Replace this file with `prentcsmacro.sty` for your meeting,
or with `entcsmacro.sty` for your meeting. Both can be
found at the ENTCS Macro Home Page.

Composition and Independence of High-Level Net Processes[1]

H. Ehrig, K. Hoffmann, K. Gabriel, J. Padberg

Institut für Softwaretechnik und Theoretische Informatik
Technische Universität Berlin
Germany

Abstract

Mobile ad-hoc networks (MANETs) are networks of mobile devices that communicate with each other via wireless links without relying on an underlying infrastructure. To model workflows in MANETs adequately a formal techniques is given by algebraic higher-order nets. For this modeling technique we here present a high-level net process semantics and results concerning composition and independence. Based on the notion of processes for low-level Petri nets we analyse in this paper high-level net processes defining the non-sequential behaviour of high-level nets. In contrast to taking low-level processes of the well known flattening construction for high-level nets our concept of high-level net processes preserves the high-level structure. The main results are the composition, equivalence and independence of high-level net processes under suitable conditions. Independence means that they can be composed in any order leading to equivalent high-level net processes which especially have the same input/output behaviour. All concepts and results are explained with a running example of a mobile ad-hoc network in the area of an university campus.

Keywords: Algebraic models, algebraic high-level nets, behavioural semantics, high-level net processes, mobility, analysis of nets, composition of processes, equivalence and independence of processes.

1 Introduction

From an abstract point of view mobile ad-hoc networks (MANETs) consist of mobile nodes which communicate with each other independently from a stable infrastructure, while the topology of the network constantly changes depending on the current position of the nodes and their availability. In our research project *Formal Modeling and Analysis of Flexible Processes in Mobile Ad-hoc Networks* we develop the modeling technique of algebraic higher-order nets. This enables the modeling of flexible workflows in MANETs and supports changes of the network topology and the subsequent transformation of workflows. Algebraic higher-order (AHO) nets are Petri nets with complex tokens, especially reconfigurable place/transition (P/T) nets in [6]. AHO-nets can be considered as a special case of algebraic high-level (AHL) nets. The main topic of this paper is to present a high-level process semantics for

[1] This work has been partly funded by the research project forMA$_l$NET (see `http://tfs.cs.tu-berlin.de/formalnet/`) of the German Research Council.

©2008 Published by Elsevier Science B. V.

AHL-nets in general, where the example in Section 2 is given as a MANET and is modeled by an AHO-net.

For low-level Petri nets it is well known that processes are essential to capture their non-sequential truly concurrent behaviour (see e.g. [9,14,1,7,13]). Processes for high-level nets are often defined as processes of the low-level net which is obtained from flatting the high-level net. In [2,5] we have defined high-level net processes for high-level nets based on a suitable notion of high-level occurrence nets which are defined independently of the flattening construction. The flattening of a high-level occurrence net is in general not a low-level occurrence net due to so called assignment conflicts in the high-level net. The essential idea is to generalise the concept of occurrence nets from the low-level to the high-level case. This means that the net structure of a high-level occurrence net has similar properties like a low-level occurrence net, i.e. unitarity, conflict freeness, and acyclicity. But we have to abandon the idea that an occurrence net captures essentially one concurrent computation. Instead, a high-level occurrence net and a high-level process are intended to capture a set of different concurrent computations corresponding to different input parameters of the process. In fact, high-level processes can be considered to have a set of initial markings for the input places of the corresponding occurrence net, whereas there is only one implicit initial marking of the input places for low-level occurrence nets.

In this paper we extend the notion of high-level net processes with initial markings by a set of corresponding instantiations. An instantiation is a subnet of the flattening defining one concurrent computation of the process. The advantage is that we fix for a given initial marking a complete firing sequence where each transition fires exactly once. The main ideas and results in this paper concern the composition of high-level net processes. In general the composition of high-level net processes is not a high-level net process, because the composition may contain forward and/or backward conflicts and also the partial order might be violated. Thus we state suitable conditions, so that the composition of high-level processes leads to a high-level process. We introduce the concept of equivalence of high-level net processes, where the net structures of these high-level net processes might be different, but they have especially the same input/output behaviour. Hence their concurrent computations are compared in the sense that they start and end up with the same marking, but even corresponding dependent transitions may be fired in a different order. In this context the main problem solved in this paper is to analyse the independence of high-level net processes, i.e. under which condition high-level processes can be composed in any order leading to equivalent processes.

The paper is organised as follows. In Section 2 we exemplarily explain the concepts and results of this paper using a mobile ad-hoc network in the area of an university campus. In Section 3 on the one hand we review the notions for high-level net processes and on the other hand we introduce the new notion of high-level net processes with instantiations. In Section 4 we present our main results concerning the composition, equivalence and independence of high-level net processes. Due to space limitation the definitions and theorems are given on an informal level, while the details can be found in the Appendix and in [4]. Finally we conclude with related work and some interesting aspects of future work in Section 5.

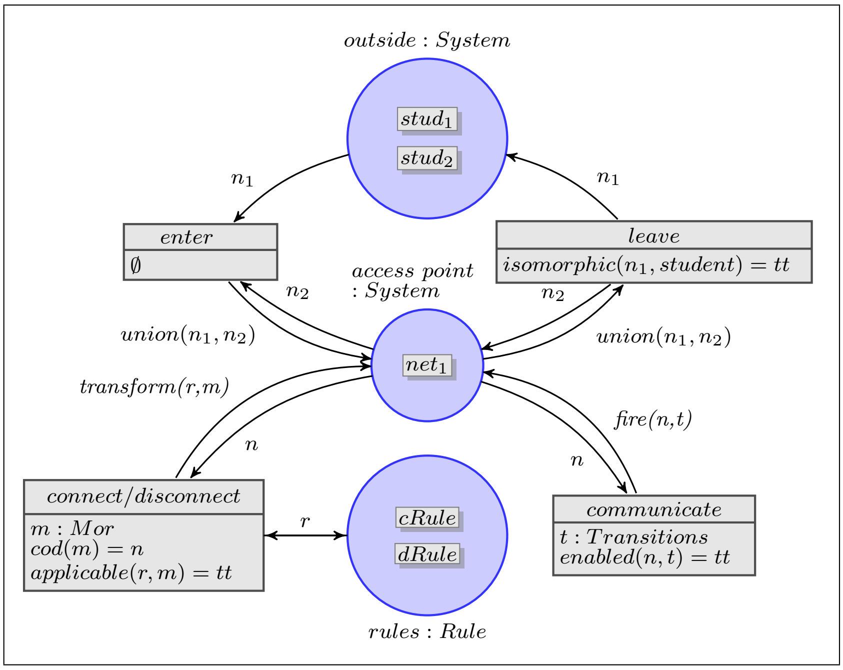


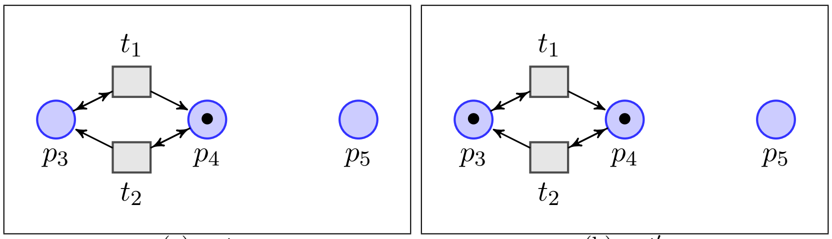
Fig. 1. AHO-net AN_{Campus}

2 Mobile Ad-Hoc Network on University Campus

In this section we introduce a simple example of a wireless network on a university campus and illustrate thereby the concepts in the following sections. As modeling technique we use algebraic higher-order (AHO) nets. AHO-nets are Petri nets with complex tokens, namely place/transition (P/T) nets and rules to support changes of the network topology. With a specific data type part (see A.1 in the Appendix) they can be considered as a special case of algebraic high-level nets.

The example models a network, where students can exchange their messages. For this reason two different locations are represented by the places *outside* and *access point* in the AHO-net AN_{Campus} in Fig. 1. The marking of the AHO-net shows the distribution of the students at different places. Initially there are two students outside the campus and three additional students are on the campus represented by the tokens $stud_1$, $stud_2$ and net_1 in Fig. 1. The mobility aspect of the students is modeled by transitions termed *enter* and *leave* in Fig. 1, while the static structure of the wireless network is changed by rule-based transformations using the rules *cRule* and *dRule*. Moreover the transition *communicate* realises the well known token game.

Subsequently we concentrate on the behaviour of the transitions *communicate* and *connect/disconnect*. On the left hand side of Fig. 2 the P/T-net net_1 of the current network is depicted, where two students, represented by the places p_3 and p_4, respectively, had established a communication structure to exchange message, while student p_5 is disconnected. The P/T-net net_1 is the token on the place *access point* in Fig. 1. To start the communication we use the transition *communicate* of the AHO-net in Fig. 1. First we give an assignment v_1 of the variables n and t in the environment of this transition and assign the network net_1 to the variable n and



(a) net_1 (b) net_1'

Fig. 2. Net tokens

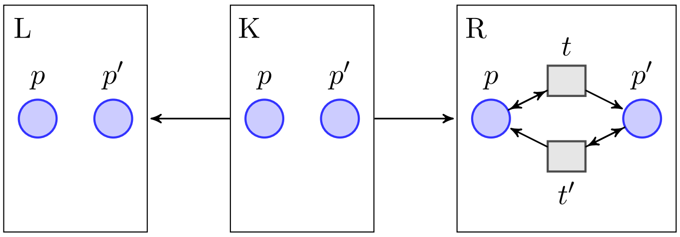
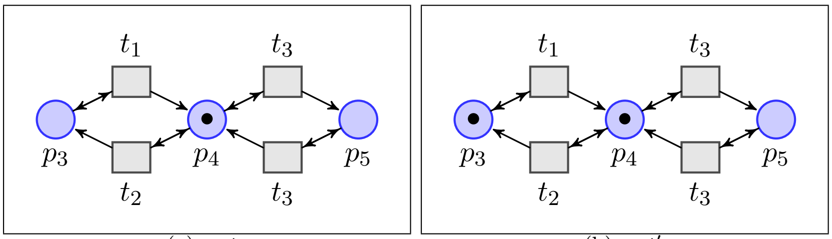


Fig. 3. Rule token *cRule*

the transition t_2 to the variable t. The firing condition checks that the student p_4 is able to send a message. This is modeled by an abstract black token on the place p_4. The evaluation of the net inscription $fire(n,t)$ realises the well-known token game by computing the follower marking of the P/T-net and so we obtain the new P/T-net net_1' depicted on the right hand side of Fig. 2, where the student p_3 has received the message.

Assume the student p_5 wants to enter the network in order to communicate with the other students. Formally, we apply the rule *cRule* in Fig. 3 that is a token on place *rules* in Fig. 1. In general a rule $r = (L \leftarrow K \rightarrow R)$ is given by three P/T-nets called left-hand side, interface, and right-hand side respectively and the application of a rule describes the replacement of the left-hand side by the right-hand side preserving the interface. The connection between the student p_4 and p_5 is established by firing the transition *connect/disconnect* in the AHO-net in Fig. 1 using the following assignment of the variables n, r and m given in the net inscriptions of this transition: $v_2'(n) = net_1'$, $v_2'(r) = cRule$ and $v_2'(m) = g$, where g is a P/T-net morphism which identifies the left hand side of the rule *cRule* in the network $net1'$. In our case the match g maps p to p_4 and p' to p_5. The firing conditions of the transition *connect/disconnect* makes sure that on the one hand the rule is applied to the P/T-net net_1' and on the other hand the rule is applicable with match g to this P/T-net. Finally we evaluate the term *transform(r,m)* yielding the direct transformation leading to the P/T-net net_2' on the right hand side in Fig. 4. The effect of firing the transition *connect/disconnect* in the AHO-net in Fig. 1 with assignments of variables as discussed above is the removal of the P/T-net net_1' from place *access point* and adding the P/T-net net_2' to the place *access point*.

Vice versa student p_5 can enter the network net_1 by the application of the rule *cRule* to the network net_1 resulting in the network net_2 on the left hand side of Fig. 4 and afterwards students p_3 and p_4 start their communication leading to net net_2' in Fig. 4. Formally this is achieved by firing the corresponding transitions in the



(a) net_2 (b) net'_2

Fig. 4. Net tokens after rule application

AHO-net in Fig. 1 in opposite order with suitable variable assignments v_2 and v'_1.

Summarising, we have explained two different firing sequences of the AHO-net in Fig. 1. The first one starts with the token firing of net_1 leading to the P/T-net net'_1 (see Fig. 2) before student p_5 enters the network (see right hand side of Fig. 4). The second one begins by introducing student p_5 into the network net_1 resulting in the network net_2 (see left hand side of Fig. 4) before students p_3 and p_4 exchange the message (see right hand side of Fig. 4).

As for processes for low-level nets we want to consider now processes for AHL-nets of which AHO-nets are a special case. These AHL-processes are based on AHL-occurrence nets. In fact the two firing sequences considered above correspond to different AHL-occurrence nets. An AHL-occurrence net is similar to a low-level occurrence net concerning unitarity, conflict freeness, and acyclicity. However, in contrast to a low-level occurrence net an AHL-occurrence net realises more than one concurrent computation depending on different initial markings and variable assignments. So we consider AHL-occurrence nets with a set of initial markings of the input places and corresponding instantiations of places and transitions by data and consistent variable assignments, respectively. For details see Section 3.

In our example we get the two AHL-occurrence nets K and K' on the left hand sides of Fig. 5 and Fig. 6 where the initial marking of the input places is given by the P/T-net net_1 and the rule *cRule*. The corresponding instantiations L_{init} and $L_{init'}$ on the right hand sides of Fig. 5 and Fig. 6 fix the two different firing sequences described above. Note that the AHL-occurrence nets K and K' have the same input and output places. But due to the firing of the transitions *communicate* and *connect/disconnect* in opposite order we use the different variable evaluations v_1 and v'_2 in L_{init} and v_2 and v'_1 in $L_{init'}$. Nevertheless the two different firing sequences end up with the same marking of the output places where the student p_5 is connected to the other students and the student p_3 received the message from student p_4 as depicted in the P/T-net net'_2 on the left hand side of Fig. 4. We show in Section 4 that there are basic AHL-occurrence nets K_1 and K_2, such that K and K' can be obtained as composition in different order of K_1 and K_2. This allows considering the corresponding processes of K and K' with instantiations as equivalent processes of the AHO-net AN_{Campus} in Fig. 1.

3 Algebraic High-Level Net Processes

In this section we review algebraic high-level nets and give a definition of high-level processes [2,5] based on high-level occurrence nets. Moreover we extend this

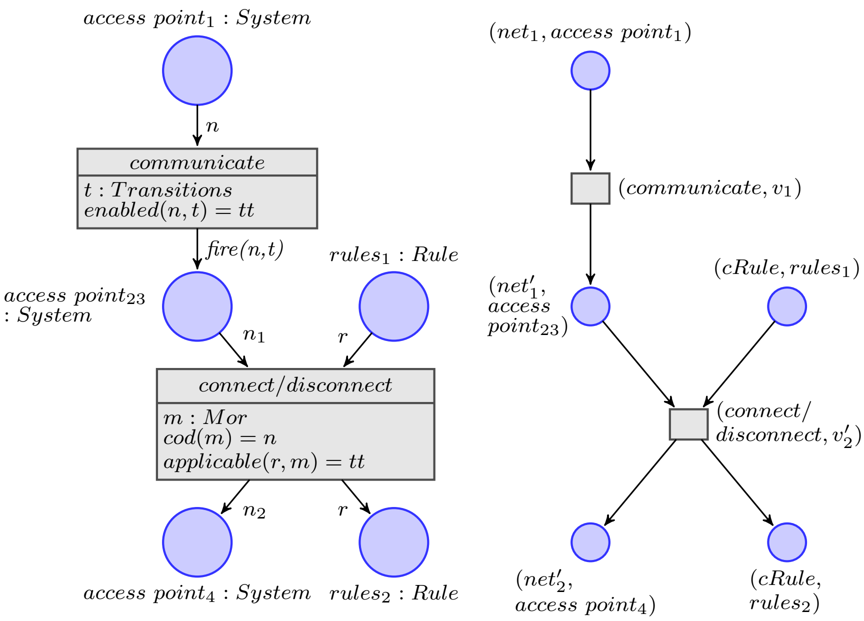


Fig. 5. AHL-occurrence net K with instantiation L_{init}

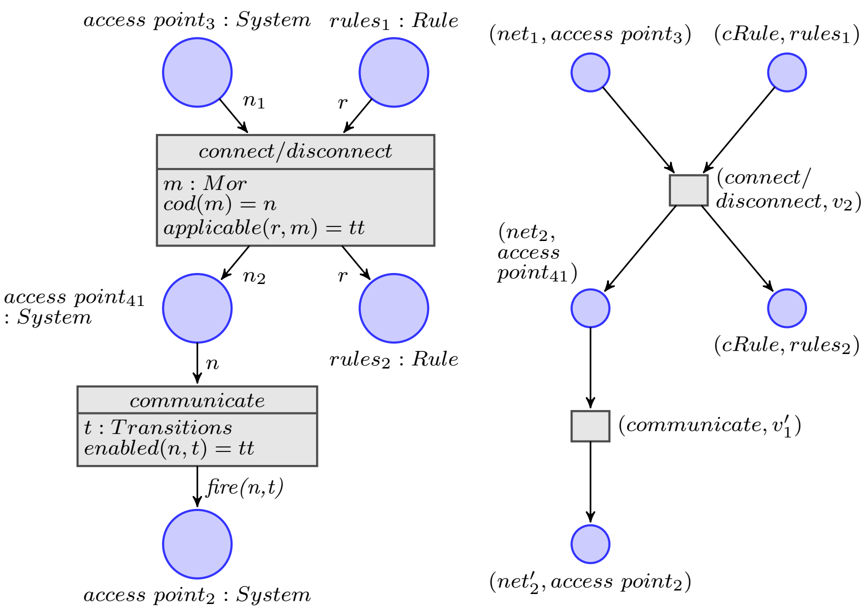


Fig. 6. AHL-occurrence net K' with instantiation $L_{init'}$

definition by a suitable notation of instantiations for each initial marking.

We use the algebraic notion of place/transition nets as in [12]. A place/transition (P/T) net $N = (P, T, pre, post)$ is given by the set of places P, the set of transitions T, and two mappings $pre, post : T \to P^{\oplus}$, the pre-domain and the post-domain, where $P^{\oplus}$ is the free commutative monoid over P that can also be considered as the set of finite multisets over P. Then we use simple homomorphisms that are generated over the set of places. These morphisms map places to places and transitions to transitions. A P/T-net morphism $f : N_1 \to N_2$ between two P/T-nets N_1 and N_2 is given by $f = (f_P, f_T)$ with functions $f_P : P_1 \to P_2$ and $f_T : T_1 \to T_2$

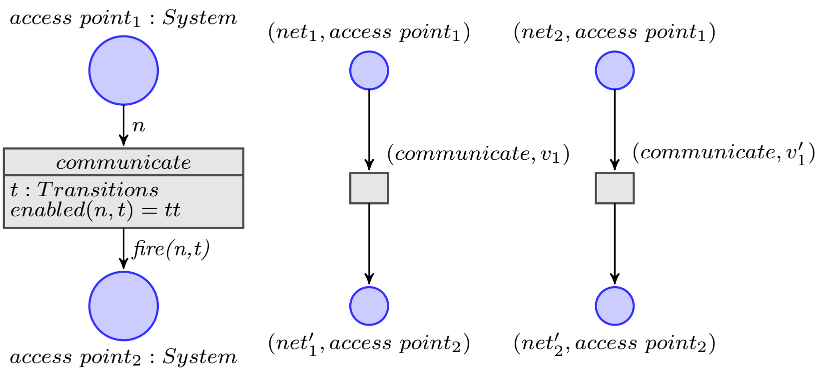


Fig. 7. AHL-occurrence net K_1 with instantiations L_{init_1} and $L_{init'_1}$

preserving the pre-domain as well as the post-domain of a transition. Note that the extension $f_P^\oplus : P_1^\oplus \to P_2^\oplus$ of $f_P : P_1 \to P_2$ is defined by $f_P^\oplus(\sum_{i=1}^n k_i \cdot p_i) = \sum_{i=1}^n k_i \cdot f_P(p_i)$. Examples of P/T nets with markings are given in Fig. 2 and Fig. 4.

An algebraic high-level (AHL) net [2,5] is essentially a P/T-net together with a suitable data type part given by an an algebraic specification and a corresponding algebra. An AHL-net morphism $f : AN_1 \to AN_2$ between two AHL-nets AN_1 and AN_2 is more or less analogously defined as a P/T-net morphism but in addition the arc inscriptions and firing conditions have to be preserved (see Def. B.3 in the Appendix). An example of an AHL-net is given in in Fig. 1, where the AHO-net AN_{Campus} is a special case of an AHL-net with specific data type part consisting of P/T-nets and rules as defined in the signature *HLRN-System-SIG* and algebra A according to [10] (see Def. A.1 in the Appendix).

Now we introduce high-level occurrence nets and high-level net processes according to [2,5], called AHL-occurrence net and AHL-process respectively. The net structure of a high-level occurrence net (see Def. B.5 in the Appendix) has similar properties like a low-level occurrence net. An AHL-occurrence net K is an AHL-net such that the pre- and post domain of its transitions are sets rather than multisets and the arc-inscriptions are unary. Moreover there are no forward and backward conflicts, the partial order given by the flow relation is irreflexive and for each element in the partial order the set of its predecessors is finite.

In contrast to low-level occurrence nets a high-level occurrence net captures a set of different concurrent computations due to different initial markings. In fact, high-level occurrence nets have a set of initial markings for the input places, whereas there is only one implicit initial marking of the input places for low-level occurrence nets. The notion of high-level net processes generalises the one of low-level net processes. An AHL-process of a AHL-net AN is a AHL-net morphism $p : K \to AN$ where K is an AHL-occurrence net described above (see Def. B.6). Examples of high-level and low-level occurrence nets are given by K and K' (resp. L_{init} and $L_{init'}$) in Fig. 5 and Fig. 6.

Because in general there exist different meaningful markings of an AHL-occurrence net K, we extend this notion by a set of initial markings $INIT$ of the input places of K (see Def. B.7) and a set of corresponding instantiations INS for each initial marking. An instantiation (see Def. B.11) defines one concurrent execution

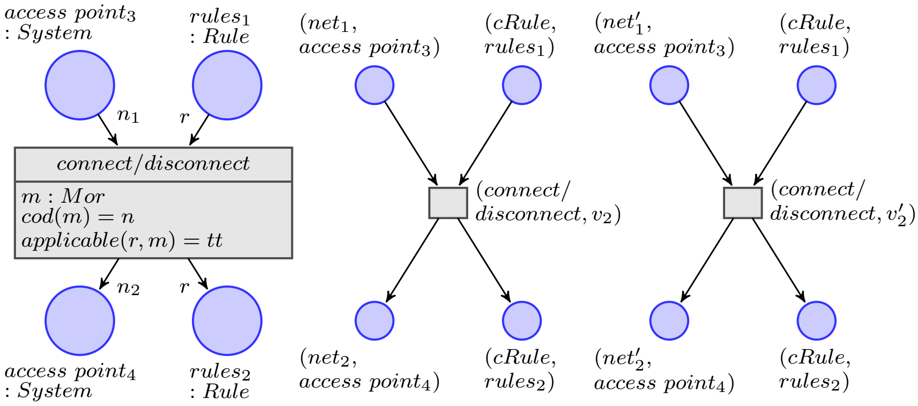


Fig. 8. AHL-occurrence net K_2 with instantiations L_{init_2} and $L_{init'_2}$

of a marked high-level occurrence net. In more detail an instantiation is a subnet of the flattening of the AHL-occurrence net corresponding to the initial marking. The flattening $Flat(AN)$ of an AHL-net AN results in a corresponding low-level net N, where the data type part (SP, A) and the firing behaviour of the AHL-net AN is encoded in the sets of places and transitions of N. Thus the flattening $Flat(AN)$ leads to an infinite P/T-net N if the algebra A is infinite (see Def. B.8). In contrast the skeleton $Skel(AN)$ of an AHL-net AN is a low-level net N' preserving the net structure of the AHL-net but dropping the net inscriptions (see Def. B.9). While there is a bijective correspondence between firing sequences of the AHL-net and firing sequences of its flattening, each firing of the AHL-net implies a firing of the skeleton, but not vice versa. In [2,5] it is shown that for a marked AHL-occurrence net there exists a complete firing sequence if and only if there exists an instantiation which net structure is isomorphic to the AHL-occurrence net and has the initial marking of the AHL-occurrence net as input places.

Note that in general for a given initial marking of an AHL-occurrence net there exists more than one instantiation. Thus different firing sequences result in different markings of the output places of the AHL-occurrence net. For this reason we fix exactly one instantiation for a given initial marking, i.e. one concurrent execution of the marked AHL-occurrence net. Thus an AHL-occurrence net with instantiations $KI = (K, INIT, INS)$ is given by an AHL-occurrence net K, a set of initial markings $INIT$ and a set of corresponding instantiations INS (see Def. B.12). An instantiated AHL-process of an AHL-net AN is defined by KI together with an AHL-net morphism $mp : K \to AN$ (see Def. B.13).

As an example the AHL-occurrence net with instantiations $KI_1 = (K_1, INIT_1, INS_1)$ is depicted in Fig. 7 according to the discussion in Section 2. The AHL-occurrence net K_1 is the AHL-net on the left hand side of Fig. 7. There are two different initial markings, i.e the set of initial markings is defined by $INIT_1 = \{(net_1, access\ point_1), (net_2, access\ point_1)\}$ and the set of the two instantiations on the right hand side of Fig. 7 by $INS_1 = \{L_{init_1}, L_{init'_1}\}$. The instantiated AHL-process is the AHL-occurrence net with instantiations KI_1 together with the AHL-net morphism $mp_1 : K_1 \to AN_{Campus}$. The morphism mp_1 consists of the inclusion of the transition *communicate*, while the places *access point*$_1$ and *access point*$_2$ are mapped to the place *access point* of the AHL-net AN_{Campus} in Fig. 1.

Further examples are given in Fig. 5 and Fig. 6, where we have the AHL-occurrence net K with one instantiation $KI = (K, \{init\}, \{L_{init}\})$ and the AHL-occurrence net K' with instantiation KI' with corresponding morphisms $mp : K \to AN_{Campus}$ and $mp' : K' \to AN_{Campus}$.

4 Composition, Equivalence and Independence of Algebraic High-Level Net Processes

In this section we define the composition of AHL-occurrence nets and AHL-processes with instantiations and introduce the concept of equivalence and independence of high-level net processes. The main result states that two independent high-level net processes can be composed in any order leading to equivalent high-level net processes which especially have the same input/output behaviour.

The composition of two AHL-occurrence nets K_1 and K_2 is defined by merging some of the output places of K_1 with some of the input places of K_2, so that the result of the composition is an AHL-occurrence net. In general this is not necessarily true, because the result of gluing two high-level occurrence nets may contain forward and/or backward conflicts and may violate the partial order.

Result 1 (Composition of AHL-Occurrence Nets (see Thm. C.2)) *The composition of two AHL-occurrence nets K_1 and K_2 given by merging some of the output places of K_1 with some of the input places of K_2 results in an AHL-occurrence net K.*

As mentioned above instantiations define one concurrent execution of a marked AHL-occurrence net. To generalise the composition given above to the composition of instantiations we have to check that the data elements of the merged output places of K_1 and input places of K_2 are coincident in the corresponding instantiations (see Def. C.3). In this case the composition of some of the instantiations of KI_1 with some of the instantiations of KI_2 leads to suitable instantiations of the AHL-occurrence net K that is the result of the composition of the two AHL-occurrence nets K_1 and K_2.

The AHL-occurrence net with instantiations $KI_2 = (K_2, INIT_2, INS_2)$ is given in Fig. 8. The sequential composition of K_1 (see Fig. 7) and K_2 is defined by merging the output place *access point*$_2$ of K_1 and the input place *access point*$_3$ of K_2 leading to the AHL-occurrence net K (see Fig. 5). The corresponding instantiations L_{init_1} in Fig. 7 and $L_{init'_2}$ in Fig. 8 can be composed analogously to the instantiation L_{init} in Fig. 5. Note that L_{init_1} and $L_{init'_2}$ are composable, because they have the same data element net'_1 in the input and output place, respectively.

Result 2 (Composition of AHL-Occurrence Nets with Instantiations (see Thm. C.4)) *The composition of two AHL-occurrence nets with instantiations $KI_1 = (K_1, INIT_1, INS_1)$ and $KI_2 = (K_2, INIT_2, INS_2)$ with composable K_1, K_2 and INS_1, INS_2, respectively, is an AHL-occurrence net with instantiations $KI = (K, INIT, INS)$, where K is the composition of K_1 and K_2 and INS is the corresponding composition of INS_1 and INS_2. The set of initial markings $INIT$ is derived by the input places of the instantiations in INS.*

Given the two basic AHL-occurrence nets with instantiations KI_1 and KI_2, then the composition of KI_1 and KI_2 results in the AHL-occurrence net with instantiation KI (see Fig. 5), while the opposite composition of KI_2 and KI_1 is the AHL-occurrence net with instantiation KI' (see Fig. 6).

The following result generalizes the composition to AHL-Processes with instantiations where in addition the AHL-net morphisms have to be taken into account.

Result 3 (Composition of AHL-Processes with Instantiations (see Thm. C.6)) *Let $KI_1 = (K_1, INIT_1, INS_1)$ and $KI_2 = (K_2, INIT_2, INS_2)$ be two AHL-occurrence nets, such that $KI = (K, INIT, INS)$ is the result of their composition. Let KI_1 together with the AHL-net morphism $mp_1 : K_1 \to AN$ and KI_2 together with the AHL-net morphism $mp_2 : K_2 \to AN$ be two instantiated AHL-processes of the AHL-net AN. If the merged output places of K_1 and input places of K_2 are mapped by mp_1 and mp_2 to the same places in AN then there is one and only one AHL-net morphism $mp : K \to AN$, and KI together with the AHL-net morphism mp is an instantiated AHL-process of the AHL-net AN.*

Because for low-level occurrence nets the input/output behaviour is fixed by the net structure, two low-level occurrence nets are considered to be equivalent if they are isormorphic. For high-level occurrence nets the input/output behaviour additionally depends on the marking of their input places and on corresponding variable assignments. Hence we introduce the equivalence of two AHL-processes with instantiations, where the net structures of equivalent AHL-processes may be different, but they have the same input/output behaviour (see Def. C.7).

In more detail the AHL-occurrence nets have (up to renaming) the same sets of transitions and places and their instantiations are equivalent, i.e. there exist corresponding instantiations with the same input/output behaviour. In this case specific firing sequences of equivalent AHL-processes are comparable in the sense that they start and end up with the same data elements as marking of their input places and output places, respectively, but in general the corresponding transitions may be fired in a different order.

The AHL-processes with instantiations $KI = (K, \{init\}, \{L_{init}\})$ in Fig. 5 and $KI' = (K, \{init'\}, \{L_{init'}\})$ in Fig. 6 together with the AHL-net morphisms $mp : K \to AN_{Campus}$ and $mp' : K \to AN_{Campus}$ are equivalent. There is a bijection between their transitions and places, which is not an isomorphism. The bijection of places is defined by mapping the input places of K to the input places of K' (and analogously the output places) and the place *access point*$_{23}$ of KI to the place *access point*$_{41}$ of K. Moreover the instantiations L_{init} in Fig. 5 and $L_{init'}$ in Fig. 6 are equivalent, because they have the same input and output places up to renaming.

The main result in this context are suitable conditions s.t. AHL-net processes with instantiation can be composed in any order leading to equivalent high-level net processes. Here we use especially the assumption that the instantiations are consistent, i.e. there is a close relation between their input and output places (see Def. C.8). Given the AHL-process with instantiations KI together with $mp : K \to AN$ and KI' together with $mp' : K' \to AN$ as results of the composition and opposite composition of KI_1 with $mp_1 : K_1 \to AN$ and KI_2 with $mp_2 : K_2 \to AN$. Now the question arises if KI with mp and KI' with mp' are equivalent processes (see Def. C.7).

In order to obtain equivalent processes we check that the instantiations INS_1 and INS_2 are consistent, i.e. they can be composed in any order leading to instantiations with the same input/output behaviour (see Def. C.8). Thus equivalence of KI and KI' intuitively means that the AHL-processes KI_1 and KI_2 with consistent instantiations can be considered to be independent, because the composition in each order leads to equivalent processes.

As an example let KI_1 and KI_2 be the two instantiated AHL-processes as described above. Their sets of instantiations INS_1 and INS_2 are consistent, because the composition of the instantiations L_{init_1} (see Fig. 7) and $L_{init'_2}$ (see Fig. 8) leads to the instantiation L_{init} (see Fig. 5) and the composition of the instantiations L_{init_2} and $L_{init'_1}$ leads to the instantiation $L_{init'}$ (see Fig. 6). This leads to the following main result.

Main Result (Equivalence and Independence of AHL-Processes (see Thm. C.9)) *Given an AHL-net AN and AHL-occurrence nets $KI_1 = (K_1, INIT_1, INS_1)$ and $KI_2 = (K_2, INIT_2, INS_2)$, which are composable in both directions, with consistent instantiations and AHL-net morphisms $mp_1 : K_1 \to AN$ and $mp_2 : K_2 \to AN$. Then we have instantiated AHL-processes $KI = (K, INIT, INS)$ with $mp : K \to AN$ and $KI' = (K', INIT', INS')$ with $mp' : K' \to AN$ defined by the composition of KI_1 and KI_2 in both directions. Moreover both are equivalent processes of AN, provided that mp_1 and mp_2 are compatible with the compositions. Under these conditions KI_1 and KI_2 are called independent w.r.t. the given composition in both directions.*

Applying this main result to the AHL-net AN_{Campus} in Fig. 1 we have: The two basic instantiated processes defined by KI_1 in Fig. 7 and KI_2 in Fig. 8 are composable with consistent instantiations and the composition in both directions leads to equivalent instantiated processes defined by KI in Fig. 5 and KI' in Fig. 6. Hence the processes defined by KI_1 and KI_2 are independent.

5 Conclusion and Related Work

In this paper we have presented main results of a line of research concerning the modeling and analysis of high-level net processes. Based on the notions of high-level net processes with initial markings in [2,5] we have introduced high-level net processes with instantiations. As main results we have presented conditions for the composition and independence of high-level net processes and under these conditions the composition of two high-level net processes leads again to a high-level net process and they can be composed in any order leading to equivalent processes. In this case the two high-level net processes are called independent.

In [8,11] the semantics of object Petri nets is defined by a suitable extension of low-level processes. Objects Petri nets are high-level nets with P/T-systems as tokens. A process for an object Petri net is given by a pair of processes, a high-level net process containing low-level processes of the corresponding P/T-systems. In contrast the approach presented in this paper extends the notion of high-level net processes for algebraic high-level nets. The token structure of an algebraic high-level net is defined in its data type part that is not restricted to P/T-systems and we also use rules as tokens. For this reason low-level processes of P/T-systems as tokens are not considered.

Our main result of independence of high-level net processes is inspired by the results of local Church-Rosser for graph resp. net transformation [15,3], where under suitable conditions transformation steps can be performed in any order leading to the same result. In [6] we have transferred these results, so that net transformations and token firing can be executed in arbitrary order provided that certain conditions are satisfied. Hence an interesting aspect of future work will be to investigate the correspondence between these different concepts of independence in more detail to gain further results for high-level net processes.

References

[1] Degano, P., J. Meseguer and U. Montanari, *Axiomatizing Net Computations and Processes*, in: *Proc. on Logic in Computer Science (LICS)* (1989), pp. 175–185.

[2] Ehrig, H., *Behaviour and Instantiation of High-Level Petri Net Processes*, Fundamenta Informaticae **65** (2005), pp. 211–247.

[3] Ehrig, H., K. Ehrig, U. Prange and G. Taentzer, "Fundamentals of Algebraic Graph Transformation," EATCS Monographs in TCS, Springer, 2006.

[4] Ehrig, H., K. Gabriel, J. Padberg and K. Hoffmann, *Composition and Independence of High-Level Net Processes*, Technical report, TU Berlin, Fak. IV (2008), (for a preliminary version see http://tfs.cs.tu-berlin.de/formalnet/results.html).

[5] Ehrig, H., K. Hoffmann, J. Padberg, P. Baldan and R. Heckel, *High-level net processes*, in: *Formal and Natural Computing*, LNCS **2300** (2002), pp. 191–219.

[6] Ehrig, H., K. Hoffmann, J. Padberg, U. Prange and C. Ermel, *Independence of net transformations and token firing in reconfigurable place/transition systems*, in: *Proc. Application and Theory of Petri Nets (ATPN)*, LNCS **4546** (2007), pp. 104–123.

[7] Engelfriet, J., *Branching Processes of Petri Nets*, Acta Informatica **28** (1991), pp. 575–591.

[8] Farwer, B. and M. Köhler, *Mobile Object-Net Systems and their Processes*, Fundam. Inform. **60** (2004), pp. 113–129.

[9] Goltz, U. and W. Reisig, *The Non-sequential Behavior of Petri Nets*, Information and Control **57** (1983), pp. 125–147.

[10] Hoffmann, K., T. Mossakowski and H. Ehrig, *High-Level Nets with Nets and Rules as Tokens*, in: *Proc. Application and Theory of Petri Nets (ATPN)*, LNCS **3536**, Springer, 2005 pp. 268–288.

[11] Köhler, M. and H. Rölke, *Reference and Value Semantics Are Equivalent for Ordinary Object Petri Nets*, in: *Proc. Application and Theory of Petri Nets (ATPN)*, LNCS **3536** (2005), pp. 309–328.

[12] Meseguer, J. and U. Montanari, *Petri Nets Are Monoids*, Information and Computation **88** (1990), pp. 105–155.

[13] Meseguer, J., U. Montanari and V. Sassone, *On the Semantics of Place/Transition Petri Nets*, Mathematical Structures in Computer Science **7** (1997), pp. 359–397.

[14] Rozenberg, G., *Behaviour of Elementary Net Systems*, in: *Petri Nets: Central Models and Their Properties, Advances in Petri Nets*, LNCS **254** (1987), pp. 60–94.

[15] Rozenberg, G., "Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations," World Scientific, 1997.

A Appendix

A.1 Signature and Algebra for P/T-Systems and Rules as Tokens

Definition A.1 [HLNR-System-SIG Signature and Algebra]
Given vocabularies T_0 and P_0, the signature HLNR-System-SIG is given by
HLNR-System-SIG =

sorts: $Transitions, Places, Bool, System, Mor, Rules$

opns: $tt,\ ff\!:\to Bool$

$enabled : System \times Transitions \to Bool$

$fire : System \times Transitions \to System$

$applicable : Rules \times Mor \to Bool$

$transform : Rules \times Mor \to System$

$union : System \times System \to System$

$student :\to System$

$isomorphic : System \times System \to Bool$

$cod : Mor \to System$

and the HLNR-System-SIG-algebra A for P/T-systems and rules as tokens is given by

- $A_{Transitions} = T_0, A_{Places} = P_0, A_{Bool} = \{true, false\}$,
- A_{System} the set of all P/T-systems over T_0 and P_0, i.e.
  $A_{System} = \{PN | PN = (P, T, pre, post, M)\ \textit{P/T-system}, P \subseteq P_0, T \subseteq T_0\} \cup \{undef\}$,
- A_{Mor} the set of all P/T-morphisms for A_{System}, i.e.
  $A_{Mor} = \{f | f : PN \to PN'\ \textit{P/T-morphism with}\ PN, PN' \in A_{System}\}$,
- A_{Rules} the set of all rules of P/T-systems, i.e.
  $A_{Rules} = \{r | r = (L \overset{i_1}{\leftarrow} I \overset{i_2}{\rightarrow} R)\ \textit{rule of P/T-systems with strict inclusions}\ i_1, i_2\}$,
- $tt_A = true,\ ff_A = false$,
- $enabled_A : A_{System} \times T_0 \to \{true, false\}$ for $PN = (P, T, pre, post, M)$ with

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

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

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

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

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

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

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

  where for $L \xrightarrow{m} G$ and $applicable_A(r,m) = true$ we have a direct transformation $G \xRightarrow{r} H$,
- $union_A : A_{System} \times A_{System} \to A_{System}$ the disjoint union (i.e. the two P/T-systems are combined without interaction) with

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

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

$$pre_3(t) = \underline{\textit{if}}\ t \in T_1\ \underline{\textit{then}}\ pre_1(t)\ \underline{\textit{else}}\ pre_2(t)$$
$$post_3(t) = \underline{\textit{if}}\ t \in T_1\ \underline{\textit{then}}\ post_1(t)\ \underline{\textit{else}}\ post_2(t)$$

- $student_A : \{\star\} \to A_{System}$ with $student_A(\star) = (\{p\}, \emptyset, \emptyset, \emptyset)$,
- $isomorphic_A : A_{System} \times A_{System} \to \{true, false\}$ with

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

  where $PN_1 \cong PN_2$ means that there is a strict P/T-morphism $f = (f_P, f_T) : PN_1 \to PN_2$ s.t. f_P, f_T are bijective functions,
- $cod_A : A_{Mor} \to A_{System}$ with $cod_A\ (f : PN_1 \to PN_2) = PN_2$.

B Algebraic High-Level Net Processes

Definition B.1 [Place/Transition Net] A place/transition (P/T) net $N = (P, T, pre, post)$ consists of sets P and T of places and transitions respectively, and pre- and post domain functions $pre, post : T \to P^{\oplus}$ where $P^{\oplus}$ is the free commutative monoid over P.

A P/T-net morphism $f : N_1 \to N_2$ is given by $f = (f_P, f_T)$ with functions $f_P : P_1 \to P_2$ and $f_T : T_1 \to T_2$ satisfying

$$f_P^{\oplus} \circ pre_1 = pre_2 \circ f_T \text{ and } f_P^{\oplus} \circ post_1 = post_2 \circ f_T$$

where the extension $f_P^{\oplus} : P_1^{\oplus} \to P_2^{\oplus}$ of $f_P : P_1 \to P_2$ is defined by $f_P^{\oplus}(\sum_{i=1}^{n} k_i \cdot p_i) = \sum_{i=1}^{n} k_i \cdot f_P(p_i)$. A P/T-net morphism $f = (f_P, f_T)$ is called injective if f_P and f_T are injective and is called isomorphism if f_P and f_T are bijective.

The category defined by P/T-nets and P/T-net morphisms is denoted by **PTNet** where the composition of P/T-net morphisms is defined componentwise for places and transitions.

Because the notion of pushouts is essential for our main results we state the construction of pushouts in the category **PTNet** of place/transition nets. Intuitively a pushout means the gluing of two nets along an interface net. The construction is based on the pushouts for the sets of transitions and places in the category **SET**. In the category **SET** of sets and functions the pushout object D for given $f_1 : A \to B$ and $f_2 : A \to C$ is defined by the quotient set $D = B \uplus C/\equiv$, short $D = B \circ_A C$, where $B \uplus C$ is the disjoint union of B and C and $\equiv$ is the equivalence relation generated by $f_1(a) \equiv f_2(a)$ for all $a \in A$. In fact, D can be interpreted as the gluing of B and C along A: Starting with the disjoint union $B \uplus C$ we glue together the elements $f_1(a) \in B$ and $f_2(a) \in C$ for each $a \in A$. The pushout object N_3 in the category **PTNet** is constructed componentwise for transitions and places in **SET** with corresponding pre- and post domain functions. For given P/T-net morphisms $f_1 : N_0 \to N_1$ and $f_2 : N_0 \to N_2$ the pushout of f_1 and f_2 is defined by the pushout diagram (PO) in **PTNet** and is denoted by $N_3 = N_1 \circ_{(N_0,f_1,f_2)} N_2$.

$$\begin{tikzcd} N_0 \arrow[r, "f_1"] \arrow[d, "f_2"'] \arrow[dr, phantom, "\textbf{(PO)}"] & N_1 \arrow[d, "f_1'"] \\ N_2 \arrow[r, "f_2'"'] & N_3 \end{tikzcd}$$

Definition B.2 [Pushouts of Place/Transition Nets] Given P/T-net morphisms $f_1 : N_0 \to N_1$ and $f_2 : N_0 \to N_2$ then the pushout diagram (1) and the pushout object N_3 in the category **PTNet**, written $N_3 = N_1 \circ_{(N_0,f_1,f_2)} N_2$, with $N_x = (P_x, T_x, pre_x, post_x)$ for $x = 0, 1, 2, 3$ is constructed as follows:

- $T_3 = T_1 \circ_{T_0} T_2$ with $f'_{1,T}$ and $f'_{2,T}$ as pushout (2) of $f_{1,T}$ and $f_{2,T}$ in **SET**.
- $P_3 = P_1 \circ_{P_0} P_2$ with $f'_{1,P}$ and $f'_{2,P}$ as pushout (3) of $f_{1,P}$ and $f_{2,P}$ in **SET**
- $pre_3(t) = \begin{cases} [pre_1(t_1)] & ; \text{ if } f'_{1,T}(t_1) = t \\ [pre_2(t_2)] & ; \text{ if } f'_{2,T}(t_2) = t \end{cases}$
- $post_3(t) = \begin{cases} [post_1(t_1)] & ; \text{ if } f'_{1,T}(t_1) = t \\ [post_2(t_2)] & ; \text{ if } f'_{2,T}(t_2) = t \end{cases}$

$$\begin{tikzcd} N_0 \arrow[r, "f_1"] \arrow[d, "f_2"'] \arrow[dr, phantom, "\textbf{(1)}"] & N_1 \arrow[d, "f_1'"] \\ N_2 \arrow[r, "f_2'"'] & N_3 \end{tikzcd} \quad \begin{tikzcd} T_0 \arrow[r, "f_{1,T}"] \arrow[d, "f_{2,T}"'] \arrow[dr, phantom, "\textbf{(2)}"] & T_1 \arrow[d, "f_{1,T}'"] \\ T_2 \arrow[r, "f_{2,T}'"'] & T_3 \end{tikzcd} \quad \begin{tikzcd} P_0 \arrow[r, "f_{1,P}"] \arrow[d, "f_{2,P}"'] \arrow[dr, phantom, "\textbf{(3)}"] & P_1 \arrow[d, "f_{1,P}'"] \\ P_2 \arrow[r, "f_{2,P}'"'] & P_3 \end{tikzcd}$$

Definition B.3 [Algebraic High-Level Net] An algebraic high-level (AHL) net $AN = (SP, P, T, pre, post, cond, type, A)$ consists of

- an algebraic specification $SP = (\Sigma, E; X)$ with signature $\Sigma = (S, OP)$, equations E, and additional variables X;
- a set of places P and a set of transitions T;
- pre- and post domain functions $pre, post : T \to (T_\Sigma(X) \otimes P)^\oplus$;
- firing conditions $cond : T \to \mathcal{P}_{fin}(Eqns(\Sigma; X))$;

- a type of places $type : P \to S$ and
- a (Σ, E)-algebra A

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

An AHL-net morphism $f : AN_1 \to AN_2$ is given by $f = (f_P, f_T)$ with functions $f_P : P_1 \to P_2 \quad and \quad f_T : T_1 \to T_2$ satisfying

(1) $(id \otimes f_P)^\oplus \circ pre_1 = pre_2 \circ f_T$ and $(id \otimes f_P)^\oplus \circ post_1 = post_2 \circ f_T$,

(2) $cond_2 \circ f_T = cond_1$ and

(3) $type_2 \circ f_P = type_1$.

The category defined by AHL-nets and AHL-net morphisms is denoted by **AHLNet** where the composition of AHL-net morphisms is defined componentwise for places and transitions.

In the following we omit the indices of functions f_P and f_T if no confusion arises.

The construction of pushouts in the category **AHLNet** of AHL-nets with fixed specification SP and algebra A can be analogously defined to the construction of pushouts in **PTNet** described above (for details see [3]).

Definition B.4 [Firing Behaviour of AHL-Nets] A marking of an AHL-net AN is given by $M \in CP^\oplus$ where $CP = (A \otimes P) = \{(a, p) | a \in A_{type(p)}, p \in P\}$.

The set of variables $Var(t) \subseteq X$ of a transition $t \in T$ are the variables of the net inscriptions in $pre(t), post(t)$ and $cond(t)$. Let $v : Var(t) \to A$ be a variable assignment with term evaluation $v^\sharp : T_\Sigma(Var(t)) \to A$, then (t, v) is a consistent transition assignment iff $cond_{AN}(t)$ is validated in A under v. The set CT of consistent transition assignments is defined by $CT = \{(t, v) | (t, v) \ \textit{consistent transition assignment}\}$.

A transition $t \in T$ is enabled in M under v iff $(t, v) \in CT$ and $pre_A(t, v) \leq M$, where $pre_A : CT \to CP^\oplus$ defined by $pre_A(t, v) = \sum_{i=1}^n (v^\sharp(term_i, p_i)$ for $pre(t) = \sum_{i=1}^n (term_i, p_i)$ and places (similar $post_A : CT \to CP^\oplus$). Then the follower marking is computed by $M' = M \ominus pre_A(t, v) \oplus post_A(t, v)$.

Definition B.5 [AHL-Occurrence Net] An AHL-occurrence net K is an AHL - net $K = (SP, P, T, pre, post, cond, type, A)$ such that for all $t \in T$ with $pre(t) = \sum_{i=1}^n (term_i, p_i)$ and notation $\bullet t = \{p_1, \ldots, p_n\}$ and similarly $t\bullet$ we have

(i) (*Unarity*): $\bullet t, t\bullet$ are sets rather than multisets for all $t \in T$, i.e. for $\bullet t$ the places $p_1 \ldots p_n$ are pairwise distinct. Hence $|\bullet t| = n$ and the arc from p_i to t has a unary arc-inscription $term_i$.

(ii) (*No Forward Conflicts*): $\bullet t \cap \bullet t' = \emptyset$ for all $t, t' \in T, t \neq t'$

(iii) (*No Backward Conflicts*): $t \bullet \cap t' \bullet = \emptyset$ for all $t, t' \in T, t \neq t'$

(iv) (*Partial Order*): the causal relation $< \subseteq (P \times T) \cup (T \times P)$ defined by the transitive closure of $\{(p, t) \in P \times T \mid p \in \bullet t\} \cup \{(t, p) \in T \times P \mid p \in t\bullet\}$ is a finitary strict partial order, i.e. the partial order is irreflexive and for each element in the partial order the set of its predecessors is finite.

Definition B.6 [AHL-Process] An AHL-process of an AHL-net AN is an AHL-net morphism $p : K \to AN$ where K is an AHL-occurrence net.

Definition B.7 [AHL-Occurrence Net with Initial Markings] An AHL-occurrence net with initial markings $(K, INIT)$ consists of an AHL-occurrence net K and a set $INIT$ of initial markings $init \in INIT$ of the input places $IN(K)$, where the input places of K are defined by $IN(K) = \{p \in P | \bullet p = \emptyset\}$ and similarly the output places of K are defined by $OUT(K) = \{p \in P | p\bullet = \emptyset\}$.

Definition B.8 [Flattening] Given AHL-net AN as above then the flattening of AN is a P/T-net $Flat(AN) = N = (CP, CT, pre_A, post_A)$ with

- $CP = A \otimes P = \{(a, p) | a \in A_{type(p)}, p \in P\}$,
- $CT = \{(t, v) | t \in T, v : Var(t) \to A$ s.t. $cond(t)$ valid in A under v $\}$ and
- pre_A and $post_A$ as defined in Def. B.4.

Given an AHL-net morphism $f : AN_1 \to AN_2$ by $f = (f_P, f_T)$ then $Flat(f) = (id_A \otimes f_P : CP_1 \to CP_2, f_C : CT_1 \to CT_2)$ is given by $id_A \otimes f_P(a, p) = (a, f_p(p))$ and $f_C(t, v) = (f_T(t), v)$.

Definition B.9 [Skeleton] Given an AHL-net AN as above then the skeleton of AN is a P/T-net $Skel(AN) = (P, T, pre_S, post_S)$ with $pre_S(t) = \sum_{i=1}^{n} p_i$ for $pre(t) = \sum_{i=1}^{n}(term_i, p_i)$ and similar for $post_S : T \to P^{\oplus}$. Given an AHL-net morphism $f : AN_1 \to AN_2$ by $f = (f_P, f_T)$ then $Skel(f) = f = (f_P : P_1 \to P_2, f_T : T_1 \to T_2)$.

Remark B.10 The flattening construction defined in Def. B.8 and the skeleton construction defined in Def. B.9 are well-defined and can be turned into a functor $Flat$: **AHLNet** $\to$ **PTNet** and a functor $Skel$: **AHLNet** $\to$ **PTNet** which preserve pushouts, i.e. given the pushout (1) in **AHLNet** then there are corresponding pushouts (2) and (3) in **PTNet**. Moreover we have for each AN a projection $proj(AN) : Flat(AN) \to Skel(AN)$ leading to a natural transformation $proj : Flat \to Skel$.

$$\begin{tikzcd}
AN_0 \arrow[r, "f_1"] \arrow[d, "f_2"'] \arrow[dr, phantom, "(1)"] & AN_1 \arrow[d, "f_1'"] \\
AN_2 \arrow[r, "f_2'"'] & AN_3
\end{tikzcd}$$

$$\begin{tikzcd}
Flat(AN_0) \arrow[r, "Flat(f_1)"] \arrow[d, "Flat(f_2)"'] \arrow[dr, phantom, "(2)"] & Flat(AN_1) \arrow[d, "Flat(f_1')"] \\
Flat(AN_2) \arrow[r, "Flat(f_2')"'] & Flat(AN_3)
\end{tikzcd}
\qquad
\begin{tikzcd}
Skel(AN_0) \arrow[r, "Skel(f_1)"] \arrow[d, "Skel(f_2)"'] \arrow[dr, phantom, "(3)"] & Skel(AN_1) \arrow[d, "Skel(f_1')"] \\
Skel(AN_2) \arrow[r, "Skel(f_2')"'] & Skel(AN_3)
\end{tikzcd}$$

Definition B.11 [Instantiations of AHL-Occurrence Net] Given an AHL-occurrence net with initial markings $(K, INIT)$ with $init \in INIT$. An instantiation L_{init} of $(K, init)$ is a low-level occurrence net $L_{init} \subseteq Flat(K)$ with input places $IN(L_{init}) = init$ such that the projection $proj : L_{init} \to Skel(K)$ defined by $proj_P(a, p) = p$ and $proj_T(t, v) = t$ is an isomorphism of low-level occurrence nets.

Definition B.12 [AHL-Occurrence Net with Instantiations] An AHL-occurrence net with instantiations $KI = (K, INIT, INS)$ is an AHL-occurrence net with initial markings $(K, INIT)$ and a set INS of instantiations, such that for each $init \in INIT$ we have a distinguished instantiation $L_{init} \in INS$, i.e. $INS = \{L_{init} | init \in INIT\}$.

An AHL-occurrence net with instantiations KI defines for each $init \in INIT$ with $IN(L_{init}) = init$ an output $out = OUT(L_{init})$ with $proj_P(out) = OUT(K)$. Let $EXIT$ be the set of all markings of the output places $OUT(K)$, then we obtain a function $inout : INIT \to EXIT$ by $inout(init) = OUT(L_{init})$.

Definition B.13 [AHL-Process with Instantiations] An instantiated AHL-process of an AHL-net AN is an AHL-occurrence net with instantiations $KI = (K, INIT, INS)$ together with an AHL-net morphism $mp : K \to AN$.

C Composition, Equivalence and Independence of Algebraic High-Level Net Processes

Definition C.1 [Composability of AHL-Occurrence Nets] Given the AHL-occurrence nets $K_x = (SP, P_x, T_x, pre_x, post_x, cond_x, type_x, A)$ for $x = 1, 2$ and $I = (SP, P_I, T_I, pre_I, post_I, cond_I, type_I, A)$ with $T_I = \emptyset$ and two injective AHL-net morphisms $i_1 : I \to K_1$ and $i_2 : I \to K_2$. Then (K_1, K_2) is composable w.r.t. (I, i_1, i_2) if $i_1(P_I) \subseteq OUT(K_1)$ and $i_2(P_I) \subseteq IN(K_2)$.

Theorem C.2 (Composition of AHL-Occurrence Nets) *Given the AHL- occurrence nets K_1, K_2 and I as above and two injective AHL-net morphisms $i_1 : I \to K_1$ and $i_2 : I \to K_2$ such that (K_1, K_2) is composable w.r.t. (I, i_1, i_2). Then the pushout digram (PO) exists in the category* **AHLNet** *and the pushout object K, with $K = K_1 \circ_{(I,i_1,i_2)} K_2$, is an AHL-occurrence net and is called composition of (K_1, K_2) w.r.t. (I, i_1, i_2).*

$$\begin{tikzcd} I \arrow[r, "i_1"] \arrow[d, "i_2"'] \arrow[dr, phantom, "\textbf{(PO)}"] & K_1 \arrow[d, "i_1'"] \\ K_2 \arrow[r, "i_2'"'] & K \end{tikzcd}$$

Proof (Sketch) For the existence and construction of pushouts in **AHLNet** we refer to [3]. As mentioned in Section 3 it can be constructed componentwise similar to pushouts in **PTNet**. It remains to show that the result of the composition of (K_1, K_2) w.r.t. (I, i_1, i_2) given by $K = (SP, P, T, pre, post, cond, type, A)$ is an occurrence net indeed:

(i) *Unarity:* is obtained as the set of transitions T is obtained by disjoint union.

(ii) *No forward conflicts:* Since AHL-net morphisms preserve the adjacencies of transitions (i.e. pre and post domain), in case of $t_1 \neq t_2$ and $p \in \bullet t_1 \cap \bullet t_2$ for $t_1, t_2 \in T$ both transitions have a preimage in T_1 and T_2, respectively. Moreover, p has a preimage in both P_1 and P_2, so one of the preimages is in the corresponding OUT set. But this contradicts the fact that this place has to be in the preset of the corresponding transition.

(iii) *No backward conflicts:* Analogously.

(iv) *Partial Order:* follows from the partial order of K_1 and K_2 and from the composability condition.

□

Definition C.3 [Composition of Instantiations] Given the AHL-occurrence nets K_1, K_2 and I as above and two injective AHL-net morphism $i_1 : I \to K_1$ and $i_2 : I \to K_2$ such that (K_1, K_2) is composable w.r.t. (I, i_1, i_2). Let $KI_x = (K_x, INIT_x, INS_x)$ for $x = 1, 2$ be two AHL-occurrence nets with instantiations and $L_{init_1} \in INS_1$ and $L_{init_2} \in INS_2$. Then (L_{init_1}, L_{init_2}) is composable w.r.t. (I, i_1, i_2) if for all $(a, p) \in A_{type(p)} \otimes P_I : (a, i_1(p)) \in OUT(L_{init_1}) \Rightarrow (a, i_2(p)) \in IN(L_{init_2})$.

From (I, i_1, i_2) we construct the induced instantiation interface (J, j_1, j_2) of (L_{init_1}, L_{init_2}) with $J = (P_J, T_J, pre_J, post_J)$ by

- $P_J = \{(a, p) | (a, i_1(p)) \in OUT(L_{init_1})\}$,
- $T_J = \emptyset$,
- $pre_J = post_J = \emptyset$ (the empty function) and
- $j_x : J \to L_{init_x}$ for $x = 1, 2$ defined by $j_{x,P} = id_A \otimes i_{x,P}$ and $j_{x,T} = \emptyset$.

$$\begin{tikzcd} J \arrow[r, "j_1"] \arrow[d, "j_2"'] \arrow[dr, phantom, "\text{(PO)}"] & L_{init_1} \arrow[d, "j_1'"] \\ L_{init_2} \arrow[r, "j_2'"'] & L_{init} \end{tikzcd}$$

The composition of (L_{init_1}, L_{init_2}) w.r.t. the instantiation interface (J, j_1, j_2) induced by (I, i_1, i_2) is defined by the pushout diagram (PO) in **PTNet** and is denoted by $L_{init} = L_{init_1} \circ_{(J,j_1,j_2)} L_{init_2}$.

Theorem C.4 (Composition of AHL-Occurrence Nets with Instantiations) *Given the AHL-occurrence nets K_1, K_2 and I as above and two injective AHL-net morphism $i_1 : I \to K_1$ and $i_2 : I \to K_2$ such that (K_1, K_2) is composable w.r.t. (I, i_1, i_2). Let $KI_x = (K_x, INIT_x, INS_x)$ for $x = 1, 2$ be two AHL-occurrence nets with instantiations. Then the composition of (KI_1, KI_2) w.r.t. (I, i_1, i_2) is defined by $KI = (K, INIT, INS)$ with*

- $K = K_1 \circ_{(I,i_1,i_2)} K_2$,
- $INS = \{L_{init_1} \circ_{(J,j_1,j_2)} L_{init_2} | L_{init_x} \in INS_x$ *for* $x = 1, 2, (L_{init_1}, L_{init_2})$ *is composable w.r.t.* (J, j_1, j_2) *induced by* $(I, i_1, i_2)\}$,
- *and* $INIT = \{IN(L_{init}) | L_{init} \in INS\}$

and $KI = KI_1 \circ_{(I,i_1,i_2)} KI_2$ is an AHL-occurrence net with instantiations.

Proof (Sketch) To prove that $KI = (K, INIT, INS)$ is well-defined, first note that K is an occurrence net due to Theorem C.2. Moreover, for each $L_{init} \in INS$ we need to show $L_{init} \subseteq Flat(K)$ and $in \circ proj(K) : L_{init} \to Skel(K)$ is an isomorphism in the diagram below where we have the following pushouts: (PO1) by construction, (PO2) since $Flat$ preserves pushout $K = K_1 \circ_I K_2$ and (PO3) since $Skel$ preserves pushout $K = K_1 \circ_I K_2$.

$proj(I), proj(K_1), proj(K_2)$ and $proj(K)$ are projections from the flattening to the skeleton construction (see Remark B.10) and in_I, in_1, in_2 are inclusions where $J \subseteq Flat(I) = (A \otimes P, \emptyset, \emptyset, \emptyset)$ and in is induced by (PO1).

$$
\begin{tikzcd}
 & L_{init1} \arrow[rrr, "in_1"] \arrow[ddr] \arrow[ddr, phantom, "\mathbf{(PB)}"] & & & Flat(K_1) \arrow[rr, "proj(K_1)"] \arrow[ddr] & & Skel(K_1) \arrow[ddr] & \\
J \arrow[ur, "j_1"] \arrow[rrr, "in_I"] \arrow[ddr, "j_2"'] \arrow[dr, phantom, "\mathbf{(PO1)}"] & & & Flat(I) \arrow[ur, "Flat(i_1)"] \arrow[rr, "proj(K_1)"] \arrow[ddr, "Flat(i_2)"] \arrow[dr, phantom, "\mathbf{(PO2)}"] & & Skel(I) \arrow[ur] \arrow[ddr] \arrow[dr, phantom, "\mathbf{(PO3)}"] & & \\
 & & L_{init} \arrow[rrr, "in"] & & & Flat(K) \arrow[rr, "proj(K)"] & & Skel(K) \\
 & L_{init2} \arrow[ur] \arrow[rrr, "in_2"] & & & Flat(K_2) \arrow[ur] \arrow[rr, "proj(K_2)"] & & Skel(K_2) \arrow[ur] &
\end{tikzcd}
$$

Since $proj(I) \circ in_I$ can be shown to be an isomorphism (using that J is pullback of $Flat(i_1)$ and in_1) and $proj(K_x) \circ in_x$ are by assumption isomorphisms for $x = 1, 2$, we conclude that $proj(K) \circ in$ is isomorphic as well. Hence in is injective and can be chosen to be an inclusion $in : L_{init} \to Flat(K)$.

□

Definition C.5 [Composability of AHL-Processes with Instantiations] Given the AHL-occurrence nets K_1, K_2 and I as above and two injective AHL-net morphism $i_1 : I \to K_1$ and $i_2 : I \to K_2$. Let $KI_x = (K_x, INIT_x, INS_x)$ together with the AHL-net morphisms $mp_x : K_x \to AN$ for $x = 1, 2$ be two instantiated AHL-processes of the AHL-net AN. Then (mp_1, mp_2) is composable w.r.t. (I, i_1, i_2) if

(i) (K_1, K_2) is composable w.r.t. (I, i_1, i_2) and

(ii) $mp_1 \circ i_1 = mp_2 \circ i_2$.

Theorem C.6 (Composition of AHL-Processes with Instantiations) *Given the AHL-occurrence nets K_1, K_2 and I as above and two injective AHL-net morphism $i_1 : I \to K_1$ and $i_2 : I \to K_2$. Let $KI_x = (K_x, INIT_x, INS_x)$ together with the AHL-net morphisms $mp_x : K_x \to AN$ for $x = 1, 2$ be two instantiated AHL-processes of the AHL-net AN such that (mp_1, mp_2) is composable w.r.t. (I, i_1, i_2). Then the instantiated AHL-occurrence net $KI = KI_1 \circ_{(I,i_1,i_2)} KI_2$ together with the induced AHL-net morphism $mp : K \to AN$ is an instantiated AHL-process of the AHL-net AN, where K is the AHL-occurrence net of KI.*

$$
\begin{tikzcd}
I \arrow[r, "i_1"] \arrow[d, "i_2"'] \arrow[dr, phantom, "\mathbf{(PO)}"] & K_1 \arrow[d, "i'_1"] \arrow[ddr, bend left, "mp_1"] & \\
K_2 \arrow[r, "i'_2"'] \arrow[drr, bend right, "mp_2"'] & K \arrow[dr, "mp"] & \\
 & & AN
\end{tikzcd}
$$

Proof (Sketch) Due to Def. C.5 , Thm. C.4 and the universal property of pushouts there is the morphism $mp : K \to AN$, that uniquely commutes $mp_1 = i'_1 \circ mp$ and $mp_2 = i'_2 \circ mp$. □

Definition C.7 [Equivalence of AHL-Processes with Instantiations] Let $KI = (K, INIT, INS)$ and $KI' = (K', INIT', INS')$ together with AHL-net morphisms $mp : K \to AN$ and $mp' : K' \to AN$ two AHL-processes of an AHL-net AN. Then these two processes are called equivalent if

(i) there are bijections $e_P : P_K \to P_{K'}$ and $e_T : T_K \to T_{K'}$ such the following diagram commutes componentwise

$$
\begin{tikzcd}
K \arrow[rr, shift left, "e_P"] \arrow[rr, shift right, "e_T"'] \arrow[dr, bend right, "mp"'] & \arrow[d, phantom, "="] & K' \arrow[dl, bend left, "mp'"] \\
 & AN &
\end{tikzcd}
$$

(ii) and the instantiations are equivalent, i.e. for each $L_{init} \in INS$ there exists a

$L_{init'} \in INS'$ and vice versa such that

$$\forall (a,p) \in A_{type(p)} \otimes P_K : (a,p) \in IN(L_{init}) \Leftrightarrow (a, e_P(p)) \in IN(L_{init'}) \text{ and}$$
$$(a,p) \in OUT(L_{init}) \Leftrightarrow (a, e_P(p)) \in OUT(L_{init'})$$

Definition C.8 [Consistency of Instantiations] Given AHL-occurrence nets K_1, K_2 and I as in Def. C.1 and injective AHL-net morphism $i_1 : I \to K_1$, $i_2 : I \to K_2$, $i_3 : I \to K_1$ and $i_4 : I \to K_2$ such that (K_1, K_2) is composable w.r.t. (I, i_1, i_2) and (K_2, K_1) is composable w.r.t. (I, i_4, i_3) with pushout (1) and (2), respectively. Moreover let $KI_x = (K_x, INIT_x, INS_x)$ be AHL-occurrence nets with instantiations for K_x $(x = 1, 2)$.

Then (INS_1, INS_2) is called consistent if for all composable $(L_{init_1}, L_{init_2}) \in INS_1 \times INS_2$ w.r.t. (J, j_1, j_2) induced by (I, i_1, i_2) with pushout (3) there are composable $(L_{init'_2}, L_{init'_1}) \in INS_2 \times INS_1$ w.r.t. (J, j_4, j_3) induced by (I, i_4, i_3) with pushout (4) and vice versa, s.t. in both cases the instantiations satisfy the following properties 1.-4. for gluing points GP defined below:

1. $IN(L_{init_x}) \setminus GP(L_{init_x}) = IN(L_{init'_x}) \setminus GP(L_{init'_x})$ and

2. $OUT(L_{init_x}) \setminus GP(L_{init_x}) = OUT(L_{init'_x}) \setminus GP(L_{init'_x})$ for $x = 1, 2$

Moreover we require for all $(a,p) \in A_{type(p)} \otimes P_I$:

3. $(a, i_3(p)) \in IN(L_{init_1}) \Leftrightarrow (a, i_2(p)) \in IN(L_{init'_2})$

4. $(a, i_1(p)) \in OUT(L_{init'_1}) \Leftrightarrow (a, i_4(p)) \in OUT(L_{init_2})$

The gluing points GP are defined by

- $GP(P_{K_1}) = i_1(P_I) \cup i_3(P_I)$, $GP(P_{K_2}) = i_2(P_I) \cup i_4(P_I)$,
- $GP(L_{init_x}) = \{(a,p) \in L_{init_x} | p \in GP(P_{K_x})\}$ and
- $GP(L_{init'_x}) = \{(a,p) \in L_{init'_x} | p \in GP(P_{K_x})\}$ for $x = 1, 2$.

$$\begin{tikzcd} I \arrow[r, "i_1"] \arrow[d, "i_2"'] \arrow[dr, phantom, "(1)"] & K_1 \arrow[d, "i'_1"] \\ K_2 \arrow[r, "i'_2"'] & K \end{tikzcd} \quad \begin{tikzcd} I \arrow[r, "i_4"] \arrow[d, "i_3"'] \arrow[dr, phantom, "(2)"] & K_2 \arrow[d, "i'_4"] \\ K_1 \arrow[r, "i'_3"'] & K' \end{tikzcd} \quad \begin{tikzcd} J \arrow[r, "j_1"] \arrow[d, "j_2"'] \arrow[dr, phantom, "(3)"] & L_{init_1} \arrow[d, "j'_1"] \\ L_{init_2} \arrow[r, "j'_2"'] & L_{init} \end{tikzcd} \quad \begin{tikzcd} J \arrow[r, "j_4"] \arrow[d, "j_3"'] \arrow[dr, phantom, "(4)"] & L_{init'_2} \arrow[d, "j'_4"] \\ L_{init'_1} \arrow[r, "j'_3"'] & L_{init'} \end{tikzcd}$$

Theorem C.9 (Equivalence and Independence of AHL-Processes) *Given an AHL-net AN and AHL-occurrence nets $KI_x = (K_x, INIT_x, INS_x)$ with consistent instantiations as in Def. C.8 with AHL-net morphisms $mp_x : K_x \to AN$ for $x = 1, 2$.*

Then we have instantiated AHL-processes $KI = (K, INIT, INS)$ with $mp : K \to AN$ and $KI' = (K', INIT', INS')$ with $mp' : K' \to AN$ defined by opposite compositions $KI = KI_1 \circ_{(I,i_1,i_2)} KI_2$ and $KI' = KI_2 \circ_{(I,i_4,i_3)} KI_1$ and both are equivalent processes of AN, provided that

(i) *K_1 and K_2 have no isolated places, i.e. $IN(K_x) \cap OUT(K_x) = \emptyset$ for $x = 1, 2$*

(ii) *mp_1 and mp_2 are compatible with i_1, i_2, i_3 and i_4, i.e. $mp_1 \circ i_1 = mp_2 \circ i_2 = mp_1 \circ i_3 = mp_2 \circ i_4 : I \to AN$.*

Under these conditions KI_1 and KI_2 wrt. i_1 and i_2 are called independent.

Proof (Sketch) The instantiated AHL-processes KI and KI' with $mp : K \to AN$ and $mp' : K' \to AN$ exist by Theorem C.6. It remains to show that they are equivalent.

Construction of bijections. The bijection $e_T : T_K \to T_{K'}$ follows from the fact that $I_T = \emptyset$ and hence $T_K \cong T_{K_1} \uplus T_{K_2}$ and $T_{K'} \cong T_{K_2} \uplus T_{K_1}$. In order to obtain the bijection $e_P : P_K \to P_{K'}$ we show that P_K and $P_{K'}$ can be represented by the following disjoint unions of gluing points GP and non gluing points NGP in pushout (1) and (2) in Def. C.8.

$$P_K = GP_1(P_K) \cup GP_2(P_K) \cup GP_3(P_K) \cup NGP(P_K) \textit{ with}$$
$$GP_1(P_K) = i_1' \circ i_3(P_I), GP_2(P_K) = i_2' \circ i_4(P_I) \textit{ and } GP_3(P_K) = i_1' \circ i_1(P_I)$$

$$P_{K'} = GP_1(P_{K'}) \cup GP_2(P_{K'}) \cup GP_3(P_{K'}) \cup NGP(P_{K'}) \textit{ with}$$
$$GP_1(P_{K'}) = i_4' \circ i_2(P_I), GP_2(P_{K'}) = i_3' \circ i_1(P_I) \textit{ and } GP_3(P_{K'}) = i_3' \circ i_3(P_I)$$

This allows to define $e_{P_x} : GP_x(P_K) \to GP_x(P_{K'})$ for $x = 1, 2, 3$ by $e_{P_1}(i_1' \circ i_3(p)) = i_4' \circ i_2(p)$ for all $p \in P_I$ and similar for e_{P_2} and e_{P_3}. Since i_1', i_3, i_4', and i_2 are all injective e_{P_1} is bijective and similar also e_{P_2} and e_{P_3} are bijective.

Finally also $e_{P_4} : NGP(P_K) \to NGP(P_{K'})$ can be defined as bijection. Using $IN(K_x) \cap OUT(K_x) = \emptyset$ for $x = 1, 2$ it can be shown that P_K (and similar $P_{K'}$) is a disjoint union of all four components leading to a bijection $e_P = e_{P_1} \cup e_{P_2} \cup e_{P_3} \cup e_{P_4} : P_K \to P_{K'}$. With these definitions it can be shown explicitly that the diagram in Def. C.7 commutes componentwise.

Equivalence of instantiations. Given $L_{init} = L_{init_1} \circ_{(J,j_1,j_2)} L_{init_2}$ with pushout (3) in Def. C.8 we have by consistency of (INS_1, INS_2) $L_{init'} = L_{init_2'} \circ_{(J,j_4,j_3)} L_{init_1'}$ with pushout (4) s.t. properties 1.-4. in Def. C.8 are satisfied. This allows to show by case distinction using the definition of e_P above that we have for all $(a, p) \in A_{type(p)} \otimes P_K$: $(a, p) \in IN(L_{init}) \Leftrightarrow (a, e_P(p)) \in IN(L_{init'})$ and $(a, p) \in OUT(L_{init}) \Leftrightarrow (a, e_P(p)) \in OUT(L_{init'})$.

The opposite direction, where $L_{init'} = L_{init_2'} \circ_{(J,j_4,j_3)} L_{init_1'}$ is given with pushout (4), follows by symmetry.

□