Concurrency in Reconfigurable Place/Transition Systems:

Independence of Net Transformations as well as Net Transformations and Token Firing

Hartmut Ehrig, Claudia Ermel, Kathrin Hoffmann,
Julia Padberg and Ulrike Prange
Institut für Softwaretechnik und Theoretische Informatik
Technische Universität Berlin, Germany
{ehrig,lieske,hoffmann,padberg,uprange}@cs.tu-berlin.de

Forschungsberichte des Fachbereichs Informatik
Bericht-Nr. 2007/02, ISSN 1436-9915

Concurrency in Reconfigurable Place/Transition Systems:*
Independence of Net Transformations as well as Net Transformations and Token Firing

Hartmut Ehrig Claudia Ermel Kathrin Hoffmann Julia Padberg Ulrike Prange

Technical University of Berlin
Institute for Software Technology and Theoretical Computer Science, Sekr. FR6-1
Franklinstr. 28/29, 10587 Berlin, Germany
{ehrig,lieske,hoffmann,padberg,uprange}@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, adhesive high-level replacement systems have been recently introduced as a new categorical framework in the double pushout approach.

In this paper, we analyze concurrency in reconfigurable place/transition systems. We show that place/transition systems are a weak adhesive high-level replacement category, which allows us to apply the developed theory also to tranformations within reconfigurable place/transition systems. Furthermore, we analyze under which conditions net transformations and token firing can be executed in arbitrary order. As an illustrating example, reconfigurable place/transition systems are applied in a mobile network scenario.

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.

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 the 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 a formal description of dynamic changes.

In [23], 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. In this paper, we give the formal foundation for transformations of P/T systems and show how to deal with conflict situations of transformation and token firing.

For rule-based transformations of P/T systems we use the framework of adhesive high-level replacement (HLR) systems [18, 19] that is inspired by graph transformation systems [36]. Adhesive HLR systems have been recently introduced as a new categorical framework for graph transformation in the double pushout approach [18, 19]. They combine the well-known framework of HLR systems with the framework of adhesive categories introduced by Lack and Sobociński [26]. 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 adhesive HLR categories, the class of all monomorphisms is replaced by a subclass $\mathcal{M}$ of monomorphisms closed under composition and decomposition.

Within the framework of adhesive HLR systems, there are many interesting results concerning the applicability of rules, the embedding and extension of transformations, parallel and sequential dependence and independence, and concurrency of rule applications.

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

In this paper, we present the formal foundations for independence of evaluation steps in P/T systems. The next evolution step of such a system can be obtained either by token firing or by the application of one of the available rules. Given two evaluation steps, the question arises whether one of these steps can be postponed after the realization of the other one, yielding the same result. For two firing steps of P/T systems, the independence conditions are well-known. Concerning the independence of transformations, 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 local confluence, parallelism and concurrency, and hence gives precise notions for concurrent or conflicting situations of transformations in reconfigurable P/T systems. Furthermore, we analyze under which conditions a net transformation step and a firing step are independent of each other leading to the notions of parallel, coparallel and sequential independence. 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 reconfigurable P/T systems. The notion of weak adhesive HLR categories and adhesive HLR systems is presented in Section 3. In Section 4, we analyze the applicability of rules to P/T systems and show that the category $\mathbf{PTSys}$ used for reconfigurable P/T systems is a weak adhesive HLR category. Our main theorems concerning the parallel and sequential independence of net transformation and token firing are achieved in Section 6. In Section 7, we show how these concepts and results can be put into the more general framework of algebraic higher-order nets. Finally, we give a conclusion and outline related and future work in Section 8.

2. Reconfigurable P/T Systems

In this section, we formalize reconfigurable P/T systems as introduced in [23]. As net formalism we use P/T systems following the notation of "Petri nets are Monoids" in [28].

Definition 1 (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 $M : P \to \mathbb{N}$, 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 2 (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 *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* $\mathbf{PTNet}$.

Now we are able to define reconfigurable P/T systems, which 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 3.

Definition 3 (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

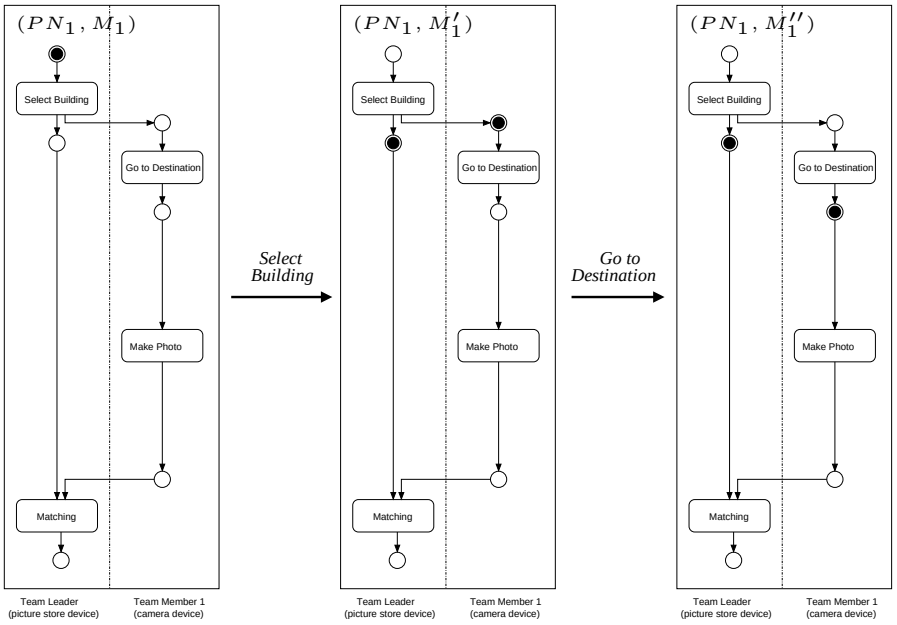


Figure 1. Cooperative process of the team and firing steps *Select Building* and *Go to Destination*

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* leading to the P/T system (PN_1, M_1') in Fig. 1. Afterwards, the next task can be executed by firing the transition *Go to Destination* leading to the P/T system (PN_1, M_1'') 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 strict P/T morphisms l and r (see Def. 6). 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 relevant parts.

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) leads to the transformation $(PN_1, M_1) \overset{prod_{photo}, m}{\Longrightarrow} (PN_2, M_2)$ 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 $(PN_2, M_2) \overset{prod_{follow}, m'}{\Longrightarrow} (PN_3, M_3)$ 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.

[1] MAIS: `http://www.mais-project.it`,
IST FP6 WORKPAD: `http://www.workpad-project.eu/`,
MOBIDIS: `http://www.dis.uniroma1.it/pub/mecella/projects/MobiDIS`

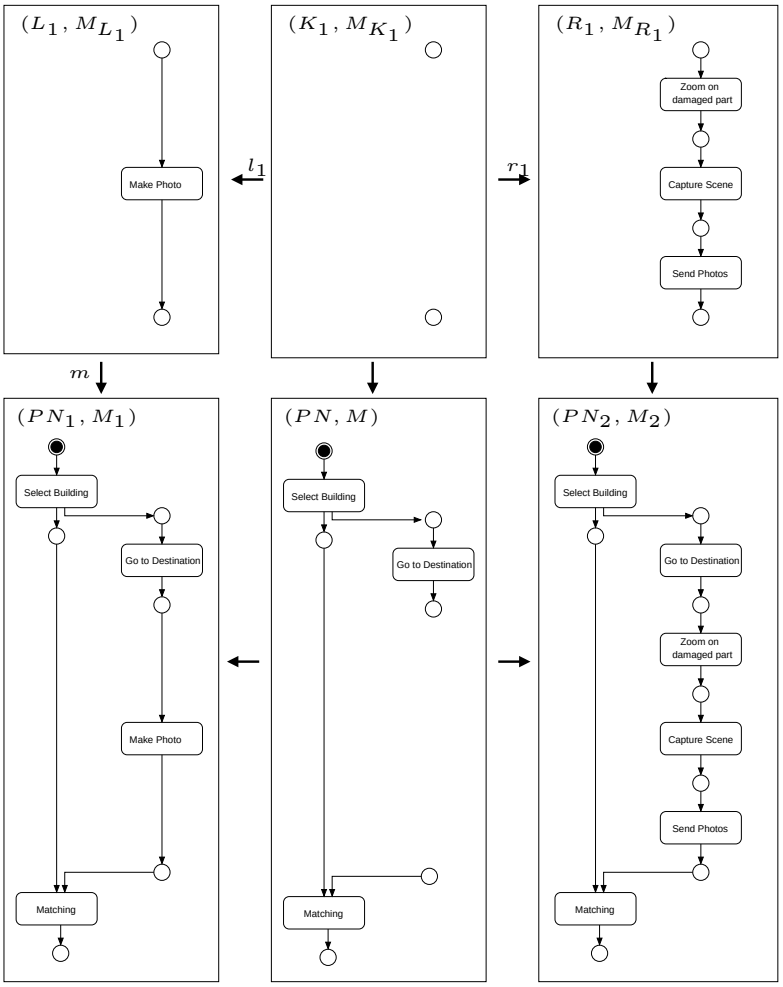


Figure 2. Transformation step $(PN_1, M_1) \overset{prod_{photo}, m}{\Longrightarrow} (PN_2, M_2)$

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 both enabled leading to firing steps $(PN_0, M_0) \overset{t_1}{\longrightarrow} (PN_0, M_0')$ and $(PN_0, M_0) \overset{t_2}{\longrightarrow} (PN_0, M_0'')$, and two transformations $(PN_0, M_0) \overset{prod_1, m_1}{\Longrightarrow} (PN_1, M_1)$ and $(PN_0, M_0) \overset{prod_2, m_2}{\Longrightarrow} (PN_2, M_2)$ 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 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 Section 6. 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 transition 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.

For square $(\mathbf{4})$, we use results for adhesive HLR systems that ensure parallel or sequential application of both rules (see Section 3).

In [18], the following main results for adhesive HLR systems are shown for weak adhesive HLR categories:

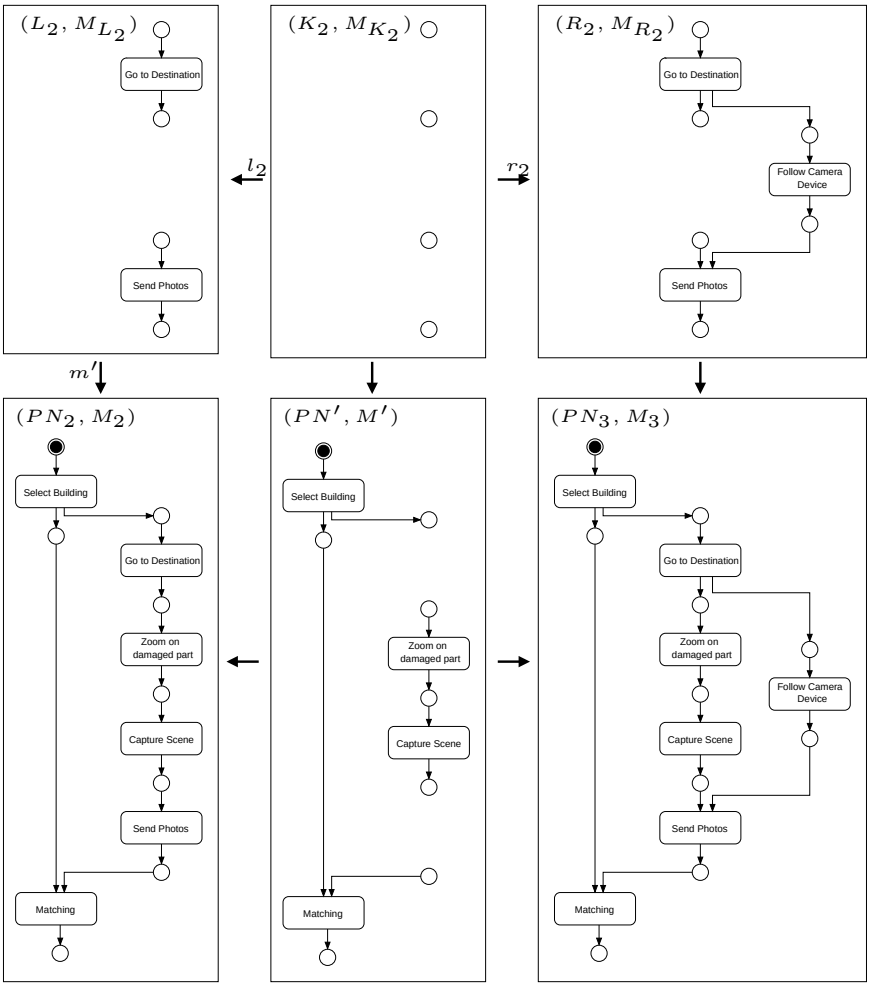


Figure 3. Transformation step $(PN_2, M_2) \overset{prod_{follow}, m'}{\Longrightarrow} (PN_3, M_3)$

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

The Local Church-Rosser Theorem allows one to apply two 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 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).

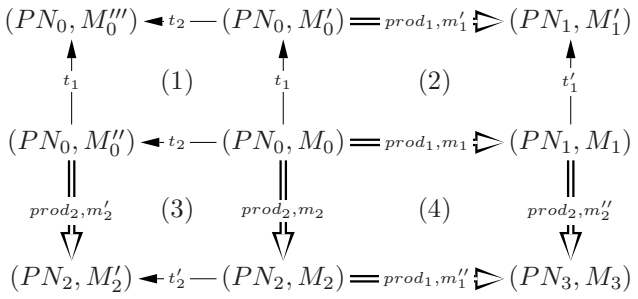


Figure 4. Concurrency in reconfigurable P/T systems

3. Adhesive HLR Categories and Systems

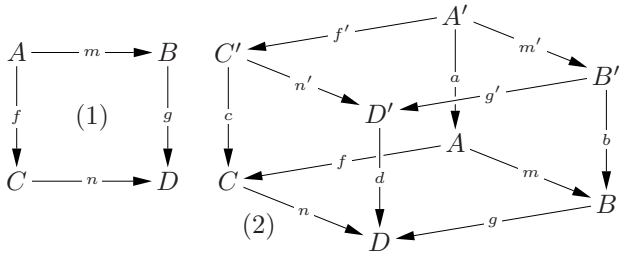
In this section, we give a short introduction to weak adhesive HLR categories and summarize some important re-

sults for adhesive HLR systems (see [18]).

The intuitive idea of (weak) adhesive HLR categories are categories 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 4 (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 holds: the top face is a pushout if and only if the front faces are pullbacks.



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. 4 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 [18, 19] and adhesive categories introduced in [26] 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 5 ((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. 4).

Remark *$\mathcal{M}$-morphisms closed under pushouts means that given a pushout (1) in Def. 4 with $m \in \mathcal{M}$ it follows that $n \in \mathcal{M}$. Analogously, $n \in \mathcal{M}$ implies $m \in \mathcal{M}$ for pullbacks.*

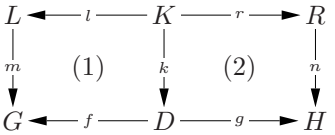
The categories $\mathbf{Sets}$ of sets and functions and $\mathbf{Graphs}$ of graphs and graph morphisms are adhesive HLR categories for the class $\mathcal{M}$ of all monomorphisms. The categories $\mathbf{ElemNets}$ of elementary nets and $\mathbf{PTNet}$ of place/transition nets with the class $\mathcal{M}$ of all corresponding monomorphisms fail to be adhesive HLR categories, but they are weak adhesive HLR categories (see [35]).

Now we are able to generalize graph transformation systems, grammars and languages in the sense of [17, 18].

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 6 (rule and transformation) Given a (weak) adhesive HLR category $(\mathbf{C}, \mathcal{M})$, a *rule* $prod = (L \stackrel{l}{\leftarrow} K \stackrel{r}{\rightarrow} R)$ 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 $prod = (L \stackrel{l}{\leftarrow} K \stackrel{r}{\rightarrow} R)$ and an object G with a morphism $m : L \to G$, called match, a *direct transformation* $G \stackrel{prod,m}{\Longrightarrow} H$ 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 $G_0 \stackrel{*}{\Longrightarrow} G_n$.



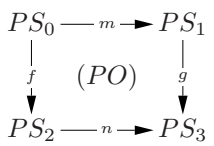
An *adhesive HLR system* $AHS = (\mathbf{C}, \mathcal{M}, RULES)$ consists of a (weak) adhesive HLR category $(\mathbf{C}, \mathcal{M})$ and a set of rules $RULES$.

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. 5.

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

Theorem 1 *Pushouts in* **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.*



Construction *Given* $f, m \in$ **PTSys** *with* $m \in \mathcal{M}_{strict}$ *we construct* PN_3 *as pushout in* **PTNet***, i.e. componentwise in* **Sets** *on places and transitions. The marking* 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 *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 **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 $M_2(f(p_0)) = M_2(f(p_0'))$.

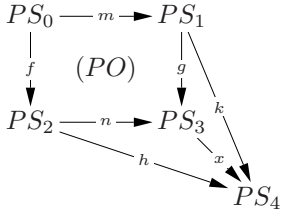
First we shall show that g, n are P/T morphisms and n is strict.

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 $M_1(p_1) = M_1(m(p_0)) \overset{m_0 \text{ strict}}{=} M_0(p_0) \overset{f \in \mathbf{PTSys}}{\leq} M_2(f(p_0)) \overset{(3)}{=} M_3(n(f(p_0))) = M_3(g(m(p_0))) = M_3(g(p_1))$.
   This means $g \in$ **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 $n \in$ **PTSys** and n is strict.

It remains to show the pushout property.

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

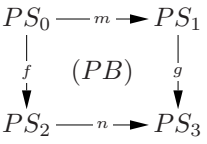


1. For $p_3 = g(p_1)$ with $p_1 \in P_1 \backslash m(P_0)$ we have $M_3(p_3) = M_3(g(p_1)) \overset{(1)}{=} M_1(p_1) \overset{k \in \mathbf{PTSys}}{\leq} M_4(k(p_1)) = M_4(x(g(p_1))) = M_4(x(p_3))$.

2. For $p_3 = n(p_2)$ with $p_2 \in P_2$ we have $M_3(p_3) = M_3(n(p_2)) \overset{(2) \text{ or } (3)}{=} M_2(p_2) \overset{h \in \mathbf{PTSys}}{\leq} M_4(h(p_2)) = M_4(x(n(p_2))) = M_4(x(p_3))$. □

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

Theorem 2 *Pullbacks in* **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.*



Construction *Given* $g, n \in$ **PTSys** *with* $n \in \mathcal{M}_{strict}$ *we construct* PN_0 *as pullback in* **PTNet***, i.e. componentwise in* **Sets** *on places and transitions. The marking* M_0 *is defined by*

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

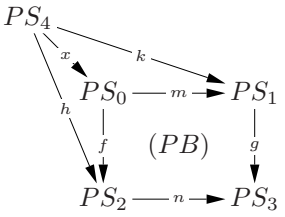
PROOF Obviously, M_0 is a well-defined marking. We have to show that f, m are P/T morphisms and m is strict.

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

2. $\forall p_0 \in P_0$ we have: $M_0(p_0) \overset{(*)}{=} M_1(m(p_0))$, this means $m \in$ **PTSys** and m is strict.

It remains to show the pullback property.

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

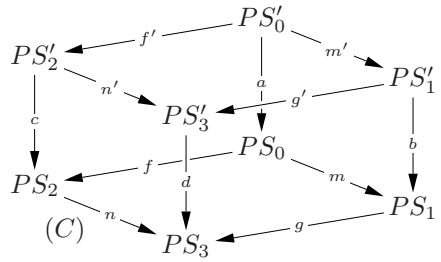


For $p_4 \in P_4$ we have $M_4(p_4) \overset{k \in \mathbf{PTSys}}{\leq} M_1(k(p_4)) = M_1(m(x(p_4))) \overset{m \text{ strict}}{=} M_0(x(p_4))$. □

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 [18, 35], 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 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.



"⇒" 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

$$(a)\ M_3(g(b(p_1'))) \stackrel{(1)}{=} M_1(b(p_1')).$$

It follows that $M_3'(g'(p_1')) \stackrel{d \text{ strict}}{=} M_3(d(g'(p_1'))) = M_3(g(b(p_1'))) \stackrel{(a)}{=} M_1(b(p_1')) \stackrel{b \text{ strict}}{=} M_1'(p_1')$.

(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 [18] to reconfigurable P/T systems. Especially, the Local Church-Rosser, Parallelism and Concurrency Theorems as discussed in Section 3 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 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 $(PN_2, M_2) \stackrel{prod_{follow}, m'}{\Longrightarrow} (PN_3, M_3)$. 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 $(PN_1, M_1) \stackrel{prod_{conc}, m''}{\Longrightarrow} (PN_3, M_3)$, integrating the effects of the two single transformations into one direct one.

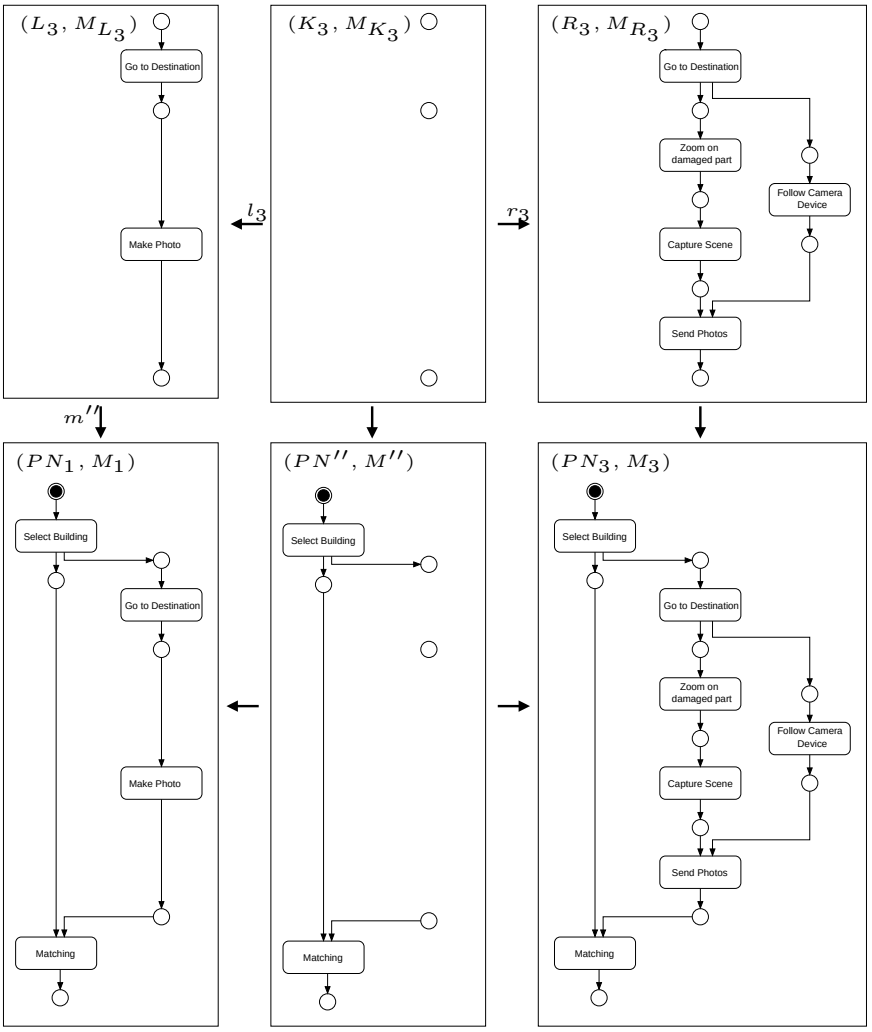


Figure 5. Direct transformation of (PN_1, M_1) via the concurrent rule $prod_{conc}$

5. Applicability of Rules to P/T Systems

In this section, we analyze under which condition a rule can be applied to a P/T system.

For the application of a rule $prod = ((L, M_L) \stackrel{l}{\leftarrow} (K, M_K) \stackrel{r}{\rightarrow} (R, M_R))$ to a P/T system (PN_1, M_1) we need pushouts in the category **PTSys**. Especially we are interested in pushouts of the form (2), where r is a strict and k is a general P/T morphism. The existence of these pushouts has been shown in Thm. 1.

$$\begin{array}{ccccc} (L, M_L) & \xleftarrow{l} & (K, M_K) & \xrightarrow{r} & (R, M_R) \\ \downarrow{\scriptstyle m} & (1) & \downarrow{\scriptstyle k} & (2) & \downarrow{\scriptstyle n} \\ (PN_1, M_1) & \xleftarrow{f} & (PN_0, M_0) & \xrightarrow{g} & (PN_2, M_2) \end{array}$$

Vice versa, for a given match m: $(L, M_L) \rightarrow (PN_1, M_1)$ we have to construct a P/T system (PN_0, M_0) such that (1) becomes a pushout. This construction requires the following gluing condition which has to be satisfied in order to apply a rule at a given match.

Definition 7 (Gluing Condition for P/T Systems) For a rule $prod = ((L, M_L) \stackrel{l}{\leftarrow} (K, M_K) \stackrel{r}{\rightarrow} (R, M_R))$ and a match $m : (L, M_L) \rightarrow (PN_1, M_1)$, the gluing points GP, the dangling points DP and the identification points IP of L are defined by

$$\begin{array}{lcl} GP & = & l(P_K \cup T_K), \\ DP & = & \{p \in P_L \mid \exists t \in (T_1 \setminus m_T(T_L)) : \\ & & \quad m_P(p) \in pre_1(t) \oplus post_1(t)\} \\ IP & = & \{p \in P_L \mid \exists p' \in P_L : \\ & & \quad p \neq p' \wedge m_P(p) = m_P(p')\}, \\ & & \cup \{t \in T_L \mid \exists t' \in T_L : \\ & & \quad t \neq t' \wedge m_T(t) = m_T(t')\}. \end{array}$$

A P/T morphism $m : (L, M_L) \to (PN_1, M_1)$ and a strict morphism $l : (K, M_K) \to (L, M_L)$ satisfy the gluing condition if all dangling and identification points are gluing points, i.e.

$$DP \cup IP \subseteq GP,$$

and m is strict on places to be deleted, i.e.

$$(*)\ \forall p \in P_L \setminus l(P_K) : M_L(p) = M_1(m(p)).$$

Example 3 In Figs. 2 and 3, two example transformations of P/T systems are shown. In both cases, we have the dangling points $DP = P_L$, while the set of identification points IP is empty. So, the given matches satisfy the gluing condition, because the gluing points GP are equal to the sets of places P_L, and all places are preserved.

Note that we have not yet considered the firing of the rule nets (L, M_L), (K, M_K) and (R, M_R) as up to now no relevant use could be found. Nevertheless, from a theoretical point of view the simultaneous firing of the nets (L, M_L), (K, M_K) and (R, M_R) is easy as the morphisms are marking strict. The firing of only one of these nets requires interesting extensions of the gluing condition.

Now we show that the gluing condition is a sufficient and necessary condition for the application of a rule $prod$ via a match m.

Theorem 5 *Given P/T morphisms $l : (K, M_K) \to (L, M_L)$ and $m : (L, M_L) \to (PN_1, M_1)$ with l being strict, then the pushout complement (PN_0, M_0) together with P/T morphisms $k : (K, M_K) \to (PN_0, M_0)$ and $f : (PN_0, M_0) \to (PN_1, M_1)$ exists in* **PTSys** *if and only if the gluing condition is satisfied.*

PROOF If the pushout complement exists, then we have $DP \cup IP \subseteq GP$ as in **PTNet**, and the gluing condition for markings is satisfied by the pushout construction for markings (see Thm. 1).

Vice versa, given the gluing condition we construct PN_0 as the pushout complement in **PTNet**, which is unique up to isomorphism, and define M_0 by

$$(4)\ \forall p_0 \in P_0 : M_0(p_0) = M_1(f(p_0)).$$

Now let M_1' be the marking of PN_1 defined by the pushout construction in Thm. 1, i.e.

(1) $\forall p \in P_L \setminus l(P_K)$: $M_1'(m(p)) = M_L(p)$

(2) and (3) $\forall p_0 \in P_0$: $M_1'(f(p_0)) = M_0(p_0)$

We have to show that $M_1 = M_1'$.

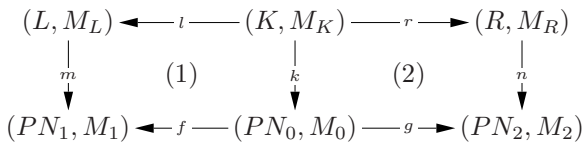
(1) $\forall p \in P_L \setminus l(P_K)$ we have $M_1'(m(p)) \overset{(1)}{=} M_L(p) \overset{(*)}{=} M_1(m(p))$.

(2) and (3) $\forall p_0 \in P_0$ we have $M_1'(f(p_0)) \overset{(2) \text{ or } (3)}{=} M_0(p_0) \overset{(4)}{=} M_1(f(p_0))$.

This means $M_1 = M_1'$ and (PN_0, M_0) is the pushout complement of m and l. □

Remark *The uniqueness of the pushout complement follows from the fact that* **PTSys** *is a weak adhesive HLR category (see Thm. 4).*

Theorem 6 *A rule $prod = ((L, M_L) \overset{l}{\leftarrow} (K, M_K) \overset{r}{\to} (R, M_R))$ is applicable at a match $m : (L, M_L) \to (PN_1, M_1)$ if and only if the gluing condition is satisfied for l and m. In this case, we obtain a P/T system (PN_0, M_0) leading to a net transformation step $(PN_1, M_1) \overset{prod,m}{\Longrightarrow} (PN_2, M_2)$ consisting of the following pushout diagrams (1) and (2). The P/T morphism $n : (R, M_R) \to (PN_2, M_2)$ is called comatch of the transformation.*



PROOF This follows directly from Thms. 1 and 5. □

6. Independence of Net Transformation and Token Firing

In this section we analyze under which conditions a net transformation and a firing step of a reconfigurable P/T system as introduced in Section 2 can be executed in arbitrary order. These conditions are called (co-)parallel and sequential independence.

We start with the situation where a transformation step and a firing step are applied to the same P/T system. This leads to the notion of parallel independence.

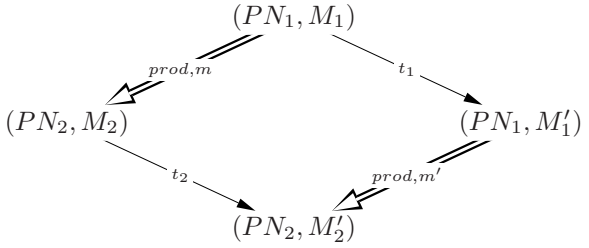
Definition 8 (Parallel Independence) Given a production $prod = ((L, M_L) \overset{l}{\leftarrow} (K, M_K) \overset{r}{\to} (R, M_R))$, a transformation step $(PN_1, M_1) \overset{prod,m}{\Longrightarrow} (PN_2, M_2)$ of P/T systems and a firing step $(PN_1, M_1) \overset{t_1}{\longrightarrow} (PN_1, M_1')$ for a transition $t_1 \in T_1$, the transformation and the firing step are called parallel independent if

(1) t_1 is not deleted by the transformation step and

(2) $M_L(p) \leq M_1'(m(p))$ for all $p \in P_L$.

Parallel independence allows the execution of the transformation step and the firing step in arbitrary order leading to the same P/T system.

Theorem 7 *Given parallel independent steps* $(PN_1, M_1) \overset{prod,m}{\Longrightarrow} (PN_2, M_2)$ *and* $(PN_1, M_1) \xrightarrow{t_1} (PN_1, M_1')$ *with* $t_1 \in T_1$ *then there is a corresponding* $t_2 \in T_2$ *with firing step* $(PN_2, M_2) \xrightarrow{t_2} (PN_2, M_2')$ *and a transformation step* $(PN_1, M_1') \overset{prod,m'}{\Longrightarrow} (PN_2, M_2')$ *with the same marking* M_2'.



Remark *In Def. 8, Cond.* (1) *is needed to fire* t_2 *in* (PN_2, M_2)*, and Cond.* (2) *is needed to obtain a valid match* m' *in* (PN_1, M_1')*. Note that* $m'(x) = m(x)$ *for all* $x \in P_L \cup T_L$.

PROOF Parallel independence implies that $t_1 \in T_1$ is preserved by the transformation step $(PN_1, M_1) \overset{prod,m}{\Longrightarrow} (PN_2, M_2)$. Hence there is a unique $t_0 \in T_0$ with $l^*(t_0) = t_1$. Let $t_2 = r^*(t_0) \in T_2$ in the following pushouts (1) and (2), where l^* and r^* are strict.

$$\begin{array}{ccccc} (L, M_L) & \xleftarrow{l} & (K, M_K) & \xrightarrow{r} & (R, M_R) \\ \downarrow{\scriptstyle m} & (1) & \downarrow & (2) & \downarrow{\scriptstyle n} \\ (PN_1, M_1) & \xleftarrow{l^*} & (PN_0, M_0) & \xrightarrow{r^*} & (PN_2, M_2) \end{array}$$

Now t_1 being enabled under M_1 in PN_1 implies $pre_1(t_1) \leq M_1$. Moreover, l^* and r^* strict implies $pre_0(t_0) \leq M_0$ and $pre_2(t_2) \leq M_2$. Hence t_2 is enabled under M_2 in PN_2 and we define $M_2' = M_2 \ominus pre_2(t_2) \oplus post_2(t_2)$.

Now we consider the second transformation step, with m' defined by $m'(x) = m(x)$ for $x \in P_L \cup T_L$.

$$\begin{array}{ccccc} (L, M_L) & \xleftarrow{l} & (K, M_K) & \xrightarrow{r} & (R, M_R) \\ \downarrow{\scriptstyle m'} & (1') & \downarrow & (2') & \downarrow{\scriptstyle n'} \\ (PN_1, M_1') & \xleftarrow{l'^*} & (PN_0, M_0') & \xrightarrow{r'^*} & (PN_2, M_2') \end{array}$$

m' is a P/T morphism if for all $p \in P_L$ we have

(a) $M_L(p) \leq M_1'(m'(p))$,

and the match m' is applicable at M_1' if

(b) $IP \cup DP \subseteq GP$ and for all $p \in P_L \backslash l(P_K)$ we have $M_L(p) = M_1'(m(p))$ (see gluing condition in Def. 7).

Cond. (a) is given by Cond. (2) in Def. 8, because we assume that $(PN_1, M_1) \overset{prod,m}{\Longrightarrow} (PN_2, M_2)$ and $(PN_1, M_1) \xrightarrow{t_1} (PN_1, M_1')$ with $t_1 \in T_1$ are parallel independent. Moreover, the match m being applicable at M_1 implies $IP \cup DP \subseteq GP$, and for all $p \in P_L \backslash l(P_K)$ we have $M_L(p) = M_1(m(p)) = M_1'(m(p))$ by Lem. 1 below using the fact that there is a firing step $(PN_1, M_1) \xrightarrow{t_1} (PN_1, M_1')$. The application of $prod$ along m' leads to the P/T system (PN_2, M_2''), where $l'^*(x) = l^*(x)$, $r'^*(x) = r^*(x)$ for all $x \in P_0 \cup T_0$, and $n'(x) = n(x)$ for all $x \in P_R \cup T_R$.

Finally, it remains to show that $M_2' = M_2''$. By construction of the transformation steps $(PN_1, M_1) \overset{prod,m}{\Longrightarrow} (PN_2, M_2)$ and $(PN_1, M_1') \overset{prod,m'}{\Longrightarrow} (PN_2, M_2'')$ we have

(1) $\forall p_0 \in P_0$: $M_2(r^*(p_0)) = M_0(p_0) = M_1(l^*(p_0))$,

(2) $\forall p \in P_R \backslash r(P_K)$: $M_2(n(p)) = M_R(p)$,

(3) $\forall p_0 \in P_0$: $M_2''(r^*(p_0)) = M_0'(p_0) = M_1'(l^*(p_0))$ and

(4) $\forall p \in P_R \backslash r(P_K)$: $M_2''(n'(p)) = M_R(p)$.

By construction of the firing steps $(PN_1, M_1) \xrightarrow{t_1} (PN_1, M_1')$ and $(PN_2, M_2) \xrightarrow{t_2} (PN_2, M_2')$ we have

(5) $\forall p_1 \in P_1$: $M_1'(p_1) = M_1(p_1) \ominus pre_1(t_1)(p_1) \oplus post_1(t_1)(p_1)$ and

(6) $\forall p_2 \in P_2$: $M_2'(p_2) = M_2(p_2) \ominus pre_2(t_2)(p_2) \oplus post_2(t_2)(p_2)$.

Moreover, l^* and r^* strict implies the injectivity of l^* and r^* and we have

(7) $\forall p_0 \in P_0$:
$pre_0(t_0)(p_0) = pre_1(t_1)(l^*(p_0)) = pre_2(t_2)(r^*(p_0))$
and
$post_0(t_0)(p_0) = post_1(t_1)(l^*(p_0)) = post_2(t_2)(r^*(p_0))$.

To show that this implies

(8) $M_2' = M_2''$,

it is sufficient to show

(8a) $\forall p \in P_R \backslash r(P_K)$: $M_2''(n'(p)) = M_2'(n(p))$ and

(8b) $\forall p_0 \in P_0$: $M_2''(r^*(p_0)) = M_2'(r^*(p_0))$.

First we show that condition (8a) is satisfied. For all $p \in P_R \backslash r(P_K)$ we have

$$M_2''(n'(p)) \overset{(4)}{=} M_R(p) \overset{(2)}{=} M_2(n(p)) \overset{(6)}{=} M_2'(n(p)),$$

because $n(p)$ is neither in the pre domain nor in the post domain of t_2, which are in $r^*(P_0)$ because t_2 is not created by the rule (see Lem. 1, applied to the inverse rule $prod^{-1}$).

Next we show that condition (8b) is satisfied. For all $p_0 \in P_0$ we have

$$\begin{aligned} M_2''(r^*(p_0)) &\overset{(3)}{=} M_0'(p_0) \\ &\overset{(3)}{=} M_1'(l^*(p_0)) \\ &\overset{(5)}{=} M_1(l^*(p_0)) \ominus pre_1(t_1)(l^*(p_0)) \\ &\quad \oplus post_1(t_1)(l^*(p_0)) \\ &\overset{(1),(7)}{=} M_2(r^*(p_0)) \ominus pre_2(t_2)(r^*(p_0)) \\ &\quad \oplus post_2(t_2)(r^*(p_0)) \\ &\overset{(6)}{=} M_2'(r^*(p_0)) \end{aligned}$$

□

It remains to show Lemma 1 which is used in the proof of Thm. 7.

Lemma 1 *For all $p \in P_L \backslash l(P_K)$ we have $m(p) \notin dom(t_1)$, where $dom(t_1)$ is union of pre and post domain of t_1, and t_1 is not deleted.*

PROOF Assume $m(p) \in dom(t_1)$.

Case 1 ($t_1 = m(t)$ for $t \in T_L$): t_1 not being deleted implies $t \in l(T_K)$. Hence there exists $p' \in dom(t) \subseteq l(P_K)$, such that $m(p') = m(p)$; but this is a contradiction to $p \in P_L \backslash l(P_K)$ and the fact that m cannot identify elements of $l(P_K)$ and $P_L \backslash l(P_K)$.

Case 2 ($t_1 \notin m(T_L)$): $m(p) \in dom(t_1)$ implies by the gluing condition in Def. 7 that $p \in l(P_K)$, but this is a contradiction to $p \in P_L \backslash l(P_K)$.

□

Example 4 Analogously to Fig. 1, firing the transition *Select Building* in (PN_2, M_2) leads to the firing step $(PN_2, M_2) \overset{\text{Select Building}}{\longrightarrow} (PN_2, M_2')$. This firing step and the transformation step $(PN_2, M_2) \overset{prod_{follow}, m'}{\Longrightarrow} (PN_3, M_3)$ (see Fig. 3) are parallel independent because the transition *Select Building* is not deleted by the transformation step and the marking M_L is empty. Thus, the firing step can be postponed after the transformation step or, vice versa, the rule $prod_{follow}$ can be applied after token firing yielding the same result (PN_3, M_3') in Fig. 6.

In contrast, the firing step $(PN_2, M_2') \overset{\text{Go to Destination}}{\longrightarrow} (PN_2, M_2'')$ and the transformation step $(PN_2, M_2') \overset{prod_{follow}, m'''}{\Longrightarrow} (PN_3, M_3')$ (see Fig. 7) are not parallel independent because the transition *Go to Destination* is deleted by the transformation step, i.e. it is not included in the interface K. In fact, the new transition

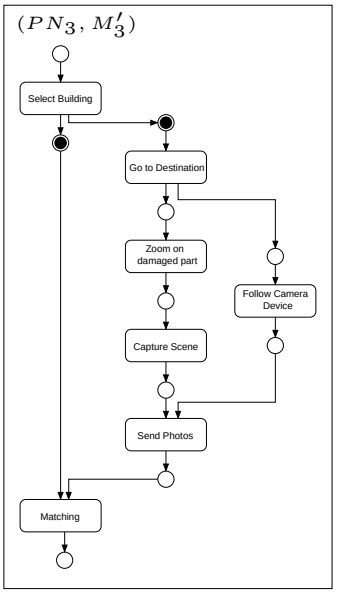
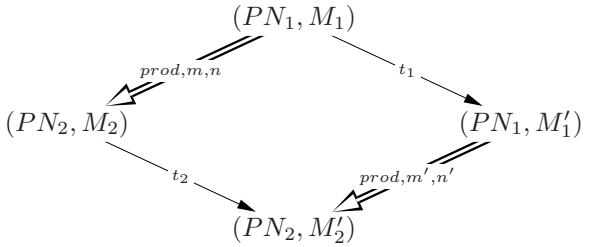


Figure 6. P/T-system (PN_3, M_3')

Go to Destination in (PN_3, M_3') could be fired leading to (PN_3, M_3'') and vice versa we could apply $prod_{follow}$ to (PN_2, M_2'') leading to the P/T system (PN_3, M_3'''), but $M_3'' \neq M_3'''$.

In the first diagram in Thm. 7, we have required that the upper pair of steps is parallel independent leading to the lower pair of steps. Now we consider the situations that the left, right or lower pairs of steps are given – with a suitable notion of independence – such that the right, left and upper pairs of steps can be constructed, respectively.

Definition 9 (Sequential and Coparallel Independence) Given the following diagram with $prod = ((L, M_L) \overset{l}{\leftarrow} (K, M_K) \overset{r}{\rightarrow} (R, M_R))$, matches m and m' with $m(x) = m'(x)$ for $x \in P_L \cup T_L$, and comatches n and n' with $n(x) = n'(x)$ for $x \in P_R \cup T_R$, we say that



1. the left pair of steps, short $((prod, m, n), t_2)$, is sequentially independent if
   (a) t_2 is not created by the transformation step and
   (b) $M_R(p) \leq M_2'(n(p))$ for all $p \in P_R$,

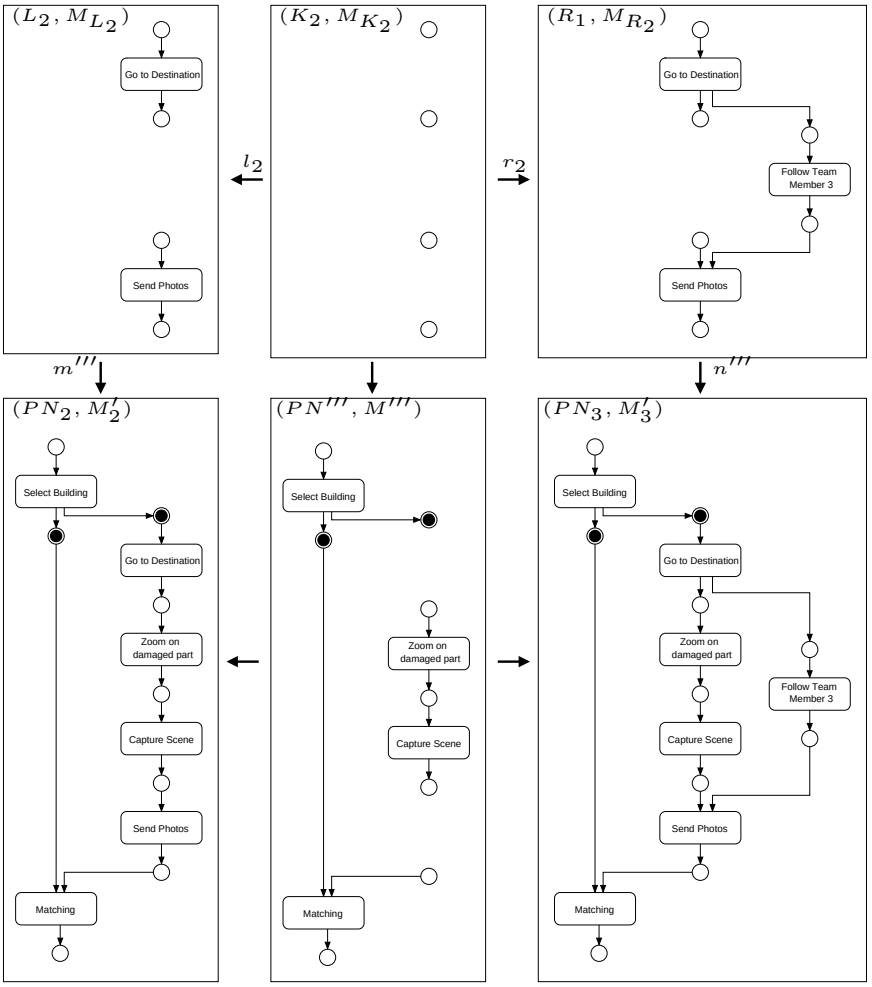


Figure 7. Transformation step $(PN_2, M_2') \overset{prod_{follow}, m'''}{\Longrightarrow} (PN_3, M_3')$

2. the right pair of steps, short $(t_1, prod(m', n'))$, is sequentially independent if
   (a) t_1 is not deleted by the transformation step and
   (b) $M_L(p) \leq M_1(m'(p))$ for all $p \in P_L$,
3. the lower pair of steps, short $(t_2, (prod, m', n'))$, is coparallel independent if
   (a) t_2 is not created by the transformation step and
   (b) $M_R(p) \leq M_2(n'(p))$ for all $p \in P_R$.

Example 5 The pair of steps (*Select Building*, $(prod_{follow}, m''', n''')$) depicted in Fig. 8 is sequentially independent because the transition *Select Building* is not deleted by the transformation step and the marking M_L is empty.

Analogously, the pair of steps $((prod_{follow}, m', n')$, *Select Building*) depicted in Fig. 9 is sequentially independent because the transition *Select Building* is not created by the transformation step and the marking M_R is empty.

For the same reason the pair (*Select Building*, $(prod_{follow}, m''', n''')$) is coparallel independent.

Remark *Note that for* $prod = ((L, M_L) \overset{l}{\leftarrow} (K, M_K) \overset{r}{\rightarrow} (R, M_R))$ *we have* $prod^{-1} = ((R, M_R) \overset{r}{\leftarrow} (K, M_K) \overset{l}{\rightarrow} (L, M_L))$ *and each direct transformation* $(PN_1, M_1) \overset{prod, m}{\Longrightarrow} (PN_2, M_2)$ *with match* m*, comatch* n *and pushout diagrams* (1) *and* (2) *as given in Def. 6 leads to a direct transformation* $(PN_2, M_2) \overset{prod^{-1}, n}{\Longrightarrow} (PN_1, M_1)$ *with match* n *and comatch* m *by interchanging pushout diagrams* (1) *and* (2).

Given a firing step $(PN_1, M_1) \xrightarrow{t_1} (PN_1, M_1')$ *with* $M_1' = M_1 \ominus pre_1(t_1) \oplus post_1(t_1)$ *we can formally define an inverse firing step* $(PN_1, M_1') \xrightarrow{t_1^{-1}} (PN_1, M_1)$ *with* $M_1 = M_1' \ominus post_1(t_1) \oplus pre_1(t_1)$ *if* $post_1(t_1) \leq M_1'$*, such*

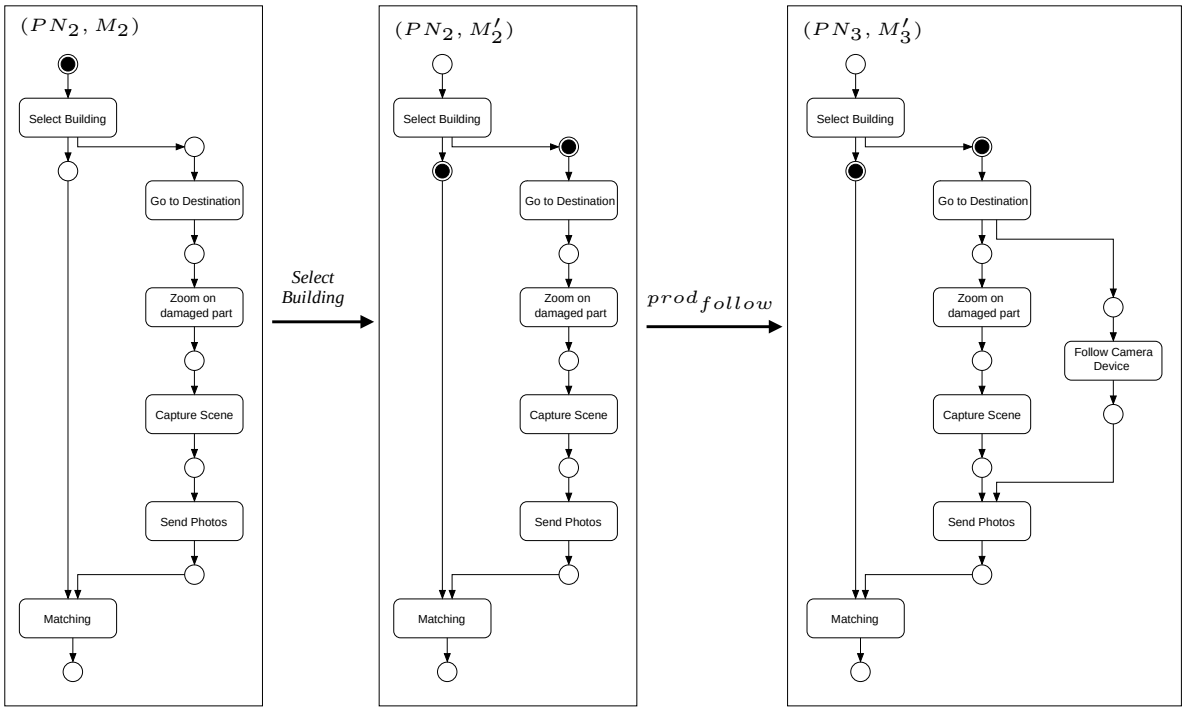


Figure 8. Pair of steps $(\textit{Select Building}, (prod_{follow}, m''', n'''))$

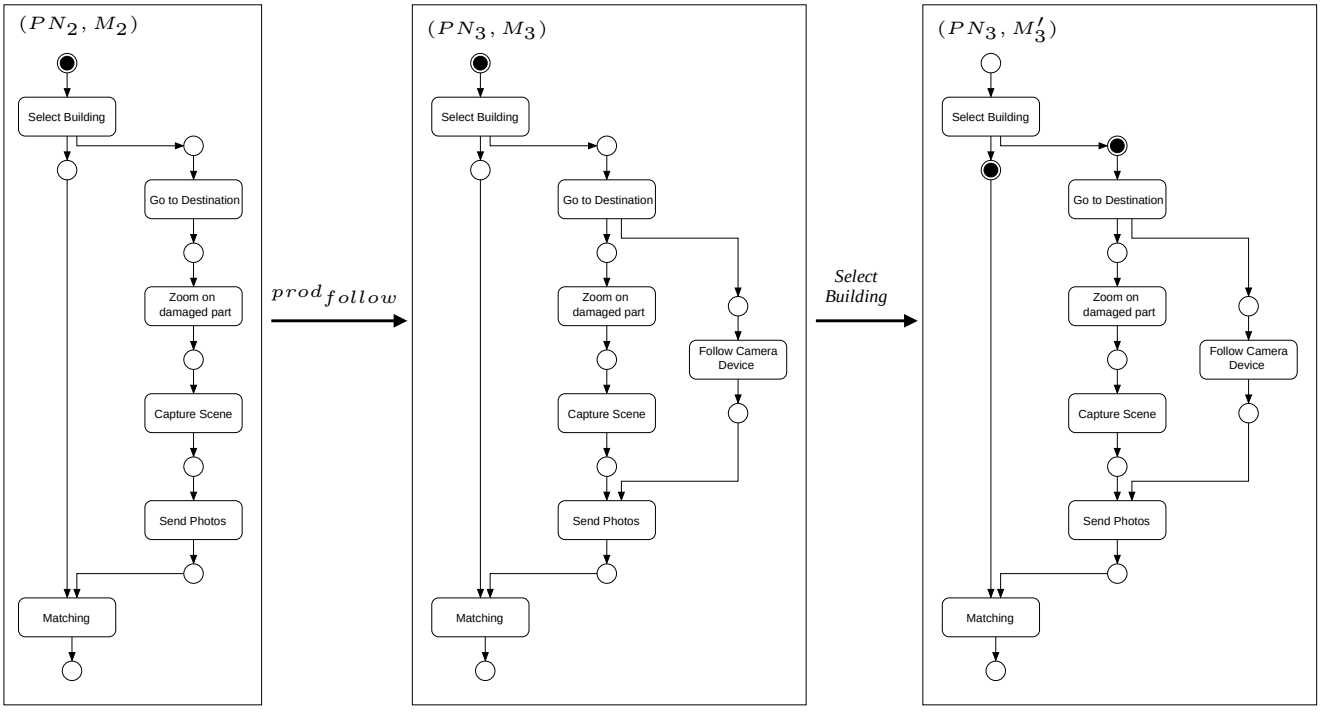
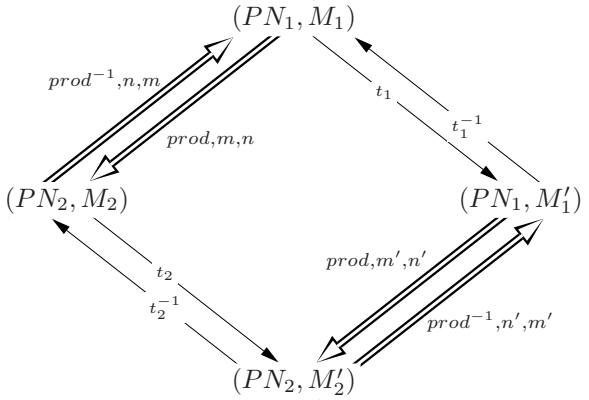


Figure 9. Pair of steps $((prod_{follow}, m', n'), \textit{Select Building})$

that firing and inverse firing are inverse to each other.

Formally, all the notions of independence in Def. 9 can be traced back to parallel independence using inverse transformation steps based on $(prod^{-1}, n, m)$ and $(prod^{-1}, n', m')$ and inverse firing steps t_1^{-1} and t_2^{-1} in the following diagram.



Then we have:

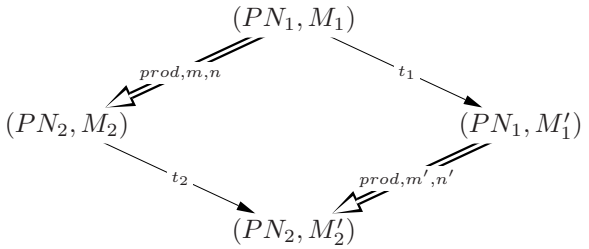
1. *$((prod, m, n), t_2)$ is sequentially independent iff $((prod^{-1}, n, m), t_2)$ is parallel independent.*

2. *$(t_1, (prod, m', n'))$ is sequentially independent iff $((prod, m', n'), t_1^{-1})$ is parallel independent.*

3. *$(t_2, (prod, m', n'))$ is coparallel independent iff $((prod^{-1}, n', m'), t_2^{-1})$ is parallel independent.*

Now we are able to extend Thm. 7 on parallel independence showing that resulting steps in the first diagram of Thm. 7 are sequentially and coparallel independent.

Theorem 8 *In Thm. 7, where we start with parallel independence of the upper steps in the following diagram with match m and comatch n, we have in addition the following sequential and coparallel independence in the following diagram:*



1. *The left pair of steps, short $((prod, m, n), t_2)$, is sequentially independent.*

2. *The right pair of steps, short $(t_1, (prod, m', n'))$, is sequentially independent.*

3. *The lower pair of steps, short $(t_2, (prod, m', n'))$, is coparallel independent.*

PROOF We use the proof of Thm. 7.

1. (a) t_2 is not created because it corresponds to $t_1 \in T_1$ which is not deleted.

   (b) We have $M_R(p) \leq M_2'(n(p))$ for all $p \in P_R$ by construction of the pushout $(2')$ with $M_2'' = M_2'$.

2. (a) t_1 is not deleted by the assumption of parallel independence.

   (b) $M_L(p) \leq M_1(m(p))$ for all $p \in P_L$ by pushout (1).

3. (a) t_2 is not created as shown in the proof of 1.(a).

   (b) $M_R(p) \leq M_2(n(p))$ for all $p \in P_R$ by pushout (2).

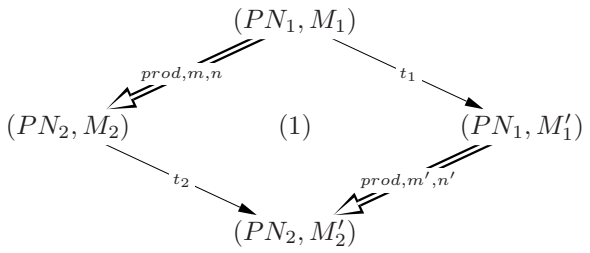
□

In Thm. 8 we have shown that parallel independence implies sequential and coparallel independence. Now we show vice versa that sequential (coparallel) independence implies parallel and coparallel (parallel and sequential) independence.

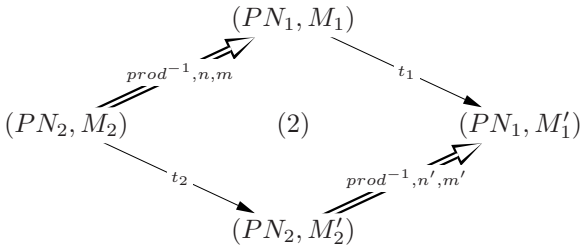
Theorem 9 *1. Given the left sequentially independent steps in diagram (1) then also the right steps exist such that the upper (right, lower) pair is parallel (sequentially, coparallel) independent.*

2. Given the right sequentially independent steps in diagram (1) then also the left steps exist such that the upper (left, lower) pair is parallel (sequentially, coparallel) independent.

3. Given the lower coparallel independent steps in diagram (1) then also the upper steps exist such that the upper (left,right) pair is parallel (sequentially, sequentially) independent.



PROOF 1. Using Rem. 6, left sequential independence in (1) corresponds to parallel independence in (2).



Applying Thms. 7 and 8 to the left pair in (2) we obtain the right pair such that the upper and lower pairs are sequentially and the right pair is coparallel independent. This implies by Rem. 6 that the upper (right, lower) pairs in (1) are parallel (sequentially, coparallel) independent.

The proofs of items 2. and 3. are analogous to the proof of item 1. □

7. General Framework of Net Transformations

In [23], we have introduced the paradigm "nets and rules as tokens" using a high-level model with suitable data type part. This model called algebraic higher-order (AHO) system (instead of high-level net and replacement system as in [23]) exploits some form of control not only on rule application but also on token firing. In general, an AHO system is defined by an algebraic high-level net with system places and rule places as for example shown in Fig. 10, where the marking is given by a suitable P/T system resp. rule on these places. For a detailed description of the data type part, i.e. the AHO-SYSTEM signature and corresponding algebra A, we refer to [23].

In the following, we review the behavior of AHO systems according to [23]. With the symbol $Var(t)$ we indicate the set of variables of a transition t, i.e. the set of all variables occurring in pre and post domains and in the firing condition of t. The marking M determines the distribution of P/T systems and rules in an AHO system, which are elements of a given higher-order algebra A.

Intuitively, P/T systems and rules can be moved along AHO system arcs and can be modified during the firing of transitions. The follower marking is computed by the evaluation of net inscriptions in a variable assignment $v : Var(t) \to A$. The transition t is enabled in a marking M if and only if (t, v) is consistent, that is if the evaluation of the firing condition is fulfilled. Then the follower marking after firing of transition t is defined by removing tokens corresponding to the net inscription in the pre domain of t and adding tokens corresponding to the net inscription in the post domain of t.

The transitions in the AHO system in Fig. 10 realize on the one hand firing steps and on the other hand transformation steps as indicated by the net inscriptions $fire(n, t)$ and $transform(r, m)$, respectively. The initial marking is the reconfigurable P/T system given in Ex. 1, i.e. the P/T system (PN_1, M_1) given in Fig. 1 is on the place p_1, while the marking of the place p_2 is given by the rules $prod_{photo}$ and $prod_{follow}$ given in Figs. 2 and 3, respectively. To compute the follower marking of the P/T system we use the transition *token game* of the AHO system. First, the variable n is assigned to the P/T-system (PN_1, M_1) and the variable t to the transition *Select Building*. Because this transition is enabled in the P/T system, the firing condition is fulfilled. Finally, due to the evaluation of the term $fire(n, t)$, we obtain the new P/T system (PN_1, M_1') (see Fig. 1).

For changing the structure of P/T systems, the transition *transformation* is provided in Fig. 10. Again, we have to give an assignment v for the variables of the transition, i.e. variables n, m and r, where $v(n) = (PN_1, M_1)$, $v(m)$ is a suitable match morphism and $v(r) = prod_{photo}$. The firing condition $cod\ m = n$ ensures that the codomain of the match morphism is equal to (PN_1, M_1), while the second condition $applicable(r, m)$ checks the gluing condition, i.e. if the rule $prod_{photo}$ is applicable with match m. Afterwards, the transformation step depicted in Fig. 2 is computed by the evaluation of the net inscription $transform(r, m)$ and the effect of firing the transition *transformation* is the removal of the P/T system (PN_1, M_1) from place p_1 and adding the P/T system (PN_2, M_2) to it.

The pair (or sequence) of firing and transformation steps discussed in the last sections is reflected by firing of the transitions one after the other in our AHO system. Thus, the results presented in this paper are most important for the analysis of AHO systems.

8. 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 transformations within reconfigurable P/T systems. Moreover, we have transferred the results of the Local Church-Rosser Theorem to the consecutive evolution of a P/T system by token firings and rule applications. We have presented necessary and sufficient conditions for (co-)parallel and sequential independence and have shown, that provided that these conditions are satisfied, firing and transformation steps can be performed in any order, yielding the same result. Also, we have correlated these conditions, i.e. that parallel independence implies sequential and coparallel independence and, vice versa, sequential (coparallel) independence implies parallel and coparallel (parallel and sequential) independence.

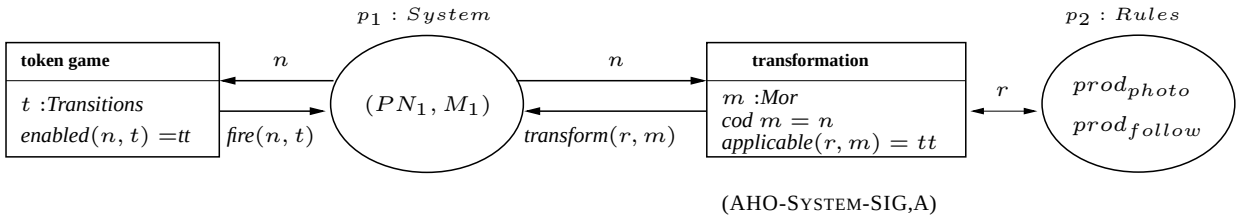


Figure 10. Algebraic higher-order system

Related Work

Transformations of nets can be considered in various ways. Transformations of Petri nets to a different Petri net class (e.g. in [8, 10, 37]), to another modeling technique or vice versa (e.g in [3, 6, 15, 25, 34, 14]) 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 [22, 16, 21, 12, 9]), or transformations are used to detect specific properties of nets (e.g. in [4, 5, 7, 29]).

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 [18]. The general approach can be restricted to transformations that preserve specific properties as safety or liveness (see [31, 33]). Closely related are those approaches that propose changing nets in specific ways in order to preserve specific semantic properties, as equivalent (I/O-) behavior (e.g in [2, 11]), invariants (e.g. in [13]) or liveness (e.g. in [20, 38]).

In [23], 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. This concept of has been used in [32] 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 [27], 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

We plan to develop a tool for our approach. For the application of net transformation rules, this tool will provide an export to AGG [1], 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 [24], 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. As future work, it would be important to verify the additional properties necessary for these results.

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 [30]. In [35], it is shown that different kinds of algebraic high-level (AHL) nets form weak adhesive HLR categories. It would be interesting to show that the corresponding AHL systems, i.e. AHL nets with markings, are also weak adhesive HLR categories. This could be verified for each kind of AHL system directly, but a more elegant solution would be to find a categorical construction integrating the marking.

References

[1] AGG Homepage. http://tfs.cs.tu-berlin.de/agg.

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

[3] F. Belli and J. Dreyer. Systems Modelling and Simulation by Means of Predicate/Transition Nets and Logic Programming. In *Proceedings of IEA/AIE'94*, pages 465–474, 1994.

[4] G. Berthelot. Checking Properties of Nets Using Transformation. In *Proceedings of ATPN'85*, volume 222 of *LNCS*, pages 19–40. Springer, 1985.

[5] G. Berthelot. Transformations and Decompositions of Nets. In *Advances in Petri Nets, Part I*, volume 254 of *LNCS*, pages 359–376. Springer, 1987.

[6] T. Bessey and M. Becker. Comparison of the Modeling Power of Fluid Stochastic Petri Nets (FSPN) and Hybrid Petri Nets (HPN). In *Proceedings of SMC'02*, volume 2, pages 354–358. IEEE, 2002.

[7] E. Best and T. Thielke. Orthogonal Transformations for Coloured Petri Nets. In *Proceedings of ATPN'97*, volume 1248 of *LNCS*, pages 447–466. Springer, 1997.

[8] J. Billington. Extensions to Coloured Petri Nets. In *Proceedings of PNPM'89*, pages 61–70. IEEE, 1989.

[9] P. Bonhomme, P. Aygalinc, G. Berthelot, and S. Calvez. Hierarchical Control of Time Petri Nets by Means of Transformations. In *Proceedings of SMC'02*, volume 4, pages 6–11. IEEE, 2002.

[10] J. Campos, B. Sánchez, and M. Silva. Throughput Lower Bounds for Markovian Petri Nets: Transformation Techniques. In *Proceedings of PNPM'91*, pages 322–331. IEEE, 1991.

[11] J. Carmona and J. Cortadella. Input/Output Compatibility of Reactive Systems. In *Proceedings of FMCAD'02*, volume 2517 of *LNCS*, pages 360–377. Springer, 2002.

[12] G. Chehaibar. Replacement of Open Interface Subnets and Stable State Transformation Equivalence. In *Proceedings of ATPN'91*, volume 674 of *LNCS*, pages 1–25. Springer, 1991.

[13] T. Cheung and Y. Lu. Five Classes of Invariant-Preserving Transformations on Colored Petri Nets. In *Proceedings of ATPN'99*, volume 1639 of *LNCS*, pages 384–403. Springer, 1999.

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

[15] J. de Lara and H. Vangheluwe. Computer Aided Multi-Paradigm Modelling to Process Petri-Nets and Statecharts. In *Proceedings of ICGT'02*, volume 2505 of *LNCS*, pages 239–253. Springer, 2002.

[16] J. Desel. On Abstraction of Nets. In *Proceedings of ATPN'90*, volume 524 of *LNCS*, pages 78–92. Springer, 1990.

[17] H. Ehrig. Introduction to the Algebraic Theory of Graph Grammars (A Survey). In V. Claus, H. Ehrig, and G. Rozenberg, editors, *Graph Grammars and their Application to Computer Science and Biology*, volume 73 of *LNCS*, pages 1–69. Springer, 1979.

[18] H. Ehrig, K. Ehrig, U. Prange, and G. Taentzer. *Fundamentals of Algebraic Graph Transformation*. EATCS Monographs. Springer, 2006.

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

[20] J. Esparza. Model Checking Using Net Unfoldings. *Science of Computer Programming*, 23(2-3):151–195, 1994.

[21] J. Esparza and M. Silva. On the Analysis and Synthesis of Free Choice Systems. In *Proceedings of ATPN'89*, volume 483 of *LNCS*, pages 243–286. Springer, 1989.

[22] S. Haddad. A Reduction Theory for Coloured Nets. In *Proceedings of ATPN'88*, volume 424 of *LNCS*, pages 209–235. Springer, 1988.

[23] K. Hoffmann, H. Ehrig, and T. Mossakowski. High-Level Nets with Nets and Rules as Tokens. In *Proceedings of ICATPN'05*, volume 3536 of *LNCS*, pages 268–288. Springer, 2005.

[24] E. Kindler and M. Weber. The Petri Net Kernel - An Infrastructure for Building Petri Net Tools. *Software Tools for Technology Transfer*, 3(4):486–497, 2001.

[25] O. Kluge. Modelling a Railway Crossing with Message Sequence Charts and Petri Nets. In *Petri Net Technology for Communication-Based Systems*, volume 2472 of *LNCS*, pages 197–218. Springer, 2003.

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

[27] M. Llorens and J. Oliver. Structural and Dynamic Changes in Concurrent Systems: Reconfigurable Petri Nets. *IEEE Transactions on Computers*, 53(9):1147–1158, 2004.

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

[29] T. Murata. Petri Nets: Properties, Analysis and Applications. In *Proceedings of the IEEE*, volume 77, pages 541–580. IEEE, 1989.

[30] J. Padberg, H. Ehrig, and L. Ribeiro. Algebraic High-Level Net Transformation Systems. *Mathematical Structures in Computer Science*, 5(2):217–256, 1995.

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

[32] J. Padberg, K. Hoffmann, H. Ehrig, T. Modica, E. Biermann, and C. Ermel. Maintaining Consistency in Layered Architectures of Mobile Ad-hoc Networks. In *Proceedings of FASE'07*, 2007. To appear.

[33] J. Padberg and M. Urbášek. Rule-Based Refinement of Petri Nets: A Survey. In *Petri Net Technology for Communication-Based Systems*, volume 2472 of *LNCS*, pages 161–196. Springer, 2003.

[34] F. Parisi-Presicce. A Formal Framework for Petri Net Class Transformations. In *Petri Net Technology for Communication-Based Systems*, volume 2472 of *LNCS*, pages 409–430. Springer, 2003.

[35] U. Prange. Algebraic High-Level Nets as Weak Adhesive HLR Categories. *Electronic Communications of the EASST*, 2, 2007. To appear.

[36] G. Rozenberg, editor. *Handbook of Graph Grammars and Computing by Graph Transformation, Vol 1: Foundations*. World Scientific, 1997.

[37] M. Urbášek. *Categorical Net Transformations for Petri Net Technology*. PhD thesis, TU Berlin, 2003.

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