COMPUTING SCIENCE

Step Persistence in the Design of GALS Systems

Johnson Fernandes, Maciej Koutny, Marta Pietkiewicz-Koutny, Danil Sokolov and Alex Yakovlev

TECHNICAL REPORT SERIES

No. CS-TR-1349 August 2012

TECHNICAL REPORT SERIES

No. CS-TR-1349 August, 2012

Step Persistence in the Design of GALS Systems

J. Fernandes, M. Koutny, M. Pietkiewicz-Koutny, D. Sokolov and A. Yakovlev

Abstract

In this paper we investigate the behaviour of GALS (Globally Asynchronous Locally Synchronous) systems in the context of VLSI circuits. The specification of a system is given in the form of a Petri net. Our aim is to re-design the system to optimise signal management, by grouping together concurrent events. Looking at the concurrent reachability graph of the given Petri net, we are interested in discovering events that appear in "bundles", so that they all can be executed in one clock tick. The best candidates for bundles are sets of events that appear and re-appear over and over again in the same configurations, forming "persistent" sets of events. Persistence was considered so far only in the context of sequential semantics. Here we introduce a notion of persistent steps and discuss their basic properties. We then introduce a formal definition of a bundle and propose an algorithm to prune the behaviour of a system, so that only bundle steps remain. The pruned reachability graph represents the behaviour of a re-engineered system, which in turn can be implemented in a new Petri net using the standard techniques of net synthesis. The proposed algorithm prunes reachability graphs of persistent and safe nets leaving bundles that represent maximally concurrent steps.

Printed and published by Newcastle University,
Computing Science, Claremont Tower, Claremont Road,
Newcastle upon Tyne, NE1 7RU, England.

Bibliographical details

FERNANDES, J., KOUTNY, M., PIETKIEWICZ-KOUTNY, M., SOKOLOV, D., YAKOVLEV, A.

Step Persistence in the Design of GALS Systems
[By] J. Fernandes, M. Koutny, M. Pietkiewicz-Koutny, D. Sokolov, A. Yakovlev

Newcastle upon Tyne: Newcastle University: Computing Science, 2012.

(Newcastle University, Computing Science, Technical Report Series, No. CS-TR-1349)

Added entries

NEWCASTLE UNIVERSITY
Computing Science. Technical Report Series. CS-TR-1349

Abstract

In this paper we investigate the behaviour of GALS (Globally Asynchronous Locally Synchronous) systems in the context of VLSI circuits. The specification of a system is given in the form of a Petri net. Our aim is to re-design the system to optimise signal management, by grouping together concurrent events. Looking at the concurrent reachability graph of the given Petri net, we are interested in discovering events that appear in "bundles", so that they all can be executed in one clock tick. The best candidates for bundles are sets of events that appear and re-appear over and over again in the same configurations, forming "persistent" sets of events. Persistence was considered so far only in the context of sequential semantics. Here we introduce a notion of persistent steps and discuss their basic properties. We then introduce a formal definition of a bundle and propose an algorithm to prune the behaviour of a system, so that only bundle steps remain. The pruned reachability graph represents the behaviour of a re-engineered system, which in turn can be implemented in a new Petri net using the standard techniques of net synthesis. The proposed algorithm prunes reachability graphs of persistent and safe nets leaving bundles that represent maximally concurrent steps.

About the authors

Johnson Fernandes is a PhD student in the School of Electrical and Electronic Engineering, Newcastle University. His research interests include modelling, design automation and optimisation of mixed synchronous-asynchronous systems.

Maciej Koutny is a Professor in the School of Computing Science, Newcastle University. His research interests centre on the theory of distributed and concurrent systems, including both theoretical aspects of their semantics and application of formal techniques to the modelling and verification of such systems; in particular, model checking based on net unfoldings. He has also investigated non-interleaving semantics of priority systems, and the relationship between temporal logic and process algebras. Recently, he has been working on the development of a formal model combining Petri nets and process algebras as well as on Petri net based behavioural models of membrane systems.

Marta Pietkiewicz-Koutny is a Lecturer in the School of Computing Science, Newcastle University. Her areas of research interest include: modelling and validation of concurrent systems, synthesis of Petri nets from transition systems and abstraction and refinement in process networks.

Danil Sokolov is a Research Associate in the School of Electrical and Electronic Engineering, Newcastle University. His research interests are modelling of heterogeneous systems, synthesis methods for energy-efficient circuits, development of CAD tools and their integration into industry-level design flows.

Alex Yakovlev is a Professor of Computing Systems Design at the School of Electrical and Electronic Engineering. He has been with Newcastle University since 1991. His research interests and publications are in the field of modelling and design of asynchronous, concurrent, real-time, power-aware and dependable circuits and systems.

Suggested keywords

ASYNCHRONOUS AND SYNCHRONOUS CIRCUITS
GALS SYSTEMS
PERSISTENCE
STEP TRANSITION SYSTEMS
PETRI NETS

Step Persistence in the Design of GALS Systems

Johnson Fernandes, Maciej Koutny, Marta Pietkiewicz-Koutny,
Danil Sokolov and Alex Yakovlev

Newcastle University
Newcastle upon Tyne, NE1 7RU, U.K.

Abstract. In this paper we investigate the behaviour of GALS (Globally Asynchronous Locally Synchronous) systems in the context of VLSI circuits. The specification of a system is given in the form of a Petri net. Our aim is to re-design the system to optimise signal management, by grouping together concurrent events. Looking at the concurrent reachability graph of the given Petri net, we are interested in discovering events that appear in "bundles", so that they all can be executed in one clock tick. The best candidates for bundles are sets of events that appear and re-appear over and over again in the same configurations, forming "persistent" sets of events. Persistence was considered so far only in the context of sequential semantics. Here we introduce a notion of persistent steps and discuss their basic properties. We then introduce a formal definition of a bundle and propose an algorithm to prune the behaviour of a system, so that only bundle steps remain. The pruned reachability graph represents the behaviour of a re-engineered system, which in turn can be implemented in a new Petri net using the standard techniques of net synthesis. The proposed algorithm prunes reachability graphs of persistent and safe nets leaving bundles that represent maximally concurrent steps.
Keywords: asynchronous and synchronous circuits, GALS systems, persistence, step transition systems, Petri nets.

1 Introduction

Traditional circuit design styles have been following one of the two main strands, namely synchronous and asynchronous. In a nutshell, these two approaches differ in their techniques of synchronising interaction between circuit elements. Asynchronous designs adopt *'on request'* synchronisation where interaction is regulated by means of handshake control signals. They are designed to be adaptive to delays of signal propagation. Synchronous designs, on the other hand, assume worst case delay between circuit elements and determine a global periodic control signal for synchronisation called the *clock*. The clock signal limits the many sequencing options considered in asynchronous control. Thus synchronous circuits are considered to be a proper subset of asynchronous circuits [6].

Asynchronous logic was the dominant design style with most early computers. In particular, David Muller's speed-independent circuits, dating back to the late 1950s, have served many interesting applications such as the ILLIAC I and

ILLIAC II computers [15]. However since 1960, an era when fabrication of integrated circuits (ICs) became a feasible business, synchronous design became the mainstream technique as it met the market needs with its shorter design cycle. Today, majority of designs are synchronous, well etched in the heart of semiconductor industry together with superior CAD tools and EDA flows.

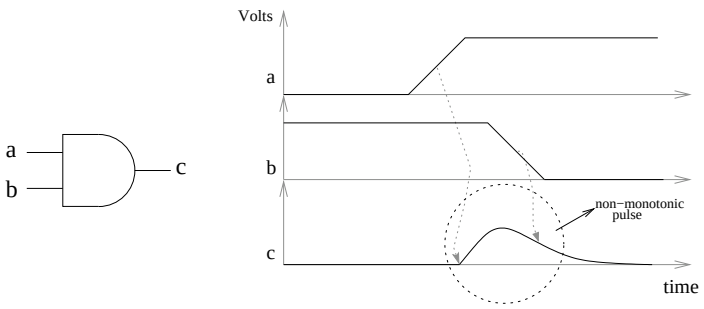


Fig. 1. Hazardous switching of an AND gate.

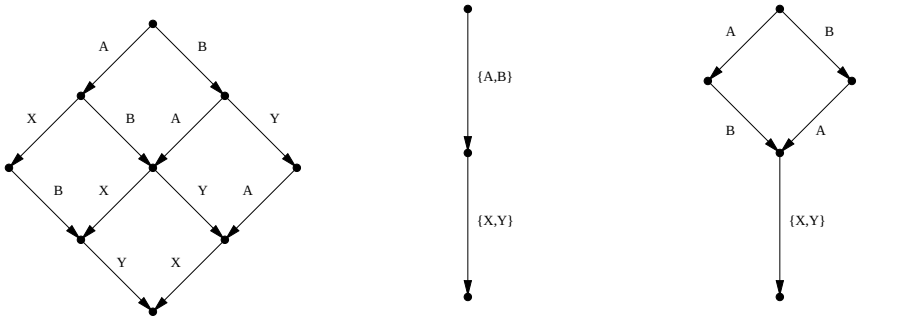
One of the main issues with the complexity of asynchronous circuits was the handling of hazards. Hazards are manifestations of undesirable switching activity called glitches. In the asynchronous style of synchronisation, the output of each circuit element is potentially sensitive to its inputs. This can give rise to non-monotonic pulses (or glitches) when transitioning between output states, as illustrated in the waveform of Figure 1 taking the case of an AND gate. Due to tight timing between the rising edge of input a and falling edge of input b, the output c produces a non-monotonic pulse before stabilising to a low. This behaviour is hazardous as it is uncertain how the fanout of the AND gate will interpret the glitch; the output c temporary switching to logical 1 or staying at logical 0 all the time.

As shown, for instance, in paper [19], the phenomenon described in the above example can be conveniently interpreted in terms of formal models such as Keller's named transition systems [11] or Petri nets [6]. In particular, what we see in this circuit is the effect where a signal that is enabled (rising edge of c) in a certain state of the circuit may become disabled without firing after the occurrence of another signal (falling edge of b). Such an effect corresponds to the violation of persistence property at the level of signal transitions if the latter are used to label the corresponding named transition system. Furthermore, when such a circuit is modelled by a labelled Petri net following the technique of [19], the Petri net would also be classified as a non-persistent one. Thus, it was shown in [19] that the modelling and analysis of an asynchronous circuit with respect to hazard-freedom is effectively reduced to the analysis of persistence of its corresponding Petri net model.

Synchronous circuits, on the other hand, do not require persistence satisfaction as they are intrinsically immune to hazardous behaviour. The principle reason being that the clock, set at worst-case latency period, filters out undesirable circuit switching. This greatly simplifies circuit design compared to asynchronous methods wherein the same circuit had to be analysed for persistence and redesigned to ensure glitch-free operation. Clocked circuits are thus preferred over asynchronous circuits for designing functionally correct (hazard-immune) ICs efficiently. However with chip sizes scaling to deep sub-micron level, semiconductors are experiencing severe variability and it is becoming extremely complicated to design chips in the synchronous fashion. This is because designing for variability requires longer safety margins which in turn reduces the clock frequency and degrades circuit performance. To cope with these challenges, asynchronous design methodologies have re-emerged owing to their *inherent* adaptiveness. However, they still suffer significant challenges such as complicated design flow, high overhead costs from control and, lack of CAD support tools and legacy design reuse. Therefore attempts are being made to find a compromise.

An on-trend intermediate solution is mixed synchronous-asynchronous design, chiefly acting in the form of Globally Asynchronous Locally Synchronous (GALS) methodology; its benefits well known in literature [10, 9, 17]. GALS system design, introduced in [5], can exploit the advantages of asynchrony and at the same time maximally reuse the products of synchronous design flow. This design technique divides a digital system into synchronous islands which communicate asynchronously by handshake mechanism. Each island has its own local clock which can be activated on demand by means of a handshake control signal. Such systems comprise a mixed temporal behaviour. Asynchronous handshakes handle switching between components where adaptability can significantly improve performance, while clocking is applied to components where worst case performance is tolerable. However, it is worthy of note that modelling GALS systems would involve detection of potential hazardous states due to presence of asynchronous components, making their design and verification a significant research challenge.

Being a recent trend, there is a lack of formal models that describe correctness of GALS designs. The complexity in modelling them begins with the investigation of persistence. It should be noted that the standard notion of persistence has been defined at the level of single actions, which is also known as interleaving semantics of concurrency. This notion has been adequate for representing the correctness of the behaviour of circuits that are fully asynchronous. In asynchronous circuits, there is concurrency between independent actions and sequential order between causally related actions. This notion is well represented by Keller's named transition systems [11]. Figure 2(a) depicts such a model capturing the asynchronous behaviour of a system with four events A, B, X and Y. Now, in synchronous circuits, the clock signal would trigger a single action or several actions. These circuits exhibit parallelism between actions in the same clock cycle and sequential order between groups of actions in adjacent clock cycles. To represent this group execution of actions, we will use *steps*, and therefore



(a) Interleaving model for asynchronous behaviour.

(b) Step model for synchronous behaviour.

(c) Mixed model for GALS behaviour.

Fig. 2. Temporal representations of systems having concurrent, parallel and mixed concurrent-parallel behaviours.

we need step transition systems to represent such a behaviour. A step represents a single action or a group of actions that are triggered simultaneously from a particular state by the clock signal. Figure 2(b) shows such a transition system model capturing the temporal behaviour of a synchronous system with the help of steps. For the case of GALS, there is a mixture of synchrony and asynchrony and hence both concurrent and parallel behaviour have to be represented. Figure 2(c) illustrates the mixed temporal behaviour seen in such systems. In all three cases, step transition systems provide a suitable behavioral model, as a single transition can be treated as a singleton step.

Synchronous and asynchronous systems have distinct techniques to guarantee functionally correct behaviour. However, for GALS systems, it is not so straightforward as correctness should be accounted from both angles. We would like to find an adequate representation of the correct behaviour of GALS systems. Here, it would be natural to define such a behaviour in analogous way as it was done for asynchronous circuits, i.e. with the use of the notion of persistence. However, when modelling GALS systems we have to consider complex actions, namely steps, and corresponding transition systems. This paper is hence centred around extending the notion of persistence to steps.

The paper is organised as follows. Section 2 recalls the basic definitions and notations concerning step transition systems and PT-nets. Section 3 introduces the notion of persistent steps and discusses their basic properties. Section 4 presents the main result of the paper, an algorithm that prunes the concurrent reachability graph of a net, which serves as an initial system specification, to obtain a representation of a desired "GALS" behavior. Finally, section 5 contains conclusions and presents directions for future work.

2 Preliminaries

In this section we recall definitions and notations concerned with step transition systems and Petri nets used in the rest of this paper.

2.1 Step Transition Systems

Let T be a finite set of net transitions representing actions of a concurrent system. A set of transitions will be called a *step*, and we will use $\alpha, \beta, \gamma, \ldots$ to range over all steps $\mathcal{P}(T)$. Sometimes we will identify a step α with its characteristic function $\alpha : T \to \{0, 1\}$. We will also write $\alpha = \sum_{t \in T} \alpha(t) \cdot t$. The size of α will be defined by the number of its elements and denoted by $|\alpha|$.

Definition 1 (st-system). *A* step transition system *(or st-system) over T is a triple*

$$STS = (Q, A, q_0)$$

consisting of a set of states *Q including the initial state $q_0 \in Q$, and a set of labelled arcs $A \subseteq Q \times \mathcal{P}(T) \times Q$. It is assumed that:*

- *the transition relation is deterministic, i.e., if $(q, \alpha, q') \in A$ and $(q, \alpha, q'') \in A$ then $q' = q''$*
- *each state is reachable, i.e., if $q \in Q$ then there are steps $\alpha_1, \ldots, \alpha_n$ $(n \geq 0)$ and states $q_1, \ldots, q_n = q$ such that $(q_{i-1}, \alpha_i, q_i) \in A$ for $1 \leq i \leq n$.*

We introduce the following notations:

- $q \xrightarrow{\alpha} q'$ and $q \xrightarrow{\alpha}$ whenever $(q, \alpha, q') \in A$.
- $En_{STS}(q) = \{\alpha \mid q \xrightarrow{\alpha}\}$ is the set of all steps enabled at a state q.
- $active_{STS}(q) = \bigcup\{\alpha \mid q \xrightarrow{\alpha}\}$ is the set of all transitions active at a state q (the transitions that feature in the steps enabled at q).
- $En_{STS} = \{\alpha \mid \exists q \in Q : \ q \xrightarrow{\alpha}\}$ is the set of all the enabled steps of STS.
- $max(q) = \{\alpha \in En_{STS}(q) \mid \forall \beta \in En_{STS}(q) : \ \alpha \not\subset \beta\}$ is the set of all maximal steps enabled at a state q.

2.2 PT-nets

A *PT-net* is a tuple $\mathcal{N} = (P, T, W, M_0)$, where P and T are disjoint sets of respectively *places* and *transitions*, $W : (P \times T) \cup (T \times P) \to \mathbb{N}$ is an arc weight function, and $M_0 : P \to \mathbb{N}$ is an *initial marking* (in general, any mapping $M : P \to \mathbb{N}$ is a marking). We will use the standard conventions concerning the graphical representation of PT-nets, as illustrated in Figure 3(a).

For every element $x \in P \cup T$, we denote

$$^\bullet x = \{y \mid W(y, x) > 0\} \ \text{(pre-set of } x),$$
$$x^\bullet = \{y \mid W(x, y) > 0\} \ \text{(post-set of } x).$$

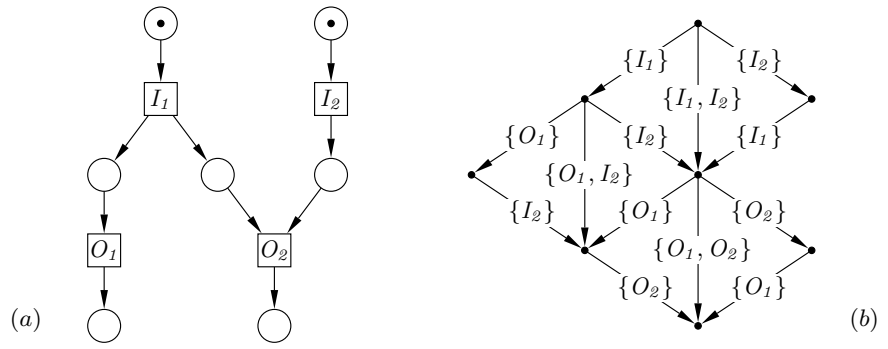


Fig. 3. Net N (a), and its concurrent reachability graph $CRG(N)$ (b).

If $x \in T$, we will call $p \in {}^\bullet x$ a pre-place of x and $p \in x^\bullet$ a post-place of x. The dot-notation extends in the usual way to sets of elements, for example, ${}^\bullet X = \bigcup_{x \in X} {}^\bullet x$.

Moreover, for every place $p \in P$ and step $\alpha \in \mathcal{P}(T)$, we denote:

$$W(p,\alpha) = \sum_{t \in T} \alpha(t) \cdot W(p,t) \quad \text{and} \quad W(\alpha,p) = \sum_{t \in T} \alpha(t) \cdot W(t,p) \; .$$

In other words, $W(p,\alpha)$ gives the number of tokens that the firing of α removes from p, and $W(\alpha,p)$ is the total number of tokens inserted into p after the execution of α.

Given a PT-net $\mathcal{N} = (P,T,W,M_0)$, a step $\alpha \in \mathcal{P}(T)$ is *enabled* and may be *fired* at a marking M if, for every place $p \in P$:

$$M(p) \geq W(p,\alpha) \; . \tag{1}$$

We denote this by $M[\alpha\rangle$. (For a singleton step $\alpha = \{t\}$, we will write $M[t\rangle$ rather than $M[\{t\}\rangle$.) Firing such a step leads to the marking M', for every place $p \in P$ defined by:

$$M'(p) = M(p) - W(p,\alpha) + W(\alpha,p) \; . \tag{2}$$

We denote this by $M[\alpha\rangle M'$.

The *concurrent reachability graph* $CRG(\mathcal{N})$ of $\mathcal{N}$ is the st-system $CRG(\mathcal{N}) = ([M_0\rangle, A, M_0)$ over T where:

$$[M_0\rangle = \{M_n \mid \exists \alpha_1, \ldots, \alpha_n \; \exists M_1, \ldots M_{n-1} \; \forall 1 \leq i \leq n \; : \; M_{i-1}[\alpha_i\rangle M_i\} \tag{3}$$

is the set of reachable markings and $(M,\alpha,M') \in A$ *iff* $M[\alpha\rangle M'$. Figure 3(b) shows the concurrent reachability graph of the PT-net in Figure 3(a). Furthermore, we will call $\alpha_1 \ldots \alpha_n$, as in the formula (3), a *step sequence* and write $M_0[\alpha_1 \ldots \alpha_n\rangle M_n$.

Definition 2 (sequential conflict). *Two distinct transitions, $t, t' \in T$, are in* sequential conflict *at a marking M if $M[t\rangle$ and $M[t'\rangle$, but $M[tt'\rangle$ does not hold.*

Definition 3 (concurrent conflict). *Two distinct transitions, $t, t' \in T$, are in* concurrent conflict *at a marking M if $M[t\rangle$ and $M[t'\rangle$, but $M[\{t,t'\}\rangle$ does not hold.*

Note that sequential conflict implies concurrent conflict, but not necessarily vice versa.

Definition 4 (safe net). *A PT-net $\mathcal{N} = (P, T, W, M_0)$ is* safe *if*

$$\forall p \in P \;\; \forall M \in [M_0\rangle : \quad M(p) \leq 1.$$

In view of the above definition, the markings of safe nets can be treated as subsets of the set of places P, where a marking is a set of places for which $M(p) = 1$.

3 Step persistence in nets

Muller's speed independent theory provided a unique method for guaranteeing hazard-freeness of asynchronous circuits [14]. The *semimodularity* condition in this work required that an excitation of a circuit element must not be removed until absorbed by the system [18]. This condition was identified by Keller in [11] [1] to be the same as the property of persistence in his named transition system model for asynchronous parallel computation. Thus satisfying the property of persistence became one of the key requirements when designing hazard-free asynchronous circuits.

Later, the idea of persistence was investigated in many papers, for example, in [1–4, 7, 13, 19]. However, with the exception of [7], it was only considered in the context of sequential executions of systems, and defined for transitions (rather than steps) as follows:

Definition 5 (persistent net, [13]). *A PT-net $\mathcal{N} = (P, T, W, M_0)$ is* persistent *if for all distinct transitions $t, t' \in T$ and any reachable marking M, $M[t\rangle$ and $M[t'\rangle$ imply $M[tt'\rangle$.*

We can re-write this definition from the point of view of single transition as follows:

Definition 6 (persistent transition). *A transition $t \in T$ is* persistent *in a PT-net $\mathcal{N} = (P, T, W, M_0)$ if*

$$\forall M \in [M_0\rangle \;\; \forall t' \in T \setminus \{t\} : \; M[t\rangle \wedge M[t'\rangle \Longrightarrow M[t't\rangle.$$

The following definition gives three versions (A, B and C) of a definition of a persistent step. In each case, we try to capture the fact that a persistent step, which is enabled at some reachable marking M, cannot be disabled by another enabled step. The difference in the versions lies either in the different understanding of what 'not to be disabled' means or what we mean by a 'different'

[1] Keller ([11]) was the first to consider persistence in the context of Petri nets.

step. Notice that the introduced notions of a persistent step are defined globally and the required conditions must be satisfied at all the markings, where a candidate for a persistent step is enabled.[2]

Definition 7 (persistent step in a net). *A step $\alpha \in \mathcal{P}(T)$ is* A-persistent, B-persistent and C-persistent *in a PT-net $\mathcal{N} = (P, T, W, M_0)$ if respectively the following hold:*

$$\begin{array}{lll} (A) & \forall M \in [M_0\rangle\ \forall \beta \in \mathcal{P}(T): & M[\alpha\rangle \wedge M[\beta\rangle \wedge \beta \neq \alpha \implies M[\beta(\alpha \setminus \beta)\rangle \\ (B) & \forall M \in [M_0\rangle\ \forall \beta \in \mathcal{P}(T): & M[\alpha\rangle \wedge M[\beta\rangle \wedge \beta \cap \alpha = \varnothing \implies M[\beta\alpha\rangle \\ (C) & \forall M \in [M_0\rangle\ \forall \beta \in \mathcal{P}(T): & M[\alpha\rangle \wedge M[\beta\rangle \wedge \beta \neq \alpha \implies M[\beta\alpha\rangle\ . \end{array}$$

As can be easily seen, each of the three versions is a conservative extension of the standard definition of a persistent transition (see Definition 6). A-persistence requires that only unexecuted part of a step α should not be disabled, while B-persistence and C-persistence insist on continued enabledness of the persistent step α. In B-persistence, two steps are considered different if their intersection is empty, while for A-persistence and C-persistence it is enough if different steps do not coincide (but they can have non-empty intersection). It turns out that A-persistence and B-persistence are equivalent, as is shown in the following proposition.

Proposition 1. *Let $\alpha \in \mathcal{P}(T)$ be a step of a PT-net $\mathcal{N} = (P, T, W, M_0)$. Then α is A-persistent in $\mathcal{N}$ iff α is B-persistent in $\mathcal{N}$.*

Proof. Suppose that α and $\beta \in \mathcal{P}(T)$ are two different non-empty steps enabled at a marking M of $\mathcal{N}$ (notice that empty step is trivially persistent according to A, B or C-persistence defined in Definition 7).
$\implies$ Let α be A-persistent in $\mathcal{N}$, and so $M[\beta(\alpha \setminus \beta)\rangle$. Suppose $\beta \cap \alpha = \varnothing$. Then we have $\alpha \setminus \beta = \alpha$, implying $M[\beta\alpha\rangle$. Hence, α is B-persistent in $\mathcal{N}$.
$\impliedby$ Let α be B-persistent in $\mathcal{N}$, and so $M[(\beta \setminus \alpha)\alpha\rangle$ as $(\beta \setminus \alpha) \cap \alpha = \varnothing$. Hence, for every place $p \in P$:

$$M(p) - W(p, \beta \setminus \alpha) + W(\beta \setminus \alpha, p) \geq W(p, \alpha)$$

which implies:

$$M(p) - W(p, \beta) + W(p, \beta \cap \alpha) + W(\beta, p) - W(\beta \cap \alpha, p) \geq W(p, \alpha)\ .$$

Hence we have

$$\begin{array}{rl} M(p) - W(p, \beta) + W(\beta, p) & \geq W(p, \alpha) - W(p, \beta \cap \alpha) + W(\beta \cap \alpha, p) \\ & = W(p, \alpha \setminus \beta) + W(\beta \cap \alpha, p) \\ & \geq W(p, \alpha \setminus \beta) \end{array}$$

implying $M[\beta(\alpha \setminus \beta)\rangle$. Hence α is A-persistent. □

[2] A more detailed analysis of persistent steps, which considers also steps that are locally persistent, is presented in a companion paper, "A Taxonomy of Persistent and Nonviolent Steps" by M. Koutny, Ł. Mikulski and M. Pietkiewicz-Koutny, submitted to ICATPN 2013 (see also [12]).

It is easy to see that C-persistence is stronger than the other two notions. Figure 4 ([12]) shows an example of a step, $\{t, \overline{t}\}$, which is A-persistent, but not C-persistent. The step $\{t, \overline{t}\}$ there is only enabled at marking M_3, where also $\{\overline{t}\}$ and $\{t\}$ steps are enabled. After executing step $\{\overline{t}\}$, step $\{t, \overline{t}\}$ is still enabled, but after executing step $\{t\}$, only unexecuted part of $\{t, \overline{t}\}$ (that means $\{\overline{t}\}$) is enabled.

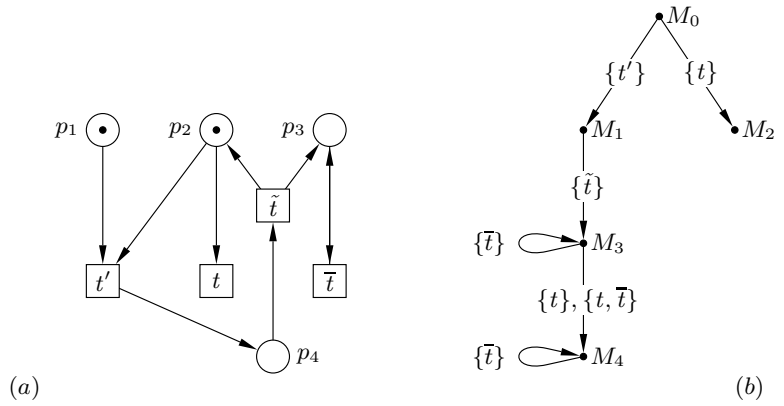


Fig. 4. A safe net $\mathcal{N}$ (a), and its concurrent reachability graph $CRG(\mathcal{N})$ (b). Step $\{t, \overline{t}\}$ is A-persistent, but not C-persistent.

For a safe PT-net $\mathcal{N}$, C-persistent non-singleton steps are built out of transitions lying on self-loops. To show this, we first prove an auxiliary result.

Proposition 2. *Let α be a C-persistent step of a safe PT-net $\mathcal{N} = (P, T, W, M_0)$ enabled at a reachable marking M of $\mathcal{N}$. Then ${}^\bullet(\alpha \cap \beta) = (\alpha \cap \beta)^\bullet$ for every step $\beta \neq \alpha$ enabled at M in $\mathcal{N}$.*

Proof. Suppose $p \in {}^\bullet(\alpha \cap \beta)$ for some step $\beta \neq \alpha$ enabled at M in $\mathcal{N}$. This and $M[\alpha\rangle$ (as α is enabled at M) imply $M(p) = 1$. Since α is C-persistent, there exists a marking M' such that $M[\beta\rangle M'[\alpha\rangle$. Again, as $M'[\alpha\rangle$ and $p \in {}^\bullet(\alpha \cap \beta)$, we have $M'(p) = 1$. From $M[\beta\rangle M'$, we have $M'(p) = M(p) - W(p, \beta) + W(\beta, p)$, and so $\sum_{t \in T} \beta(t) \cdot W(t, p) = \sum_{t \in T} \beta(t) \cdot W(p, t)$. Let $\beta = \{t_1, \ldots, t_n\}$. Then, $W(t_1, p) + \ldots + W(t_n, p) = W(p, t_1) + \ldots + W(p, t_n)$ and, by $\mathcal{N}$ being safe, all the arc weights in this formula are 0 or 1. We now consider two cases:

1. There is $i \leq n$ such that $W(t_i, p) = 1$ and $W(p, t_i) = 0$. Since $M[t_i\rangle$ (as $M[\beta\rangle$), $M(p) = 1$ and $\mathcal{N}$ is safe, we have a contradiction, because t_i, when fired, would deposit another token in p.
2. There is $j \leq n$ such that $W(t_j, p) = 0$ and $W(p, t_j) = 1$. This means that there is $i \leq n$, $W(t_i, p) = 1$ and $W(p, t_i) = 0$. But this was already ruled out by the first case. So, contradiction again.

As a result, for each transition $t_i \in \beta$, $W(p, t_i) = W(t_i, p)$. Hence $p \in (\alpha \cap \beta)^\bullet$. Consequently, ${}^\bullet(\alpha \cap \beta) \subseteq (\alpha \cap \beta)^\bullet$.

Suppose now that $p \in (\alpha \cap \beta)^\bullet \setminus {}^\bullet(\alpha \cap \beta)$. Then, by $M[\alpha \cap \beta\rangle$ and the safeness of $\mathcal{N}$, $M(p) = 0$. Hence, by $M[\alpha\rangle$ and $M[\beta\rangle$, we must have $p \notin {}^\bullet\alpha \cup {}^\bullet\beta$. Consequently, since there is M'' such that $M[\beta\alpha\rangle M''$, we obtain $M''(p) \geq 2$, a contradiction with $\mathcal{N}$ being safe. Hence ${}^\bullet(\alpha \cap \beta) = (\alpha \cap \beta)^\bullet$. □

Theorem 1. *Let α be a non-singleton C-persistent step of a safe PT-net $\mathcal{N} = (P, T, W, M_0)$ which is enabled in at least one reachable marking. Then all the transitions of α lie on self-loops, i.e., ${}^\bullet t = t^\bullet$ for $t \in \alpha$.*

Proof. Suppose that $t \in \alpha$ and M be a reachable marking enabling α. Since $\{t\} \neq \alpha$ and $M[t\rangle$ for any marking M such that $M[\alpha\rangle$, we have, from Proposition 2, ${}^\bullet(\alpha \cap \{t\}) = (\alpha \cap \{t\})^\bullet$. Hence ${}^\bullet t = t^\bullet$. □

We now want to relate the persistence of a step with the persistence of its constituent transitions in safe nets. We first consider A-persistent steps, but as we already know the results will also hold for B-persistent steps.

First, we prove a simple, but important, fact concerning pre-sets and post-sets of transitions in steps of safe nets.

Fact 1 *If α is a step enabled in a reachable marking of a safe PT-net $\mathcal{N}$, then $({}^\bullet t \cup t^\bullet) \cap ({}^\bullet u \cup u^\bullet) = \varnothing$, for all distinct transitions $t, u \in \alpha$.*

Proof. Let $t, u \in \alpha$ and M be a reachable marking such that $M[\alpha\rangle$.

Suppose that $p \in {}^\bullet t \cap {}^\bullet u$. Since $M[\alpha\rangle$, we obtain $M[\{t, u\}\rangle$. That means $M(p) \geq W(p, t) + W(p, u) = 2$, a contradiction with $\mathcal{N}$ being safe. As a result, ${}^\bullet t \cap {}^\bullet u = \varnothing$.

Suppose now that $p \in t^\bullet \cap u^\bullet$. Since $M[\alpha\rangle$, we obtain $M[\{t, u\}\rangle M'$. Hence $M'(p) = M(p) - W(p, t) - W(p, u) + W(t, p) + W(u, p)$. As t and u cannot share a pre-place, $W(p, t)$ and $W(p, u)$ cannot both be 1. If one of them has p as its pre-place, then $M(p) = 1$, as otherwise one of the transitions would not be enabled at M, and both are enabled at M. So, the right hand side of the equation yields 2, but the left hand side cannot as the net is safe. We have a contradiction. As a result, $t^\bullet \cap u^\bullet = \varnothing$.

Suppose now that $p \in t^\bullet \cap {}^\bullet u$. By $M[u\rangle$, $M(p) = 1$. On the other hand, we know that $p \notin {}^\bullet t \cap {}^\bullet u$ and so $p \in t^\bullet \setminus {}^\bullet t$. Since $M[t\rangle M''$, for some marking M'', we obtain $M''(p) = 2$, a contradiction with $\mathcal{N}$ being safe. Hence, $t^\bullet \cap {}^\bullet u = \varnothing$. □

Theorem 2. *Let α be a step in a safe PT-net $\mathcal{N} = (P, T, W, M_0)$ which is enabled in at least one reachable marking. If all the transitions in α are persistent in $\mathcal{N}$, then α is A-persistent in $\mathcal{N}$.*

Proof. Let M be a reachable marking and $\beta \neq \alpha$ be a step in $\mathcal{N}$ such that $M[\alpha\rangle$ and $M[\beta\rangle$. We need to show that $M[\beta(\alpha \setminus \beta)\rangle$.

Assume that $\alpha \cap \beta = \{t_1, \ldots, t_m\}$, $\alpha \setminus \beta = \{w_1, \ldots, w_n\}$ and $\beta \setminus \alpha = \{u_1, \ldots, u_k\}$. Note that all the transitions in these three sets are different. From $M[\beta\rangle$ we have $M[t_1 \ldots t_m u_1 \ldots u_k\rangle$. Now, since each w_i is persistent and enabled

at M, we have that $M[t_1 \dots t_m u_1 \dots u_k w_1 \dots w_n\rangle$. Since α and β are steps in a safe net $\mathcal{N}$ enabled at some marking (M), we have, from Fact 1, that transitions in α and β have disjoint pre-sets and post-sets. Hence we have $M[\beta(\alpha \setminus \beta)\rangle$. □

We now consider C-persistent steps. In this case the antecedent in the implication is stronger.

Theorem 3. *Let α be a step in a safe PT-net $\mathcal{N} = (P, T, W, M_0)$ which is enabled in at least one reachable marking. If all the transitions in α are persistent and lie on self-loops in $\mathcal{N}$, then α is C-persistent in $\mathcal{N}$.*

Proof. Let M be a reachable marking and $\beta \neq \alpha$ be a step in $\mathcal{N}$ such that $M[\alpha\rangle$ and $M[\beta\rangle$. We need to show that $M[\beta\alpha\rangle$.

Proceeding similarly as in the previous proof we can show that

$$M[t_1 \dots t_m u_1 \dots u_k w_1 \dots w_n\rangle \ .$$

Now, since all transitions in α lie on self-loops we further obtain

$$M[t_1 \dots t_m u_1 \dots u_k t_1 \dots t_m w_1 \dots w_n\rangle \ .$$

Hence we have $M[\beta\alpha\rangle$. □

In both Theorem 2 and Theorem 3, the implications in the opposite direction do not hold. A counterexample is shown in the Figure 5.

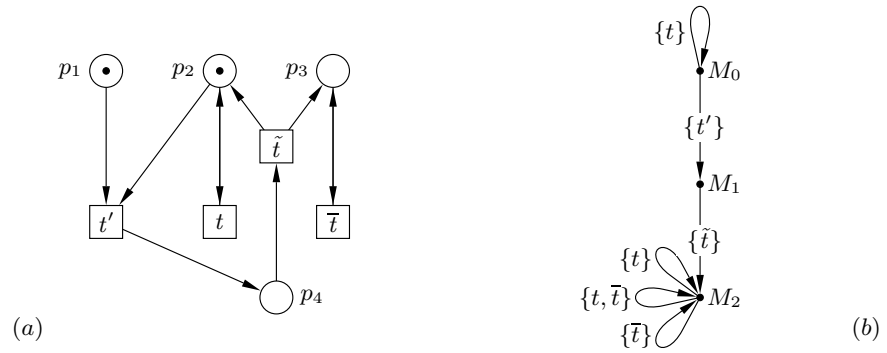


Fig. 5. A safe net $\mathcal{N}$ (a), and its concurrent reachability graph $CRG(\mathcal{N})$ (b). A persistent step $\alpha = \{t, \overline{t}\}$ of $\mathcal{N}$ contains a non-persistent transition t.

Observe that step $\alpha = \{t, \overline{t}\}$ in Figure 5 is both A-persistent and C-persistent, but $t \in \alpha$ is not persistent at $M_0 = \{p_1, p_2\}$, because there exists $t' \neq t$ such that $M_0[t\rangle$ and $M_0[t'\rangle$, but $M_0[t't\rangle$ does not hold.

4 Pruning reachability graphs

The main motivation for studying persistent steps in this paper is to discover which sets of transitions can be executed synchronously and therefore be treated as some kind of "atomic actions", giving rise to new "bigger" transitions, which would execute in a "hazard-free" way. We will call them *bundles*. Looking at our application area of asynchronous circuits, bundling actions would reduce signal management by merging concurrent signals into one event. This merging must be done in a consistent fashion. The best candidates for bundles are persistent steps, but if we want to form "bigger" transitions from them, we must make sure that one enabled persistent step does not include another enabled persistent step. All the transitions in a bundle must always appear together, in the same configurations. In the ideal situation (we say ideal, because it might be difficult to achieve), we do not want to allow, for example, three persistent steps $\{a, b\}$, $\{a\}$ and $\{b\}$ to be enabled in a given transition system. We need to choose: either to opt for $\{a, b\}$ and delete $\{a\}$ and $\{b\}$, or the other way round. Therefore, we need to develop an algorithm which, for a given net $\mathcal{N} = (P, T, W, M_0)$, would allow us to prune its reachability graph $CRG(\mathcal{N})$ in such a way that all persistent steps would satisfy an additional "non-inclusion" condition. The "pruned" transition system would represent the desired behaviour, which then we would like to implement in a form of a Petri net in a process of synthesis.

First we define a sub-st-system, which will be obtained as a result of pruning a given reachability graph.

Definition 8 (sub-st-system). *An st-system $STS = (Q, A, q_0)$ is a* sub-st-system *of an st-system $STS' = (Q', A', q_0)$ if $Q \subseteq Q'$, $A \subseteq A'$ and, for every $q \in Q$, $active_{STS}(q) = active_{STS'}(q)$. We denote this by $STS \preccurlyeq STS'$.*

In the above definition, En_{STS} of a "properly pruned" reachability graph STS' will be a set of <u>bundles</u>. What we mean by "properly pruned" will be described by conditions stated in Problem 1.

First, we need to re-define the three notions of step persistence, that were used for nets, in the context of transition systems. Once we start pruning an st-system, we need to decide whether the remaining steps that were previously persistent remain persistent. Checks for persistence will be done in a transition system (that might not be a reachability graph of any net).

Definition 9 (persistent step in a transition system). *A step $\alpha \in En_{STS}$ is* A-persistent, B-persistent and C-persistent *in an st-system $STS = (Q, A, q_0)$ if respectively the following hold for all states $q \in Q$ and steps β such that $\stackrel{\alpha}{\longleftarrow} q \stackrel{\beta}{\longrightarrow}$:*

$$
\begin{array}{lrcl}
(A) & \beta \neq \alpha & \Longrightarrow & q \xrightarrow{\beta(\alpha \setminus \beta)} \\
(B) & \beta \cap \alpha = \varnothing & \Longrightarrow & q \xrightarrow{\beta\alpha} \\
(C) & \beta \neq \alpha & \Longrightarrow & q \xrightarrow{\beta\alpha} .
\end{array}
$$

Remark 1. A step α is A-persistent in a net $\mathcal{N}$ iff α is A-persistent in a transition system $CRG(\mathcal{N})$. The same can be said in the case of B- and C-persistence.

We have the following relationships between the three step persistence notions defined for transition systems.

Proposition 3. *Let $STS = (Q, A, q_0)$ be a st-system.*

1. *If $\alpha \in En_{STS}$ is A-persistent, then it is also B-persistent.*
2. *If $\alpha \in En_{STS}$ is C-persistent, then it is also B-persistent.*

Proof. (1) Let $q \in Q$ and $\stackrel{\alpha}{\leftarrow} q \stackrel{\beta}{\rightarrow}$ be such that $\beta \cap \alpha = \varnothing$. Since $\alpha \in En_{STS}$ is A-persistent, we have $q \xrightarrow{\beta(\alpha\setminus\beta)}$. Hence $q \xrightarrow{\beta\alpha}$ which means that $\alpha \in En_{STS}$ is B-persistent.

(2) Follows directly from Definition 9. □

Note that in the class of general step transition systems, B-persistence does not imply A-persistence of steps, as it was proved for nets. Indeed, let $\alpha \in En_{STS}$ be B-persistent step in STS, and $\beta \neq \alpha$ and $q \in Q$ be such that $\stackrel{\alpha}{\leftarrow} q \stackrel{\beta}{\rightarrow}$. We know that $\beta \cap (\alpha \setminus \beta) = \varnothing$. However, with such assumptions, we cannot in general guarantee that $q \xrightarrow{\alpha\setminus\beta}$. Though latter is true for concurrent reachability graphs of PT-nets, we must also consider step transition systems resulting from the pruning of such reachability graphs.

Problem 1 *Let $\mathcal{N}$ be a PT-net and $CRG(\mathcal{N})$ be its concurrent reachability graph. Find an st-system STS such that $STS \preccurlyeq CRG(\mathcal{N})$ and additionally satisfying $(D)\&(E)$ or $(D)\&(F)$, where the three conditions are defined as follows:*

(D) All steps in En_{STS} are B-persistent in STS. [3]
(E) $\alpha \not\subset \beta$ for all nonempty steps $\alpha, \beta \in En_{STS}$.
(F) $\alpha \not\subset \beta$ for all states q and all nonempty steps $\alpha, \beta \in En_{STS}(q)$.

We denote this respectively by

$$STS \preccurlyeq^{global}_{pers} CRG(\mathcal{N}) \quad \textit{and} \quad STS \preccurlyeq^{local}_{pers} CRG(\mathcal{N}) \ .$$

We also refer to the condition described in (E) as global non-inclusion*, and to the condition described in (F) as* local non-inclusion*.*

The difference between $\preccurlyeq^{global}_{pers}$ and $\preccurlyeq^{local}_{pers}$ is that the latter only requires non-inclusion of bundles locally for each state, whereas the former insists that non-inclusion holds globally. We therefore have

Proposition 4. *$STS \preccurlyeq^{global}_{pers} CRG(\mathcal{N})$ implies $STS \preccurlyeq^{local}_{pers} CRG(\mathcal{N})$.*

In our first attempt to solve Problem 1, we will concentrate on PT-nets that are persistent according to Definition 5. We then have the following result.

[3] Alternatively, we could require A-persistence or C-persistence. We opted here for B-persistence, because it is the weakest of the three notions.

Theorem 4. *If $\mathcal{N}$ is persistent (according to Definition 5), then there is at least one STS satisfying STS $\preccurlyeq^{global}_{pers}$ $CRG(\mathcal{N})$.*

Proof. It suffices to take $CRG(\mathcal{N})$ and delete all non-singleton steps. □

As the above proof produces completely sequential solution, we will now search for a more concurrent one. We will also require that the original PT-net in not only persistent, but as well safe.

Proposition 5. *If $\mathcal{N}$ is persistent (according to Definition 5) and safe, then every step $\alpha \in En_{CRG(\mathcal{N})}$ is B-persistent in $CRG(\mathcal{N})$.*

Proof. Let $\alpha \in En_{CRG(\mathcal{N})}$. If $\mathcal{N}$ is persistent (according to Definition 5), all transitions in α are persistent (according to Definition 6). Hence, from Theorem 2 and the fact that $\mathcal{N}$ is safe, we have that α is A-persistent in $\mathcal{N}$, and also B-persistent in $\mathcal{N}$ (see Proposition 1). Following Remark 1, we conclude that α is B-persistent in $CRG(\mathcal{N})$. □

The above proposition guarantees B-persistence of steps in $CRG(\mathcal{N})$ of a persistent and safe net $\mathcal{N}$, but the non-inclusion conditions ((E) or (F)) are, in general, not satisfied in $CRG(\mathcal{N})$, as for all its states q, $q \xrightarrow{\alpha}$ implies $q \xrightarrow{\beta}$ for any step $\beta \subset \alpha$. To satisfy the non-inclusion conditions, we need to prune $CRG(\mathcal{N})$ in such a way that B-persistence of steps is maintained. We will now explore what happens if we decide to prune all but the maximal steps at every reachable marking.

In what follows the st-system $CRG^{max}(\mathcal{N})$ is obtained from $CRG(\mathcal{N})$, the concurrent reachability graph of a PT-net $\mathcal{N}$, by deleting at every reachable marking M, all the arcs labelled by non-maximal non-empty [4] steps, and then removing the nodes that became unreachable from the initial marking by the removal of such steps.

Proposition 6. $CRG^{max}(\mathcal{N}) \preccurlyeq CRG(\mathcal{N})$.

Proof. Follows from definitions and the fact that for each enabled step there is a maximal step enabled at the same marking. □

Proposition 7. *$CRG^{max}(\mathcal{N})$ satisfies condition (F) from Problem 1.*

Proof. Follows from the fact that maximal steps are in-comparable (see definition of maximal steps in section 2.1). □

Figures: 6, 7, 8 and 9 show the examples of persistent and safe nets for which the described pruning procedure works as their $CRG^{max}(\mathcal{N})$ graphs contain only B-persistent steps. In all the mentioned examples the pruned reachability graph satisfies $CRG^{max}(\mathcal{N}) \preccurlyeq^{local}_{pers} CRG(\mathcal{N})$, and in case of the example in Figure 6, we even have $CRG^{max}(\mathcal{N}) \preccurlyeq^{global}_{pers} CRG(\mathcal{N})$. So, the pruning procedure helped to

[4] For technical reasons we do not want to delete empty steps, as they might be important in future algorithms.

achieve local non-inclusion without jeopardising B-persistence of the remaining steps. Notice, however, that in Figures 7(c) and 9(c) this B-persistence in initial markings is achieved only, because the steps enabled there are not disjoint and therefore satisfy B-persistence condition trivially.

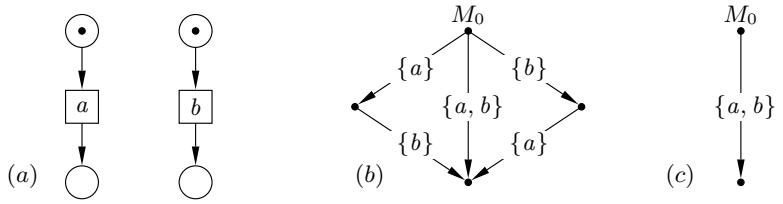


Fig. 6. A persistent and safe net $\mathcal{N}$ (a), its concurrent reachability graph $CRG(\mathcal{N})$ (b), and $CRG^{max}(\mathcal{N}) \preccurlyeq^{local}_{pers} CRG(\mathcal{N})$ obtained in the pruning procedure (c).

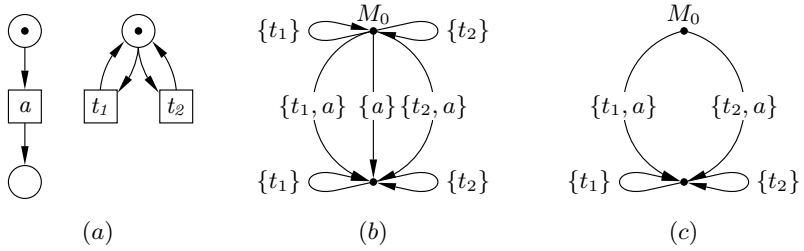


Fig. 7. A persistent and safe net $\mathcal{N}$ (a), its concurrent reachability graph $CRG(\mathcal{N})$ (b), and $CRG^{max}(\mathcal{N}) \preccurlyeq^{local}_{pers} CRG(\mathcal{N})$ obtained in the pruning procedure (c).

In general, pruning non-maximal steps may make some of the remaining steps non-B-persistent. Figure 10 shows that the initially enabled step $\{b\}$ is not B-persistent after the pruning procedure. After executing step $\{a\}$ it is not longer enabled. Instead step $\{b, c\}$ is enabled, because it was the maximal step in the marking M. Having said that, we propose a weaker version of condition (B) which holds for safe and persistent PT-nets.

Proposition 8. *If $\mathcal{N}$ is persistent (according to Definition 5) and safe, then, for every marking M in $CRG^{max}(\mathcal{N})$, $\stackrel{\alpha}{\leftarrow} M \stackrel{\beta}{\rightarrow}$ implies:*

$$(B')\ \beta \cap \alpha = \varnothing \quad \Longrightarrow \quad \exists \gamma :\ \alpha \subseteq \gamma \wedge M \stackrel{\beta\gamma}{\longrightarrow} .$$

Proof. From Proposition 5 we know that $M \stackrel{\beta\alpha}{\longrightarrow}$ in $CRG(\mathcal{N})$. Moreover, there is a maximal step γ available (as it remained after pruning) after executing β at M such that $\alpha \subseteq \gamma$. Hence $M \stackrel{\beta\gamma}{\longrightarrow}$ in $CRG^{max}(\mathcal{N})$. □

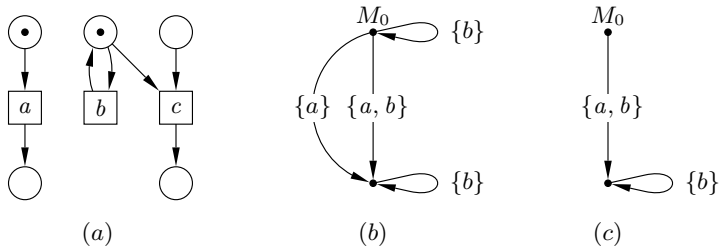


Fig. 8. A persistent and safe net $\mathcal{N}$ (a), its concurrent reachability graph $CRG(\mathcal{N})$ (b), and $CRG^{max}(\mathcal{N}) \preccurlyeq^{local}_{pers} CRG(\mathcal{N})$ obtained in the pruning procedure (c).

Hence, pruning non-maximal steps may result in the loss of persistence when $\alpha \subset \gamma$ in (B'). In such a case we may, however, 'repair' $\mathcal{N}$ by making the step γ non-enabled. The mechanism for achieving this is simple, namely we select one transition from α, one transition from $\gamma \setminus \alpha$, and make sure that they cannot be executed simultaneously.

Let $\mathcal{N}$ be a PT-net and $t \neq u$ be two transitions. Then $\mathcal{N}_{t \leftrightarrow u}$ is a PT-net obtained from $\mathcal{N}$ by adding a new place p marked initially with one token, and such that $W(p,t) = W(t,p) = W(p,u) = W(u,p) = 1$. This construction is illustrated in Figure 11, where we try to fix the problem of the net $\mathcal{N}$ in Figure 10. We added a new place p and chose b and c as our t and u (the only choice in this example) creating a new net $\mathcal{N}' = \mathcal{N}_{b \leftrightarrow c}$. The new place disables the concurrent step $\{b, c\}$ at M leaving the singleton steps $\{b\}$ and $\{c\}$ enabled at M, as they are now maximal steps at this marking. In fact, in this simple example, we have only non-singleton steps in the concurrent reachability graph, which makes it maximal without pruning. More involved example is shown in Figure 12 and 13, in the Appendix.

In the following propositions we show that after the proposed modification the net generates a reachability graph, which is the sub-st-system of the reachability graph of the initial net. Also, the modified net is still persistent (according to Definition 5) and safe.

Proposition 9. *Let $\mathcal{N}$ be persistent (according to Definition 5) and safe net. The reachable markings of $CRG(\mathcal{N}_{t \leftrightarrow u})$ and $CRG(\mathcal{N})$ are the same, if we identify each reachable marking M of $\mathcal{N}$ with the reachable marking $M \cup \{p\}$ of $\mathcal{N}_{t \leftrightarrow u}$. Furthermore, $CRG(\mathcal{N}_{t \leftrightarrow u}) \preccurlyeq CRG(\mathcal{N})$.*

Proof. Follows from definitions. □

Proposition 10. *If $\mathcal{N}$ is persistent (according to Definition 5) and safe, then $CRG(\mathcal{N}_{t \leftrightarrow u})$ is also persistent (according to Definition 5) and safe.*

Proof. Follows from definitions. □

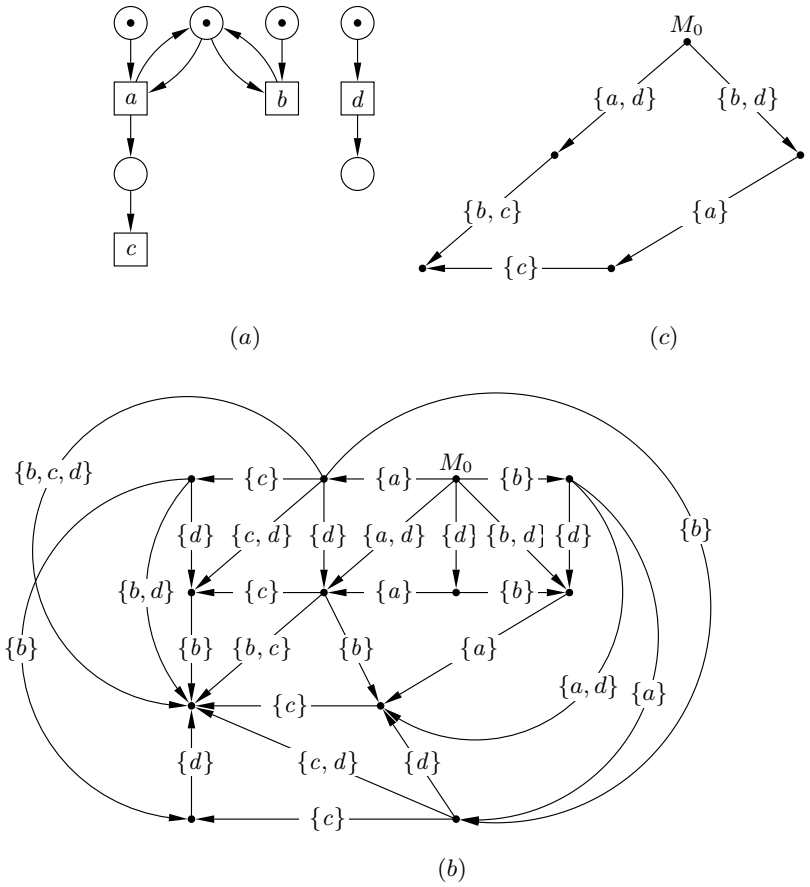


Fig. 9. A persistent and safe net $\mathcal{N}$ (a), its concurrent reachability graph $CRG(\mathcal{N})$ (b), and $CRG^{max}(\mathcal{N}) \preccurlyeq^{local}_{pers} CRG(\mathcal{N})$ obtained in the pruning procedure (c).

We can now propose a dynamic way of pruning embodied by the following algorithm:

while $\neg(CRG^{max}(\mathcal{N}) \preccurlyeq^{local}_{pers} CRG(\mathcal{N}))$ **do**
  choose M, α, β, γ *in* $CRG^{max}(\mathcal{N})$ *satisfying* (B') *with* $\alpha \subset \gamma$
  choose $t \in \alpha,\ u \in \gamma \setminus \alpha$
  $\mathcal{N} := \mathcal{N}_{t \leftrightarrow u}$

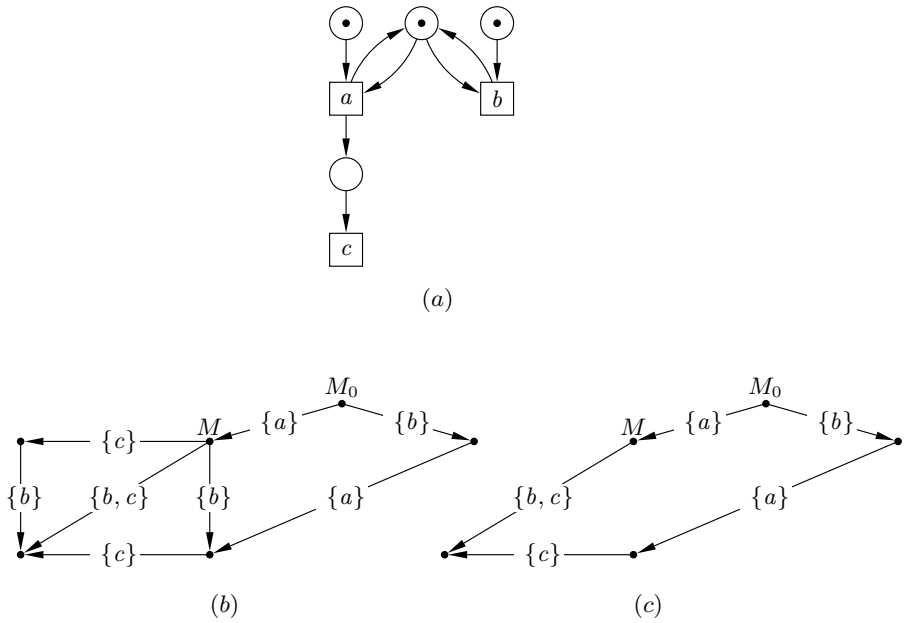


Fig. 10. A persistent and safe net $\mathcal{N}$ (a), its concurrent reachability graph $CRG(\mathcal{N})$ (b), and $CRG^{max}(\mathcal{N})$ obtained in the pruning procedure, which does not satisfy $CRG^{max}(\mathcal{N}) \preccurlyeq^{local}_{pers} CRG(\mathcal{N})$ (c).

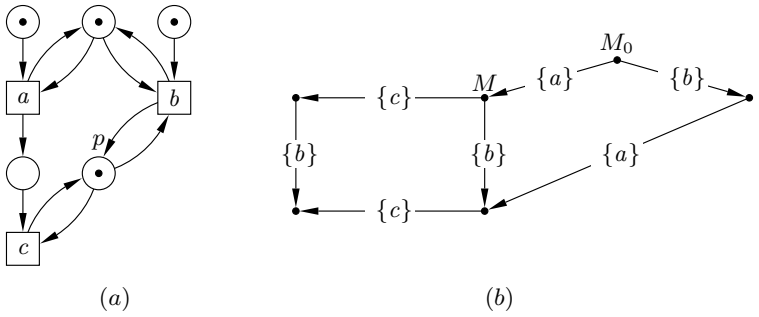


Fig. 11. Fixing the problem in the example in Figure 10: persistent and safe net $\mathcal{N}' = \mathcal{N}_{b \leftrightarrow c}$ (a), its concurrent reachability graph $CRG(\mathcal{N}') = CRG^{max}(\mathcal{N}')$, which trivially satisfies $CRG^{max}(\mathcal{N}') \preccurlyeq^{local}_{pers} CRG(\mathcal{N}')$ (and also $CRG^{max}(\mathcal{N}') \preccurlyeq^{global}_{pers} CRG(\mathcal{N}')$) (b).

It follows from what we already demonstrated that the above algorithm always terminates and for the final PT-net $\mathcal{N}$ we have:

$$CRG^{max}(\mathcal{N}) \preccurlyeq^{local}_{pers} CRG(\mathcal{N}) \ .$$

Since the algorithm is non-deterministic, we may try various strategies for choosing t and u.

5 Conclusions

In GALS, bundling is envisioned to reduce signal management, and could reduce the cost of scheduling and control, and improve system performance. The ideal way to model mixed synchronous-asynchronous systems is to start with a concurrent model that is persistent and fully asynchronous in behaviour. Then run several iterations that derive a combination of bundles that represents the temporal nature the designer requires. Careful selection of bundles is essential so that the pruned behaviour of the fully asynchronous model still exhibits some characteristics of its parent and is persistent. Step persistence is hence an important characteristic that will guarantee true persistent behaviour for mixed synchronous-asynchronous models.

In this paper we developed a pruning procedure for reachability graphs of persistent and safe nets. This procedure constructs a step transition system that contains only bundles. The bundles in our algorithm represent maximally concurrent steps of the initial system. In future we intend to investigate other possible pruning algorithms, weakening our constraints and allowing the initial system's behaviour to be given by a net that is not necessarily persistent. Furthermore, we plan to allow in the algorithms the choice of non-maximal bundles in certain cases. For example, input signals are usually behaving in fully asynchronous way and should not be bundled.

Acknowledgments

This research was supported by the Epsrc Gaels project.

References

1. Barylska, K., Ochmański, E.: Levels of Persistency in Place/Transition Nets. Fundamenta Informaticae 93(1-3) (2009) 33–43.
2. Barylska, K., Mikulski, Ł., Ochmański, E.: On Persistent Reachability in Petri Nets. Information and Computation 223 (2013) 67-77.
3. Best, E., Darondeau, P.: Decomposition Theorem for Bounded Persistent Petri Nets. 29th International Conference, Petri Nets 2008, Xi'an, China, June 2008, LNCS 5062 (2008) 33–51.
4. Best, E., Darondeau, P.: Separability in Persistent Petri Nets. 31st International Conference, Petri Nets 2010, Braga, Portugal, June 2010, LNCS 6128 (2010) 246–266.
5. Chapiro, D.M.: Globally-Asynchronous Locally-Synchronous Systems. PhD thesis, Stanford University (1984).
6. Cortadella, J., Kishinevsky, M., Kondratyev, A., Lavagno, L., Yakovlev, A.: Logic Synthesis for Asynchronous Controllers and Interfaces. Springer Series in Advanced Microelectronics, 8, Springer-Verlag GmbH (2002).

7. Dasgupta, S., Yakovlev, A.: Desynchronisation Technique using Petri Nets. Electronic Notes in Theoretical Computer Science 245 (2009) 51–67.
8. Davis, A., Nowick, S.M.: An introduction to asynchronous circuit design. The Encyclopedia of Computer Science and Technology (1997).
9. Gurkaynak, F., Oetiker, S., Kaeslin, H., Felber, N., Fichtner, W.: GALS at ETH Zurich: Success or Failure ? Proceedings of the 12th IEEE International Symposium on Asynchronous Circuits and Systems, Grenoble, France, March 2006.
10. Iyer, A., Marculescu, D.: Power and performance evaluation of globally asynchronous locally synchronous processors. Computer Architecture 4901 (2002).
11. Keller, R.: A fundamental theorem of asynchronous parallel computation. Lecture Notes in Computer Science 24 (1975) 102–112.
12. Koutny, M., Mikulski, Ł., Pietkiewicz-Koutny, M.: A Taxonomy of Persistent and Nonviolent Steps. Technical Report 1361, School of Computing Science, Newcastle University (2012).
13. Landweber, L.H., Robertson, E.L.: Properties of Conflict-Free and Persistent Petri Nets. JACM 25(3) (1978) 352–364.
14. Muller, D.E., Bartky, W.S.: A theory of asynchronous circuits. Proceedings of an International Symposium on the Theory of Switching, Harvard University Press (1959) 204-243.
15. Myers, C.: Asynchronous Circuit Design. Wiley (2004).
16. Sparso, J., Furber, S.: Principles of asynchronous circuit design: a systems perspective. Kluwer Academic Publishers (2001).
17. Stevens, K.S., Gebhardt, D., You, J., Xu, Y., Vij, V., Das, S., Desai, K.: The Future of Formal Methods and GALS Design. Electronic Notes in Theoretical Computer Science 245 (2009) 115–134.
18. Yakovlev, A.V.: Designing self-timed systems. VLSI SYSTEMS DESIGN VI (1985) 70–90.
19. Yakovlev, A.V., Koelmans, A.M., Semenov, A., Kinniment, D.J.: Modelling, Analysis and Synthesis of Asynchronous Control Circuits Using Petri Nets. INTEGRATION, the VLSI Journal 21 (1996) 143–170.

Appendix

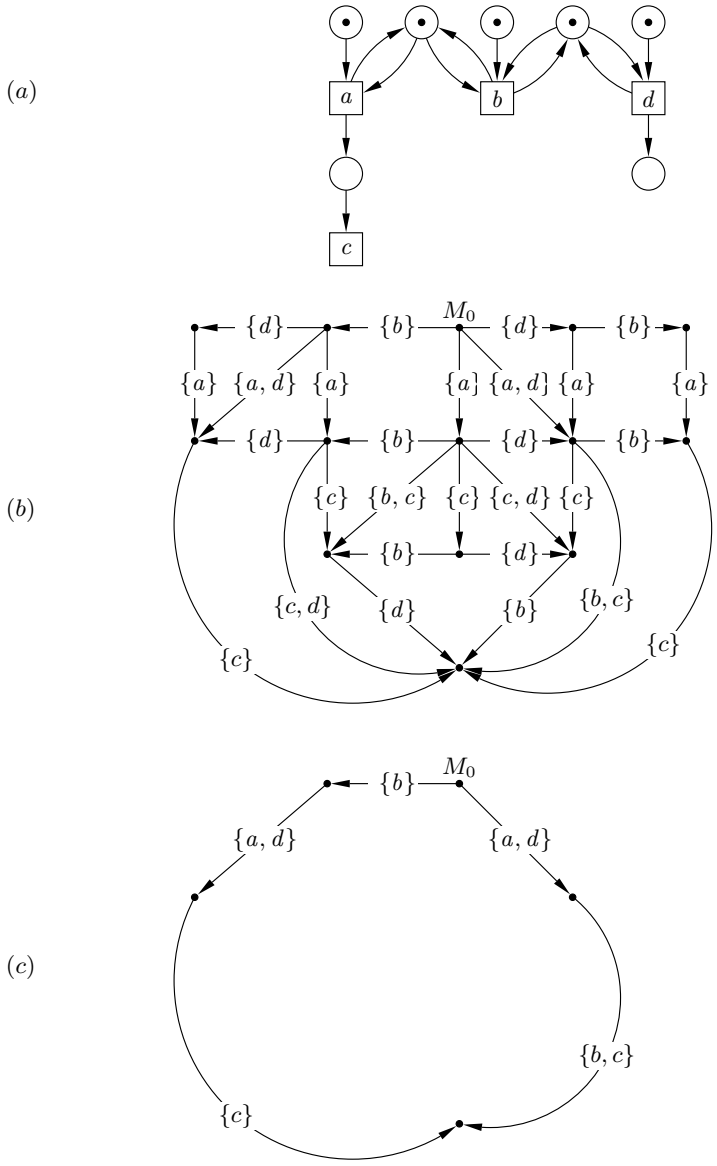
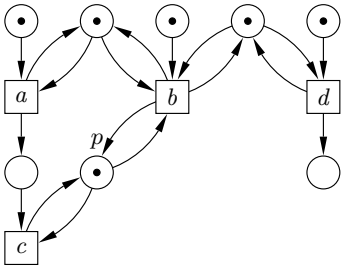
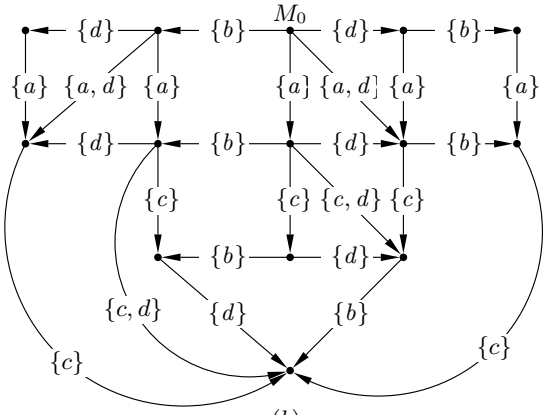


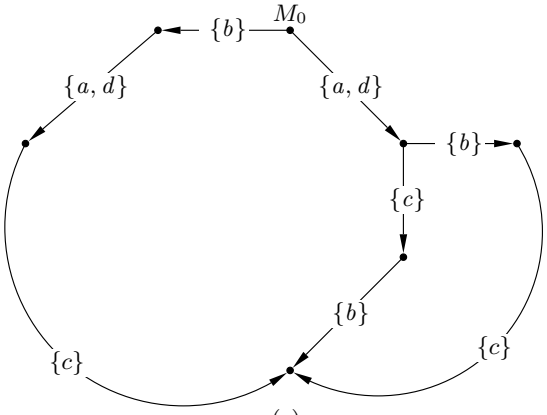
Fig. 12. A persistent and safe net $\mathcal{N}$ (a), its concurrent reachability graph $CRG(\mathcal{N})$ (b), and $CRG^{max}(\mathcal{N})$ obtained in the pruning procedure, which does not satisfy $CRG^{max}(\mathcal{N}) \preccurlyeq^{local}_{pers} CRG(\mathcal{N})$ (c).



(a)



(b)



(c)

Fig. 13. Fixing the problem in the example in Figure 12: persistent and safe net $\mathcal{N}' = \mathcal{N}_{b \leftrightarrow c}$ (a), its concurrent reachability graph $CRG(\mathcal{N}')$ (b), and $CRG^{max}(\mathcal{N}') \preccurlyeq^{local}_{pers} CRG(\mathcal{N}')$ obtained in the pruning procedure (c). The st-system in (c) satisfies as well $CRG^{max}(\mathcal{N}') \preccurlyeq^{global}_{pers} CRG(\mathcal{N}')$.