Timed Output Synchronized Petri Nets and basics of Synchronized State Class Graph

Mouna Gaouar, Rabah Ammour, Isabel Demongodin, Dimitri Lefebvre

Timed Output Synchronized Petri Nets and basics of Synchronized State Class Graph ⋆

Mouna Gaouar * **Rabah Ammour** * **Isabel Demongodin** *
Dimitri Lefebvre **

* *Aix-Marseille University, CNRS, LIS, Marseille, France, (email: {mouna.gaouar, rabah.ammour, isabel.demongodin}@lis-lab.fr)*
** *Université Le Havre Normandie, GREAH, Le Havre, France, (email: dimitri.lefebvre@univ-lehavre.fr)*

Abstract: This paper presents a timed extension of Synchronized Petri nets that can be controlled and observed. Output Synchronized Petri Nets have been previously defined for allowing a Cyber-Physical System to be controlled through input event associated with controllable transitions and to be observed thanks to output events issued by the marking values and/or marking changes. The proposed new timed extension, called timed Output Synchronized Petri nets, is obtained by assigning firing durations to transitions. As the state space of such formalism involves dense time variables, we propose a state class graph abstraction called Synchronized State Class Graph.

Keywords: Timed Petri net, Synchronized Petri net, State Class, Cyber-physical systems

1. INTRODUCTION

In recent years, there has been a growing focus on security of Cyber-Physical Systems (CPS) leading to the development of various formalisms and methods particularly within the framework of Discrete Event Systems (Oliveira et al., 2023). Combining computer networks with physical processes and controllers, such CPS are characterized by their complexity, communication capabilities and synchronous behaviors. In the Petri net (PN) framework, Synchronized Petri nets are a very suitable formalism for representing such a system as the enablings of transitions are governed by the occurrence of input events and a set of enabled transitions are fired at once (also called a maximal step in the PN literature). One extension of this formalism, called Output Synchronized Petri nets (OutSynPN) (Ammour et al., 2021) has been proposed for generating output events. Hence, the system can be controlled through input event associated with controllable transitions and could be observed thanks to output events issued by the marking values and/or marking changes. This formalism allows us to design observers and analyze vulnerability of CPS under stealthy actuators attacks (Ammour et al., 2022, 2023). However, there remains a crucial need to extend this formalism to temporal aspects, enabling the detection of cyber attacks based on timing conditions. This need motivates the new formalism proposed in this paper called Timed Output Synchronized Petri Net (TOutSynPN), which extends OutSynPNs with a firing duration associated with each transition of the net. Thanks to TOutSynPN formalism, the model is suitable to describe both the physical and cyber layers of CPS in an intuitive way, and to be consistent with most of the existing classes of Petri nets, as synchronized PNs, interpreted PNs, and of course, timed and time PNs. However, due to the density of the time domain, the state space of a TOutSynPN model is usually infinite. Hence, there is a need for contracting the infinite state space into a finite structure. In the time setting with interval, there exist several finite representations abstracting the state space of a time Petri net, one of them being the State Class Graph (SCG) (Berthomieu and Menasche, 1983). One of its benefits is to be finite as long as the time PN is bounded. Several extensions have been defined on such a SCG, as in (Leclercq et al., 2023) where it is proposed a state class based controller synthesis approach for time Petri nets or, the Modified SCG proposed in (Basile et al., 2015; He et al., 2019) which seems to be well adapted to describe in a compact and exhaustive manner all possible behaviors of labeled time Petri nets. This last SCG has also been used for time interpreted PNs (Basile and Ferrara, 2023) in which the firing of transitions could be conditioned by the occurrence of an input event. Inspired from these previous works, we propose in this paper a finite structure for TOutSynPN, called Synchronized State Class Graph (SynSCG). It allows the reachable marking to be represented while taking into account the semantics of TOutSynPN such as the enabling conditions related to the occurrence of input events as well as the synchronized firing of timed transitions. The practical aim of our work is to advance CPS cyber-security by offering a comprehensive time-dependent framework for modeling and detecting cyber threats.

The content of this paper is as follow. Section II is about the concepts and definitions of OutSynPNs. and, proposes the timed extension of this formalism. Basics of Synchronized State Class Graph are then presented in Section III. Section IV concludes the paper.

⋆ This work has been partially supported by the French National Research Agency under grant agreement ANR-22-CE10-0002.

2. TIMED OUTPUT SYNCHRONIZED PETRI NETS

After recalling some concepts about Output Synchronized Petri nets (see (Ammour et al., 2021, 2022) for more details), this section presents temporal extension added as deterministic (or constant) "firing durations" assigned to transitions of the net, qualifying the Timed OutSynPN as a particular class of timed-transition Petri nets.

2.1 Output Synchronized Petri Nets

To begin with, we denote by $\mathcal{B}_r$ the set of Boolean functions that can be defined on $\{0,1\}^r$. This allows us to define the output function that is marking dependent.

Definition 1. An *output synchronized Petri net* (OutSynPN) is a structure $N_{os} = \langle N, E, f, \Sigma, \Gamma, Q, g\rangle$. A marked OutSynPN is $\langle N_{os}, M_0\rangle$ such that:

- $N = \langle P, T, Pre, Post\rangle$ is a Petri net, where P is a set of m places, T is a set of n transitions, $Pre : P \times T \to \mathbb{N}$ and $Post : P \times T \to \mathbb{N}$ are the *pre*− and *post*− incidence matrices that specify the weights of directed arcs from places to transitions and vice versa.
- A *marking* is a vector $M : P \to \mathbb{N}^m$ that assigns to each place a non-negative integer. We denote by $M(p_i)$ or m_i, the marking of place p_i and, by M_0 an initial marking.
- E is an alphabet of external input events.
- $f : T \to E_\lambda = E \cup \lambda$ where λ is the always occurring event. f is an input function that associates with each transition $t \in T$ an event in E_λ.
- $\Sigma \subseteq \{\uparrow M(p_i), \downarrow M(p_i) \mid p_i \in P\}$ is a non empty set of events associated with a marking change of places, where $\downarrow$ and $\uparrow$ represent any decreasing and any increasing of a place marking, respectively;
- $\Gamma \subseteq \{M(p_i) \sim h, \mid p_i \in P, h \in \mathbb{N}, \ \sim \ \in \{=, \neq, \geq, \leq, >, <\}\}$ is a set of conditions on the place marking;
- Q is an alphabet of output events, and $Q_\varepsilon = Q \cup \varepsilon$ where ε is a silent output event;
- $g : Q \to \{0,1\}$ is an output function such that $\forall q_i \in Q$, $g(q_i) = \Upsilon(B_\Gamma(q_i)) \wedge \Theta(B_\Sigma(q_i))$ where $B_\Gamma(q_i) \in \mathcal{B}_{|\Gamma|}$, $B_\Sigma(q_i) \in \mathcal{B}_{|\Sigma|}$ and, $\Upsilon : \mathcal{B}_{|\Gamma|} \to \{0,1\}$ is a Boolean function depicting the conditions on the marking value of places to generate output q_i and $\Upsilon(.) = 1$ when no condition on the marking values is involved for output q_i; $\Theta : \mathcal{B}_{|\Sigma|} \to \{0,1\}$ is a Boolean function depicting the conditions on the marking change events to generate output q_i and $\Theta(.) = 0$ when no event on the marking change is involved for output q_i. ▲

$C = Post - Pre$ is the incidence matrix. The *preset* and *postset* of transitions are respectively: ${}^\bullet t = \{p \in P \mid Pre(p,t) > 0\}$ and $t^\bullet = \{p \in P \mid Post(p,t) > 0\}$. Two transitions, $t, t' \in T$, are said to be in *structural conflict* if they share at least one common input place, i.e., $({}^\bullet t \cap {}^\bullet t') \neq \emptyset$.

The set of transitions associated with an input event $e \in E_\lambda$ is denoted by: $T_e = \{t \in T : f(t) = e\}$. The set of transitions enabled at marking M and associated with input event e, called *receptive transitions set for event e at marking M*, is denoted by $\zeta_e(M) = \{t \in T_e : M \geq Pre(\cdot, t)\}$.

In a standard autonomous discrete Petri net, the behaviour is driven by asynchronous dynamics, meaning that only one transition is fired at a marking. The behaviour of an OutSynPN is defined by *synchronous dynamics in regards to transitions* and also by *asynchronous dynamics in regards to external input events*. In other terms, at marking M, only one external input event can occur at once, while several transitions can be fired simultaneously. In addition, at a marking, if the receptive transitions set for the always occurring event is not empty, the $\lambda-$input event prevails over any external event (called *choice policy* in the next). As a consequence, at the occurrence of event $e \in E_\lambda$, receptive transitions associated with e ($t_j \in \zeta_e(M)$), may be fired simultaneously in one step. This leads to the concept of *Elementary Firing Sequence (EFS)*, denoted as $\sigma_e(M) = [t_{j_1}, \cdots, t_{j_k}]$, where the brackets are used to denote that the firing order of the k transitions can be arbitrarily chosen. Hence, at marking M, the firing of an EFS $\sigma_e(M)$ yields to marking M', denoted as $M[\sigma_e(M)\ \rangle M'$, with $M' = M + C \cdot \tilde{\sigma}_e(M)$ where $\tilde{\sigma}_e(M) \in \mathbb{N}^{|T|}$ is *the EFS vector* of $\sigma_e(M)$.

When $M' \neq M$, this marking change can produce a set of output events $Q(M,e) = \{q \in Q : g(q) = 1\}$. If no output event is generated, it is noted as a silent output ε. Observe that this is always the case when $M = M'$, as there is no marking change. Finally, $M[\sigma_e(M)\ \rangle\ M' : Q(M,e)$, denotes that, from marking M, the occurrence of event e resulting in EFS σ_e, yields to marking M' and generates an output set $Q(M,e)$.

2.2 Timed Output Synchronized Petri Nets

Definition 2. A *timed output synchronized Petri net* (TOutSynPN) is a structure, $N_{tos} = \langle N_{os}, D\rangle$ such that:

- $N_{os} = \langle N, E, f, \Sigma, \Gamma, Q, g\rangle$ is an OutSynPN;
- $D : T \to \mathbb{Q}_0^+$ is a function that associates with each transition t_j a firing duration d_j. ▲

The *marking* M of a timed output synchronized Petri net $\langle N_{os}, D\rangle$ is a vector $M : P \to \mathbb{N}^m$, where m is the number of places in the net, that assigns to each place a non-negative integer. It represents the logical part of a state of a TOutSynPN.

For timing requirements, the service policy considered in this paper is *mono server semantics with respect to input events*. This means that no matter how many times an event successively occurs at a given state, only one occurrence will be considered. We also assume *mono-server semantics with respect to transitions*, that implies no transition can be enabled or activated more than once at the same marking. To represent in the time domain such a mono-server semantics, a single local clock is associated with each transition. Hence, the state of a TOutSynPN is defined as follows.

Definition 3. A *state* of a marked TOutSynPN $\langle N_{tos}, M_0\rangle$ is a pair $s = (M, \mathcal{O})$, composed of the marking of places and the value of all local clocks, $\mathcal{O} = \{o_t \in \mathbb{R}_0^+ \cup \{\sharp, +\infty\}, t \in T\}$, where $\sharp$ stands for a large positive rational number that can be assimilated to $+\infty$ in terms of properties. ▲

Inspired by the three-phases firing semantics previously proposed by Ramchandani and extended with zero firing delay by Popova-Zeugmann (2013), we consider for TOut-

SynPN a *preselection policy* as firing rules. When a timed transition is activated, it consists of reserving the input tokens of the transition, waiting until the firing duration is reached (delay) and next, deleting reserved tokens and creating the output tokens of the transition. This firing process when initiated cannot be interrupted or stopped. To represent such a preselection policy, the marking of a TOutSynPN is decomposed in two components.

Definition 4. The *marking* M of a timed output synchronized Petri net $\langle N_{tos}, M_0 \rangle$ is composed of a *reserved marking* M^R and a *non reserved marking* M^{NR}, such that: $M = M^R + M^{NR}$. ▲

We assume that there is no reserved marking at the initial marking, i.e., $M_0^{NR} = M_0$ and $M_0^R = 0$.

a) Enabling, activation, firing and local clock of a timed transition

The rules that govern the dynamics of a timed transition in a TOutSynPN differ from the OutSynPN in the sense that they depend on the non reserved marking.

Definition 5. At state $s = (M, \mathcal{O})$ with $M = M^R + M^{NR}$, a timed transition t is *enabled* if the non reserved marking satisfies the logical precondition, $M^{NR} \geq Pre(\cdot, t)$. ▲

Definition 6. At state $s = (M, \mathcal{O})$ with $M = M^R + M^{NR}$, an enabled timed transition t is *activated* when its associated event, $f(t) = e \in E_\lambda$ occurs. ▲

Due to the preselection policy, when transition t is activated, the marking needed for its firing is reserved: $M'^R = M^R + Pre(\cdot, t_j)$ and $M'^{NR} = M^{NR} - Pre(\cdot, t_j)$.

In (Ammour et al., 2021), we assume that the net is deterministic, meaning that the same input event cannot be shared by transitions in structural conflict. In the present paper we relax this assumption and define a synchronized conflict by an input event as follows.

Definition 7. Let $t, t' \in T$ be two transitions in structural conflict with $f(t) = f(t') = e \in E_\lambda$. At state $s = (M, \mathcal{O})$, transitions t and t' are in *synchronized conflict by input event* e if they are both enabled and event e occurs at state s with $(M^{NR} \geq Pre(.,t)) \wedge (M^{NR} \geq Pre(.,t')) \wedge (M^{NR} < Pre(.,t) + Pre(.,t'))$. ▲

Hence, in case of a synchronized conflict by an input event, both transitions could be activated whereas the marking cannot satisfy both firings at once. To handle such a situation, we assume that the conflict is externally resolved by a priority relation based on the priority function π which assigns to each transition $t \in T$ in structural conflict an integer $\pi(t) \in \mathbb{N}$, called the priority of t. Transition t is said to have priority over transition t', denoted as $t \gg t'$, if $\pi(t) > \pi(t')$. Moreover, it is assumed in the rest of this paper that two transitions t, t' in structural conflict do not have the same priority, i.e., $\pi(t) \neq \pi(t')$. Hence, the priority relation $\gg$ is assumed irreflexive, asymmetric and transitive. As a result, a priority-based conflict policy of a TOutSynPN is defined as follows.

Definition 8. Let t and t' be two transitions with $t \gg t'$. The *priority*-based conflict policy is defined such that, at state s, in case of synchronized conflict between t and t' by input event $f(t) = f(t')$, enabled transition t which has priority over t', is activated, whereas transition t' is disabled and, consequently, not activated. ▲

The previous conflict policy is adopted in the rest of this paper. Note that due to the mono-server policy, no transition can be enabled or activated more than once. Moreover, we assume an "as soon as possible" (asap) semantics (also called the maximal progress semantics) which means, for TOutSynPN, that enabled transitions are activated immediately at the occurrence of their input events, after applying the priority rule if necessary. With this asap semantics, the firing duration associated with a transition refers to the exact time required for the firing of a transition after its activation.

In order to track the state of a transition and the time elapsed between its activation and its firing, the possible values of its local clock are as follows.
(i) If t_j is not enabled, its local clock o_{t_j} is set on $\sharp$.
(ii) As soon as transition t_j is enabled, its local clock o_{t_j} is instantaneously set on $+\infty$, which denotes that it is receptive to its associated event $f(t_j)$.
(iii) When transition t_j is activated, its local clock instantaneously goes from $+\infty$ to d_j, it is set to its firing duration, i.e., $o_{t_j} = d_j$, we then say that the clock is *triggered*, and the marking needed for the activation of t_j is reserved.
(iv) As soon as t_j is activated, its clock o_{t_j} starts ticking down, and thus, $d_j \geq o_{t_j} \geq 0$. Its value represents the time left until the firing of t_j. When it reaches 0, transition t_j is fired, then the clock is reset to $\sharp$.

Note that, due to the preselection policy, when a transition is activated, it cannot be desactivated until its firing. In other terms, in a TOutSynPN with a preselection policy and a conflict policy as previously defined, all transitions are persistent, which means that their deactivation is only a consequence of their own firing.

b) Events leading the dynamics

The dynamics of a TOutSynPN is driven by two types of instantaneous events:
1- The external events are the input events in the set E. Their occurrence is one of the conditions for the activation of transitions and the triggering of local clocks. The always occurring event, λ, is also considered as an external event, although it is always occurring.
2- The internal events correspond to the firing of transitions whose local clock reaches zero.

The dynamics of a TOutSynPN is event driven meaning that, the occurrence of events drives the net from one state to another one. A change of state implies a change of the local clocks values and possibly a change in the marking values of the places.

c) Elementary sequences

Due to the timing aspects, we need to distinguish the activation of transitions from their firing. Thus, we introduce here two different elementary sequences: the elementary activating sequence and the timed elementary firing sequence. The former corresponds to an external event while the latter is dedicated to internal ones.

Enabling memory is used as memory policy with respect to newly activated transitions and to timed elementary firing sequence, meaning that the clock of the transitions not newly activated or fired, keep their values.

We extend the notion of the receptive transitions set of OutSynPNs to TOutSynPNs such that: $\zeta_e(s) = \{t \in T_e : o_t = +\infty\}$. It is defined as the set of all enabled transitions that can be activated by the occurrence of e. Suppose e occurs at s, transitions in $\zeta_e(s)$ which are not in synchronized conflict are activated. For those in synchronized conflict by input event e, the priority-based conflict policy is applied and thus, the prioritized transitions are activated.

Let us define by $\zeta_{\mathcal{A}}(s) = \{t_j \in T : o_{t_j} \notin \{\sharp, +\infty\}\}$ the *set of activated transitions in state s.*

An *Elementary Activating Sequence (EAS) of an input event* $e \in E_\lambda$, denoted by $\sigma_e(s)$, is here assimilated to the set of newly activated transitions when event e occurs. We still denote by $\tilde{\sigma}_e(s)$ the firing vector associated to $\sigma_e(s)$ of dimension $n = |T|$ such that $\tilde{\sigma}_e^j(s)$ corresponds to the number of times transition t_j appears in $\sigma_e(s)$. Note that, due to the mono-server semantics, transition t_j appears at most once in $\sigma_e(s)$, i.e., $\tilde{\sigma}_e^j(s) \in \{0,1\}, \forall t_j \in T$. Hence, when event e occurs at state $s = (M, \mathcal{O})$, an EAS $\sigma_e(s)$ leads to a new state, denoted by $s[\sigma_e(s)\rangle s'$, such that $s' = (M', \mathcal{O}')$ is given by:

- $M' = M$ with $\begin{cases} M'^R = M^R + Pre \cdot \tilde{\sigma}_e(s) \\ M'^{NR} = M^{NR} - Pre \cdot \tilde{\sigma}_e(s) \end{cases}$
- $o'_{t_j} = \begin{cases} d_j & \text{if } t_j \in \sigma_e(s) \\ o_{t_j} & \text{otherwise} \end{cases}$

Observe that, the activated transition set becomes: $\zeta_{\mathcal{A}}(s') = \zeta_{\mathcal{A}}(s) \cup \{t_j \in T : o'_t = d_j\}$.

Suppose now that at state s, a set of local clocks reach 0, i.e., $\{t_j \in \zeta_{\mathcal{A}}(s) : o_{t_j} = 0\}$. The transitions associated with those clocks are simultaneously fired at s through a *Timed Elementary Firing Sequence (TEFS)* noted $\sigma(s) = [t_{j1}, ..., t_{jk}]$. The firing of a TEFS implies that the clock of the fired transitions are reset (i.e., set on $\sharp$), whereas the clocks of other transitions keep their values. Hence, at state $s = (M, \mathcal{O})$, the firing of TEFS $\sigma(s)$ leads to a new state $s' = (M', \mathcal{O}')$, i.e., $s[\sigma(s) \rangle\ s'$, such that:

- $M' = M + C \cdot \tilde{\sigma}(s)$ with $\begin{cases} M'^R = M^R - Pre \cdot \tilde{\sigma}(s) \\ M'^{NR} = M^{NR} + Post \cdot \tilde{\sigma}(s) \end{cases}$
- $o'_{t_j} = \begin{cases} \sharp & \text{if } t_j \in \sigma(s) \\ o_{t_j} & \text{otherwise} \end{cases}$

The set of activated transition thus becomes: $\zeta_{\mathcal{A}}(s') = \zeta_{\mathcal{A}}(s) \setminus \{t_j \in \zeta_{\mathcal{A}}(s) : o_{t_j} = 0\}$.

Observe that the firing itself of a TEFS does not consume time and represents the multiple internal events occurrence. Moreover, compared to the logical setting, the transitions that form a TEFS are not necessarily activated by the same input event.

d) Multiple external events occurrence

In a Timed OutSynPN, multiple external events may occur simultaneously while the system is in state $s = (M, \mathcal{O})$. Let $\mathcal{E}(s)$ be the set of external input events $e \in E$ that occur at state s. For every event $e \in \mathcal{E}(s)$, if $\zeta_e(s) = \emptyset$, no clocks are triggered by it. Else, if $\zeta_e(s) \neq \emptyset$ thus there exists an elementary activating sequence associated with event e, whose transitions are necessarily fired after the elapsing of a given time. We can then define an *Elementary Activating Sequence (EAS) associated with a set of events* $\mathcal{E}(s)$ such that $\sigma_{\mathcal{E}(s)} = [\sigma_{e_i}(s) : e_i \in \mathcal{E}(s)]$. Note that if there exists a $\lambda-$receptive set at state s, the occurrence of any event in E does not affect the changing of state which is simply defined by $s[\sigma_\lambda(s)\rangle$. When events in $\mathcal{E}(s)$ occur, the system in state $s = (M, \mathcal{O})$ reaches, instantly, state $s' = (M', \mathcal{O}')$, i.e., $s[\sigma_{\mathcal{E}(s)} \rangle\ s'$, according to:

- $M' = M$ with $\begin{cases} M'^R = M^R + Pre \cdot (\sum\limits_{e_i \in \mathcal{E}(s)} \tilde{\sigma}_{e_i}(s)) \\ M'^{NR} = M^{NR} - Pre \cdot (\sum\limits_{e_i \in \mathcal{E}(s)} \tilde{\sigma}_{e_i}(s)) \end{cases}$
- $o'_{t_j} = \begin{cases} d_j & \text{if } f(t_j) \in \mathcal{E}(s) \text{ and } o_{t_j} = +\infty \\ o_{t_j} & \text{otherwise} \end{cases}$

We extend here Definition 7, synchronized conflict by an input event, to multiple external event occurrence. Let $t, t' \in T$ be two timed transitions in structural conflict, such that $f(t), f(t') \in E_\lambda$ with $f(t) \neq f(t')$. Enabled transitions t and t' are said to be in synchronized conflict if $f(t)$ and $f(t')$ occur simultaneously at state $s = (M, \mathcal{O})$ (i.e., $f(t), f(t') \in \mathcal{E}(s)$), such that $(M^{NR} \geq Pre(., t)) \wedge (M^{NR} \geq Pre(., t')) \wedge (M^{NR} < Pre(., t) + Pre(., t'))$. However, we assume in this paper that it is not possible for different events to occur simultaneously if they are associated with transitions in structural conflict, although these events may occur simultaneously with other events. This means that synchronized conflicts only occur for transitions associated with the same input event $e \in E_\lambda$. In that case, the previously priority-based conflict policy is applied meaning that, if $t \gg t'$, t is activated and added in EAS $\sigma_{\mathcal{E}(s)}$, whereas t' is disabled and $o'_{t'} = \sharp$, consequently.

e) Elapsing of time

The dynamics of a TOutSynPN is led by the occurrence of input events or the firing of transitions. If none occur however, clock values still change with the elapsing of time. Starting from state $s = (M, \mathcal{O})$, if no event $e \in E_\lambda$ occurs while $\zeta_e(M) \neq \emptyset$ and no activated transitions are fired (i.e., $\nexists o_{t_j} = 0$), after the elapsing of a time $\tau < min(\mathcal{O})$, a new state $s' = (M', \mathcal{O}')$ is reached, i.e., $s[\tau \rangle\ s'$, given by: $M' = M$ with $M'^{NR} = M^{NR}$, $M'^R = M^R$ and, $o'_{t_j} = \begin{cases} o_{t_j} - \tau & \text{if } o_{t_j} \in \mathbb{R}^+, \\ o_{t_j} & \text{otherwise.} \end{cases}$

f) Resume

The event driven continuous time dynamics of a TOutSynPN is governed by the occurrence of internal and external events, both of which instantaneously incur state changes. Between the occurrence of such events, delays attributed to the elapsing of time can be observed. It should be noted that a change in the marking can produce output events, as is the case with OutSynPN. To sum up, the state of the model evolves either by time progression or by applying of elementary sequences as below:

$s = (M, \mathcal{O}) \xrightarrow{\sigma(s), Q(M, \sigma(s))} s' = (M', \mathcal{O}')$: firing of a TEFS;

$s = (M, \mathcal{O}) \xrightarrow{\sigma_{\mathcal{E}(s)}} s' = (M, \mathcal{O}')$: applying of an EAS;

$s = (M, \mathcal{O}) \xrightarrow{\tau} s' = (M, \mathcal{O}')$: elapsing of time.

A fundamental difference between OutSynPN and TOutSynPN lies in the definition of a state, as for the timed extension, it is a pair that consists of not only the marking but also the values of the clocks. Although these values are in $\mathbb{R}_0^+$, the fact that external events can occur within infinite intervals $[0, +\infty[$, implies that the number of reachable states of a TOutSynPN is infinite.

3. SYNCHRONIZED STATE CLASS GRAPH

Due to the density of time and/or because the net may be unbounded, the state graph of a timed OutSynPN is infinite, and therefore, enumeration analysis of this model must go through state abstractions. We therefore propose in this paper, a state graph contraction, based on state class graphs (SCG) (Berthomieu and Menasche, 1983), that we call Synchronized State Class Graphs (SynSCG). This labeled digraph is composed by nodes (or classes) and arcs, where the temporal information is seen as activation and firing domains, instead of clock functions.

3.1 Structure of a SynSCG

Definition 9. A *synchronized state class* C is a set of states of the TOutSynPN that can be defined as a couple $C = (M, \Theta)$ such that:

- M is a reachable marking of the net.
- Θ is a domain defined by a set of linear inequalities, called *the firing and activation domains* of the transitions of the net. ▲

States belonging to the same class possess the same reserved and non-reserved marking, as they are only differentiated by the elapsing of time. The domain of a class $C = (M, \Theta)$ is defined by two sets of constraints, $\Theta = (\Theta^f, \Theta^a)$, as presented below.

- Θ^f is the set of *firing constraints* dedicated to the firing of transitions. It gives for each activated transition t_j, the set of delays after which the timing constraint of t_j is satisfied for its firing. It is a system of linear inequalities of the following forms, where variables θ^f_j are bijectively associated with activated transitions in class C.
  - $\forall t_j \in \zeta_{\mathcal{A}}(C) : \alpha_{t_j} \leq \theta^f_j \leq \beta_{t_j}$,
  - $\theta^f_j - \theta^f_{j'} \leq \gamma_{j,j'}$ for all transitions $t_j, t_{j'}$ that are not newly activated at C,

  with $\alpha_{t_j}, \beta_{t_j}, \gamma_{j,j'} \in \mathbb{R}^+_0$ and $\zeta_{\mathcal{A}}(C)$ stands for the set of activated transitions.
- Θ^a is the set of *activation constraints* of the class, dedicated to occurrences of input events. It gives for each possible external event e_i, the set of delays for which event e_i can occur from class C while affecting the state of the net (i.e., activate one or more enabled transitions). It is also described by a system of linear inequalities where variables θ^a_i are bijectively associated with external events in E.
  - $\forall e_i \in E$ with $\zeta_{e_i}(C) \neq \emptyset : 0 \leq \theta^a_i$,
  - For the always occurring event λ with $\zeta_\lambda(C) \neq \emptyset$ it holds: $0 \leq \theta^a_\lambda \leq 0$, i.e., $\theta^a_\lambda = 0$,

  where $\zeta_{e_i}(C)$ (resp. $\zeta_\lambda(C)$) stands for receptive transitions set for event e_i (resp. λ) defined by: $\zeta_{e_j}(C) = \{t \in T : t \notin \zeta_{\mathcal{A}}(C), M^{NR} \geq Pre(\cdot, t)\}$, $\forall e_j \in E_\lambda$.

The arcs linking two different classes C and C' correspond to: (i) either an occurrence of one input event (i.e., $e \in E_\lambda$) or, the occurrence of several internal events (i.e., a TEFS σ) with a set of generated outputs (i.e., $\{q \in Q_\varepsilon : g(q) = 1\}$); and, (ii) a time interval leading the dynamics of the model (i.e., $\Delta = [l, u]$ with $l, u \in \mathbb{R}^+_0 \cup \{+\infty\}$ and $l \leq u$).

3.2 Basics on SynSCG construction

Let $\langle \mathcal{N}, M_0 \rangle$ be a marked TOutSynPN, under preselection, single-server, choice and priority policies as seen before.

First of all, let us focus on the firing domain of transitions when an input event occurs. Suppose that at class C, transition t_k is newly activated, i.e., $\theta^f_k = \alpha_{t_k} = \beta_{t_k} = d_k \geq 0$. Assume that input event e_i occurs from C (i.e., $\zeta_{e_i}(C) \neq \emptyset$) within a time interval $[0, u]$ leading to class C'. The firing domain of t_k in class C' is then updated by: $max(0, d_k - u) \leq \theta'^f_k \leq d_k$. As the occurrence of input event e_i from C is only considered before the firing of t_k, it holds $u \leq d_k$ and, consequently, $(d_k - u) \leq \theta'^f_k \leq d_k$. Suppose now that several transitions are activated in class C, i.e., $\forall t_j \in \zeta_{\mathcal{A}}(C) \neq \emptyset$, $\alpha_{t_j} \leq \theta^f_j \leq \beta_{t_j}$. The occurrence of e_i in $[0, u]$ generates for those transitions in class C' a changing of their firing domain given by: $\alpha'_{t_j} \leq \theta'^f_j \leq \beta'_{t_j}$ with $\alpha'_{t_j} = max(0, (\alpha_{t_j} - u))$ and $\beta'_{t_j} = \beta_{t_j}$. However, the occurrence of e_i implies some other transitions to be newly activated. From $\zeta_{e_i}(C)$, and after applying the priority-based conflict policy if necessary, the elementary activating sequence of e_i, noted σ_{e_i}, is determined. The set of activated transitions is, consequently, increased in regards to σ_{e_i}, i.e., $\zeta_{\mathcal{A}}(C') = \zeta_{\mathcal{A}}(C) \cup \{t_j \in \sigma_{e_i}\}$, and activation domains of these newly activated transitions are initialized, i.e., $\forall t_j \in \sigma_{e_i}$, $\theta'^f_j = d_j$. In the meantime, the marking is reserved for their activation, i.e., $M'^R = M^R + Pre \cdot \tilde{\sigma}_{e_i}$ and $M'^{NR} = M^{NR} - Pre \cdot \tilde{\sigma}_{e_i}$, while the marking of C' remains equals to that of C, i.e., $M' = M$. Note that, the activation domain in C' is the one of C by removing the activation constraint of e_i.

Another specification of TOutSynPN is that a set of activated transitions are fired at once. Let us denote by σ this set of firings leading to class C'' from class C within a time interval $[u_\sigma, l_\sigma]$. In such a case, the marking of class C'' is directly given by the state equation: $M'' = M + C \cdot \tilde{\sigma}$ with $M''^R = M^R - Pre \cdot \tilde{\sigma}$ and $M''^{NR} = M^{NR} + Post \cdot \tilde{\sigma}$, while the set of activated transitions is reduced in regards to the fired transitions: $\zeta_{\mathcal{A}}(C'') = \zeta_{\mathcal{A}}(C) \setminus \{t \in \sigma\}$. The output events set of the input arc is thus determined by $Q(C, C'') = \{q \in Q_\varepsilon : g(q) = 1\}$. However, the new marking M'' may allow some other transitions to be receptive to input events occurrences. As a consequence, the receptive transitions sets are determined for class C'' by: $\forall e_j \in E_\lambda$ $\zeta_{e_j}(C'') = \{t \in T : t \notin \zeta_{\mathcal{A}}(C''), M''^{NR} \geq Pre(\cdot, t)\}$ and, the activation domain is defined by the following activation constraints: $0 \leq \theta^a_i, \forall e_i \in E$ such that $\zeta_{e_i}(C'') \neq \emptyset$, and $\theta^a_\lambda = 0$ if $\zeta_\lambda(C'') \neq \emptyset$. Concerning the firing domain, it is updated according to the rules of the SCG defined in Berthomieu and Menasche (1983), by observing that no transition can be newly activated in C''.

Concerning the arcs, from class $C = (M, \Theta)$, new reachable classes C' (or C'') are created by applying the following steps (when the associated conditions are fulfilled):
Step 1: determine internal events (if $\Theta^f \neq \emptyset$). First of all, the upper bound of any time interval in which an activated transition can be fired is given by: $u_\sigma = min_{t_k \in \zeta_{\mathcal{A}}(C)}(\beta_{t_k})$ if $\zeta_\lambda(C) = \emptyset$, or $u_\sigma = 0$ if $\zeta_\lambda(C) \neq \emptyset$. We define $T^f(C) = \{t_j \in \zeta_{\mathcal{A}}(C) : \alpha_{t_j} \leq u_\sigma\}$ as the set of all firable transitions from class C. To represent the synchronous

dynamics of TOutSynPN, a single arc is created for firable transitions that are certain to be fired at the same time, i.e., $\sigma = \{t_j \in T^f(C) : \alpha_{t_j} = \beta_{t_j} = u_\sigma\}$. This arc labeled by $(\sigma : Q(C, C'), \Delta_\sigma = [u_\sigma, u_\sigma])$ links class C to a new class C'. For each remaining firable transition $t_j \in T^f(C)$ and $t_j \notin \sigma$, an arc is added from class C to a new class C'^j and labeled by $(t_j : Q(C, C'^j), \Delta_j = [\alpha_{t_j}, u_\sigma])$.

Step 2: determine $\lambda-$event (if $\theta^a_\lambda = 0$). A new class C'' is created and an arc is added from C to C'' labeled by $(\lambda, \Delta = [0, 0])$. As $\lambda-$event prevails, all the other input events are disregarded and, consequently, no more arcs are created from class C.

Step 3: determine external events (if $\Theta^a \neq \emptyset \wedge \zeta_\lambda(C) = \emptyset \wedge min_{t_j \in \zeta_\mathcal{A}(C) \neq \emptyset}(\beta_{t_j}) \neq 0$). For each input event $e_i \in E$ such that $0 \leq \theta^a_i$ in class C, a class C'' is created and an arc from C to C'' is added, labeled by $(e_i, \Delta = [0, min_{t_j \in \zeta_\mathcal{A}(C)}(\beta_{t_j}, +\infty)])$.

According to the previous basics, the SynSCG of a TOutSynPN $\langle \mathcal{N}, M_0 \rangle$, can be constructed starting from $C_0 = (M_0, \Theta_0)$ where Θ_0 is given by: $\theta^a_\lambda = 0$ if $\zeta_\lambda(C_0) \neq \emptyset, 0 \leq \theta^a_i$ if $\zeta_{e_i}(C_0) \neq \emptyset \ \forall e_i \in E$, and $\Theta^f_0 = \emptyset$ with $\zeta_\mathcal{A}(C_0) = \emptyset$. In addition, if two classes in the graph are equivalent, it is necessary to merge them into one, keeping that two classes $C = (M, \Theta)$ and $C' = (M', \Theta')$ are equivalent if their markings are the same, $M = M'$ and $M^R = M'^R$, and their firing and activation domains have equal solution sets (see (Berthomieu and Menasche, 1983) for more details on classes equivalence).

Example 1. Consider the TOutSynPN in Figure 1, called $\mathcal{N}$ where $E = \{e_1, e_2\}$, $T_{e_1} = \{t_1, t_4\}, T_{e_2} = \{t_2, t_6\}$, $T_\lambda = \{t_3, t_5\}$, and $Q = \{A, B, C\}$. We assume $t_1 \gg t_4$, as both transitions are in structural conflict. The SynSCG (see Figure 2) of $\mathcal{N}$ is built starting from $C_0 = (M_0, \Theta_0)$ where $M_0 = (2p_1)$ (with $M^R_0 = 0$ and $M^{NR}_0 = (2p_1)$), and Θ_0 is defined by the inequality $0 \leq \theta^a_1$. From C_0, conditions of steps 1 and 2 are not fulfilled and only step 3 is applied since $\Theta^f = \emptyset$ and $\zeta_\lambda(C_0) = \emptyset$. It results in the creation of an arc labeled $(e_1, [0, +\infty[)$ leading to new class C_1, where transitions t_1 and t_4 are newly activated, implying $\theta^f_1 = 1$, $\theta^f_4 = 1$ and $M^R_1 = (2p_1)$. From C_1, step 1 implies a synchronized firing of $[t_1, t_4]$, as $u_\sigma = \min(1, 1) = \alpha_{t_1} = \alpha_{t_4} = \beta_{t_1} = \beta_{t_4} = 1$. Thus, an arc labeled $([t_1, t_4] : \{A\}, [1, 1])$ is added from C_1 to new class C_2. As marking of p_2 increases, this arc is labeled by output event A. In class C_2, as Θ_2 is defined by $\theta^a_\lambda = 0$ and $0 \leq \theta^a_2$, according to step 2, input event e_2 is disregarded and only one arc labeled $(\lambda, [0, 0])$ is added leading to new class C_3. From this class, despite the activation constraint for e_2, only one arc $([t_5] : \{B\}, [0, 0])$ is created by step 1 whereas step 3 is not applied as $\beta_{t_5} = 0$. And so on. Observe that from M_0, a temporal input/output sequence could be expressed by: (e_1) $(A, [1, 1])$ $(B, 0)$ (e_2) $(C, [3, 3])$ (e_1) $(A, [1, 1])$ (e_2) $(C, [3, 3])$.

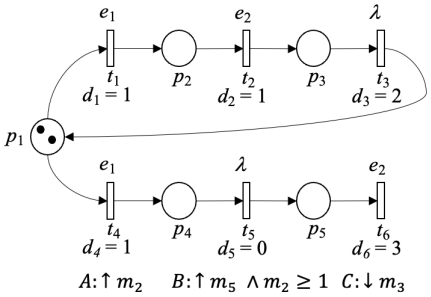

Fig. 1. TOutSynPN $\mathcal{N}$

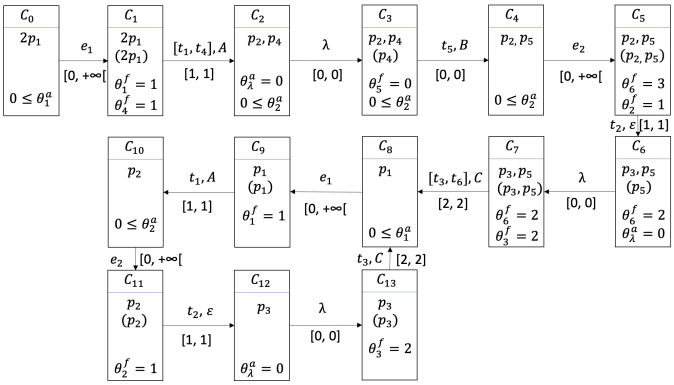

Fig. 2. SynSCG of TOutSynPN $\mathcal{N}$

4. CONCLUSIONS

In this paper we present a timed output synchronized Petri net formalism, that could be extended to time intervals but also to infinite server semantics, and its associated state class graphs, called Synchronized State Class Graph. In the challenging context of security of cyber-physical systems, we believe that timed OutSynPNs present a suitable framework for investigating this issue in a timed setting. The explicit information provided by the knowledge of inputs, the measurement of outputs and associated time stamps will be discussed and used to develop efficient detection schemes for cyber-attacks in CPSs.

REFERENCES

Ammour, R., Amari, S., Brenner, L., Demongodin, I., and Lefebvre, D. (2021). Observer design for bounded output synchronized Petri nets. In *European Control Conference (ECC)*, 746–751.

Ammour, R., Amari, S., Brenner, L., Demongodin, I., and Lefebvre, D. (2022). Observer design for labeled finite automata with inputs under stealthy actuators attacks. *IFAC-PapersOnLine (WODES)*, 55(28), 46–51.

Ammour, R., Amari, S., Brenner, L., Demongodin, I., and Lefebvre, D. (2023). Robust stealthy attacks based on uncertain costs and labeled finite automata with inputs. *IEEE Robotics and Automation Letters*, 8(5), 2732–2739.

Basile, F., Cabasino, M.P., and Seatzu, C. (2015). State estimation and fault diagnosis of labeled time Petri net systems with unobservable transitions. *IEEE Trans. on Automatic Control*, 60(4), 997–1009.

Basile, F. and Ferrara, L. (2023). Validation of industrial automation systems using a timed model of system requirements. *IEEE Trans. on Control Systems Technology*, 31(1), 130–143.

Berthomieu, B. and Menasche, M. (1983). An enumerative approach for analyzing time petri nets. In *IFIP*, 41–46. Elsevier Science Publishers.

He, Z., Li, Z., Giua, A., Basile, F., and Seatzu, C. (2019). Some remarks on "state estimation and fault diagnosis of labeled time Petri net systems with unobservable transitions". *IEEE Trans. on Automatic Control*, 64(12), 5253–5259.

Leclercq, L., Lime, D., and Roux, O.H. (2023). A state class based controller synthesis approach for time Petri nets. In *Petri Nets 2023*, volume 13929 of *LNCS*, 393–414.

Oliveira, S., Leal, A.B., Teixeira, M., and Lopes, Y.K. (2023). A classification of cybersecurity strategies in the context of discrete event systems. *Annual Reviews in Control*, 56, 100907.

Popova-Zeugmann, L. (2013). *Time and Petri Nets*. Springer.