Modular System Development with Pullbacks*

Marek A. Bednarczyk[1], Luca Bernardinello[2], Benoît Caillaud[3], Wiesław Pawłowski[1], and Lucia Pomello[2]

[1] IPI PAN, Gdańsk, Poland
[2] DISCO, Università degli Studi di Milano-Bicocca, Milano, Italy
[3] IRISA, Rennes, France

Abstract. Two, seemingly different modular techniques for concurrent system development are investigated from a categorical perspective. A novel approach is presented in which they turn out to be merely special instances of *pullback*, a general categorical limit construction. Interestingly, the approach is based on truly concurrent semantics of systems.

1 Mathematical Preliminaries

A *transition system* is a tuple $\mathcal{S} = (S, \hat{s}, A, T)$ where S is a set of *states*, $\hat{s} \in S$ is the *initial state* of $\mathcal{S}$, A is a set of *actions*, and $T \subseteq S \times A \times S$ is a *transition relation*. The transition relation captures the idea of dynamic evolution of systems whereby the execution of an action results in a change of the current state.

In the sequel, various decorations of systems are inherited by their components, e.g., the initial state of $\mathcal{S}_3$ is $\hat{s}_3$, etc., and the usual notational conventions apply. Thus, we write $p \xrightarrow{a} q$ whenever $\langle p, a, q\rangle \in T$, and call it an *atomic step* in $\mathcal{S}$. Similarly, $p \not\xrightarrow{a} q$ if $p \xrightarrow{a} q$ does not hold, and $p \xrightarrow{a}$ if $p \xrightarrow{a} q$ holds for some $q \in S$, respectively. The atomic step notation inductively extends to *paths*, i.e., finite sequences of subsequent steps. Thus, for the empty sequence $\varepsilon \in A^*$ we let $p \xrightarrow{\varepsilon} q$ iff $p = q$. For a nonempty sequence $a\varpi$ we let $p \xrightarrow{a\varpi} q$ iff there exists a state r such that $p \xrightarrow{a} r$ and $r \xrightarrow{\varpi} q$.

Morphisms of transition systems were invented to explain how the dynamic behaviour of one system is simulated within another system.

Definition 1. *A* morphism $f : \mathcal{S}_1 \to \mathcal{S}_2$ *of transition systems is a pair* $f = \langle \sigma, \alpha\rangle$ *where* $\sigma : S_1 \to S_2$ *is a total while* $\alpha : A_1 \rightharpoonup A_2$ *is a partial function which satisfy*

- $\sigma \hat{s}_1 = \hat{s}_2$.
- $p \xrightarrow{a} q$ *in* $\mathcal{S}_1$ *and* αa*-defined implies* $\sigma p \xrightarrow{\alpha a} \sigma q$ *in* $\mathcal{S}_2$.
- $p \xrightarrow{a} q$ *in* $\mathcal{S}_1$ *and* αa*-undefined implies* $\sigma p = \sigma q$ *in* $\mathcal{S}_2$.

* Partially supported by CATALYSIS, a programme within CNRS/PAN cooperation framework, and by MIUR and CNR/PAN exchange programme.

W.M.P. van der Aalst and E. Best (Eds.): ICATPN 2003, LNCS 2679, pp. 140–160, 2003.

The first condition simply says that morphisms preserve the initial states. According to the second, a step in $\mathcal{S}_1$ caused by an action observable in $\mathcal{S}_2$ via α is mapped into an atomic step of $\mathcal{S}_2$. Finally, according to the third condition, steps caused in $\mathcal{S}_1$ by actions unobservable in $\mathcal{S}_2$ via α do not have effects observable in $\mathcal{S}_2$ via σ. Together, the conditions guarantee that each *computation* of $\mathcal{S}_1$, i.e., a path $\hat{s}_1 \xrightarrow{\varpi} p$ in $\mathcal{S}_1$, gets mapped to a computation in $\mathcal{S}_2$, namely to $\hat{s}_2 \xrightarrow{\alpha\varpi} \sigma p$. One can also consider the *image* of $\mathcal{S}_1$ in $\mathcal{S}_2$ via f, i.e., a subsystem $\mathcal{S} = f(\mathcal{S}_1)$ of $\mathcal{S}_2$ defined as follows.

- $S = \{\sigma p \in S_2 \,|\, p \in S_1\}$, with $\hat{s} = \hat{s}_2$.
- $A = \{\alpha a \in A_2 \,|\, a \in A_1, \alpha a\text{-defined}\}$.
- $\sigma p \xrightarrow{\alpha a} \sigma q$ in $\mathcal{S}$ whenever $p \xrightarrow{a} q$ in $\mathcal{S}_1$.

It is convenient to introduce artificial *empty* steps $p \xrightarrow{\emptyset} p$ for each state p. Then one can think that steps caused in $\mathcal{S}_1$ by α-unobservable actions are mapped to the empty steps in $\mathcal{S}_2$. With this convention we can succinctly rewrite the second and third conditions of Def. 1 as follows.

$$p \xrightarrow{a} q \text{ in } \mathcal{S}_1 \quad \text{implies} \quad \sigma p \xrightarrow{\alpha a} \sigma q \text{ in } \mathcal{S}_2 \tag{1}$$

The notion of morphism introduced in Def. 1 seems to be the most commonly accepted in the literature. In fact, one often restricts attention to the subclass of morphisms which are total on actions. On the other hand, many models of concurrent systems, Petri nets in particular, offer a broader framework, in which *sets*, or even *bags* of actions contribute to system's evolution in the form of a complex, non-atomic step. Thus, one could consider morphisms more general than those allowed by Def. 1. The simplest would be to allow α to map an action to a set, or a bag of actions so that atomic steps in the source system get mapped to complex steps in the target system.

Formally, this generalization is correct in the sense that computations in the source get mapped to computations in the target. From this perspective an even more general notion of morphism can be considered in which actions are mapped to paths. Yet, only in the area of action refinement one can track these ideas.

The lukewarm support can be attributed to conceptual problems that come along. One of them stems from the apparent difficulty to define the image construction. Consider, for instance, a morphism $f : \mathcal{N}_1 \to \mathcal{N}_2$ of net systems. Nowadays it is uniformly accepted that this should induce a morphism $\mathbb{CG}(f) : \mathbb{CG}(\mathcal{N}_1) \to \mathbb{CG}(\mathcal{N}_2)$, i.e., a simulation of the abstract behaviour of $\mathcal{N}_1$, the *case graph* of $\mathcal{N}_1$, within the abstract behaviour of $\mathcal{N}_2$. Now, what could it mean for f to map a single event a of $\mathcal{N}_1$ to a set $\{a', a''\}$ of events of $\mathcal{N}_2$? Should we consider $\{a', a''\}$ as a new atomic event in the image of $\mathcal{N}_1$ via f? If so, would there be this connection between such complex atom and a', in case $a' = \alpha b$ for another event b of $\mathcal{S}_1$?

To convey the ideas we consider a class of simple concurrent systems. *Elementary net systems* offer a concrete and elegant model of concurrent computations. Ehrenfeucht and Rozenberg, see [7], considered a class of deterministic transition systems called *elementary transition systems*, and showed that they are exactly

the case graphs of elementary nets. More formally, they showed that for any elementary transition system $\mathcal{S}$ there exists an elementary net system $\mathcal{N} = \mathbb{SN}(\mathcal{S})$ with case graph isomorphic to $\mathcal{S}$, i.e., $\mathbb{CG}(\mathcal{N}) \simeq \mathcal{S}$. This result was among the first two constructive solutions to the problem of *synthesis* of a distributed realization $\mathbb{SN}(\mathcal{S})$ of a given abstract behaviour $\mathcal{S}$. The other solution was provided earlier by Zielonka, see [13].

Later, see [10], this result was strengthened, and the correspondence was revealed to take the form of an adjunction between the category $\mathbb{ETS}$ of elementary transition systems, and a category $\mathbb{ENS}$ of elementary net systems. More precisely, Nielsen et al., showed that the process of synthesizing an elementary net is a functor, with $\mathbb{SN} : \mathbb{ETS} \to \mathbb{ENS}$ being left adjoint and inverse to $\mathbb{CG} : \mathbb{ENS} \to \mathbb{ETS}$. Therefore, the adjunction is in fact a coreflection.

Such a close correspondence between two categories has important consequences. For instance, universal categorical constructions are preserved by the functors: colimits by the left adjoint functor $\mathbb{SN}$, and limits by the right adjoint functor $\mathbb{CG}$. Due to $\mathbb{CG}(\mathbb{SN}(\mathcal{S})) \simeq \mathcal{S}$ one can also identify elementary transition systems as a subcategory of elementary net systems.

Many constructions on nets are specified up to the properties of their case graphs, for instance by providing the specification of a system in temporal logic. It is tempting to perform all the work in the realm of transition systems. Then, if the construction was based on limits, one can call upon the coreflection, and translate the components back to the realm of Petri nets via synthesis. If the category of Petri nets is sufficiently complete, see [3], the same construction can then be performed on nets. Moreover, the case graph of the resulting net will be isomorphic to the construction performed in the category of transition systems.

Consequently, in this note we shall be concerned with Petri nets rather indirectly, especially on the technical side. The main attention will be devoted to the existence of limits in the categories of Petri net systems abstract behaviours.

Elementary transition systems satisfy the following conditions.

No short loops	$p \xrightarrow{a} q \quad \text{implies} \quad p \neq q$	(2)
No parallel arrows	$p \xrightarrow{a} q \quad \text{and} \quad p \xrightarrow{b} q \quad \text{implies} \quad a = b$	(3)
State reachability	$(\exists \varpi)\hat{s} \xrightarrow{\varpi} p$	(4)
Action reachability	$(\exists p, q)p \xrightarrow{a} q$	(5)

The essential notion necessary to define elementary transition systems and to facilitate the adjunction result is that of a region, see [7].

A *region* in $\mathcal{S}$ is a set R of states of $\mathcal{S}$ which is consistent with respect to action crossing the borders of R. Thus, $\mathrm{R} \ni p \xrightarrow{a} q \notin \mathrm{R}$ implies that for any other a-step $r \xrightarrow{a} s$ one gets the same picture: $r \in \mathrm{R} \not\ni s$. The same holds for actions entering the region. The set of all regions of $\mathcal{S}$ is denoted $\mathcal{R}_{\mathcal{S}}$. In what follows we write $\mathrm{R}^{\circ}a$, resp., $a^{\circ}\mathrm{R}$, to indicate that action a leaves, resp., enters, region R. Consequently, ${}^{\circ}a = \{\mathrm{R} \in \mathcal{R}_{\mathcal{S}} \mid \mathrm{R}^{\circ}a\}$, $\mathrm{R}^{\circ} = \{a \in A \mid \mathrm{R}^{\circ}a\}$, etc.

A transition system is *elementary* if it satisfies conditions (2)-(5) and, additionally, the following regional axioms.

$$\textit{State Separation} \qquad p \neq q \;\text{ implies }\; (\exists \mathrm{R} \in \mathcal{R}_{\mathcal{S}})\, p \in \mathrm{R} \not\ni q \tag{6}$$

$$\textit{State-Action Separation} \qquad p \not\xrightarrow{a} q \;\text{ implies }\; (\exists \mathrm{R} \in \mathcal{R}_{\mathcal{S}})\, \mathrm{R}^{\circ} a \wedge p \notin \mathrm{R} \tag{7}$$

The first regional axiom states that two different states can be separated by a region. The second axiom states that if all regions exited by a contain p then $p \xrightarrow{a}$ must hold in $\mathcal{S}$.

Transition system $\mathcal{S}$ is *deterministic* provided $p \xleftarrow{a} q \xrightarrow{a} r$ implies $p = r$. Note that from the state separation axiom it follows that each elementary transition system is deterministic. In the sequel only deterministic transition systems are considered.

2 Introduction — A Primer on Modular Synthesis of Concurrent Systems

Following Ehrenfeucht and Rozenberg an elementary net system $\mathbb{SN}(\mathcal{S})$ corresponding to a given elementary transition system $\mathcal{S}$ can be constructed as follows.

Take the regions of $\mathcal{S}$ as places. Let the set of actions of $\mathcal{S}$ be the set of events of $\mathbb{SN}(\mathcal{S})$. The flow relation F is given by $F(\mathrm{R}, a)$ iff $\mathrm{R}^{\circ} a$ and $F(a, \mathrm{R})$ iff $a^{\circ}\mathrm{R}$. Finally, declare place R to be marked initially iff $\hat{s} \in \mathrm{R}$.

The above procedure is global and unstructured — the net is constructed in a single step. Modular approaches concentrate, instead, on gradual and systematic ways of system construction. Here, two such approaches pertaining to elementary net/transition systems are recalled.

2.1 Synthesis via Action Identification

An alternative way to look at elementary net systems was put forward by Bernardinello, see [5]. Namely, one can characterise each elementary net as a *product* of simple components, so called state machines. This observation paves the way to a modular presentation of net synthesis.

A *state machine* is a reachable marked Petri net of a very simple type: each event has exactly one pre-condition and one post-condition, while the initial marking consists of a single place with one token in it. Just like in the case of elementary nets no loops are allowed, and every two elements have different pre- or post-elements. Clearly, in a state machine every reachable marking is a singleton. Consequently, the behaviour of every state machine is purely sequential — no two events can ever be fired concurrently.

The idea of the product of elementary net systems and the decomposition into such sequential components is best explained on a toy example. In the middle of Fig. 1 an elementary net system is presented. It admits decomposition into two state machine components presented on the left and on the right. The net in

the middle can be seen as a product of the two state machines. The product is computed by putting the nets side by side, separately, and then identifying events with identical names. To stay within the realm of elementary nets one should, in general, also clean things up. For instance, non-reachable places and/or events should be removed, while indistinguishable places glued together.

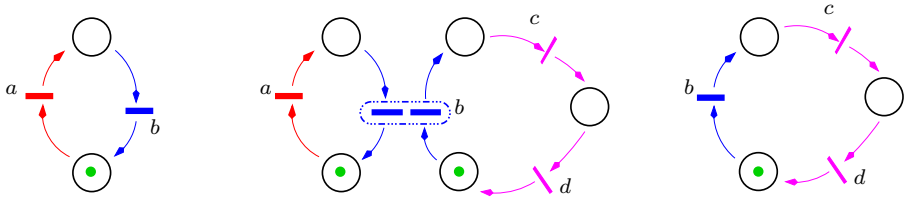

Fig. 1. Decomposition of an elementary net into sequential components.

This product, it turns out, is a categorical product in a category of elementary net systems with *rigid morphisms*, i.e., morphisms in the sense of Def. 1 which can either delete an event or map it to itself, but never rename it, see [4] for examples and details. We have already remarked that the case graph functor, as a right adjoint, preserves all limits that exist in its domain. Moreover, the adjunction cuts down to the subcategories with rigid morphisms. Thus, the case graph of the rigid product of two net systems computed in $\mathbb{ENS}$, is a rigid product of their case graphs in $\mathbb{ETS}$. This is demonstrated on Fig. 2. On the left and on the right the case graphs of the state machines from Fig. 1 are presented. Their rigid product computed in accord to Def. 2, see below, is depicted in the middle.

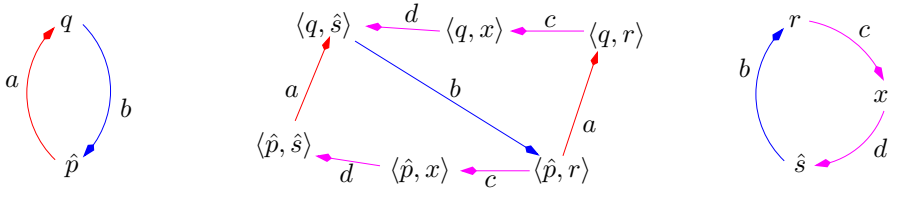

Fig. 2. Product of two elementary transition systems.

A formal definition of this product follows. Note that it applies to all transition systems, not just the elementary, and has been used in many situations since the beginning of Computing Science. One can argue, for instance that it corresponds to one of the CSP operators.

Definition 2. *Consider transition systems* $\mathbb{S}_i$, $i = 1, 2$. *Their* rigid product, *sometimes denoted* $\mathbb{S}_1 \prod^r \mathbb{S}_2$, *is the transition system* $\mathbb{S}$ *defined as follows.*

- $S = S_1 \times S_2$, *with projections* $\pi_i : S \to S_i$, $i = 1, 2$.
- $\hat{s} = \langle \hat{s}_1, \hat{s}_2 \rangle$.
- $A = A_1 \cup A_2$, *with partial projections* $\alpha_i : A \rightharpoonup A_i$, *for* $i = 1, 2$, *given by* $\alpha_i a = a$ *if* $a \in A_i$, *and* $\alpha_i a$ *undefined otherwise.*
- $p \xrightarrow{a} q$ *in* $\mathcal{S}$ *iff* $\pi_1 p \xrightarrow{\alpha_1 a} \pi_1 q$ *in* $\mathcal{S}_1$ *and* $\pi_2 p \xrightarrow{\alpha_2 a} \pi_2 q$ *in* $\mathcal{S}_2$.

By taking $\mathbb{REACH}(\mathcal{S}_1 \prod^r \mathcal{S}_2)$, *i.e., the reachable subsystem of* $\mathcal{S}$, *one obtains a rigid product in the full subcategory of reachable systems.*

Let us remark here, that for any transition system $\mathcal{S}$ there exists its maximal subsystem $\mathbb{REACH}(\mathcal{S})$ satisfying the required reachability conditions: either (4) or (5) or both. Let us state without proof the following result.

Proposition 1. *Transition systems with morphisms given in Def. 1 constitute a complete category* $\mathbb{TS}$. *Its subcategory* $\mathbb{TS}^r$ *with rigid morphisms is also complete, with binary products as given in Def 2.*

A limit in the subcategories with reachable systems is obtained by taking the reachable subsystem of the limit. This cuts down to the category of elementary systems, and its subcategory with rigid morphisms.

More about the existence of limits in the categories of transition systems follows in Sec. 4. Let us only remark that the operation of computing $\mathbb{REACH}(\mathcal{S})$ is functorial for the various reachability criteria imposed by (4), (5) or both. In fact, the functors are right adjoint to the inclusions. Thus, again, it is enough to prove that $\mathcal{S}_1 \prod^r \mathcal{S}_2$ is a product in the category of all transition systems with rigid morphisms to deduce that $\mathbb{REACH}(\mathcal{S}_1 \prod^r \mathcal{S}_2)$ is a product in the subcategory of reachable systems.

In summary, each elementary transition system can be computed by applying the rigid product constructor to *basic* elementary transition systems — those, in which every action participates in a single transition. The same does not hold for all elementary net systems. There is no problem, however, if one is interested in nets as the concurrent realisations of their abstract behaviours. For instance, the saturated net systems considered by Ehrenfeucht and Rozenberg ([7]), and those constructed from minimal regions following Bernardinello ([5]) can be constructed from state machines by identification of common events, i.e., by repeated application of the rigid product construction.

2.2 Synthesis via Identification of Regions

Recently, a novel modular composition technique applicable to elementary net and transition systems has been proposed, see [6]. The idea is to glue together two systems on pairs of *complementary* places/regions.

We refrain from explaining the details of the construction as defined on nets, consult [6] for details. Instead, the example on Fig. 3 demonstrates the idea at work. Consider the two nets $\mathcal{N}_1$ and $\mathcal{N}_2$ in the upper corners of the figure, each with the distinguished pair of places denoted R and $\bar{\text{R}}$. These places are complementary in the sense that the pre-events of R are the post-events of the

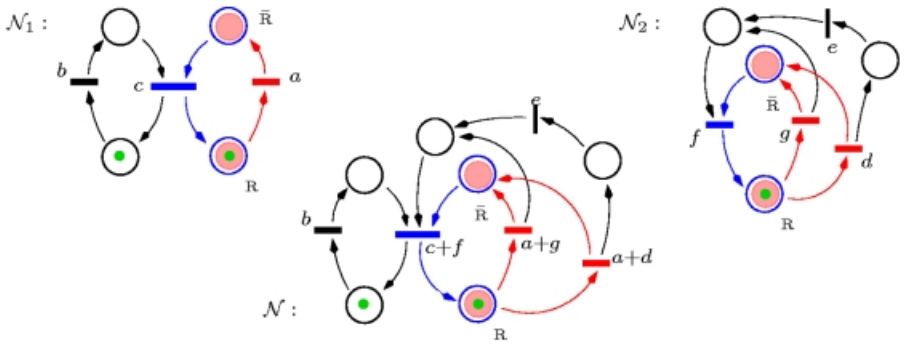

Fig. 3. Composition of two elementary nets by identification of places.

corresponding $\bar{\text{R}}$, and vice versa. It is required that both places R are initially marked, and thus places $\bar{\text{R}}$ are not. Net $\mathcal{N}$ (in the middle) shows the effect of the composition. Again, we start by putting disjoint copies of the two nets side by side. Then, both places R are identified, and so are those labelled $\bar{\text{R}}$.

The events, like b or e, which neither input nor output from R and $\bar{\text{R}}$ are not affected. The flow relation for such events is inherited from the components. Other events, like a, must either empty R and fill $\bar{\text{R}}$, or, like c, fill R and empty $\bar{\text{R}}$. We take all matching pairs of such events, like $a+g$ and $c+f$, with one event from each component. Note that in this way duplicate copies of some events may give rise to several complex events, viz. $a+g$ and $a+d$. The flow relation for the complex events is the union of flows of their elements.

Finally, if one is interested in elementary nets one should restrict attention to reachable places and events.

Looking at Fig. 3 one can easily see how the original nets can be recovered from their composition. This can be formalized by establishing suitable morphisms from the resulting net to the components.

This method of building nets from simpler components is quite intuitive. It is mimicked by a suitable operation on the transition systems as shown on Fig. 4. In the upper corners we see the case graphs of the two component nets presented on Fig. 3. Regions corresponding to the distinguished places are also depicted. The transition system in middle of Fig. 4 is the result of the composition. The details of its construction are described in Def. 3. Note that, following [6], elementary transition systems are composable on regions only under certain assumptions.

Definition 3. *Assume that elementary transition systems* $\mathcal{S}_1$ *and* $\mathcal{S}_2$ *are* composable on regions R_1 and R_2*, i.e., that the following conditions are satisfied.*

- $\hat{s}_1 \in \text{R}_1$ *and* $\hat{s}_2 \in \text{R}_2$.
- $\{a \in A_1 \mid \text{R}_1{}^\circ a\} = \emptyset$ *iff* $\{a \in A_2 \mid \text{R}_2{}^\circ a\} = \emptyset$.
- $\{a \in A_1 \mid a^\circ \text{R}_1\} = \emptyset$ *iff* $\{a \in A_2 \mid a^\circ \text{R}_2\} = \emptyset$.

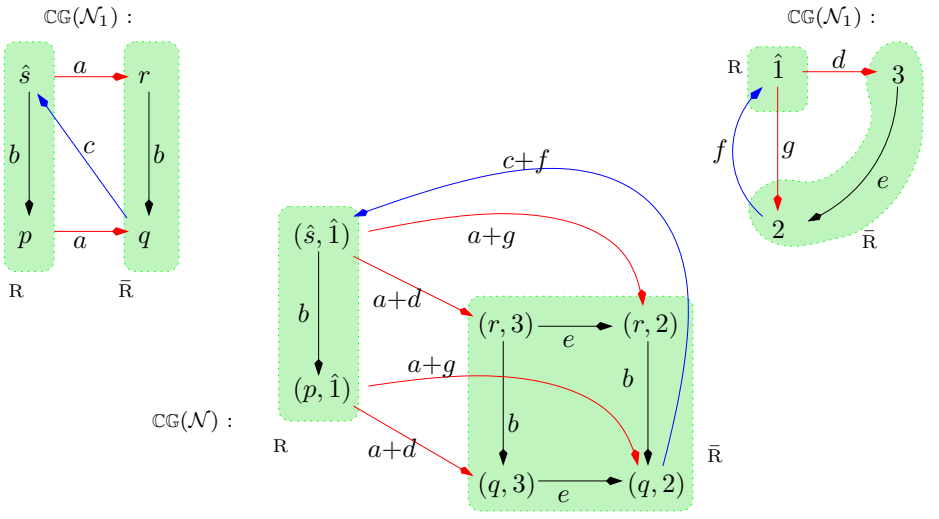

Fig. 4. Composition of two elementary transition systems by identification of places.

Then, consider $\mathcal{S}$ *defined as follows, where* $\bar{\text{R}}_i = S_i \setminus \text{R}_i$, *for* $i = 1, 2$.

- $S = (\text{R}_1 \times \text{R}_2) \cup (\bar{\text{R}}_1 \times \bar{\text{R}}_2)$. *The corresponding projections* $\pi_i : S \to S_i$ *are defined by* $\pi_i\langle p_1, p_2\rangle = p_i$, *for* $i = 1, 2$.
- $\hat{s} = \langle \hat{s}_1, \hat{s}_2\rangle$.
- $A = A_1 \setminus ({\text{R}_1}^\circ \cup {}^\circ\text{R}_1) \cup A_2 \setminus ({\text{R}_2}^\circ \cup {}^\circ\text{R}_2) \cup ({\text{R}_1}^\circ \times {\text{R}_2}^\circ) \cup ({}^\circ\text{R}_1 \times {}^\circ\text{R}_2)$. *The corresponding partial projections* $\kappa_i : A \to A_i$ *are defined by* $\kappa_i a = a$, *if* $a \in A_i$ *and undefined otherwise, and* $\kappa_i\langle a_1, a_2\rangle = a_i$, *for* $i = 1, 2$.
- $\langle p_1, p_2\rangle \xrightarrow{a} \langle q_1, q_2\rangle$ *iff* $p_1 \xrightarrow{\kappa_1 a} q_1$ *and* $p_2 \xrightarrow{\kappa_2 a} q_2$.

Finally, composition of $\mathcal{S}_1$ *and* $\mathcal{S}_2$ *on regions* R_1 *and* R_2, *denoted* $\mathcal{S}_1[\text{R}, \bar{\text{R}}]\mathcal{S}_2$, *is the result of taking the reachable subsystem of* $\mathcal{S}$.

One can show that the composition of two composable elementary transition systems is elementary. The main result of [6] is that the two composition operations, of elementary net systems and elementary transition systems agree: $\mathbb{CG}(\mathcal{N}_1[\text{R}, \bar{\text{R}}]\mathcal{N}_2) \simeq \mathbb{CG}(\mathcal{N}_1)[\text{R}, \bar{\text{R}}]\mathbb{CG}(\mathcal{N}_2)$, i.e., the case graph of the composed net is isomorphic to the composition of the case graphs. Until now, however, there was no categorical explanation of the result. The reasons for that failure are discussed in the following section.

3 Synchronisation as a Pullback?

Clearly, $\langle \pi_i, \kappa_i\rangle : \mathcal{S}_1[\text{R}, \bar{\text{R}}]\mathcal{S}_2 \to \mathcal{S}_i$, constitute morphisms of transition systems for $i = 1, 2$. The composability conditions implicitly refer to a transition system **2**

presented on the top of Fig. 5, and the two morphisms presented there. System **2** captures the essence of dividing the state space of a transition system into two complementary regions, with two actions corresponding to moving out from the region to its complement and back. The state components σ_i send all states in the corresponding region R to the state R of **2**, and the states in the region $\bar{\text{R}}$ to the state $\bar{\text{R}}$ of **2**. The action components α_1 map all actions crossing from R to $\bar{\text{R}}$ in $\mathcal{S}_i$ to action *out*, actions crossing R in the opposite direction are mapped to *in*. All other actions that do not cross R are α_i-undefined.

More generally, the choice of a region R, with $\hat{s} \in$ R, in *any* elementary transition system $\mathcal{S}$ uniquely determines a morphism $f : \mathcal{S} \to \mathbf{2}$ defined as above. Conversely, *any* region in a transition system $\mathcal{S}$ is the inverse image of the state component of a morphism $f : \mathcal{S} \to \mathbf{2}$. In this sense system **2** is the *type* of elementary nets, see [1].

The square of morphisms on Fig. 5 is commutative. This strengthens the intuition that system $\mathcal{S}_1[\text{R}, \bar{\text{R}}]\mathcal{S}_2$ is in fact a *synchronisation* of $\mathcal{S}_1$ and $\mathcal{S}_2$ induced by a specific choice of regions. Equivalently, it is a synchronisation induced by a choice of abstractions of $\mathcal{S}_1$ and $\mathcal{S}_2$ within **2** as a common interface.

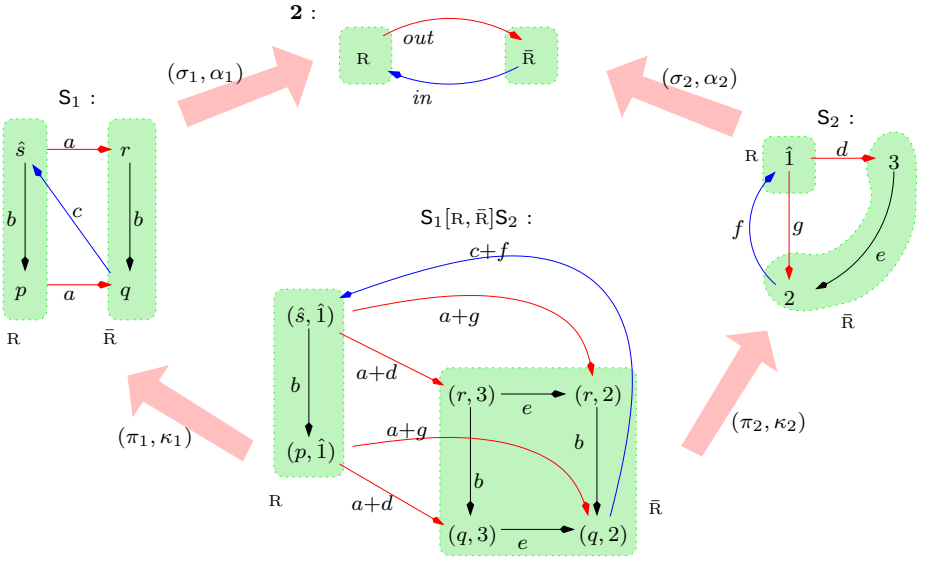

Fig. 5. System $\mathcal{S}_1[\text{R}, \bar{\text{R}}]\mathcal{S}_2$ as a synchronisation of $\mathcal{S}_1$ and $\mathcal{S}_2$ via interface **2**.

There is an evident analogy between this synchronisation operation, and the construction of a *pullback* — one of the universal categorical limits. To see this consider Fig. 6.

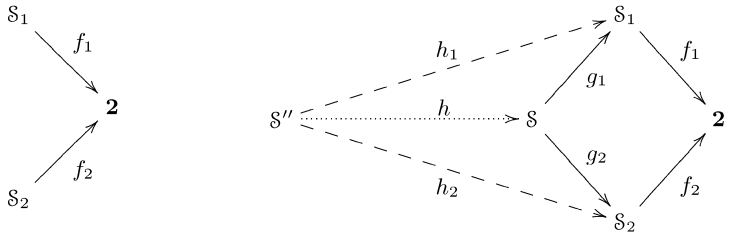

Fig. 6. The interface of synchronisation and the pullback

The starting point of the construction of $\mathcal{S}_1[\mathrm{R}, \bar{\mathrm{R}}]\mathcal{S}_2$ is presented on the left, where we are given two morphisms $f_1 : \mathcal{S}_1 \to \mathbf{2}$ and $f_2 : \mathcal{S}_2 \to \mathbf{2}$ with a common *synchronisation interface* **2**. Clearly, this presentation generalises the compatibility conditions given in Def 3. The same situation is at the beginning of pullback construction. The difference is that one is not willing to accept *any* solution, i.e., a system $\mathcal{S}$ with a pair of *projections* $g_1 : \mathcal{S} \to \mathcal{S}_1$ and $g_2 : \mathcal{S} \to \mathcal{S}_2$ which make the square commute. This merely captures that the solution *adheres* to the restrictions imposed by the interface. One strives to find $\mathcal{S}$ which is an *optimal* solution. Formally, given any other solution $\mathcal{S}''$ with a pair of morphisms $h_1 : \mathcal{S}'' \to \mathcal{S}_1$ and $h_2 : \mathcal{S}'' \to \mathcal{S}_2$ which adhere to the interface, there should exist a unique mediating morphism $h : \mathcal{S}'' \to \mathcal{S}$ through which both h_1 and h_2 would factorize: $g_1 \circ h = h_1$ and $g_2 \circ h = h_2$. The situation is depicted on the right of Fig. 6

Now, it is natural to ask whether the synchronisation operation proposed in [6] is an instance of a pullback construction. An answer can be given, provided we fix a category in which the pullbacks are sought. A natural choice would be to consider transition systems with the class of morphisms described in Def. 1. This category is complete. This is a good news, since the pullbacks can be constructed for all interfaces. Sadly, it turns out that the pullback construction computed in this category does not coincide with the synchronisation operation proposed by Bernardinello et al., as described in Def. 3. The difference is vividly demonstrated on Fig. 7. The result $\mathcal{S}_1[\mathrm{R}, \bar{\mathrm{R}}]\mathcal{S}_2$ of synchronisation of $\mathcal{S}_1$ and $\mathcal{S}_2$, is recalled on the left. Their pullback, with respect to the same interface, computed in this category, is described on the right of Fig. 7. Clearly, $\mathcal{S}_1[\mathrm{R}, \bar{\mathrm{R}}]\mathcal{S}_2$ is a proper subsystem of the pullback, and so we have hidden in the pullback the labels of all transitions except the new one. There is one new complex action in the pullback which contributes to one new transition — the one drawn with thick dashed arrow.

Having demonstrated that the synchronisation construction is not universal one is left with two options. First, if one believes that the category is *the* right one, perhaps the best idea is to forget about the construction. Most likely the construction will not have any interesting properties. The second option is to investigate other categories in which the construction might turn out to be universal.

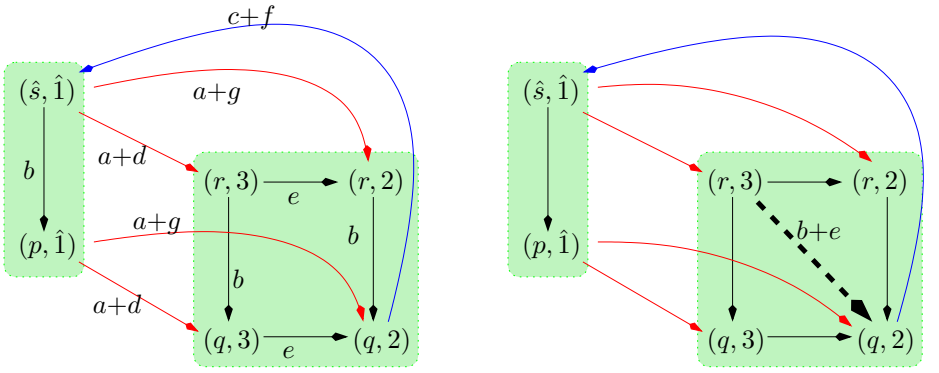

Fig. 7. Synchronisation versus pullback in $\mathbb{TS}$.

In our case one can see that the pullback construction behaves somewhat strangely. The interface does not impose any restrictions between action b from the component $\mathcal{S}_1$, and action e from the component $\mathcal{S}_2$. Thus, it would be most natural to have them implemented as independent events, compare Fig. 3. Yet, additionally, the pullback construction introduces an action $b{+}e$ which, via projections, corresponds to a *simultaneous* execution of these two, seemingly independent actions.

Thus, one would want to work within a model in which the tight synchronisation $b{+}e$ could be prohibited. At the same time we would like to keep actions like $a{+}d$ or $c{+}f$ which also are tight synchronisations of actions from the two components. These, seemingly contradictory requirements can be met in the category studied in the next Section.

4 Asynchronous Systems with Step Semantics

Let us formally introduce a more general model. We could continue with elementary transition systems, but there would be a price to pay. First, elementary transition systems are conceptually less suitable than *asynchronous systems*, see [12,2], the model we are about to introduce. This is reflected not only in the number of axioms imposed in both cases. More importantly, the crucial notion of *action independence* is brought forward only in the definition of asynchronous systems, whereas it remains hidden in the structure of elementary transition systems.

Definition 4. *An* asynchronous system *is a tuple* $\mathcal{A} = (S, \hat{s}, A, \|, T)$ *where* $\mathcal{S} = (S, \hat{s}, A, T)$ *is a deterministic transition system* underlying $\mathcal{A}$*, and* $\| \subseteq A \times A$ *is an irreflexive and symmetric binary relation of* independence*, provided the following* swap *property holds.*

$$p \xrightarrow{a} q \xrightarrow{b} r \quad \textit{and} \quad a \parallel b \quad \textit{implies} \quad p \xrightarrow{b} s \xrightarrow{a} r \quad \textit{for some } s \in S. \tag{8}$$

Note that an elementary transition system $\mathcal{S}$ gives rise to an asynchronous system when the independence relation is defined as follows.

$$a \parallel b \quad \text{iff} \quad p \xrightarrow{a} q \xrightarrow{b} \text{ in } \mathcal{S} \quad \text{and} \quad \xleftarrow{a} r \xrightarrow{b} \text{ in } \mathcal{S}$$

It can be shown that the asynchronous systems obtained by taking $\parallel$ are *concrete*, i.e., they are rigid products of their sequential components, see [4] for details.

For a binary relation like $\parallel \subseteq A \times A$ consider the set $\mathsf{Cliques}(\parallel)$ of cliques of mutually $\parallel$-related elements.

$$\mathsf{Cliques}(\parallel) = \{x \subseteq A \mid (\forall a, b \in x) a \neq b \Rightarrow a \parallel b\}$$

Note that $\emptyset, \{a\} \in \mathsf{Cliques}(\parallel)$. Thus, identifying elements of A with the corresponding singleton cliques the relation $\parallel$ can be conservatively extended to $\parallel \subseteq \mathsf{Cliques}(\parallel) \times \mathsf{Cliques}(\parallel)$ by letting $x \parallel y$ iff $a \parallel b$ for all $a \in x$ and $b \in y$. Clearly, $x \parallel \emptyset$ holds for any $x \in \mathsf{Cliques}(\parallel)$.

The idea behind the independence of actions, as captured by the swap property (8), is that two independent actions can occur concurrently whenever they can occur one after another. Thus, we can extend the atomic step relation to arbitrary sets of mutually independent actions as follows.

$$p \xrightarrow{x} q \quad \text{iff} \quad p \xrightarrow{\varpi} q \quad \text{for some linearization } \varpi \text{ of } x \in \mathsf{Cliques}(\parallel)$$

Note that due to (8), $p \xrightarrow{\varpi} q$ holds for some linearization ϖ of x iff it holds for *any* linearization of x. Also, $p \xrightarrow{\emptyset} p$ holds for any p. In the sequel, notation $p \xrightarrow{a} q$ and $p \xrightarrow{a+c} q$ is used rather than the formally correct $p \xrightarrow{\{a\}} q$ and $p \xrightarrow{\{a,c\}} q$, respectively.

One-safe Petri nets give rise to asynchronous systems. Indeed, the marking graph of any P/T net is a deterministic transition system. Then, if $\mathcal{N}$ is 1-safe, one can associate with each net an independence relation in two universal ways. One way is to choose the *dynamic independence* and say that two events a and b are independent in $\mathcal{N}$ if a step $a + b$ can be fired from a reachable marking. The other is to choose *structural independence* and declare a and b as independent if the pre- and post-conditions of a are disjoint with the pre- and post-conditions of b.

An asynchronous system with empty independence relation is just a deterministic transition system. Let us also look at conservative extensions of morphisms as given in Def. 1.

One such conservative extension has already been proposed, cf. [2]. The idea is to add the requirement that the action part of a morphism weakly preserves concurrency in the sense that if $a \parallel_1 b$ and both αa and αb are defined then $\alpha a \parallel_2 \alpha b$ holds. With our convention extending independence relation to steps the same can be more succinctly described as follows.

$$a \parallel_1 b \quad \text{implies} \quad \alpha a \parallel_2 \alpha b \tag{9}$$

Clearly, if the independence is empty, the above holds trivially.

It turns out that asynchronous systems and morphisms introduced above form a complete category. An example of a product, i.e., the pull-back of two systems with respect to the unique morphisms to the terminal object of the category, is depicted on Fig. 8. The actions from the components of the product

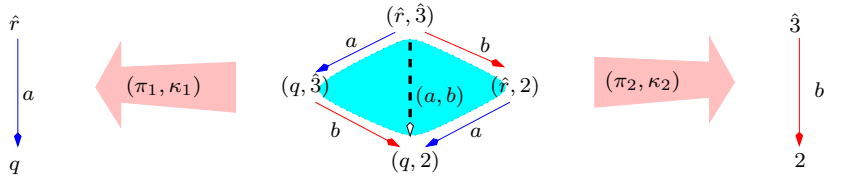

Fig. 8. A product of asynchronous systems in the category with flat morphisms.

are preserved in the product as independent actions. Unfortunately, their synchronisation is also added. So, the example demonstrates the same unwelcome phenomenon that we have faced before, cf. Fig. 7.

The notion recalled above is not, however, the only possible conservative extension of the notion of a morphism of deterministic transition systems. Another, more liberal proposal follows.

Definition 5. *Let* $\mathcal{A}_i = (S_i, \hat{s}_i, A_i, \|_i, T_i)$ *be two asynchronous systems, for* $i = 1, 2$. *Their* synchronising morphism $f : \mathcal{A}_1 \to \mathcal{A}_2$ *is a pair* $f = \langle \sigma, \alpha \rangle$ *where*

- $\sigma : S_1 \to S_2$ *preserves the initial states:* $\sigma \hat{s}_1 = \hat{s}_2$.
- $\alpha : A_1 \to \mathsf{Cliques}(\|_2)$ *preserves independence:* $a \parallel_1 b$ *implies* $\alpha a \parallel_2 \alpha b$

Together they should map atomic steps of $\mathcal{A}_1$ *into steps of* $\mathcal{A}_2$ *as follows.*

- $p \xrightarrow{a} q$ *in* $\mathcal{A}_1$ *implies* $\sigma p \xrightarrow{\alpha a} \sigma q$ *in* $\mathcal{A}_2$.

Thus, the only difference in reference to the classical definition is that instead of sending an action to null or to another action, a possibility to send it to a set of mutually independent actions is added. Preservation of the independence relation means that not only atomic, but also each multi-step in the source system is mapped to a multi-step in the target system.

The completeness of the category of asynchronous systems equipped with synchronising morphisms is investigated in the next section. Let us finish by stating some basic facts.

Proposition 2. *Asynchronous systems with synchronising morphisms as given in Definition 5 form a category* $\mathbb{AS}$.

The subcategory $\mathbb{AS}^f$ *with* flat *morphisms obtained by imposing that the cardinality of* αa *is not greater than* 1 *is isomorphic to the category of asynchronous systems studied in [2].*

Let us finally remark that in the richer category in which non-flat morphisms are allowed the product of the systems depicted in Fig. 8 will be the same diamond, but without the extra action (a, b). The details of the constructions of limits in this, and in related categories are studied in the sequel sections.

5 Pullbacks, and Other Limits

Computing a pullback, one of the universal categorical constructions, can be seen as a generalisation of binary products. Indeed, assume that a category admits a *terminal object* $\mathbf{1}$, i.e., that for any object $\mathcal{A}$ there exists a unique morphism $! : \mathcal{A} \to \mathbf{1}$. Given objects $\mathcal{A}_1$ and $\mathcal{A}_2$ assume that there exists a pullback $p_i : \mathcal{A} \to \mathcal{A}_i$, $i = 1, 2$, of the morphisms $!_1 : \mathcal{A}_1 \to \mathbf{1}$ and $!_2 : \mathcal{A}_2 \to \mathbf{1}$. It follows then easily that $\mathcal{A}$ is a categorical product of $\mathcal{A}_1$ and $\mathcal{A}_2$.

Conversely, products can be used to compute pullbacks, provided the category admits equalizers. Let us remind that given a pair of parallel morphisms $f, g : \mathcal{A}' \to \mathcal{A}''$ their *equalizer* is a morphism $\mathbf{eq}(f, g) : \mathcal{A} \to \mathcal{A}'$, which equalizes f and g, i.e., $f \circ \mathbf{eq}(f, g) = g \circ \mathbf{eq}(f, g)$, and does it in the optimal way, i.e., any other morphism equalizing f and g factors uniquely through $\mathbf{eq}(f, g)$. The construction of a pullback via product and equalizer is described on Fig. 9. Thus,

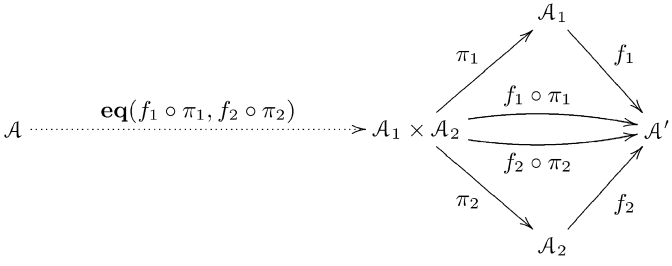

Fig. 9. Pullback constructed from product and equalizer

given $f_i : \mathcal{A}_i \to' \mathcal{A}'$, $= 1, 2$, first compute the categorical product $\mathcal{A}_1 \times \mathcal{A}_2$ with projections π_1 and π_2. Then, enforce commutation of the square by equalizing the morphisms $f_1 \circ \pi_1$ and $f_2 \circ \pi_2$.

The above observation is valid in any category. In fact, from products and equalizers one can compute arbitrary finite limits, see [8]. Our idea is to apply this characterisation of pullbacks to demonstrate the existence of the pullbacks of asynchronous systems with synchronising morphisms.

An asynchronous system consists of two components, while a morphism comprises two maps for manipulating the components. Consequently, one can view the category defined above as being built over two simpler *component* categories. One, concerning the state part, is the category $\mathbb{SET}_\star$ of pointed sets $(S, \hat{s})$, with $\hat{s} \in S$, and functions between the sets which preserve the distinguished points.

The other category, concerning the action part, is the category $\mathbb{SET}_{\|}$ of sets with independence relation $(A, \|)$, and multi-functions $\alpha : A_1 \to \mathsf{Cliques}(\|_2)$ which preserve the independence relation. Clearly, forgetting about actions, and hence the independence and transition relations, results in a forgetful functor from $\mathbb{AS}$ to $\mathbb{SET}_{\star}$. Similarly, a forgetful functor from $\mathbb{AS}$ to $\mathbb{SET}_{\|}$ is obtained.

In the context of the present section it is useful to learn how the limits are constructed in the component categories. The reason is simple — as we shall see the limits in the category of asynchronous systems are reflected by these forgetful functors.

5.1 Pullbacks in $\mathbb{SET}_{\star}$

Binary product of $(S_1, \hat{s}_1)$ and $(S_2, \hat{s}_2)$ in $\mathbb{SET}_{\star}$ is just the cartesian product $S_1 \times S_2$ with the pair $\langle \hat{s}_1, \hat{s}_2 \rangle$ as the designated element.

The construction of equalizer is also simple. Given $\sigma, \varsigma : (S_1, \hat{s}_1) \to (S_2, \hat{s}_2)$ consider $S = \{ s \in S_1 \mid \sigma s = \varsigma s \}$. Clearly, $\hat{s}_1 \in S$. It is immediate that the inclusion $\imath : S \hookrightarrow S_1$ equalizes σ and ς. The inclusion is also optimal in the sense, that *any* $\rho : S' \to S_1$ that equalizes σ and ς necessarily factors through $\imath$, and does it in a unique way. Namely, the image $\{ \rho s' \in S_1 \mid s' \in S' \}$ of S' via ρ is included in S, since $\sigma(\rho s') = \varsigma(\rho s')$ by assumption. Thus, the characterisation of pullbacks in $\mathbb{SET}_{\star}$ described on Fig. 9 gives the following result.

Proposition 3. *Let $\sigma_i : (S_i, \hat{s}_i) \to (S', \hat{s}')$, for $i = 1, 2$. Then $(S, \langle \hat{s}_1, \hat{s}_2 \rangle)$ where $S = \{ \langle p_1, p_2 \rangle \in S_1 \times S_2 \mid \sigma_1 p_1 = \sigma_2 p_2 \}$ with projections $\varsigma_i : S \to S_i$ given by $\varsigma_i \langle s_1, s_2 \rangle = s_i$, $i = 1, 2$, is a pullback in $\mathbb{SET}_{\star}$.*

5.2 Products in $\mathbb{SET}_{\|}$

Characterisation of finite limits in $\mathbb{SET}_{\|}$ is more complicated. Let us start by introducing some notions and notation, and making a number of observations.

Given an object $(A, \|)$ in $\mathbb{SET}_{\|}$ and $x, y \in \mathsf{Cliques}(\|)$ let us say that x and y are *consistent* whenever $x \cup y \in \mathsf{Cliques}(\|)$.

Lemma 1. *Let $x, y \in \mathsf{Cliques}(\|)$ and consider $\alpha : (A, \|) \to (A', \|')$ in $\mathbb{SET}_{\|}$. Then*

1. *$x \| y$ iff $x \cup y \in \mathsf{Cliques}(\|)$ and $x \cap y = \emptyset$.*
2. *$x \| y$ implies $\alpha x, \alpha y \in \mathsf{Cliques}(\|')$ and $\alpha x \|' \alpha y$.*
3. *$x \cup y \in \mathsf{Cliques}(\|)$ implies $\alpha(x \cap y) = \alpha x \cap \alpha y$.*

Lemma 1(3) says that morphisms in $\mathbb{SET}_{\|}$ preserve *consistent* meets. Without this proviso this is not valid. For instance, let $A = \{a, b, c\}$ with $a \| b \| c$ and $A' = \{d, e\}$ with $d \| e$. Then $\alpha : A \to A'$ given by $\alpha a = \alpha c = e$ and $\alpha b = d$ is a valid morphism. However, $\{d\} = \alpha(\{a, b\} \cap \{b, c\}) \neq \alpha\{a, b\} \cap \alpha\{b, c\} = \{d, e\}$.

Now, with products the situation is rather easy.

Consider $(A_1, \|_1)$ and $(A_2, \|_2)$. Without loss of generality one can assume $A_1 \cap A_2 = \emptyset$. Their product $(A, \|)$ in $\mathbb{SET}_{\|}$ is given by $A = A_1 \cup A_2$ and $\| = \|_1 \cup$

$\|_2 \cup A_1 \times A_2 \cup A_2 \times A_1$. Clearly, $\|$ is symmetric and irreflexive. Define projections $\pi_i : A \to A_i$, for $i = 1, 2$, as follows.

$$\pi_i a = \begin{cases} \emptyset & a \notin A_i \\ \{a\} & a \in A_i \end{cases}$$

Note that $\pi_i x = x \cap A_i$, $i = 1, 2$ follows from the above definition.

Lemma 2. $(A, \|)$ *with projections defined above is a product of* $(A_1, \|_1)$ *and* $(A_2, \|_2)$ *in* $\mathbb{SET}_\|$.

Let us also notice that the projections are flat.

5.3 Equalizers in $\mathbb{SET}_\|$

With equalizers the situation is more complex. Let $\alpha, \beta : (A_1, \|_1) \to (A_2, \|_2)$.

Again, we would like to take the sub-object of $(A_1, \|_1)$ which is equalized by α and β. The problem is, however, that it is not enough to restrict attention to the elements of A_1 on which α and β agree.

Example 1. Put $A_1 = \{a, b\}$ with $a \|_1 b$, and let $(A_1, \|_1) = (A_2, \|_2)$. Consider α and β given by $\alpha b = \beta a = b$ and $\alpha a = \beta b = a$. One morphism equalizing α and β is obtained by taking the embedding of the empty set, with the empty independence relation. Another is given by taking $A = \{\star\}$ and $\lambda\star = a{+}b$. The other is, in fact, an equalizer of α and β in $\mathbb{SET}_\|$.

Thus, even when α and β agree on no singleton, they may nevertheless agree on some larger cliques, and this leads to non-trivial equalizers.

Let us therefore consider $\mathcal{X}_{\alpha\beta} = \{x \in \mathsf{Cliques}(\|_1) \mid \alpha x = \beta x\}$.

Lemma 3. $\mathcal{X}_{\alpha\beta}$ *is closed with respect to consistent: meets, sums and differences.*

From lemma 3 it follows that *atoms*, i.e., non-empty minimal elements of $\mathcal{X}_{\alpha\beta}$, are irreducible in the following sense, see [3].

Corollary 1. *Let* $x, y, z \in \mathcal{X}_{\alpha\beta}$, x *an atom, satisfy* $x, y \subseteq z$. *Then either* $x \subseteq y$ *or* $x \subseteq z \setminus y$. *Any two different consistent atoms* $x, y \in \mathcal{X}_{\alpha\beta}$ *are disjoint. Thus,* $x \|_1 y$.

As a consequence one obtains that each element y of $\mathcal{X}_{\alpha\beta}$ admits unique decomposition into a sum of disjoint atoms of $\mathcal{X}_{\alpha\beta}$. In what follows $Atoms(\mathcal{X}_{\alpha\beta})$ denotes the set of all atoms of $\mathcal{X}_{\alpha\beta}$. Also, let $\mathcal{X}^y_{\alpha\beta} = \{x \in Atoms(\mathcal{X}_{\alpha\beta}) \mid x \subseteq y\}$ for all $y \in \mathcal{X}_{\alpha\beta}$.

Proposition 4. *Every* $y \in \mathcal{X}_{\alpha\beta}$ *admits* $\mathcal{X}^y_{\alpha\beta}$ *as a unique decomposition into a sum of disjoint atoms of* $\mathcal{X}_{\alpha\beta}$.

The existence of unique decompositions of the elements of $\mathsf{Cliques}(\|_1)$ equalized by α and β paves the way for the construction of equalizers. Namely, consider $(Atoms(\mathcal{X}_{\alpha\beta}), \|)$ with $x \parallel y \Leftrightarrow x \parallel_1 y$.

Thus, the idea is to consider the atoms of $\mathcal{X}_{\alpha\beta}$ as new actions, the hitherto complex structure of which will now be forgotten. Actually, we do keep track of their past in the evaluation mapping $[\alpha, \beta] : Atoms(\mathcal{X}_{\alpha\beta}) \to \mathsf{Cliques}(\|_1)$ defined by $[\alpha, \beta]x = x$.

The independence between new atoms is inherited from independence of the cliques of mutually independent actions in $(A_1, \|_1)$. Therefore, $[\alpha, \beta]$ is in fact a morphism $[\alpha, \beta] : (Atoms(\mathcal{X}_{\alpha\beta}), \|) \to (A_1, \|_1)$ in $\mathbb{SET}_{\|}$.

Proposition 5. *$[\alpha, \beta]$ is an equalizer of α and β in $\mathbb{SET}_{\|}$.*

Let us note that Example 1 demonstrates that even if both, α and β, are flat morphisms, their equalizer $[\alpha, \beta]$ in the category with synchronising morphisms may fail to be flat.

5.4 Pullbacks in $\mathbb{SET}_{\|}$

The construction of products and equalizers can be used, as described on Fig. 9, to compute pullbacks. Putting the constructions together result in the following elementary characterisation of pullbacks in the category $\mathbb{SET}_{\|}$. Since the categorical constructions are defined up to isomorphism we assume, without loss of generality, that the objects in the pullback situation are disjoint.

Definition 6. *Let $\alpha_1 : (A_1, \|_1) \to (A', \|') \leftarrow (A_2, \|_2) : \alpha_2$ and assume that $A_1 \cap A_2 = \emptyset$. Define $(A, \|)$ as follows.*

- $A = \{\{a_1\} \mid a_1 \in A_1, \alpha_1 a_1 = \emptyset\} \cup \{\{a_2\} \mid a_2 \in A_2, \alpha_2 a_2 = \emptyset\}$
$$\cup \left\{ x_1 \cup x_2 \;\middle|\; \begin{array}{l} x_i \in \mathsf{Cliques}(\|_i),\ \textit{for } i = 1, 2, \alpha_1 x_1 = \alpha_2 x_2 \neq \emptyset, \\ \forall y_1, y_2 \textit{ such that } x_1 \supseteq y_1 \neq \emptyset \neq y_2 \subseteq x_2 \\ \qquad \alpha_1 y_1 = \alpha_2 y_2 \Rightarrow y_1 = x_1 \wedge y_2 = x_2 \end{array} \right\}$$
with projections $\kappa_i : A \to \mathsf{Cliques}(\|_i)$ given by $\kappa_i(x_1 \cup x_2) = x_i$, for $i = 1, 2$.
- $x \parallel y$ *iff* $\kappa_1 x \parallel_1 \kappa_1 y$ *and* $\kappa_2 x \parallel_2 \kappa_2 y$.

The following result is almost immediate.

Proposition 6. *$(A, \|)$ together with $\kappa_i : (A, \|) \to (A_i, \|_i)$, $i = 1, 2$, is a pullback of $\alpha_i : (A_i, \|_i) \to (A', \|')$, $i = 1, 2$ in $\mathbb{SET}_{\|}$. If α_1 and α_2 are flat then so are κ_1 and κ_2.*

5.5 Pullbacks in the Category of Asynchronous Systems

We have shown the existence and characterized the construction of binary products and equalizers in the component categories $\mathbb{SET}_{\star}$ and $\mathbb{SET}_{\|}$. In fact, in both categories the construction of binary products easily generalises to arbitrary products. This guarantees the existence of arbitrary limits, see [8], i.e., both categories are complete.

Moreover, the forgetful functors from the category $\mathbb{AS}$ of asynchronous systems to the component categories $\mathbb{SET}_\star$ and $\mathbb{SET}_\parallel$ together *reflect* the limits. That is, to construct a limit of a diagram in $\mathbb{AS}$ one can compute the limits in the component categories, and then equip the results with an appropriate transition relation. Rather than proving such a general result let us verify that it works for pullbacks.

The following provides an elementary characterisation of pullbacks.

Definition 7. *Let $f_i = \langle \sigma_i, \alpha_i \rangle : \mathcal{A}_i \to \mathcal{A}'$, for $i = 1, 2$, be such that $A_1 \cap A_2 = \emptyset$. Consider $\mathcal{A}$ defined as follows.*

- $S = \{ \langle p_1, p_2 \rangle \in S_1 \times S_2 \mid \sigma_1 p_1 = \sigma_2 p_2 \}$,
  together with projections $\varsigma_i : S \to S_i$ given by $\varsigma_i \langle s_1, s_2 \rangle = s_i$ for $i = 1, 2$.
- $\hat{s} = \langle \hat{s}_1, \hat{s}_2 \rangle$.
- $A = \{ \{a_1\} \mid a_1 \in A_1, \alpha_1 a_1 = \emptyset \} \cup \{ \{a_2\} \mid a_2 \in A_2, \alpha_2 a_2 = \emptyset \}$
  $\cup \left\{ x_1 \cup x_2 \;\middle|\; \begin{array}{l} x_i \in \mathsf{Cliques}(\|_i),\ \textit{for } i = 1, 2, \alpha_1 x_1 = \alpha_2 x_2 \neq \emptyset, \\ \forall y_1, y_2 \textit{ such that } x_1 \supseteq y_1 \neq \emptyset \neq y_2 \subseteq x_2 \\ \qquad \alpha_1 y_1 = \alpha_2 y_2 \Rightarrow y_1 = x_1 \wedge y_2 = x_2 \end{array} \right\}$
  with projections $\kappa_i : A \to \mathsf{Cliques}(\|_i)$ given by $\kappa_i(x_1 \cup x_2) = x_i$, $i = 1, 2$.
- $a \parallel b$ *iff* $\kappa_1 a \parallel_1 \kappa_1 b$ *and* $\kappa_2 a \parallel_2 \kappa_2 b$.
- $p \xrightarrow{a} q$ *iff* $\varsigma_1 p \xrightarrow{\kappa_1 a} \varsigma_1 q$ *and* $\varsigma_2 p \xrightarrow{\kappa_2 a} \varsigma_2 q$.

The following result is now straightforward.

Theorem 1. *$\mathcal{A}$ constructed above is an asynchronous system which, together with $g_i = \langle \varsigma_i, \kappa_i \rangle : \mathcal{A} \to \mathcal{A}_i$, for $i = 1, 2$, forms a pullback in the category of asynchronous systems with synchronising morphisms. The pullback in the full subcategory of reachable systems is obtained by taking $\mathbb{REACH}(\mathcal{A})$, i.e., the reachable part of $\mathcal{A}$. Morphisms g_1 and g_2 are flat whenever f_1 and f_2 are both flat.*

6 Applications and Future Work

The remaining part of the paper demonstrates the utility of the construction of pullbacks in the category of asynchronous systems with synchronising morphisms. In particular, we show that the two, seemingly quite different methods of system synthesis discussed in Sec. 2.1 and in Sec. 2.2 are in fact special cases of pullbacks.

6.1 Rigid Product as a Pullback

We have argued in section 2.1 that elementary systems, both transition systems and nets, can be synthesised by conjoining a number of simple sequential systems via rigid product construction, cf. Def. 2 and Prop. 1. Morin was the first to notice that a large class of asynchronous systems can be characterised as rigid products of their sequential components in a category of state-reachable asynchronous systems with rigid flat morphisms, see [9]. This result has been used in the studies of the functorial synthesis of 1-safe Petri nets with labelled transitions

from asynchronous systems, see [4]. There, the need to restrict attention to rigid morphisms proved to be a severe limitation. Here, we show that rigid products are instances of pullback construction in the richer category of asynchronous systems with synchronising morphisms. Since Def. 2 deals with deterministic systems, we should start by generalising it to asynchronous systems first. The following extension simply adds the missing description of the independence relation.

Definition 8. *Consider asynchronous systems $\mathcal{A}_1$ and $\mathcal{A}_2$ Their* rigid product *is the asynchronous system $\mathcal{A}$ defined as follows.*

- $S = S_1 \times S_2$, *with projections* $\pi_i : S \to S_i$, $i = 1, 2$.
- $\hat{s} = \langle \hat{s}_1, \hat{s}_2 \rangle$.
- $A = A_1 \cup A_2$, *with partial projections* $\alpha_i : A \rightharpoonup A_i$, *for* $i = 1, 2$, *given by* $\alpha_i a = a$ *if* $a \in A_i$, *and* $\alpha_i a = \emptyset$ *otherwise.*
- $a \parallel b$ *iff* $\alpha_1 a \parallel_1 \alpha_1 b$ *and* $\alpha_2 a \parallel_2 \alpha_2 b$.
- $p \xrightarrow{a} q$ *in* $\mathcal{S}$ *iff* $\pi_1 p \xrightarrow{\alpha_1 a} \pi_1 q$ *in* $\mathcal{S}_1$ *and* $\pi_2 p \xrightarrow{\alpha_2 a} \pi_2 q$ *in* $\mathcal{S}_2$.

By taking $\mathbb{REACH}_\sigma(\mathcal{A})$, *the state-reachable subsystem of* $\mathcal{A}$, *one obtains a rigid product in the full subcategory of reachable systems.*

Now, consider $\mathcal{A}_1$ and $\mathcal{A}_2$. Their rigid product $\mathcal{A}$ as described in Def. 8 forces synchronisation of the components on common actions. It is also worth noting that as a result of the composition certain actions which were declared as independent in $\mathcal{A}_1$, say, may become dependent in the result, since they were not independent in $\mathcal{A}_2$.

Now, consider $\mathbf{1}_{A_1 A_2} = (\{\star\}, \star, A_1 \cap A_2, \|_\star, T_\star)$ defined as follows.

$$a \parallel_\star b \quad \text{iff} \quad a \neq b \qquad \text{and} \qquad \star \xrightarrow{a} \star, \quad \text{for all } a \in A_1 \cap A_2$$

Clearly, $\mathbf{1}_{A_1 A_2}$ is an asynchronous system. Moreover, the system can be used to synchronise $\mathcal{A}_1$ and $\mathcal{A}_2$ via morphisms $f_i = \langle !_i, \alpha_i \rangle : \mathcal{A}_i \to \mathbf{1}_{A_1 A_2}$ defined as follows for $i = 1, 2$.

$$!_i s = \star \qquad \text{and} \qquad \alpha_i a = \begin{cases} a & a \in A_1 \cap A_2 \\ \emptyset & a \notin A_1 \cap A_2 \end{cases}$$

Proposition 7. *The rigid product of asynchronous systems $\mathcal{A}_1$ and $\mathcal{A}_2$ defined in Def. 8 is a pullback of $f_1 : \mathcal{A}_1 \to \mathbf{1}_{A_1 A_2} \leftarrow \mathcal{A}_2 : f_2$.*

6.2 Synchronisation as a Pullback

The synchronisation of elementary transition systems via identification of regions was motivated and introduced in section 2.2. In section 3 we have discussed the apparent affinity of this idea to the idea of pullback. We have also demonstrated that the construction put forward by Bernardinello et al. is not a pullback in

any category of transition/asynchronous systems with *flat* morphisms. Nevertheless, this construction turns out to be a pullback in the richer category with synchronising morphisms.

Indeed, consider the following pullback situation

$$f_1 : \mathcal{A}_1 \to \mathbf{2} \leftarrow \mathcal{A}_2 : f_2 \tag{10}$$

where $\mathbf{2} = (\{\mathrm{R}, \bar{\mathrm{R}}\}, \mathrm{R}, \{in, out\}, \emptyset, T)$ is the two state elementary transition system described on Fig. 5. The interface system $\mathbf{2}$ is sequential, i.e., the independence relation is empty. Thus, morphisms $f_1 = \langle \sigma_1, \alpha_1 \rangle$ and $f_2 = \langle \sigma_2, \alpha_2 \rangle$ are necessarily flat. Moreover, each of them determines a pair of complementary regions $\mathrm{R}_i = \sigma_i^{-1}(\mathrm{R})$ and $\bar{\mathrm{R}}_i = \sigma_i^{-1}(\bar{\mathrm{R}})$, for $i = 1, 2$.

Proposition 8. *Let the transition systems $\mathcal{S}_1$ and $\mathcal{S}_2$ underlying $\mathcal{A}_1$ and $\mathcal{A}_2$ from (10) be elementary and composable on regions R_1 and R_2, cf. Def. 3. Then, the reachable transition system underlying the pullback of f_1 and f_2 is isomorphic to $\mathcal{S}_1[\mathrm{R}, \bar{\mathrm{R}}]\mathcal{S}_2$.*

6.3 Future Work

The importance of the method of putting systems together by means of synchronising their activities on common actions has been recognized very early. It is one of the basic constructors in Hoare's CSP, as well as in several process algebras. It was also one of the first composition operations studied by Arnold and Nivat. Nowadays, it is commonly found in the area of DES synthesis and control. As noticed by Morin this operation is the corner-stone of synthesis for a large class of concurrent systems ([9]) and their Petri net realisations ([4]).

The above together with another, seemingly quite different method of putting systems together proposed by Bernardinello et al., have been shown here to be instances of *pullback*, a conceptually simple categorical construction.

This observation prompts several natural generalisations. Firstly, one can use the full power of pullbacks and allow arbitrary interfaces for system synchronisation, not just $\mathbf{2}$ or $\mathbf{1}_{A_1 A_2}$. Secondly, one can use limits more general than pullbacks for system development. For instance, one can consider putting together three components $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$, synchronised via two interfaces $\mathcal{A}'$ and $\mathcal{A}''$ given by $f_i : \mathcal{A}_i \to \mathcal{A}'$ and $g_j : \mathcal{A}_j \to \mathcal{A}''$, for $i = 1, 2, 3$ and $j = 1, 2$. The task now would be to find a systems $\mathcal{A}$ which would be a simultaneous synchronisation of $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$ which fulfills the restrictions imposed by the choice of f_i and g_j, for $i = 1, 2, 3$ and $j = 1, 2$, *all* at the same time. This, is clearly possible if limits more general than pullbacks are allowed.

As demonstrated here, one can play the same game with pullbacks, and indeed with arbitrary finite limits, in the much larger class of behaviours of asynchronous systems. We also believe that the results of [3] can be easily extended to cope with the more liberal notion of morphisms of Petri nets envisaged here. Therefore, the resulting category of safe Petri nets will also turn out to be finitely complete. Then, continuity of the case graph functor would guarantee that the

case graph of a net constructed as a limit from some finite diagram of nets will be isomorphic to the result of the same pullbacks applied to the case graphs of the respective nets.

In the categorical characterisation of rigid products, cf. Prop. 7, and composition by region identification, cf. Prop. 8, the projections and the morphisms involved in the constructions are flat. Yet, within the classical framework with flat morphisms the pullback construction offers results different than expected, see Fig. 8. This observation seems to offer an interesting insight. Namely, one is tempted to believe that an elegant abstract characterisation of synchronisation can only be achieved when *truly concurrent* semantics of systems is called upon, and the synchronising morphisms are allowed into the picture. Better understanding this facet is also worth further investigations.

References

1. Badouel, E., and Ph. Darondeau. Dualities between nets and automata induced by schizophrenic objects. Proc. 6th Intnl. Conf. CTCS, LNCS **953**, pp.: 24–43, Springer-Verlag, 1995.
2. Bednarczyk, M. A. *Categories of asynchronous systems*. PhD Thesis, University of Sussex, 1–88, 1988.
3. Bednarczyk, M. A, Borzyszkowski, A. M. and R. Somla. Finite completeness of categories of Petri nets. *Fundamenta Informaticae*, **43**, 1–4, pp.: 21–48, 2000.
4. Bednarczyk, M. A., and A. M. Borzyszkowski. On concurrent realization of reactive systems and their morphisms. H. Ehrig et al. (eds.) *Unifying Petri Nets — Advances in Petri nets*, LNCS **2128**, pp.: 346–379, Springer-Verlag, 2001.
5. Bernardinello, L. Synthesis of net systems. Proc. *Application and Theory of Petri Nets*, LNCS **691**, pp.: 89–105, Springer-Verlag, 1993.
6. Bernardinello, L, Ferigato, C and L. Pomello. Towards modular synthesis of EN systems. In B. Caillaud et al. (eds.) *Synthesis and Control of Discrete Event Systems*, 103–113, Kluwer Academic Publishers, 2002.
7. Ehrenfeucht, A and G. Rozenberg. Partial (set) 2 structures, I & II. *Acta Informatica*, **27**, 4, pp.: 315–368, 1990.
8. MacLane, S. *Categories for the Working Mathematician*. Graduate Texts in Mathematics, Springer-Verlag, 1971.
9. Morin, R. Decompositions of asynchronous systems. In *Proc. CONCUR'98*, LNCS **1466**, pp. 549–564. Springer, 1998.
10. Nielsen, M., Rozenberg, G. and P.S. Thiagarajan. Elementary transition systems. *Theoretical Computer Science*, **96**, 1, pp.: 3–32, 1992.
11. Nielsen, M., Rozenberg, G. and P.S. Thiagarajan. Elementary transition systems and refinement. *Acta Informatica*, **29**, pp.: 555–578, 1992.
12. Shields, M. *Deterministic asynchronous automata*. In E. J. Neuhold and G. Chroust (Eds.) *Formal Methods in Programming*, pp. 317–345, North-Holland, 1985.
13. Zielonka, W. Notes on finite asynchronous automata. *RAIRO, Informatique Théoretique et Applications*, vol. 21, pp.: 99–135, 1987.