Electronic Communications of the EASST
Volume 40 (2011)

Proceedings of the
4th International Workshop on Petri Nets and Graph Transformation
(PNGT 2010)

Formal Relationship between Petri Net and Graph Transformation Systems based on Functors between $\mathcal{M}$-adhesive Categories

Maria Maximova, Hartmut Ehrig and Claudia Ermel

18 pages

Guest Editors: Claudia Ermel, Kathrin Hoffmann
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Formal Relationship between Petri Net and Graph Transformation Systems based on Functors between $\mathcal{M}$-adhesive Categories

Maria Maximova, Hartmut Ehrig and Claudia Ermel

Institut für Softwaretechnik und Theoretische Informatik
Technische Universität Berlin, Germany
mascham@cs.tu-berlin.de, ehrig@cs.tu-berlin.de, claudia.ermel@tu-berlin.de

Abstract: Various kinds of graph transformations and Petri net transformation systems are examples of $\mathcal{M}$-adhesive transformation systems based on $\mathcal{M}$-adhesive categories, generalizing weak adhesive HLR categories. For typed attributed graph transformation systems, the tool environment AGG allows the modeling, the simulation and the analysis of graph transformations. A corresponding tool for Petri net transformation systems, the RON-Environment, has recently been developed which implements and simulates Petri net transformations based on corresponding graph transformations using AGG. Up to now, the correspondence between Petri net and graph transformations is handled on an informal level. The purpose of this paper is to establish a formal relationship between the corresponding $\mathcal{M}$-adhesive transformation systems, which allow the translation of Petri net transformations into graph transformations with equivalent behavior, and, vice versa, the creation of Petri net transformations from graph transformations. Since this is supposed to work for different kinds of Petri nets, we propose to define suitable functors, called $\mathcal{M}$-functors, between different $\mathcal{M}$-adhesive categories and to investigate properties allowing us the translation and creation of transformations of the corresponding $\mathcal{M}$-adhesive transformation systems.

Keywords: $\mathcal{M}$-adhesive transformation system, equivalence, graph transformation, Petri net transformation, $\mathcal{M}$-adhesive category

1 Introduction

Modeling the adaptation of a dynamic system to a changing environment gets more and more important. Application areas cover e.g. computer supported cooperative work, multi agent systems or mobile networks. One approach to combine formal modeling of dynamic systems and controlled model adaption are reconfigurable Petri nets. The main idea is the stepwise development of place/transition nets by applying net transformation rules [EHP$^+$08, PEHP08]. This approach increases the expressiveness of Petri nets and allows in addition to the well known token game a formal description and analysis of structural changes.

Rule-based Petri net transformation is related to graph transformation [EEPT06]. For typed attributed graph transformation systems, the well-established tool AGG [AGG09] allows the modeling, the simulation and the analysis of graph transformations. Recently, a tool for reconfigurable Petri nets, called RON-Tool [TFS07, BEHM07] (*Reconfigurable Object Nets*), executes and analyzes Petri net transformations based on corresponding graph transformations using AGG. As a matter of fact, the correspondence between Petri net and graph transformations is handled on an informal level up to now. Since both graph and net transformation systems are formally defined, the aim of this paper is to propose formal criteria ensuring a semantical correspondence of reconfigurable Petri nets and their corresponding representations as graph transformation systems.

An $\mathscr{M}$-adhesive transformation system is a general categorical transformation framework based on $\mathscr{M}$-adhesive categories, which rely on a class $\mathscr{M}$ of monomorphisms, generalizing weak adhesive HLR categories. The double-pushout approach, based on categorical constructions, is a suitable description of transformations leading to results like the Local Church-Rosser, Parallelism, Concurrency, Embedding, Extension, and Local Confluence Theorems [EEPT06].

A set of rules over an $\mathscr{M}$-adhesive category according to the double-pushout approach constitutes an $\mathscr{M}$-adhesive transformation system [EGH10].

Aiming for a more general approach to ensure a semantical correspondence of different transformation systems, we establish a formal relationship between two corresponding $\mathscr{M}$-adhesive transformation systems. This correspondence allows us especially the translation of Petri net transformations into graph transformations and, vice versa, the creation of Petri net transformations from graph transformations in order to analyze the behavior of Petri net transformation systems by analyzing their translation in terms of typed attributed graph transformation systems using the tool AGG [AGG09]. We propose to define suitable functors, called $\mathscr{M}$-functors, between different $\mathscr{M}$-adhesive categories and to investigate properties, which allow us the translation and creation of transformations of the corresponding $\mathscr{M}$-adhesive transformation systems.

The paper is structured as follows: Section 3 introduces the formal notions $\mathscr{M}$-adhesive transformation systems and $\mathscr{M}$-functors. The first main result given in Section 4 states that an $\mathscr{M}$-functor translates rules in a way that applicablility and transformation results are translated as well. Vice versa, the second main result states that an $\mathscr{M}$-functor also creates applicability of rules in the other direction. Section 5 applies these new main result to the translation and creation of Petri net transformations by constructing and analyzing an $\mathscr{M}$-functor from the category of place/transition nets to the category of typed attributed graphs with corresponding type graph[1]. In Section 6, we conclude and propose interesting future research directions.

2 Related Work

In [MM90], Meseguer and Montanari represented Petri nets as graphs equipped with operations for composition of transitions. They introduced categories for Petri nets with and without initial markings and functors expressing duality and invariants. Their constructions provide a formal basis for expressing concurrency in terms of algebraic structures over graphs and categories. Based on categorical Petri nets, in [DS02], Petri nets are related to automata with concurrency relations by establishing a correspondence as coreflection between the associated categories. A first approach to relate Petri nets and graph transformation systems has been proposed by Kreowski in [Kre81], where Petri net firing behavior is expressed by graph transformation rules. In our approach, we want to consider Petri net transformations in addition. Moreover, we aim for a

[1] For the results in Section 5, we give only proof ideas. More detailed proofs are given in [MEE11].

more general approach that establishes a semantical correspondence not only between Petri net and graph transformation systems but between any kind of formally defined rule-based transformation systems that can be generalized as $\mathcal{M}$-adhesive transformation systems.

In order to transform not only graphs, but also high-level structures as Petri nets and algebraic specifications, high-level replacement (HLR) categories were established in [EHKP91a, EHKP91b], which require a list of so-called HLR properties to hold. They were based on a morphism class $\mathcal{M}$ used for the rule morphisms. This framework allowed a rich theory of transformations for all HLR categories, but the HLR properties were difficult and lengthy to verify for each category. Combining adhesive categories [LS04] and HLR categories lead to (weak) adhesive HLR categories in [EHPP06] and to $\mathcal{M}$-adhesive categories in [EGH10], where a subclass $\mathcal{M}$ of monomorphisms is considered and only pushouts over $\mathcal{M}$-morphisms have to fulfill the *van Kampen property* (a certain compatibility of pushouts and pullbacks). Not only many kinds of graphs, but also different kinds of place/transition nets and algebraic high-level nets are $\mathcal{M}$-adhesive and also weak adhesive HLR categories which allows the application of the theory to all these kinds of structures [EEPT06, PE07, MGE+09]. In fact, all results in [EEPT06] for weak adhesive HLR categories are also valid for $\mathcal{M}$-adhesive categories [EGH10].

3 $\mathcal{M}$-Adhesive Categories, Transformation Systems and $\mathcal{M}$-Functors

An $\mathcal{M}$-adhesive category [EGH10], consists of a category $\mathbf{C}$ together with a class $\mathcal{M}$ of monomorphisms as defined in Definition 1 below. The concept of $\mathcal{M}$-adhesive categories generalises that of weak adhesive, adhesive HLR and adhesive categories [LS04]. The category of typed attributes graphs and several categories of Petri nets are weak adhesive HLR (see [EEPT06]) and hence also $\mathcal{M}$-adhesive.

Definition 1 ($\mathcal{M}$-Adhesive Category)
An $\mathcal{M}$-adhesive category $(\mathbf{C}, \mathcal{M})$ is a category $\mathbf{C}$ together with a class $\mathcal{M}$ of monomorphisms satisfying

- $\mathbf{C}$ has pushouts (POs) and pullbacks (PBs) along $\mathcal{M}$-morphisms,
- $\mathcal{M}$ is closed under composition, decomposition, POs and PBs,
- POs along $\mathcal{M}$-morphisms are $\mathcal{M}$-VK-squares, i.e. the VK-property holds for all commutative cubes, where the given PO with $m \in \mathcal{M}$ is in the bottom, the back faces are PBs and all vertical morphisms a, b, c and d are in $\mathcal{M}$. The VK-property means that the top face is a PO iff the front faces are PBs.

$$\begin{tikzcd}
& A' \arrow[rr, "m'"] \arrow[dl, "f'"'] \arrow[dd, dashed, "a"] & & B' \arrow[dd, "b"] \arrow[dl, "g'"] \\
C' \arrow[rr, "n'"] \arrow[dd, "c"'] & & D' \arrow[dd, "d"'] & \\
& A \arrow[rr, dashed, "m"] \arrow[dl, dashed, "f"] & & B \arrow[dl, "g"] \\
C \arrow[rr, "n"'] & & D &
\end{tikzcd}$$

Definition 2 ($\mathcal{M}$-Adhesive Transformation System and Independence)
Given an $\mathcal{M}$-adhesive category $(\mathbf{C}, \mathcal{M})$.

- An $\mathcal{M}$-adhesive transformation system $AS = (\mathbf{C}, \mathcal{M}, P)$ has in addition a set P of productions of the form $\rho = (L \xleftarrow{l} K \xrightarrow{r} R)$ with $l, r \in \mathcal{M}$.

A direct transformation $G \overset{\rho,m}{\Longrightarrow} H$ via production ρ and match m consists of two POs (1) and (2) as shown in the diagram to the right, where $n : R \to H$ is called comatch of m. A production ρ is applicable via m to G, if we have a PO complement D in (1), such that (1) becomes a PO.

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

- Two (direct) transformations $G \overset{\rho_1,m_1}{\Longrightarrow} H_1$ and $G \overset{\rho_2,m_2}{\Longrightarrow} H_2$ are called parallel independent, if there are morphisms $d_{12} : L_1 \to D_2$, $d_{21} : L_2 \to D_1$ such that $l^*{}_1 \circ d_{21} = m_2$ and $l^*{}_2 \circ d_{12} = m_1$. Dually $G \overset{\rho_1,m_1}{\Longrightarrow} H_1$ and $H_1 \overset{\rho_2,m_2}{\Longrightarrow} H_2$ are sequentially independent if $H_1 \overset{\rho_1{}^{-1},n_1}{\Longrightarrow} G$ and $H_1 \overset{\rho_2,m_2}{\Longrightarrow} H_2$ are parallel independent, where $\rho_1{}^{-1} = (R_1 \xleftarrow{r_1} K_1 \xrightarrow{l_1} L_1)$ and n_1 is the comatch of m_1.

$$\begin{tikzcd}
R_1 \arrow[d] & K_1 \arrow[l] \arrow[r, "l_1"] \arrow[d] & L_1 \arrow[dr, "m_1"] \arrow[rrrd, dashed, "d_{12}"] & & L_2 \arrow[dl, "m_2"] \arrow[llld, dashed, "d_{21}"'] & K_2 \arrow[l, "l_2"'] \arrow[r] \arrow[d] & R_2 \arrow[d] \\
H_1 & D_1 \arrow[l] \arrow[rr, "l^*{}_1"'] & & G & & D_2 \arrow[ll, "l^*{}_2"] \arrow[r] & H_2
\end{tikzcd}$$

In order to study translation and creation of transformations between different $\mathscr{M}$-adhesive transformation systems we introduce the notion of an $\mathscr{M}$-functor. An $\mathscr{M}$-functor establishes a semantical correspondence between different $\mathscr{M}$-adhesive transformation systems.

Definition 3 ($\mathscr{M}$-Functor)
A functor $\mathscr{F} : (\mathbf{C}_1, \mathscr{M}_1) \to (\mathbf{C}_2, \mathscr{M}_2)$ between $\mathscr{M}$-adhesive categories is called $\mathscr{M}$-functor, if $\mathscr{F}(\mathscr{M}_1) \subseteq \mathscr{M}_2$ and $\mathscr{F}$ preserves pushouts along $\mathscr{M}$-morphisms.

On purpose we don't require that an $\mathscr{M}$-functor preserves pullbacks along $\mathscr{M}$-morphisms, VK-squares, or other properties, but later additional properties of $\mathscr{F}$ will be required in order to achieve specific results.

Remark 1
If we want to consider only (direct) transformations with injective matches, as in the case of Petri net transformations in the next section, then it is sufficient to define the functor $\mathscr{F}$ on injective morphisms only. Moreover, this restriction is necessary, if $\mathscr{F}$ is not well-defined for non-injective morphisms.

For this case we need to define a special kind of an $\mathscr{M}$-functor: a *restricted $\mathscr{M}$-functor*.

Definition 4 (Restricted $\mathscr{M}$-Functor)
A functor $\mathscr{F} : \mathbf{C}_1|_{\mathscr{M}_1} \to \mathbf{C}_2|_{\mathscr{M}_2}$ between $\mathscr{M}$-adhesive categories $(\mathbf{C}_1, \mathscr{M}_1)$ and $(\mathbf{C}_2, \mathscr{M}_2)$ with $\mathbf{C}_i|_{\mathscr{M}_i}$ the restriction of $\mathbf{C}_i$ to $\mathscr{M}_i$-morphisms for $i = 1,2$ is called a *restricted $\mathscr{M}$-functor*, if $\mathscr{F}(\mathscr{M}_1) \subseteq \mathscr{M}_2$ and $\mathscr{F}$ translates POs along $\mathscr{M}_1$-morphisms in $(\mathbf{C}_1, \mathscr{M}_1)$ into POs along $\mathscr{M}_2$-morphisms in $(\mathbf{C}_2, \mathscr{M}_2)$.

4 Translation and Creation of Transformations

To obtain a semantical correspondence between any two transformation systems we need to ensure that the respective transformation systems together with their relevant properties are translated and reflected properly.

Given an $\mathcal{M}$-adhesive transformation system $AS_1 = (\mathbf{C}_1, \mathcal{M}_1, P_1)$ with an $\mathcal{M}$-adhesive category $(\mathbf{C}_1, \mathcal{M}_1)$ and productions P_1. We want to translate transformations from AS_1 to $AS_2 = (\mathbf{C}_2, \mathcal{M}_2, P_2)$ with $\mathcal{M}$-adhesive category $(\mathbf{C}_2, \mathcal{M}_2)$ and suitable productions P_2. This can be done using an $\mathcal{M}$-functor $\mathcal{F} : (\mathbf{C}_1, \mathcal{M}_1) \to (\mathbf{C}_2, \mathcal{M}_2)$ for $P_2 = \mathcal{F}(P_1)$.

Theorem 1 (Translation of Transformations)
An $\mathcal{M}$-functor $\mathcal{F} : (\mathbf{C}_1, \mathcal{M}_1) \to (\mathbf{C}_2, \mathcal{M}_2)$ translates applicability of productions, construction of (direct) transformations , as well as parallel and sequential independence of transformations.

Proof.
$AS_2 = (\mathbf{C}_2, \mathcal{M}_2, \mathcal{F}(P_1))$ is a well-defined $\mathcal{M}$-adhesive transformation system, because $\mathcal{F}$ translates $\mathcal{M}_1$-morphisms into $\mathcal{M}_2$-morphisms for the productions and each direct transformation $G \overset{\rho,m}{\Longrightarrow} H$ in AS_1 given by pushouts (1) and (2) leads to a direct transformation $\mathcal{F}(G) \overset{\mathcal{F}(\rho),\mathcal{F}(m)}{\Longrightarrow} \mathcal{F}(H)$ in AS_2 given by pushouts (3) and (4), because $\mathcal{F}$ preserves pushouts along $\mathcal{M}$-morphisms.

$$\begin{tikzcd}
L \arrow[d, "m"'] \arrow[dr, phantom, "(1)"] & K \arrow[l, "l"'] \arrow[r, "r"] \arrow[d] \arrow[dr, phantom, "(2)"] & R \arrow[d] & \Rightarrow & \mathcal{F}(L) \arrow[d, "\mathcal{F}(m)"'] \arrow[dr, phantom, "(3)"] & \mathcal{F}(K) \arrow[l, "\mathcal{F}(l)"'] \arrow[r, "\mathcal{F}(r)"] \arrow[d] \arrow[dr, phantom, "(4)"] & \mathcal{F}(R) \arrow[d] \\
G & D \arrow[l] \arrow[r] & H & & \mathcal{F}(G) & \mathcal{F}(D) \arrow[l] \arrow[r] & \mathcal{F}(H)
\end{tikzcd}$$

Moreover, the functor property of $\mathcal{F}$ implies that $\mathcal{F}$ translates parallel and sequential independence of transformations. $\square$

As shown above, we need for translation of transformations from AS_1 to AS_2 only the basic properties of an $\mathcal{M}$-functor. This is no longer true for creation of transformations in AS_1 from transformations in AS_2 with $P_2 = \mathcal{F}(P_1)$ as above.

Definition 5 (Creation of Applicability and Direct Transformations)

1. An $\mathcal{M}$-functor $\mathcal{F} : (\mathbf{C}_1, \mathcal{M}_1) \to (\mathbf{C}_2, \mathcal{M}_2)$ creates applicability of a production $\rho = (L \overset{l}{\leftarrow} K \overset{r}{\to} R)$ to object G, if applicability of $\mathcal{F}(\rho)$ to $\mathcal{F}(G)$ with match $m' : \mathcal{F}(L) \to \mathcal{F}(G)$ implies applicability of ρ to G with some match $m : L \to G$ and $\mathcal{F}(m) = m'$.

2. $\mathcal{F}$ creates direct transformations, if for each direct transformation $\mathcal{F}(G) \overset{\mathcal{F}(\rho),m'}{\Longrightarrow} H'$ in AS_2 there is a direct transformation $G \overset{\rho,m}{\Longrightarrow} H$ in AS_1 with $\mathcal{F}(m) = m'$ and $\mathcal{F}(H) \cong H'$ leading to $\mathcal{F}(G) \overset{\mathcal{F}(\rho),\mathcal{F}(m)}{\Longrightarrow} \mathcal{F}(H)$ in AS_2:

$$\begin{tikzcd}
\mathscr{F}(L) \arrow[d, "m'"'] \arrow[dr, phantom, "(1)"] & \mathscr{F}(K) \arrow[l, "\mathscr{F}(l)"'] \arrow[r, "\mathscr{F}(r)"] \arrow[d, dashed] \arrow[dr, phantom, "(2)"] & \mathscr{F}(R) \arrow[d, dashed] & & L \arrow[d, "m"'] \arrow[dr, phantom, "(3)"] & K \arrow[l, "l"'] \arrow[r, "r"] \arrow[d, dashed] \arrow[dr, phantom, "(4)"] & R \arrow[d, dashed] \\
\mathscr{F}(G) & D' \arrow[l, dashed] \arrow[r, dashed] & H' & \Longrightarrow & G & D \arrow[l, dashed] \arrow[r, dashed] & H
\end{tikzcd}$$

3. $\mathscr{F}$ creates parallel (and similarly sequential) independence, if parallel independence of $\mathscr{F}(H_1) \overset{\mathscr{F}(\rho_1),\mathscr{F}(m_1)}{\Longleftarrow} \mathscr{F}(G) \overset{\mathscr{F}(\rho_2),\mathscr{F}(m_2)}{\Longrightarrow} \mathscr{F}(H_2)$ in AS_2 implies parallel independence of $H_1 \overset{\rho_1,m_1}{\Longleftarrow} G \overset{\rho_2,m_2}{\Longrightarrow} H_2$ in AS_1.

Remark 2
If $\mathscr{F}$ creates parallel (sequential) independence, then $\mathscr{F}$ characterises parallel (sequential) independence, i.e., parallel (sequential) independence in AS_1 is equivalent to parallel (sequential) independence in AS_2, because $\mathscr{F}$ already preserves parallel (sequential) independence by Theorem 1.

In the following we formulate the properties for an $\mathscr{M}$-functor $\mathscr{F}$, such that we have creation of applicability, direct transformations and parallel (sequential) independence. But first we review the notion of initial pushouts motivated by Remark 3 below.

Definition 6 (Initial Pushout)
Given a morphism $f : G \to G'$ in an $\mathscr{M}$-adhesive category $(\mathbf{C}, \mathscr{M})$. (1) is an *initial pushout (IPO)* over f with boundary B, context C and $b, c \in \mathscr{M}$, if
(1) is PO $\wedge$ $\forall$ POs (2) over f(defined by the outer diagram) with $h, h' \in \mathscr{M} \Longrightarrow$
$\exists! b^* : B \to B', c^* : C \to C'.\ h \circ b^* = b \ \wedge \ h' \circ c^* = c \wedge (3)$ is a PO.

$$\begin{tikzcd}
& B' \arrow[d, "k'"'] \arrow[rr, bend left=50, tail, "h", "="'] \arrow[dr, phantom, "(3)"] & B \arrow[l, dashed, tail, "b^*"'] \arrow[r, tail, "b"] \arrow[d, "k"] \arrow[dr, phantom, "(1)"] & G \arrow[d, "f"] \\
(2) & C' \arrow[rr, bend right=50, tail, "h'"', "="] & C \arrow[l, dashed, tail, "c^*"] \arrow[r, tail, "c"] & G'
\end{tikzcd}$$

Remark 3
*For each match $m : L \to G$ with initial pushout (4) and $b \in \mathscr{M}_1$, a production $\rho = (L \xleftarrow{l} K \xrightarrow{r} R)$ is applicable with match $m : L \to G$, iff the following "gluing condition" is satisfied:
there is $b' : B \to K$ in $\mathscr{M}_1$ with $l \circ b' = b$. In this case the pushout complement D in (5) can be constructed as pushout of $b' \in \mathscr{M}_1$ and a leading to $h : C \to D, k : K \to D$ and an induced morphism $d : D \to G$, s.t., (5) is pushout and (7) commutes (see [EEPT06]).*

$$\begin{tikzcd}
B \arrow[r, "b"] \arrow[d, "a"'] \arrow[rr, bend left=50, dashed, "b'", "(6)"'] \arrow[dr, phantom, "(4)"] & L \arrow[d, "m"] \arrow[dr, phantom, "(5)"] & K \arrow[l, "l"'] \arrow[d, dashed, "k"] \\
C \arrow[r] \arrow[rr, bend right=50, dashed, "h"', "(7)"] & G & D \arrow[l, dashed, "d"]
\end{tikzcd}$$

Definition 7 (Properties of $\mathscr{M}$-Functors)

1. An $\mathscr{M}$-functor $\mathscr{F} : (\mathbf{C}_1, \mathscr{M}_1) \to (\mathbf{C}_2, \mathscr{M}_2)$ creates morphisms, if for all $m' : \mathscr{F}(L) \to \mathscr{F}(G)$ in $(\mathbf{C}_2, \mathscr{M}_2)$ there is exactly one morphism $m : L \to G$ with $\mathscr{F}(m) = m'$.

2. $\mathscr{F}$ preserves initial pushouts, if for each initial pushout (IPO) (1) over $m : L \to G$, also (2) is initial pushout over $\mathscr{F}(m) : \mathscr{F}(L) \to \mathscr{F}(G)$.

$$\begin{tikzcd}
 & B \arrow[r, "b"] \arrow[dd] \arrow[ddr, phantom, "(1)"] & L \arrow[dd, "m"] & & \mathscr{F}(B) \arrow[r, "\mathscr{F}(b)"] \arrow[dd] \arrow[ddr, phantom, "(2)"] & \mathscr{F}(L) \arrow[dd, "\mathscr{F}(m)"] & \\
\text{IPO in } (\mathbf{C}_1, \mathscr{M}_1) & & & \Longrightarrow & & & \text{IPO in } (\mathbf{C}_2, \mathscr{M}_2) \\
 & C \arrow[r] & G & & \mathscr{F}(C) \arrow[r] & \mathscr{F}(G) &
\end{tikzcd}$$

This leads to the following theorem on creation of transformations by $\mathscr{M}$-functors:

Theorem 2 (Creation of Transformations)
Given an $\mathscr{M}$-functor $\mathscr{F} : (\mathbf{C}_1, \mathscr{M}_1) \to (\mathbf{C}_2, \mathscr{M}_2)$ with initial pushouts in $(\mathbf{C}_1, \mathscr{M}_1)$, which creates morphisms and preserves initial pushouts, then $\mathscr{F}$ creates applicability of productions, direct transformations, as well as parallel and sequential independence of transformations.

Proof.
1. *$\mathscr{F}$ creates applicability of productions*
Given $\rho = (L \xleftarrow{l} K \xrightarrow{r} R)$ and match $m' : \mathscr{F}(L) \to \mathscr{F}(G)$, s.t., $\mathscr{F}(\rho)$ is applicable to m'. Since $\mathscr{F}$ creates morphisms we have a unique $m : L \to G$ with $\mathscr{F}(m) = m'$. Let (1) be an initial pushout over m in the diagram below. By assumption on $\mathscr{F}$, (2) is initial pushout over $\mathscr{F}(m)$ and $(4), (5)$ are POs. This means, that $\mathscr{F}(\rho)$ is applicable to $m' = \mathscr{F}(m)$. According to Remark 3, this implies the existence of $b'' : \mathscr{F}(B) \to \mathscr{F}(K)$ in $\mathscr{M}_2$ with $\mathscr{F}(l) \circ b'' = \mathscr{F}(b)$.

$$\begin{tikzcd}
B \arrow[r, "b"] \arrow[rr, dashed, bend left, "b'"] \arrow[d, "a"] \arrow[dr, phantom, "(1)"] & L \arrow[d, "m"] \arrow[dr, phantom, "(3)"] & K \arrow[l, "l"'] \arrow[r, "r"] \arrow[d, dashed] \arrow[dr, phantom, "(6)"] & R \arrow[d, dashed] & \Longleftarrow & \mathscr{F}(B) \arrow[r, "\mathscr{F}(b)"] \arrow[rr, dashed, bend left, "b''"] \arrow[d, "\mathscr{F}(a)"'] \arrow[dr, phantom, "(2)"] & \mathscr{F}(L) \arrow[d, "m'"] \arrow[dr, phantom, "(4)"] & \mathscr{F}(K) \arrow[l, "\mathscr{F}(l)"'] \arrow[r, "\mathscr{F}(r)"] \arrow[d] \arrow[dr, phantom, "(5)"] & \mathscr{F}(R) \arrow[d] \\
C \arrow[r] & G & D \arrow[l, dashed] \arrow[r, dashed] & H & & \mathscr{F}(C) \arrow[r] & \mathscr{F}(G) & D' \arrow[l] \arrow[r] & H'
\end{tikzcd}$$

Since $\mathscr{F}$ creates morphisms there is a unique morphism $b' : B \to K$ with $\mathscr{F}(b') = b''$. Moreover, uniqueness of creation of morphisms implies $l \circ b' = b$ and hence $b' \in \mathscr{M}_1$ by decomposition property of $\mathscr{M}_1$. Hence the gluing condition is satisfied and we have applicability of ρ to G with match $m : L \to G$ and $\mathscr{F}(m) = m'$ with pushout complement D in (3).

2. *$\mathscr{F}$ creates direct transformations*
Given the direct transformation $\mathscr{F}(G) \overset{\mathscr{F}(\rho),m'}{\Longrightarrow} H'$ in AS_2 by pushouts (4) and (5) in $(\mathbf{C}_2, \mathscr{M}_2)$. We have already constructed pushout (3) in $(\mathbf{C}_1, \mathscr{M}_1)$ and can construct pushout (6) along $r \in \mathscr{M}_1$ leading to a direct transformation $G \overset{\rho,m}{\Longrightarrow} H$. Since $\mathscr{F}$ preserves pushouts along $\mathscr{M}$-morphisms and pushout complements in $(\mathbf{C}_2, \mathscr{M}_2)$ and they are unique up to isomorphism. We have $\mathscr{F}(D) \cong D'$, $\mathscr{F}(H) \cong H'$ and hence also $\mathscr{F}(G) \overset{\mathscr{F}(\rho),\mathscr{F}(m)}{\Longrightarrow} \mathscr{F}(H)$ in AS_2.

3. *$\mathscr{F}$ creates parallel (sequential) independence*
By parallel independence of $\mathscr{F}(H_1) \overset{\mathscr{F}(\rho_1),\mathscr{F}(m_1)}{\Longleftarrow} \mathscr{F}(G) \overset{\mathscr{F}(\rho_2),\mathscr{F}(m_2)}{\Longrightarrow} \mathscr{F}(H_2)$ in AS_2 we have mor-

phisms $d'_{12} : \mathscr{F}(L_1) \to \mathscr{F}(D_2)$ with $\mathscr{F}(l^*{}_2) \circ d'_{12} = \mathscr{F}(m_1)$ and $d'_{21} : \mathscr{F}(L_2) \to \mathscr{F}(D_1)$ with $\mathscr{F}(l^*{}_1) \circ d'_{21} = \mathscr{F}(m_2)$ leading to corresponding morphisms $d_{12} : L_1 \to D_2$ and $d_{21} : L_2 \to D_1$ with $l^*{}_2 \circ d_{12} = m_1$ and $l^*{}_1 \circ d_{21} = m_2$, because $\mathscr{F}$ creates morphisms uniquely and preserves composition. □

Remark 4 *For the case described in the Remark 1 we have to show for Theorem 1 that $\mathscr{F}$ translates pushouts of $\mathscr{M}_1$-morphisms in $(\mathbf{C}_1, \mathscr{M}_1)$ into pushouts of $\mathscr{M}_2$-morphisms in $(\mathbf{C}_2, \mathscr{M}_2)$. For Theorem 2 we need in addition, that $\mathscr{F}$ creates $\mathscr{M}$-morphisms, i.e., for each $(m' : \mathscr{F}(L) \to \mathscr{F}(G)) \in \mathscr{M}_2$ there is exactly one morphism $(m : L \to G) \in \mathscr{M}_1$ with $\mathscr{F}(m) = m'$ and $\mathscr{F}$ preserves initial pushouts over $\mathscr{M}_1$-morphisms. Note, that we cannot replace the $\mathscr{M}$-adhesive categories $(\mathbf{C}_i, \mathscr{M}_i)$ for $i = 1,2$ by $(\mathbf{C}_i|_{\mathscr{M}_i}, \mathscr{M}_i)$, because $(\mathbf{C}_i|_{\mathscr{M}_i}, \mathscr{M}_i)$ are in general not $\mathscr{M}$-adhesive.*

5 Translation and Creation of Petri Net Transformations

According to our overall aim in Section 1 we want to construct a functor from Petri nets to typed attributed graphs and show how to apply the main results of Theorem 1 and Theorem 2 in order to translate and create Petri net transformations using graph transformations. For this purpose we review on one hand the $\mathscr{M}$-adhesive categories $(\mathbf{PTINet}, \mathscr{M}_1)$ of Petri nets with individual tokens and class $\mathscr{M}_1$ of all injective morphisms, which is defined and shown to be $\mathscr{M}$-adhesive in [MGE+09]. On the other hand we review typed attributed graphs $(\mathbf{AGraphs_{ATG}}, \mathscr{M}_2)$, which are shown to be $\mathscr{M}$-adhesive in [EEPT06] and we define a suitable attributed Petri net type graph $ATG = PNTG$. Moreover we construct a functor $\mathscr{F}$ between both categories, which, however, is only defined on injective morphisms $\mathscr{M}_1$.

Note, that we do not use Petri nets with “classical initial markings”, known as Petri net systems [Rei85], because the corresponding $\mathscr{M}$-adhesive category requires a class $\mathscr{M}$ leading to Petri net rules which are marking preserving. Marking preserving rules are not adequate to model firing steps as direct transformations since tokens must not be created or deleted. Other choices for $(\mathbf{C}_1, \mathscr{M}_1)$ would be Petri nets without initial marking or algebraic high-level nets (see [EEPT06, Rei85, MGE+09]).

In fact, we can construct a functor $\mathscr{F} : \mathbf{PTINet}|_{\mathscr{M}_1} \to \mathbf{AGraphs_{PNTG}}|_{\mathscr{M}_2}$ between the categories restricted to $\mathscr{M}$-morphisms, but not an $\mathscr{M}$-functor $\mathscr{F} : (\mathbf{PTINet}, \mathscr{M}_1) \to (\mathbf{AGraphs_{PNTG}}, \mathscr{M}_2)$, because $\mathscr{F}$ is not well-defined on non-injective morphisms (see counterexample in Figure 1 below, where $\mathscr{F}(f)$ does not preserve attributes in and w_{pre}). This means, we proceed as discussed in Remark 4, which allows the application of Theorem 1 and Theorem 2 in order to obtain translation and creation of Petri net transformations with injective morphisms. For application of Theorem 1 we need steps 1.-5., and for Theorem 2 in addition steps 6. and 7.

1. Definition of Petri nets with individual Tokens: **PTINet**.
2. Definition of typed attributed graphs over Petri net type graph $PNTG$: $\mathbf{AGraphs_{PNTG}}$.
3. Translation of PTI nets into $PNTG$-typed attributed graphs (definition of functor $\mathscr{F}$ on objects).
4. Translation of restricted **PTINet**-morphisms into restricted $\mathbf{AGraphs_{PNTG}}$-morphisms (definition of functor $\mathscr{F} : \mathbf{PTINet}|_{\mathscr{M}_1} \to \mathbf{AGraphs_{PNTG}}|_{\mathscr{M}_2}$ on morphisms).

5. $\mathscr{F}$ translates pushouts of $\mathscr{M}_1$-morphisms in $(\mathbf{PTINet}, \mathscr{M}_1)$ into pushouts of $\mathscr{M}_2$-morphisms in $(\mathbf{AGraphs_{PNTG}}, \mathscr{M}_2)$.
6. $\mathscr{F}$ creates $\mathscr{M}_1$-morphisms.
7. $\mathscr{F}$ preserves initial pushouts over $\mathscr{M}_1$-morphisms.

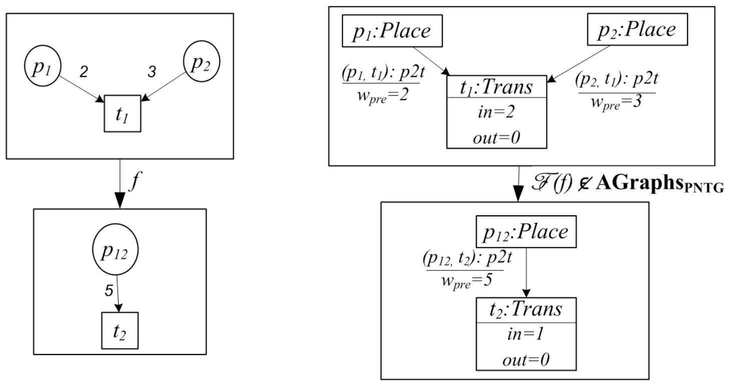


Figure 1: Counterexample for general (non-injective) morphisms

5.1 Petri Nets with Individual Tokens: PTINet

For classical place / transition (P/T) nets N we adopt the approach of Meseguer and Montanari [MM90] using free commutative monoids $P^{\oplus}$ over P, where $N = (P, T, pre, post)$ with places P, transitions T, functions $pre, post : T \to P^{\oplus}$ and markings $M \in P^{\oplus}$.
Petri nets $NI = (P, T, pre, post, I, m)$ with individual tokens are place/transition nets $N = (P, T, pre, post)$ together with a set I of individual tokens and a marking function $m : I \to P$ assigning a place $m(x) \in P$ to each $x \in I$. Therefore two (or more) different individual tokens $x, y \in I$ may be on the same place, i.e. $m(x) = m(y)$, while in the standard "collective token approach" the marking $M \in P^{\oplus}$ tells us only how many tokens we have on each place, but we are not able to distinguish between two tokens on the same place.

A formal definition of a Petri net with individual tokens is as follows ([MGE+09]).

Definition 8 (Petri Net with Individual Tokens)
A Petri net with individual tokens $NI = (P, T, pre, post, I, m)$ is given by a classical P/T net $N = (P, T, pre : T \to P^{\oplus}, post : T \to P^{\oplus})$, where $P^{\oplus}$ is the free commutative monoid over P, a (possibly infinite) set of individual tokens I, and the marking function $m : I \to P$, assigning to each individual token $x \in I$ the corresponding place $m(x) \in P$.

PTINet-morphisms now define not only a mapping between two P/T nets but also between their individual tokens:

Definition 9 (**PTINet**-Morphism)
A **PTINet**-morphism $f : NI_1 \to NI_2$ is given by a triple of functions $f = (f_P : P_1 \to P_2, f_T : T_1 \to T_2, f_I : I_1 \to I_2)$, such that the following diagrams commute with *pre* and *post* respectively.

$$\begin{tikzcd}
T_1 \arrow[r, shift left, "pre_1"] \arrow[r, shift right, "post_1"'] \arrow[d, "f_T"'] \arrow[dr, phantom, "="] & P_1^{\oplus} \arrow[d, "f_P^{\oplus}"] \\
T_2 \arrow[r, shift left, "pre_2"] \arrow[r, shift right, "post_2"'] & P_2^{\oplus}
\end{tikzcd}
\qquad
\begin{tikzcd}
I_1 \arrow[r, "m_1"] \arrow[d, "f_I"'] \arrow[dr, phantom, "="] & P_1 \arrow[d, "f_P"] \\
I_2 \arrow[r, "m_2"] & P_2
\end{tikzcd}$$

It is also shown in [MGE+09], that $(\mathbf{PTINet}, \mathcal{M}_1)$ with the class $\mathcal{M}_1$ of all injective morphisms is an $\mathcal{M}$-adhesive category, where pushouts and pullbacks are constructed componentwise (see Figure 2, where (1) is an example for a pushout in $(\mathbf{PTINet}, \mathcal{M}_1)$, with individual tokens colored in black).

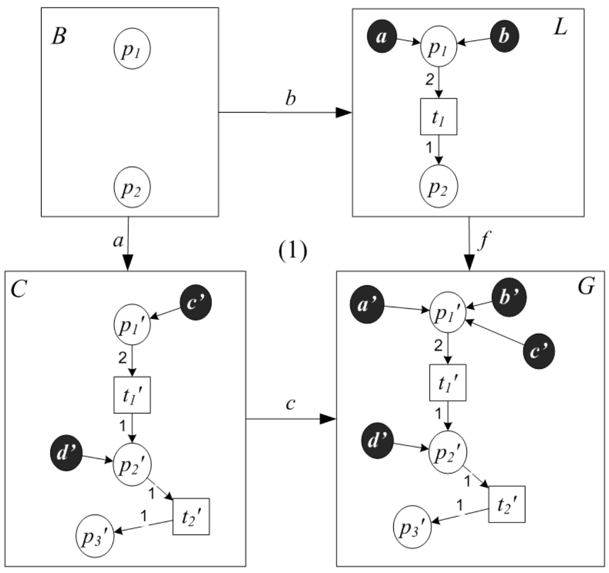
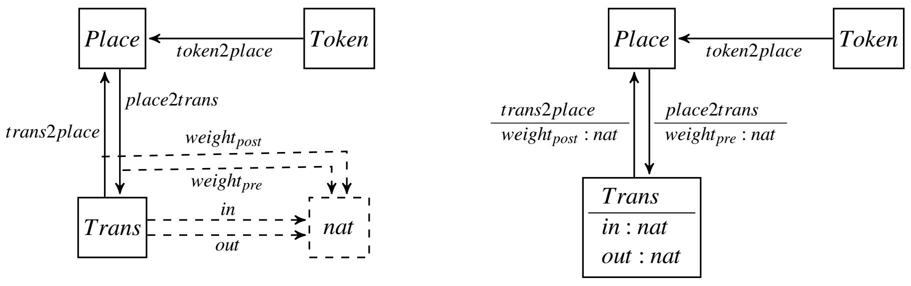


Figure 2: PO in **PTINet**

In the following we only consider the restriction of **PTINet** to $\mathcal{M}_1$-morphisms, $\mathbf{PTINet}|_{\mathcal{M}_1}$, in order to define the functor $\mathcal{F}$ in Subsection 5.4, because $\mathcal{F}$ is not well-defined on general morphisms. But we use the $\mathcal{M}$-adhesive category $(\mathbf{PTINet}, \mathcal{M}_1)$ in order to define pushouts, because $(\mathbf{PTINet}|_{\mathcal{M}_1}, \mathcal{M}_1)$ is not $\mathcal{M}_1$-adhesive due to the well-known fact, that the induced morphism of $\mathcal{M}_1$-morphisms is in general not an $\mathcal{M}_1$-morphism.

5.2 Typed Atributed Graphs over Petri Net Type Graph PNTG

According to [EEPT06] the category $(\mathbf{AGraphs_{ATG}}, \mathcal{M}_2)$ of typed attributed graphs with class $\mathcal{M}_2$ of all injective morphisms with isomorphism on the data type part is $\mathcal{M}$-adhesive, where pushouts along $\mathcal{M}_2$-morphisms are constructed componentwise in the graph part.



$\Sigma - nat$	$sorts :$	nat
	$opns :$	$z : \to nat$
		$succ : nat \to nat$

NAT	$NAT_{nat} = \mathbb{N}$
	$z_{nat} = 0 \in \mathbb{N}$
	$succ_{nat} : \mathbb{N} \to \mathbb{N} \quad x \mapsto x+1$

Figure 3: Type graph PNTG with data type signature $\Sigma - nat$ and algebra NAT

$$\begin{tikzcd}
 & E_G \arrow[rr, shift left, "s_G"] \arrow[rr, shift right, "t_G"'] & & V_G & \\
E_{EA} \arrow[ur, "s_{EA}"] \arrow[drr, "t_{EA}"'] & & & & E_{NA} \arrow[ul, "s_{NA}"'] \arrow[dll, "t_{NA}"] \\
 & & V_D & &
\end{tikzcd}$$

Objects in **AGraphs** are pairs (G,D) of an E-Graph G with signature E (shown to the right), and $\Sigma - nat$ data type D, where in the following we only use $D = T_{\Sigma - nat} \cong NAT$. This means, G is given by $G = (V_G^G, V_D^G = \mathbb{N}, E_G^G, E_{NA}^G, E_{EA}^G, (s_j^G, t_j^G)_{j \in \{G,NA,EA\}})$, where V_G^G resp. V_D^G are the graph resp.-data nodes of G, E_G^G, E_{NA}^G, resp. E_{EA}^G are the graph edges resp. node attribute and edge attribute edges of G and s_j^G, t_j^G are corresponding source and target functions for the edges.

In our case, the type graph ATG is the Petri net type graph $PNTG$ shown in Figure 3 with data type signature $\Sigma{-}nat$ and algebra $T_{\Sigma - nat} \cong NAT$ for rules and graphs, where the E-Graph of $PNTG$ is shown on the left and its attribute notation on the right of Figure 3. Objects in $\mathbf{AGraphs_{PNTG}}$ are pairs $(AN, type)$ with attributed graph $AN = (G,D)$ with $D = NAT$ and **AGraphs**-morphism $type : (G,D) \to (PNTG, D_{fin})$ with final $\Sigma{-}nat$ data type D_{fin}. Morphisms in $\mathbf{AGraphs_{PNTG}}$ are defined componentwise and are type compatible with morphisms in **AGraphs**. Four sample morphisms in $\mathbf{AGraphs_{PNTG}}$ are shown in Figure 4, where a pushout is constructed.

5.3 Translation of PTI Nets into PNTG-typed Attributed Graphs

A formal definition of the functor $\mathscr{F}$ on objects is given as follows.

Definition 10 (Translation of **PTINet**-Objects)
Given a PTI net $NI = (P, T, pre, post, I, m)$. We define the object $\mathscr{F}(NI) = ((G, NAT), type)$ in $\mathbf{AGraphs_{PNTG}}$ with $type : (G, NAT) \to (PNTG, D_{fin})$ and $G = (V_G^G, V_D^G = \mathbb{N}, E_G^G, E_{NA}^G, E_{EA}^G,$ $(s_j^G, t_j^G)_{j \in \{G,NA,EA\}})$ as follows, where we use the following abbreviations: $token2place \triangleq to2p$, $place2trans \triangleq p2t, trans2place \triangleq t2p, weight_{pre} \triangleq w_{pre}, weight_{post} \triangleq w_{post}$ and $pre(t)(p) = n_P \in \mathbb{N}$ for $pre(t) = \sum_{p \in P} n_P \cdot p \in P^{\oplus}$ and similar for $post(t)(p)$.

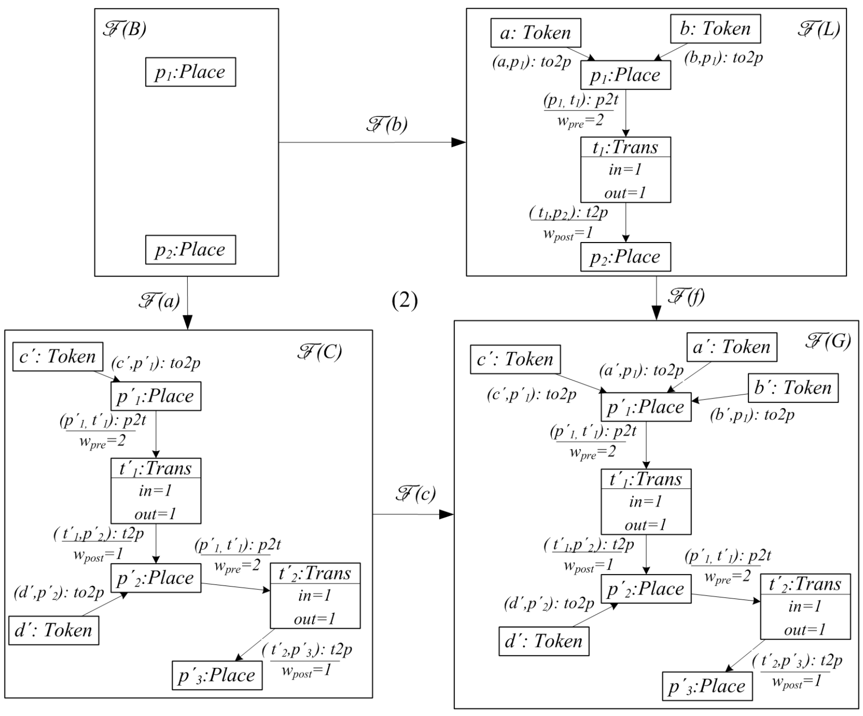


Figure 4: PO in $\mathbf{AGraphs_{PNTG}}$

$V_G^G = P \uplus T \uplus I$

$E_G^G = E_{to2p}^G \uplus E_{p2t}^G \uplus E_{t2p}^G$ with

$E_{to2p}^G = \{(x,p) \in I \times P \mid m(x) = p\}$, $E_{p2t}^G = \{(p,t) \in P \times T \mid pre(t)(p) > 0\}$ and

$E_{t2p}^G = \{(t,p) \in T \times P \mid post(t)(p) > 0\}$

$E_{NA}^G = E_{in}^G \uplus E_{out}^G$ with

$E_{in}^G = \{(t,n,in) \mid (t,n) \in T \times \mathbb{N} \wedge |\bullet t| = n\}$,

$E_{out}^G = \{(t,n,out) \mid (t,n) \in T \times \mathbb{N} \wedge |t \bullet| = n\}$,

where $\bullet t$ and $t\bullet$ are the pre- and post-domains of $t \in T$ with cardinalities $|\bullet t|$ and $|t \bullet|$.

$E_{EA}^G = E_{w_{pre}}^G \uplus E_{w_{post}}^G$ with

$E_{w_{pre}}^G = \{(p,t,n) \in E_{p2t}^G \times \mathbb{N} \mid pre(t)(p) = n\}$

$E_{w_{post}}^G = \{(t,p,n) \in E_{t2p}^G \times \mathbb{N} \mid post(t)(p) = n\}$

$s_G^G, t_G^G : E_G^G \to V_G^G$ defined by $s_G^G(a,b) = a$ resp. $t_G^G(a,b) = b$

$s_{NA}^G : E_{NA}^G \to V_G^G$, $t_{NA}^G : E_{NA}^G \to \mathbb{N}$ defined by $s_{NA}^G(t,n,x) = t$ resp. $t_{NA}^G(t,n,x) = n$

$s_{EA}^G : E_{EA}^G \to E_G^G$ defined by $s_{EA}^G(p,t,n) = (p,t)$ and $s_{EA}^G(t,p,n) = (t,p)$

$t_{EA}^G : E_{EA}^G \to \mathbb{N}$ defined by $t_{EA}^G(p,t,n) = n$ and $t_{EA}^G(t,p,n) = n$

The corresponding *type*-morphism is given in Definition 11 below.

An example for using the functor $\mathscr{F}$ on objects is shown in Figure 4, where the four typed attributed graphs are translations of the corresponding four PTI nets in Figure 2.

Definition 11 (**AGraphs$_{\mathbf{PNTG}}$**-Morphism *type*)
The **AGraphs$_{\mathbf{PNTG}}$**-morphism $type : (G, NAT) \to (PNTG, D_{fin})$ is given by final morphism of data types and $type^G : G \to PNTG$ given by E-graph morphism $type^G = (type_{V_G}, type_{V_D}, type_{E_G}, type_{E_{NA}}, type_{E_{EA}})$ where

$$\begin{aligned}
type_{V_G} &: V_G^G \to V_G^{PNTG} \text{ with } x \mapsto Place\ (x \in P),\ x \mapsto Trans\ (x \in T),\ x \mapsto Token\ (x \in I)\\
type_{V_D} &: \mathbb{N} \to D_{fin_{nat}} \text{ with } x \mapsto nat\ (x \in \mathbb{N})\\
type_{E_G} &: E_G^G \to E_G^{PNTG} \text{ with } x \mapsto y \text{ for } x \in E_y^G \text{ and } y \in \{to2p, p2t, t2p\}\\
type_{E_{NA}} &: E_{NA}^G \to E_{NA}^{PNTG} \text{ with } x \mapsto y \text{ for } x \in E_y^G \text{ and } y \in \{in, out\}\\
type_{E_{EA}} &: E_{EA}^G \to E_{EA}^{PNTG} \text{ with } x \mapsto y \text{ for } x \in E_y^G \text{ and } y \in \{w_{pre}, w_{post}\}
\end{aligned}$$

5.4 Translation of Restricted PTINet-Morphisms into Restricted AGraphs$_{\mathbf{PNTG}}$-Morphisms

We now define the functor $\mathscr{F} : \mathbf{PTINet}|_{\mathscr{M}_1} \to \mathbf{AGraphs_{PNTG}}|_{\mathscr{M}_2}$ on injective morphisms. A counterexample for the translation of non-injective morphisms is given in Figure 1, examples for injective morphisms in Figure 2 and corresponding translated morphisms in Figure 4.

Definition 12 (Translation of **PTINet**-Morphisms)
For each **PTINet**-morphism $f : NI_1 \to NI_2$ with $f = (f_P, f_T, f_I) \in \mathscr{M}_1$, i.e. f_P, f_T, f_I injective, we define $\mathscr{F}(f) : \mathscr{F}(NI_1) \to \mathscr{F}(NI_2)$ where
$\mathscr{F}(NI_i) = (V_{iG}, \mathbb{N}, E_{iG}, E_{iNA}, E_{iEA}, (s_{ij}, t_{ij})_{j \in \{G,NA,EA\}})$ with $i = 1,2$
by $\mathscr{F}(f) = f' = (f'_{V_G}, f'_{V_D}, f'_{E_G}, f'_{E_{NA}}, f'_{E_{EA}})$ with

$$\begin{aligned}
&f'_{V_G} : V_{1G} \to V_{2G} \text{ with } V_{iG} = P_i \uplus T_i \uplus I_i \text{ for } i = 1,2 \text{ by } f'_{V_G} = f_P \uplus f_T \uplus f_I\\
&f'_{V_D} : \mathbb{N} \to \mathbb{N} \text{ by } f'_{V_D} = id_{\mathbb{N}}\\
&f'_{E_G} : E_{1G} \to E_{2G} \text{ with } E_{iG} = E_{ito2p} \uplus E_{ip2t} \uplus E_{it2p} \text{ by}\\
&\quad f'_{E_G}(x,p) = (f_I(x), f_P(p)) \text{ for } (x,p) \in E_{1to2p}\\
&\quad f'_{E_G}(p,t) = (f_P(p), f_T(t)) \text{ for } (p,t) \in E_{1p2t}\\
&\quad f'_{E_G}(t,p) = (f_T(t), f_P(p)) \text{ for } (t,p) \in E_{1t2p}\\
&f'_{E_{NA}} : E_{1NA} \to E_{2NA} \text{ with } E_{iNA} = E_{iin} \uplus E_{iout} \text{ by}\\
&\quad f'_{E_{NA}}(t,n,i) = (f_T(t), n, i) \text{ for } (t,n,i) \in E_{1in} \uplus E_{1out} \wedge i \in \{in, out\}\\
&f'_{E_{EA}} : E_{1EA} \to E_{2EA} \text{ with } E_{iEA} = E_{iw_{pre}} \uplus E_{iw_{post}} \text{ by}\\
&\quad f'_{E_{EA}}(p,t,n) = (f_P(p), f_T(t), n) \text{ for } (p,t,n) \in E_{1w_{pre}}\\
&\quad f'_{E_{EA}}(t,p,n) = (f_T(t), f_P(p), n) \text{ for } (t,p,n) \in E_{1w_{post}}
\end{aligned}$$

Lemma 1 (Well-Definedness of Morphism Translation)
For each $f : NI_1 \to NI_2$ in **PTINet** *with $f \in \mathscr{M}_1$ is $\mathscr{F}(f) : \mathscr{F}(NI_1) \to \mathscr{F}(NI_2)$ in* $\mathbf{AGraphs_{PNTG}}$ *well-defined with $\mathscr{F}(f) \in \mathscr{M}_2$. Moreover $\mathscr{F}$ preserves inclusions.*

Proof.
A detailed proof is given in [MEE11] showing the following steps:

1. $f'_{V_G}, f'_{V_D}, f'_{E_G}, f'_{E_{NA}}, f'_{E_{EA}}$ are well-defined w.r.t. codomain.
2. The components of $\mathscr{F}(f)$ are compatible with sources and targets.
3. The components of $\mathscr{F}(f)$ are compatible with typing morphisms.
4. $f \in \mathscr{M}_1$ (inclusion) implies $\mathscr{F}(f) \in \mathscr{M}_2$ (inclusion).

□

5.5 Translation of Pushouts

We have to show, that if (1) is a PO in **PTINet** with $f_i \in \mathscr{M}_1$, then we have that (2) is a PO in $\mathbf{AGraphs_{PNTG}}$ with $\mathscr{F}(f_i) \in \mathscr{M}_2$.

$$\begin{tikzcd}
NI_0 \arrow[r, "f_1"] \arrow[d, "f_2"'] \arrow[dr, phantom, "(1)"] & NI_1 \arrow[d, "f_4"] \\
NI_2 \arrow[r, "f_3"] & NI_3
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathscr{F}(NI_0) \arrow[r, "\mathscr{F}(f_1)"] \arrow[d, "\mathscr{F}(f_2)"'] \arrow[dr, phantom, "(2)"] & \mathscr{F}(NI_1) \arrow[d, "\mathscr{F}(f_4)"] \\
\mathscr{F}(NI_2) \arrow[r, "\mathscr{F}(f_3)"] & \mathscr{F}(NI_3)
\end{tikzcd}$$

Since POs in **PTINet** are constructed componentwise, we know that the P-, T- and I-components of (1) are POs in **Sets**. Since also POs in $\mathbf{AGraphs_{ATG}}$ and $\mathbf{AGraphs_{PNTG}}$ are constructed componentwise we have to show that the V_G-, V_D-, E_G-, E_{NA}- and E_{EA}-components of (2) are POs in **Sets**. This is clear for the V_G-components $f_{iV_G} = f_{iP} \uplus f_{iT} \uplus f_{iI}$, because POs are compatible with coproducts and for f_{iD}, because all components are identities. For the E_G-component we have to show, that (3) is PO in **Sets**, which follows if (4) and similar $(4a)$ resp. $(4b)$ with "to2p" and "$f_{iI} \times f_{iP}$" replaced by "p2t" and "$f_{iP} \times f_{iT}$" resp. "t2p" and "$f_{iT} \times f_{iP}$" are POs.

$$\begin{tikzcd}
E_{0G} \arrow[r, "\mathscr{F}(f_1)_G"] \arrow[d, "\mathscr{F}(f_2)_G"'] \arrow[dr, phantom, "(3)"] & E_{1G} \arrow[d, "\mathscr{F}(f_4)_G"] \\
E_{2G} \arrow[r, "\mathscr{F}(f_3)_G"] & E_{3G}
\end{tikzcd}
\qquad
\begin{tikzcd}
E_{0to2p} \arrow[r, "f_{1I} \times f_{1P}"] \arrow[d, "f_{2I} \times f_{2P}"'] \arrow[dr, phantom, "(4)"] & E_{1to2p} \arrow[d, "f_{4I} \times f_{4P}"] \\
E_{2to2p} \arrow[r, "f_{3I} \times f_{3P}"] & E_{3to2p}
\end{tikzcd}$$

For the E_{NA}- and E_{EA} components, it is sufficient to show POs (5) and (6) and similar $(5a)$ with "*in*" replaced by "*out*" and $(6a)$ with "*pre*" replaced by "*post*".

$$\begin{tikzcd}
E_{0in} \arrow[r, "f_{1T} \times id_{\mathbb{N}}"] \arrow[d, "f_{2T} \times id_{\mathbb{N}}"'] \arrow[dr, phantom, "(5)"] & E_{1in} \arrow[d, "f_{4T} \times id_{\mathbb{N}}"] \\
E_{2in} \arrow[r, "f_{3T} \times id_{\mathbb{N}}"] & E_{3in}
\end{tikzcd}
\qquad
\begin{tikzcd}
E_{0w_{pre}} \arrow[r, "f_{1P} \times f_{1T}"] \arrow[d, "f_{2P} \times f_{2T}"'] \arrow[dr, phantom, "(6)"] & E_{1w_{pre}} \arrow[d, "f_{4P} \times f_{4T}"] \\
E_{2w_{pre}} \arrow[r, "f_{3P} \times f_{3T}"] & E_{3w_{pre}}
\end{tikzcd}$$

All these diagrams commute, because each product component commutes by assumption. But it is more difficult to show explicitly, that they are POs (see for example Lemma 2 below), because products of POs are in general not POs. An example is the translation of the PO in **PTINet** shown in Figure 2 to the PO in **AGraphs$_{\mathbf{PNTG}}$** shown in Figure 4.

Lemma 2 (Translation of Pushouts)
Diagrams (4) *and* $(4a)$ *are pushouts.*
Proof. See [MEE11]. □

5.6 Creation of Injective Morphisms

Given $\mathscr{F}(NI_1), \mathscr{F}(NI_2)$ and $f' : \mathscr{F}(NI_1) \to \mathscr{F}(NI_2) \in \mathscr{M}_2$ with type compatible morphisms

$$\begin{aligned}
f'_{V_G} &: V_{1G} \to V_{2G} \text{ with } V_{iG} = P_i \uplus T_i \uplus I_i \text{ for } i = 1,2\\
f'_{V_D} &: \mathbb{N} \to \mathbb{N} \text{ with } f'_{V_D} = id_{\mathbb{N}}\\
f'_{E_G} &: E_{1G} \to E_{2G} \text{ with } E_{iG} = E_{ito2p} \uplus E_{ip2t} \uplus E_{it2p}\\
f'_{E_{NA}} &: E_{1NA} \to E_{2NA} \text{ with } E_{iNA} = E_{iin} \uplus E_{iout}\\
f'_{E_{EA}} &: E_{1EA} \to E_{2EA} \text{ with } E_{iEA} = E_{iw_{pre}} \uplus E_{iw_{post}}
\end{aligned}$$

Define $f : NI_1 \to NI_2$ with $NI_j = (P_j, T_j, pre_j, post_j, I_j, m_j)$ for $j = 1,2$ by $f = (f_P : P_1 \to P_2, f_T : T_1 \to T_2, f_I : I_1 \to I_2)$ with

$$\begin{aligned}
f_T(t) &= f'_{V_G}(t) \text{ for } t \in T_1 \subseteq V_{1G}\\
f_P(p) &= f'_{V_G}(p) \text{ for } p \in P_1 \subseteq V_{1G}\\
f_I(x) &= f'_{V_G}(x) \text{ for } x \in I_1 \subseteq V_{1G}
\end{aligned}$$

Well-definedness of $f : NI_1 \to NI_2 \in \mathscr{M}_1$ follows from Lemma 3 below, where the proof of part 2 is based on Lemma 4. The proofs of both Lemma are given in [MEE11].

Lemma 3 (Well-Definedness of Creation of Injective Morphisms)
Given the construction above for $f : NI_1 \to NI_2$. *The following holds:*

1. $f'_{V_G}(t) \in T_2$, $f'_{V_G}(p) \in P_2$, $f'_{V_G}(x) \in I_2$, *and*
2. *squares* $(1), (2)$ *to the right commute with injective* f_P, f_T, f_I.

$$\begin{tikzcd}
T_1 \arrow[r, shift left, "pre_1"] \arrow[r, shift right, "post_1"'] \arrow[d, "f_T"'] \arrow[dr, phantom, "(1)"] & P_1^{\oplus} \arrow[d, "f_P^{\oplus}"] \\
T_2 \arrow[r, shift left, "pre_2"] \arrow[r, shift right, "post_2"'] & P_2^{\oplus}
\end{tikzcd}
\qquad
\begin{tikzcd}
I_1 \arrow[r, "m_1"] \arrow[d, "f_I"'] \arrow[dr, phantom, "(2)"] & P_1 \arrow[d, "f_P"] \\
I_2 \arrow[r, "m_2"] & P_2
\end{tikzcd}$$

Lemma 4 (PTI-Morphism-Lemma)
$f : NI_1 \to NI_2$ *is an injective* **PTINet***-morphism* $\Leftrightarrow$
$f = (f_P, f_T, f_I)$ *is injective with* $1-4$, *where*

1. $\forall t \in T_1 . p \in \bullet t \Leftrightarrow f_P(p) \in \bullet f_T(t)$ *and* $\forall t \in T_1 . p \in t\bullet \Leftrightarrow f_P(p) \in f_T(t)\bullet$
2. $\forall (p,t) \in P_1 \otimes T_1 = E_{1p2t} . \, (p,t,n) \in E_{1w_{pre}} \Leftrightarrow (f_P(p), f_T(t), n) \in E_{2w_{pre}}$ *and*
$\forall (t,p) \in T_1 \otimes P_1 = E_{1t2p} . \, (t,p,n) \in E_{1w_{post}} \Leftrightarrow (f_T(t), f_P(p), n) \in E_{2w_{post}}$

3. $\forall t \in T_1$.
   $card(\bullet t) = n \Leftrightarrow card(\bullet f_T(t)) = n$ *and* $card(t\bullet) = n \Leftrightarrow card(f_T(t)\bullet) = n$ *with*
   $\bullet t = \{p \in P_1 \mid pre_1(t)(p) > 0\}$ *and* $t\bullet = \{p \in P_1 \mid post_1(t)(p) > 0\}$

4. $\forall x \in I_1.(x,p) \in E_{1to2p} \Leftrightarrow (f_I(x), f_P(p)) \in E_{2to2p}$

5.7 Preservation of Initial Pushouts

The proof of this property is based on the initial PO constructions for **PTINet** in [MGE+09] and for **AGraphs**$_{\mathbf{ATG}}$ in [EEPT06]. Details of the proof are given in [MEE11]. An example is given in Figure 5, where (1) is an initial PO over f in **PTINet**, (2) the induced PO over $\mathscr{F}(f)$, and the initial PO over $\mathscr{F}(f)$ in **AGraphs**$_{\mathbf{PNTG}}$ is given by the outer diagram with corners $B', C', \mathscr{F}(L), \mathscr{F}(G)$. Since i' and j' are isomorphisms, diagram (2) is already initial PO over $\mathscr{F}(f)$.

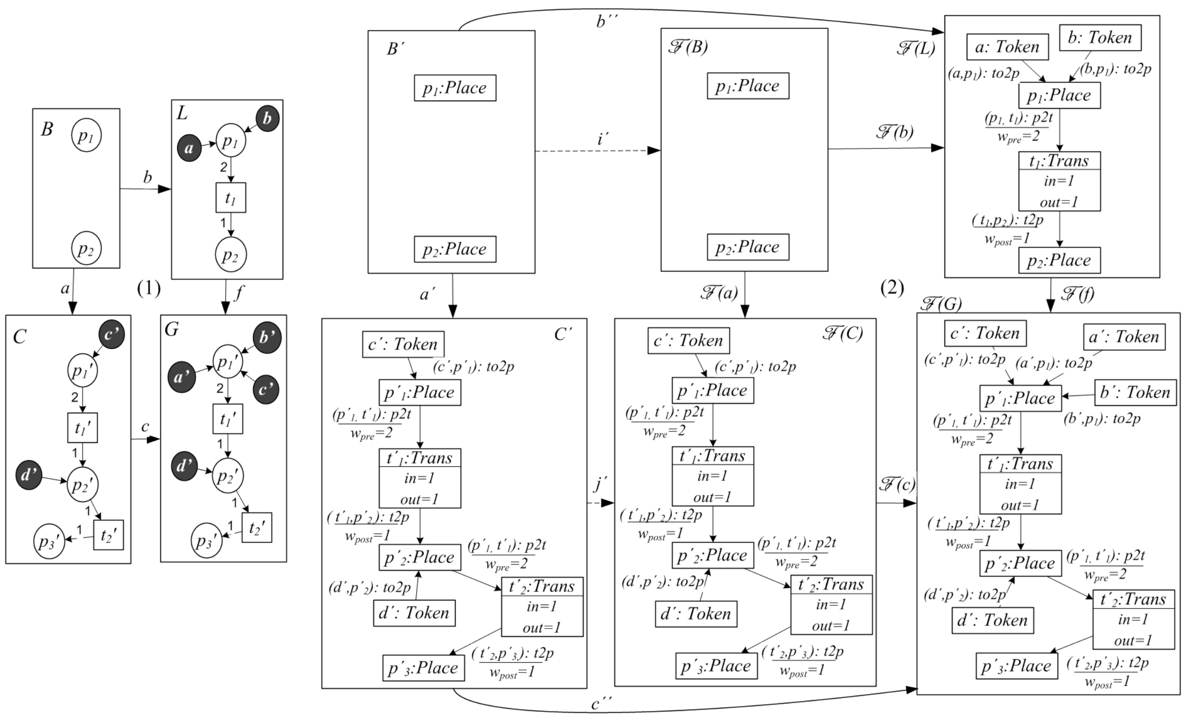


Figure 5: Preservation of Initial Pushouts

6 Conclusion and Future Work

As pointed out already in Section 1 we want to develop a general framework to establish a formal relationship between different $\mathcal{M}$-adhesive transformation systems based on $\mathcal{M}$-adhesive categories. The main idea is to construct a suitable $\mathcal{M}$-functor between the corresponding $\mathcal{M}$-adhesive categories, which translates pushouts, creates morphisms and preserves initial pushouts.

This allows by Theorem 1 and Theorem 2 the translation and creation of transformations between the corresponding $\mathscr{M}$-adhesive transformation systems, including parallel and sequential independence of transformations. Moreover, we have discussed the restriction to injective matches via $\mathscr{M}_1$-morphisms, which requires only a functor for $\mathscr{M}_1$-morphisms.
In Section 5 we have discussed a corresponding functor from Petri nets with individual tokens to typed attributed graphs. We have verified that this functor translates pushouts of $\mathscr{M}_1$-morphisms, creates $\mathscr{M}_1$-morphisms and preserves initial pushouts over $\mathscr{M}_1$-morphisms, which allows the application of Theorem 1 and Theorem 2 in connection with Remark 4.

In future work, we will provide sufficient conditions in order to ensure that the $\mathscr{M}-$functor preserves initial pushouts[2]. In the long run, this should allow the analysis of interesting properties of Petri net transformation systems, like termination and local confluence in addition to parallel and sequential independence, using corresponding results and analysis tools like AGG for graph transformation systems. Moreover, it is interesting to study the relationship between other $\mathscr{M}$-adhesive transformation systems using this approach, e.g. high-level Petri nets and typed attributed graphs as well as triple graphs and flattening of triple graphs.

Acknowledgements: Some of the authors are partly supported by the German Research Council project *Behaviour Simulation and Equivalence of Systems Modelled by Graph Transformation (Behaviour-GT)*.

References

[AGG09] TFS-Group, TU Berlin. AGG. 2009. http://tfs.cs.tu-berlin.de/agg.

[BEHM07] E. Biermann, C. Ermel, F. Hermann, T. Modica. A Visual Editor for Reconfigurable Object Nets based on the ECLIPSE Graphical Editor Framework. In *Proc. Workshop on Algorithms and Tools for Petri Nets (AWPN'07)*. 2007. http://tfs.cs.tu-berlin.de/publikationen/Papers07/BEHM07.pdf

[DS02] M. Droste, R. M. Shortt. From Petri Nets to Automata with Concurrency. *Applied Categorical Structures* 10:173–191, 2002. 10.1023/A:1014305610452. http://dx.doi.org/10.1023/A:1014305610452

[EEPT06] H. Ehrig, K. Ehrig, U. Prange, G. Taentzer. *Fundamentals of Algebraic Graph Transformation*. EATCS Monographs in Theor. Comp. Science. Springer, 2006.

[EGH10] H. Ehrig, U. Golas, F. Hermann. Categorical Frameworks for Graph Transformation and HLR Systems based on the DPO Approach. *Bulletin of the EATCS* 102:111–121, 2010. http://tfs.cs.tu-berlin.de/publikationen/Papers10/EGH10.pdf

[EHKP91a] H. Ehrig, A. Habel, H.-J. Kreowski, F. Parisi-Presicce. From Graph Grammars to high level replacement systems. In *4th Int. Workshop on Graph Grammars and their Application to Computer Science*. LNCS 532, pp. 269–291. Springer, 1991.

[2] First steps in this direction are Lemma 5 and Lemma 6 in [MEE11].

[EHKP91b] H. Ehrig, A. Habel, H.-J. Kreowski, F. Parisi-Presicce. Parallelism and Concurrency in High-Level Replacement Systems. *Math. Struct. in Comp. Science* 1:361–404, 1991.

[EHP+08] H. Ehrig, K. Hoffmann, J. Padberg, C. Ermel, U. Prange, E. Biermann, T. Modica. Petri Net Transformations. In *Petri Net Theory and Applications*. Pp. 1–16. I-Tech Education and Publication, 2008.

[EHPP06] H. Ehrig, A. Habel, J. Padberg, U. Prange. Adhesive High-Level Replacement Systems: A New Categorical Framework for Graph Transformation. *Fundamenta Informaticae* 74(1):1–29, 2006.

[Kre81] H.-J. Kreowski. A Comparison between Petri Nets and Graph Grammars. In *5th International Workshop on Graph-Theoretic Concepts in Computer Science*. LNCS 100, pp. 1–19. Springer, 1981.

[LS04] S. Lack, P. Sobociński. Adhesive Categories. In *Proc. FOSSACS 2004*. LNCS 2987, pp. 273–288. Springer, 2004.

[MEE11] M. Maximova, H. Ehrig, C. Ermel. Functors between $\mathscr{M}$-adhesive Categories Applied to Petri Net and Graph Transformation Systems. Technical report 2011/04, TU Berlin, 2011.
http://www.eecs.tu-berlin.de/menue/forschung/forschungsberichte/2011

[MGE+09] T. Modica, K. Gabriel, H. Ehrig, K. Hoffmann, S. Shareef, C. Ermel, U. Golas, F. Hermann, E. Biermann. Low- and High-Level Petri Nets with Individual Tokens. Technical report 2009/13, Technische Universität Berlin, 2009. http://www.eecs.tu-berlin.de/menue/forschung/forschungsberichte/2009.

[MM90] J. Meseguer, U. Montanari. Petri Nets are Monoids. *Information and Computation* 88(2):105–155, 1990.

[PE07] U. Prange, H. Ehrig. From Algebraic Graph Transformation to Adhesive HLR Categories and Systems. In *Algebraic Informatics. Proceedings of CAI 2007*. LNCS 4728, pp. 122–146. Springer, 2007.

[PEHP08] U. Prange, H. Ehrig, K. Hoffman, J. Padberg. Transformations in Reconfigurable Place/Transition Systems. In *Concurrency, Graphs and Models: Essays Dedicated to Ugo Montanari on the Occasion of His 65th Birthday*. LNCS 5065, pp. 96–113. Springer, 2008.

[Rei85] W. Reisig. *Petri Nets: An Introduction*. EATCS Monographs on Theoretical Computer Science 4. Springer, 1985.

[TFS07] TFS-Group, TU Berlin. Reconfigurable Object Nets Environment. 2007.
http://www.tfs.cs.tu-berlin.de/roneditor