# Articulation of Transition Systems and Its Application to Petri Net Synthesis

Raymond Devillers(✉)

Université Libre de Bruxelles, Boulevard du Triomphe C.P. 212,
1050 Bruxelles, Belgium
rdevil@ulb.ac.be

**Abstract.** In order to speed up the synthesis of Petri nets from labelled transition systems, a divide and conquer strategy consists in defining LTS decomposition techniques and corresponding PN composition operators to recombine the solutions of the various components. The paper explores how an articulation decomposition, possibly combined with a product and addition technique developed in previous papers, may be used in this respect and generalises sequence operators, as well as looping ones.

**Keywords:** Labelled transition systems · Composition · Decomposition · Petri net synthesis

## 1 Introduction

Instead of analysing a given system to check if it satisfies a set of desired properties, the synthesis approach tries to build a system "correct by construction" directly from those properties. In particular, more or less efficient algorithms have been developed to build a bounded Petri net (possibly of some subclass) the reachability graph of which is isomorphic to (or close to) a given finite labelled transition system [2,7,10,11,15].

The synthesis problem is usually polynomial in terms of the size of the LTS, with a degree between 2 and 5 depending on the subclass of Petri nets one searches for [2,3,7,10], but can also be NP-complete [4]. Hence the interest to apply a "divide and conquer" synthesis strategy when possible. The general idea is to decompose the given LTS into components, to synthesise each component separately and then to recombine the results in such a way to obtain a solution to the global problem. This has been applied successfully to disjoint products of LTS, which correspond to disjoint sums of Petri nets [12,13]. But it has also been observed that such products may be hidden inside other kinds of components, for instance in sequences of LTS, as illustrated in Fig. 1 (borrowed from [12]; the initial states are slightly fatter than the other ones), and developed in the algebra of Petri nets[1] [8,9].

[1] Note that this theory uses labelled Petri nets, where several transitions may have the same label, or multiset of labels, while here we shall only consider unlabelled Petri nets.

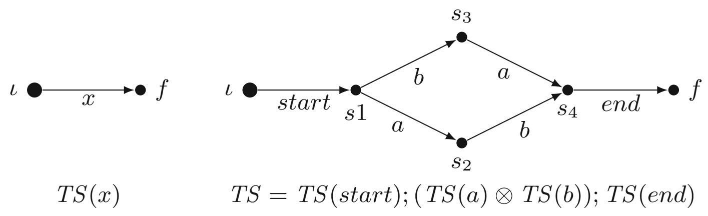


**Fig. 1.** Combination of sequence operators with a product.

We shall here develop this last point, but we shall also generalise the idea of sequences into an *articulation* operator. The difference is that it will sometimes be possible to come back to the first component after having executed the second one, and repeat the alternation between these two components. A side consequence will be that the articulation operator will not always be anti-commutative, like the sequence is, and that it includes a choice as well as a looping feature.

The structure of the paper is as follows. After recalling the bases of labelled transition systems, a new articulation operator is introduced, and its basic properties are analysed. Then, the bases of the Petri net synthesis problem are recalled, and Sect. 4 shows how synthesis applies to the components of an articulation. Next, it is shown how articulations may be used to simplify a synthesis problem, by composing a solution of the given system from the solutions of the articulated components. A procedure is then detailed to show how to decompose a given LTS into articulated components, when possible. As usual, the last section concludes.

## 2 Labelled Transition Systems and Articulations

A classic way for representing the possible (sequential) evolutions of a dynamic system is through its labelled transition system [1].

**Definition 1.** LABELLED TRANSITION SYSTEMS
A *labelled transition system* (LTS for short) with initial state is a tuple $TS = (S, \to, T, \iota)$ with node (or state) set $S$, edge label set $T$, edges $\to \subseteq (S \times T \times S)$, and an initial state $\iota \in S$. We shall denote $s[t\rangle$ for $t \in T$ if there is an arc labelled $t$ from $s$, $[t\rangle s$ if there is an arc lalbelled $t$ going into $s$, and $s[\alpha\rangle s'$ if there is a path labelled $\alpha \in T^*$ from $s$ to $s'$. Such a path will also be called an evolution of the LTS (from $s$ to $s'$).

Two LTSs $TS_1 = (S_1, \to_1, T, \iota_1)$ and $TS_2 = (S_2, \to_2, T, \iota_2)$ with the same label set $T$ are (state-)isomorphic, denoted $TS_1 \equiv_T TS_2$ (or simply $TS_1 \equiv TS_2$ if $T$ is clear from the context), if there is a bijection $\zeta\colon S_1 \to S_2$ with $\zeta(\iota_1) = \iota_2$ and $(s, t, s') \in \to_1 \Leftrightarrow (\zeta(s), t, \zeta(s')) \in \to_2$, for all $s, s' \in S_1$ and $t \in T$. We shall usually consider LTSs up to isomorphism.

We shall also assume each LTS is totally reachable (i.e., $\forall s \in S \exists \alpha \in T^* : \iota[\alpha\rangle s$), that it is weakly live (i.e., each label $t \in T$ occurs at least once in $\rightarrow$), and $T \neq \emptyset$.

Let $T_1 \subseteq T$. We shall denote by $adj(T_1) = \{s \in S | \exists t \in T_1 : s[t\rangle \text{ or } [t\rangle s\}$ the *adjacency set* of $T_1$, i.e., the set of states connected to $T_1$. Let $\emptyset \subset T_1 \subset T$, $T_2 = T \setminus T_1$ and $s \in S$. We shall say that $TS$ is *articulated*[2] by $T_1$ and $T_2$ around $s$ if $adj(T_1) \cap adj(T_2) = \{s\}$, $\forall s_1 \in adj(T_1) \exists \alpha_1 \in T_1^* : \iota[\alpha_1\rangle s_1$ and $\forall s_2 \in adj(T_2) \exists \alpha_2 \in T_2^* : s[\alpha_2\rangle s_2$.

Let $TS_1 = (S_1, \rightarrow_1, T_1, \iota_1)$ and $TS_2 = (S_2, \rightarrow_2, T_2, \iota_2)$ two (totally reachable) LTSs with $T_1 \cap T_2 = \emptyset$ and $s \in S_1$. Thanks to isomorphisms we may assume that $S_1 \cap S_2 = \{s\}$ and $\iota_2 = s$. We shall then denote by $TS_1 \triangleleft s \triangleright TS_2 = (S_1 \cup S_2, T_1 \cup T_2, \rightarrow_1 \cup \rightarrow_2, \iota_1)$ the *articulation* of $TS_1$ and $TS_2$ around $s$. □

Several easy but interesting properties may be derived for this articulation operator.

Note first that this operator is only defined up to isomorphism since we may need to rename the state sets (usually the right one, but we may also rename the left one, or both). The only constraint is that, after the relabellings, $s$ is the unique common state of $TS_1$ and $TS_2$, and is the state where the two systems are to be articulated. Figure 2 illustrates this operator. It also shows that the articulation highly relates on the state around which the articulation takes part. It may also be observed that, if $TS_0 = (\{\iota\}, \emptyset, \emptyset, \iota)$ is the trivial empty LTS, we have that, for any state $s$ of $TS$, $TS \triangleleft s \triangleright TS_0 \equiv TS$, i.e., we have a kind of right neutral trivial articulation. Similarly, $TS_0 \triangleleft \iota \triangleright TS \equiv TS$, i.e., we have a kind of left neutral trivial articulation. However, these neutrals will play no role in the following of this paper, so that we shall exclude them from our considerations (that is why we assumed the edge label sets to be non-empty).

**Corollary 1.** Both forms of articulation are equivalent
*If $TS = (S, \rightarrow, T, \iota)$ is articulated by $T_1$ and $T_2$ around $s$, then the structures $TS_1 = (adj(T_1), \rightarrow_1, T_1, \iota)$ and $TS_2 = (adj(T_2), \rightarrow_2, T_2, s)$, where $\rightarrow_1$ is the restriction of $\rightarrow$ to $T_1$ (i.e., $\rightarrow_1 = \rightarrow \cap adj(T_1) \times T_1 \times adj(T_1)$), and similarly for $\rightarrow_2$, are (totally reachable) LTSs, $TS \equiv_{T_1 \uplus T_2} TS_1 \triangleleft s \triangleright TS_2$ (in that case we do not need to apply a relabelling to $TS_1$ and $TS_2$).*

*Conversely, $TS_1 \triangleleft s \triangleright TS_2$ is articulated by the label sets of $TS_1$ and $TS_2$ around $s$, if these LTSs are totally reachable.* □

**Corollary 2.** Evolutions of an articulation
*If $TS \equiv TS_1 \triangleleft s \triangleright TS_2$, $\iota[\alpha\rangle s'$ is an evolution of $TS$ iff it is an alternation of evolutions of $TS_1$ and $TS_2$ separated by occurrences of $s$, i.e., either $\alpha \in T_1^*$ or $\alpha = \alpha_1 \alpha_2 \ldots \alpha_n$ such that $\alpha_i \in T_1^*$ if $i$ is odd, $\alpha_i \in T_2^*$ if $i$ is even, $\iota[\alpha_1\rangle s$ and $\forall i \in \{1, 2, \ldots, n-1\} : [\alpha_i\rangle s[\alpha_{i+1}\rangle$.* □

[2] This notion has some similarity with the cut vertices (or articulation points) introduced for connected unlabelled undirected graphs, whose removal disconnects the graph. They have been used for instance to decompose such graphs into biconnected components [14,16].

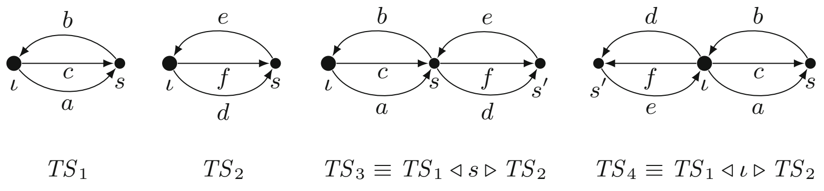

b
c
a
ι
s
e
f
d
ι
s
b
e
c
f
a
d
ι
s
s′
d
b
f
c
e
a
s′
ι
s
$TS_1$
$TS_2$
$TS_3 \equiv TS_1 \lhd s \rhd TS_2$
$TS_4 \equiv TS_1 \lhd \iota \rhd TS_2$

**Fig. 2.** Some articulations.

For instance, in $TS_3$ from Fig. 2, a possible evolution is $\iota[abc\rangle s[fede\rangle s[b\rangle\iota$, but also equivalently $\iota[a\rangle s[\varepsilon\rangle s[bc\rangle s[fe\rangle s[\varepsilon\rangle[de\rangle s[b\rangle\iota$ (where $\varepsilon$ is the empty sequence).

**Corollary 3.** Associativity of articulations
*Let us assume that $TS_1$, $TS_2$ and $TS_3$ are three LTSs with label sets $T_1$, $T_2$ and $T_3$ respectively, pairwise disjoint. Let $s_1$ be a state of $TS_1$ and $s_2$ be a state of $TS_2$. Then, $TS_1 \lhd s_1 \rhd (TS_2 \lhd s_2 \rhd TS_3) \equiv_{T_1 \cup T_2 \cup T_3} (TS_1 \lhd s_1 \rhd TS_2) \lhd s_2' \rhd TS_3$, where $s_2'$ corresponds in $TS_1 \lhd s_1 \rhd TS_2$ to $s_2$ in $TS_2$ (let us recall that the articulation operator may rename the states of the second operand).* □

This is illustrated by Fig. 3.

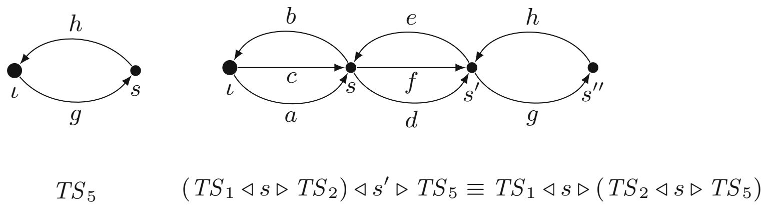

h
g
ι
s
b
e
h
c
f
a
d
g
ι
s
s′
s″
$TS_5$
$(TS_1 \lhd s \rhd TS_2) \lhd s' \rhd TS_5 \equiv TS_1 \lhd s \rhd (TS_2 \lhd s \rhd TS_5)$

**Fig. 3.** Associativity of articulations.

**Corollary 4.** Commutative articulations
*If $TS_1 = (S_1, \rightarrow_1, T, \iota_1)$ and $TS_2 = (S_2, \rightarrow_2, T, \iota_2)$ with disjoint label sets (i.e., $T_1 \cap T_2 = \emptyset$), then $TS_1 \lhd \iota_1 \rhd TS_2 \equiv_{T_1 \cup T_2} TS_2 \lhd \iota_2 \rhd TS_1$.* □

For instance, in Fig. 2, $TS_4 \equiv TS_1 \lhd \iota \rhd TS_2 \equiv TS_2 \lhd \iota \rhd TS_1$.

**Corollary 5.** Commutative associativity of articulations
*Let us assume that $TS_1$, $TS_2$ and $TS_3$ are three LTSs with label sets $T_1$, $T_2$ and $T_3$ respectively, pairwise disjoint. Let $s_2$ and $s_3$ be two states of $TS_1$ ($s_2 = s_3$ is allowed). Then, $(TS_1 \lhd s_2 \rhd TS_2) \lhd s_3 \rhd TS_3 \equiv_{T_1 \cup T_2 \cup T_3} (TS_1 \lhd s_3 \rhd TS_3) \lhd s_2 \rhd TS_2$ (Fig. 4).* □

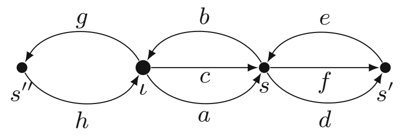

g
b
e
c
f
h
a
d
$s''$
$\iota$
$s$
$s'$
$(TS_1 \lhd s \rhd TS_2) \lhd \iota \rhd TS_5 \equiv (TS_1 \lhd \iota \rhd TS_5) \lhd s \rhd TS_2$

**Fig. 4.** Commutative associativity of articulations.

**Corollary 6.** Sequence articulations
*If $TS_1 = (S_1, \rightarrow_1, T, \iota_1)$ and $TS_2 = (S_2, \rightarrow_2, T, \iota_2)$ with disjoint label sets (i.e., $T_1 \cap T_2 = \emptyset$), if $\forall s_1 \in S_1 \exists \alpha_1 \in T_1^* : s_1[\alpha_1\rangle s$ (s is a home state in $TS_1$) and $\nexists t_1 \in T_1 : s[t_1\rangle$ (s is a dead end in $TS_1$), then $TS_1 \lhd s \rhd TS_2$ behaves like a sequence, i.e., once $TS_2$ has started, it is no longer possible to execute $T_1$.*

*The same occurs when $\iota_2$ does not occur in a non-trivial cycle, i.e., $\iota_2[\alpha_2\rangle\iota_2 \wedge \alpha_2 \in T_2^* \Rightarrow \alpha_2 = \varepsilon$: once $TS_2$ has started, it is no longer possible to execute $T_1$.* □

This is illustrated in Fig. 5. It may be observed that sequences in [9] correspond to the intersection of both cases.

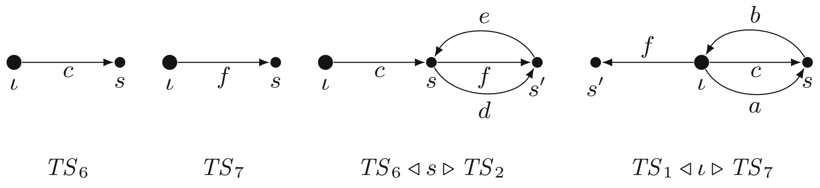


**Fig. 5.** Sequential articulations.

## 3 Petri Nets and Synthesis

**Definition 2.** Petri Nets
An *initially marked Petri net* (PN for short) is denoted as $N = (P, T, F, M_0)$ where $P$ is a set of places, $T$ is a disjoint set of transitions ($P \cap T = \emptyset$), $F$ is the flow function $F\colon ((P \times T) \cup (T \times P)) \rightarrow \mathbb{N}$ specifying the arc weights, and $M_0$ is the initial marking (where a marking is a mapping $M\colon P \rightarrow \mathbb{N}$, indicating the number of tokens in each place).

Two Petri nets $N_1 = (P_1, T, F_1, M_0^1)$ and $N_2 = (P_2, T, F_2, M_0^2)$ with the same transition set $T$ are isomorphic, denoted $N_1 \equiv_T N_2$ (or simply $N_1 \equiv N_2$ if $T$ is clear from the context), if there is a bijection $\zeta\colon P_1 \rightarrow P_2$ such that, $\forall p_1 \in P_1, t \in T\colon M_0^1(p_1) = M_0^2(\zeta(p_1))$, $F_1(p_1, t) = F_2(\zeta(p_1), t)$ and $F_1(t, p_1) = F_2(t, \zeta(p_1))$. We shall usually consider Petri nets up to isomorphism.

A transition $t \in T$ is enabled at a marking $M$, denoted by $M[t\rangle$, if $\forall p \in P\colon M(p) \geq F(p,t)$. The firing of $t$ leads from $M$ to $M'$, denoted by $M[t\rangle M'$, if $M[t\rangle$ and $M'(p) = M(p) - F(p,t) + F(t,p)$. This can be extended, as usual, to $M[\sigma\rangle M'$ for sequences $\sigma \in T^*$, and $[M\rangle$ denotes the set of markings reachable from $M$. The net is bounded if there is $k \in \mathbb{N}$ such that $\forall M \in [M_0\rangle, p \in P : M(p) \leq k$.

The reachability graph $RG(N)$ of $N$ is the labelled transition system with the set of vertices $[M_0\rangle$, initial state $M_0$, label set $T$, and set of edges $\{(M,t,M') \mid M, M' \in [M_0\rangle \wedge M[t\rangle M'\}$. If an LTS $TS$ is isomorphic to the reachability graph of a Petri net $N$, we say[3] that $TS$ is *solvable* and that $N$ *solves* $TS$. A synthesis problem consists in finding a PN solution for a given LTS, when possible.

Let $M_1$ and $M_2$ be two reachable markings of some Petri net $N$. We shall say that $M_1$ is dominated by $M_2$ if $M_1 \lneqq M_2$, i.e., $M_1$ is distinct from $M_2$ and componentwise not greater. □

**Corollary 7.** Independence from isomorphisms
*Let $N_1$ and $N_2$ be two Petri nets. If $N_1 \equiv_T N_2$, then $RG(N_1) \equiv_T RG(N_2)$, so that if $N_1$ solves some LTS $TS$, $N_2$ also solves $TS$.*
*Let $TS_1$ and $TS_2$ be two LTS. If $TS_1 \equiv_T TS_2$ and some Petri net $N$ solves $TS_1$, then $N$ also solves $TS_2$.* □

## 4 Petri Net Synthesis and Articulation

We shall first see that if an articulation is solvable, then each component is individually solvable too.

**Proposition 1.** Synthesis of components of an articulation
*If $TS = (S, \rightarrow, T_1 \uplus T_2, \iota)$ is articulated by $T_1$ and $T_2$ around $s$, so that $T \equiv TS_1 \lhd s \rhd TS_2$ with $TS_1 = (adj(T_1), \rightarrow_1, T_1, \iota)$ and $TS_2 = (adj(T_2), \rightarrow_2, s)$ (see Corollary 1), and is PN-solvable, so is each component $TS_1$ and $TS_2$. Moreover, in the corresponding solution for $TS_1$, the marking corresponding to $s$ is not dominated by any other reachable marking. The same happens for the marking corresponding to $\iota_2$ in $TS_2$ if the latter is finite.*

**Proof:** Let $N = (P, T, F, M_0)$ be a solution for $TS$. It is immediate that $N_1 = (P, T_1, F_1, M_0)$, where $F_1$ is the restriction of $F$ to $T_1$, is a solution for $TS_1$ (but there may be many other ones).

Similarly, if $M$ is the marking of $N$ (and $N_1$) corresponding to $s$, it may be seen that $N_2 = (P, T_2, F_2, M)$, where $F_2$ is the restriction of $F$ to $T_2$, is a solution for $TS_2$ (but there may be many other ones).

Moreover, if $s[t_2\rangle$ for some label $t_2 \in T_2$ and $M'$ is a marking of $N_1$ corresponding to some state $s'$ in $TS_1$ with $M' \gneqq M$, then $s \neq s'$, $s'[t_2\rangle$ and $s$ is not the unique articulation between $T_1$ and $T_2$.

If $M'$ is a reachable marking of $N_2$ with $M' \gneqq M$, then, it is well known that $PN_2$ is unbounded, hence $TS_2$ may not be finite. □

[3] Note that an LTS may be unsolvable, but if it is solvable there are many solutions, sometimes with very different structures.

Note that there may also be solutions to $TS_1$ (other than $N_1$) such that the marking $M$ corresponding to $s$ is dominated, but not if the LTS is reversible, i.e., if $\forall s_1 \in S_1 \exists \alpha_1 \in T_1^* : s_1[\alpha_1\rangle\iota_1$, due to the same infiniteness argument as above. This is illustrated in Fig. 6.

## 5 Recomposition

The other way round, let us now assume that $TS = TS_1 \lhd s \rhd TS_2$ is an articulated LTS and that it is possible to solve $TS_1$ and $TS_2$. Is it possible from that to build a solution of $TS$?

To do that, we shall add the constraint already observed in Proposition 1 that, in the solution of $TS_1$ as well as in the one of $TS_2$, the marking corresponding to $s$ is not dominated by another reachable marking. If this is satisfied we shall say that the solution is *adequate* (with respect to $s$). Hence, in the treatment of the system in Fig. 6, we want to avoid considering the solution $N_1'$ of $TS_1$; on the contrary, $N_1$ or $N_1''$ will be acceptable.

If $TS_2$ is finite, as already mentioned, it is immediate that the initial marking $M_0^2$ (corresponding to $s$) in the solution of $TS_2$ is not dominated by any reachable marking, otherwise there is a path $M_0^2[\alpha\rangle M$ in that solution such that $M_0^2 \lneqq M$ and an infinite path $M_0^2[\alpha^\infty\rangle$, hence also an infinite path $\iota_2[\alpha^\infty\rangle$ in $TS_2$, contradicting the finiteness assumption.

If $TS_1$ is finite and reversible, from a similar argument, no marking reachable in the solution of $TS_1$ is dominated by another one, so that the constraint on $s$ is satisfied. Otherwise, it is possible to force such a solution (if there is one) in the following way:

**Proposition 2.** Forcing an adequate solution for $TS_1$
*Let us add to $TS_1$ an arc $s[u\rangle s$ where $u$ is a new fresh label. Let $TS_1'$ be the LTS so obtained. If $TS_1'$ is not solvable, there is no adequate solution. Otherwise, solve $TS_1'$ and erase $u$ from the solution. Let $N_1$ be the net obtained with the procedure just described: it is a solution of $TS_1$ with the adequate property that the marking corresponding to $s$ is not dominated by another one.*

**Proof:** If there is an adequate solution $N_1$ of $TS_1$, with a marking $M$ corresponding to $s$, let us add a new transition $u$ to it with, for each place $p$ of $N_1$, $W(p,u) = M(p) = W(u,p)$: the reachability graph of this new net is (isomorphic to) $TS_1'$ since $u$ is enabled by marking $M$ (or any larger one, but there is none) and does not modify the marking. Hence, if there is no adequate solution of $TS_1$, there is no solution of $TS_1'$.

Let us now assume there is a solution $N_1'$ of $TS_1'$. The marking $M$ corresponding to $s$ is not dominated otherwise there would be a loop $M'[s\rangle M'$ elsewhere in the reachability graph of $N_1'$, hence also in $TS_1'$. Hence, dropping $u$ in $N_1'$ will lead to an adequate solution of $TS_1$. □

For instance, when applied to $TS_1$ in Fig. 6, this will lead to $N_1''$, and not $N_1'$ ($N_1$ could also be produced, but it is likely that a 'normal' synthesis procedure will not construct the additional isolated place).

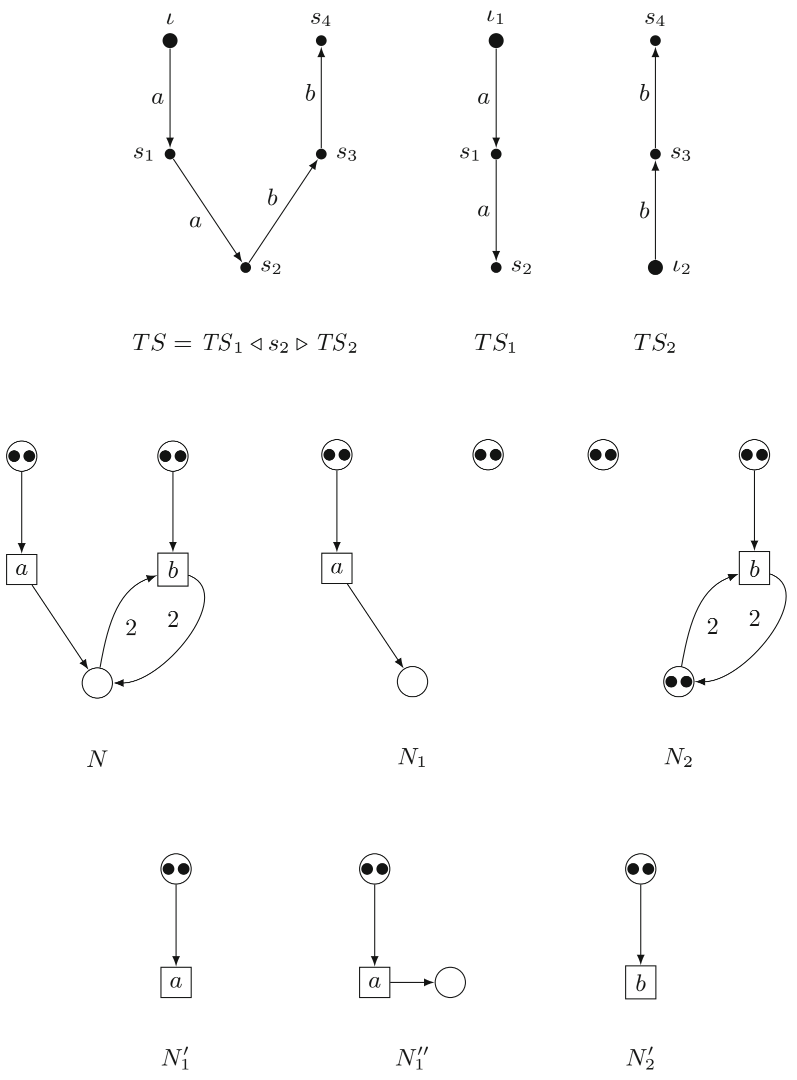


**Fig. 6.** The lts $TS$ is articulated around $s_2$, with $T_1 = \{a\}$ and $T_2 = \{b\}$, hence leading to $TS_1$ and $TS_2$. It is solved by $N$, and the corresponding solutions for $TS_1$ and $TS_2$ are $N_1$ and $N_2$, respectively. $TS_1$ also has the solution $N'_1$ but the marking corresponding to $s_2$ is then empty, hence it is dominated by the initial marking (as well as by the intermediate one). This is not the case for the other solution $N''_1$ (obtained from $N_1$ by erasing the useless isolated place: we never claimed that $N_1$ is a minimal solution). $TS_2$ also has the solution $N'_2$.

Now, to understand how one may generate a solution for $TS$ from the ones obtained for $TS_1$ and $TS_2$, let us first consider the example illustrated in Fig. 7. This leads to the following construction.

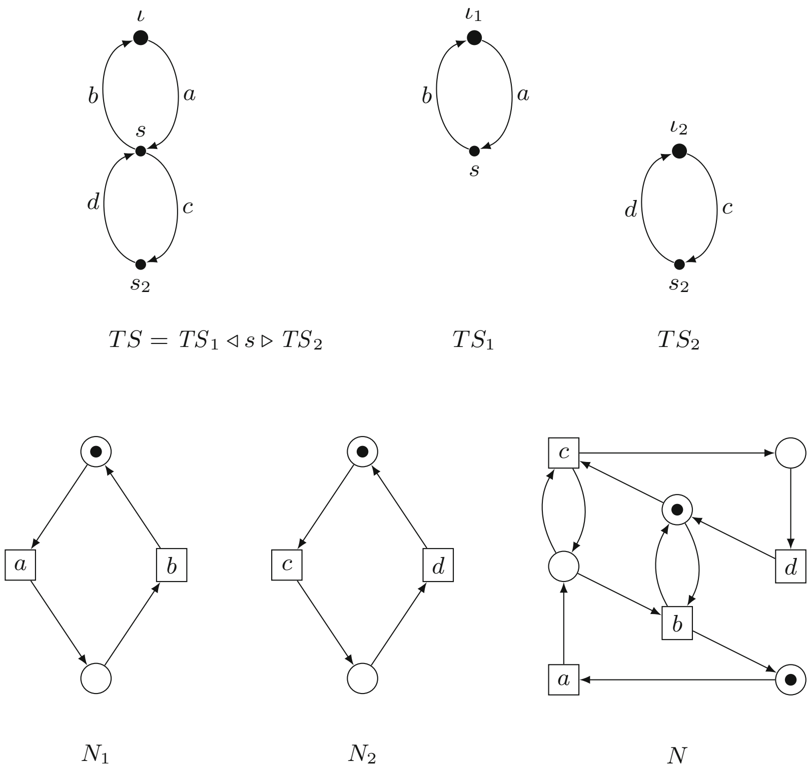


**Fig. 7.** The lts $TS$ is articulated around $s$, with $T_1 = \{a, b\}$ and $T_2 = \{c, d\}$, hence leading to $TS_1$ and $TS_2$. It is solved by $N$, and the corresponding solutions for $TS_1$ and $TS_2$ are $N_1$ and $N_2$, respectively. In $N$, we may recognise $N_1$ and $N_2$, connected by two kinds of side conditions: the first one connects the label $b$ out of $s$ in $TS_1$ to the initial marking of $N_2$, the other one connects the label $c$ out of $\iota_2$ in $TS_2$ to the marking of $N_1$ corresponding to $s$.

**Construction**
Let $TS = TS_1 \lhd s \rhd TS_2$ be an articulation of the LTS $TS$ around $s$ for the partition $T = T_1 \uplus T_2$.
Let $N_1$ be a Petri net solution of $TS_1$, with a non-dominated marking $M_1$ corresponding to $s$, and $N_2$ be a Petri net solution of $TS_2$, with an initial marking $M_2$ that we know to be non-dominated.
Let us assume that the places of $N_1$ and $N_2$ are disjoint, which is possible since we consider nets up to isomorphism, and let us put them side by side.

For each transition $t_1$ out of $s$ in $TS_1$, and each place $p_2$ such that $M_2(p_2) > 0$, let us create a side condition $F(t_1, p_2) = F(p_2, t_1) = M_2(p_2)$.
For each transition $t_2$ out of $\iota_2$ in $TS_2$, and each place $p_1$ such that $M_1(p_1) > 0$, let us create a side condition $F(t_2, p_1) = F(p_1, t_2) = M_1(p_1)$.
The result is a Petri net $N$.
**End of Construction**

**Proposition 3.** Synthesis of articulation
*If $TS_1$ or $TS_2$ are not solvable, so is $TS$.*
*Otherwise, the net $N$ constructed as above is a solution of $TS$.*

**Proof:** The property arises from the observation that $N_1$ with the additional side conditions behaves like the original $N_1$ provided that, when we reach $M_1$, $N_2$ does not leave $M_2$. Similarly, $N_2$ with the added side conditions behaves like the original $N_2$ provided $N_1$ reached $M_1$ and stays there, until $N_2$ returns to $M_2$. □

Note that we do not claim this is the only solution, but the goal is to find a solution when there is one.

## 6 Decomposition

It remains to show when and how an LTS may be decomposed by an articulation (or several ones). Let us thus consider some LTS $TS = (S, \rightarrow, T, \iota)$. We may assume it is totally reachable (the states which may not be reached from $\iota$ play no role in the evolutions of the system) and that the label set $T$ is finite (otherwise, it may happen that the finest decomposition is infinite. Usually we shall also assume that the state set $S$ is also finite, otherwise there may be a problem to implement the procedure we are about to describe in a true algorithm. We may also assume it is deterministic, i.e., $(s[t\rangle \wedge s[t'\rangle) \Rightarrow t = t'$ and $([t\rangle s \wedge [t'\rangle s) \Rightarrow t = t'$ for any state $s \in S$ and labels $t, t' \in T$, otherwise there may be no unlabelled Petri net solution.

First, we may observe that, for any two distinct labels $t, t' \in T$, if $|adj(\{t\}) \cap adj(\{t'\})| > 1$, $t$ and $t'$ must belong to a same subset for defining an articulation (if any). Let us extend the function $adj$ to non-empty subsets of labels by stating $adj(T') = \cup_{t \in T'} adj(t)$ when $\emptyset \subset T' \subset T$ . We then have that, if $\emptyset \subset T_1, T_2 \subset T$ and we know that all the labels in $T_1$ must belong to a same subset for defining an articulation, and similarly for $T_2$, $|adj(T_1) \cap adj(T_2)| > 1$ implies that $T_1 \cup T_2$ must belong to a same subset of labels defining an articulation (if any). If we get the full set $T$, that means that there is no possible articulation (but trivial ones, that we excluded from this study).

Hence, starting from any partition $\mathcal{T}$ of $T$ (initially, if $T = \{t_1, t_2, \ldots, t_n\}$, we shall start from the finest partition $\mathcal{T} = \{\{t_1\}, \{t_2\}, \ldots, \{t_n\}\}$), we shall construct the finest partition compatible with the previous rule:

**while there is $T_1, T_2 \in \mathcal{T}$ such that $T_1 \neq T_2$ and $|adj(T_1) \cap adj(T_2)| > 1$, replace $T_1$ and $T_2$ in $\mathcal{T}$ by $T_1 \cup T_2$.**

At the end, if $\mathcal{T} = \{T\}$, we may stop with the result: *there is no non-trivial articulation.*

Otherwise, we may define a finite bipartite undirected graph whose nodes are the members of the partition $\mathcal{T}$ and some states of $S$, such that if $T_i, T_j \in \mathcal{T}, T_i \neq T_j$ and $adj(T_i) \cap adj(T_j) = \{s\}$, there is a node $s$ in the graph, connected to $T_i$ and $T_j$ (and this is the only reason to have a state as a node of the graph). Since $TS$ is weakly live and totally reachable, this graph is connected, and each state occurring in it has at least two neighbours (on the contrary, a subset of labels may be connected to a single state). Indeed, since $TS$ is weakly live, $\cup_{T' \in \mathcal{T}} adj(T') = S$. Each state $s$ occurring as a node in the graph is connected to at least two members of the $\mathcal{T}$, by the definition of the introduction of $s$ in the graph. Let $T_1$ be the member of $\mathcal{T}$ such that $\iota \in adj(T_1)$, let $T_i$ be any other member of $\mathcal{T}$, and let us consider a path $\iota[\alpha\rangle$ ending with some $t \in T_i$ (we may restrict our attention to a short such path, but this is not necessary): each time there is a sequence $t't''$ in $\alpha$ such that $t'$ and $t''$ belong to two different members $T'$ and $T''$ of $\mathcal{T}$, we have $[t'\rangle s[t''\rangle$, where $s$ is the only state-node connected to $T'$ and $T''$, hence in the graph we have $T' \to s \to T''$. This will yield a path in the constructed graph going from $T_1$ to $T_i$, hence the connectivity.

If there is a cycle in this graph, that means that there is no way to group the members of $\mathcal{T}$ in this cycle in two subsets such that the corresponding adjacency sets only have a single common state. Hence we need to fuse all these members, for each such cycle, leading to a new partition, and we also need to go back to the refinement of the partition in order to be compatible with the intersection rule, and to the construction of the graph.

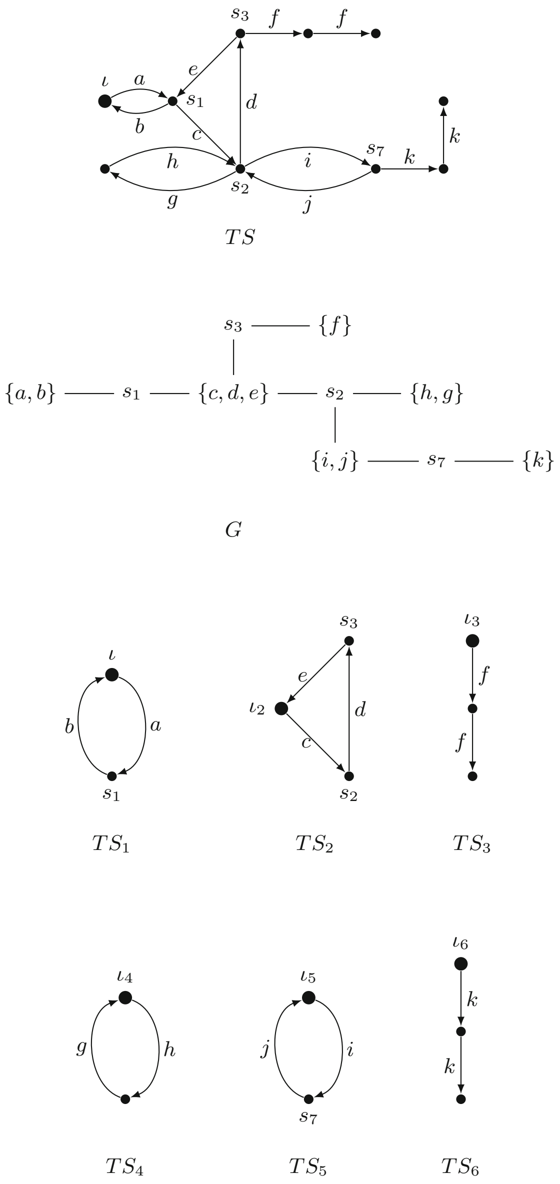
Finally, we shall get an acyclic graph $G$, with at least three nodes (otherwise we stopped the articulation algorithm).

We shall now define a procedure $articul(SG)$ that builds an LTS expression based on articulations from a subgraph $SG$ of $G$ with a chosen state-node root. We shall then apply it recursively to $G$, leading finally to an articulation-based (possibly complex) expression equivalent to the original LTS $TS$.

The basic case will be that, if $SG$ is a graph composed of a state $s$ connected to a subset node $T_i$, $articul(SG)$ will be the LTS $TS_i = (adj(T_i), T_i, \to_i, s)$ (as usual $\to_i$ is the projection of $\to$ on $T_i$; by construction, it will always be the case that $s \in adj(T_i)$).

First, if $\iota$ is a state-node of the graph, $G$ then has the form of a star with root $\iota$ and a set of satellite subgraphs $G_1, G_2, \ldots, G_n$ ($n$ is at least 2). Let us denote by $SG_i$ the subgraph with root $\iota$ connected to $G_i$: the result will then be the (commutative, see Corollary 4) articulation around $\iota$ of all the LTSs $articul(SG_i)$.

Otherwise, let $T_1$ be the (unique) label subset in the graph such that $\iota \in adj(T_1)$. $G$ may then be considered as a star with $T_1$ at the center, surrounded by subgraphs $SG_1, SG_2, \ldots, SG_n$ (here $n$ may be 1), each one with a root $s_i$ connected to $T_1$ (we have here that $s_i \in adj(T_1)$, and we allow $s_i = s_j$): the result is then $((\ldots((adj(T_1), T_1, \to_1, \iota) \lhd s_1 \rhd articul(SG_1)) \lhd s_2 \rhd articul(SG_2)) \ldots) \lhd s_n \rhd articul(SG_n))$. Note that, if $n > 1$, the order in which we consider the subgraphs is irrelevant from Corollary 5.



$$TS \equiv TS_1 \lhd s_1 \rhd (((TS_2 \lhd s_3 \rhd TS_3) \lhd s_2 \rhd TS_4) \lhd s_2 \rhd (TS_5 \lhd s_7 \rhd TS_6))$$

**Fig. 8.** The lts $TS$ leads to the graph $G$. The corresponding components are $TS_1$ to $TS_6$, which may easily be synthesised; note that, from the total reachability of $TS$, they are all totally reachable themselves. This leads to the articulated expression below.

Finally, if a subgraph starts from a state $s'$, followed by a subset $T'$, itself followed by subgraphs $SG_1$, $SG_2, \ldots, SG_n$ ($n \geq 1$; if it is 0 we have the base case), each one with a root $s_i$ connected to $T'$ (we have here that $s' \in adj(T')$, and we allow $s_i = s_j$): the result is then $((\ldots((adj(T'), T', \rightarrow', s') \lhd s_1 \rhd articul(SG_1)) \lhd s_2 \rhd articul(SG_2)) \ldots) \lhd s_n \rhd articul(SG_n))$. Again, if $n > 1$, the order in which we consider the subgraphs is irrelevant from Corollary 5.

This procedure is illustrated in Fig. 8.

## 7 Concluding Remarks

We have developed a theory around a new operator acting on labelled transition systems, that we called articulation. Its main algebraic properties have been exhibited, and it was shown how this may be used to construct syntheses from the solutions of the various components. Since the latter are simpler than the original LTS (when articulation is possible), it is also much simpler to synthesise them (when possible), hence speeding up the global synthesis, or the detection that this is not possible (while pointing at the culprit components). A procedure has also been devised to decompose a given LTS into articulated components, when possible.

It remains to build effectively the corresponding procedures, and to incorporate them in some existing tool, like APT [11].

Other possible issues are to examine how this may be specialised for some subclasses of Petri nets, like Choice-Free ones, where each place has at most one outgoing transition: this is exactly the class of Petri nets allowing fully distributed implementations [6], and they present interesting behavioural properties [5,10] which could be exploited.

Another possible extension would be to look how these articulations behave in the context of approximate solutions devised when an exact synthesis is not possible, in the spirit of the notions and procedures developed in [15].

Finally, other kinds of LTS operators could be searched for, having interesting decomposition procedures, and corresponding to compositions of component solutions allowing to speed up synthesis problems.

**Acknowledgements.** The author thanks Eike Best as well as the anonymous referees for their useful remarks and suggestions.

## References

1. Arnold, A.: Finite Transition Systems - Semantics of Communicating Systems. Prentice Hall International Series in Computer Science. Prentice Hall, Hertfordshire (1994)
2. Badouel, E., Bernardinello, L., Darondeau, P.: Petri Net Synthesis. TTCSAES. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47967-4

3. Badouel, E., Bernardinello, L., Darondeau, P.: Polynomial algorithms for the synthesis of bounded nets. In: Mosses, P.D., Nielsen, M., Schwartzbach, M.I. (eds.) CAAP 1995. LNCS, vol. 915, pp. 364–378. Springer, Heidelberg (1995). https://doi.org/10.1007/3-540-59293-8_207
4. Badouel, E., Bernardinello, L., Darondeau, P.: The synthesis problem for elementary net systems is NP-complete. Theor. Comput. Sci. **186**(1–2), 107–134 (1997). https://doi.org/10.1016/S0304-3975(96)00219-8
5. Best, E., Devillers, R., Schlachter, U., Wimmel, H.: Simultaneous Petri net synthesis. Sci. Ann. Comput. Sci. **28**(2), 199–236 (2018)
6. Best, E., Darondeau, P.: Petri net distributability. In: Clarke, E., Virbitskaite, I., Voronkov, A. (eds.) PSI 2011. LNCS, vol. 7162, pp. 1–18. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29709-0_1
7. Best, E., Devillers, R.: Characterisation of the state spaces of live and bounded marked graph Petri nets. In: Dediu, A.-H., Martín-Vide, C., Sierra-Rodríguez, J.-L., Truthe, B. (eds.) LATA 2014. LNCS, vol. 8370, pp. 161–172. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-04921-2_13
8. Best, E., Devillers, R., Koutny, M.: Petri Net Algebra. Monographs in Theoretical Computer Science. An EATCS Series. (2001). https://doi.org/10.1007/978-3-662-04457-5
9. Best, E., Devillers, R., Koutny, M.: The box algebra = Petri nets + process expressions. Inf. Comput. **178**(1), 44–100 (2002). https://doi.org/10.1006/inco.2002.3117
10. Best, E., Devillers, R., Schlachter, U.: Bounded choice-free Petri net synthesis: algorithmic issues. Acta Informatica **55**(7), 575–611 (2018)
11. Best, E., Schlachter, U.: Analysis of Petri nets and transition systems. In: Proceedings 8th Interaction and Concurrency Experience, ICE 2015, Grenoble, France, 4–5th June 2015, pp. 53–67 (2015). https://doi.org/10.4204/EPTCS.189.6
12. Devillers, R.: Factorisation of transition systems. Acta Informatica **55**(4), 339–362 (2018)
13. Devillers, R., Schlachter, U.: Factorisation of Petri net solvable transition systems. In: Application and Theory of Petri Nets and Concurrency - 39th International Conference, PETRI NETS 2018, Bratislava, Slovakia, pp. 82–98 (2018)
14. Hopcroft, J.E., Tarjan, R.E.: Efficient algorithms for graph manipulation [H] (algorithm 447). Commun. ACM **16**(6), 372–378 (1973)
15. Schlachter, U.: Over-approximative Petri net synthesis for restricted subclasses of nets. In: Klein, S.T., Martín-Vide, C., Shapira, D. (eds.) LATA 2018. LNCS, vol. 10792, pp. 296–307. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-77313-1_23
16. Westbrook, J., Tarjan, R.E.: Maintaining bridge-connected and biconnected components on-line. Algorithmica **7**(5&6), 433–464 (1992). https://doi.org/10.1007/BF01758773