\mathcal{M}$-morphisms and $\mathcal{M}$-morphisms are closed under pushouts and pullbacks,*
3. *pushouts in $\mathbf{C}$ along $\mathcal{M}$-morphisms are (weak) VK squares.*

For a weak VK square, the VK square property holds for all commutative cubes with $m \in \mathcal{M}$ and ($f \in \mathcal{M}$ or $b, c, d \in \mathcal{M}$) (see Def. 1).

Remark 1. $\mathcal{M}$-morphisms closed under pushouts means that given a pushout (1) in Def. 1 with $m \in \mathcal{M}$ it follows that $n \in \mathcal{M}$. Analogously, $n \in \mathcal{M}$ implies $m \in \mathcal{M}$ for pullbacks.

The categories $\mathbf{Sets}$ of sets and functions and $\mathbf{Graphs}$ of graphs and graph morphisms are adhesive HLR categories for the class $\mathcal{M}$ of all monomorphisms. The categories $\mathbf{ElemNets}$ of elementary nets and $\mathbf{PTNet}$ of place/transition nets with the class $\mathcal{M}$ of all corresponding monomorphisms fail to be adhesive HLR categories, but they are weak adhesive HLR categories (see [8]). Elementary Petri nets, also called condition/event nets, have a weight restricted to one, while place/transition nets allow arbitrary finite arc weights. Instead of the original set theoretical notations used in [9, 10] we have used in [4] a more algebraic version based on power set or monoid constructions as introduced in [1].

Now we are able to generalize graph transformation systems, grammars and languages in the sense of [11, 4].

In general, an adhesive HLR system is based on rules (or productions) that describe in an abstract way how objects in this system can be transformed. An application of a rule is called a direct transformation and describes how an object is actually changed by the rule. A sequence of these applications yields a transformation.

Definition 3 (rule and transformation). *Given a (weak) adhesive HLR category $(\mathbf{C}, \mathcal{M})$, a* rule *$prod = (L \xleftarrow{l} K \xrightarrow{r} R)$ consists of three objects L, K and R called left hand side, gluing object and right hand side, respectively, and morphisms $l : K \to L$, $r : K \to R$ with $l, r \in \mathcal{M}$.*

Given a rule $prod = (L \xleftarrow{l} K \xrightarrow{r} R)$ *and an object* G *with a morphism* $m : L \to G$*, called match, a* direct transformation $G \overset{prod,m}{\Longrightarrow} H$ *from* G *to an object* H *is given by the following diagram, where (1) and (2) are pushouts. A sequence* $G_0 \Longrightarrow G_1 \Longrightarrow ... \Longrightarrow G_n$ *of direct transformations is called a* transformation *and is denoted as* $G_0 \overset{*}{\Longrightarrow} G_n$.

$$\begin{tikzcd}
L \arrow[d, "m"'] \arrow[dr, phantom, "(1)"] & K \arrow[l, "l"'] \arrow[r, "r"] \arrow[d, "k"] \arrow[dr, phantom, "(2)"] & R \arrow[d, "n"] \\
G & D \arrow[l, "f"] \arrow[r, "g"'] & H
\end{tikzcd}$$

An adhesive HLR system $AHS = (\mathbf{C}, \mathcal{M}, RULES)$ *consists of a (weak) adhesive HLR category* $(\mathbf{C}, \mathcal{M})$ *and a set of rules* $RULES$.

Remark 2. Note that given a rule $prod$ and a match m pushout (1) is constructed as the pushout complement, which requires a certain gluing condition to be fulfilled.

3 Reconfigurable P/T Systems

In this section, we formalize reconfigurable P/T systems as introduced in [2]. As net formalism we use P/T systems following the notation of "Petri nets are Monoids" in [1].

Definition 4 (P/T system). *A* P/T net *is given by* $PN = (P, T, pre, post)$ *with places* P*, transitions* T*, and pre and post domain functions* $pre, post : T \to P^{\oplus}$.

A P/T system $PS = (PN, M)$ *is a P/T net* PN *with marking* $M \in P^{\oplus}$.

$P^{\oplus}$ is the free commutative monoid over P. The binary operation $\oplus$ leads to the monoid notation, e.g. $M = 2p_1 \oplus 3p_2$ means that we have two tokens on place p_1 and three tokens on p_2. Note that M can also be considered as a function $M : P \to \mathbf{N}$, where only for a finite set $P' \subseteq P$ we have $M(p) \geq 1$ with $p \in P'$. We can switch between these notations by defining $\sum_{p \in P} M(p) \cdot p = M \in P^{\oplus}$. Moreover, for $M_1, M_2 \in P^{\oplus}$ we have $M_1 \leq M_2$ if $M_1(p) \leq M_2(p)$ for all $p \in P$. A transition $t \in T$ is M-enabled for a marking $M \in P^{\oplus}$ if we have $pre(t) \leq M$, and in this case the follower marking M' is given by $M' = M \ominus pre(t) \oplus post(t)$ and $(PN, M) \xrightarrow{t} (PN, M')$ is called a firing step. Note that $\ominus$ is the inverse of $\oplus$, and $M_1 \ominus M_2$ is only defined if we have $M_2 \leq M_1$.

In order to define rules and transformations of P/T systems we introduce P/T morphisms which preserve firing steps by Condition (1) below. Additionally they require that the initial marking at corresponding places is increasing (Condition (2)) or equal (Condition (3)).

Definition 5 (P/T Morphism). *Given P/T systems* $PS_i = (PN_i, M_i)$ *with* $PN_i = (P_i, T_i, pre_i, post_i)$ *for* $i = 1, 2$*, a* P/T morphism $f : (PN_1, M_1) \to (PN_2, M_2)$ *is given by* $f = (f_P, f_T)$ *with functions* $f_P : P_1 \to P_2$ *and* $f_T : T_1 \to T_2$ *satisfying*

(1) $f_P^{\oplus} \circ pre_1 = pre_2 \circ f_T$ *and* $f_P^{\oplus} \circ post_1 = post_2 \circ f_T$*,*
(2) $M_1(p) \leq M_2(f_P(p))$ *for all* $p \in P_1$*.*

Note that the extension $f_P^{\oplus} : P_1^{\oplus} \to P_2^{\oplus}$ *of* $f_P : P_1 \to P_2$ *is defined by* $f_P^{\oplus}(\sum_{i=1}^n k_i \cdot p_i) = \sum_{i=1}^n k_i \cdot f_P(p_i)$*. (1) means that* f *is compatible with pre and post domains, and (2) that the initial marking of* PN_1 *at place* p *is smaller or equal to that of* PN_2 *at* $f_P(p)$*.*

Moreover, the P/T morphism f *is called* strict *if* f_P *and* f_T *are injective and*

(3) $M_1(p) = M_2(f_P(p))$ *for all* $p \in P_1$*.*

P/T systems and P/T morphisms form the category $\mathbf{PTSys}$*, where the composition of P/T morphisms is defined componentwise for places and transitions.*

Remark 3. For our morphisms we do not always have $f_P^{\oplus}(M_1) \leq M_2$. E.g., $M_1 = p_1 \oplus p_2, M_2 = p$ and $f_P(p_1) = f_P(p_2) = p$ implies $f_P^{\oplus}(M_1) = 2p > p = M_2$, but $M_1(p_1) = M_1(p_2) = 1 = M_2(p)$.

P/T Nets and morphisms satisfying (1) form the category $\mathbf{PTNet}$.

Based on the category $\mathbf{PTSys}$ and the morphism class $\mathcal{M}_{strict}$ of all strict P/T morphisms we are now able to define reconfigurable P/T systems. They allow the modification of the net structure using rules and net transformations of P/T systems, which are instantiations of the corresponding categorical concepts defined in Section 2.

Definition 6 (Reconfigurable P/T System). *Given a P/T system* (PN, M) *and a set* $RULES$ *of rules, a* reconfigurable P/T system *is defined by* $((PN, M), RULES)$*.*

Example 1. We will illustrate the main idea of reconfigurable P/T systems in the area of a mobile scenario. This work is part of a collaboration with some research projects where the main focus is on an adaptive workflow management system for mobile ad-hoc networks, specifically targeted to emergency scenarios [1].

Our scenario takes place in an archaeological disaster/recovery mission: after an earthquake, a team (led by a team leader) is equipped with mobile devices (laptops and PDAs) and sent to the affected area to evaluate the state of archaeological sites and the state of precarious buildings. The goal is to draw a situation map in order to schedule restructuring jobs. The team is considered as an overall mobile ad-hoc network in which the team leader's device coordinates the other team members' devices by providing suitable information (e.g. maps, sensible objects, etc.) and assigning activities. For our example, we assume a team consisting of a team leader as picture store device and two team members as camera device and bridge device, respectively. A typical cooperative process to be enacted by a team is shown in Fig. 1 as P/T system (PN_1, M_1), where only the team leader and one of the team members are yet involved in activities.

The work of the team is modeled by firing steps. So to start the activities of the camera device the follower marking of the P/T system (PN_1, M_1) is computed by firing the transition *Select Building*, then the task *Go to Destination* can be executed etc.

As a reaction to changing requirements, rules can be applied to the net. A rule $prod = ((L, M_L) \xleftarrow{l} (K, M_K) \xrightarrow{r} (R, M_R))$ is given by three P/T systems and a span of two

[1] IST FP6 WORKPAD: `http://www.workpad-project.eu`

(PN_1, M_1)

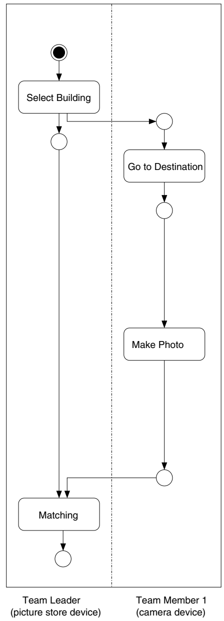

Fig. 1. Cooperative process of the team

strict P/T morphisms l and r (see Def. 3). For the application of the rule to the P/T system (PN_1, M_1), we additionally need a match morphism m that identifies the left-hand side L in PN_1.

The activity of taking a picture can be refined into single steps by the rule $prod_{photo}$, which is depicted in the top row of Fig. 2. The application of this rule to the net (PN_1, M_1) leading to the transformation $(PN_1, M_1) \stackrel{prod_{photo},m}{\Longrightarrow} (PN_2, M_2)$ is shown in Fig. 2.

To predict a situation of disconnection, a movement activity of the bridge device has to be introduced in our system. In more detail, the workflow has to be extended by a task to follow the camera device. For this reason we provide the rule $prod_{follow}$ depicted in the upper row in Fig. 3. Then the transformation step $(PN_2, M_2) \stackrel{prod_{follow},m'}{\Longrightarrow} (PN_3, M_3)$ is shown in Fig. 3.

Summarizing, our reconfigurable P/T system $((PN_1, M_1), \{prod_{photo}, prod_{follow}\})$ consists of the P/T system (PN_1, M_1) and the set of rules $\{prod_{photo}, prod_{follow}\}$ as described above.

Conflicts in Reconfigurable P/T Systems

The traditional concurrency situation in P/T systems without capacities is that two transitions with overlapping pre domain are both enabled and together require more tokens than available in the current marking. As the P/T system can evolve in two different ways, the notions of conflict and concurrency become more complex. We illustrate the situation in Fig. 4, where we have a P/T system (PN_0, M_0) and two transitions that are

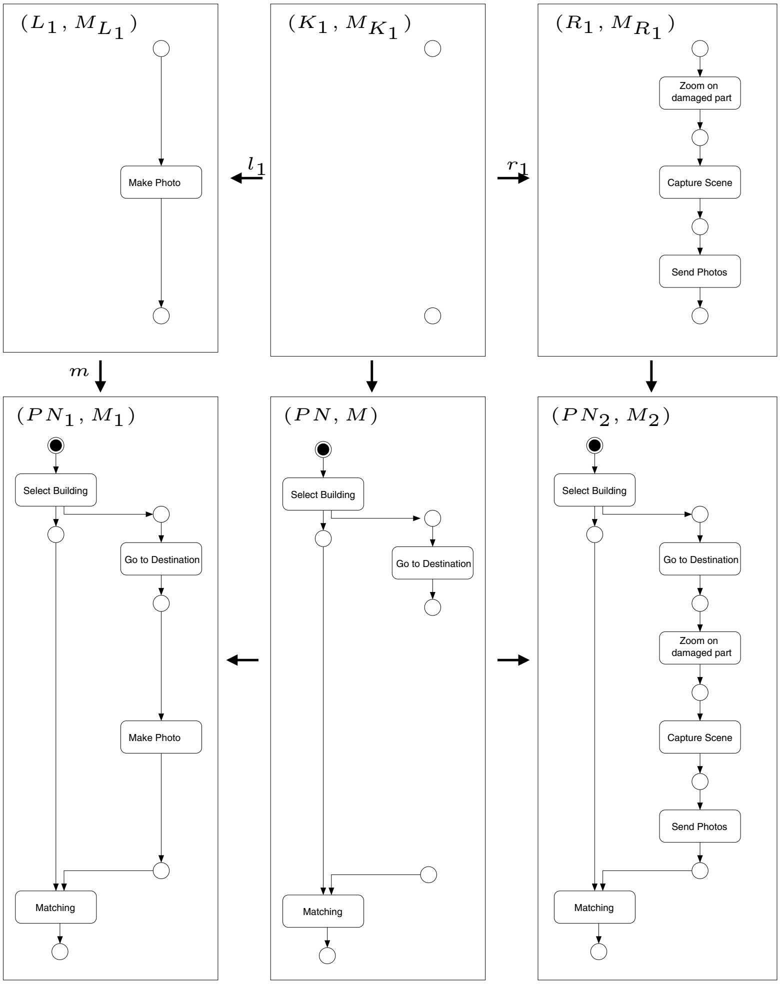

Fig. 2. Transformation step $(PN_1, M_1) \stackrel{prod_{photo},m}{\Longrightarrow} (PN_2, M_2)$

both enabled leading to firing steps $(PN_0, M_0) \xrightarrow{t_1} (PN_0, M_0')$ and $(PN_0, M_0) \xrightarrow{t_2} (PN_0, M_0'')$, and two transformations $(PN_0, M_0) \stackrel{prod_1,m_1}{\Longrightarrow} (PN_1, M_1)$ and $(PN_0, M_0) \stackrel{prod_2,m_2}{\Longrightarrow} (PN_2, M_2)$ via the corresponding rules and matches.

The squares $(1) \ldots (4)$ can be obtained under the following conditions:

For square $(\mathbf{1})$, we have the usual condition for P/T systems that t_1 and t_2 need to be conflict free, so that both can fire in arbitrary order or in parallel yielding the same marking.

For squares $(\mathbf{2})$ **and** $(\mathbf{3})$, we require parallel independence as introduced in [3]. Parallel independence allows the execution of the transformation step and the firing step in arbitrary order leading to the same P/T system. Parallel independence of a transi-

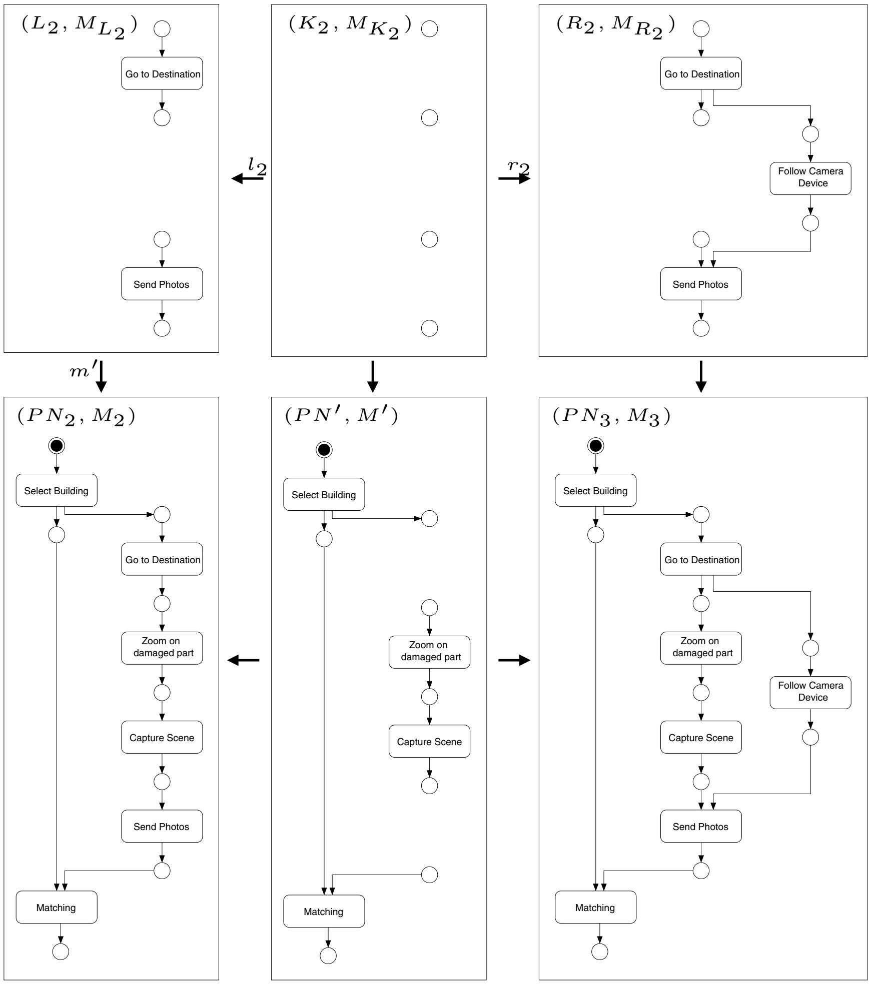

Fig. 3. Transformation step $(PN_2, M_2) \overset{prod_{follow}, m'}{\Longrightarrow} (PN_3, M_3)$

tion and a transformation is given – roughly stated – if the corresponding transition is not deleted by the transformation and the follower marking is still sufficient for the match of the transformation. A detailed formal presentation and analysis of this case is given in [3].

For square (4), we have up to now no conditions to ensure parallel or sequential application of both rules. In this paper, we give these conditions by using results for adhesive HLR systems (see Section 2).

Note that in our framework it is not possible to reduce the conflicts to the case of square (4) by implementing the firing steps by rules. This is due to the fact that the rule morphisms have to be marking strict. Moreover, not only rules but rule schemas would

$$\begin{tikzcd}
(PN_0, M_0''') & (PN_0, M_0') \arrow[l, "t_2"'] \arrow[r, Rightarrow, "{prod_1, m_1'}"] & (PN_1, M_1') \\
(PN_0, M_0'') \arrow[u, "t_1"] \arrow[d, Rightarrow, "{prod_2, m_2'}"'] \arrow[ur, phantom, "(1)"] & (PN_0, M_0) \arrow[l, "t_2"'] \arrow[r, Rightarrow, "{prod_1, m_1}"] \arrow[u, "t_1"'] \arrow[d, Rightarrow, "{prod_2, m_2}"] \arrow[ur, phantom, "(2)"] \arrow[dl, phantom, "(3)"] \arrow[dr, phantom, "(4)"] & (PN_1, M_1) \arrow[u, "t_1'"'] \arrow[d, Rightarrow, "{prod_2, m_2''}"] \\
(PN_2, M_2') & (PN_2, M_2) \arrow[l, "t_2'"] \arrow[r, Rightarrow, "{prod_1, m_1''}"'] & (PN_3, M_3)
\end{tikzcd}$$

Fig. 4. Concurrency in reconfigurable P/T systems

be needed leading to one rule for each kind of transition with n ingoing and m outgoing arcs.

In [4], the following main results for adhesive HLR systems are shown for weak adhesive HLR categories:

1. Local Church-Rosser Theorem,
2. Parallelism Theorem,
3. Concurrency Theorem.

The Local Church-Rosser Theorem allows one to apply two graph transformations $G \Longrightarrow H_1$ via $prod_1$ and $G \Longrightarrow H_2$ via $prod_2$ in an arbitrary order leading to the same result H, provided that they are parallel independent. In this case, both rules can also be applied in parallel, leading to a parallel graph transformation $G \Longrightarrow H$ via the parallel rule $prod_1 + prod_2$. This second main result is called the Parallelism Theorem and requires binary coproducts together with compatibility with $\mathcal{M}$ (i.e. $f, g \in \mathcal{M} \Rightarrow f + g \in \mathcal{M}$). The Concurrency Theorem is concerned with the simultaneous execution of causally dependent transformations, where a concurrent rule $prod_1 * prod_2$ can be constructed leading to a direct transformation $G \Longrightarrow H$ via $prod_1 * prod_2$ (see Ex. 2 in Section 4).

4 P/T Systems as Weak Adhesive HLR Category

In this section, we show that the category $\mathbf{PTSys}$ used for reconfigurable P/T systems together with the class $\mathcal{M}_{strict}$ of strict P/T morphisms is a weak adhesive HLR category. Therefore, we have to verify the properties of Def. 2.

First we shall show that pushouts along $\mathcal{M}_{strict}$-morphisms exist and preserve $\mathcal{M}_{strict}$-morphisms.

Theorem 1. *Pushouts in* $\mathbf{PTSys}$ *along* $\mathcal{M}_{strict}$ *exist and preserve* $\mathcal{M}_{strict}$*-morphisms, i.e. given P/T morphisms* f *and* m *with* m *strict, then the pushout (PO) exists and* n *is also a strict P/T morphism.*

$$\begin{array}{ccc} PS_0 & \xrightarrow{m} & PS_1 \\ {\scriptstyle f}\downarrow & (PO) & \downarrow{\scriptstyle g} \\ PS_2 & \xrightarrow[n]{} & PS_3 \end{array}$$

Construction. Given P/T systems $PS_i = (PN_i, M_i)$ for $i = 0, 1, 2$ and $f, m \in$ $\mathbf{PTSys}$ with $m \in \mathcal{M}_{strict}$ we construct PN_3 as pushout in $\mathbf{PTNet}$, i.e. componentwise in $\mathbf{Sets}$ on places and transitions. The marking M_3 leading to the P/T system $PS_3 = (PN_3, M_3)$ is defined by

(1) $\forall p_1 \in P_1 \backslash m(P_0)$: $M_3(g(p_1)) = M_1(p_1)$

(2) $\forall p_2 \in P_2 \backslash f(P_0)$: $M_3(n(p_2)) = M_2(p_2)$

(3) $\forall p_0 \in P_0$: $M_3(n \circ f(p_0)) = M_2(f(p_0))$

Remark 4. Actually, we have $M_3 = g^{\oplus}(M_1 \ominus m^{\oplus}(M_0)) \oplus n^{\oplus}(M_2)$. (2) and (3) can be integrated, i.e. it is sufficient to define $\forall p_2 \in P_2$: $M_3(n(p_2)) = M_2(p_2)$.

Proof. Since PN_3 is a pushout in $\mathbf{PTNet}$ with g, n jointly surjective we construct a marking for all places $p_3 \in P_3$. (1) and (2) are well-defined because g and n are injective on $P_1 \backslash m(P_0)$ and $P_2 \backslash f(P_0)$, respectively. (3) is well-defined because for $n(f(p_0)) = n(f(p_0'))$, n being injective implies $f(p_0) = f(p_0')$ and hence $M_2(f(p_0)) = M_2(f(p_0'))$.

First we shall show that g, n are P/T morphisms and n is strict.

1. $\forall p_1 \in P_1$ we have:
   1. $p_1 \in P_1 \backslash m(P_0)$ and $M_1(p_1) \overset{(1)}{=} M_3(g(p_1))$ or
   2. $\exists p_0 \in P_0$ with $p_1 = m(p_0)$ and $M_1(p_1) = M_1(m(p_0)) \overset{m \text{ strict}}{=} M_0(p_0) \overset{f \in \mathbf{PTSys}}{\leq} M_2(f(p_0)) \overset{(3)}{=} M_3(n(f(p_0))) = M_3(g(m(p_0))) = M_3(g(p_1))$.

   This means $g \in \mathbf{PTSys}$.
2. $\forall p_2 \in P_2$ we have:
   1. $p_2 \in P_2 \backslash f(P_0)$ and $M_2(p_2) \overset{(2)}{=} M_3(n(p_2))$ or
   2. $\exists p_0 \in P_0$ with $p_2 = f(p_0)$ and $M_2(p_2) = M_2(f(p_0)) \overset{(3)}{=} M_3(n(f(p_0))) = M_3(n(p_2))$.

   This means $n \in \mathbf{PTSys}$ and n is strict.

It remains to show the universal property of the pushout.

Given morphisms $h, k \in \mathbf{PTSys}$ with $h \circ f = k \circ m$, we have a unique induced morphism x in $\mathbf{PTNet}$ with $x \circ n = h$ and $x \circ g = k$. We shall show that $x \in \mathbf{PTSys}$, i.e. $M_3(p_3) \leq M_4(x(p_3))$ for all $p_3 \in P_3$.

$$\begin{tikzcd} PS_0 \arrow[r, "m"] \arrow[d, "f"'] \arrow[dr, phantom, "(PO)"] & PS_1 \arrow[d, "g"] \arrow[ddr, "k"] & \\ PS_2 \arrow[r, "n"] \arrow[drr, "h"'] & PS_3 \arrow[dr, "x"] & \\ & & PS_4 \end{tikzcd}$$

1. For $p_3 = g(p_1)$ with $p_1 \in P_1 \backslash m(P_0)$ we have $M_3(p_3) = M_3(g(p_1)) \stackrel{(1)}{=} M_1(p_1) \stackrel{k \in \mathbf{PTSys}}{\leq} M_4(k(p_1)) = M_4(x(g(p_1))) = M_4(x(p_3))$.
2. For $p_3 = n(p_2)$ with $p_2 \in P_2$ we have $M_3(p_3) = M_3(n(p_2)) \stackrel{(2) \text{ or } (3)}{=} M_2(p_2) \stackrel{h \in \mathbf{PTSys}}{\leq} M_4(h(p_2)) = M_4(x(n(p_2))) = M_4(x(p_3))$. $\square$

As next property, we shall show that pullbacks along $\mathcal{M}_{strict}$-morphisms exist and preserve $\mathcal{M}_{strict}$-morphisms.

Theorem 2. *Pullbacks in* $\mathbf{PTSys}$ *along* $\mathcal{M}_{strict}$ *exist and preserve* $\mathcal{M}_{strict}$-*morphisms, i.e. given P/T morphisms* g *and* n *with* n *strict, then the pullback (PB) exists and* m *is also a strict P/T morphism.*

$$\begin{tikzcd} PS_0 \arrow[r, "m"] \arrow[d, "f"'] \arrow[dr, phantom, "(PB)"] & PS_1 \arrow[d, "g"] \\ PS_2 \arrow[r, "n"'] & PS_3 \end{tikzcd}$$

Construction. Given P/T systems $PS_i = (PN_i, M_i)$ for $i = 1, 2, 3$ and $g, n \in \mathbf{PTSys}$ with $n \in \mathcal{M}_{strict}$ we construct PN_0 as pullback in $\mathbf{PTNet}$, i.e. componentwise in $\mathbf{Sets}$ on places and transitions. The marking M_0 leading to the P/T system $PS_0 = (PN_0, M_0)$ is defined by

$$(*) \;\; \forall p_0 \in P_0 : M_0(p_0) = M_1(m(p_0)).$$

Proof. Obviously, M_0 is a well-defined marking. We have to show that f, m are P/T morphisms and m is strict.

1. $\forall p_0 \in P_0$ we have: $M_0(p_0) \stackrel{(*)}{=} M_1(m(p_0)) \stackrel{g \in \mathbf{PTSys}}{\leq} M_3(g(m(p_0))) = M_3(n(f(p_0))) \stackrel{n \text{ strict}}{=} M_2(f(p_0))$. This means $f \in \mathbf{PTSys}$.
2. $\forall p_0 \in P_0$ we have: $M_0(p_0) \stackrel{(*)}{=} M_1(m(p_0))$, this means $m \in \mathbf{PTSys}$ and m is strict.

It remains to show the universal property of the pullback.

Given morphisms $h, k \in \mathbf{PTSys}$ with $n \circ h = g \circ k$, we have a unique induced morphism x in $\mathbf{PTNet}$ with $f \circ x = h$ and $m \circ x = k$. We shall show that $x \in \mathbf{PTSys}$, i.e. $M_4(p_4) \leq M_0(x(p_4))$ for all $p_4 \in P_4$.

$$\begin{tikzcd}
PS_4 \arrow[dr, "x"] \arrow[drr, "k"] \arrow[ddr, "h"'] & & \\
& PS_0 \arrow[r, "m"] \arrow[d, "f"'] \arrow[dr, phantom, "(PB)"] & PS_1 \arrow[d, "g"] \\
& PS_2 \arrow[r, "n"] & PS_3
\end{tikzcd}$$

For $p_4 \in P_4$ we have $M_4(p_4) \overset{k\in\mathbf{PTSys}}{\leq} M_1(k(p_4)) = M_1(m(x(p_4))) \overset{m \text{ strict}}{=} M_0(x(p_4))$. □

It remains to show the weak VK property for P/T systems. We know that $(\mathbf{PTNet}, \mathcal{M})$ is a weak adhesive HLR category for the class $\mathcal{M}$ of injective morphisms [4, 8], hence pushouts in $\mathbf{PTNet}$ along injective morphisms are van Kampen squares. But we have to give an explicit proof for the markings in $\mathbf{PTSys}$, because diagrams in $\mathbf{PTSys}$ as in Thm. 1 with $m, n \in \mathcal{M}_{strict}$, which are componentwise pushouts in the P- and T-component, are not necessarily pushouts in $\mathbf{PTSys}$, since we may have $M_3(g(p_1)) > M_1(p_1)$ for some $p_1 \in P_1 \backslash m(P_0)$.

Theorem 3. *Pushouts in* $\mathbf{PTSys}$ *along* $\mathcal{M}_{strict}$*-morphisms are weak van Kampen squares.*

Proof. Given the following commutative cube (C) with $m \in \mathcal{M}_{strict}$ and ($f \in \mathcal{M}_{strict}$ or $b, c, d \in \mathcal{M}_{strict}$), where the bottom face is a pushout and the back faces are pullbacks, we have to show that the top face is a pushout if and only if the front faces are pullbacks.

$$\begin{tikzcd}
& & PS'_0 \arrow[dll, "f'"'] \arrow[dr, "m'"] \arrow[dd, "a"] & \\
PS'_2 \arrow[dr, "n'"] \arrow[dd, "c"] & & & PS'_1 \arrow[dll, "g'"'] \arrow[dd, "b"] \\
& PS'_3 \arrow[dd, "d"] & PS_0 \arrow[dll, "f"'] \arrow[dr, "m"] & \\
PS_2 \arrow[dr, "n"] & & & PS_1 \arrow[dll, "g"] \\
(C) & PS_3 & &
\end{tikzcd}$$

"⇒" If the top face is a pushout then the front faces are pullbacks in $\mathbf{PTNet}$, since all squares are pushouts or pullbacks in $\mathbf{PTNet}$, respectively, where the weak VK property holds. For pullbacks as in Thm. 2 with $m, n \in \mathcal{M}_{strict}$, the marking M_0 of PN_0 is completely determined by the fact that $m \in \mathcal{M}_{strict}$. Hence a diagram in $\mathbf{PTSys}$ with $m, n \in \mathcal{M}_{strict}$ is a pullback in $\mathbf{PTSys}$ if and only if it is a pullback in $\mathbf{PTNet}$ if and only if it is a componentwise pullback in $\mathbf{Sets}$. This means, the front faces are also pullbacks in $\mathbf{PTSys}$.

"⇐" If the front faces are pullbacks we know that the top face is a pushout in $\mathbf{PTNet}$. To show that it is also a pushout in $\mathbf{PTSys}$ we have to verify the conditions (1)-(3) from the construction in Thm. 1.

(1) For $p'_1 \in P'_1 \backslash m'(P'_0)$ we have to show that $M'_3(g'(p'_1)) = M'_1(p'_1)$.
If f is strict then also g and g' are strict, since the bottom face is a pushout and the right front face is a pullback, and $\mathcal{M}_{strict}$ is preserved by both pushouts and pullbacks. This means that $M'_1(p'_1) = M'_3(g'(p'_1))$.
Otherwise b and d are strict. Since the right back face is a pullback we have $b(p'_1) \in P_1 \backslash m(P_0)$. With the bottom face being a pushout we have

$$(a)\ M_3(g(b(p'_1))) \overset{(1)}{=} M_1(b(p'_1)).$$

It follows that $M'_3(g'(p'_1)) \overset{d \text{ strict}}{=} M_3(d(g'(p'_1))) = M_3(g(b(p'_1))) \overset{(a)}{=} M_1(b(p'_1)) \overset{b \text{ strict}}{=} M'_1(p'_1)$.

(2) and (3) For $p'_2 \in P'_2$ we have to show that $M'_3(n'(p'_2)) = M'_2(p'_2)$.
With m being strict also n and n' are strict, since the bottom face is a pushout and the left front face is a pullback, and $\mathcal{M}_{strict}$ is preserved by both pushouts and pullbacks. This means that $M'_2(p'_2) = M'_3(n'(p'_2))$.

□

We are now ready to show that the category of P/T systems with the class $\mathcal{M}_{strict}$ of strict P/T morphisms is a weak adhesive HLR category.

Theorem 4. *The category* $(\mathbf{PTSys}, \mathcal{M}_{strict})$ *is a weak adhesive HLR category.*

Proof. By Thm. 1 and Thm. 2, we have pushouts and pullbacks along $\mathcal{M}_{strict}$-morphisms in $\mathbf{PTSys}$, and $\mathcal{M}_{strict}$ is closed under pushouts and pullbacks. Moreover, $\mathcal{M}_{strict}$ is closed under composition and decomposition, because for strict morphisms $f : PS_1 \to PS_2$, $g : PS_2 \to PS_3$ we have $M_1(p) = M_2(f(p)) = M_3(g \circ f(p))$ and $M_1(p) = M_3(g \circ f(p))$ implies $M_1(p) = M_2(f(p)) = M_3(g \circ f(p))$. By Thm. 3, pushouts along strict P/T morphisms are weak van Kampen squares, hence $(\mathbf{PTSys}, \mathcal{M}_{strict})$ is a weak adhesive HLR category. □

Since $(\mathbf{PTSys}, \mathcal{M}_{strict})$ is a weak adhesive HLR category, we can apply the results for adhesive HLR systems given in [4] to reconfigurable P/T systems. Especially, the Local Church-Rosser, Parallelism and Concurrency Theorems as discussed in Section 2 are valid in $\mathbf{PTSys}$, where only for the Parallelism Theorem we need as additional property binary coproducts compatible with $\mathcal{M}_{strict}$, which can be easily verified.

Example 2. If we analyze the two transformations from Ex. 1 in Section 3 depicted in Figs. 2 and 3 we find out that they are sequentially dependent, since $prod_{photo}$ creates the transition *Send Photos* which is used in the match of the transformation $(PN_2, M_2) \overset{prod_{follow}, m'}{\Longrightarrow} (PN_3, M_3)$. In this case, we can apply the Concurrency Theorem and construct a concurrent rule $prod_{conc} = prod_{photo} * prod_{follow}$ that describes the concurrent changes of the net done by the transformations. This rule is depicted in the top row of Fig. 5 and leads to the direct transformation $(PN_1, M_1) \overset{prod_{conc}, m''}{\Longrightarrow} (PN_3, M_3)$, integrating the effects of the two single transformations into one direct one.

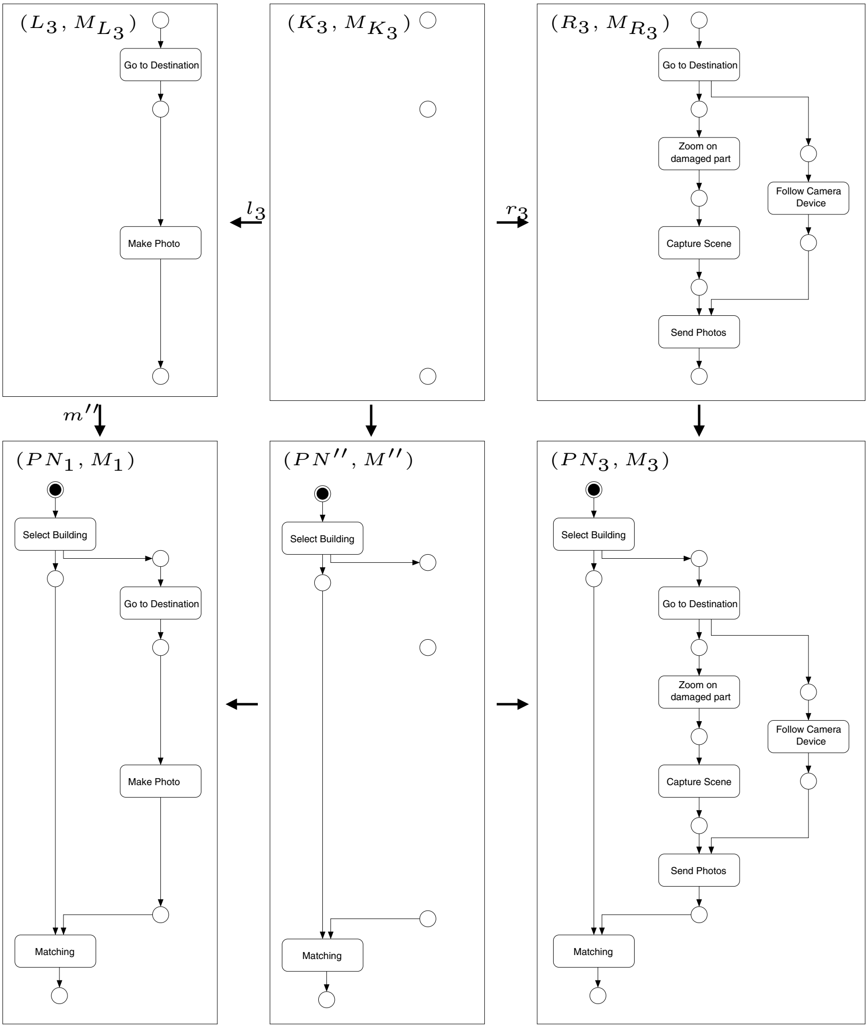

Fig. 5. Direct transformation of (PN_1, M_1) via the concurrent rule $prod_{conc}$

5 Conclusion

In this paper, we have shown that the category $\mathbf{PTSys}$ of P/T systems, i.e. place/transition nets with markings, is a weak adhesive HLR category for the class $\mathcal{M}_{strict}$ of strict P/T morphisms. This allows the application of the rich theory for adhesive HLR systems like the Local Church-Rosser, Parallelismus and Concurrency Theorems to net transformations within reconfigurable P/T systems.

Related Work

Transformations of nets can be considered in various ways. Transformations of Petri nets to a different Petri net class (e.g. in [12, 13, 14]), to another modeling formalism or vice versa (e.g in [15, 16, 17, 18, 19, 20]) are well examined and have yielded many important results. Transformation of one net into another without changing the net class is often used for purposes of forming a hierarchy in terms of reductions or abstraction (e.g. in [21, 22, 23, 24, 25]), or transformations are used to detect specific properties of nets (e.g. in [26, 27, 28, 29]). For the relationship of Petri nets with process algebras and applications to workflow management we refer to [30] and [31], respectively.

Net transformations that aim directly at changing the net in arbitrary ways as known from graph transformations were developed as a special case of HLR systems e.g. in [4]. The general approach can be restricted to transformations that preserve specific properties as safety or liveness (see [14, 32, 33]). Closely related are those approaches that propose changing nets in specific ways in order to preserve specific semantic properties, as behaviour-preserving reconfigurations of open Petri nets (e.g. in [34]), equivalent (I/O-) behavior (e.g in [35, 36]), invariants (e.g. in [37]) or liveness (e.g. in [38, 31]).

In [2], the concept of ”nets and rules as tokens” has been introduced that is most important to model changes of the net structure while the system is kept running, while [3] continues our work by transferring the results of local Church-Rosser, which are well known for term rewriting and graph transformations, to the consecutive evolution of a P/T system by token firing and rule applications. The concept of ”nets and rules as tokens” has been used in [39] for a layered architecture for modeling workflows in mobile ad-hoc networks, so that changes given by net transformation are taken into account and the way consistency is maintained is realized by the way rules are applied.

In [40], rewriting of Petri nets in terms of graph grammars are used for the reconfiguration of nets as well, but this approach lacks the ”nets as tokens”-paradigm.

Future Work

Ongoing work concerns a prototype system for the editing and simulation of such distributed workflows. For the application of net transformation rules, this tool will provide an export to AGG [41], a graph transformation engine as well as a tool for the analysis of graph transformation properties like termination and rule independence. Furthermore, the token net properties could be analyzed using the Petri Net Kernel [42], a tool infrastructure for Petri nets of different net classes.

On the theoretical side, there are other relevant results in the context of adhesive HLR systems which could be interesting to apply within reconfigurable P/T systems. One of them is the Embedding and Extension Theorem, which deals with the embedding of a transformation into a larger context. Another one is the Local Confluence Theorem, also called Critical Pair Lemma, which gives a criterion when two direct transformations are locally confluent. Moreover, it would be interesting to integrate these aspects with those of property preserving transformations, like lifeness and safety, studied in [14, 32, 33]. As future work, it would be important to verify the additional properties necessary for these results.

Another extension will be to consider rules with negative application conditions, which restrict the applicability of a rule by defining structures that are not allowed to exist. In [43], a theory of adhesive HLR systems with negative application conditions is developed, which should be applied and extended to reconfigurable P/T systems.

For the modeling of complex systems, often not only low-level but also high-level Petri nets are used, that combine Petri nets with some data specification [44]. In [8, 45], it is shown that different kinds of algebraic high-level (AHL) nets and systems form weak adhesive HLR categories. More theory for reconfigurable Petri systems based on high-level nets is needed, since the integration of data and data dependencies leads to more appropriate models for many practical problems.

References

[1] Meseguer, J., Montanari, U.: Petri Nets are Monoids. Information and Computation 88(2), 105–155 (1990)

[2] Hoffmann, K., Ehrig, H., Mossakowski, T.: High-Level Nets with Nets and Rules as Tokens. In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, pp. 268–288. Springer, Heidelberg (2005)

[3] Ehrig, H., Hoffmann, K., Padberg, J., Prange, U., Ermel, C.: Independence of Net Transformations and Token Firing in Reconfigurable Place/Transition Systems. In: Kleijn, J., Yakovlev, A. (eds.) ICATPN 2007. LNCS, vol. 4546, pp. 104–123. Springer, Heidelberg (2007)

[4] Ehrig, H., Ehrig, K., Prange, U., Taentzer, G.: Fundamentals of Algebraic Graph Transformation. EATCS Monographs. Springer, Heidelberg (2006)

[5] Ehrig, H., Habel, A., Padberg, J., Prange, U.: Adhesive High-Level Replacement Systems: A New Categorical Framework for Graph Transformation. Fundamenta Informaticae 74(1), 1–29 (2006)

[6] Rozenberg, G. (ed.): Handbook of Graph Grammars and Computing by Graph Transformation: Foundations, vol. 1. World Scientific, Singapore (1997)

[7] Lack, S., Sobociński, P.: Adhesive and Quasiadhesive Categories. Theoretical Informatics and Applications 39(3), 511–546 (2005)

[8] Prange, U.: Algebraic High-Level Nets as Weak Adhesive HLR Categories. Electronic Communications of the EASST 2, 1–13 (2007)

[9] Reisig, W.: Petri Nets. EATCS Monographs on Theoretical Computer Science, vol. 4. Springer, Berlin (1985)

[10] Nielsen, M., Rozenberg, G., Thiagarajan, P.S.: Elementary Transition Systems. Theoretical Computer Science 96(1), 3–33 (1992)

[11] Ehrig, H.: Introduction to the Algebraic Theory of Graph Grammars (A Survey). In: Ng, E.W., Ehrig, H., Rozenberg, G. (eds.) Graph Grammars 1978. LNCS, vol. 73, pp. 1–69. Springer, Heidelberg (1979)

[12] Billington, J.: Extensions to Coloured Petri Nets. In: Proceedings of PNPM 1989, pp. 61–70. IEEE, Los Alamitos (1989)

[13] Campos, J., Sánchez, B., Silva, M.: Throughput Lower Bounds for Markovian Petri Nets: Transformation Techniques. In: Proceedings of PNPM 1991, pp. 322–331. IEEE, Los Alamitos (1991)

[14] Urbášek, M.: Categorical Net Transformations for Petri Net Technology. PhD thesis, TU Berlin (2003)

[15] Belli, F., Dreyer, J.: Systems Modelling and Simulation by Means of Predicate/Transition Nets and Logic Programming. In: Proceedings of IEA/AIE 1994, pp. 465–474 (1994)

[16] Bessey, T., Becker, M.: Comparison of the Modeling Power of Fluid Stochastic Petri Nets (FSPN) and Hybrid Petri Nets (HPN). In: Proceedings of SMC 2002, vol. 2, pp. 354–358. IEEE, Los Alamitos (2002)

[17] de Lara, J., Vangheluwe, H.: Computer Aided Multi-Paradigm Modelling to Process Petri-Nets and Statecharts. In: Corradini, A., Ehrig, H., Kreowski, H.-J., Rozenberg, G. (eds.) ICGT 2002. LNCS, vol. 2505, pp. 239–253. Springer, Heidelberg (2002)

[18] Kluge, O.: Modelling a Railway Crossing with Message Sequence Charts and Petri Nets. In: Ehrig, H., Reisig, W., Rozenberg, G., Weber, H. (eds.) Petri Net Technology for Communication-Based Systems. LNCS, vol. 2472, pp. 197–218. Springer, Heidelberg (2003)

[19] Parisi-Presicce, F.: A Formal Framework for Petri Net Class Transformations. In: Ehrig, H., Reisig, W., Rozenberg, G., Weber, H. (eds.) Petri Net Technology for Communication-Based Systems. LNCS, vol. 2472, pp. 409–430. Springer, Heidelberg (2003)

[20] Corts, L., Eles, P., Peng, Z.: Modeling and Formal Verification of Embedded Systems Based on a Petri Net Representation. Journal of Systems Architecture 49(12-15), 571–598 (2003)

[21] Haddad, S.: A Reduction Theory for Coloured Nets. In: Rozenberg, G. (ed.) APN 1989. LNCS, vol. 424, pp. 209–235. Springer, Heidelberg (1990)

[22] Desel, J.: On Abstraction of Nets. In: Rozenberg, G. (ed.) APN 1991. LNCS, vol. 524, pp. 78–92. Springer, Heidelberg (1991)

[23] Esparza, J., Silva, M.: On the Analysis and Synthesis of Free Choice Systems. In: Rozenberg, G. (ed.) APN 1990. LNCS, vol. 483, pp. 243–286. Springer, Heidelberg (1991)

[24] Chehaibar, G.: Replacement of Open Interface Subnets and Stable State Transformation Equivalence. In: Rozenberg, G. (ed.) APN 1993. LNCS, vol. 674, pp. 1–25. Springer, Heidelberg (1993)

[25] Bonhomme, P., Aygalinc, P., Berthelot, G., Calvez, S.: Hierarchical Control of Time Petri Nets by Means of Transformations. In: Proceedings of SMC 2002, vol. 4, pp. 6–11. IEEE Computer Society Press, Los Alamitos (2002)

[26] Berthelot, G.: Checking Properties of Nets Using Transformation. In: Rozenberg, G. (ed.) APN 1985. LNCS, vol. 222, pp. 19–40. Springer, Heidelberg (1986)

[27] Berthelot, G.: Transformations and Decompositions of Nets. In: Brauer, W., Reisig, W., Rozenberg, G. (eds.) APN 1986. LNCS, vol. 254, pp. 359–376. Springer, Heidelberg (1987)

[28] Best, E., Thielke, T.: Orthogonal Transformations for Coloured Petri Nets. In: Azéma, P., Balbo, G. (eds.) ICATPN 1997. LNCS, vol. 1248, pp. 447–466. Springer, Heidelberg (1997)

[29] Murata, T.: Petri Nets: Properties, Analysis and Applications. In: Proceedings of the IEEE, vol. 77, pp. 541–580. IEEE Computer Society Press, Los Alamitos (1989)

[30] Best, E., Devillers, R., Koutny, M.: The Box Algebra = Petri Nets + Process Expressions. Information and Computation 178(1), 44–100 (2002)

[31] van der Aalst, W.: The Application of Petri Nets to Workflow Management. Journal of Circuits, Systems and Computers 8(1), 21–66 (1998)

[32] Padberg, J., Gajewsky, M., Ermel, C.: Rule-based Refinement of High-Level Nets Preserving Safety Properties. Science of Computer Programming 40(1), 97–118 (2001)

[33] Padberg, J., Urbášek, M.: Rule-Based Refinement of Petri Nets: A Survey. In: Ehrig, H., Reisig, W., Rozenberg, G., Weber, H. (eds.) Petri Net Technology for Communication-Based Systems. LNCS, vol. 2472, pp. 161–196. Springer, Heidelberg (2003)

[34] Baldan, P., Corradini, A., Ehrig, H., Heckel, R., König, B.: Bisimilarity and Behaviour-Preserving Reconfigurations of Open Petri Nets. In: Mossakowski, T., Montanari, U., Haveraaen, M. (eds.) CALCO 2007. LNCS, vol. 4624, pp. 126–142. Springer, Heidelberg (2007)

[35] Balbo, G., Bruell, S., Sereno, M.: Product Form Solution for Generalized Stochastic Petri Nets. IEEE Transactions on Software Engineering 28(10), 915–932 (2002)

[36] Carmona, J., Cortadella, J.: Input/Output Compatibility of Reactive Systems. In: Aagaard, M.D., O'Leary, J.W. (eds.) FMCAD 2002. LNCS, vol. 2517, pp. 360–377. Springer, Heidelberg (2002)
[37] Cheung, T., Lu, Y.: Five Classes of Invariant-Preserving Transformations on Colored Petri Nets. In: Donatelli, S., Kleijn, J. (eds.) ICATPN 1999. LNCS, vol. 1639, pp. 384–403. Springer, Heidelberg (1999)
[38] Esparza, J.: Model Checking Using Net Unfoldings. Science of Computer Programming 23(2-3), 151–195 (1994)
[39] Padberg, J., Hoffmann, K., Ehrig, H., Modica, T., Biermann, E., Ermel, C.: Maintaining Consistency in Layered Architectures of Mobile Ad-hoc Networks. In: Dwyer, M.B., Lopes, A. (eds.) FASE 2007. LNCS, vol. 4422, pp. 383–397. Springer, Heidelberg (2007)
[40] Llorens, M., Oliver, J.: Structural and Dynamic Changes in Concurrent Systems: Reconfigurable Petri Nets. IEEE Transactions on Computers 53(9), 1147–1158 (2004)
[41] AGG Homepage (2007), `http://tfs.cs.tu-berlin.de/agg`
[42] Kindler, E., Weber, M.: The Petri Net Kernel - An Infrastructure for Building Petri Net Tools. Software Tools for Technology Transfer 3(4), 486–497 (2001)
[43] Lambers, L., Ehrig, H., Prange, U., Orejas, F.: Parallelism and Concurrency in Adhesive High-Level Replacement Systems with Negative Application Conditions. Electronic Notes in Theoretical Computer Science (to appear 2008)
[44] Padberg, J., Ehrig, H., Ribeiro, L.: Algebraic High-Level Net Transformation Systems. Mathematical Structures in Computer Science 5(2), 217–256 (1995)
[45] Prange, U.: Algebraic High-Level Systems as Weak Adhesive HLR Categories. Electronic Notes in Theoretical Computer Science (to appear, 2008)