The dynamics of deterministic systems – A survey

Luis M. Torres[1], Annegret K. Wagler[2]

luis.torres@epn.edu.ec, wagler@isima.fr

[1] Centro de Modelizatión Matemática (ModeMat)
Escuela Politécnica Nacional, Quito, Ecuador
[2] LIMOS (UMR 6158 CNRS)
University Blaise Pascal, Clermont-Ferrand, France

Abstract. We present a model for the dynamics of discrete deterministic systems, based on an extension of the Petri net framework. Our model relies on the definition of a priority relation between conflicting transitions, which is encoded in a compact manner by orienting the edges of a transition conflict graph. The benefit is that this allows the use of a successor oracle for the study of dynamic processes from a global point of view, independent from a particular initial state and the (complete) construction of the reachability graph. We provide a characterization, in terms of a local consistency condition, of those deterministic systems whose dynamic behavior can be encoded using our approach and consider the problem of recognizing when an orientation of the transition conflict graph is valid for this purpose . Finally, we address the problem of gaining the information that allows to provide an appropriate priority relation gouverning the dynamic behavior of the studied system and dicuss some further implications and generalizations of the studied approach.

1 Introduction

Petri nets constitute a well-established framework for modeling complex dynamic systems. Their broad application range includes the design of asynchronous hardware circuits [31], the analysis of production and workflow systems [2], the analysis and control of batch processes [9], the design of distributed algorithms for networks of agents [26], and the modeling and simulation of biological networks [10,11,21], to cite only some prominent examples.

Petri nets also turned out to be a flexible modeling framework that has been extended in various ways for dealing with different applications. For instance, colored and high-level Petri nets have been widely used for protocol specification in communication networks [5,14]. Stochastic Petri nets are used in cases where uncertainty is attached to input data, to describe external noise generated by the environment, as well as noise that might be intrinsic to a system [16,3]. Hybrid Petri nets allow to model systems were both continuous and discrete processes

M Heiner, AK Wagler (Eds.): BioPPN 2015, a satellite event of PETRI NETS 2015,
CEUR Workshop Proceedings Vol. 1373, 2015.

coexist [6], and the inclusion of additional features required for modeling certain biological networks has led to the definition of Hybrid Functional Petri nets [22]. In this paper we consider another possible extension that aims at representing systems whose dynamic behavior exhibits deterministic features.

More formally, a *network* $G = (P, T, A, w)$ reflects the involved components (like network elements, technical components, biological entities etc.) by places $p \in P$ and their interactions (like transformations, causal dependencies, chemical reactions etc.) by transitions $t \in T$, linked by weighted directed arcs. Each place $p \in P$ can be marked with an integral number of tokens defining a system state $x \in \mathbb{Z}_+^P$, and dynamic processes are represented by sequences of state changes, starting from an initial state x^0 and performed by consecutively switching or firing enabled transitions (see Section 2 for more details).

Usually, the dynamic behavior is the result of several conflicting as well as concurrent ongoing dynamic processes. A *Petri net* is a pair (G, x^0) consisting of a network G together with an initial state x^0, and its state space $\mathcal{X}(x^0)$ is understood as the set of all further system states which can be reached from x^0 by switching or firing sequences of transitions, see e.g. [25] for more information. In this framework, describing the dynamics of a system might be done by model animation, i.e., by simulating the flow of tokens inside the network as transitions are switched [12]. Given a network G and an initial state $x^0 \in \mathcal{X}$, some central problems that are usually studied in this context are:

- *Reachability*: Determine whether the system may eventually reach one of a set of target states after a finite sequence of transition switches.
- *Boundedness*: Determine if there are sequences of transition switches that lead to the accumulation of an unlimited number of tokens at some place of the network.
- *Existence of deadlocks*: Determine whether the system can reach a state at which no transitions are enabled.
- *Liveness*: Determine whether no sequence of transition switches can put the system into a state where some transition is permanently disabled.

The problems mentioned above are in general hard from a computational complexity point of view. For instance, *reachability* was proven to require exponential space [15] and decidability of this problem could only be established some years later [23,24].

Partial versus global point of view on a system. Dynamic processes can be encoded as directed paths in a state digraph $\mathcal{G}$ where nodes represent states and there is a directed arc between two states x, x' if x' can be obtained from x by switching a single transition. It is common practice to use the term *reachability* or *marking graph* to refer to the subgraph $\mathcal{G}(x^0)$ of $\mathcal{G}$ induced by the set $\mathcal{X}(x^0)$ of nodes corresponding to those states that can be reached from the initial state x^0 of the network.

Considering $\mathcal{G}(x^0)$ allows only a partial view on the studied system which is, e.g., suitable for a technical system performing exactly one process. However, already a failure of one or several components of such a system can change the

initial state, and a more global view is required to coherently model both the normal and the mal function of the system, and to detect the faulty element(s) by an according failure analysis.

Similarly, the study of biological systems by performing experiments can be seen as putting the system in a particular initial state and observing the evolution of the system in terms of resulting sequences of state changes. Due to the intrinsic complexity of biological systems, the dynamic behavior of such systems can rarely be understood by performing a single experiment. In general, several experiments starting from different initial states are required and need to be coherently interpreted in a single model.

To study dynamic systems and processes therein from a more global point of view (independent from a particular initial state), we therefore suggest to consider the state digraph $\mathcal{G}$ on the potential state space $\mathcal{X}$ of the system, i.e., on the set of all theoretically possible states.

Non-deterministic versus deterministic systems. While studying dynamic processes, a particular situation occurs when so-called dynamic conflicts are present at states, in which two transitions are enabled, but switching one disables the other. In this case, model animation does not allow definite conclusions about any system properties, as mentioned in [11]. The reason is that the occurrence of dynamic conflicts is understood as alternative (branching) system behavior, where a decision between these alternatives is taken non-deterministically.

However, there are examples of systems that show a deterministic behavior despite the existence of dynamic conflicts. We call a dynamic system *deterministic* if any state $x \in \mathcal{X}$ has a unique successor state $\mathrm{succ}(x)$. For technical systems, a deterministic behavior is often crucial in order to guarantee the reliability of the performed processes. In addition, also some biological systems are deterministic, as stimulating them in a certain way triggers always the same response (see e.g. the light-induced sporulation of *Physarum polycephalum plasmodia* or the phototaxis of halobacterial cells described in [18,17,19]). Petri nets, as originally defined, can be used to model such systems only in some trivial cases.

A compact encoding for deterministic systems. Our aim is to overcome the above mentioned difficulties by presenting a way to model deterministic systems from a global point of view (independent from a particular initial state), and to predict the systems behavior for all possible system states (whithout generating explicitely the state digraph). For that, we define a *successor function* $\mathsf{succ} : \mathcal{X} \to \mathcal{X}$ which returns $\mathrm{succ}(x)$ for every $x \in \mathcal{X}$. An explicit encoding of succ, for instance pointwise, is about of the same size as the state digraph $\mathcal{G}$, and hence at least exponential in the size of G. Here we propose a more compact way of representing the dynamics of a deterministic system, by encoding the global dynamic behavior of the system through a local selection criterion that chooses among all currently enabled transitions the one which switches to the successor state $\mathrm{succ}(x)$.

In the next section, we formally provide all definitions and concepts needed for Petri nets as models for deterministic systems. Our model relies on the defi-

nition of a priority relation between conflicting transitions, which is encoded in a compact manner by orienting the edges of a transition conflict graph. This is subject of Section 3, where the transition conflict graph is introduced together with suitable orientations of its edges in order to impose priorities among transitions as a selection criterion. This leads to a compact encoding of both a successor and a predecessor oracle. We also provide a characterization of the deterministic systems that can be encoded by our model. The issue of recognizing and characterizing such suitable orientations of the transition conflict graph is addressed in Section 4, while Section 5 is devoted to the problem of finding such orientations. We finally summarize all proposed concepts and their benefits and discuss some further implications and generalizations of the studied approach.

2 Petri nets with a deterministic behavior

In this section, we will formally provide all definitions and concepts needed for Petri nets as models for systems with a deterministic behavior.

Recall that a *network* $G = (P, T, A, w)$ is used to encode the structure of the underlying system, where the set P of places represents the system's components, the set T of transitions stands for their interactions, and the weighted directed arcs (p, t) or $(t, p) \in A$ exclusively connect places and transitions. A network $G = (P, T, A, w)$ can also be represented by its *incidence matrix* $M := (m_{pt}) \in \mathbb{Z}^{P \times T}$ where each row corresponds to a place $p \in P$ and each column to a transition $t \in T$. We have

$$m_{pt} := \begin{cases} -w_{pt} & \text{if } p \in P^-(t), \\ +w_{tp} & \text{if } p \in P^+(t), \\ 0 & \text{otherwise,} \end{cases}$$

where $P^-(t) := \{p \in P : (p, t) \in A\}$, denotes the set of *pre-places* of $t \in T$ and $P^+(t) := \{p \in P : (t, p) \in A\}$ the set of its *post-places*.

Some places $B \subseteq P$ may have bounded capacities, which are given by the positive integral vector $u \in \mathbb{Z}_+^B$. Each place $p \in P$ can be marked with an integral number x_p of (at most u_p) tokens, and any marking defines a state of the system that can be represented as an integral nonnegative vector $x \in \mathbb{Z}_+^P$. The *potential state space* of a capacitated network (G, u) is the set of all theoretically possible states $\mathcal{X} := \left\{x \in \mathbb{Z}_+^P : x_p \leq u_p,\ \forall p \in B\right\}$. It is at least exponential in $|P|$ and is finite if $B = P$ holds.

Dynamic processes are described as sequences $x^1, \ldots, x^k$ of consecutive system states[3], where state x^{i+1} is obtained from x^i by *switching* or *firing* a (single) transition $t \in T$. Thereby, t consumes w_{pt} tokens from each pre-place $p \in P^-(t)$ and produces w_{tp} new tokens on each post-place $p \in P^+(t)$. A transition $t \in T$ is *enabled* at a state $x \in \mathcal{X}$ if switching t yields a valid successor state $x + M_{\cdot t} \in \mathcal{X}$, where $M_{\cdot t}$ is the column of M associated with t, and *disabled* otherwise. Thus, $t \in T$ is enabled in $x \in \mathcal{X}$ if

[3] Throughout this paper, we will use superindices to reference different states and subindices to specify places. Thus, x_p^i is the number of tokens assigned to place p at state x^i.

E1 $x_p - w_{pt} \geq 0$ for all $p \in P^-(t)$ and
E2 $x_p + w_{tp} \leq u_p$ for all $p \in P^+(t) \cap B$.

Remark 1. An extended Petri net $(P, T, (A \cup A_R \cup A_I), w)$ is a Petri net which has, besides the (standard) arcs in A, two additional sets of so-called control-arcs: the set of read-arcs $A_R \subset P \times T$ and the set of inhibitor-arcs $A_I \subset P \times T$. In such a Petri net, switching of transitions is also controlled by read- and inhibitor-arcs: a transition $t \in T$ is enabled at $x \in \mathcal{X}$ if E1 and E2 are satisfied and it additionally holds that

E3 $x_p \geq w_{pt}$ for all p with $(p, t) \in A_R$, and
E4 $x_p < w_{tp}$ for all p with $(p, t) \in A_I$.

Note that also in an extended Petri net, switching of an enabled transition t at x yields $x + M_{\cdot t}$ as succesor state since control-arcs do not affect the marking.

In general, it is possible that more than one transition is enabled in a state $x \in \mathcal{X}$; we denote by $T(x)$ the set of all such transitions. Conversely,

$$X(t) := \left\{ x \in \mathcal{X} : x_p \geq w_{pt},\, \forall p \in P^-(t);\;\; x_p \leq u_p - w_{tp},\, \forall p \in P^+(t) \cap B \right\}$$

denotes the set of states at which transition t is enabled.

Recall that the state digraph of a system is defined by $\mathcal{G} := (\mathcal{X}, \mathcal{A})$ where nodes represent potential states and a node x has an outgoing arc $(x, x') \in \mathcal{A}$ if and only if $x' = x + M_{\cdot t}$ for some $t \in T(x)$, i.e., if x' can be obtained from x by switching a single transition. We call $x \in \mathcal{X}$ a *branching state* if there are at least two transitions in $T(x)$. The existence of a branching state implies either an alternative or a concurrent behavior of the system.

An alternative behavior occurs when at least two transitions $t, t' \in T(x)$ are in *dynamic conflict*, i.e., if they cannot switch simultaneously as for some place $p \in P$, we have

$$x_p - w_{pt} - w_{pt'} < 0, \quad \text{or} \quad x_p + w_{tp} + w_{t'p} > u_p,$$

thus, if $x + M_{\cdot t} + M_{\cdot t'} \notin \mathcal{X}$ holds. While animating the model, *either* t *or* t' has to be selected at x. In the case of concurrency, on the contrary, no two transitions from $T(x)$ are in a dynamic conflict. All transitions in $T(x)$ could be switched simultaneously, resulting in the valid state

$$x' = x + \sum_{t \in T(x)} M_{\cdot t} \;\in \mathcal{X}.$$

In this case, $\mathcal{G}$ contains paths for all possible interleaving sequences between x and x', corresponding to all possible permutations of the transitions in $T(x)$.

Petri nets are an adequate model for simulating the dynamic behavior of deterministic systems, only if no branching states are present. Otherwise, a mechanism has to be specified that allows to decide "which is the right path to choose" in the state digraph. This includes in particular a way of resolving dynamic conflicts.

We call a Petri net *deterministic* if each state $x \in \mathcal{X}$ has a unique successor $\mathrm{succ}(x) \in \mathcal{X}$. The dynamic behavior of a deterministic system can be specified in the form of a *successor-oracle* which returns, for every state $x \in \mathcal{X}$, the value of $\mathrm{succ}(x)$. Our goal is to propose a compact implementation for this oracle. Hereby, we restrict our attention to systems where the successor-oracle can be expressed via a *transition selection function* $\mathsf{trans} : \mathcal{X} \to T$, which assigns to every state $x \in \mathcal{X}$ a unique transition $\mathsf{trans}(x) \in T(x)$ that must be switched in order to reach state $\mathrm{succ}(x)$. Not all deterministic systems can be modeled in this way, see the discussion in Section 6, and even if it is possible, an explicit (i.e., pointwise) encoding of trans is extensive as it requires an amount of space proportional to the size of the state digraph $\mathcal{G}$, which, as pointed out in the previous section, is at least exponential in the number of places of the network.

In [20], it was proposed to use priorities between the transitions of the network as additional activation rules to determine which transition from $T(x)$ has to be selected as $\mathsf{trans}(x)$ in order to reach $\mathrm{succ}(x)$. Our contribution consists in providing a compact scheme for encoding trans based on such priorities.

Remark 2. The priority relations proposed in [20] shall reflect the relative reaction rates of the (chemical) reactions represented by the transitions of the network with the idea that faster reactions have higher priorities and are taken. On the model side, priorities can be seen as a discrete extreme case of firing rates of transitions defined by probability distributions, where exactly one transition in $T(x)$ (namely, the highest-priority transition $\mathsf{trans}(x)$) has probability 1, and all other transitions in $T(x)$ have probability 0.

As we show in the next section, if a deterministic system $\mathcal{S} = (G, u, \mathsf{succ})$ satisfies a quite natural consistency condition, then there exists a digraph $\mathbb{D} := (T, \mathbb{A})$ on the set of transitions with the following properties:

- for every $x \in \mathcal{X}$, the set $T(x)$ induces a subgraph having a unique sink $t(x)$
- and $t(x) = \mathsf{trans}(x)$ is the required transition to switch from x to $\mathrm{succ}(x)$.

Observe that this graph has a size of $O(|T|^2)$, which is polynomial in the size of G. Indeed, the arcs of $\mathbb{D}$ can be interpreted as priority relations between pairs of transitions, with $\mathsf{trans}(x)$ being the transition with highest priority among all transitions in $T(x)$.

3 Encoding valid priority relations

As observed in the previous section, one key issue for modeling the dynamic behavior of a deterministic system is the specification of a mechanism for the (unambiguous) resolution of dynamic conflicts between enabled transitions.

Definition 1. *Given a capacitated network $G = (P, T, A, w)$ with $u \in \mathbb{Z}_+^B$, its* transition conflict graph *is an undirected graph $\mathbb{K}_{(G,u)} = (T, \mathbb{E})$ having as nodes the transitions from G, where two transitions t, t' are joined by an edge if and only if there exists at least one state where both are enabled, i.e.,*

$$tt' \in \mathbb{E} \quad \Leftrightarrow \quad X(t) \cap X(t') \neq \emptyset.$$

As long as there is no risk of confusion, we will simply write $\mathbb{K}$ instead of $\mathbb{K}_{(G,u)}$. It follows straightforwardly from this definition that, for every state $x \in \mathcal{X}$, the set $T(x)$ of enabled transitions induces a clique in $\mathbb{K}$, i.e., a set of mutually adjacent nodes.

Remark 3. The converse is not necessarily true, as it is shown in [29]. However, as the sets $X(t)$ are boxes, it is straightforward to prove that at least the inclusionwise maximal cliques in $\mathbb{K}$ are associated with states of the system.

The transition conflict graph $\mathbb{K}$ can be constructed in $O(|P|\ |T|^2)$ time using a straightforward algorithm to check for box intersections $X(t) \cap X(t')$. The next lemma from [29] provides a more efficient way of its computation.

Lemma 1. *Consider a capacitated network (G, u), with $G = (P, T, A, w)$, $u \in \mathbb{Z}_+^B$ and $X(t) \neq \emptyset\ \forall t \in T$. Then $t, t' \in T$ are in conflict if and only if*

$$w_{pt} \leq u_p - w_{t'p} \quad \forall p \in P^-(t) \cap P^+(t') \cap B \text{ and}$$
$$w_{pt'} \leq u_p - w_{tp} \quad \forall p \in P^-(t') \cap P^+(t) \cap B \text{ holds.}$$

Observe that, in the particular case where we have capacities $u = \mathbb{1}$, it follows as a corollary from the last lemma that two transitions $t, t' \in T$ are in conflict if and only if

$$P^-(t) \cap P^+(t') \cap B = \emptyset \quad \text{and} \quad P^-(t') \cap P^+(t) \cap B = \emptyset. \tag{1}$$

The following example illustrates the previous results.

Example 1. Consider the capacitated network $(G, \mathbb{1})$, with $G = (P \cup T, A, \mathbb{1})$, from Figure 1(a). Checking the pre- and post-places for all transitions in T yields:

t_i	$P^-(t_i)$	$P^+(t_i)$
t_1	$\{2\}$	$\{1\}$
t_2	$\{2\}$	$\{3\}$
t_3	$\{2\}$	$\{5\}$
t_4	$\{5\}$	$\{1, 4\}$
t_5	$\{2, 5\}$	$\{3\}$

According to (1), all pairs of transitions are in conflict, except (t_3, t_4) and (t_3, t_5), and we obtain the transition conflict graph shown in Figure 1(b).

As stated in the introduction, our main purpose is to encode the dynamics of a given deterministic system $\mathcal{S} = (G, u, \mathsf{trans})$ in a compact way, by introducing enough priority relations among transitions, as to be able to determine $\mathsf{trans}(x)$ for every $x \in \mathcal{X}$. We have observed that the set of enabled transitions at each state corresponds to a clique in the transition conflict graph $\mathbb{K}$, and hence $\mathbb{K}$ is a plausible candidate for a framework where these priorities could be embedded. This idea motivates the following definition.

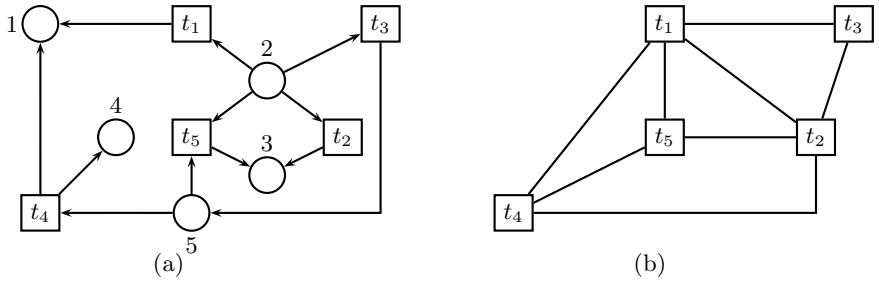

Fig. 1. (a) A capacitated network of a dynamic system, and (b) its transition conflict graph. All place capacities and arc weights are assumed to be equal to one.

Definition 2. *Let* $\mathbb{D}_{(G,u)} = (T, \mathbb{A})$ *be a directed graph obtained by orienting the edges of* $\mathbb{K}_{(G,u)}$. $\mathbb{D}_{(G,u)}$ *is* valid for the system $\mathcal{S} = (G, u, \mathsf{trans})$ *if, for every state* $x \in \mathcal{X}$ *with* $T(x) \neq \emptyset$*, the subgraph induced by the nodes from* $T(x)$ *has a unique sink, and this sink coincides with* $\mathsf{trans}(x)$.

As for the unoriented transition conflict graph, we will write in the following $\mathbb{D}$ instead of $\mathbb{D}_{(G,u)}$ whenever there is no risk of confusion. Observe that the existence of a valid orientation implicitly imposes a further requirement on the nature of a dynamic system. Namely, if two states $x, x^* \in \mathcal{X}$ have the same set $T(x) = T(x^*)$ of enabled transitions, then both induce the same subgraph of $\mathbb{D}$ and, therefore, $\mathsf{trans}(x) = \mathsf{trans}(x^*)$ must hold. Moreover, if a transition t is enabled at two states $x, x' \in \mathcal{X}$ and t is the highest-priority transition at x, then either t is also the highest-priority transition at x' or $\mathsf{trans}(x') \notin T(x)$. The next result from [29] shows that this condition is also sufficient for the existence of a valid orientation.

Theorem 1. *A system* $\mathcal{S} = (G, u, \mathsf{trans})$ *has a valid orientation if and only if the following* consistency condition *holds for any pair of states* x *and* x'*: if* $\mathsf{trans}(x) \in T(x) \cap T(x')$ *then* either $\mathsf{trans}(x') = \mathsf{trans}(x)$ or $\mathsf{trans}(x') \in T(x') \setminus T(x)$.

As illustrated in Algorithm 1, a valid orientation completely encodes the dynamic behavior of a deterministic system, since it can be used to obtain an implicit implementation of the successor-oracle. Given a state $x \in \mathcal{X}$, all we need to do in order to find its successor is to determine the highest-priority transition $\mathsf{trans}(x)$ at x. This can be accomplished by searching for the unique sink in the subgraph of $\mathbb{D}$ induced by the set $T(x)$ of enabled transitions. (If no transitions are enabled at x, the algorithm just returns the same current state x as successor.)

A *predecessor* of x is a state $y \in \mathcal{X}$ with $\mathrm{succ}(y) = x$. For that, we must have $x = y + M_{\cdot \mathsf{trans}(y)}$. Since there are at most $|T|$ candidates for the transition $\mathsf{trans}(y)$, it follows that the set $\mathrm{pred}(x)$ of all possible predecessors of x has cardinality not larger than $|T|$ and can be constructed calling the successor oracle at most $|T|$ times, as shown in Algorithm 2. Here, first a set $\mathrm{cand}(x)$ of

Input: $(G, u, \mathbb{D})$, $x \in \mathcal{X}$
Output: succ(x)
Construct the set $T(x)$ of enabled transitions
if $T(x) = \emptyset$ **then**
  return x
end if
{Compute out-degree of transitions in the subgraph of $\mathbb{D}$ induced by $T(x)$}
for $t \in T(x)$ **do**
  $\delta^-(t) := |\{tt' \in \mathbb{A} : t' \in T(x)\}|$
end for
{Return successor state}
$t^* \leftarrow t \in T(x)$ with $\delta^-(t) = 0$
return $x + M_{\cdot t^*}$

Algorithm 1: Implementing a successor-oracle by a valid orientation.

all candidate transitions $t \in T$ fulfilling $y := x - M_{\cdot t} \in \mathcal{X}$ is computed, then the condition $t = \mathsf{trans}(y)$ tested for each candidate by calling the successor oracle.

Input: $(G, u, \mathbb{D})$, $x \in \mathcal{X}$
Output: pred(x)
{Construct set of transition candidates}
cand$(x) \leftarrow \left\{t \in T : x_p + w_{pt} \leq u_p, \forall p \in P^-(t) \cap B;\ \ x_p - w_{tp} \geq 0, \forall p \in P^+(t)\right\}$
{Call successor-oracle to construct sets of predecessors}
pred$(x) \leftarrow \emptyset$
for $t \in$ cand(x) **do**
  $y \leftarrow x - M_{\cdot t}$
  if $\mathsf{trans}(y) = t$ **then**
    pred$(x) \leftarrow$ pred$(x) \cup \{y\}$
  end if
end for
{Return set of possible predecessors}
return pred(x)

Algorithm 2: An implementation for a predecessor-oracle.

Computing successors and sets of predecessors is a key operation within simulation algorithms to study the dynamic behavior of a system and in particular to address the different questions described in the previous section. To the best of our knowledge, currently no solution algorithm for these problems is known, which does not rely on storing explicit descriptions of the state digraph or some equivalent structure to compute succ(x) at every state x, which strongly limits the size of the networks that can be considered, due to the high memory requirements. Moreover, in certain application areas such as systems biology, the explicit values of succ(x) can only be determined by carrying out expensive and time-consuming wet lab experiments. Having, however, a valid orientation it is possible to determine or predict succ(x) for each state $x \in \mathcal{X}$.

4 Recognizing and characterizing valid orientations

Another issue of interest concerns the *recognition* of valid orientations: Given a capacitated network (G, u), and an orientation $\mathbb{D}$ of the edges in the transition conflict graph, determine whether $\mathbb{D}$ induces a valid dynamic behavior on G, in the sense that for every state $x \in \mathcal{X}$, the corresponding clique in $\mathbb{D}$ has a unique sink. Observe that a clique cannot have more than one sink, and that every clique without a sink contains a directed cycle. Hence, if $\mathbb{D}$ is acyclic then every clique of the graph has a unique sink, and we immediately obtain:

Observation 1 *Every acyclic orientation of* $\mathbb{K}$ *is valid.*

On the other hand, the next example shows that there are deterministic systems for which the valid orientation encoding the dynamic behavior contains directed cycles.

Example 2. Figure 2 depicts a network $(G, \mathbb{1})$ with $w = \mathbb{1}$ together with a valid orientation $\mathbb{D}$ of the transition conflict graph that contains a directed cycle $C = (t_1, t_2, t_3)$. Table 1 lists all branching states of the system and the corresponding sets of enabled transitions. It is straightforward to check that each of these sets induces a clique in $\mathbb{D}$ with a unique sink $t(x)$.

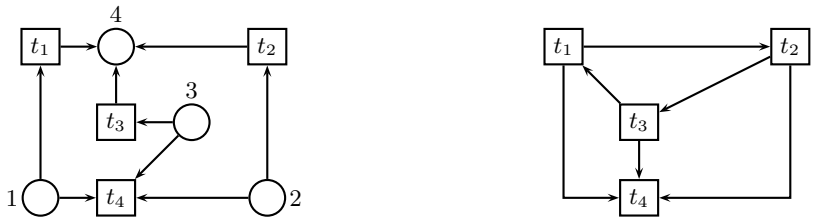

Fig. 2. A capacitated network and a valid orientation for its transition conflict graph which contains a directed cycle $C = (t_1, t_2, t_3)$. All arc weights and capacities of the places are assumed to be equal to one.

Table 1. All branching states of the system from Figure 2.

x_1	x_2	x_3	x_4	$T(x)$	$t(x)$
1	1	0	0	$\{t_1, t_2\}$	t_2
0	1	1	0	$\{t_2, t_3\}$	t_3
1	0	1	0	$\{t_1, t_3\}$	t_1
1	1	1	0	$\{t_1, t_2, t_3, t_4\}$	t_4

The fact that one has to check, for each clique Q in $\mathbb{K}$, if there is a corresponding state $x \in \mathcal{X}$ with $T(x) = Q$ makes the recognition of valid orientations hard in general, as there might be up to $2^{|T|}$ cliques. In the following we describe a special class of orientations for which this recognition problem can be solved efficiently.

In [29], it is shown that two different dynamic systems might share the same transition conflict graph. This motivated the definition of the following equivalence relation between capacitated networks

$$(G, u) \sim_{\mathbb{K}} (G', u') \quad \Leftrightarrow \quad \mathbb{K}_{(G,u)} \cong \mathbb{K}_{(G',u')},$$

where $\cong$ stands for graph isomorphism. An orientation is *strongly valid* if it is valid for all networks of some equivalence class of $\sim_{\mathbb{K}}$. Acyclic orientations are one specific example of strongly valid orientations. In general, any orientation where *every* clique has a unique sink is (trivially) strongly valid. Moreover, the converse is also true, as it follows from a result in [29] that each equivalence class contains a network with the property that every clique in $\mathbb{K}$ is associated to a state. This fact makes strongly valid orientations easy to recognize.

Theorem 2. *An orientation $\mathbb{D}$ of the transition conflict graph of a capacitated network (G, u) is strongly valid if and only if it does not contain any directed cycle of length 3.*

In this context, one is tempted to wonder whether it is possible to gain more insights on (strongly) valid orientations by exploring structural properties of $\mathbb{K}$. In [29], it is shown that for any undirected graph H, there is a network G having H as its transition conflict graph. Thus, in general, neither $\mathbb{K}$ nor $\mathbb{D}$ can admit particular graph-theoretical properties. Studying the problem however from a hypergraph-theoretical point of view as done in [30], opens the possibility of characterizing valid orientations further.

A hypergraph is a generalization of a graph in which it is possible to connect any number of nodes. Formally, a hypergraph is a pair $\mathcal{H} = (V, \mathcal{E})$ where V is a set of elements, called nodes or vertices, and $\mathcal{E}$ is a family of non-empty subsets of V, called hyperedges. We further use two other combinatorial structures related to hypergraphs. The dual $\mathcal{H}^*$ of $\mathcal{H}$ is a hypergraph whose nodes and hyperedges are interchanged, so that the nodes are given by all $E_i \in \mathcal{E}$ and there is one hyperedge $V_v = \{E_j | v \in E_j\}$ for each $v \in V$. The intersection graph $G(\mathcal{H})$ of $\mathcal{H}$ is a graph whose nodes represent the hyperedges of $\mathcal{H}$, and two nodes are joined by an edge if and only if the corresponding hyperedges intersect.

Recall that, by definition, for every state $x \in \mathcal{X}$, the set $T(x)$ of enabled transitions induces a clique in $\mathbb{K}$, whereas the converse is not necessarily true (but at least the inclusion-wise maximal cliques in $\mathbb{K}$ are associated with states of the system), see Remark 2.

The equivalence relation on $\mathcal{X}$, where $x \sim x'$ iff $T(x) = T(x')$, partitions the state space into $r \leq 2^{|T|}$ equivalence classes $\mathcal{X}_1, \ldots, \mathcal{X}_r$ of states that share the same sets of enabled transitions. Let $\tilde{x}^i \in \mathcal{X}_i$ with $1 \leq i \leq r$ be (arbitrarily chosen) representative elements for each of these classes, and define the *transition*

hypergraph $\mathcal{H}_T = (T, \mathcal{E}_T)$ *of* G to be the hypergraph on the set T of transitions, whose family $\mathcal{E}_T$ of hyperedges is given by $\mathcal{E}_T = \left\{T(\tilde{x}^i) : 1 \leq i \leq r\right\}$. Thus, the size of $\mathcal{H}_T$ is exponential in the number of transitions of the network, but it is always finite, as there are at most $2^{|T|}$ different subsets of T.

The *state hypergraph of* G is the dual hypergraph $\mathcal{H}_{\tilde{\mathcal{X}}}$ of $\mathcal{H}_T$ and has $\tilde{\mathcal{X}} = \left\{\tilde{x}^1, \ldots, \tilde{x}^r\right\}$ as node set, where its family of hyperedges is determined by $\mathcal{E}_{\tilde{\mathcal{X}}} = \left\{X(t) \cap \tilde{\mathcal{X}} : t \in T\right\}$. Directly from the definition of $\mathcal{H}_{\tilde{\mathcal{X}}}$ we deduce:

Lemma 2. *The intersection graph* $G(\mathcal{H}_{\tilde{\mathcal{X}}})$ *of the state hypergraph* $\mathcal{H}_{\tilde{\mathcal{X}}}$ *is exactly the transition conflict graph* $\mathbb{K}$*.*

A hypergraph $\mathcal{H} = (V, \mathcal{E})$ has the *Helly property* if, for any family $\mathcal{E}' \subseteq \mathcal{E}$ of pairwise intersecting hyperedges, there exists a node $v \in V$ contained in all hyperedges from $\mathcal{E}'$. A well-known result in hypergraph theory [4] states that $\mathcal{H}$ has the Helly property if and only if the inclusion-wise maximal hyperedges of its dual hypergraph $\mathcal{H}^*$ are precisely the inclusion-wise maximal cliques of the intersection graph $G(\mathcal{H})$ of $\mathcal{H}$ [4]. This implies:

Lemma 3. $\mathcal{H}_{\tilde{\mathcal{X}}}$ *satisfies the Helly property.*

A direct consequence from Lemma 3 and the preceding observations is obtained in [30]:

Theorem 3. *The inclusion-wise maximal hyperedges from* $\mathcal{E}_T$ *are exactly the inclusion-wise maximal cliques of* $\mathbb{K}$*. Thus, for every inclusion-wise maximal clique* Q *there is some state* $\tilde{x}^i \in \tilde{\mathcal{X}}$*,* $1 \leq i \leq r$*, satisfying* $T(\tilde{x}^i) = Q$*.*

We next show that $\mathbb{K}$ has an interesting property, provided that $\mathcal{H}_T$ and $\mathcal{H}_{\tilde{\mathcal{X}}}$ belong to certain classes of hypergraphs. A *cycle* of length k in a hypergraph $\mathcal{H} = (V, \mathcal{E})$ is a sequence $(v_1, E_1, v_2, E_2, \ldots, v_k, E_k, v_{k+1})$ such that $v_i \in V$ for all $1 \leq i \leq k+1$, $E_i \in \mathcal{E}$ for all $1 \leq i \leq k$, $E_i \neq E_j$ if $i \neq j$, $v_i, v_{i+1} \in E_i$ for all $1 \leq i \leq k$, and $v_{k+1} = v_1$. Due to [1], a hypergraph $\mathcal{H}$ is *acyclic* if and only if $\mathcal{H}$ is conformal (i.e., its dual $\mathcal{H}^*$ has the Helly property) and for every cycle of length at least 3 in $\mathcal{H}$, some edge of $\mathcal{H}$ contains at least 3 nodes of the cycle. On the other hand, a hypergraph $\mathcal{H}^* = (V^*, \mathcal{E}^*)$ is called *arboreal* if there exists a tree T^* on the node set V^* such that every hyperedge of $\mathcal{H}^*$ induces a subtree in T^*. Due to structural characterizations in [7,8,27], arboreal hypergraphs are dual to acyclic hypergraphs and their intersection graphs are *chordal*, i.e. each cycle having four or more nodes contains a chord: an edge joining two nodes that are not adjacent in the cycle. Moreover, it can be shown that arboreal hypergraphs have the Helly property.

Combining all these properties, the following result is obtained in [30].

Theorem 4. *The following assertions are equivalent:*

- $\mathcal{H}_T$ *is acyclic.*
- $\mathcal{H}_{\tilde{\mathcal{X}}}$ *is arboreal.*
- $\mathbb{K}$ *is chordal.*

Note that a chord (t, t') within a directed cycle C in $\mathbb{D}$ induces a new directed cycle with a smaller number of nodes, as C contains both a directed path from t to t' and a directed path from t' to t. Hence, an orientation of a chordal graph contains a directed cycle of length 3 if and only if it contains any directed cycle. This observation was used in [30] to obtain a new characterization of acyclic transition hypergraphs:

Theorem 5. *The transition hypergraph $\mathcal{H}_T$ is acyclic if and only if, for any orientation $\mathbb{D}$ of $\mathbb{K}$, the following two statements are equivalent:*

1. *$\mathbb{D}$ is strongly valid.*
2. *$\mathbb{D}$ is acyclic.*

5 Finding valid orientations

In the previous sections, we addressed the problem of the existence of a valid orientation and how to recognize and characterize such orientations. This motivates a canonical further question, namely, once the existence of a valid orientation has been confirmed for a deterministic system, can we also find it?

As pointed out in [28], infering a valid orientation of a transition conflict graph shall be done by taking observations on dynamic processes in the underlying deterministic system into account. That is: Based on the knowledge of the values of $\mathrm{succ}(x)$ at *some* states $x \in \mathcal{X}$, we aim at finding a valid orientation that encodes the global dynamic behavior of the system since $\mathrm{succ}(x)$ can be determined with the help of this orientation for *all* states $x \in \mathcal{X}$.

In the following we consider a deterministic system $\mathcal{S} = (G, u, \mathsf{trans})$ that fulfills the consistency condition from Theorem 1 and hence trans can be encoded as a valid orientation $\mathbb{D}$ of the transition conflict graph. Given as input the capacitated network (G, u) and an *oracle* for returning the value of $\mathsf{trans}(x)$ at any state $x \in \mathcal{X}$, we want to determine $\mathbb{D}$ by calling the oracle as few times as possible since in practice, a call to the oracle stands for the execution of a (probably expensive and time-consuming) experiment.

We call a set $\mathcal{X}' \subseteq \mathcal{X}$ of states a *valid test set* if the information about the corresponding highest-priority transitions $\{\mathsf{trans}(x) : x \in \mathcal{X}'\}$ suffices for inferring the direction of all arcs in $\mathbb{D}$. Consequently, we are interested in the following problem, formulated in [28].

Definition 3 (Minimum Valid Test Set Problem (MVTP)). *Given a deterministic system $\mathcal{S}$ together with an oracle for returning highest-priority transitions, find a valid test set of minimum cardinality.*

The meaning of *inferring* an orientation deserves further explanation. Let us denote by *partial orientation* a (mixed) graph $\mathbb{D}' := (T, \mathbb{A}', \mathbb{E}')$ obtained by fixing the direction for *some* edges of $\mathbb{K}$, with $\mathbb{A}'$ being the set of the corresponding oriented arcs, and $\mathbb{E}'$ the set of the remaining unoriented edges. A partial orientation is *extendible* if it is possible to choose directions for all unoriented edges in such a way that a valid orientation is obtained. Given an extendible

partial orientation $\mathbb{D}' := (T, \mathbb{A}', \mathbb{E}')$, a yet unoriented edge $tt' \in \mathbb{E}'$ is said to be *inferable as* (t, t') if the partial orientation $\hat{\mathbb{D}}' := (T, \mathbb{A}' \cup \{(t', t)\}, \mathbb{E}' \setminus \{tt'\})$ is not extendible. Moreover, $\mathbb{D}'$ is *sufficient* for inferring a valid orientation $\mathbb{D} := (T, \mathbb{A})$ if all edges in $\mathbb{E}'$ are inferable as the corresponding arcs in $\mathbb{A}$.

Example 3. Figure 3 illustrates these concepts. The partial orientation depicted in (a) is not extendible, as choosing any direction for edge t_2t_3 produces one inclusionwise maximal clique without sink. In contrast, any direction chosen for this edge in (b) leads to a valid orientation, so t_2t_3 is not inferable in the second example. A sufficient partial orientation is shown in (c): the two unoriented edges t_2t_3 and t_3t_4 *must* be oriented as (t_3, t_2) and (t_3, t_4). Orienting any of them in the opposite direction leads to a non-extendible partial orientation.

This shows that the optimal solution of MVTP may depend on the underlying valid (complete) orientation $\mathbb{D}$, and not on the transition conflict graph alone.

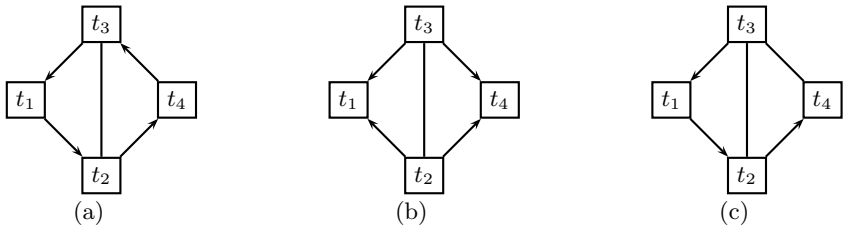

Fig. 3. Different types of partial orientations: (a) non-extendible; (b) extendible, but non-sufficient; and (c) sufficient

In general, recognizing if a partial orientation is extendible, or if an unoriented edge is inferable constitutes a difficult task, due to the lack of a characterization of valid orientations from a graph theoretical point of view. In this section we consider the particular case when $\mathbb{D}$ is acyclic. As mentioned earlier, acyclic orientations are always (strongly) valid and the *only* strongly valid orientations for deterministic dynamic systems with acyclic transition hypergraph $\mathcal{H}_T$ (see Theorem 5). Trivially, if $\mathbb{D}$ is acyclic then any partial orientation cannot contain a directed cycle. On the other hand, in [28] it is proved the following:

Lemma 4. *Any acyclic partial orientation is extendible.*

Recall from Theorem 2 that an orientation is strongly valid if and only if it does not contain a directed cycle of length three. Hence, for any extendible partial orientation $\mathbb{D}' := (T, \mathbb{A}', \mathbb{E}')$, if $(t, t'), (t', t'') \in \mathbb{A}'$ and $tt'' \in \mathbb{E}'$ then tt'' is inferable as (t, t'').

It follows from the previous lemma that an edge $tt' \in \mathbb{E}'$ in an acyclic partial orientation $\mathbb{D}' = (T, \mathbb{A}', \mathbb{E}')$ can be inferred as (t, t') if and only if the digraph $(T, \mathbb{A}')$ contains a directed path from t to t'. As a consequence, $\mathbb{D}'$ is sufficient if for every $tt' \in \mathbb{E}'$ the digraph $(T, \mathbb{A}')$ contains either a directed path from t to t', or a directed path from t' to t. We say in this case that $\mathbb{D}'$ has the *path-property*.

Conversely, given a (complete) orientation $\mathbb{D} = (T, \mathbb{A})$, an arc $a := (t, t') \in \mathbb{A}$ is called *essential* if the digraph $(T, \mathbb{A} \setminus \{a\})$ does not contain a directed path from t to t'. Observe that the direction of a cannot be inferred in any partial orientation extendible to $\mathbb{D}$. Hence, any sufficient partial orientation must contain all essential arcs from $\mathbb{D}$. The next result from [28] shows that no further arcs are required.

Theorem 6. *Let $\mathbb{D} = (T, \mathbb{A})$ be a valid orientation and $\mathbb{A}^* \subseteq \mathbb{A}$ the set of essential arcs. The partial orientation $\mathbb{D}^* := (T, \mathbb{A}^*, \mathbb{E}^*)$, where $\mathbb{E}^*$ is the set of edges from $\mathbb{K}$ corresponding to the arcs in $\mathbb{A} \setminus \mathbb{A}^*$, is sufficient for inferring $\mathbb{D}$.*

Observe that the set $\mathbb{A}^*$ is unique, and that it can be computed from $\mathbb{D}$ in $O(m^2)$ time complexity, with $m := |\mathbb{A}|$, for example by several runnings of a breadth-first search algorithm to determine if each arc of $\mathbb{A}$ is essential or not.

Determining the direction of essential arcs is specially important for reconstructing valid orientations. The following lemma from [28] implies that any valid test set has cardinality larger than or equal to the number of essential arcs in $\mathbb{D}$.

Lemma 5. *For any state $x \in \mathcal{X}$, knowledge of the highest-priority transition from $T(x)$ can be used to orient at most one essential arc.*

As a consequence, the following result is obtained in [28].

Theorem 7. *A valid test set $\mathcal{X}' \subseteq \mathcal{X}$ is optimal for MVTP if and only if $A(x) \cap \mathbb{A}^* \neq \emptyset$ for every $x \in \mathcal{X}'$, where $A(x) := \{(t, \mathsf{trans}(x)) : t \in T(x),\ t \neq \mathsf{trans}(x)\}$ is the set of arcs whose directions are revealed by testing the system at x.*

Note that such an optimal valid test set can be easily determined if $\mathbb{D}$ is known. However, in the setting of the MVTP it is not possible to devise a "winning strategy" that guarantees that each $A(x)$ contains an essential arc, as the next example shows.

Example 4. Consider the conflict graph given in Figure 5(a). Observe that $\mathbb{K}$ contains three cliques of size 3 (triangles) and seven edges, and that every edge is contained in one triangle. It can be checked that for any edge $e \in \mathbb{E}$, there exists a valid orientation where e is not essential. Hence, any winning strategy for MVTP must start by querying the oracle at a state $x^* \in \mathcal{X}$ related to one of the cliques of size 3. But for any of these cliques, there is a valid orientation where $A(x^*)$ contains no essential arc. Indeed, if $T(x^*) = \{t_1, t_2, t_3\}$ or $T(x^*) = \{t_1, t_3, t_4\}$, choose the orientation from Figure 5(c). Otherwise, if $T(x^*) = \{t_1, t_4, t_5\}$, choose the orientation from Figure 5(b).

Even if it is not possible to devise a "winning strategy" that guarantees to orient an essential arc with each oracle call in general, a valid orientation can be obtained by Algorithm 3, provided that the underlying system has one state associated with each clique of the transition conflict graph $\mathbb{K} = (T, \mathbb{E})$. Observe that in this case the obtained orientation is strongly valid.

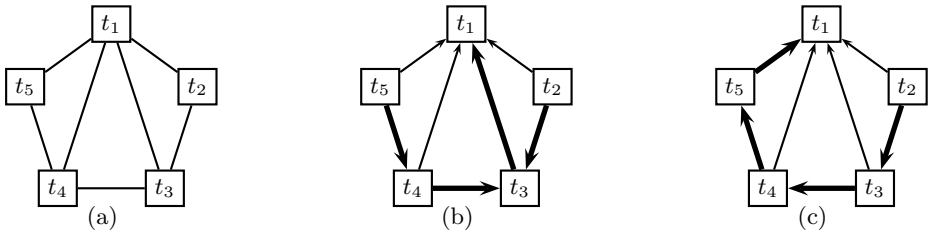

Fig. 4. A transition conflict graph (a) and two valid acyclic orientations (b)-(c). Essential arcs are marked bold.

Input: $\mathbb{K}$ {transition conflict graph}
$\mathcal{Q} = \{Q_1, \ldots, Q_k\}$ {partition of the edges into cliques}
Output: $\mathbb{D}$ {strongly valid orientation}
for $i \leftarrow 1, \ldots, k$ **do**
  while Q_i contains more than one node **do**
    determine $x \in \mathcal{X}$ with $T(x) = Q_i$
    call oracle and determine $\mathsf{trans}(x)$
    orient arcs $\{(t, \mathsf{trans}(x)) : t \in T(x),\, t \neq \mathsf{trans}(x)\}$
    remove $\mathsf{trans}(x)$ from Q_i
  end while
end for

Algorithm 3: Inferring a strongly valid orientation.

Algorithm 3 takes as input a partition of the edges in $\mathbb{E}$ into a set of cliques $\mathcal{Q} := \{Q_1, \ldots, Q_k\}$. At each iteration of the outer loop, it processes a clique $Q_i \in \mathcal{Q}$. At first, the oracle is called to orient all edges incident to the unique sink node of Q_i. Then this node is removed from Q_i and the oracle is called again on the remaining subclique. The procedure is repeated until Q_i contains only one node, which means that all its edges have been oriented. Since the inner loop is executed $|Q_i| - 1$ times, a total of $\sum_{i=1}^{k} |Q_i| - k$ calls to the oracle are required to find the valid orientation $\mathbb{D}$. This quantity is strictly smaller than m if $|Q_i| \geq 3$ holds for at least some $Q_i \in \mathcal{Q}$, since then there is at least one iteration where two edges are oriented simultaneously, and no edge is oriented more than once.

This indeed ensures:

Theorem 8. *If the underlying system has one state associated with each clique of the transition conflict graph, then Algorithm 3 computes a strongly valid orientation.*

6 Discussion

Petri nets constitute a well-established framework for modeling complex dynamic systems. It is common practice to study dynamic processes in terms of directed paths in the reachability or marking graph $\mathcal{G}(x^0)$ starting from a designated

initial state x^0. Hereby, a particular situation occurs when dynamic conflicts are present at states, in which two transitions are enabled, but switching one disables the other. In this case, model animation does not allow definite conclusions about any system properties [11]. The reason is that the occurrence of dynamic conflicts is understood as alternative (branching) system behavior, where a decision between these alternatives is taken non-deterministically. Therefore, the classical reachability analysis has the drawbacks that it

- explores the studied system only from a local point of view, starting from a particular initial state x^0;
- is only applicable for systems of a limited size since the reachability graph $\mathcal{G}(x^0)$ needs to be explicitly constructed, but grows exponentially in the size of the network;
- cannot be used to study systems that exhibit a deterministic behavior.

Our aim was to overcome these difficulties by presenting a way to compactly model deterministic systems from a global point of view (independent from a particular initial state), and to predict the systems behavior for all possible system states (whithout generating explicitly the state digraph). For that, we have examined a new approach for encoding the dynamic behavior of certain deterministic discrete systems that relies on extending the familiar framework of Petri nets. Our encoding consists in a realization of the successor-oracle as a valid orientation of the edges of the transition conflict graph $\mathbb{K}$, together with Algorithm 1. This encoding is compact in the sense that the amount of space required for its storage is polynomial in the size of the network. Therefore, it is well-suited for being integrated into simulation algorithms like [12] to study the dynamics of large complex deterministic systems, and to address issues such as reachability, boundedness, existence of deadlocks, and liveness, among others.

A possible interesting extension of our model concerns the study of dynamic systems where the concurrent switch of various transitions can occur. Throughout this paper we have assumed that a change from a state $x \in \mathcal{X}$ to its successor state $\mathrm{succ}(x)$ can always be explained by the switch of a *single* transition $\mathsf{trans}(x)$. However, there are deterministic systems where this does not hold, as shown in the following example.

Example 5. Consider a deterministic system $(G, \mathbb{1}, \mathsf{succ})$ with network G as depicted in Figure 5 and $w = \mathbb{1}$. It is straightforward to check that the system has the four branching states $x^0, \ldots, x^3$ shown in the figure. The following table lists the sets of enabled transitions at each of these states, as well as the corresponding successor states specified by succ:

j	branching state x^j	$T(x^j)$	$\mathrm{succ}(x^j)$
0	$(1,1,0,0)^T$	$\{\mathbf{t_1}, \mathbf{t_2}\}$	$(0,0,1,1)^T$
1	$(0,1,1,0)^T$	$\{t_2, \mathbf{t_3}\}$	$(0,1,0,1)^T$
2	$(1,0,0,1)^T$	$\{t_1, \mathbf{t_4}\}$	$(0,1,0,0)^T$
3	$(1,1,1,0)^T$	$\{t_2, \mathbf{t_3}\}$	$(1,1,0,1)^T$

Observe that t_1 and t_2 have to be switched *concurrently* at state x^0 to reach $\mathrm{succ}(x^0)$, whereas at each of the other branching states a single transition is switched to reach the corresponding successor state, namely, t_3 at x^1 or x^4, and t_4 at x^2. The enabled transitions are highlighted in gray in Figure 5, while the transitions actually switched are marked in bold.

The above successor function *cannot* be realized with the help of a transition selection function trans due to the following reason. Considering x^0 as initial state, trans has to select *one* of the transitions $t_1, t_2 \in T(x^0)$. However, any choice leads to an intermediate state (x^1 for $\mathsf{trans}(x^0) = t_1$, x^2 for $\mathsf{trans}(x^0) = t_2$) where the remaining transition from $T(x^0)$ is still enabled but must *not* be selected, as both $\mathsf{trans}(x^1) = t_3$ and $\mathsf{trans}(x^2) = t_4$ are implied by the specification of succ. (See the reachability graph in Figure 5.) Thus, none of the potential interleaving sequences between x^0 and $\mathrm{succ}(x^0)$ can be expressed by means of single transition switches such that every switch is determined by the value of trans at the corresponding state.

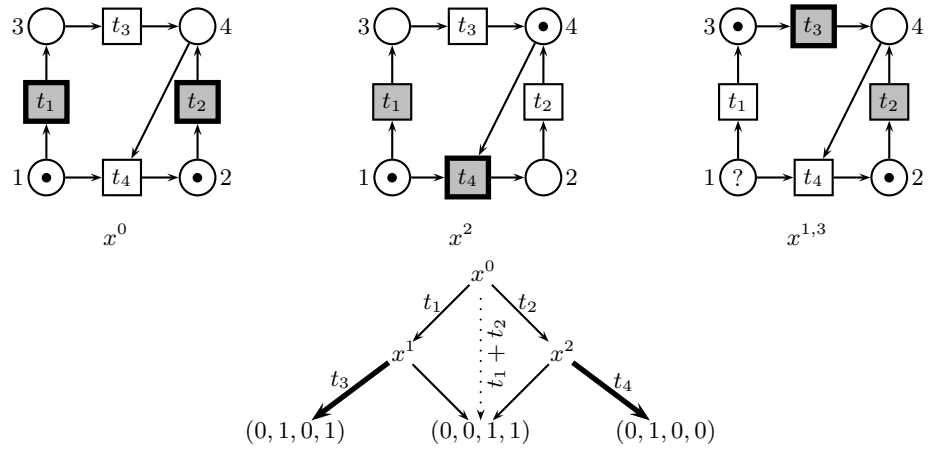

Fig. 5. A deterministic system whose dynamic behavior cannot be modeled via a transition selection function. The enabled transitions at each state are highlighted in gray, the next transitions to be switched are shown in bold. All place capacities and arc weights are assumed to be equal to one.

One way of dealing with these systems could be through the inclusion of pseudo-nodes in the transition conflict graph, to account for the simultaneous switching of various transitions. The advantage would be that all results from the previous sections directly carry over to the extended setting. The disadvantage of such a modification is, however, the dramatic increase in the size of the transition conflict graph, as (in the worst-case) the number of required pseudo-nodes may grow exponentially with respect to $|T|$.

An alternative approach consists in working on a slightly different transition conflict graph, where an edge between two transitions means that they are in dynamic conflict (i.e. when switching one transition disables the other). In this

case, $T(x)$ does not longer induce a clique in $\mathbb{K}$. Yet we can now define an orientation to be valid if for any state $x \in \mathcal{X}$ the set $T(x)$ of enabled transitions induces a subgraph containing *at least one* sink. These sinks reveal an *anti-chain* of transitions with maximal priorities, which have to be switched concurrently. Theorem 1 can be generalized to this new setting in a straightforward manner, characterizing which deterministic systems admit valid orientations. The advantage of this scheme is that it is still compact with respect to the size of the network G. However, as the sets $T(x)$ do not induce cliques in the conflict graph, many of the combinatorial properties pointed out in Section 3-5 do not hold anymore. Moreover, since all transitions that *may* switch concurrently are *required* to do so, it is again possible to construct examples of deterministic systems whose behavior cannot be modeled in this way. Hence, further research is necessary to ensure a compact encoding of a successor oracle for all deterministic systems. For a large number of such systems, however, the here presented results allow us already the use of such an oracle for the study of dynamic processes from a global point of view, independent from a particular initial state and the (complete) construction of the reachability graph.

References

1. Acharya, B.D., Las Vergnas, M.: Hypergraphs with cyclomatic number zero, triangulated hypergraphs and an inequality. J. Combin. Theory (B) **33**, 52–56 (1982)
2. Adam, N.R., Atluri, V., Huang, W.K.: Modeling and analysis of workflows using Petri nets. J. Intell. Inf. Syst. **10**(2), 131–158 (1998).
3. Balbo, G.: Introducation to stochastic Petri nets. In: Lectures on formal methods and performance analysis: first EEF/Euro summer school on trends in computer science, pp. 84–155. Springer-Verlag New York, Inc., New York, NY, USA (2002)
4. Berge, C., Duchet, P.: A generalisation of Gilmore's theorem. In: M. Fiedler (ed.) Recent Advances in Graph Theory, pp. 49–55. Acad. Praha, Prague (1975)
5. Billington, J., Diaz, M., Rozenberg, G.: Application of Petri Nets to Communication Networks, Advances in Petri Nets. Springer-Verlag, London, UK (1999)
6. David, R., Alla, H.: Discrete, Continuous, and Hybrid Petri Nets. Springer-Verlag Berlin Heidelberg, Heidelberg (2005)
7. Duchet, P.: Propriété de Helly et problèmes de représentations. In: Problèmes Combin. et Théorie des Graphes, Coll. Orsay 1976, pp. 117–118. CNRS Paris (1978)
8. Flament, C.: Hypergraphes arborés. Discrete Math. **21**, 223–226 (1978)
9. Gu, T., Bahri, P.A.: A survey of Petri net applications in batch processes. Comput. Ind. **47**(1), 99–111 (2002).
10. Hardy, S., Robillard, P.N.: Modeling and simulation of molecular biology systems using Petri nets: modeling goals of various approaches. Journal of Bioinformatics and Computational Biology **2**(4), 619–637 (2004)
11. M. Heiner, D. Gilbert, R. Donaldson: Petri nets for systems and synthetic biology, In proceedings of SFM 2008, Springer, LNCS 5016:215–264, 2008.
12. M. Heiner, M. Herajy, F. Liu, C. Rohr, M. Schwarick: Snoopy – a unifying Petri net tool. In proceedings of PETRI NETS 2012, Hamburg, Springer, LNCS 7347:398–407, 2012.
13. M. Heiner, C. Rohr, M. Schwarick: MARCIE - Model checking And Reachability analysis done effiCIEntly; In proceedings of PETRI NETS 2013, Milano, Springer, LNCS 7927:389–399, 2013.

14. Jensen, K.: Coloured Petri nets: basic concepts, analysis methods and practical use, volume 3. Springer-Verlag New York, Inc., New York, NY, USA (1997)
15. Lipton, R.: The reachability problem requires exponential space. Research Report 62, Yale University, Computer Science Dept. (1976)
16. Marsan, M., Balbo, G., Donatelli, S., Franceschinis, G., Conte, G.: Modelling with generalized stochastic Petri nets. Wiley Series in Parallel Computing (1995)
17. Marwan, W.: Theory of time-resolved somatic complementation and its use for the analysis of the sporulation control network of physarum polycephalum. Genetics **164**, 105–115 (2003)
18. Marwan, W., Starostzik, C.: The sequence of regulatory events in the sporulation control network of physarum polycephalum analysed by time-resolved somatic complementation of mutants. Protist **153**, 391–400 (2002)
19. Marwan, W., Sujatha, A., Starostzik, C.: Reconstructing the regulatory network controlling commitment and sporulation in physarum polycephalum based on hierarchical Petri net modeling and simulation. J. Th. Biology **236**, 349–365 (2005)
20. Marwan, W., Wagler, A., Weismantel, R.: A mathematical approach to solve the network reconstruction problem. Math. Meth. Oper. Research **67**, 117–132 (2008)
21. W. Marwan, A. Wagler, R. Weismantel: Petri nets as a framework for the reconstruction and analysis of signal transduction pathways and regulatory networks, Natural Computing **10**, 639–654 (2011)
22. Matsuno, H., Aoshima, H., Doi, A., Tanaka, Y., Matsui, M.: Biopathways representation and simulation on hybrid functional Petri net. In Silico Biology **3**(3), 389–404 (2003)
23. Mayr, E.W.: An algorithm for the general Petri net reachability problem. In: Proceedings of the 13th Ann. ACM Symposium on Theory of Computing, pp. 238–246. ACM Press (1981)
24. Mayr, E.W.: An algorithm for the general Petri net reachability problem. SIAM J. Comput. **13**(3), 441–460 (1984)
25. Reisig, W.: Petri nets: an introduction. Springer-Verlag New York, Inc., New York, NY, USA (1985)
26. Reisig, W.: Elements of distributed algorithms: modeling and analysis with Petri nets. Springer-Verlag New York, Inc., New York, NY, USA (1998)
27. Slater, P.J.: A characterization of soft hypergraphs. Canad. Math. Bull. **21**, 335–337 (1978)
28. Torres, L.M., Wagler, A.: Model reconstruction for discrete deterministic systems. Electronic Notes of Discrete Mathematics **36**, 175–182 (2010)
29. Torres, L.M., Wagler, A.: Encoding the dynamics of deterministic systems. Mathematical Methods of Operations Research **73**(3), 281–300 (2011)
30. Torres, L.M., Wagler, A.: Analyzing the dynamics of deterministic systems from a hypergraph theoretical point of view. RAIRO Oper. Research **47**, 321–330 (2013)
31. Yakovlev, A., Koelmans, A., Semenov, A., Kinniment, D.: Modelling, analysis and synthesis of asynchronous control circuits using Petri nets. Integr., VLSI J. **21**(3), 143–170 (1996).