Electronic Communications of the EASST
Volume 51 (2012)

Proceedings of the 5th International Workshop on Petri Nets, Graph Transformation and other Concurrency Formalisms (PNGT 2012)

Transfer of Local Confluence and Termination between Petri Net and Graph Transformation Systems Based on $\mathscr{M}$-Functors

Maria Maximova, Hartmut Ehrig and Claudia Ermel

12 pages

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

Transfer of Local Confluence and Termination between Petri Net and Graph Transformation Systems Based on $\mathscr{M}$-Functors

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: Recently, a formal relationship between Petri net and graph transformation systems has been established using the new framework of $\mathscr{M}$-functors $\mathscr{F}$: $(\mathbf{C_1}, \mathscr{M}_1) \to (\mathbf{C_2}, \mathscr{M}_2)$ between $\mathscr{M}$-adhesive categories. This new approach allows to translate transformations in $(\mathbf{C_1}, \mathscr{M}_1)$ into corresponding transformations in $(\mathbf{C_2}, \mathscr{M}_2)$ and, vice versa, to create transformations in $(\mathbf{C_1}, \mathscr{M}_1)$ from those in $(\mathbf{C_2}, \mathscr{M}_2)$. This is helpful because our tool for reconfigurable Petri nets, the RON-tool, performs the analysis of Petri net transformations by analyzing corresponding graph transformations using the AGG-tool. Up to now, this correspondence has been implemented as a converter on an informal level. The formal correspondence results given by our framework make the RON-tool more reliable. In this paper we extend this framework to the transfer of local confluence, termination and functional behavior. In particular, we are able to create these properties for transformations in $(\mathbf{C_1}, \mathscr{M}_1)$ from corresponding properties of transformations in $(\mathbf{C_2}, \mathscr{M}_2)$, where $(\mathbf{C_1}, \mathscr{M}_1)$ are Petri nets with individual tokens and $(\mathbf{C_2}, \mathscr{M}_2)$ typed attributed graphs. This allows us to apply the well-known critical pair analysis for typed attributed graph transformations supported by the AGG-tool in order to analyze these properties for Petri net transformations.

Keywords: $\mathscr{M}$-adhesive transformation system, graph transformation, Petri net transformation, confluence, termination, functional behavior

1 Introduction

Reconfigurable Petri nets have been introduced to enable formal modeling of controlled reconfiguration of dynamic systems, which has proven to be useful for application areas such as computer supported cooperative work [HM10] or mobile networks [PEH07].

Petri net reconfiguration is modeled in a rule-based setting, where the structure of place/transition nets may be changed by applying net transformation rules [EHP+08, PEHP08]. The approach is related to graph transformation [EEPT06], where structural changes are modeled in the double-pushout (DPO) approach for the category of (typed, attributed) graphs, and has been generalized to $\mathscr{M}$-adhesive categories, which rely on a class $\mathscr{M}$ of monomorphisms, generalizing weak adhesive HLR categories. The DPO approach is a suitable description of transformations leading to results like the Local Church-Rosser, Parallelism, Concurrency, Embedding, Extension, and Local Confluence Theorems [EEPT06]. The well-established tool AGG [AGG09] supports modeling and analysis of (typed, attributed) graph transformation systems.

In our previous paper [MEE11], we have proposed formal criteria ensuring a semantical correspondence of reconfigurable Petri nets and their corresponding representations as graph transformation systems. The aim of our previous work was to establish a formal basis allowing us to translate Petri net transformations into graph transformations and, vice versa, to create Petri net transformations from graph transformations such that the behavior of Petri net transformations can be simulated by simulating their translation using the graph transformation tool AGG.

In [MEE11], we established the new framework of $\mathscr{M}$-functors $\mathscr{F} : (\mathbf{C_1}, \mathscr{M}_1) \to (\mathbf{C_2}, \mathscr{M}_2)$ between $\mathscr{M}$-adhesive categories. This framework allows to translate transformations in $(\mathbf{C_1}, \mathscr{M}_1)$ into corresponding transformations in $(\mathbf{C_2}, \mathscr{M}_2)$ and, vice versa, to create transformations in $(\mathbf{C_1}, \mathscr{M}_1)$ from those in $(\mathbf{C_2}, \mathscr{M}_2)$.

Building on this previous work, we now extend our framework to allow the analysis of interesting properties of Petri net transformation systems, like termination, local confluence and functional behavior, in addition to parallel and sequential independence, using corresponding results and analysis tools like AGG for graph transformation systems. We have shown in [MEE11] that we can create these properties for transformations in $(\mathbf{C_1}, \mathscr{M}_1)$ from corresponding properties of transformations in $(\mathbf{C_2}, \mathscr{M}_2)$, where $(\mathbf{C_1}, \mathscr{M}_1)$ are Petri nets with individual tokens and $(\mathbf{C_2}, \mathscr{M}_2)$ typed attributed graphs. This allows us in particular to apply the well-known critical pair analysis for typed attributed graph transformations supported by the AGG-tool in order to analyze these properties for Petri net transformations. In our tool for reconfigurable Petri nets, (the RON-tool [RON07, BEHM07] for modeling and analyzing *Reconfigurable Object Nets*), the analysis of Petri net transformations is performed by analyzing corresponding graph transformations using AGG. Up to now, this correspondence has been implemented as a converter from RON models to graph transformation systems on an informal level. The formal correspondence results for analysis properties in this paper make our tool environment much more reliable.

The paper is structured as follows: Section 2 introduces the basic notions of $\mathscr{M}$-adhesive transformation system and $\mathscr{M}$-functor to define a formal relationship between two different $\mathscr{M}$-adhesive categories. In Section 3, we recall the definition of the $\mathscr{M}$-functor between Petri net and graph transformation systems from [MEE11]. The main new results are elaborated in Section 4 ($\mathscr{F}$-transfer of local confluence), and in Section 5 ($\mathscr{F}$-transfer of termination and functional behavior). In Section 6, we compare our approach to related work, conclude the paper and give an outlook to future research directions. For detailed proofs see [MEE12].

2 Basic Concepts

In this section we concentrate on some basic concepts and results that are important for our approach. Our considerations are based on the framework of $\mathscr{M}$-adhesive categories. An $\mathscr{M}$-adhesive category [EGH10], consists of a category $\mathbf{C}$ together with a class $\mathscr{M}$ of monomorphisms such that the following properties hold: $\mathbf{C}$ has pushouts (POs) and pullbacks (PBs) along $\mathscr{M}$-morphisms, $\mathscr{M}$ is closed under isomorphisms, composition, decomposition, POs and PBs, and POs along $\mathscr{M}$-morphisms are $\mathscr{M}$-VK-squares (see Figure 1), i.e. the VK-property holds for all commutative cubes, where the given PO with $m \in \mathscr{M}$ is in the bottom, the back faces are PBs and all vertical morphisms a, b, c and d are in $\mathscr{M}$. The VK-property means that the top face is a PO iff the front faces are PBs.

The concept of $\mathscr{M}$-adhesive categories generalizes that of adhesive [LS04], adhesive HLR, and weak adhesive HLR categories [EEPT06]. The category of typed attributed graphs and several categories of Petri nets are weak adhesive HLR (see [EEPT06]) and hence also $\mathscr{M}$-adhesive. A set of transformation rules in an $\mathscr{M}$-adhesive category constitutes an $\mathscr{M}$-adhesive transformation system [EGH10].

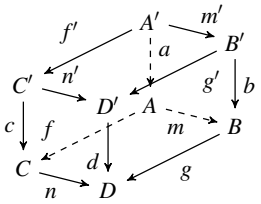


Figure 1: $\mathscr{M}$-VK-square

Definition 1 ($\mathscr{M}$-Adhesive Transformation System)
Given an $\mathscr{M}$-adhesive category $(\mathbf{C},\mathscr{M})$, an $\mathscr{M}$-adhesive transformation system $AS=(\mathbf{C},\mathscr{M},P)$ has a set P of productions of the form $\rho=(L \xleftarrow{l} K \xrightarrow{r} R)$ with $l,r\in\mathscr{M}$. A direct transformation $G \overset{\rho,m}{\Longrightarrow} H$ via ρ and match m consists of two pushouts according to the DPO approach [EEPT06].

We use the notion of an $\mathscr{M}$-functor [MEE11] to define a formal relationship between two different $\mathscr{M}$-adhesive transformation systems.

Definition 2 ($\mathscr{M}$-Functor)
A functor $\mathscr{F}:(\mathscr{C}_1,\mathscr{M}_1)\to(\mathscr{C}_2,\mathscr{M}_2)$ 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.

Given an $\mathscr{M}$-adhesive transformation system $AS_1=(\mathscr{C}_1,\mathscr{M}_1,P_1)$, we want to translate transformations from AS_1 to $AS_2=(\mathscr{C}_2,\mathscr{M}_2,P_2)$ with translated productions $P_2=\mathscr{F}(P_1)$ and, vice versa, we want to create transformations in AS_1 from the corresponding transformations in AS_2. This can be handled by Theorem 1 below, shown in [MEE11].

By definition, each $\mathscr{M}$-functor $\mathscr{F}:(\mathscr{C}_1,\mathscr{M}_1)\to(\mathscr{C}_2,\mathscr{M}_2)$ translates each production $\rho=(L \xleftarrow{l} K \xrightarrow{r} R)$ in P_1 with $l,r\in\mathscr{M}_1$ into $\mathscr{F}(\rho)=(\mathscr{F}(L) \xleftarrow{\mathscr{F}(l)} \mathscr{F}(K) \xrightarrow{\mathscr{F}(r)} \mathscr{F}(R))$ in $P_2=\mathscr{F}(P_1)$ with $\mathscr{F}(l),\mathscr{F}(r)\in\mathscr{M}_2$ and each direct transformation $G \overset{\rho,m}{\Longrightarrow} H$ in AS_1 given by DPO $(1)+(2)$ into a direct transformation $\mathscr{F}(G) \overset{\mathscr{F}(\rho),\mathscr{F}(m)}{\Longrightarrow} \mathscr{F}(H)$ in AS_2 given by DPO $(3)+(4)$.

$$\begin{array}{ccccc} L & \xleftarrow{l} & K & \xrightarrow{r} & R \\ {\scriptstyle m}\downarrow & (1) & \downarrow{\scriptstyle k} & (2) & \downarrow \\ G & \xleftarrow{} & D & \xrightarrow{} & H \end{array} \quad\Longrightarrow\quad \begin{array}{ccccc} \mathscr{F}(L) & \xleftarrow{\mathscr{F}(l)} & \mathscr{F}(K) & \xrightarrow{\mathscr{F}(r)} & \mathscr{F}(R) \\ {\scriptstyle \mathscr{F}(m)}\downarrow & (3) & \downarrow{\scriptstyle k'} & (4) & \downarrow \\ \mathscr{F}(G) & \leftarrow & \mathscr{F}(D) & \rightarrow & \mathscr{F}(H) \end{array}$$

Vice versa, we say $\mathscr{F}$ creates direct transformations, if for each direct transformation $\mathscr{F}(G) \overset{\mathscr{F}(\rho),m'}{\Longrightarrow} H'$ in AS_2 there is a direct transformation $G \overset{\rho,m}{\Longrightarrow} H$ in AS_1 with $\mathscr{F}(m)=m'$ and $\mathscr{F}(H)\cong H'$ leading to $\mathscr{F}(G) \overset{\mathscr{F}(\rho),\mathscr{F}(m)}{\Longrightarrow} \mathscr{F}(H)$ in AS_2. In the following, we provide two conditions in order to show creation of direct transformations and transformations, i.e. sequences of direct transformations written $G \overset{*}{\Rightarrow} H$.

Theorem 1 (Translation and Creation of Transformations)
Each $\mathscr{M}$-functor $\mathscr{F}:(\mathscr{C}_1,\mathscr{M}_1)\to(\mathscr{C}_2,\mathscr{M}_2)$ translates (direct) transformations.
Vice versa, $\mathscr{F}$ creates (direct) transformations if we have the following two conditions:

- *($\mathscr{F}$ **creates morphisms**):* *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 $\mathscr{F}(m) = m'$.*
- *($\mathscr{F}$ **translates initial pushouts**):* *$(\mathbf{C_1}, \mathscr{M}_1)$ has initial pushouts and for each initial pushout (1) over $m : L \to G$ also (2) is initial pushout over $\mathscr{F}(m)$ in $(\mathbf{C_2}, \mathscr{M}_2)$.*

$$\begin{array}{ccc} B & \xrightarrow{b} & L \\ \downarrow & (1) & \downarrow m \\ C & \longrightarrow & G \end{array} \quad \Rightarrow \quad \begin{array}{ccc} \mathscr{F}(B) & \xrightarrow{\mathscr{F}(b)} & \mathscr{F}(L) \\ \downarrow & (2) & \downarrow \mathscr{F}(m) \\ \mathscr{F}(C) & \longrightarrow & \mathscr{F}(G) \end{array}$$

The proof for Theorem 1 is given in [MEE11]. Moreover, it is shown under the same assumptions that $\mathscr{F}$ translates and creates parallel and sequential independence of transformations. Concerning the definition and the role of initial pushouts for the applicability of productions we refer to [EEPT06, MEE11].

3 $\mathscr{M}$-Functor from Petri Net to Graph Transformation Systems

In this section we recall on the one hand the $\mathscr{M}$-adhesive category $(\mathbf{PTINet}, \mathscr{M}_1)$ of Petri nets with individual tokens (together with the class $\mathscr{M}_1$ of all injective morphisms), on the other hand, we consider the well-known $\mathscr{M}$-adhesive category $(\mathbf{AGraphs_{ATG}}, \mathscr{M}_2)$ of typed attributed graphs (together with the class $\mathscr{M}_2$ of all injective morphisms with isomorphism on the data type part) with a suitable attributed Petri net type graph $ATG = PNTG$ shown in Figure 2 with data type signature $\Sigma{-}nat$ and algebra $T_{\Sigma-nat} \cong NAT$ for rules and graphs. Note that we use the term algebra $T_{\Sigma-nat}$ without variables. Moreover, we explain by example the construction of the restricted $\mathscr{M}$-functor $\mathscr{F} : \mathbf{PTINet}|_{\mathscr{M}_1} \to \mathbf{AGraphs_{PNTG}}|_{\mathscr{M}_2}$ between both categories (see [MEE11]), which is only defined on injective morphisms $\mathscr{M}_1$. We construct the functor between the categories restricted to $\mathscr{M}$-morphisms, but this is 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 [MEE11], where the constructed typed attributed graph morphism does not preserve attributes *in* and w_{pre}). On the other hand, the restriction to injective Petri net morphisms for match and rule morphisms makes sense in view of preserving the firing behavior. This implies that we need only a very restrictive version of morphisms for typed attributed graphs, namely $\mathscr{M}_2$-morphisms, to simulate and create Petri net transformations using rules without variables.

Definition 3 (Category **PTINet** [MGH11])

- Objects in **PTINet** are Petri nets with individual tokens (PTI nets) defined by $NI = (N, I, m)$, where $N = (P, T, pre, post)$ is a classical place/transition net, I is a set of individual tokens and $m : I \to P$ a marking function assigning a place $m(x) \in P$ to each $x \in I$.
- **PTINet**-morphisms are given by a triple of functions $f = (f_P : P_1 \to P_2, f_T : T_1 \to T_2, f_I : I_1 \to I_2) : NI_1 \to NI_2$ s.t. the following diagrams commute with *pre* and *post*, respectively.

$$\begin{array}{ccc} T_1 & \overset{pre_1}{\underset{post_1}{\rightrightarrows}} & P_1^{\oplus} \\ {\scriptstyle f_T}\downarrow & = & \downarrow{\scriptstyle f_P^{\oplus}} \\ T_2 & \overset{pre_2}{\underset{post_2}{\rightrightarrows}} & P_2^{\oplus} \end{array} \qquad \begin{array}{ccc} I_1 & \xrightarrow{m_1} & P_1 \\ {\scriptstyle f_I}\downarrow & = & \downarrow{\scriptstyle f_P} \\ I_2 & \xrightarrow[m_2]{} & P_2 \end{array}$$

The notion of attributed graphs combined with the typing concept leads to the well-known category of typed attributed graphs $\mathbf{AGraphs}_{ATG}$, where attributed graphs are typed over an attributed type graph *ATG* [EEPT06]. Here, we consider a specific type graph *PNTG* (see [MEE11]) to express PTI nets as graphs, which is shown in Figure 2. The meaning of every depicted element of *PNTG* is as follows: The nodes *Place*, *Trans*, and *Token* represent the elements of PTI nets that can be depicted as nodes in typed attributed graphs and hence give the possible node typing. The edges *place2trans* and *trans2place* have the attributes $weight_{pre}$ and $weight_{post}$ which give weights for edges in pre- and postdomain of transitions in the corresponding PTI net. Node *Trans* has two attributes *in* and *out* that encode the number of places in the pre- and postdomain of a transition (to ensure the preservation of a transition node's environment using graph morphisms). All node and edge attributes are typed over natural numbers.

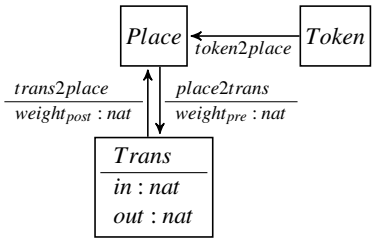


Figure 2: Attributed type graph *PNTG*

In [MEE11] the functor $\mathscr{F}$: $\mathbf{PTINet}|_{\mathscr{M}_1} \to \mathbf{AGraphs}_{\mathbf{PNTG}}|_{\mathscr{M}_2}$ translating PTI nets into typed attributed graphs is formally defined. An example for using the functor on objects is shown in Figure 3, where the typed attributed graph on the right side is the translation of the corresponding PTI net on the left side. Intuitively, the translation works as follows: Individual tokens, places, and transitions are translated into the nodes of the corresponding types *Token*, *Place*, and *Trans*. Edges between individual tokens, places, and transitions are translated into the graph edges of the corresponding types *token2place*, *trans2place*, *place2trans*. Weights of the edges between places and transitions are translated into the $weight_{pre}$ and $weight_{post}$ attributes of the corresponding graph edges. Finally, the number of places in the pre- and postdomain of a transition is recorded as values of the *in* and *out* attributes of the transition node.

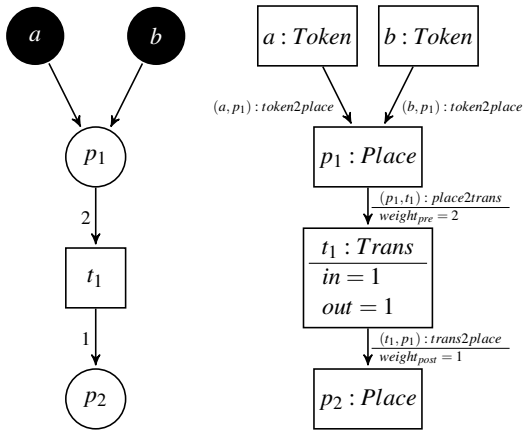


Figure 3: Applying functor $\mathscr{F}$ to a PTI net

Although we do not have an $\mathscr{M}$-functor $\mathscr{F}$: $(\mathbf{PTINet}, \mathscr{M}_1) \to (\mathbf{AGraphs}_{\mathbf{PNTG}}, \mathscr{M}_2)$, it is shown in [MEE11] that we can apply the theory of Section 2, especially Theorem 1, to the restricted $\mathscr{M}$-functor $\mathscr{F}$: $\mathbf{PTINet}|_{\mathscr{M}_1} \to \mathbf{AGraphs}_{\mathbf{PNTG}}|_{\mathscr{M}_2}$ constructed above. In fact, it is sufficient to verify the following adapted properties, which are shown in [MEE11], to apply the theory to transformations with injective matches in $\mathscr{M}_1$ resp. $\mathscr{M}_2$: $\mathscr{F}$ translates pushouts of $\mathscr{M}_1$-morphisms in $(\mathbf{PTINet}, \mathscr{M}_1)$ into pushouts of $\mathscr{M}_2$-morphisms in $(\mathbf{AGraphs}_{\mathbf{PNTG}}, \mathscr{M}_2)$, $\mathscr{F}$ creates $\mathscr{M}_1$-morphisms and $\mathscr{F}$ preserves initial pushouts over $\mathscr{M}_1$-morphisms.

Hence, according to [MEE11], we know already that $\mathscr{F}$ translates and creates rule applicability, construction of (direct) transformations with injective matches as well as parallel and sequential independence of transformations.

4 $\mathscr{F}$-Transfer of Local Confluence

In this section, we consider under which conditions local confluence can be translated by $\mathscr{M}$-functor $\mathscr{F}:(\mathbf{C_1},\mathscr{M}_1)\to(\mathbf{C_2},\mathscr{M}_2)$ from one transformation system $AS_1=(\mathbf{C_1},\mathscr{M}_1,P)$ to another one $AS_2=(\mathbf{C_2},\mathscr{M}_2,\mathscr{F}(P))$ with translated productions $\mathscr{F}(P)$ and, vice versa, under which conditions local confluence of AS_1 can be created by $\mathscr{F}$ from local confluence of AS_2.

According to [EEPT06], an $\mathscr{M}$-adhesive transformation system $(\mathbf{C},\mathscr{M},P)$ is locally confluent, if for all direct transformations $G\Rightarrow H_1$ and $G\Rightarrow H_2$ there is an object X together with transformations $H_1\overset{*}{\Rightarrow}X$ and $H_2\overset{*}{\Rightarrow}X$. In the case of confluence this property is required for transformations $G\overset{*}{\Rightarrow}H_1$ and $G\overset{*}{\Rightarrow}H_2$.

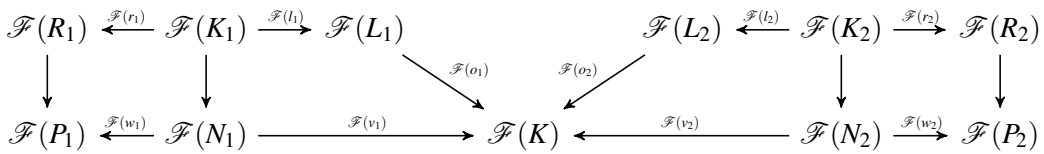
Theorem 2 (Translation and Creation of Local Confluence)
Let $AS_1=(\mathbf{C_1},\mathscr{M}_1,P)$, $AS_2=(\mathbf{C_2},\mathscr{M}_2,\mathscr{F}(P))$ be $\mathscr{M}$-adhesive transformation systems and $\mathscr{F}:(\mathbf{C_1},\mathscr{M}_1)\to(\mathbf{C_2},\mathscr{M}_2)$ be an $\mathscr{M}$-functor that translates and creates (direct) transformations as well as creates morphisms (see Theorem 1). Then AS_1 is locally confluent for all transformation spans $H_1\overset{\rho_1,m_1}{\Longleftarrow}G\overset{\rho_2,m_2}{\Longrightarrow}H_2$ iff AS_2 is locally confluent for all translated transformation spans $\mathscr{F}(H_1)\overset{\mathscr{F}(\rho_1),\mathscr{F}(m_1)}{\Longleftarrow}\mathscr{F}(G)\overset{\mathscr{F}(\rho_2),\mathscr{F}(m_2)}{\Longrightarrow}\mathscr{F}(H_2)$.

Proof.

1. **(Translation):** Assume local confluence of $H_1\overset{\rho_1,m_1}{\Longleftarrow}G\overset{\rho_2,m_2}{\Longrightarrow}H_2$ in AS_1. Then there exist an object X and transformations $H_1\overset{*}{\Rightarrow}X$, $H_2\overset{*}{\Rightarrow}X$ via P. Due to the assumption that $\mathscr{F}$ translates (direct) transformations, there exist the object $X'=\mathscr{F}(X)$ and transformations $\mathscr{F}(H_1)\overset{*}{\Rightarrow}\mathscr{F}(X)$, $\mathscr{F}(H_2)\overset{*}{\Rightarrow}\mathscr{F}(X)$ via $\mathscr{F}(P)$. Hence, the translated transformation span $\mathscr{F}(H_1)\overset{\mathscr{F}(\rho_1),\mathscr{F}(m_1)}{\Longleftarrow}\mathscr{F}(G)\overset{\mathscr{F}(\rho_2),\mathscr{F}(m_2)}{\Longrightarrow}\mathscr{F}(H_2)$ is locally confluent in AS_2.
2. **(Creation):** Assume local confluence 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. Then there is an object X' and transformations $\mathscr{F}(H_1)\overset{*}{\Rightarrow}X'$, $\mathscr{F}(H_2)\overset{*}{\Rightarrow}X'$ via $\mathscr{F}(P)$. Due to the assumption that $\mathscr{F}$ creates (direct) transformations, there exist objects X_1,X_2 and transformations $H_1\overset{*}{\Rightarrow}X_1$, $H_2\overset{*}{\Rightarrow}X_2$ via P with $\mathscr{F}(X_1)\cong X'\cong\mathscr{F}(X_2)$. Hence, $\mathscr{F}(X_1)\cong\mathscr{F}(X_2)$ and the assumption that $\mathscr{F}$ creates morphisms and hence also isomorphisms implies that $X_1\cong X_2$. We get that $H_1\overset{\rho_1,m_1}{\Longleftarrow}G\overset{\rho_2,m_2}{\Longrightarrow}H_2$ is locally confluent in AS_1. □

A well-known approach for the verification of local confluence is the analysis of critical pairs. A critical pair $P_1\overset{\rho_1,o_1}{\Longleftarrow}K\overset{\rho_2,o_2}{\Longrightarrow}P_2$ is a pair of parallel dependent transformations with a minimal overlapping K of the left sides of the rules.

Definition 4 ($\mathscr{F}$-Reachable Critical Pair)
Given an $\mathscr{M}$-functor $\mathscr{F}:(\mathbf{C_1},\mathscr{M}_1)\to(\mathbf{C_2},\mathscr{M}_2)$. An $\mathscr{F}$-reachable critical pair of productions $\mathscr{F}(\rho_1)$ and $\mathscr{F}(\rho_2)$ is a critical pair in AS_2 of the form

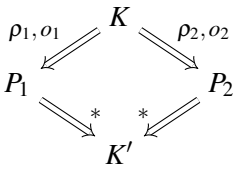


where all morphisms of type $\mathscr{F}(A) \to \mathscr{F}(B)$ are of the form $\mathscr{F}(f)$ for some morphism $f : A \to B$.

In the following, we assume that $\mathscr{F}$ is compatible with pair factorization, where pair factorization intuitively means that for every pair of objects there is a smallest overlapping embedded into a given context (see [EEPT06]). For details concerning compatibility with pair factorization see also Definition 6 in [MEE12].

Another important well-known result that we use in our approach is the lemma *Completeness of Critical Pairs* [EEPT06]. This lemma states that for each pair of parallel dependent direct transformations, we have a critical pair that can be embedded into the given parallel dependent transformation span.

Furthermore, we use the Local Confluence Theorem [EEPT06] to analyze whether a given $\mathscr{M}$-adhesive transformation system is locally confluent. This is the case, if all critical pairs $P_1 \overset{\rho_1,o_1}{\Longleftarrow} K \overset{\rho_2,o_2}{\Longrightarrow} P_2$ of the given transformation system are strictly confluent. Strictness means intuitively that the largest substructure of K that is preserved by the critical pair is also preserved by the merging transformation steps $P_1 \overset{*}{\Rightarrow} K'$ and $P_2 \overset{*}{\Rightarrow} K'$ (see the diagram to the right).



The following proposition describes in which case a pair of translated transformations is locally confluent. The detailed proof of Proposition 1 is given in [MEE12]. It is based on the fact that for each translated pair of parallel dependent transformations, we can construct the corresponding embedded $\mathscr{F}$-reachable critical pair, which is by assumption strictly confluent.

Proposition 1 (Local Confluence of a Translated Transformation Span)
Given $\mathscr{M}$-adhesive transformation systems $AS_1 = (\mathbf{C_1}, \mathscr{M}_1, P)$, $AS_2 = (\mathbf{C_2}, \mathscr{M}_2, \mathscr{F}(P))$ and an $\mathscr{M}$-functor $\mathscr{F} : (\mathbf{C_1}, \mathscr{M}_1) \to (\mathbf{C_2}, \mathscr{M}_2)$ that creates (direct) transformations and morphisms (see Theorem 1) and is compatible with pair factorization.
Then, a translated transformation span $\mathscr{F}(H_1) \overset{\mathscr{F}(\rho_1),\mathscr{F}(m_1)}{\Longleftarrow} \mathscr{F}(G) \overset{\mathscr{F}(\rho_2),\mathscr{F}(m_2)}{\Longrightarrow} \mathscr{F}(H_2)$ is locally confluent if all $\mathscr{F}$-reachable critical pairs of $\mathscr{F}(\rho_1)$ and $\mathscr{F}(\rho_2)$ in AS_2 are strictly confluent.

Now we summarize the results concerning the creation of local confluence based on $\mathscr{F}$-reachable critical pairs.

Theorem 3 (Creation of Local Confluence Based on $\mathscr{F}$-Reachable Critical Pairs)
Given an $\mathscr{M}$-functor $\mathscr{F} : (\mathbf{C_1}, \mathscr{M}_1) \to (\mathbf{C_2}, \mathscr{M}_2)$ with the assumptions of Proposition 1. An $\mathscr{M}$-adhesive transformation system AS_1 is locally confluent for all transformation spans $H_1 \overset{\rho_1,m_1}{\Longleftarrow} G \overset{\rho_2,m_2}{\Longrightarrow} H_2$ if all $\mathscr{F}$-reachable critical pairs of $\mathscr{F}(\rho_1)$ and $\mathscr{F}(\rho_2)$ in AS_2 are strictly confluent.

Proof.
The strict confluence of all $\mathscr{F}$-reachable critical pairs of $\mathscr{F}(\rho_1)$ and $\mathscr{F}(\rho_2)$ implies that AS_2 is locally confluent for all translated transformation spans $\mathscr{F}(H_1) \overset{\mathscr{F}(\rho_1),\mathscr{F}(m_1)}{\Longleftarrow} \mathscr{F}(G) \overset{\mathscr{F}(\rho_2),\mathscr{F}(m_2)}{\Longrightarrow} \mathscr{F}(H_2)$ by Proposition 1. This implies the local confluence of AS_1 for all transformation spans $H_1 \overset{\rho_1,m_1}{\Longleftarrow} G \overset{\rho_2,m_2}{\Longrightarrow} H_2$ by Theorem 2. □

Application to Petri Net and Graph Transformation Systems

The functor $\mathscr{F}$: **PTINet**$|_{\mathscr{M}_1} \to$ **AGraphs**$_{\mathbf{PNTG}}|_{\mathscr{M}_2}$ described in Section 3 is compatible with pair factorization, and hence satisfies the requirements of Theorem 3 (see Proposition 2 in [MEE12]), such that we have creation of local confluence based on critical pairs. Hence, by application of Theorem 3, a PTI net transformation system (N,P) with a PTI net N and a set of productions P is locally confluent, if all $\mathscr{F}$-reachable critical pairs of corresponding graph rules $\mathscr{F}(\rho_1)$ and $\mathscr{F}(\rho_2)$ are strictly confluent with $\rho_1,\rho_2 \in P$.

Example 1 (Mobile Dining Philosophers)
Let us consider a slight extension of the well-known Dining Philosophers *model, where mobile philosophers now may also leave or join a table (*MoDiPhi*). The firing of the PTI net transitions models the traditional behavior of the philosophers, switching between thinking and eating (see the PTI net in the left part of Figure 4 modelling five philosophers sitting at one table).*

The additional structural changes of the net occurring when a philosopher joins or leaves a table are modelled by the set $P_{MoDiPhi} = \{JoinTable, LeaveTable\}$ *of PTI net transformation rules. The right part of Figure 4 shows the PTI net transformation rule* JoinTable *for reconfiguring the table when another philosopher joins it*[1]*. The second transformation rule* LeaveTable *(not shown) is inverse to rule* JoinTable*, i.e., left-hand and right-hand sides are interchanged. The dia-*

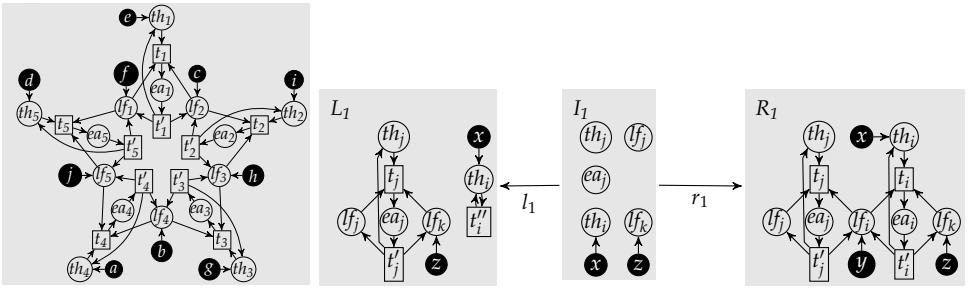


Figure 4: PTI net (left) and PTI net rule *JoinTable* (right) of transformation system *MoDiPhi*

gram in Figure 5 shows the strict confluence of the $\mathscr{F}$*-reachable critical pair of the* $\mathscr{F}$*-translated graph rules* $(\mathscr{F}(JoinTable), \mathscr{F}(JoinTable))$[2]*. The spans* $\mathscr{F}(P_1) \xleftarrow{\mathscr{F}(w_1)} \mathscr{F}(N_1) \xrightarrow{\mathscr{F}(v_1)} \mathscr{F}(K)$ *and* $\mathscr{F}(K) \xleftarrow{\mathscr{F}(v_2)} \mathscr{F}(N_2) \xrightarrow{\mathscr{F}(w_2)} \mathscr{F}(P_2)$ *in the upper half of the diagram denote the two conflicting rule applications of the rule* $\mathscr{F}(JoinTable)$*, where one rule application deletes the two transitions that are used by the other rule application and creates new ones. Graph* $\mathscr{F}(N)$ *in the center is the largest substructure of* $\mathscr{F}(K)$ *that is preserved by the critical pair. In the lower half of the diagram, we can see how the two direct graph transformations* $\mathscr{F}(K) \Rightarrow \mathscr{F}(P_1)$ *and* $\mathscr{F}(K) \Rightarrow \mathscr{F}(P_2)$ *can be merged again by applying once more the rule* $\mathscr{F}(JoinTable)$ *at an adequate match to the graphs* $\mathscr{F}(P_1)$ *and* $\mathscr{F}(P_2)$*. Moreover, the strictness condition is satisfied because* $\mathscr{F}(N)$ *is also preserved by the merging transformations* $\mathscr{F}(P_1) \Rightarrow \mathscr{F}(K'')$ *and* $\mathscr{F}(P_2) \Rightarrow \mathscr{F}(K'')$*.*

As we can show strict confluence also for all other $\mathscr{F}$*-reachable critical pairs of our example,*

[1] Note that the token inscriptions x,y,z are not variables but identifiers indicating the rule morphisms.
[2] A screenshot of the AGG analysis tool showing this critical pair can be found in [MEE12], Figure 6.

we obtain by Theorem 3 that the PTI net transformation system MoDiPhi *is locally confluent.*

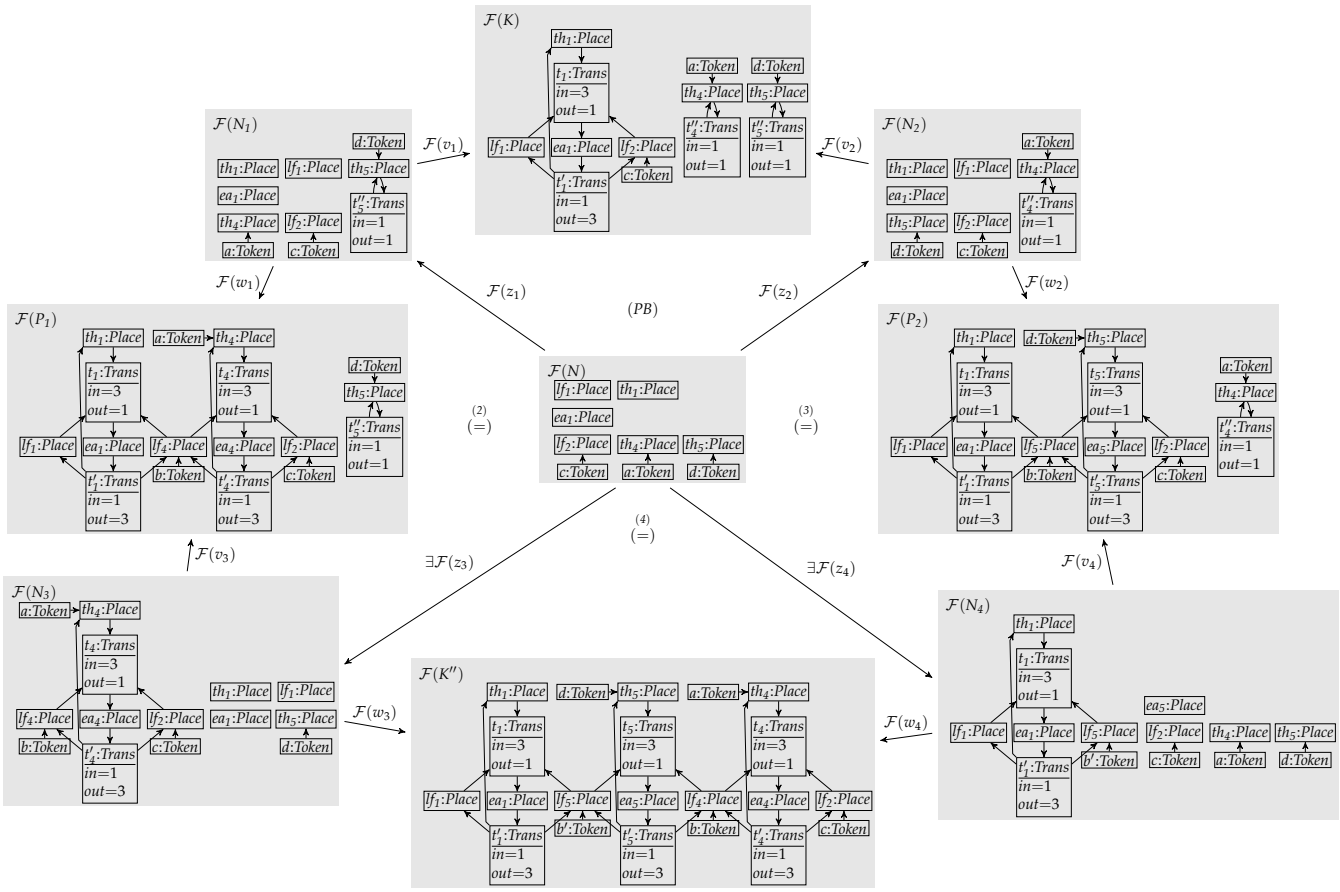


Figure 5: Strict confluence of the $\mathscr{F}$-reachable critical pair of the $\mathscr{F}$-translated graph rules $(\mathscr{F}(JoinTable), \mathscr{F}(JoinTable))$

5 $\mathscr{F}$-Transfer of Termination and Functional Behavior

A transformation $G \overset{*}{\Rightarrow} H$ is called terminating if no production is applicable to H anymore. A formal definition of termination of an $\mathscr{M}$-adhesive transformation system is as follows.

Definition 5 (Termination and $\mathscr{F}$-Termination of an $\mathscr{M}$-Adhesive Transformation System)

- An $\mathscr{M}$-adhesive transformation system $(\mathbf{C_1}, \mathscr{M}_1, P)$ is called *terminating* if there is no infinite sequence $G_0 \overset{\rho_1,m_1}{\Longrightarrow} G_1 \overset{\rho_2,m_2}{\Longrightarrow} G_2 \overset{\rho_3,m_3}{\Longrightarrow} \dots$ with $\rho_1, \rho_2, \rho_3, \dots \in P$ and matches $m_1, m_2, m_3, \dots$
- Given an $\mathscr{M}$-functor $\mathscr{F} : (\mathbf{C_1}, \mathscr{M}_1) \to (\mathbf{C_2}, \mathscr{M}_2)$, a translated $\mathscr{M}$-adhesive transformation system $(\mathbf{C_2}, \mathscr{M}_2, \mathscr{F}(P))$ is called *$\mathscr{F}$-terminating* if there is no infinite sequence $\mathscr{F}(G_0) \overset{\mathscr{F}(\rho_1),m'_1}{\Longrightarrow} G'_1 \overset{\mathscr{F}(\rho_2),m'_2}{\Longrightarrow} G'_2 \overset{\mathscr{F}(\rho_3),m'_3}{\Longrightarrow} \dots$ with $\mathscr{F}(\rho_1), \mathscr{F}(\rho_2), \mathscr{F}(\rho_3), \dots \in \mathscr{F}(P)$ and matches $m'_1, m'_2, m'_3, \dots$

$\mathscr{M}$-functors transfer termination according to the following theorem.

Theorem 4 ($\mathscr{F}$-Transfer of Termination)
Given $\mathscr{M}$-adhesive transformation systems $AS_1 = (\mathbf{C_1}, \mathscr{M}_1, P)$, $AS_2 = (\mathbf{C_2}, \mathscr{M}_2, \mathscr{F}(P))$ and an $\mathscr{M}$-functor $\mathscr{F} : (\mathbf{C_1}, \mathscr{M}_1) \to (\mathbf{C_2}, \mathscr{M}_2)$ that translates and creates (direct) transformations (see Theorem 1). Then, an $\mathscr{M}$-adhesive transformation system AS_1 is terminating iff the corresponding translated $\mathscr{M}$-adhesive transformation system AS_2 is $\mathscr{F}$-terminating.

Proof.
By contraposition: Given a nonterminating sequence in AS_2 (see diagram to the right). It is possible to generate stepwise a nonterminating sequence in AS_1 by application of the assumption that the given $\mathscr{M}$-functor creates direct transformations. Analogously, given a nonterminating sequence in AS_1, it is possible to generate stepwise a nonterminating sequence in AS_2 by application of the assumption that the given $\mathscr{M}$-functor translates direct transformations.

$$\begin{array}{ccc}
G_0 & \overset{\mathscr{F}}{\dashrightarrow} & \mathscr{F}(G_0) \\
\Downarrow{\scriptstyle \rho_1, m_1} & & \Downarrow{\scriptstyle \mathscr{F}(\rho_1), m_1' = \mathscr{F}(m_1)} \\
G_1 & \overset{\mathscr{F}}{\dashrightarrow} & G_1' = \mathscr{F}(G_1) \\
\Downarrow{\scriptstyle \rho_2, m_2} & & \Downarrow{\scriptstyle \mathscr{F}(\rho_2), m_2' = \mathscr{F}(m_2)} \\
G_2 & \overset{\mathscr{F}}{\dashrightarrow} & G_2' = \mathscr{F}(G_2) \\
\Downarrow{\scriptstyle \rho_3, m_3} & & \Downarrow{\scriptstyle \mathscr{F}(\rho_3), m_3' = \mathscr{F}(m_3)} \\
\vdots & & \vdots
\end{array}$$

□

According to [EEPT06] a locally confluent and terminating transformation system AS is confluent and has functional behavior in the following sense: For each object G there is an object H together with a terminating transformation $G \overset{*}{\Rightarrow} H$ in AS and H is unique up to isomorphism. Moreover, each pair of transformations $G \overset{*}{\Rightarrow} H_1$ and $G \overset{*}{\Rightarrow} H_2$ can be extended to terminating transformations $G \overset{*}{\Rightarrow} H_1 \overset{*}{\Rightarrow} H$ and $G \overset{*}{\Rightarrow} H_2 \overset{*}{\Rightarrow} H$ with the same object H.

Corollary 1 ($\mathscr{F}$-Transfer of Local Confluence, Termination and Functional Behavior)
Let $\mathscr{F} : (\mathbf{C_1}, \mathscr{M}_1) \to (\mathbf{C_2}, \mathscr{M}_2)$ be an $\mathscr{M}$-functor between $\mathscr{M}$-adhesive transformation systems AS_1 and AS_2 with the assumptions of Theorem 2. AS_1 is locally confluent and terminating iff AS_2 is locally confluent and $\mathscr{F}$-terminating. Moreover, AS_1 has functional behavior if AS_2 is locally confluent and $\mathscr{F}$-terminating.

Proof. This follows directly from Theorem 2, Theorem 4 and [EEPT06]. □

Application to Petri Net and Graph Transformation Systems

The functor $\mathscr{F} : \mathbf{PTINet}|_{\mathscr{M}_1} \to \mathbf{AGraphs_{PNTG}}|_{\mathscr{M}_2}$ described in Section 3 satisfies already the assumptions of Theorem 4 and Corollary 1. Hence, by application of Theorem 4 and Corollary 1, we have $\mathscr{F}$-transfer of local confluence, termination and functional behavior. Moreover, Theorem 3 allows us to create local confluence of a Petri net transformation system by critical pair analysis of the corresponding graph transformation system.

Example 2
Our MoDiPhi *PTI net transformation system with rule set $P_{MoDiPhi} = \{JoinTable, LeaveTable\}$ (see Example 1) is obviously not terminating because the two rules are inverse to each other. Considering a restricted rule set containing only rule LeaveTable, the transformation system becomes terminating because rule LeaveTable reduces the size (number of places, transitions and*

tokens) of the PTI net it is applied to with each rule application. Constructing the $\mathscr{F}$-translated rule $\mathscr{F}(LeaveTable)$, we obviously get an $\mathscr{F}$-terminating graph transformation system, because also $\mathscr{F}(LeaveTable)$ reduces the size of the graph it is applied to with each rule application.

6 Related Work and Conclusion

In our previous paper [MEE11] we have developed a general framework to establish a formal relationship between different $\mathscr{M}$-adhesive transformation systems. In Section 2 of the present paper we have reviewed the main result of [MEE11] showing under which conditions transformations can be translated and created between different $\mathscr{M}$-adhesive transformation systems. This result is based on suitable properties of $\mathscr{M}$-functors between the corresponding $\mathscr{M}$-adhesive categories. Especially we have constructed in [MEE11] and reviewed in Section 3 an $\mathscr{M}$-functor from Petri nets with individual tokens to typed attributed graphs, which is restricted to $\mathscr{M}$-morphisms and satisfies the adapted properties given at the end of Section 3. As main new results of this paper we have extended this framework to the transfer of local confluence, termination and functional behavior. This allows us to use the critical pair analysis of the AGG-tool for typed attributed graphs to analyze Petri net transformation systems. Furthermore, we have pointed out in Section 4 that the concrete restricted $\mathscr{M}$-functor constructed in [MEE11] fulfills the additional property (the compatibility of $\mathscr{F}$ with pair factorization), so that the main results from Section 4 and Section 5 can be applied. On this basis, a straightforward extension is to consider also firing steps as transformation rules in $\mathbf{AGraphs}_{\mathbf{PNTG}}$, and to analyze dependencies and conflicts between net transformations and firing steps using AGG.

In future work we will analyze how nested application conditions [HP05, EHL+10] can be handled in this framework in order to transfer critical pairs and local confluence of $\mathscr{M}$-adhesive transformation systems with nested application conditions.

Furthermore, the restricted $\mathscr{M}$-functor from Petri nets with individual tokens to typed attributed graphs can be considered to become an equivalence of categories between Petri nets with individual tokens and a suitable subcategory of typed attributed graphs. In future work we will analyze, whether it is easier to verify the required properties for the corresponding inclusion functor than for the original restricted $\mathscr{M}$-functor. Moreover, we will apply the general framework also to other $\mathscr{M}$-adhesive transformation systems.

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.

[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.

[EHL+10] H. Ehrig, A. Habel, L. Lambers, F. Orejas, U. Golas. Local Confluence for Rules with Nested Application Conditions. In Ehrig et al. (eds.), *Proceedings of Intern. Conf. on Graph Transformation*. LNCS 6372, pp. 330–345. Springer, 2010.

[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.

[HM10] K. Hoffmann, T. Modica. Formal Modeling of Communication Platforms using Reconfigurable Algebraic High-Level Nets. *ECEASST* 30:1–25, 2010.

[HP05] A. Habel, K.-H. Pennemann. Nested constraints and application conditions for high-level structures. In Kreowski et al. (eds.), *Formal Methods in Software and Systems Modeling*. LNCS 3393, pp. 294–308. Springer, 2005.

[LS04] S. Lack, P. Sobociński. Adhesive Categories. In *Proc. FOSSACS' 04*. 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. In Ermel and Hoffmann (eds.), *Proc. Int. Workshop on Petri Nets and Graph Transformation Systems*. Volume 40. ECEASST, 2011.

[MEE12] M. Maximova, H. Ehrig, C. Ermel. Transfer of Local Confluence and Termination between Petri Net and Graph Transformation Systems Based on $\mathscr{M}$-Functors: Extended Version. Technical report 2012/08, TU Berlin, 2012.
http://www.eecs.tu-berlin.de/menue/forschung/forschungsberichte/2012

[MGH11] T. Modica, K. Gabriel, K. Hoffmann. Transformation of Petri Nets with Individual Tokens. *ECEASST* 40, 2011.

[PEH07] J. Padberg, H. Ehrig, K. Hoffmann. Formal Modeling and Analysis of flexible Processes in Mobile Ad-Hoc Networks. *Bulletin of the EATCS* 91:128–132, 2007.

[PEHP08] U. Prange, H. Ehrig, K. Hoffman, J. Padberg. Transformations in Reconfigurable Place/Transition Systems. In *Concurrency, Graphs and Models*. LNCS 5065, pp. 96–113. Springer, 2008.

[RON07] TFS-Group, TU Berlin. Reconfigurable Object Nets Environment. 2007.
http://www.tfs.cs.tu-berlin.de/roneditor