A partial trace semantics for Petri nets*

V. Diekert

Institut für Informatik, Universität Stuttgart, Breitwiesenstrasse 20–22, D-70565 Stuttgart, Germany

Abstract


We introduce the notion of partial trace. A partial trace is an equivalence class of some labelled partial orders over a dependence alphabet (Σ, SΔ). Partial traces arise in a natural way by the synchronization of semi-traces. They form a monoid which is shown to be free partially commutative. We prove an embedding theorem which shows that any partial trace has a canonical representation as a tuple of words. We then apply this concept to Petri nets. We define the behavior of a P/T-system in terms of partial traces. We consider local morphisms between Petri nets and we show how this relates to the partial trace behavior. In particular, we obtain the desired result that the partial trace behavior of a synchronized system is the synchronization of its local behavior.

1. Introduction

Since the initiating work of Mazurkiewicz, traces have been successfully applied to Petri nets [11, 12]. This is particularly clear for elementary net systems due to the fact that for such systems the dependence relation is a priori symmetric and fixed by the static net topology. For P/T-systems (Petri nets) traces do not cope with the dynamic behavior of the system. A first step is to allow an asymmetric dependence (or independence) relation which leads to the notion of semi-trace, see e.g. [13].

Unfortunately, this does not solve the general problem. Therefore, one may consider semi-words as a suitable partial-order semantics for P/T-systems. A semi-word is a labelled partial order without auto-concurrency. Here we propose a slightly

*This research has been partly supported by the ESPRIT Basic Research Actions No. 6317 ASMICS II.

Correspondence to: V. Diekert, Institut für Informatik, Universität Stuttgart, Breitwiesenstrasse 20-22, D-70565 Stuttgart, Germany. Email: diekert@informatik.uni-stuttgart.de.

0304-3975/94/$07.00 © 1994—Elsevier Science B.V. All rights reserved
SSDI 0304-3975(94)00005-4
different approach. We collect semi-words into one equivalence class, if they induce the same set of semi-traces. Such an equivalence class is called a partial trace. The procedure to define a partial trace from a semi-word is exactly the same as one obtains a trace from a word. One basic fact which makes partial traces attractive is that the operation of synchronization behaves well, contrary to the case of semi-traces.

The set of partial traces forms a monoid in a natural way. Since we will see that Levi's lemma holds and partial traces have a length, we can prove that this monoid is in fact free partially commutative. Thus, algebraically it has the same structure as a usual trace monoid. One difference however is that the monoid of partial traces is infinitely generated, in general. On the other hand, we can prove an embedding theorem which generalizes the projection lemma \[5, 3\]. It yields a canonical embedding into a finite direct product of free monoids on two letters, only. As an application, we can represent any partial trace as a tuple of words. This allows, for example, an easy implementation of various basic algorithms on partial traces (which will work in linear time over a fixed alphabet). Once we have developed an abstract theory of partial traces, we define the behavior of a P/T-system in terms of partial traces. In the presentation, we follow \[6, Ch. 3\] where a concept of local morphisms was introduced and studied for Mazurkiewicz traces. In the present paper, we show that this concept can be fully generalized to the notion of partial trace. As a corollary we obtain the result that the partial trace behavior of a synchronized system is the synchronization of its local behavior.

2. Semi-traces

Let \( \Sigma \) be a finite alphabet. A semi-dependence relation (over \( \Sigma \)) is a reflexive relation \( SD \subseteq \Sigma \times \Sigma \), the pair \((\Sigma, SD)\) is called a semi-dependence alphabet. It will be represented as a directed graph without drawing the self-loops. A semi-dependence gives rise to a semi-commutation by the following set of rules: \( SC = \{ab \Rightarrow ba | (a, b) \in SD, a, b \in \Sigma \} \). The definition of a semi-trace induced by a word \( u \in \Sigma^* \) is the set of words derivable from \( u \) by application of semi-commutation rules:

\[
[u] = \{ v \in \Sigma^* | u \xrightarrow{SC} v \}.
\]

The topic of semi-commutations and semi-traces has been subject of several papers \[2-4, 14, 7, 10\]. Here we are going to generalize this concept. However, before we do so, we recall some more facts on semi-traces. The set of semi-traces is denoted by \( \mathcal{M}(\Sigma, SD) \); it is a monoid by \( [u] \cdot [v] = [uv] \) and the neutral element \( [1] = \{ 1 \} \), where \( 1 \) is the empty word. In order to verify that the formula \( [u] \cdot [v] = [uv] \) is well-defined, it is enough to see that we have \( [u] = [v] \) if and only if \( u \xrightarrow{SC} v \). It follows that, from
A partial trace semantics for Petri nets

In order to simplify reading, we will usually drop the prefix semi in the following. Thus, in this paper a dependence alphabet can be asymmetric and trace means semi-trace.

If we want to emphasize the fact that we deal with a trace over a symmetric dependence relation, we shall speak of a Mazurkiewicz trace. The additional aspect of traces instead of Mazurkiewicz traces is that we have a nontrivial order relation between traces. We can write $[u] \leq [v]$ if $[u] \subseteq [v]$, or what is the same if $v \not\rightarrow u$. Thus, $\mathcal{M}(\Sigma, SD)$ is endowed with a nondiscrete topology where the closed sets are the downward closed subsets.

In the following, we will use a graphical representation of traces. Let $u = a_1 \ldots a_n \in \Sigma^*$ be a word, $a_i \in \Sigma$ for $1 \leq i \leq n$. Then the generated trace $[u]$ is given by the labelled acyclic graph $[V, E, \lambda]$ where the set of vertices is any $n$-point set, say $V = \{1, \ldots, n\}$ with the labelling $\lambda(i) = a_i$, and arcs are from $i$ to $j$ if both $i < j$ and $(a_i, a_j) \in SD$.

In order to see whether or not a given directed graph represents a trace it is useful to introduce the so-called soft arcs: Let $[V, E, \lambda]$ be any labelled acyclic graph without self-loops such that $(x, y) \in E$ implies $(\lambda(x), \lambda(y)) \in SD$ for all $x, y \in V$. Then for all vertices $x, y \in V$, $x \neq y$, such that $(x, y) \not\in E$ but $(\lambda(x), \lambda(y)) \in SD$, we introduce a soft arc from vertex $y$ to $x$.

The idea of a soft arc from $y$ to $x$ is that in the original word representation of the semi-trace $y$ has been before $x$, because otherwise the arc $(x, y)$ would have been visible in the edge set $E$.

A given directed graph $[V, E, \lambda]$ is now a trace if and only if both

- arcs are between dependent vertices only and
- the graph together with its soft arcs is acyclic.

Note that for a trace $[V, E, \lambda]$ either we have $(x, y) \in E$ or $(y, x) \in E$ whenever $x \neq y$ and $(\lambda(x), \lambda(y)) \in SD \cap SD^{-1}$. The operation of synchronization is partially defined. It takes two traces $s_1 \in \mathcal{M}(\Sigma_1, SD_1)$, $s_2 \in \mathcal{M}(\Sigma_2, SD_2)$ and may yield a trace $s \in \mathcal{M}(\Sigma_1 \cup \Sigma_2, SD_1 \cup SD_2)$. A necessary precondition of synchronization is that for all $a \in \Sigma_1 \cap \Sigma_2$ the number of occurrences of $a$ in $s_1$ and $s_2$ is the same. In the following, we denote the number of occurrences of a letter $a$ in a labelled graph $s$ by $|s|_a$. Let $s_i = [V_i, E_i, \lambda_i]$, $i = 1, 2$, and assume that $|s_1|_a = |s_2|_a$ for all $a \in \Sigma_1 \cap \Sigma_2$. Then we can identify the $n$th $a$ in $V_1$ with the $n$th $a$ in $V_2$ for $a \in \Sigma_1 \cap \Sigma_2$, $1 \leq n \leq |s_i|_a$, and we may define the union of both graphs over this identification. (Note that for a standard representation of $s_i$ with vertex set $V_i = \{(a, n) \mid a \in \Sigma, 1 \leq n \leq |s_i|_a\}$, $i = 1, 2$, this union is the set-theoretical union.) We obtain a labelled graph $s = [V_1 \cup V_2, E_1 \cup E_2, \lambda_1 \cup \lambda_2]$. If this graph contains any cycle then certainly $s_1$ and $s_2$ are not synchronizable. If this graph is acyclic, then we would like to say that this graph is the synchronization. However, we may have constructed a labelled acyclic graph which is no semi-trace anymore. This will be shown in the example below. It is a phenomenon which we do not meet when dealing with symmetric dependence relations, i.e., with Mazurkiewicz...
Fig. 1. A dependence alphabet with asymmetric dependencies.

\[
\begin{align*}
\Sigma, SD & = (a, b, c, d) \\
c & \leftarrow d \\
\end{align*}
\]

Fig. 2. The synchronization \([cabd]\) \([bdca]\) does not yield any (semi-) trace.

traces. Thus, synchronization can be seen as the basic motivation that a generalization to asymmetric dependence relations forces us to consider even more general objects than (semi-)traces.

**Example.** Let \(SD = \{(a, b), (a, c), (d, b), (d, c)\}\), i.e., \((\Sigma, SD)\) is as in Fig. 1. The same picture of Fig. 1 represents the trace \(s = [abdc]\). Consider the traces \(s_1 = [cabd]\) and \(s_2 = [bdca]\). As a graph theoretical synchronization, we obtain the partial order of Fig. 2 which is not a trace anymore due to the cycle induced by the soft arcs from \(c\) to \(a\) and from \(b\) to \(d\). However, it makes sense to say that \(s_1\) and \(s_2\) are synchronizable, since the three traces \([abdc],[dcab],[adbc]\) are possible executions which respect the ordering of \(s_1\) and \(s_2\). We only have to look for a suitable domain where \(s_1 \parallel s_2\) is defined.

3. Partial traces

Following the terminology of Starke [15], a semi-word over \(\Sigma\) is a pomset (partially ordered multi-set) without auto-concurrency, i.e., vertices with the same label are ordered. (In [1] a semi-word is called a PTrace.) Semi-words are a widely adopted model to express the semantics of concurrent systems. The restriction that vertices with the same label must be ordered is technically a strong but helpful restriction. For example, it implies that a semi-word over a fixed alphabet can always be stored in place linear to the number of vertices instead of square. This makes linear-time algorithms available with respect to the number of vertices. Our formal definition of a semi-word over \(\Sigma\) is a labelled partial order \([V, \leq, \lambda]\) where we require that the
A partial trace semantics for Petri nets

91

restriction of the ordering to the set of vertices with the same label is well-ordered. Since we deal here only with finite structures, well-order means linear. (The above general definition leads in the direction to include infinite and in fact transfinite objects.) The idea of partial traces is to abstract from a semi-word the ordering between independent actions. A partial trace over \((\Sigma, SD)\) is therefore an equivalence class of semi-words. We say that two semi-words \(s_1\) and \(s_2\) are equivalent if they become equal (i.e., isomorphic) after the following procedure which simply forgets the ordering between independent vertices:

1. Represent a semi-word as a labelled acyclic graph.
2. Cancel all arcs \((x, y)\) where \((\lambda(x), \lambda(y)) \notin SD\).
3. Consider the new induced labelled partial order.

Note that the above procedure puts two semi-words in the same class of a partial trace exactly as one obtains a usual trace starting from words. Thus, this procedure abstracts from a semi-word in the same way as a trace abstracts from a (totally ordered) word.

Thus, we may view a partial trace as a set of semi-words or as a set of P-traces in the terminology of Arnold [1], where CCI-sets of P-traces are studied. The set of partial traces is denoted by \(P(\Sigma, SD)\) in the following. Of course, if \(SD = \Sigma \times \Sigma\) is full, then \(P(\Sigma, SD)\) coincides with the set of semi-words which is denoted by \(SW(\Sigma)\).

Before we start our investigations on \(P(\Sigma, SD)\), it is useful to have a unique normal form for partial traces. We present a partial trace as a directed labelled acyclic graph \([V, E, \lambda]\) with the following restrictions on the edge set \(E\):

1. If \(x \neq y\) and \(\lambda(x) = \lambda(y)\), then either \((x, y) \in E\) or \((y, x) \in E\), but not both.
2. If \((x, y) \in E\), then we have \((\lambda(x), \lambda(y)) \in SD\).
3. If \((\lambda(x), \lambda(y)) \in SD\), \(x \neq y\), and there is a path from \(x\) to \(y\), then \((x, y) \in E\).

The above graph is called the dependence graph of the partial trace. It is clear that this is a unique representation of a partial trace. It is also clear that the whole information is contained in its Hasse diagram. Thus, it is enough to show the Hasse diagram.

There is a natural ordering between partial traces. We define \(p \leq q\) if the dependence graph of \(p\) is the same as the dependence graph of \(q\), but \(p\) may have more arcs. The semantics of \(p \leq q\) is that \(p\) is less concurrent than \(q\). Thus, if \(p < q\) then \(p\) allows less (sequential) executions than \(q\) does.

The following proposition is easy and relates the equality of partial traces to the equality of sets of traces.

**Proposition 3.1.** Let \(p, q \in P(\Sigma, SD)\). Then, we have \(p = q\) if and only if \(\{s \in M(\Sigma, SD) | p \succeq s\} = \{s \in M(\Sigma, SD) | q \succeq s\}\).

The set of partial traces \(P(\Sigma, SD)\) forms a monoid with the following concatenation: Let \([V_1, E_1, \lambda_1]\), \([V_2, E_2, \lambda_2]\) be dependence graphs. Then define \([V, E, \lambda] = [V_1, E_1, \lambda_1] \cdot [V_2, E_2, \lambda_2]\) by the disjoint union together with new arcs from all \(x_1\) in \(V_1\) to \(x_2\) in \(V_2\) such that \((\lambda_1(x_1), \lambda_2(x_2)) \in SD\), i.e., \(V = V_1 \cup V_2\), \(E = E_1 \cup E_2 \cup \{(x_1, x_2) \in V_1 \times V_2 | (\lambda_1(x_1), \lambda_2(x_2)) \in SD\}\), and \(\lambda(x) = \lambda_i(x)\) for \(x \in V_i\), \(i = 1, 2\). Note that this
concatenation extends immediately to \(\omega\)-products, or more general, to any \(\alpha\)-power where \(\alpha\) is an ordinal; we simply have to allow infinite partial traces.

**Corollary 3.2.** \(\mathcal{M}(\Sigma, \mathcal{SD})\) is a submonoid of \(\mathcal{P}(\Sigma, \mathcal{SD})\) and this embedding respects the ordering. Moreover, \(\mathcal{M}(\Sigma, \mathcal{SD})\) is the image of the canonical homomorphism \(\varphi: \Sigma^* \to \mathcal{P}(\Sigma, \mathcal{SD})\).

**Proposition 3.3.** Let \((\Sigma_1, \mathcal{SD}_1), (\Sigma_2, \mathcal{SD}_2)\) be disjoint dependence alphabets. Then \(\mathcal{P}(\Sigma_1 \cup \Sigma_2, \mathcal{SD}_1 \cup \mathcal{SD}_2)\) is isomorphic to the direct product \(\mathcal{P}(\Sigma_1, \mathcal{SD}_1) \times \mathcal{P}(\Sigma_2, \mathcal{SD}_2)\).

It should be mentioned that for the theory of Mazurkiewicz traces there is an analogue of Proposition 3.3 with respect to the complex products for graphs and free products, see [6, Proposition 1.4.2]. For partial traces such an analogue does not exist.

The key observation on partial traces is that Levi’s lemma holds. For this it is convenient to introduce the notion of independence. Two partial traces \(s, t\) are called independent, if neither \((a, b) \in \mathcal{SD}\) nor \((b, a) \in \mathcal{SD}\) for all labels \(a\) of \(s\) and all labels \(b\) of \(t\). We also write \((s, t) \in I\) in this case.

**Lemma 3.4** (Levi’s lemma). Let \(x, y, z, t \in \mathcal{P}(\Sigma, \mathcal{SD})\) be partial traces such that \(xy = zt\). Then there exist \(r, u, v, s \in \mathcal{P}(\Sigma, \mathcal{SD})\) such that \(x = ru\), \(y = vs\), \(z = rv\), \(t = us\), and \((u, v) \in I\).

**Proof.** Represent \(xy = zt\) as a dependence graph. In this graph we may identify \(x, y, z, t\) as subgraphs. Define \(r = x \cap z\), \(u = x \cap t\), \(v = y \cap z\), \(s = y \cap t\). Then indeed \(r, u, v, s\) are dependence graphs and it holds \(x = ru\), \(y = vs\), \(z = rv\), and \(t = us\). To see that \((u, v) \in I\), consider a vertex of \(u\) with label \(a\) and a vertex of \(v\) with label \(b\). Assume \((a, b) \in \mathcal{SD}\), then there is an arc between these \(a, b\) in the graph \(xy = zt\). However, this would mean that there is some arc from \(t\) to \(z\) which is impossible. Hence, \((a, b) \notin \mathcal{SD}\) and by symmetry \((b, a) \notin \mathcal{SD}\), too. \(\square\)

Clearly, we can define a length function of partial traces by the number of vertices. Since we have just seen that Levi’s lemma holds, a result of Duboc [8] applies (see also e.g. [6, Theorem 1.53]). We obtain the following theorem.

**Theorem 3.5.** Let \((\Sigma, \mathcal{SD})\) be a dependence alphabet. Then the monoid of partial traces \(\mathcal{P}(\Sigma, \mathcal{SD})\) is free partially commutative.

In general, this monoid is not finitely generated. For example, consider the monoid \(\mathcal{P}(a - b)\). This means we have \(\Sigma = \{a, b\}\) and a full dependence relation \(\{(a, a), (a, b), (b, a), (b, b)\}\). Clearly, the independence relation is empty. Hence, \(\mathcal{P}(a - b)\) is free by Theorem 3.5. (In the following we denote partial traces sometimes by vectors
A partial trace semantics for Petri nets

over words, if the components are pairwise independent.) Among others, generators of $P(a - b)$ are the graphs.

$$\begin{pmatrix}a^n \\ b^n\end{pmatrix}, \; m, n \geq 0$$

and no arcs between $a$’s and $b$’s or vice versa. Thus, $P(a - b)$ is an infinitely generated free monoid.

What happens if we consider $P(a \rightarrow b)$, i.e., instead of a full dependence relation, we have $(a, b) \in SD$ but $(b, a) \notin SD$.

Clearly, by the same argument (if $s$ and $t$ are independent partial traces then at least one is empty) this monoid must be free, again by Theorem 3.5. However the interesting fact is that $P(a \rightarrow b)$ is finitely generated. More precisely, we have a canonical identification $P(a \rightarrow b) = \{a, b\}^*$. Since this fact becomes important below, we state this as a proposition. It will allow us to embed the monoid of partial traces into a direct product of finitely generated free monoids in a natural way.

**Proposition 3.6.** The canonical morphism $\varphi: \{a, b\}^* \rightarrow P(a \rightarrow b)$ is an isomorphism.

**Proof.** Since $P(a \rightarrow b)$ is free and not commutative it is enough to see that $\varphi$ is surjective. Let $s$ be a partial trace such that $|s|_a = m, |s|_b = n$. We show that $s$ is in the image of $\varphi$. For this, we may assume that $m \geq 1$ and $n \geq 1$. In a partial trace of $P(a \rightarrow b)$ there are no arcs from any $b$ to any $a$. Choose $j$ maximal, $0 \leq j \leq n$ such that there is no arc from the first $a$ to the $j$th $b$. Then we have a factorization $s = \varphi(b^j a)t$ for some $t \in P(a \rightarrow b)$ of smaller size. By induction, it follows that $\varphi$ is surjective, and hence the result. The translation from words to partial traces and vice versa can also be deduced from the example in Fig. 3. □

Recall from Corollary 3.2 that $M(\Sigma, SD)$ is always a submonoid of $P(\Sigma, SD)$. The next theorem characterizes those dependence alphabets where we have $M(\Sigma, SD) = P(\Sigma, SD)$. (The simplest nontrivial case was given in Proposition 3.6.)

**Theorem 3.7.** Let $(\Sigma, SD)$ be a dependence alphabet and $G = (\Sigma, D)$ be the underlying undirected graph with vertex set $\Sigma$ and edge set $\{xy \mid (x, y) \in SD \cup SD^{-1}, x \neq y\}$. Then the

![Fig 3. The partial trace $\varphi(a^3b^2bab^2a) \in P(a \rightarrow b)$.](image-url)
following assertions are equivalent:
(i) \( M(C, SD) = P(C, SD) \),
(ii) \( P(C, SD) \) is finitely generated,
(iii) the undirected graph \( G \) has no cycles and \((\Sigma, SD)\) has no symmetric dependency.
(The latter condition means \( SD \cap SD^{-1} = \text{id}_\Sigma \).)

Proof. (i)\( \Rightarrow \) (ii): trivial, (ii)\( \Rightarrow \) (iii): If \( P'(C, SD) \) is finitely generated, then we have seen above that \((\Sigma, SD)\) must not contain any undirected edge \((a, b) \in SD \cap SD^{-1} \setminus \text{id}_\Sigma \).
Assume now by contradiction that \( G \) contains a cycle of minimal length \((a_1, a_2, \ldots, a_n)\) with \( n \geq 3 \). Since this cycle has more than three edges, at least two must have the same direction with respect to \((\Sigma, SD)\). Hence, we may assume that \((a_1, a_r) \in SD\) and \((a_{m+1}, a_m) \in SD\) for some \( 1 \leq m \leq n - 1 \). For \( k \geq 1 \), consider the partial trace \( p_k \), which is induced by the word \( a_1 a_2 \ldots a_r \), but where the arcs from all \( a_1 \)'s to \( a_r \) are omitted. This is possible since \((a_m, a_{m+1}) \notin SD\). Factorize \( p_k = q_1 \ldots q_r \) into a product with \( r \) maximal and \( q_i \neq 1 \) for \( 1 \leq i \leq r \). It is enough to show that we have \( r = 1 \), since then all \( p_k \) are necessary generators for \( k \geq 1 \). Let \( 1 \leq i \leq n - 1 \). A vertex with the label \( a_i \) belongs to some factor \( q_j \) and the vertex with the label \( a_{i+1} \) belongs to some factor \( q_{j'} \). Whether or not \((a_i, a_{i+1}) \in SD\), we must have \( j < j' \). Therefore, \( a_i \in q_1 \) and \( a_{i+1} \in q_r \). If we would have \( r > 1 \), then there would be an arc from \( a_1 \) to \( a_n \). However, this is impossible, due to the definition of \( p_k \), and hence \( r = 1 \).

(iii)\( \Rightarrow \) (i): Recall the notion of soft arc from Section 2 and assume that some \( p \in \mathcal{P}(\Sigma, SD) \setminus M(\Sigma, SD) \) exists. Adding soft arcs to \( p \), we obtain a directed cycle, since \( p \) is not a trace. It is easy to see that we may assume that all letters of this cycle are different; hence, we obtain a cycle \((a_1, \ldots, a_n)\) such that \( a_i \in \Sigma, a_i \neq a_j \) for \( i \neq j \), and \((a_i, a_{i+1}) \in (SD \cup SD^{-1})\) for all \( i \mod n \). If \( n = 2 \), then we found a symmetric dependency \((a_1, a_2) \in (SD \cap SD^{-1}) \setminus \text{id}_\Sigma \). If \( n \geq 3 \), then this is a cycle in the underlying undirected graph \( G \). \( \Box \)

4. Morphisms

A morphism of dependence alphabets \( h : (\Sigma, SD) \rightarrow (\Sigma', SD') \) is a mapping \( h : \Sigma \rightarrow \Sigma' \) such that \((a, b) \in SD\) implies \((h(a), h(b)) \in SD'\). We are going to define a homomorphism \( h^* : \mathcal{P}(\Sigma', SD') \rightarrow \mathcal{P}(\Sigma, SD) \) in the opposite direction. Let \([V', E', \lambda'] \in \mathcal{P}(\Sigma', SD')\) be a dependence graph. Then the image \([V, E, \lambda] = h^*([V', E', \lambda'])\) is constructed as follows. Each vertex of \( V' \) with label \( a' \) is replaced by the set \( h^{-1}(a') \subseteq \Sigma \). To be more formal, we take

\[
V = \bigcup_{v' \in V'} (h^{-1}(\lambda'(v'))) \times \{v'\}.
\]

The second component is used only to identify \( v' \) by \( h^{-1}(\lambda'(v')) \). To simplify the notation, we drop the second component and we think of a disjoint union

\[
V = \bigcup_{v' \in V'} h^{-1}(\lambda'(v')).
\]
Clearly, each vertex of \( V \) has a natural labelling in \( \Sigma \). Now, let \( a \in h^{-1}(\lambda'(v')) \) and \( b \in h^{-1}(\lambda'(w')) \). Then we draw an arc from \( a \) to \( b \) if and only if both \((a, b) \in SD\) and \((v', w') \in E'\). This completes the definition of \( h^* \).

We have to verify that \( h^*: \mathcal{P}(\Sigma', SD') \to \mathcal{P}(\Sigma, SD) \) is a homomorphism of monoids.

Since the neutral element is represented by the empty graph, the formula \( h^*(1) = 1 \) is immediate. Let us show \( h^*(s't') = h^*(s')h^*(t') \).

It is enough to see that if \( a \in h^*(s') \) and \( b \in h^*(t') \) such that \((a, b) \in SD\), then there must be an arc from \( a \) to \( b \) in the graph \( h^*(s't') \). However, this is clear since \((a, b) \in SD\) implies \((h(a), h(b)) \in SD'\). Hence, there is a corresponding arc from \( s' \) to \( t' \).

Consider, for example, the morphism of dependence alphabets as in Fig. 4. Then the image under \( h^* \) of the trace \( [a(1)(2)] \) is the partial trace

\[
\begin{pmatrix}
  a_1 \\
  a_2 \\
  c_1
\end{pmatrix}
\cdot
\begin{pmatrix}
  1 \\
  c_2 \\
  c_3
\end{pmatrix}
\cdot
\begin{pmatrix}
  a_1 \\
  a_2
\end{pmatrix}
\cdot
\begin{pmatrix}
  d_1
\end{pmatrix}
\]

This is also given in Fig. 5.

The following proposition characterizes the fact when the image of a trace is a trace.

**Proposition 4.1.** Let \( h: (\Sigma, SD) \to (\Sigma', SD') \) be a morphism of dependence alphabets and \( s' \in \mathcal{M}(\Sigma', SD') \subseteq \mathcal{P}(\Sigma', SD') \) be a trace. Then the image \( h^*(s') \subseteq \mathcal{P}(\Sigma, SD) \) is a trace in \( \mathcal{M}(\Sigma, SD) \) if and only if the induced directed subgraph \( h^{-1}(a') \subseteq (\Sigma, SD) \) is acyclic for all letters \( a' \) occurring in \( s' \).

![Fig. 4. A morphism \( h \) of dependence alphabets (indicated by the labelling).](image)

![Fig. 5. A partial trace, which is the image of a trace under \( h^* \).](image)
Proof. Let \((a_1, \ldots, a_n)\) be a directed cycle in \(h^{-1}(a')\), i.e., \((a_i, a_{i+1}) \in SD\) for all \(i \mod n, n \geq 2\). Then \(h^* (s')\) contains this cycle of soft arcs and it is therefore no trace.

For the other direction, assume that \(h^{-1}(a')\) is acyclic for all letters \(a'\) occurring in \(s'\). Then \(h^* (a')\) is a trace and since \(s'\) is a trace, \(s'\) can be written as a product of letters. The result follows since \(M(\Sigma, SD)\) is a submonoid. 

The main theorem related to this construction is the following generalization of the well-known embedding theorem or projection lemma.

Theorem 4.2. Let \(h : (\Sigma, SD) \rightarrow (\Sigma', SD')\) be a morphism of dependence alphabets. Then the homomorphism \(h^* : \mathcal{P}(\Sigma', SD') \rightarrow \mathcal{P}(\Sigma, SD)\) is injective if and only if \(h(SD) = SD'\), i.e., if \(h\) is surjective on vertices and edges.

Proof. One direction is easy. If there exists some \(a \in \Sigma' \setminus h(\Sigma)\), then \(h^* (a) = 1\) and \(h^*\) is not injective. If \((a', b') \in SD' \setminus h(SD), a' \neq b'\), then there exist partial traces \(s' = a' \rightarrow b'\) and \(t' = (b' a')\) which are different and which have the same image \(h^* (s') = h^* (t')\). (This is a graph without any arcs.)

For the other direction, we assume \(h(SD) = SD'\). Let \(s' \in \mathcal{P}(\Sigma', SD')\) be a partial trace and \(s = h^* (s')\) be the constructed dependence graph. The construction itself provides us with a natural mapping on the concrete dependence graphs \(h_\ast : s \rightarrow s'\). The definition of \(h_\ast\) is as follows. Let \(a \in \Sigma\) be any letter which occurs in \(s\). Then the \(i\)th \(a\) of \(s\) is mapped to the \(i\)th \(h(a)\) of \(s'\). Since \(h(SD) \subseteq SD'\), this induces a mapping from the arc set of \(s\) to the arc set of \(s'\). If \(h(SD) = SD'\), then \(h_\ast : s \rightarrow s'\) is surjective onto the vertices and arcs of \(s'\), too. Hence, we have \(h_\ast(h^* (s')) = s'\) and \(h^* (s') = h^* (t')\) implies \(h_\ast(h^* (s')) = s' = t' = h_\ast(h^* (t'))\). 

Corollary 4.3. Let \((\Sigma, SD)\) be a dependence alphabet and \(\Gamma \subseteq \Sigma\) be the subset of isolated vertices. Then the monoid of partial traces \(\mathcal{P}(\Sigma, SD)\) admits a canonical embedding

\[
\mathcal{P}(\Sigma, SD) \rightarrow \prod_{(a, b) \in SD \setminus \id_\Sigma} \{a, b\}^* \times \prod_{c \in \Gamma} c^*.
\]

Proof. A usual we may view \((\Sigma, SD)\) as an undirected graph (without drawing the self-loops). Define a new graph by the disjoint union of all arcs \((a, b) \in SD \setminus \id_\Sigma\) and isolated vertices \(c \in \Gamma\). Since disjoint union corresponds to direct product, this yields by Proposition 3.6 a dependence alphabet \((\Sigma', SD')\) such that

\[
\mathcal{P}(\Sigma', SD') = \prod_{(a, b) \in SD \setminus \id_\Sigma} \{a, b\}^* \times \prod_{c \in \Gamma} c^*.
\]

The morphism \(h : (\Sigma', SD') \rightarrow (\Sigma, SD)\) is induced by the embedding of the set of arcs and vertices from each part of the disjoint union to the graph \((\Sigma, SD)\). It is clearly surjective, so \(h^*\) is injective by the above theorem. 

The explicit description of the embedding given in Corollary 4.3 is given by the projections \(\pi_{(a, b)} : \mathcal{P}(\Sigma, SD) \rightarrow \{a, b\}^*\) for \((a, b) \in SD \setminus \id_\Sigma\) and \(\pi_c : \mathcal{P}(\Sigma, SD) \rightarrow c^*\) for \(c \in \Gamma\).
For a partial trace \( p = [V, E, \lambda] \) and \( c \in I \), the image \( \pi_c(p) \) is simply \( c^{|e \in \lambda^{-1}(V(t)) \cap c|} \), i.e., \( \pi_c(p) = |p|_c \) if we identify \( c^* \) with \( \mathbb{N} \). For \( p = [V, E, \lambda] \) and \( (a, b) \in SD - \text{id}_x \), let \( p_{(a,b)} \in \mathbb{P}(a \rightarrow b) \) be the restriction of \( p \) to the vertices in \( \lambda^{-1}([a, b]) \). The image \( \pi_{(a,b)}(p) \in [a, b]^* \) is obtained by the canonical isomorphism between \( \mathbb{P}(a \rightarrow b) \) and \( [a, b]^* \) due to Proposition 3.6. Clearly, these projections are computable in linear time.

**Corollary 4.4.** For a fixed alphabet there is an algorithm to decide equality of partial traces, which is linear time in the number of vertices.

**Example.** Let \( (\Sigma, SD) = b \rightarrow a \rightarrow c \rightarrow d \) as in Fig. 4. Consider the following two partial traces:

\[
\begin{array}{ccc}
 & b & \\
 a & \rightarrow & a \\
 c & \rightarrow & d \\
\end{array}
\quad,

\begin{array}{ccc}
 & b & \\
 a & \rightarrow & a \\
 c & \rightarrow & d \\
\end{array}
\]

The projections are given by the following equations:

\[
\begin{align*}
\pi_{(a,b)}(p) &= \pi_{(b,a)}(p) = aba \in \{a, b\}^*, \\
\pi_{(a,c)}(p) &= \pi_{(c,a)}(p) = ac \in \{a, c\}^*, \\
\pi_{(c,d)}(p) &= \pi_{(d,c)}(p) = cd \in \{c, d\}^*, \\
\pi_{(a,b)}(q) &= \pi_{(b,a)}(q) = ab \in \{a, b\}^*, \\
\pi_{(a,c)}(q) &= \pi_{(c,a)}(q) = ac \in \{a, c\}^*, \\
\pi_{(c,d)}(q) &= \pi_{(d,c)}(q) = dc \in \{c, d\}^*. \\
\end{align*}
\]

Since \( p \) is a trace, i.e., \( p \in \mathbb{M}(\Sigma, SD) \), it is clear that \( \pi_{(x,y)}(p) = \pi_{(y,x)}(p) \) for all \( (x, y) \in SD \cap SD^{-1} \), but this is no characterization of traces.

### 4.1. The synchronization operator

Let \( (\Sigma_1, SD_1), (\Sigma_2, SD_2) \) be two dependence alphabets, \( \Sigma' = \Sigma_1 \cap \Sigma_2 \) and \( (\Sigma, SD) = (\Sigma_1 \cup \Sigma_2, SD_1 \cup SD_2) \). We define the synchronization of \( s_1 \in \mathbb{P}(\Sigma_1, SD_1) \) and \( s_2 \in \mathbb{P}(\Sigma_2, SD_2) \), only if \( |s_1|_a = |s_2|_a \) for all \( a \in \Sigma' \). In this case, we may take the union of \( s_1 \)
and $s_2$ by identifying the corresponding vertices with label in $\Sigma'$. This yields some labelled graph over $(\Sigma, SD) = (\Sigma_1 \cup \Sigma_2, SD_1 \cup SD_2)$. If this graph is acyclic then there is a unique way to complete it such that it becomes a dependence graph. This graph is denoted by $s_1 \parallel s_2$; it is called the synchronisation of $s_1$ and $s_2$. The partial traces $s_1, s_2$ are called synchronizable. Of course, if $s_1, s_2$ are synchronizable traces then $s_1 \parallel s_2$ coincides with the definition given in Section 2.

Note that the inclusion $(\Sigma_i, SD_i) \rightarrow (\Sigma, SD)$ yields a projection

$$p_i : \mathcal{P}(\Sigma, SD) \rightarrow \mathcal{P}(\Sigma_i, SD_i), \quad i = 1, 2.$$  

Since for all $p \in \mathcal{P}(\Sigma, SD)$ we have $p = p_1(p) \parallel p_2(p)$, it is convenient to define

$$\mathcal{P}(\Sigma, SD) = \mathcal{P}(\Sigma_1, SD_1) \parallel \mathcal{P}(\Sigma_2, SD_2).$$

On the other hand, for $s_i \in \mathcal{P}(\Sigma_i, SD_i), i = 1, 2$, we will have neither $p_1(s_1 \parallel s_2) = s_1$ nor $p_2(s_1 \parallel s_2) = s_2$, in general. It is possible that the synchronization forces additional arcs. (In fact, this is the reason why the synchronization of semi-traces leads naturally to the notion of partial trace.) Recall that similar to traces there is a natural ordering for partial traces. We write $s \leq t$ if we obtain $s$ from $t$ by adding new arcs between existing vertices. Using this ordering we always have $p_i(s_1 \parallel s_2) \leq s_i$ for $i = 1, 2$ and $s_1 \parallel s_2 = p_1(s_1 \parallel s_2) \parallel p_2(s_1 \parallel s_2)$. Let us call $s_1, s_2$ directly synchronizable if both $s_1 \parallel s_2$ exists and $p_i(s_1 \parallel s_2) = s_i$ for $i = 1, 2$. It is straightforward to extend both notations to languages: $L_1 \parallel L_2 = \{s_1 \parallel s_2 | s_i \in L_i \text{ for } i = 1, 2\}$, $L_1 \parallel_{\text{direct}} L_2 = \{s_1 \parallel s_2 | s_1 \in L_1, s_2 \in L_2 \text{ are directly synchronizable}\}$.

We are mostly interested in closed languages, i.e., in languages $L$ where $s \leq t \in L$ implies $s \in L$. For such languages both notions coincide and direct synchronization is sufficient. This has an important consequence.

**Proposition 4.5.** Let $L_i \subseteq \mathcal{P}(\Sigma_i, SD_i)$ be closed languages, $i = 1, 2$. Consider the canonical inclusion given by Theorem 4.2:

$$\iota : \mathcal{P}(\Sigma, SD) \rightarrow \mathcal{P}(\Sigma_1, SD_1) \times \mathcal{P}(\Sigma_2, SD_2).$$

where

$$\mathcal{P}(\Sigma, SD) = \mathcal{P}(\Sigma_1, SD_1) \parallel \mathcal{P}(\Sigma_2, SD_2).$$

Then we have

$$L_1 \parallel L_2 = (L_1 \times L_2 \cap \iota(\mathcal{P}(\Sigma, SD))).$$

5. Applications to Petri nets

A Petri net is a tuple $N = (P, T, F, B)$ where $P$ is a finite set of places, $T$ is a finite set of transitions, $F$ and $B$ are $P \times T$-matrices over $\mathbb{N}$, the forward and backward incidence matrices. We also view $F$ and $B$ as mappings $F, B : T \rightarrow \mathbb{N}^P$. By $\mathbb{N}^P$ we
denote as usual the set of mappings from $P$ to the nonnegative integers. It is the free commutative monoid over $P$. By linear extension, we also view $F, B$ as homomorphisms $F, B : \mathbb{N}^T \to \mathbb{N}^P$. If a Petri net is viewed as a graph then $F(p, a)$, if it is nonzero, denotes the weight of an arc from $p$ to $a$, if $F(p, a)$ is zero then there is no such arc, and analogously a nonzero value of $B(p, a)$ is the weight of an arc from $a$ to $p$.

A marking of $N$ is an element of $\mathbb{N}^P$. A transition $a \in T$ is enabled under a marking $m \in \mathbb{N}^P$ if $m \geq F(a)$. If $a \in T$ is enabled under $m \in \mathbb{N}^P$ then the follower marking $m' = m - F(a) + B(a)$ may occur. We denote this by $m[a]m'$. For a sequence $w = a_1a_2 \ldots a_n \in T^*$, we write $m[w]m'$ if $m[a_1]m_1[a_2]m_2 \ldots m_{n-1}[a_n]m'$ for some $m_1, m_2, \ldots, m_{n-1} \in \mathbb{N}^P$. We also write $m[w]$ to denote that $m[w]m'$ holds for some $m' \in \mathbb{N}^P$.

Often, a net is given with an initial marking $m_0 \in \mathbb{N}^P$. Then we shall speak of a **system**. If $N$ is a system, then $L(N) = \{w \in T^* | m_0[w] \}$ denotes the string-language of $N$. We say that a set of transitions $s \subseteq T$ is **concurrently enabled** under a marking $m \in \mathbb{N}^P$ if $m \geq \sum_{a \in s} F(a)$.

This leads to the definition when a semi-word $s \in SW(T)$ is enabled. For this recall that we have an ordering on semi-words where $s \leq t$ means that $s$ allows less concurrency. Thus, $p \leq q$ if and only if $q$ can be transformed into $p$ by introducing new arcs. There is also a natural notion of prefix. A prefix $s$ of a semi-word $p$ is a downward closed subset with respect to the labelled partial order $p$. If $s$ is a prefix of $p$, then we have $st \leq p$ for some $t$, where the concatenation of semi-words is the same as for a full dependence relation. The idea is now that after the execution of the prefix $s$ the set of minimal letters of $t$, denoted by $\text{min}(t)$, must be concurrently enabled. The following coincides with the definition given in $[9]$.

**Definition 5.1.** Let $N = (P, T, F, B)$ be a Petri net and $m \in \mathbb{N}^P$ some marking. Extending $F$ and $B$ to mappings $F, B : SW(T) \to \mathbb{N}^P$ by using the Parikh images of semi-words, we say that a semi-word $p \in SW(T)$ is enabled under $m$, if we have for all $p \geq st$ both $m - F(s) + B(s) \geq 0$ and $\text{min}(t)$ is concurrently enabled for the marking $m - F(s) + B(s)$. If $p$ is enabled under $m$ then it may yield the follower marking $m' = m - F(p) + B(p)$. We also use the notation $m[p]m'$ to express this fact.

With the above definition we can define a suitable behavior of a system as the set of semi-words enabled under the initial marking. However, this does not use the information about the concurrency relation given by the static net topology. This information can be used and leads naturally to the concept of partial traces. Thus, we simply base our semantics on the dependence relation given by the net topology.

We say that transition $a, b \in T$ are semi-dependent if $a = b$ or $a$ is in the preset and $b$ is in the postset of some common place $p$, i.e., we have $a = b$ or $B(p, a) \cdot F(p, b) \neq 0$ for some $p$.

The semi-dependence relation of $N$ is denoted by $SD(N)$. With a Petri net $N$ we henceforth associate the dependence alphabet $(T, SD(N))$ and the monoid of partial traces $\mathbb{P}(N) = \mathbb{P}(T, SD(N))$. 

---

*A partial trace semantics for Petri nets*
Lemma 5.2. Let $N=(P,T,F,B)$ be a Petri net and $m\in\mathbb{N}^P$ some marking. Let $s,s'\in SW(T)$ be semi-words such that $s$ and $s'$ become equal as partial traces, i.e., $[s]=[s']\in\mathbb{P}(N)$. Then either both semi-words are enabled under $m$ or none of $s, s'$.

Proof. This follows from a more general assertion, see [9, Proposition 5.3.3] or [16, Theorem 1.11]. A direct verification uses best the observation that enabling is defined in such a way that it can be verified locally at places. □

By the above lemma we can define the partial trace behavior of a system $N=(P,T,F,B,m_0)$ by

$$L_\rho(N) = \{s\in\mathbb{P}(N) | m_0[s]m' \text{ for some } m'\}.$$  

The rest of the paper is devoted to study $L_\rho(N)$. We are going to generalize the results of [6, Ch. 3]. These results were based on a certain type of morphisms between nets. Since we are working with partial traces instead of (Mazurkiewicz) traces, we can in fact even consider a larger class of local morphisms.

Definition 5.3. A local morphism $h: N'\to N$ from a Petri net $N'=(P',T',F',B')$ to a Petri net $N=(P,T,F,B)$ is a pair of mappings $h=(h_P,h_T)$ such that $h_P: P'\to P$ and $h_T: T'\to T$ satisfy for all $a\in T$ and for all $p'\in P'$ the following two equations:

$$\sum_{a'\in h_T^{-1}(a)} F'(p',a') = F(h_P(p'),a),$$
$$\sum_{a'\in h_T^{-1}(a)} B'(p',a') = B(h_P(p'),a).$$

The condition expresses the compatibility of the mappings $h_P$ and $h_T$ with the incidence matrices $F$ and $B$. In particular, forward arcs are mapped to forward arcs and backward arcs are mapped to backward arcs.

Restricted to the neighborhood of a place, a local morphism may look like in Fig. 6. In particular, a local morphism may create cycles. This would have been forbidden in the trace case. Note that any mapping $f: A\to B$ of finite sets induces a homomorphism $f^*: \mathbb{N}^B\to \mathbb{N}^A$ defined by $f^*(m)=mf: A\overset{\rightarrow}{\to} B^m \to \mathbb{N}$. Applied to a local morphism $(h_P,h_T): N'\to N$, we obtain homomorphisms $h_P^*: \mathbb{N}^T\to \mathbb{N}^{T'}$ and $h_T^*: \mathbb{N}^P\to \mathbb{N}^{P'}$. Writing the elements of $\mathbb{N}^T$ and $\mathbb{N}^P$ as formal sums $\sum_{a\in T} n_a a$ and $\sum_{p\in P} n_p p$, respectively, where $n_a, n_p\in\mathbb{N}$ we then have

$$F\left(\sum_{a\in T} n_a a\right) = \sum_{p\in P} \left(\sum_{a\in T} n_a F(p,a)\right)p$$

and

$$B\left(\sum_{a\in T} n_a a\right) = \sum_{p\in P} \left(\sum_{a\in T} n_a B(p,a)\right)p.$$
A partial trace semantics for Petri nets

Fig. 6. A local morphism at a place.

Fig. 7. The commuting diagram of a local morphism.

An easy calculation shows that the condition of a local morphism may be rephrased by saying that the diagram of homomorphisms in Fig. 7 commutes.

Of course, a local morphism relates the behavior of the two respective nets to each other. This works as follows: a local morphism $h = (h_p, h_T): N' \to N$ induces a morphism of dependence alphabets $(T', SD(N')) \to (T, SD(N))$. Therefore, we may define a homomorphism in the opposite direction $h^*: \mathcal{P}(N) \to \mathcal{P}(N')$. The definition of a local morphism asserts that $h^*_p$ maps the partial trace language of $N$ to that of $N'$.

**Lemma 5.4.** Let $(h_p, h_T): N' \to N$ be a local morphism of Petri nets, $m_1, m_2 \in \mathcal{N}_p$ be markings, and $s \subseteq T$ be a set of transitions of $N$. Assume that $s$ is concurrently enabled under $m_1$. Then $m_1 \mathcal{L} s \mathcal{L} m_2$ implies $h^*_p(m_1) \geq \sum_{a \in h^{-1}_T(s)} F'(a)$ and hence $h^*_p(m_1)[h^{-1}_T(s)] h^*_p(m_2)$.

**Proposition 5.5.** Let $(h_p, h_T): N' \to N$ be a local morphism of Petri nets. Let $m_0 \in \mathcal{N}_p$ be the initial marking of $N$ and $h^*(m_0)$ be the initial marking of $N'$. Then the restriction of the homomorphism $h^*_T: \mathcal{P}(N) \to \mathcal{P}(N')$ induces a mapping

$$h^*: L_p(N) \to L_p(N') \cap h^*_T(\mathcal{P}(N)).$$

We are mainly interested in the case where $h^*$ above is bijective, which is true for the following type of local morphisms:

**Definition 5.6.** A local morphism $(h_p, h_T): N' \to N$ of Petri nets is called a covering if $h_p$ is surjective on places and $h_T$ is surjective on transitions.
The main theorem of this section is a generalization of [6, Theorem 3.1.8].

**Theorem 5.7.** Let \( h = (h_p, h_T): N' \to N \) be a covering of Petri nets. Let \( m_0 \) be the initial marking of \( N \) and \( h_p^*(m_0) \) be the initial marking of \( N' \). Then the mapping

\[
h_p^*: I_p(N) \to I_p(N') \cap h_T^*(I_p(N))
\]

is bijective.

**Proof.** Since \((h_p, h_T)\) is a covering, it can be easily seen that \( h_T^*: (T', SD(N')) \to (T, SD(N)) \) is surjective on vertices and edges. Hence, we may apply the embedding theorem on partial traces to conclude that \( h_T^* \) is injective. We show \( h_T^*(L_p(N)) = L_p(N') \cap h_T^*(P(N)) \).

Let \( p' = h_T^*(p) \in L_p(N') \) for some partial trace \( p \in L_p(N) \). We show that \( p \) is enabled under \( m_0 \). Choose any \( s, t \in P(N) \) such that \( p \models st \).

Then \( p' = h_T^*(p) \models h_T^*(s)h_T^*(t) \). Hence, we have \( h_T^*(m_0) = F'(h_T^*(s)) + B'(h_T^*(s)) \geq 0 \). Since \( F' h_T^* = h_T^* F, B' h_T^* = h_T^* B \), and \( h_T^* \) is surjective on places, \( h_T^* \) is injective, and we have \( m_0 - F(s) + B(s) \geq 0 \). Now, let \( r \leq \min(t) \). Then we have \( h_T^*(r) \leq \min(h_T^*(t)) \). Since \( \min(h_T^*(t)) \) is concurrently enabled after the execution of \( h_T^*(s) \), the same is true for \( h_T^*(r) \) and, hence, \( r \) is concurrently enabled after the execution of \( s \). Therefore, \( p \in L_p(N) \) and the result follows.

The above theorem tells us that if \( h: N' \to N \) is a covering then we may split the computation of \( L_p(N) \) into two parts: the (local) computation of \( L_p(N') \) which might be simpler, and the (global) computation of \( h^*(P(N)) \). Another easy application of the above theorem is the case of a local morphism \((h_p, h_T)\), where \( h_p \) is surjective and \( h_T \) is bijective.

**Corollary 5.8.** Let \((h_p, h_T): N' \to N \) be a covering of Petri nets such that \( h_T \) is bijective. Let \( m_0 \) be the initial marking of \( N \) and \( h_p^*(m_0) \) be the initial marking of \( N' \) and use \( h_T \) as an identification of \( T' \) and \( T \). Then we have \( P(N) = P(N') \) and \( L_p(N) = L_p(N') \).

**Proof.** Since \((h_p, h_T)\) is covering and \( h_T \) is bijective, we may identify the dependence alphabets \((T, SD(N))\) and \((T', SD(N'))\). The result follows directly from the above theorem.

6. Synchronization of Petri nets

A Petri net \( N' = (P', T', F', B') \) is called a subnet of \( N = (P, T, F, B) \) if \( S' \subseteq S, T' \subseteq T \) and \( F', B' \) are the restrictions of \( F, B \). A subnet is called transition-bounded if it contains with every place all transitions adjacent to that place. It is easy to see that a subnet is transition bounded if and only if the inclusion is a local morphism [6, Lemma 3.2.1].
Note that if \((i_p, i_f) : N' \to N\) is a local morphism where the mapping \(i_p\) is injective then \(i_p^* : N^p \to N'^p\) is just the restriction of a marking of \(N\) to a marking of \(N'\).

The synchronization of two nets is defined over a common transition-bounded subnet. We restrict ourselves to this case in order to have Theorem 6.2.

**Definition 6.1.** Let \(N_i = (P_i, T_i, F_i, B_i)\) \(i = 1, 2\), be Petri nets which contain a common transition-bounded subnet \(N' = (P', T', F', B')\). Then the synchronization of \(N_1\) and \(N_2\) over \(N'\) is defined by \(N_1 11 N_2 = (P, T, F, B)\) where \(P\) and \(T\) are the disjoint unions \(P = (P_1 \setminus P') \cup (P_2 \setminus P') \cup P', T = (T_1 \setminus T') \cup (T_2 \setminus T') \cup T'\), and \(F, B\) are the obvious extensions to \(P\) and \(T\). If \(N_1, N_2\) have initial markings \(m_1, m_2\) which coincide on \(N'\), then we give \(N_1 \parallel N_2\) as initial marking the natural extension of \(m_1\) and \(m_2\) to \(P\).

The synchronization over the empty subnet, which is transition-bounded, yields the disjoint union \(N_1 \cup N_2\) of nets. This is the direct sum in our category. If we have two nets \(N_1, N_2\) with a common transition bounded subnet \(N'\) then we obtain a natural covering

\[
p : N_1 \cup N_2 \to N_1 \parallel N' \parallel N_2.
\]

This covering will be used to compute the trace language of a synchronized net. Let us now assume that the net \(N'\) is in fact the intersection of nets \(N_1\) and \(N_2\) (we may do this in any case by taking suitable isomorphic copies of \(N_1\) and \(N_2\)), and let us simply write \(N_1 \parallel N_2\) in this case. Since \(N'\) is transition-bounded, two transitions are semi-dependent in \(N_1 \parallel N_2\) if and only if they are semi-dependent in \(N_1\) or in \(N_2\). Thus \(P(N_1 \parallel N_2) = P(N_1) \parallel P(N_2)\).

Recall that the synchronization of partial traces \(s_i \in P(N_i), i = 1, 2\), was defined as a graph theoretical union which identifies corresponding transitions in \(T_1 \cap T_2\). Let \(p_i : P(N_1 \parallel N_2) \to P(N_i)\) the natural projection. If \(s = s_1 \parallel s_2\) is the synchronization then we have \(s = p_1(s) \parallel p_2(s)\) and \(s_i \geq p_i(s)\) for \(i = 1, 2\).

**Lemma 6.2.** Let \(N_1 \parallel N_2\) be the synchronization of two systems and \(L_i \subseteq P(N_i)\) for \(i = 1, 2\) the partial trace behavior of the components. Then we have

\[
L_1 \parallel L_2 = \{ s \in P(N_1 \parallel N_2) \mid p_i(s) \in L_i \text{ for } i = 1, 2\}.
\]

**Proof.** This follows from the fact that \(L_i \subseteq P(N_i)\) are closed languages. □

The lemma is crucial for the following theorem, which turns out to be a special case of Theorem 5.7.

**Theorem 6.3.** Let \(N_1, \ldots, N_k\) be Petri nets with initial markings, \(k \geq 2\), such that for all \(1 \leq i, j \leq k, i \neq j\) the intersections \(P_i \cap P_j, T_i \cap T_j\) induce a common transition bounded subnet of \(N_i\) and \(N_j\) where the initial marking coincides. Then we have

\[
L_\emptyset(N_1 \parallel \ldots \parallel N_k) = L_\emptyset(N_1) \parallel \ldots \parallel L_\emptyset(N_k).
\]
Proof. By induction, we have to consider the case $k = 2$, only. The natural mapping of the disjoint union to the synchronization, $p : N_1 \cup N_2 \rightarrow N_1 \parallel N_2$, is a covering. Thus, by Theorem 5.7, we have $p^*(L_p(N_1 \parallel N_2)) = L_p(N_1 \cup N_2) \cap p^*(\mathcal{P}(N_1 \parallel N_2))$. Up to a natural identification, we have $\mathcal{P}(N_1 \cup N_2) = \mathcal{P}(N_1) \times \mathcal{P}(N_2)$ and $L_p(N_1 \cup N_2) = L_p(N_1) \times L_p(N_2)$. Therefore, we can write

$$L_p(N_1) \times L_p(N_2) \cap p^*(\mathcal{P}(N_1 \parallel N_2))$$

$$= p^*(\{ t \in \mathcal{P}(N_1 \parallel N_2) \mid p_i(t) \in L_p(N_i) \text{ for } i = 1, 2 \}).$$

By Lemma 6.2 this set is equal to $p^*(L_p(N_1) \parallel L_p(N_2))$. Therefore, we obtain

$$p^*(L_p(N_1) \parallel L_p(N_2)) = p^*(L_p(N_1) \parallel L_p(N_2)).$$

Since $p$ is a covering, the mapping $p^*$ is injective. Hence, the result. \qed

As a special case we obtain our final result.

**Corollary 6.3.** Let $N = (P, T, F, B)$ be a Petri net with initial marking. For $p \in P$, let $\text{Atom}(p)$ the subnet with place $p$ and adjacent transitions. Then we have

$$L_P(N) = \|_{p \in P} L_p(\text{Atom}(p)).$$

**Acknowledgment**

I would like to thank Walter Vogler and the anonymous referee for his detailed comments, which led in particular to Theorem 3.7. A first part of this paper was written during my stay in Milan in June 92. I am grateful to Giancarlo Mauri and Nicoletta Sabadini for valuable discussions.

**References**


