Synthesis of Persistent Systems

Eike Best[1,⋆] and Raymond Devillers[2]

[1] Department of Computing Science,
Carl von Ossietzky Universität Oldenburg, Germany
eike.best@informatik.uni-oldenburg.de
[2] Département d'Informatique, Université Libre de Bruxelles, Belgium
rdevil@ulb.ac.be

Abstract. This paper presents efficient, specialised synthesis and reengineering algorithms for the case that a transition system is finite, persistent and reversible. It also shows by means of a complex example that structural properties of the synthesised Petri nets may not necessarily be entailed.

Keywords: Cyclic Behaviour, Persistency, Labelled Transition Systems, Parikh Vectors, Petri Nets, Region Theory, System Synthesis, Reengineering.

1 Introduction

In the realm of (asynchronous) hardware, persistency [14] is a significant, desirable property, as it is related to the absence of hazards [12,18,19], as well as to arbiter-free synchronisation [13]. Persistency means that an enabled transition can never become disabled through occurrences of other transitions.

In this paper, we focus our attention on a class of persistent labelled transition systems (lts) generated by Petri nets. A relatively weak structural restriction for a Petri net to have a persistent reachability graph is that its places have at most one outgoing transition. Such Petri nets will be called ON in this paper, for "(place-)output-nonbranching".

We investigate conditions under which, conversely, a given persistent lts is isomorphic to the reachability graph of an unknown ON Petri net. Moreover, we define an algorithm that creates such a net in case it is theoretically possible. Eventually, such an algorithm could be useful in the automatic generation of asynchronous hardware from persistent specifications.

These results can be seen as a variant of the Petri net synthesis problem addressed by region theory [1,3]. Since one of the conditions we use is that the given lts is already generated by some Petri net, our results can also be seen as a variant of the reengineering problem, asking whether a given Petri net can be transformed – under invariance of its reachability graph – into an ON Petri net.

The paper is structured as follows. Labelled transition systems, Petri nets, and regions are briefly introduced in section 2. Section 3 delineates the class of

⋆ The first author gratefully acknowledges the support of Université d'Évry-Val-d'Essonne and Université Paris-Est Créteil-Val-de-Marne.

lts we shall study and lists several auxiliary results about this class. Sections 4 and 5 contain the main results: a dedicated test which allows to check the ON implementability of a persistent lts specification; an efficient algorithm producing a generating ON Petri net if one exists; and examples demonstrating the necessity and (non-)sufficiency of various conditions. Section 6 concludes.

2 Labelled Transition Systems, Petri Nets, and Regions

Definition 1. LTS, REACHABILITY, PARIKH VECTORS, CYCLES, EQUIVALENCES

An lts (labelled transition system with initial state) is a tuple $(S, \to, T, s_0)$, where S is a set of *states*; T is a set of *labels* with $S \cap T = \emptyset$; $\to \subseteq (S \times T \times S)$ is the *transition relation*; and $s_0 \in S$ is an *initial state*. A label t is *enabled* in a state s, denoted by $s[t\rangle$, if there is some state s' such that $(s, t, s') \in \to$. We also use the notation $s[t\rangle s'$, meaning that s' is *reachable* from s through the execution of t, instead of $(s, t, s') \in \to$. We denote by $s^\bullet = \{t \in T \mid s[t\rangle\}$ the set of labels enabled at s, and by ${}^\bullet s = \{t \in T \mid s'[t\rangle s \text{ for some } s' \in S\}$ the set of labels leading to s. The definitions of enabledness and of the reachability relation are extended to label sequences (or directed paths) $\sigma \in T^*$:

$s[\varepsilon\rangle$ and $s[\varepsilon\rangle s$ are always true;

$s[\sigma t\rangle$ ($s[\sigma t\rangle s'$) iff there is some s'' with $s[\sigma\rangle s''$ and $s''[t\rangle$ ($s''[t\rangle s'$, respectively). A state s' is reachable from state s if there exists a label sequence σ such that $s[\sigma\rangle s'$. By $[s\rangle$, we denote the set of states reachable from s. For a finite sequence $\sigma \in T^*$ of labels, the *Parikh vector* $\Psi(\sigma)$ is a T-vector (i.e., a vector of natural numbers with index set T), where $\Psi(\sigma)(t)$ denotes the number of occurrences of t in σ. $s[\sigma\rangle s'$ is called a *cycle* (at state s) if $s = s'$. The cycle is *nontrivial* if $\sigma \neq \varepsilon$. A nontrivial cycle $s[\sigma\rangle s$ around a reachable state $s \in [s_0\rangle$ is called *small* if there is no nontrivial cycle $s'[\sigma'\rangle s'$ with $s' \in [s_0\rangle$ and $\Psi(\sigma') \lneqq \Psi(\sigma)$.

Two lts $(S_1, \to_1, T, s_{01})$ and $(S_2, \to_2, T, s_{02})$ over the same set of labels will be called *language-equivalent* if their initially enabled sequences coincide, i.e., if $\forall \sigma \in T^* \colon s_{01}[\sigma\rangle \iff s_{02}[\sigma\rangle$, and *isomorphic* if there is a bijection $\zeta \colon S_1 \to S_2$ with $\zeta(s_{01}) = s_{02}$ and $(s, t, s') \in \to_1 \iff (\zeta(s), t, \zeta(s')) \in \to_2$, for all $s, s' \in S_1$. □ 1

Definition 2. BASIC PROPERTIES OF LTS

A labelled transition system $(S, \to, T, s_0)$ is called *finite* if S and T (hence also $\to$) are finite sets; *deterministic* if for any reachable state s and label a, $s[a\rangle s'$ and $s[a\rangle s''$ imply $s' = s''$; *totally reachable* if $S = [s_0\rangle$ and $\forall t \in T \exists s \in [s_0\rangle \colon s[t\rangle$; *reversible* if $\forall s \in [s_0\rangle \colon s_0 \in [s\rangle$; *persistent* if for all reachable states s and labels t, u, if $s[t\rangle$ and $s[u\rangle$ with $t \neq u$, then there is some state $r \in S$ such that both $s[tu\rangle r$ and $s[ut\rangle r$. □ 2

Definition 3. PETRI NETS, MARKINGS, REACHABILITY GRAPHS

A (finite, initially marked, place-transition, arc-weigthed) Petri net is a tuple (P, T, F, M_0) such that P is a finite set of *places*, T is a finite set of *transitions*, with $P \cap T = \emptyset$, F is a *flow* function $F \colon ((P \times T) \cup (T \times P)) \to \mathbb{N}$, M_0 is the *initial marking*, where a *marking* is a mapping $M\colon P \to \mathbb{N}$, indicating the number of *tokens* in each place. A transition $t \in T$ is *enabled by* a marking M, denoted by $M[t\rangle$, if for all places $p \in P$, $M(p) \geq F(p,t)$. If t is enabled at M, then t can *occur* (or *fire*) in M, leading to the marking M' defined by $M'(p) = M(p) - F(p,t) + F(t,p)$ (notation: $M[t\rangle M'$, and $[M_0\rangle$ again denotes the set of reachable markings). The *reachability graph of* N, with initial marking M_0, is the labelled transition system with the set of vertices $[M_0\rangle$ and set of arcs $\{(M,t,M') \mid M,M' \in [M_0\rangle \wedge M[t\rangle M'\}$. If an lts TS is isomorphic to the reachability graph of a Petri net N, then we will say that N *solves* TS. $\square$ 3

Definition 4. Basic properties of Petri nets

For a place p of a Petri net $N = (P,T,F,M_0)$, let ${}^\bullet p = \{t \in T \mid F(t,p) > 0\}$ its pre-places, and $p^\bullet = \{t \in T \mid F(p,t) > 0\}$ its post-places. N is called *connected* if it is weakly connected as a graph; *plain* if $cod(F) \subseteq \{0,1\}$; *pure* or *side-condition free* if $p^\bullet \cap {}^\bullet p = \emptyset$ for all places $p \in P$; ON if $|p^\bullet| \leq 1$ for all places $p \in P$; a *marked graph* if it is plain and $|p^\bullet| \leq 1$ and $|{}^\bullet p| \leq 1$ for all places $p \in P$.

N is called *weakly live* if $\forall t \in T \exists M \in [M_0\rangle\colon M[t\rangle$ (i.e., there are no unfireable transitions); *k-bounded*, for some $k \in \mathbb{N}$, if $\forall M \in [M_0\rangle \forall p \in P\colon M(p) \leq k$ (i.e., the number of tokens on any place never exceeds k); *bounded* if it is k-bounded for some k; *persistent* (*reversible*) if so is its reachability graph. $\square$ 4

The class of ON nets has also been called CF (for *Choice-Free* nets) in [16], but to avoid an easy confusion with free-choice nets [10] or conflict-free nets [14], we shall here stick to the above terminology.

In the remainder of this paper, attention will be restricted to bounded and weakly live Petri nets. It is easy to see that the reachability graphs of such nets are finite (by boundedness), deterministic (coming from a Petri net), and totally reachable (by weak liveness).

The synthesis problem consists of finding a Petri net solving a given lts in the sense of Definition 3. In order to study conditions for such solutions to exist, regions have been introduced as follows.

Definition 5. Regions of lts

Let $TS = (S, \to, T, s_0)$ be an lts. A triple

$$\rho = (\mathbb{R}, \mathbb{B}, \mathbb{F}) \in (S \to \mathbb{N}, T \to \mathbb{N}, T \to \mathbb{N})$$

is a *region* of TS if, for all $s[t\rangle s'$ with $s \in [s_0\rangle$, $\mathbb{R}(s) \geq \mathbb{B}(t)$ and $\mathbb{R}(s') = \mathbb{R}(s) - \mathbb{B}(t) + \mathbb{F}(t)$. $\square$ 5

A region mimicks, at the level of an lts, the properties of a Petri net place p. More precisely, $\mathbb{R}(s)$ mimicks the marking of p in state s, $\mathbb{B}(t)$ the weight of the arc from p to t, and $\mathbb{F}(t)$ weight of the arc from t to p ($\mathbb{B}$ stands for "backward", $\mathbb{F}$ for "forward", as seen from transitions). For instance, suppose that TS is the reachability graph of a Petri net $N = (P,T,F,M_0)$, and let $p \in P$. For any $M \in [M_0\rangle$, define $\mathbb{R}_p(M) = M(p)$, and for any $t \in T$, define $\mathbb{B}_p(t) = F(p,t)$ and

$\mathbb{F}_p(t) = F(t,p)$. Then $(\mathbb{R}_p, \mathbb{B}_p, \mathbb{F}_p)$ is a region of TS. The region properties in Definition 5 correspond to the notions of enabling and firing in Definition 3.

An lts $TS = (S, \rightarrow, T, s_0)$ satisfies SSP (state separation property) iff

$$\forall s, s' \in [s_0\rangle\colon \quad s \neq s' \quad \Rightarrow \quad \exists \text{ region } \rho = (\mathbb{R}, \mathbb{B}, \mathbb{F}) \text{ with } \mathbb{R}(s) \neq \mathbb{R}(s')$$

meaning that it is possible to distinguish the various states in terms of markings. TS satisfies ESSP (event/state separation property) iff

$$\forall s \in [s_0\rangle\, \forall t \in T\colon \quad (\neg s[t\rangle) \quad \Rightarrow \quad \exists \text{ region } \rho = (\mathbb{R}, \mathbb{B}, \mathbb{F}) \text{ with } \mathbb{R}(s) < \mathbb{B}(t)$$

meaning that it is possible to exclude forbidden transitions through a marking.

Theorem 1. Basic region theorem for place/transition nets
A (finite, deterministic, totally reachable) lts TS is isomorphic to the reachability graph of a (possibly non-plain, or non-pure) Petri net iff TS satisfies SSP *and* ESSP*.* □ 1

In the proof of this result (e.g. [1,3]), it turns out that ESSP without SSP allows to build a Petri net with the same language as the given lts, but not necessarily satisfying the requested isomorphism.

3 Some Classes of Labelled Transition Systems

In section 3.1, the class of lts considered in this paper is motivated and introduced. In section 3.2, some basic properties of this class of lts are documented.

3.1 Persistency, Uniform Small Cycles, and the ON Property

Let $TS = (S, \rightarrow, T, s_0)$ be some lts. Let $\Upsilon\colon T \rightarrow \mathbb{N} \setminus \{0\}$ be a fixed vector with no zero entries. The principal properties we study are the following ones.

rg	TS is the reachability graph of some bounded Petri net.
r	TS is reversible.
p	TS is persistent.
PΥ	All small cycles of TS have Parikh vector Υ.

Special cases of **P**Υ are **P**1 (Υ is the all-ones vector), **P**2 (Υ is the all-twos vector), and so on. For instance, Figure 1 shows an lts satisfying all properties **rg** to **P**1. Two solutions of this lts are depicted: a plain non-ON one (in the middle of the figure), and a non-plain ON one (on the right-hand side). Figure 2 shows an lts satisfying **rg** to **P**2. This lts has a solution, as shown in the figure, but no ON solution, as will be proved later.

The interest of property **P**Υ arises from results in [4]. These results show that if an lts satisfies **rg**, **r** and **p**, then it may essentially be expressed as a direct product of label-disjoint lts, each of which satisfies **P**Υ, for some vector Υ. It is

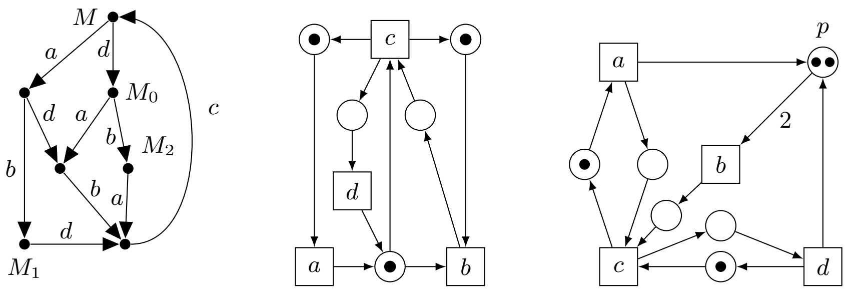

Fig. 1. An lts satisfying **rg**, **r**, **p**, and **P1**, with two different solutions

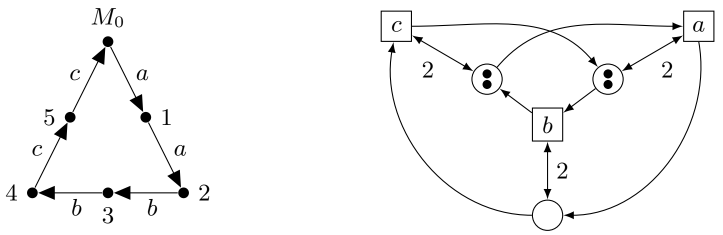

Fig. 2. An lts satisfying **rg**, **r**, **p**, and **P2**, but having no ON Petri net solution. A non-pure, non-plain, and non-ON solution is shown on the right-hand side.

also shown in the same paper that there is a small cycle around each state, and that the Parikh vector of each cycle is a linear combination of the Υ.

If some transition t is the only outgoing transition of a place p, then no other transition can reduce the number of tokens on p. Thus, ON Petri nets are a subclass of persistent Petri nets, and the results of [4] can be specialised as follows:

Theorem 2. Properties of ON Petri nets
The reachability graph TS of a connected, bounded, weakly live, reversible, ON Petri net N is finite and satisfies **rg**, **r**, **p**, *and* **P**Υ, *for some* Υ.

Proof: All claims but **P**Υ are obvious.

If a place p is isolated (i.e., $p^\bullet = \emptyset = {}^\bullet p$), then since the net is connected, the transition set is empty, and **P**Υ is vacuously true.

If p is not isolated, it must have both input and output transitions, because the net is weakly live and reversible. Since the net is ON, p has a unique output transition; let it be t. From results in [4], t belongs to some small cycle; let T' be the unique set of labels occurring in this cycle. If p has an input transition t' not in T', p is not bounded, since there is a (small) cycle containing t' but not t; it is possible to reach it; and following indefinitely that cycle will indefinitely increase the marking of p. Hence, since the net is connected, $T' = T$, and **P**Υ is satisfied with Υ being the Parikh vector of any small cycle. □ 2

This result suggests a close relationship between persistent lts and ON Petri nets, motivating the following (in a sense, converse) question which was raised in [5]:

> *If an lts satisfies persistency and a set of other strong properties, viz.* **rg** *and* **r** *and* $\mathbf{P}\Upsilon$*, does there always exist an ON Petri net generating it?*

Figure 2 shows that the answer is negative for general $\mathbf{P}\Upsilon$. However, it was not known until more recently that the answer is still negative if $\mathbf{P}\Upsilon$ is strengthened to $\mathbf{P}1$, and further conditions are imposed.

The theory developed in section 4 will lead to an efficient algorithm allowing to synthesise (and reengineer, if possible) labelled transition systems satisfying **rg**, **r**, **p**, and $\mathbf{P}\Upsilon$, such as the one shown in Figure 1. The same theory also leads to a method for proving that examples such as the one shown in Figure 2, as well as a more complicated one we will exhibit later, do *not* have ON solutions.

3.2 Some Properties of lts Satisfying rg, r, p, and $\mathbf{P}\Upsilon$

Let $TS = (S, \to, T, s_0)$ be an lts satisfying properties **rg**, **r**, **p**, and $\mathbf{P}\Upsilon$.

First, we briefly recapitulate Keller's theorem [11]. For sequences $\sigma, \tau \in T^*$, $\tau \overset{\bullet}{-} \sigma$ denotes the *residue* of τ with respect to σ, i.e., the sequence left after cancelling successively in τ the leftmost occurrences of all symbols from σ, read from left to right. Formally and inductively: for $t \in T$, $\tau \overset{\bullet}{-} t = \tau$ if $\Psi(\tau)(t) = 0$; $\tau \overset{\bullet}{-} t = \tau_1\tau_2$ if $\tau = \tau_1 t \tau_2$ and $\Psi(\tau_1)(t) = 0$; $\tau \overset{\bullet}{-} \varepsilon = \tau$; and $\tau \overset{\bullet}{-} (t\sigma) = (\tau \overset{\bullet}{-} t) \overset{\bullet}{-} \sigma$.

Theorem 3. Keller's theorem
If $s[\tau\rangle$ and $s[\sigma\rangle$ for some $s \in [s_0\rangle$, then $s[\tau(\sigma \overset{\bullet}{-} \tau)\rangle s'$ and $s[\sigma(\tau \overset{\bullet}{-} \sigma)\rangle s''$ as well as $\Psi(\tau(\sigma \overset{\bullet}{-} \tau)) = \Psi(\sigma(\tau \overset{\bullet}{-} \sigma))$ and $s' = s''$. □ 3

Definition 6. Short paths, and distances
Let r, s be two states of TS. A path $r[\tau\rangle s$ will be called *short* if $|\tau| \le |\tau'|$ for every path $r[\tau'\rangle s$, where $|\tau|$ denotes the length of τ. We shall denote by $\Delta_{r,s}$ the Parikh vector of some short path from r to s, and call it the *distance* between r and s. □ 6

According to Lemma 2 below, the definition of $\Delta_{r,s}$ does not depend on the choice of the short path from r to s. For a label t, the number $\Delta_{r,s}(t)$ thus simply indicates how often t occurs on *any* short path from r to s.

Lemma 1. Characterisation of short paths
Suppose that $s[\tau\rangle s'$. Then $s[\tau\rangle s'$ is short iff $\neg(\Upsilon \le \Psi(\tau))$.

Proof: ($\Rightarrow$): By contraposition. Suppose that $s[\tau\rangle s'$ and that $\Upsilon \le \Psi(\tau)$. By results in [4], there is some cycle $s[\kappa\rangle s$ with $\Psi(\kappa) = \Upsilon$. By Keller's theorem, $s[\tau\rangle s'[\kappa \overset{\bullet}{-} \tau\rangle s''$ and $s[\kappa\rangle s[\tau \overset{\bullet}{-} \kappa\rangle s''$. But $\Psi(\kappa) = \Upsilon \le \Psi(\tau)$ implies $\kappa \overset{\bullet}{-} \tau = \varepsilon$. Therefore, $s' = s''$ and $s[\tau \overset{\bullet}{-} \kappa\rangle s'$. Since κ contains every transition at least once and $\Upsilon \le \Psi(\tau)$, $|\tau \overset{\bullet}{-} \kappa| < |\tau|$. Hence $s[\tau\rangle s'$ is not short.

($\Leftarrow$): Suppose that $s[\tau\rangle s'$ and $\neg(\Upsilon \leq \Psi(\tau))$. Consider any other path $s[\tau'\rangle s'$; we show $|\tau| \leq |\tau'|$. By reversibility, there is some path ρ from s' to s. Both $s'[\rho\tau\rangle s'$ and $s'[\rho\tau'\rangle s'$ are cycles at s'. By results from [4], they can be permuted into sequences of small cycles. Therefore, $\Psi(\rho\tau) = \ell{\cdot}\Upsilon$ and $\Psi(\rho\tau') = \ell'{\cdot}\Upsilon$. If $\ell > \ell'$, then $\Psi(\tau) \geq \Psi(\tau) - \Psi(\tau') = \Psi(\rho\tau) - \Psi(\rho\tau') = (\ell - \ell'){\cdot}\Upsilon \geq \Upsilon$, contradicting $\neg(\Upsilon \leq \Psi(\tau))$. Hence $\ell \leq \ell'$ and $\Psi(\tau) \leq \Psi(\tau')$ and $|\tau| \leq |\tau'|$. □ 1

Lemma 2. Uniqueness of short Parikh vectors
Suppose that $s[\tau\rangle s'$ and $s[\tau'\rangle s'$ are both short. Then $\Psi(\tau) = \Psi(\tau')$.

Proof: By Lemma 1($\Rightarrow$), both $\neg(\Upsilon \leq \Psi(\tau))$ and $\neg(\Upsilon \leq \Psi(\tau'))$. As in the proof of Lemma 1($\Leftarrow$), $\Psi(\tau) \leq \Psi(\tau')$ and $\Psi(\tau') \leq \Psi(\tau)$, hence $\Psi(\tau) = \Psi(\tau')$. □ 2

Lemma 3. Characterisation of Parikh vectors of paths
Suppose that $s[\tau\rangle s'$. Then $\Psi(\tau) = \Psi(\tau') + m{\cdot}\Upsilon$, with some number $m \in \mathbb{N}$, where $s[\tau'\rangle s'$ is any short path from s to s'.

Proof: Assume that $s[\tau\rangle s'$. Let m be the maximal number in $\mathbb{N}$ such that $\Psi(m{\cdot}\Upsilon) \leq \Psi(\tau)$. Let $s[\kappa\rangle s$ be some cycle with $\Psi(\kappa) = \Upsilon$. Then also $s[\kappa^m\rangle s$, with $\Psi(\kappa^m) = m{\cdot}\Upsilon$. By Keller's theorem, $s[\kappa^m\rangle s[\tau'\rangle s'$, with $\tau' = \tau \stackrel{\bullet}{-} \kappa^m$. By the maximality of m, $s[\tau'\rangle s'$ is short, and by $\Psi(\kappa^m) \leq \Psi(\tau)$, $\Psi(\tau)$ can be written as $\Psi(\tau) = \Psi(\tau') + \Psi(\kappa^m)$. By Lemma 2, the choice of τ' is arbitrary. □ 3

Lemma 4. Existence of short paths
Suppose that s, s' are states. There is a short path from s to s'.

Proof: By reversibility, $s[\tau\rangle s'$ for some τ. Just take the path $s[\tau'\rangle s'$ from the proof of Lemma 3. □ 4

Lemma 5. A repeat lemma for plain nets
Assume that the Petri net N generating TS by **rg** *is plain, and that b does not occur in τ. If $s[\tau\tau' b\rangle$ and $\Psi(\tau) = \Psi(\tau')$, then also $s[\tau b \tau'\rangle$.*

Proof: Suppose $s[\tau\rangle s'[\tau'\rangle s''[b\rangle s'''$ and assume that b is not enabled in state s'. By plainness, this implies that in N, there is some pre-place p of b which has zero tokens in s'. The total effect of τ' on p is to create at least one token on p, because b is enabled at s''. But since $\Psi(\tau) = \Psi(\tau')$, the total effect of τ is the same as that of τ', which implies that at state s', place p has at least one token. Hence the assumption was wrong, and instead, b is enabled in s'. By persistency, since b does not occur in τ', also $s'[b\tau'\rangle$, and hence $s[\tau b \tau'\rangle$, as claimed. □ 5

4 Solving an lts, Using rg, r, p, and PΥ

Throughout this section, we continue to assume that some given, finite lts $TS = (S, \rightarrow, T, s_0)$ satisfies all properties **rg**, **r**, **p**, and **PΥ**. Our aim will be to derive conditions under which an ON Petri net solution exists for TS. In section 4.1, we will identify important subsets of states. Using these sets, section 4.2

presents an algorithm which is able to produce an ON Petri net from TS, under certain conditions specifying exactly when this is possible. Section 4.3 contains the correctness proof of this algorithm. Finally, section 4.4 discusses how it may be checked that *no* ON Petri net can be constructed for TS.

The lts shown in Figures 1 and 2 will serve as motivating and as running examples. Note that both of them satisfy all required properties (one with **P**1, the other with **P**2), but for the first one, an ON solution exists while for the second one, *no* ON solution exists (although we still did not prove this). Henceforth, examples of the first type will be called *positive* while examples of the second kind will be called *negative*.

4.1 Sequentialising States

In Figure 1, state M does not enable b, but all of its successor states do, no matter whether they are reached by a or by d. We might say that state M *sequentialises* the set of labels $\{a, d\}$ with regard to b. The ON solution shown on the right-hand side of Figure 1 contains a place, called p, with ingoing transitions a, d (each with weight 1) and a single outgoing transition b (with weight 2). We might interpret this place as *realising* the sequentialisation of $\{a, d\}$ with regard to b.

The basic idea, to be developed in the following, is to generalise this observation: If all sequentialising states are enumerated and adequate corresponding places are introduced, does there result an ON net solving the original lts? In section 4.2, it will be shown that, upon closer inspection of this idea, the following definition plays a crucial role.

Definition 7. UNIQUE INPUT STATES AND SEQUENTIALISATION STATES
For any label $x \in T$, we shall denote by

$$\boxed{\begin{array}{ll} \mathit{NUI}(x) & = \{s \in S \mid \neg s[x\rangle \wedge {}^\bullet s = \{x\}\} \\ \mathit{Seq}(x) & = \{s \in S \mid \neg s[x\rangle \wedge \forall t \in s^\bullet \colon s[tx\rangle\} \end{array}}$$

the set of states with unique input x not enabling x, and the set of states from which x is sequentialised, respectively. □ 7

Example: In Figure 1, $\mathit{NUI}(b) = \{M_1, M_2\}$ and $\mathit{Seq}(b) = \{M\}$.
In Figure 2, $\mathit{NUI}(a) = \{2\}$ and $\mathit{Seq}(a) = \{5\}$. *End of example*

4.2 Checking ON-solvability

Assuming that $T = \{x, a_1, a_2, \ldots, a_m\}$ is the set of labels of TS, we now wish to determine under which conditions a pure place p with outgoing transition x and incoming transitions $a_1, a_2, \ldots, a_m$ can serve as a part of an ON Petri net solving TS. The general form of such a place is shown in Figure 3. For generality reasons, it is assumed that all transitions a_j but the outgoing one, x, are inputs of such a place, even if some of the weights k_j can possibly turn out to be zero.

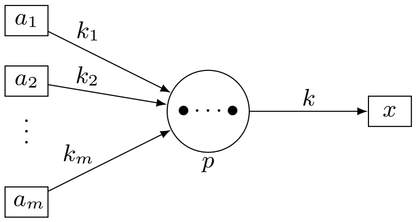

Fig. 3. A general pure ON place p

The parameters to be determined are the arc weights $k_1, k_2, \ldots, k_m, k$, all ≥ 0, and the initial marking $M_0(p)$ corresponding to the initial state s_0 of TS.

First, we may observe that we do not change the dynamics of the place if we multiply all the weights by some integer $n > 0$ and the initial marking by the same factor; we may even replace the initial marking by any value $n \cdot M_0(p) + \ell$ provided $0 \leq \ell < n$. Conversely, if all the weights have a common factor $n > 0$, the dynamics of the place is not modified if we divide the weights by n and replace the initial marking by $M_0(p) \div n$. Hence, we may always assume

$$\gcd\{k, k_1, k_2, \ldots, k_m\} \;=\; 1 \tag{1}$$

Next, since all cycles have a Parikh vector multiple of Υ, we must have

$$\sum_{1 \leq j \leq m} k_j \cdot \Upsilon(a_j) \;=\; k \cdot \Upsilon(x) \tag{2}$$

Example: The lts in Figure 1 satisfies **P**1. For place p, (2) becomes: sum of the weights of p's incoming arcs = weight of p's outgoing arc = 2. A similar equality is true for all other places in Figure 1. *End of example*

Now, let us determine the constraints the initial marking of this place must satisfy. Since it is necessary that each path allowed by the lts is also allowed by the place, the initial marking must be large enough. By the shape of the place shown in Figure 3 and by the firing rule, the marking of place p at an arbitrary state r is $M_r(p) = M_0(p) + \sum_j k_j \cdot \Delta_{s_0,r}(a_j) - k \cdot \Delta_{s_0,r}(x)$. This sum must always be nonnegative, that is, we must have

$$\forall r \in S: \quad M_0(p) \;\geq\; k \cdot \Delta_{s_0,r}(x) \;-\; \sum_{1 \leq j \leq m} k_j \cdot \Delta_{s_0,r}(a_j) \tag{3}$$

In addition, since there is no side-condition around p, if these inequalities are satisfied for all states r, no path of the lts will be prevented by place p.

Another way to interpret the constraints (2) and (3) is to see that these are exactly the conditions to be satisfied such that the triple $(\mathbb{R}, \mathbb{B}, \mathbb{F})$ where $\mathbb{R}(r) = M_0(p) + \sum_j k_j \cdot \Delta_{s_0,r}(a_j) - k \cdot \Delta_{s_0,r}(x)$, $\mathbb{B}(x) = k$ (0 otherwise), and $\mathbb{F}(a_j) = k_j$ (for $j = 1, ..., m$; 0 otherwise) is a region in the sense of Definition 5.

The constraints (3) yield $|S|$ inequalities, but they are not all useful. If $r[x\rangle r'$, the constraint for r' entails the one for r since $M_{r'}(p) = M_r(p) - k < M_r(p)$.

Similarly, if $r[a_j\rangle r'$, the constraint for r entails the one for r' since $M_{r'}(p) = M_r(p) + k_j \geq M_r(p)$. As a consequence, the only interesting states for inequalities (3) are the states $r \in NUI(x)$, since the other ones correspond to higher markings, so that the positivity of the marking (or region) is ensured for the latter if it is for the states in $NUI(x)$.

The values for the initial marking which satisfy inequalities (3) are upward closed, and for any choice of the weights satisfying the equations (2) and (1), it is always possible to choose $M_0(p) = \max_{r \in NUI(x)}(k \cdot \Delta_{s_0,r}(x) - \sum_j k_j \cdot \Delta_{s_0,r}(a_j))$. This is the least possible value for $M_0(p)$ and, in that case, for at least one of the states $r \in NUI(x)$, the marking $M_r(p)$ is 0 (otherwise a lower value for $M_0(p)$ could have been chosen).

Example: For the lts in Figure 1, $\max_{r \in \{M_1,M_2\}}(2 \cdot \Delta_{M_0,r}(b) - (\Delta_{M_0,r}(a) + \Delta_{M_0,r}(d))) = 2$, the number of tokens initially on place p. *End of example*

Next, let s be any state of the lts which does not enable x. In a pure ON solution of TS, and corresponding to ESSP, there must be at least one place p_s of the kind shown in Figure 3 that forbids this transition. Hence we must have:

$$M_0(p_s) \;<\; k \cdot (\Delta_{s_0,s}(x) + 1) \;-\; \sum_{1 \leq j \leq m} k_j \cdot \Delta_{s_0,s}(a_i) \tag{4}$$

because otherwise, due to the "+1", the marking $M_s(p_s)$ does not prevent x from occurring at state s. Note that we do not forbid that $p_s = p_{s'}$, if both s and s' exclude an x-move and the same place works for both. It could even happen that a single place works for all the exclusions of x.

Again, some of those constraints (hence some places) entail other ones. For instance, if $s[a_j\rangle s'$, while $\neg s[x\rangle$ and $\neg s'[x\rangle$, the place $p_{s'}$ also does the job for s, since its marking at state s is not higher than at state s', so that $p_{s'}$ excludes x from s if it does so from s'. As a consequence, we only have to consider the inequalities (4) for states $s \in \mathit{Seq}(x)$. For this reason, in view of (4), only the sequentialisation states for x are interesting.

The constraints (3) and (4) both concern the initial marking, but it is possible to express them without it, since the system (3) yields the minimal initial marking. One has to find, for each transition x and each state $s \in \mathit{Seq}(x)$, arc weights such that

$$\forall r \in NUI(x)\colon\; k \cdot \Delta_{s_0,r}(x) - \sum_j k_j \cdot \Delta_{s_0,r}(a_j) < k \cdot (\Delta_{s_0,s}(x) + 1) - \sum_j k_j \cdot \Delta_{s_0,s}(a_j)$$

or equivalently

$$\forall r \in NUI(x)\colon\; 0 \;<\; k \cdot (\Delta_{r,s}(x) + 1) - \sum_j k_j \cdot \Delta_{r,s}(a_j) \tag{5}$$

since, using Lemma 3, $\Delta_{r,s} = \Delta_{s_0,s} - \Delta_{s_0,r} + m \cdot \Upsilon$ for some integer m, and the coefficients satisfy equation (2). With the aid of equation (2), we may even eliminate k from these inequalities. As a result, for each state $s \in \mathit{Seq}(x)$ we obtain the system of inequations

$$\boxed{\forall r \in NUI(x)\colon\; 0 \;<\; \textstyle\sum_{1 \leq j \leq m} k_j \cdot (\Upsilon(a_j) \cdot (1 + \Delta_{r,s}(x)) - \Upsilon(x) \cdot \Delta_{r,s}(a_j))} \tag{6}$$

The reasoning above shows that, if the considered lts is solvable by a pure ON net, then for each label $x \in T$ and each state $s \in Seq(x)$ the system of inequations (6) is solvable in the $\mathbb{N}$ domain. A converse is also true. Assume that, for each label $x \in T$ and each state $s \in Seq(x)$, the system of inequations (6) is solvable. So, we get nonnegative integer values for all the weights k_j. It may still happen that they do not lead to an integer value for k satisfying equation (2). But since the inequations are homogeneous, any non-null set of integers proportional to a solution is also a solution (while there may be other, non-proportional, solutions as well). Hence it is always possible to choose a solution which may be extended with an integer k such that equation (2), as well as (1), is satisfied, and choose the initial marking $\max_{r \in NUI(x)}\{\sum_{1 \le j \le m} k_j \cdot (\frac{\Upsilon(a_j)}{\Upsilon(x)} \cdot \Delta_{s_0,r}(x) - \Delta_{s_0,r}(a_j))\}$.

Then, from (6) and (2), the constraints (3) and (4) are satisfied, and a pure ON place with integer weights on all of its adjacent arcs may be constructed.

First example: Consider label $x = b$ and state $s = M$ in the lts depicted in Figure 1. For states $r = M_1 \in NUI(b)$ and $r = M_2 \in NUI(b)$, inequations (6) respectively reduce to $0 < k_a$ and $0 < k_d$ after setting the Υ terms to 1 and evaluating the Δ terms on the lts. This system is simultaneously solvable by $k_a = 1$, $k_c = 0$ and $k_d = 1$, describing the interface at place p with regard to its incoming transitions. Using (2), the weight k of the arc from p to its outgoing transition b can then be set to $k = k_a + k_d = 2$. Observe that apart from the minimal solution given in this instance, there are plenty of other solutions arising, for instance, by uniform multiplication by a constant number. All of them serve the same purpose, and redundant ones can be omitted. *End of first example*

It can be verified that the system (6) of inequations can always be solved for this example, not just for $x = b$ and $s = M$. Moreover, the ON solution shown on the right-hand side of Figure 1 can be obtained by assembling places yielded by such solutions. In the next section, it will be shown that this is a general property of the construction defined in the present section.

Second example: Consider label $x = a$ and state $s = 5$ in the lts depicted in Figure 2. For the only state $r = 2 \in NUI(a)$, inequations (6) become $0 < k_b \cdot (-2)$, which is not solvable in the $\mathbb{N}$ domain. *End of second example*

This second example illustrates the opposite case. As we have just seen, there exists some label x and some state $s \in Seq(x)$ such that (6) is unsolvable. In the next section, it will be shown that this entails, in general, that no ON Petri net solving the given lts exists.

Note: The constructions also work in case $|T| = |\{x\}| = 1$ (then $Seq(x) = \emptyset$, and a single, isolated transition x is created) as well as in case $T = \emptyset$ (then an empty net with empty initial marking is created). *End of note*

Figure 4 summarises the resulting algorithm.

4.3 Correctness and Optimisations

In this section, the shorthand "(6) is solvable" means that for all pairs $x \in T$ and $s \in Seq(x)$, the system of inequations (6) is solvable in $\mathbb{N}$.

Theorem 4. Language equivalence

```
**input** an lts $TS = (S, \to, T, s_0)$ and a $T$-vector $\Upsilon \geq 1$ satisfying **rg**, **r**, **p**, **P**$\Upsilon$;
**initially** $T$ is the set of transitions, and $P := \emptyset$;
**for** every $x \in T$ and $s \in Seq(x)$ **do**
  construct the system (6) for $x$ and $s$, as well as equations (1) and (2) for $x$;
  **if** $\neg \exists k_1, \ldots, k_m \in \mathbb{N}$ solving (6) **then**
    {**output** "$TS$ not ON-solvable, due to $x$, $s$ and system (6)"; **stop**};
  choose a set of integers $(k_1, \ldots, k_m, k)$ satisfying (6), (1) and (2);
  add to $P$ a place as in Fig. 3, with weights $k_1, \ldots, k_m, k$ and initial marking
      $\max_{r \in NUI(x)}\{\sum_{1 \leq j \leq m} k_j \cdot (\frac{\Upsilon(a_j)}{\Upsilon(x)} \cdot \Delta_{s_0,r}(x) - \Delta_{s_0,r}(a_j))\}$
**end for**; **output** "The net with transitions $T$ and places $P$ ON-solves $TS$".
```

Fig. 4. An algorithm checking ON-solvability and constructing an adequate solution

A finite lts TS satisfying properties **rg**, **r**, **p** *and* **P**Υ *has the same language as some pure ON net if and only if* (6) *is solvable.*

Proof

($\Leftarrow$): Suppose that (6) can be solved. Then the construction exhibited in the previous section yields a pure ON Petri net N, whose set of places correspond to regions satisfying ESSP. As noted after Theorem 1, ESSP entails language-equivalence between TS and N.

($\Rightarrow$): Suppose that (6) cannot be solved, for some $x \in T$ and $s \in Seq(x)$. It was shown in the previous section that it is impossible to separate x at s by any pure ON place. Thus there is no pure ON net with the same language as TS (nor solving TS). □ 4

It is possible to strengthen language-equivalence to isomorphism, as follows.

Theorem 5. Reachability graph isomorphism

A finite lts satisfying properties **rg**, **r**, **p** *and* **P**Υ *is isomorphic to the reachability graph of some pure ON net if and only if* (6) *is solvable.*

Proof: ($\Rightarrow$) follows from Theorem 4($\Rightarrow$).

($\Leftarrow$): Let N be a pure ON net as constructed by the algorithm in the previous section. From Theorem 4($\Leftarrow$), one only has to check also that no two states s_1 and s_2 of the lts correspond to the same marking of N. If it would be the case, since the lts is strongly connected, there would be a sequence σ of transitions leading from s_1 to s_2. Since both places correspond to the same marking of the net, the sequence σ should also be allowed iteratively from s_2, leading to states s_3, s_4, ..., and since the system is finite, at some point we will have $s_i = s_j$ with $i < j$. Hence, from s_i we have a cycle with Parikh vector $(j-i)\cdot\Psi(\sigma)$, and by **rg** (as successor markings depend only on Parikh vectors of paths), σ generates a cycle from s_1 and $s_2 = s_1$. Consequently, SSP holds and, by Theorem 1, the reachability graph of the constructed net is isomorphic to the lts we started from. □ 5

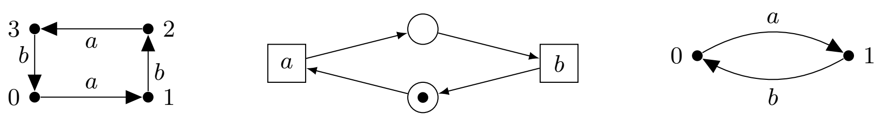

Fig. 5. An lts, the constructed Petri net, and its reachability graph

The significance of **rg** in part ($\Leftarrow$) of Theorem 5 is illustrated by the example shown in Figure 5. The lts on the left-hand side (which does not satisfy **rg**) leads to solvable systems (6). A corresponding net is shown in the middle of the figure, and its reachability graph on the right. In this case, the places of the constructed net yield regions of the given lts satisfying ESSP, but not SSP. The culprit is the fact that, in the lts we start from, $0[abab\rangle 0$ form a cycle, with $\Psi(abab) = \Upsilon = (2,2)$, but $0[ab\rangle 2$ is not a cycle while $\Psi(ab) = (1,1) = \Upsilon/2$.

Some remarks: The previous theorems hold even if "pure ON" is replaced by "ON" in their statements. That is, allowing side-conditions does not afford any true new degree of freedom.

If one is interested in synthesis problems, instead of reengineering ones, it is possible to weaken **rg**. For instance, it is possible to show that if **rg** is replaced by "the lts is finite, deterministic, and totally reachable" plus "$\gcd_{t\in T}\{\Upsilon(t)\} = 1$", Theorem 5 is still valid. Theorem 5 is also still valid if **rg** is replaced by "the lts is finite, deterministic, and totally reachable" plus a marginally stronger form of cycle-consistency [4]. One only has to make sure that the proof of 5($\Leftarrow$) goes through.

For these remarks, the details are described in [6]. *End of some remarks*

The aim of the algorithm exhibited in Figure 4 is correctness, not minimality. Various optimisations may be considered, such as:

- when a new place is constructed, for some $x \in T$ and $s \in Seq(x)$, it may happen that the arc weights around a place constructed for a previously considered $s' \in Seq(x)$ also satisfy the current system (6), as well as (1) and (2) since the latter have not changed; hence the new place the algorithm would construct is redundant and can be dropped;
- conversely, when a new place is constructed for some $x \in T$ and $s \in Seq(x)$, it may happen that its arcs weights also satisfy one or more systems (6) constructed for previous states $s' \in Seq(x)$, so that the places constructed from the latter are redundant and may be dropped;
- even if the algorithm does not produce redundant places, it can happen that a set of natural numbers $k_1, \ldots, k_m$ solves simultaneously many systems (6), for some x and many states s, leading to a place allowing to drop many redundant places; this kind of optimisation may be detected by searching for maximal subsets $S \subseteq Seq(x)$ such that the system $\cup_{s\in S}(6)_s$ is solvable, where $(6)_s$ denotes the system (6) constructed for x and s; then one only has to consider the subset $Seq(x) \setminus S$ for continuing the construction of places with output x;

- from the way we constructed a place from a solution of (6), or $\cup_{s\in S}(6)_s$, if $\Upsilon(x) > 1$, it may happen that a "better" place may be found when starting from a greater solution of the considered system.

4.4 Checking Non-ON-Solvability

Using the previous results, checking ON-solvability amounts to checking the *solvability* of the system of inequations (6) for all $x \in T$ and $s \in Seq(x)$, while checking non-ON-solvability amounts to checking the *unsolvability* of (6) for one such label x and state s. By means of linear-algebraic duality and by considering a dual system of inequalities, it is possible to exchange these two methods:

Theorem 6. ON INCOMPATIBILITY
A finite lts satisfying properties **rg**, **r**, **p** *and* **P**Υ *is* not *language-equivalent to the reachability graph of some (pure) ON Petri net if and only if, for some label* $x \in T$ *and sequentialising state* $s \in Seq(x)$, *the system of constraints in* y_r*'s*

$$\forall j, 1 \le j \le m:\ 0\ \ge \sum_{r\in NUI(x)} y_r \cdot (\,\Upsilon(a_j)\cdot(1+\Delta_{r,s}(x)) - \Upsilon(x)\cdot\Delta_{r,s}(a_j)\,) \quad (7)$$

has a nonnull solution in $\mathbb{N}$.

Proof: This is an immediate consequence of Theorem 4 and the alternation result of Ville [17]. Among several similar results (e.g., Farkas's lemma), Ville's theorem has a convenient formulation to imply directly that the system of inequations (6) has a solution in $\mathbb{N}$ if and only if the system (7) has no other solution in $\mathbb{N}$ than $y_r = 0$ for each r. □ 6

Specialising Theorem 6($\Leftarrow$) by considering solution vectors with only 0 and 1 for the unknowns, we obtain the following:

Corollary 1. BAD CONFIGURATIONS FOR ON NETS
If, for a finite lts TS *satisfying properties* **rg**, **r**, **p** *and* **P**Υ, *there exist a label* $x \in T$, *a state* $s \in Seq(x)$, *and a subset* $\emptyset \ne R \subseteq NUI(x)$ *such that*

$$\forall j, 1 \le j \le m:\ 0\ \ge \sum_{r\in R}(\,\Upsilon(a_j)\cdot(1+\Delta_{r,s}(x)) - \Upsilon(x)\cdot\Delta_{r,s}(a_j)\,) \quad (8)$$

then no ON net is language-equivalent (nor, a fortiori, *reachability graph-isomorphic) to* TS. □ 1

Example: In Figure 2, there is a bad configuration corresponding to label $x = a$, state $s = 5$, and state set $R = \{2\}$. *End of example*

Even though this corollary constitutes only a very special case, we mention it explicitly, because for all the examples we have considered up to now, checking (8) was sufficient to show non-ON-solvability, and we conjecture that this is a general property.

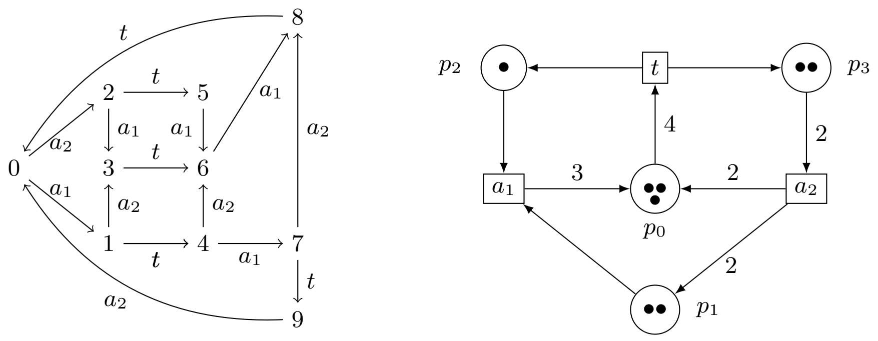

Fig. 6. An lts with unique Parikh cycle and a possible ON Petri net solution

5 Examples of the Constructions

We shall now illustrate the constructions (and optimisations) developed in section 4 on two rather more substantial examples.

5.1 A Worked, Positive Example

Let us consider the lts in Figure 6. It satisfies all the requested preconditions, with $\Upsilon = (a1 \mapsto 2, a2 \mapsto 1, t \mapsto 2)$. We get $NUI(a1) = \{1,7\}$, $Seq(a1) = \{3,8,9\}$, $NUI(a2) = \{2\}$, $Seq(a2) = \{8\}$, $NUI(t) = \{4,5,9\}$ and $Seq(t) = \{0,6\}$.

Let us first consider the sequentialisation of t at state 0 through $a1$ and $a2$, which should correspond to the place p_0 in the ON net on the right.
The system (6) has the form

$$\begin{aligned} &0 < 2 \cdot k_{a1} \\ &0 < 2 \cdot k_{a2} \\ &0 < 2 \cdot k_{a1} - k_{a2} \end{aligned}$$

the least solution is $k_{a1} = 1, k_{a2} = 1$, but this does not lead to an integer value for k_t; but multiplying it by 2 leads to a valid solution $k_{a1} = 2, k_{a2} = 2, k_t = 3$, with the initial marking 2.

For the sequentialisation of t at state 6 through a_1 we get the system:

$$\begin{aligned} &0 < 2 \cdot k_{a1} - k_{a2} \\ &0 < k_{a2} \\ &0 < 2 \cdot k_{a1} - 2 \cdot k_{a2} \end{aligned}$$

again, the least solution ($k_{a1} = 2, k_{a2} = 1$) is not adequate; twice it is adequate: $k_{a1} = 4, k_{a2} = 2, k_t = 5$, but there is a smaller one: $k_{a1} = 3, k_{a2} = 2, k_t = 4$, with the initial marking 3, corresponding to place p_0 in the Petri net. This shows that the least solution of the system (6) does not always lead to the smallest solution of (5) (but it leads nevertheless to an acceptable solution). Moreover, it may be observed that this place also works for the previous sequentialisation, while not the least one in this case, so that it may be useful not to stick to the smallest solution in terms of weights: a non-minimal one may work for many different systems.

There are two different sequentialisation configurations for a_1 through t: from state 3 and state 8. The first one leads to the system

$$\begin{aligned} 0 &< -k_{a2} + 2 \cdot k_t \\ 0 &< -2 \cdot k_{a2} + 2 \cdot k_t \end{aligned}$$

with minimal solution $k_{a1} = 1, k_{a2} = 0, k_t = 1$, with the initial marking 1, corresponding to place p_2 in the Petri net.
The second one leads to a different system

$$\begin{aligned} 0 &< 2 \cdot k_t \\ 0 &< -k_{a2} + 2 \cdot k_t \end{aligned}$$

but with the same minimal solution, and place.

However, there is another sequentialisation for a_1, from state 9 through a_2. Here the system to be solved is

$$\begin{aligned} 0 &< 2 \cdot k_{a2} \\ 0 &< k_{a2} \end{aligned}$$

and the minimal solution is $k_{a1} = 1, k_{a2} = 2, k_t = 0$, with the initial marking 2, corresponding to place p_1 in the net.

Finally, the sequentialisation of a_2 from state 8 through t yields the (very simple) system

$$0 < k_t$$

with minimal solution $k_{a1} = 0, k_{a2} = 2, k_t = 1$, with the initial marking 2, corresponding to place p_3 in the net.

5.2 A Worked, Negative Example

In this section, the following assertion will be proved:

Proposition 1. A VERY REGULAR, PERSISTENT LTS WITHOUT ON SOLUTION
There exists a finite lts, satisfying properties **r**, **p**, **P**1, *and* **rg** *with a plain and pure solution, which cannot be solved by any ON Petri net.*

This proposition implies, amongst other things, that the class of persistent (plain, pure) Petri nets is "essentially larger" than the class of ON Petri nets, even if for the latter, arbitrary arc weights and side-conditions are allowed. Note that the latter class is, in turn, much larger than the class of marked graphs, as exemplified by Figure 1.

Proof: *By example.* The lts presented in Figure 7, where a triple (s, t, s') represents an arc $s[t\rangle s'$ and $s0$ is the initial state, has 10 labels, 89 states and 180 arcs. We have verified, with `synet` [8] and our own tools [9,15], that it satisfies **r**, **p**, **P**1 and a strong form of **rg** since it is generated by a plain and pure net. However it has no ON solution. The culprit – by construction of the lts, and also found by [9] – is label $y1$ with sequentialising state $s6 \in \mathit{Seq}(y1)$ and states $\mathit{NUI}(y1) = \{s58, s83\}$. The system (6) yields the constraints

$$\begin{array}{rcll} 0 & < & -2 \cdot k_{b1} + k_{a2} - k_{b2} - k_{b3} & (\text{ from a short path } s58 \rightsquigarrow s6 \text{ }) \\ 0 & < & -2 \cdot k_{a1} - k_{a2} + k_{b2} - k_{a3} & (\text{ from a short path } s83 \rightsquigarrow s6 \text{ }) \end{array}$$

whose sum $0 < -2 \cdot k_{b1} - k_{b3} - 2 \cdot k_{a1} - k_{a3}$ has no solution in $\mathbb{N}$. In fact, this corresponds to a bad configuration in the sense of Corollary 1, with $x = y1$, $s = s6$, and $R = \{s58, s83\}$. □ 1

(s0,a1,s1) (s0,b1,s2) (s0,z,s3) (s1,b1,s25) (s1,z,s4) (s2,a1,s25) (s2,z,s5) (s3,a1,s4)
(s3,b1,s5) (s4,a2,s31) (s4,b1,s6) (s5,a1,s6) (s5,b2,s7) (s6,a2,s8) (s6,b2,s9) (s7,a1,s9)
(s7,y1,s26) (s8,b2,s10) (s8,y1,s11) (s9,a2,s10) (s9,y1,s87) (s10,y1,s12) (s11,b2,s12)
(s11,y2,s33) (s12,y2,s13) (s13,a3,s14) (s13,b3,s15) (s13,x,s16) (s14,b3,s17) (s14,x,s21)
(s15,a3,s17) (s15,x,s18) (s16,a3,s21) (s16,b3,s18) (s17,x,s0) (s18,a3,s0) (s18,b1,s19)
(s18,z,s20) (s19,a3,s2) (s19,z,s22) (s20,a3,s3) (s20,b1,s22) (s21,a1,s27) (s21,b3,s0)
(s21,z,s28) (s22,a3,s5) (s22,b2,s23) (s23,a3,s7) (s23,y1,s24) (s24,a3,s26) (s24,y2,s54)
(s25,z,s6) (s26,a1,s87) (s26,y2,s55) (s27,b3,s1) (s27,z,s29) (s28,a1,s29) (s28,b3,s3)
(s29,a2,s30) (s29,b3,s4) (s30,b3,s31) (s30,y1,s32) (s31,b1,s8) (s31,y1,s59) (s32,b3,s59)
(s32,y2,s44) (s33,a3,s34) (s33,b2,s13) (s33,x,s35) (s34,b2,s14) (s34,x,s36) (s35,a3,s36)
(s35,b2,s16) (s36,a1,s37) (s36,b2,s21) (s36,z,s38) (s37,b2,s27) (s37,z,s39) (s38,a1,s39)
(s38,b2,s28) (s39,a2,s40) (s39,b2,s29) (s40,b2,s30) (s40,y1,s41) (s41,b2,s32) (s41,y2,s42)
(s42,a3,s43) (s42,b2,s44) (s43,b2,s45) (s43,x,s49) (s44,a3,s45) (s44,b3,s46) (s45,b3,s47)
(s45,x,s48) (s46,a3,s47) (s46,b1,s33) (s46,x,s62) (s47,b1,s34) (s47,x,s50) (s48,b3,s50)
(s49,b2,s48) (s50,a1,s51) (s50,b1,s36) (s50,z,s52) (s51,b1,s37) (s51,z,s53) (s52,a1,s53)
(s52,b1,s38) (s53,a2,s57) (s53,b1,s39) (s54,a3,s55) (s54,b3,s56) (s55,a1,s78) (s55,b3,s63)
(s55,x,s79) (s56,a3,s63) (s56,x,s64) (s57,b1,s40) (s57,y1,s58) (s58,b1,s41) (s58,y2,s60)
(s59,b1,s11) (s59,y2,s46) (s60,a3,s61) (s60,b1,s42) (s61,b1,s43) (s61,x,s88) (s62,a3,s50)
(s62,b1,s35) (s63,a1,s65) (s63,x,s66) (s64,a3,s66) (s65,a2,s15) (s65,x,s67) (s66,a1,s67)
(s66,b1,s68) (s66,z,s69) (s67,a2,s18) (s67,b1,s70) (s67,z,s74) (s68,a1,s70) (s68,z,s71)
(s69,a1,s74) (s69,b1,s71) (s70,a2,s19) (s70,z,s72) (s71,a1,s72) (s71,b2,s82) (s72,a2,s22)
(s72,b2,s73) (s73,a2,s23) (s73,y1,s75) (s74,a2,s20) (s74,b1,s72) (s75,a2,s24) (s75,y2,s76)
(s76,a2,s54) (s76,b3,s77) (s77,a2,s56) (s77,x,s81) (s78,a2,s13) (s78,b3,s65) (s78,x,s80)
(s79,a1,s80) (s79,b3,s66) (s80,a2,s16) (s80,b3,s67) (s81,a2,s64) (s82,a1,s73) (s82,y1,s83)
(s83,a1,s75) (s83,y2,s84) (s84,a1,s76) (s84,b3,s85) (s85,a1,s77) (s85,x,s86) (s86,a1,s81)
(s87,a2,s12) (s87,y2,s78) (s88,b1,s49)

Fig. 7. An lts satisfying **rg** with a plain and pure net, **r**, **p**, and **P**1, without an ON solution

So far, we did not check whether the example shown in Figure 7 has a minimal number of states. Nevertheless, the following two arguments lead us to believe that the number of states cannot be reduced considerably.

- We found an example satisfying **rg**, **r**, **p**, **P**1, that has no plain or pure solution, cannot be reduced, and already has 23 states [6].

- This kind of counterexample is derived rather painstakingly from a skeleton, akin to a bad configuration, which ensures the unsolvability of (6) *in spite of* **rg**. Such a skeleton already has quite a few states, and Lemma 5 (in conjunction with persistency) adds more states to it. It is hard to see that considerably fewer states would suffice.

6 Concluding Remarks

In this paper, we have developed a synthesis / reengineering algorithm for lts satisfying several nice properties, which allows to create an ON Petri net whenever it is theoretically possible. Moreover, we have shown that the existence of structurally pleasant solutions cannot always be guaranteed. In parallel work [7], we were able to show, however, that the state spaces of live and bounded marked graphs (and their bounds) can actually be characterised by adding a single further property, namely **bp**, an analogue of persistency in backward direction.

Several questions remain open, and new ones have been detected.

Generalisations and extensions. Very simple examples show that persistency cannot be dropped (quite naturally not, since the ON property is intimately related to persistency). But what happens exactly if reversibility is weakened to liveness? What happens for unbounded nets? Can concurrency semantics, especially step semantics, play a useful role?

Restrictions. Can one find a weak (preferrably, structural) property, to be imposed besides **rg**, **r**, **p**, and $\mathbf{P}\Upsilon$ or **P**1, which is weak enough not to imply **bp**, but also strong enough to guarantee ON solvability? For example, it is not known whether Proposition 1 in section 5.2 remains true if the net which exists by **rg** is assumed, in addition, to be 1-bounded.

Simplifications. Can the crucial system of constraints, (6), be reduced further by considering only those $r \in NUI(x)$ from which x-free directed paths lead to s? Can the number of states be reduced in the example discussed in section 5.2?

Special cases. Can one find exact criteria for cyclic lts (generalising Figures 2 and 5), or for lts of other regular shapes?

Complexity analysis. As compared, for instance, with [2], our constructions are more efficient for two reasons. The first reason is that, as a synthesis method, the algorithm given in section 4.3 applies to a special case. This is reflected by the fact that we identify the two sets $NUI(x) \subseteq S$ and $Seq(x) \subseteq S$ which are normally much smaller than S, leading to much fewer linear-algebraic calculations needing to be done. However, it is still a matter of research to find out exactly how the sizes of these sets are related to each other in the average or worst cases. The second reason is that our algorithm comes with an in-built reengineering method, which is absent in the general synthesis algorithm. If one wanted to use the latter for reengineering in a brute-force way, one would have to cycle through all region bases in order to check whether there exists a “nice”, desirable one. Such an additional loop implies a considerable additional layer of complexity to general synthesis. It is absent in our approach because it was specifically tailored to the desired “niceness” criterion, viz. ON solvability.

Acknowledgments. We are grateful to Philippe Darondeau and Hanna Klaudel for inspirations and discussions. The remarks of the anonymous reviewers allowed to improve the presentation of the paper.

References

1. Badouel, É., Bernardinello, L., Darondeau, P.: Petri Net Synthesis, 330 pages. Springer (in preparation, 2014)
2. Badouel, É., Bernardinello, L., Darondeau, P.: Polynomial Algorithms for the Synthesis of Bounded Nets. In: Mosses, P.D., Nielsen, M., Schwartzbach, M.I. (eds.) TAPSOFT 1995. LNCS, vol. 915, pp. 364–378. Springer, Heidelberg (1995)
3. Badouel, É.: Theory of Regions. In: Reisig, W., Rozenberg, G. (eds.) APN 1998. LNCS, vol. 1491, pp. 529–586. Springer, Heidelberg (1998)
4. Best, E., Darondeau, P.: A Decomposition Theorem for Finite Persistent Transition Systems. Acta Informatica 46, 237–254 (2009)
5. 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)
6. Best, E., Devillers, R.: Solving LTS with Parikh-unique Cycles. TR 2/14, Dep. Informatik, Carl von Ossietzky Universität Oldenburg, 80 pages (February 2014)
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, Heidelberg (2014)
8. Caillaud, B.: `http://www.irisa.fr/s4/tools/synet/`
9. Devillers, R.: `plain.c`, `pure.c`, `frag.c`: Specially tailored programs written in `C++`
10. Hack, M.: Analysis of production schemata by Petri nets, M.S. thesis, D.E.E. MIT. Cambridge Mass. Project MAC-TR 94 (1972)
11. Keller, R.M.: A Fundamental Theorem of Asynchronous Parallel Computation. In: Tse-Yun, F. (ed.) Parallel Processing. LNCS, vol. 24, pp. 102–112. Springer, Heidelberg (1975)
12. Kondratyev, A., Cortadella, J., Kishinevsky, M., Pastor, E., Roig, O., Yakovlev, A.: Checking Signal Transition Graph Implementability by Symbolic BDD Traversal. In: Proc. European Design and Test Conference, Paris, France, pp. 325–332 (1995)
13. Lamport, L.: Arbiter-Free Synchronization. Distributed Computing 16(2/3), 219–237 (2003)
14. Landweber, L.H., Robertson, E.L.: Properties of Conflict-Free and Persistent Petri Nets. J. ACM 25(3), 352–364 (1978)
15. Schlachter, U., et al.: `https://github.com/renke/apt`
16. Teruel, E., Colom, J.M., Silva, M.: Choice-Free Petri nets: a model for deterministic concurrent systems with bulk services and arrivals. IEEE Transactions on Systems, Man and Cybernetics, Part A, 73–83 (1997)
17. Ville, J.: Sur la théorie générale des jeux où intervient l'habileté des joueurs. In: Borel, E. (ed.) Traité du calcul des probabilités et de ses applications, vol. 4, pp. 105–113. Gauthiers-Villars (1938)
18. Yakovlev, A.: Designing control logic for counterflow pipeline processor using Petri nets. Formal Methods in Systems Design 12(1), 39–71 (1998)
19. Yakovlev, A.: Theory and practice of using models of concurrency in hardware design. DSc Thesis, University of Newcastle upon Tyne (August 2005)