Testing the executability of scenarios in general inhibitor nets

Robert Lorenz, Sebastian Mauser, Robin Bergenthum
Department of Applied Computer Science
Catholic University of Eichstätt-Ingolstadt
85072 Eichstätt, Germany
firstname.lastname@ku-eichstaett.de

Abstract

In this paper we introduce executions of place/transition Petri nets with weighted inhibitor arcs (PTI-net) as enabled labeled stratified order structures (LSOs) and present a polynomial algorithm to decide whether a scenario given by an LSO is an execution of a given PTI-net.

The algorithm is based on an equivalent characterization of enabled LSOs called token flow property. Although the definition of the token flow property involves exponential many objects in the size of the LSO, there is a nontrivial transformation into a flow optimization problem which can be solved in polynomial time.

1 Introduction

Specifications of concurrent systems are often formulated in terms of scenarios expressing causal dependencies and concurrency among events. In other words it is often part of the specification that some scenario should or should not be an execution of the system. Thus, given a system, a natural question is whether a scenario is an execution of the system. In [5] we presented a polynomial algorithm to answer this question when the system is given by a place/transition Petri net (p/t-net) and a scenario is given as a labeled partial order (LPO).

"Petri nets with inhibitor arcs are intuitively the most direct approach to increase the modeling power of Petri nets" [9] and have been found appropriate in various application areas [1, 2]. In fact, it is well known that such nets are even equivalent to Turing-machines (w.r.t. to their sequential behavior) and thus several decision problems such as the reachability problem which are decidable for p/t-nets are undecidable for nets with inhibitor arcs. Nevertheless, we can show in this paper that the results from [5] can be generalized to p/t-nets with weighted inhibitor arcs (PTI-nets), the most general notion of Petri nets with inhibitor arcs. For such nets, scenarios can be formally given as labeled stratified order structures (LSOs), a proper generalization of LPOs.

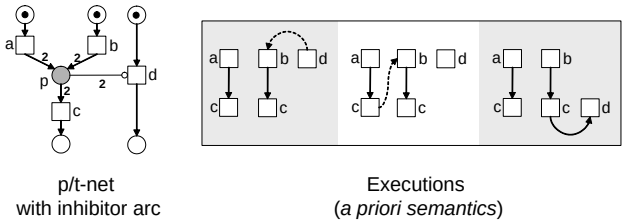

Figure 1. PTI-net with executions.

As an example, Figure 1 shows a p/t-net with inhibitor arc (p, d) having weight 2. This inhibitor arc specifies that d is only enabled to occur if p is not marked by more than two tokens. Thus the enabledness of a transition depends on tests via weighted inhibitor arcs whether the number of tokens in places does not exceed the weight of the inhibitor arc (beside the classical enabling conditions of p/t-nets). Throughout the paper, we consider the *a priori semantics* of PTI-nets (other semantics are briefly discussed in the conclusion). According to the a priori semantics the test of markings of places via inhibitor arcs precedes the execution of transitions. In Figure 1 the behavior of the transitions a, b and c is not restricted by inhibitor arcs: In the initial marking the transitions a and b can be executed concurrently (that means in any order and as well at the same time). Then transition c can be executed twice and for example consumes once the tokens produced by a and once the tokens produced by b. That means the transitions a and b are executed "earlier than" transition c (respectively). Consider now transition d: Since after the occurrence of a and b the place p is marked by four tokens, d cannot be executed concurrently to a and b (since then the occurrence of d is prohibited by the inhibitor arc (p, d)). But d can be executed concurrently to a, if it does not occur later than b (since then the number of tokens in place p does not exceed the inhibitor arc weight 2). In other words, the transitions b and d, when executed concurrently to a (independent from a), cannot occur con-

currently or sequentially in order $b \rightarrow d$. But they still can occur at the same time (because of the occurrence rule "testing before execution") or sequentially in order $d \rightarrow b$ – this is exactly the behavior described by "d not later than b".

The described causal relations between transitions of the net are illustrated by the execution shown most left in Figure 1. The solid arcs represent the "earlier than" relation between events and the dashed arc depicts the "not later than" relation explained above. There are also other possible executions of the PTI-net from Figure 1: If c is executed once "not later than" b, then the number of tokens in the place p can't exceed the value 2 and thus d can be executed concurrently to the executions of a, b and (two times) c (see the second execution in Figure 1). Further (with the same argument), it is possible that c is executed once "earlier than" d (see the third execution in Figure 1).

Of course also symmetric "not later than" relations are in general possible between events, in which case these events can only occur at the same time, but not sequentially in any order. Such events are called *synchronous*. With partial orders one can only model "earlier than" relations between events but it is not possible to describe relationships, where synchronous occurrence is possible but concurrency is not existent. In [7] causal semantics based on LSOs like the executions in Figure 1 consisting of a combination of "earlier than" and "not later than" relations between events were proposed to cover such cases. Thus, we consider scenarios to be formally given through LSOs.

In this paper we present a polynomial algorithm running in $O(|P|n^4)$ time to test whether a given LSO is an execution of a given PTI-net, where n is the number of nodes of the LSO and $|P|$ is the number of places of the PTI-net. Since up to now for PTI-nets there is no formal definition of executions, it is thereto first necessary to lift this notion for p/t-nets to the PTI-net level. There are three equivalent characterizations (definitions) of executions of p/t-nets, namely (i) LPOs *enabled* w.r.t. a p/t-net, (ii) LPOs *executable* in a p/t-net, and (iii) LPOs fulfilling the *token flow property* w.r.t. a p/t-net. The first two characterizations do not lead to efficient tests. Their equivalence was shown in [6, 11]. In [5] we introduced the token flow property of LPOs, showed its equivalence to the other two characterizations and developed an efficient algorithm to test whether a given LPO satisfies the token flow property w.r.t. a given marked p/t-net. Of course, it is desirable to extend all three characterizations to LSOs and PTI-nets and prove their equivalence also on the PTI-net level.

We propose a definition of *LSOs enabled w.r.t. a PTI-net* which is a proper generalization of the definition of LPOs enabled w.r.t. a p/t-net and allows the representation of executions with minimal causal dependencies between transition occurrences of PTI-nets (Subsection 3.1). We also define the *token flow property* of LSOs w.r.t. PTI-nets as a generalization of the respective notion for LPOs and p/t-nets (Subsection 3.2) and show the equivalence of these two notions (Subsection 3.3).[1] The polynomial algorithm is then developed from the token flow property (Section 4). It is based on a nontrivial transformation of the token flow property into a flow optimization problem. In the Appendix we briefly explain necessary results from flow theory for completeness of the presentation to support the reviewing procedure. In Figure 2 the relationships between the different characterizations of executions are depicted for p/t-nets (left part) and PTI-nets (right part) thus illustrating the theorems shown in this paper in relation to analogue results known for p/t-nets. We start with a short introduction of LSOs and PTI-nets in Section 2.

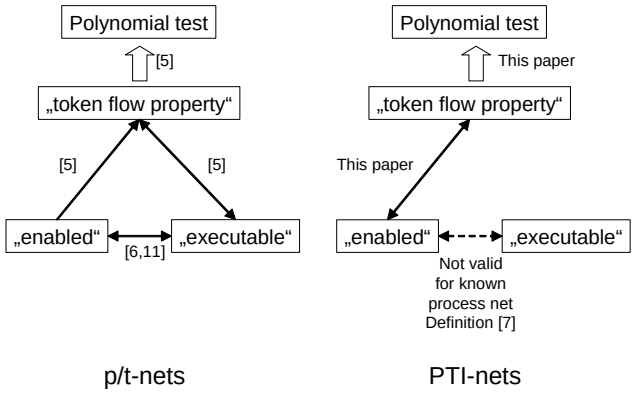

Figure 2. Theorems in this paper.

2 Preliminaries

In this section we recall the basic definitions of *stratified order structures* and *p/t-nets with weighted inhibitor arcs*.

We use $\mathbb{N}$ to denote the nonnegative integers. Given a function f from A to B and a subset C of A we write $f|_C$ to denote the restriction of f to the set C. Given a finite set A, the symbol $|A|$ denotes the cardinality of A. The set of all multi-sets over a set A is denoted by $\mathbb{N}^A$. Finally, we denote the identity relation over A by id_A.

2.1 Stratified order structures

We start with some basic notions preparative to the definition of *stratified order structures* (*so-structures*). A *directed graph* is a pair $(V, \rightarrow)$, where V is a finite *set of*

[1] The definition of *executable* LPOs is strongly related to the definition of process nets of p/t-nets. Since the most general notion of process nets existent for PTI-nets ([7]) does not define minimal causal dependencies between transition occurrences, there are enabled LSOs which are not executable when lifting this notion to the PTI-net level. Therefore, we do not consider executable LSOs here. The exact relationship between the generalizations of the characterizations (i) and (ii) to the PTI-net level is examined in another paper we submitted to the International Conference on *Application and Theory of Petri Nets (ATPN)* 2007.

nodes and $\rightarrow \subseteq V \times V$ is a binary relation over V called the *set of arcs*. As usual, given a binary relation $\rightarrow$, we write $a \rightarrow b$ to denote $(a,b) \in \rightarrow$. Two nodes $a, b \in V$ are called *independent* w.r.t. the binary relation $\rightarrow$ if $a \not\rightarrow b$ and $b \not\rightarrow a$. We denote the set of all pairs of nodes independent w.r.t. $\rightarrow$ by $co_{\rightarrow} \subseteq V \times V$. A *partial order* is a directed graph $\mathrm{po} = (V, <)$, where $<$ is an irreflexive and transitive binary relation on V. If $co_{<} = id_V$ then $(V, <)$ is called *total*. Given two partial orders $\mathrm{po}_1 = (V, <_1)$ and $\mathrm{po}_2 = (V, <_2)$, we say that po_2 is a *sequentialization* (or *extension*) of po_1 if $<_1 \subseteq <_2$.

So-structures are, loosely speaking, combinations of two binary relations on a set of events where one is a partial order representing an "earlier than" relation and the other represents a "not later than" relation. Thus so-structures describe finer causalities than partial orders. Formally, so-structures are *relational structures* satisfying certain properties. A *relational structure* (rel-structure) is a triple $\mathcal{S} = (V, \prec, \sqsubset)$, where V is a set (of *events*), and $\prec \subseteq V \times V$ and $\sqsubset \subseteq V \times V$ are binary relations on V. A rel-structure $\mathcal{S}' = (V, \prec', \sqsubset')$ is said to be an *extension* of another rel-structure $\mathcal{S} = (V, \prec, \sqsubset)$, written $\mathcal{S} \subseteq \mathcal{S}'$, if $\prec \subseteq \prec'$ and $\sqsubset \subseteq \sqsubset'$.

Definition 1 (Stratified order structure [7])**.** *A rel-structure $\mathcal{S} = (V, \prec, \sqsubset)$ is called* stratified order structure *(so-structure) if the following conditions are satisfied for all $u, v, w \in V$:*
(C1) $u \not\sqsubset u$.
(C2) $u \prec v \implies u \sqsubset v$.
(C3) $u \sqsubset v \sqsubset w \wedge u \neq w \implies u \sqsubset w$.
(C4) $u \sqsubset v \prec w \vee u \prec v \sqsubset w \implies u \prec w$.

In figures $\prec$ is graphically expressed by solid arcs and $\sqsubset$ by dashed arcs. According to (C2) a dashed arc is omitted if there is already a solid arc. Moreover, we omit arcs which can be deduced by (C3) and (C4). It is shown in [7] that $(V, \prec)$ is a partial order. Therefore so-structures are a generalization of partial orders which turned out to be adequate to model the causal relations between events of PTI-nets under the a priori semantics. In this context $\prec$ represents the ordinary "earlier than" relation (as for p/t-nets) while $\sqsubset$ models a "not later than" relation. According to [7] for nodes $u, v \in V$ there is an extension $\mathcal{S}' = (V, \prec', \sqsubset')$ of $\mathcal{S}$ with $u \prec' v$ if and only if $v \not\sqsubset u$ and $u \neq v$. In particular, there holds $u \prec v \implies v \not\sqsubset u$. These properties justify the described causal interpretation of $\prec$ and $\sqsubset$. The executions shown in Figure 1 are so-structures with labeled nodes.

We introduce a subclass of so-structures which turns out to be associated to sequences of (synchronous) transition steps of PTI-nets.

Definition 2 (Total linear so-structures)**.** *Let $\mathcal{S} = (V, \prec, \sqsubset)$ be an so-structure, then $\mathcal{S}$ is called* total linear *if $co_{\prec} = (\sqsubset \setminus \prec) \cup id_V$. The set of all* total linear extensions *(or* linearizations*) of an so-structure $\mathcal{S}$ is denoted by $lin(\mathcal{S})$.*

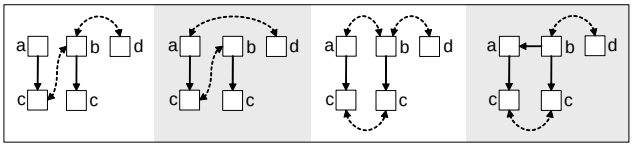

Figure 3. Total linear so-structures.

Figure 3 shows four total linear so-structures with labeled nodes (LSOs), where the first and second LSOs are extensions of the second execution shown in Figure 3 and the third and fourth LSOs are extensions of the first execution shown in Figure 3 (using the results from [7] about augmenting so-structures one can conclude that every so-structure is extendable to a total linear so-structure).

For the definition of the token flow property for LSOs we need the notion of prefixes (of nodes) of so-structures. These are defined by subsets of nodes which are downward closed w.r.t. the $\sqsubset$-relation:

Definition 3 (Prefix (of a node))**.** *Let $\mathcal{S} = (V, \prec, \sqsubset)$ be an so-structure and let $V' \subseteq V$ be such that $(u' \in V' \wedge u \sqsubset u') \implies u \in V'$. Then $\mathcal{S}' = (V', \prec|_{V' \times V'}, \sqsubset|_{V' \times V'})$ is called* prefix *of $\mathcal{S}$. We say that the prefix $\mathcal{S}'$ is* defined by *V'. If additionally $\{v' \in V \mid v' \prec v\} \subseteq V'$ and $\{v' \in V \mid v = v' \vee v \sqsubset v'\} \cap V' = \emptyset$ for some $v \in V$, then $\mathcal{S}'$ is called* prefix of v (w.r.t. $\mathcal{S}$).

This definition of prefixes is compatible with the set of linearizations of an so-structure in the following sense:

Lemma 4. *Let $\mathcal{S} = (V, \prec, \sqsubset)$ be an so-structure, $V' \subseteq V$ and $v \in V$. Then V' defines a prefix of v w.r.t. $\mathcal{S}$ if and only if there is a linearization $\mathcal{S}' \in lin(\mathcal{S})$ such that V' defines a prefix of v w.r.t. $\mathcal{S}'$.*

Proof. **if**: Let $\mathcal{S}' = (V, \prec', \sqsubset') \in lin(\mathcal{S})$ and let $V' \subset V$ define a prefix of v w.r.t. $\mathcal{S}'$. Consider nodes $u' \in V'$ and $u \in V$ with $u \sqsubset u'$. Since $\mathcal{S}'$ is an extension of $\mathcal{S}$ this implies $u \sqsubset' u'$. Because V' defines a prefix of $\mathcal{S}'$ we get $u \in V'$. Thus V' also defines a prefix of $\mathcal{S}$. Let further $v' \prec v$. Again, since $\mathcal{S}'$ is an extension of $\mathcal{S}$ this implies $v' \prec' v$ and therefore we have $v' \in V'$. Finally, if $v \sqsubset v'$ then $v \sqsubset' v'$ and therefore $v' \notin V'$. Thus, V' defines in fact a prefix of v.

only if: Let V' define a prefix of v w.r.t. $\mathcal{S}$. We construct a linearization $\mathcal{S}' = (V, \prec', \sqsubset')$ of $\mathcal{S}$ such that V' also defines a prefix of v w.r.t. $\mathcal{S}'$. For this denote $V_0 \subseteq V'$ the set of all nodes which are minimal w.r.t. $\prec$ in $\mathcal{S}$. Then consider the restriction of $\mathcal{S}$ onto the node set $V \setminus V_0$ and denote $V_1 \subseteq V'$ the set of all nodes which are minimal w.r.t. $\prec$ in this new so-structure. Following this technique, we define inductively $V_n \subseteq V'$ as the set of nodes which are minimal

w.r.t. the restriction of $\prec$ onto the node set $V \setminus (\bigcup_{i=0}^{n-1} V_i)$, as long as $V' \setminus (\bigcup_{i=0}^{n-1} V_i) \neq \emptyset$. If $V' \setminus (\bigcup_{i=0}^{N-1} V_i) = \emptyset$ and $V \setminus (\bigcup_{i=0}^{N-1} V_i) \neq \emptyset$ for some N, then further define $V_N \subseteq V$ as the set of nodes which are minimal w.r.t. the restriction of $\prec$ onto the node set $V \setminus (\bigcup_{i=0}^{N-1} V_i)$ and so on (note that $v \in V_N$ because V' defines a prefix of v).

We now can define $\mathcal{S}'$ through $\prec' = \bigcup_{i<j} V_i \times V_j$ and $\sqsubset' = ((\bigcup_i V_i \times V_i) \setminus id_{V_i}) \cup \prec'$. By construction $\mathcal{S}'$ is a total linear so-structure. It remains to show that $\prec \subseteq \prec'$, $\sqsubset \subseteq \sqsubset'$, $\{v' \in V \mid v' \prec' v\} \subseteq V'$ and $\{v' \in V \mid v = v' \vee v \sqsubset' v'\} \cap V' = \emptyset$.

Let $u, v \in V$ with $u \prec v$: Since V' defines a prefix of $\mathcal{S}$, it is not possible that $v \in V'$ and $u \notin V'$. Suppose $u, v \in V'$, $u, v \in V \setminus V'$ or $u \in V'$ and $v \notin V'$: Then by construction there must be $i < j$ with $u \in V_i$ and $v \in V_j$. This gives $u \prec' v$.

Let $u, v \in V$ with $u \sqsubset v$: Since V' defines a prefix of $\mathcal{S}$, it is not possible that $v \in V'$ and $u \notin V'$. Suppose $u, v \in V'$ or $u, v \in V \setminus V'$: Let $u \in V_i$ and $v \in V_j$. Assume that v is minimal w.r.t. $\prec$ in an earlier step than u. Then in this step there is $u' \prec u$ but $u' \not\prec v$. This contradicts (C4). Therefore either u and v are minimal in the same step or u is minimal in a step earlier than v. This gives $u \sqsubset' v$. Suppose $u \in V'$ and $v \notin V'$: Then by construction there must be $i < j$ with $u \in V_i$ and $v \in V_j$. This gives $u \prec' v$.

Let $v' \in V$ with $v' \prec' v$: Since by construction $v \in V_N$ there is $n < N$ with $v' \in V_n \subseteq V'$. If finally $v \sqsubset' v'$, then $v' \in V_n$ for some $n \geqslant N$, i.e. $v' \notin V'$. □

We will often use *labeled so-structures* (LSOs) in the following. These are so-structures $\mathcal{S} = (V, \prec, \sqsubset)$ together with a *set of labels* T and a *labeling function* $l : V \to T$. We use the above notations defined for so-structures also for LSOs. If T is a set of labels of $\mathcal{S}$, i.e. $l : V \to T$, then for a set $U \subseteq V$, we define the multi-set $|U|_l \subseteq \mathbb{N}^T$ by $|U|_l(t) = |\{v \in V \mid v \in U \wedge l(v) = t\}|$.

We will consider LSOs only up to isomorphism. As usual, two LSOs $(V, \prec, \sqsubset, l)$ and $(V', \prec', \sqsubset', l')$ are called *isomorphic*, if there is a bijective mapping $\psi : V \to V'$ such that $l(v) = l'(\psi(v))$ for $v \in V$, $v \prec w \Leftrightarrow \psi(v) \prec' \psi(w)$ and $v \sqsubset w \Leftrightarrow \psi(v) \sqsubset' \psi(w)$ for $v, w \in V$.

We will use the same notions for LPOs, too (since an LPO can be understood as an LSO with a not later than relation that equals the earlier than relation).

2.2 PTI-nets

A *net* is a triple (P, T, F), where P is a finite set of *places*, T is a finite set of *transitions*, satisfying $P \cap T = \emptyset$, and $F \subseteq (P \cup T) \times (T \cup P)$ is a *flow relation*. Let (P, T, F) be a net and $x \in P \cup T$ be an element. The *preset* $\bullet x$ is the set $\{y \in P \cup T \mid (y, x) \in F\}$, and the *post-set* $x\bullet$ is the set $\{y \in P \cup T \mid (x, y) \in F\}$. Given a set $X \subseteq P \cup T$, this notation is extended by $\bullet X = \bigcup_{x \in X} \bullet x$ and $X \bullet = \bigcup_{x \in X} x \bullet$. For technical reasons, we consider only nets in which every transition has a nonempty pre-set and post-set.

A *place/transition net* (shortly *p/t-net*) N is a quadruple (P, T, F, W), where (P, T, F) is a net and $W : F \to \mathbb{N}^+$ is a *weight function*. We extend the weight function W to pairs of net elements $(x, y) \in (P \times T) \cup (T \times P)$ satisfying $(x, y) \notin F$ by $W((x, y)) = 0$.

Definition 5 (PTI-net)**.** *A* PTI-net N *is a quadruple* (P, T, F, W, I)*, where* (P, T, F, W) *is a p/t-net and* $I : P \times T \to \mathbb{N} \cup \{\omega\}$ *is the* weighted inhibitor relation*. If* $I(p, t) \neq \omega$*, then* $(p, t) \in P \times T$ *is called* (weighted) inhibitor arc *and* p *is an* inhibitor place of t*.*

In the following we denote $n < \omega$ for $n \in \mathbb{N}$. A *marking* of a PTI-net $N = (P, T, F, W, I)$ is a function $m : P \to \mathbb{N}$, i.e. a multi-set over P. A *marked PTI-net* is a pair (N, m_0), where N is a PTI-net and m_0 is a marking of N called *initial marking*. A transition t can only be executed if p carries at most $I((p, t))$ tokens. In particular, if $I((p, t)) = 0$ then p must be empty. $I((p, t)) = \omega$ means that t can never be prevented from occurring by the presence of tokens in p. In diagrams, inhibitor arcs have small circles as arrowheads. Just as normal arcs, inhibitor arcs are annotated with their weights. Now however, the weight 0 is not shown. Figure 1 shows a marked PTI-net.

According to the a priori semantics of PTI-nets, the inhibitor test for enabledness of a transition precedes the consumption and production of tokens in places. Thus, a multi-set (a step) of transitions is (synchronously) enabled in a marking, only if in this marking each transition in the step obeys the inhibitor constraints before the step is executed.

Definition 6 (Occurrence rule)**.** *Let* $N = (P, T, F, W, I)$ *be a* PTI-net*. A multi-set of transitions* τ *(a step) is* enabled to occur in a marking m w.r.t. the a priori semantics *if* $m(p) \geq \sum_{t \in \tau} \tau(t) W((p, t))$ *for every place* $p \in \bullet\tau$ *and* $m(p) \leq I((p, t))$ *for each place* p *and each transition* $t \in \tau$*.*

The *occurrence* of a step (of transitions) τ leads to the new marking m' defined by $m'(p) = m(p) - \sum_{t \in \tau} \tau(t)(W((p, t)) - W((t, p)))$ for every $p \in P$. We write $m \xrightarrow{\tau} m'$ to denote that τ is enabled to occur in m and that its occurrence leads to m'. A finite sequence of steps $\sigma = \tau_1 \dots \tau_n$, $n \in \mathbb{N}$, is called a *step occurrence sequence enabled in a marking* m *and leading to* m_n, denoted by $m \xrightarrow{\sigma} m_n$, if there exists a sequence of markings $m_1, \dots, m_n$ such that $m \xrightarrow{\tau_1} m_1 \xrightarrow{\tau_2} \dots \xrightarrow{\tau_n} m_n$. The marking m_n is said to be *reachable from the marking* m. Moreover each marking is reachable from itself too, by the occurrence of the empty occurrence sequence. In a marked p/t-net, markings reachable from the initial marking m_0 are shortly called *reachable markings*.

A step occurrence sequence can be understood as a possible single *observation* of the behavior of a PTI-net, where the occurrences of transitions in one step are observed *at the same time* or *synchronously*.

We will use the same notions for (marked) p/t-nets, too (since a p/t-net can be understood as a PTI-net with an inhibitor relation which equals the constant ω).

3 Executions

In this Section we lift the notions of "enabled LPOs" and "token flow property" known for LPOs w.r.t. p/t-nets to the setting of PTI-nets w.r.t. the a priori semantics.

3.1 Enabled LSOs

We now introduce LSOs to model executions of PTI-nets. For this, the two relations of an LSO are interpreted as "earlier than" resp. "not later than" relations between transition occurrences. If two transition occurrences are in not later than relation, that means they can be observed (are allowed to be executed) synchronously or sequentially in one specific order. If two transitions are neither in earlier than relation nor in not later than relation, they are concurrent and can be observed (are allowed to be executed) synchronously or sequentially in any order. In this sense one LSO "allows" many observations (step sequences). If all these observations are enabled step occurrence sequences, this LSO is called *enabled*.

Formally, the set of step sequences "allowed" by an LSO is defined as the set of step sequences extending (sequentializing) the LSO, where a step sequence can be easily interpreted itself as an LSO: Each step is represented by a set of events labeled by transitions (transition occurrences) which are in not later than relation with each other (representing synchronous transition occurrences) and transition occurrences in different steps are ordered in appropriate earlier than relation. Formally, for a sequence of transition steps $\sigma = \tau_1 \dots \tau_n$ define the LSO $\mathcal{S}_\sigma = (V, \prec, \sqsubset, l)$ *underlying* σ by: $V = \bigcup_{i=1}^{n} V_i$ and $l : V \to T$ with $|V_i|_l(t) = \tau_i(t)$, $\prec = \bigcup_{i<j} V_i \times V_j$ and $\sqsubset = ((\bigcup_i V_i \times V_i) \cup \prec) \setminus id_V$.

It is easy to see that LSOs underlying a sequence of transition steps are total linear. This is because $co_\prec = \bigcup_{i=1}^{n} V_i \times V_i$ (see Definition 2). For example, the LSOs shown in Figure 3 are associated to the sequences of transition steps (from left to right): $a(b+c+d)c$, $(a+d)(b+c)c$, $(a+b+d)(2c)$ and $(b+d)a(2c)$. Of course, also the reverse is valid, i.e. for each total linear LSO $\mathcal{S} = (V, \prec, \sqsubset, l)$ there is a step sequence σ such that $\mathcal{S}$ and $\mathcal{S}_\sigma$ are isomorphic. Thus, for LSOs which have transition names as labels we can identify total linear LSOs with sequences of transition steps.

Definition 7 (enabled LSO)**.** *Let (N, m_0), $N = (P, T, F, W, I)$, be a marked PTI-net.*

An LSO $\mathcal{S} = (V, \prec, \sqsubset, l)$ with $l : V \to T$ is called enabled (to occur) w.r.t. (N, m_0) (in the a priori semantics) *if the following statement holds: Each finite step sequence $\sigma = \tau_1 \dots \tau_n$ with $\mathcal{S}_\sigma \in lin(\mathcal{S})$ is a step occurrence sequence of (N, m_0). Its occurrence leads to the marking $m'(p)$ given by $m'(p) = m(p) + \sum_{v \in V}(W((l(v), p)) - W((p, l(v))))$.*

This definition is consistent with and a proper generalization of the notion of enabled LPOs: An LPO $\mathrm{lpo} = (V, \prec, l)$ with $l : V \to T$ is *enabled to occur in a marking* m of a marked p/t-net (P, T, F, W, m_0) if each step sequence which extends (sequentializes) lpo is a step occurrence sequence enabled in m_0. Since in LPOs concurrent and synchronous transition occurrences are not distinguished, here a step is considered as a set of events labeled by transitions (transition occurrences) which are concurrent.

Beside this there are two general semantical arguments justifying this definition: First an (enabled) LSO $\mathcal{S}$ is completely represented by the set of its linearizations $lin(\mathcal{S})$ consisting of step occurrence sequences (in the sense that it can be reconstructed from $lin(\mathcal{S})$ through intersection, as shown in [7]). Second the set $lin(\mathcal{S})$ can express arbitrary concurrency relations between transition occurrences of a PTI-net, since concurrency equals the possibility of synchronous and sequential occurrence in any order.

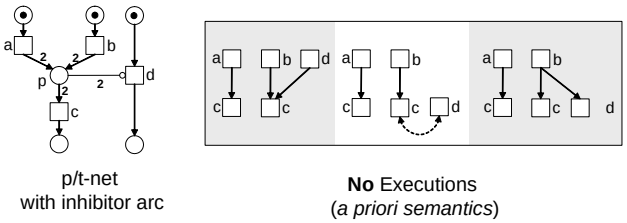

Figure 4. LSOs, not enabled.

It is easy to check that the LSOs from Figure 1 are indeed enabled LSOs w.r.t. the shown PTI-net. As a further example consider the three LSOs shown in Figure 4: In all three cases, the step sequence $(a+b)(c+d)c$ is a linearization of the LSO, but it is not an enabled step occurrence sequence. This is because after the execution of $(a+b)$ the place p carries four tokens disabling the following step $(c+d)$. Therefore, all three LSOs are not enabled w.r.t. the shown marked PTI-net.

Observe that there is no efficient test of definition 7 since there may be exponential many sequences of transition steps in the number of nodes linearizing the LSO.

3.2 Token flow property

In this subsection we extend the notions of token flow function and token flow property known for LPOs and p/t-

nets to the setting of PTI-nets. In [5] it is shown that LPOs are enabled if and only if they fulfill the token flow property w.r.t. a p/t-net. Our aim is to show the same for LSOs and PTI-nets.

Fix a marked PTI-net (N, m_0), $N = (P, T, F, W, I)$, a place p of N and an LSO $\mathcal{S} = (V, \prec, \sqsubset, l)$ with $l : V \to T$. Assume that $\mathcal{S}$ is enabled to occur w.r.t. (N, m_0) in the a priori semantics. Since the inhibitor relation I of (N, m_0) restricts the behavior of the underlying p/t-net $(N', m_0) = (P, T, F, W, m_0)$, $\mathcal{S}$ is then also enabled w.r.t. (N', m_0). In a p/t-net transitions which can be executed as one step also can be executed in arbitrary order. Therefore, also the LPO $\mathrm{lpo}_{\mathcal{S}} = (V, \prec, l)$ underlying $\mathcal{S}$ is enabled w.r.t. the p/t-net (N', m_0). Altogether we get that the enabledness of the LPO underlying $\mathcal{S}$ w.r.t. the p/t-net underlying (N, m_0) is a necessary condition for the enabledness of $\mathcal{S}$ w.r.t. (N, m_0).

Lemma 8. *Let $\mathcal{S} = (V, \prec, \sqsubset, l)$ be an LSO enabled w.r.t. a marked PTI-net (N, m_0), $N = (P, T, F, W, I)$, according to the a priori semantics. Then the LPO $(V, \prec, l)$ is enabled w.r.t. the marked p/t-net (P, T, F, W, m_0).*

That means the token flow property for $\mathcal{S}$ w.r.t. (N, m_0) should include the token flow property for $\mathrm{lpo}_{\mathcal{S}}$ w.r.t. (N', m_0). The token flow property for LPOs w.r.t. p/t-nets is based on the notion of *token flow functions*. For every place p a token flow function x_p assigns non-negative integers to the edges of an LPO. The value $x_p((v, v'))$ of an edge (v, v') is interpreted as the number of tokens which are produced by the transition $l(v)$ and consumed by the transition $l(v')$ in the place p. By this construction we still cannot specify the number of tokens which are consumed by a transition from the initial marking of a place, and the number of tokens which are produced by some transition in a place p, but not consumed by further transitions (and thus remain in the final marking). Therefore, we extend a considered LPO by an *initial event* which is interpreted as the occurrence of a transition producing the initial marking, and a *final event* which is interpreted as the occurrence of a transition consuming the final marking.

Definition 9 ($\star$-extension of LPOs)**.** *Let $\mathrm{lpo} = (V, \prec, l)$ be an LPO. Then an LPO $\mathrm{lpo}^\star = (V^\star, \prec^\star, l^\star)$, where $V^\star = (V \cup \{v_0, v_{\max}\})$, $v_0, v_{\max} \notin V$, $\prec^\star = \prec \cup (\{v_0\} \times V) \cup (V \times \{v_{\max}\}) \cup \{(v_0, v_{\max})\}$, $l^\star(v_0), l^\star(v_{\max}) \notin l(V)$, $l^\star(v_0) \neq l^\star(v_{\max})$ and $l^\star|_V = l$, is called* $\star$-extension of lpo.

By assigning natural numbers to the arcs of a $\star$-extension of an LPO we define a so called token flow function x of this LPO (with v_0 as its only smallest element and $v_{\max}$ as its only maximal element). It is clear that equally (with the same transition) labeled events should produce and consume the same overall number of tokens in a place. The overall number of tokens produced by an event v of $\mathrm{lpo}^\star = (V^\star, \prec^\star, l^\star)$ is called the *outtoken flow of v w.r.t. x* and is denoted and defined by $Out(v, x) = \sum_{v <^\star w} x((v, w))$. The overall number of tokens consumed by an event v of $\mathrm{lpo}^\star$ is called the *intoken flow of v w.r.t. x* and is denoted and defined by $In(v, x) = \sum_{w \prec^\star v} x((w, v))$.

Definition 10 (Token flow function of LPOs)**.** *Let $\mathrm{lpo} = (V, \prec, l)$ be an LPO and $\mathrm{lpo}^\star = (V^\star, \prec^\star, l^\star)$ be a $\star$-extension of lpo. A function $x : \prec^\star \to \mathbb{N}$ is called* token flow function *of lpo, if equally labeled nodes have equal intoken and outtoken flow* $(*)$ $\forall v, w \in V : l(v) = l(w) \implies (In(v, x) = In(w, x) \wedge Out(v, x) = Out(w, x))$.

An LPO $\mathrm{lpo} = (V, \prec, l)$ satisfies the *token flow property w.r.t. a marked p/t-net* if for each place p of this net there is a token flow function x_p compatible with p in the sense that its intoken and outtoken flows respect the weight function and the initial marking of the net as follows:

Definition 11 (Token flow property of LPOs)**.** *Let (N', m_0), $N' = (P, T, F, W)$, be a marked p/t-net and let $\mathrm{lpo} = (V, \prec, l)$ be an LPO with $l(V) = T$ and let $lpo^\star = (V^\star, \prec^\star, l^\star)$ be a $\star$-extension of lpo. Denote $W((l(v_0), p)) = m_0(p)$ for each place p. We say that lpo fulfills the* token flow property w.r.t. (N, m_0) *if the following statement holds: For every place $p \in P$ there exists a token flow function $x_p : \prec^\star \to \mathbb{N}$ such that*
(IN) $\forall v \in V : In(v, x_p) = W((p, l(v)))$
(OUT) $\forall v \in V \cup \{v_0\} : Out(v, x_p) = W((l(v), p))$.

In [5] we showed that an LPO fulfills the token flow property w.r.t. a marked p/t-net if and only if it is enabled w.r.t. this net.[2]

We now change back to the consideration of $\mathcal{S}$. Since the not later than relation of $\mathcal{S}$ does not describe the flow of tokens (since token flow always produces an earlier than relation between events), the token flow of $\mathcal{S}$ w.r.t. a place can be given by a token flow function of $\mathrm{lpo}_{\mathcal{S}}$. Clearly (as argued above), for each place there must be a token flow function satisfying (IN) and (OUT), if $\mathcal{S}$ is enabled.

The other way round the existence of such token flow functions x_p satisfying (IN) and (OUT) is not enough to ensure that $\mathcal{S}$ is enabled. This is because the execution of a prefix of $\mathcal{S}$ still might produce too many tokens in a place p (according to x_p) disabling a subsequent transition which tests this place via an inhibitor arc. Thus, we now require that token flow functions fulfill an additional property. This property should only allow token flow functions x_p according to which for each event the execution of one of its prefixes does not put too many tokens into p. In other words, each marking which is reachable through the execution of a prefix of some event should respect the inhibitor relations of the corresponding transition to all places.

[2] In particular, we showed that a token flow function satisfying (IN) and (OUT) w.r.t. a place abstracts from the individuality of conditions of a process of the net and encodes the flow relation of this process by natural numbers.

Assume that we have given a token flow function x_p on the edges of $\mathrm{lpo}^\star_S$ satisfying (IN) and (OUT) for some place p. We have to compute the number of tokens in this place after the execution of some prefix of $\mathcal{S}$. Let the prefix be defined by the set of events V'. On the one hand, by construction, the value of x_p on edges between events in V' correspond to tokens which are produced and consumed by events in this prefix. On the other hand, the value of x_p on edges from events in V' to events in $V \setminus V'$ corresponds to tokens which are produced by events in V' and remain in p after the execution of the prefix. Thus, the marking of the place after the execution of the prefix is given by the sum of the values of x_p on such edges. We define this sum for arbitrary token flow functions and call its value the *final marking* of the prefix w.r.t. the token flow function. Formally, the initial event of $\mathrm{lpo}^\star_S$ belongs to each prefix.

Definition 12 (Final marking)**.** *Let $\mathcal{S} = (V, \prec, \sqsubset, l)$ be an LSO and let $\mathcal{S}' = (V', \prec', \sqsubset', l')$ be a prefix of $\mathcal{S}$. Let further $x : V^\star \to \mathbb{N}$ be a token flow function of $(V, \prec, l)$ and let v_0 be the initial event of $(V^\star, \prec^\star, l^\star)$. The* final marking of $\mathcal{S}'$ (w.r.t. x) *is denoted and defined by $m_{\mathcal{S}'}(x) = \sum_{u \in V',\, v \notin V',\, u \prec^\star v} x((u,v)) + \sum_{v \notin V'} x((v_0, v))$.*

We are now ready to state the token flow property for LSOs w.r.t. PTI-nets: For each event the final marking of every of its prefixes must not exceed the weight of the inhibitor arc between the corresponding transition and place. Formally we also lift the notions of $\star$-extension and token flow function to LSOs.

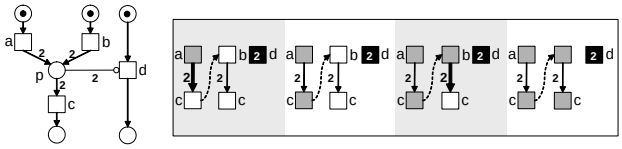

Figure 5. LSO with token flow function.

Definition 13 (Token flow property of LSOs)**.** *Let $\mathcal{S} = (V, \prec, \sqsubset, l)$ be an LSO. An LSO $\mathcal{S}^\star = (V^\star, \prec^\star, \sqsubset^\star, l^\star)$, where $(V^\star, \prec^\star, l^\star)$ is a $\star$-extension of the LPO $(V, \prec, l)$ and $\sqsubset^\star = \sqsubset \cup \prec^\star$, is called* $\star$-extension of $\mathcal{S}$. *A function $x : \prec^\star \to \mathbb{N}$ is called* token flow function of $\mathcal{S}$*, if it is a token flow function of $(V, \prec, l)$.*

Let further (N, m_0), $N = (P, T, F, W, I)$, be a marked PTI-net and let $l(V) = T$. We say that $\mathcal{S}$ fulfills the token flow property w.r.t. (N, m_0) *if the following statement holds: For every place $p \in P$ there exists a token flow function $x_p : \prec^\star \to \mathbb{N}$ satisfying (IN), (OUT) and*
(FIN) For all nodes $v \in V$ and all prefixes $\mathcal{S}'$ of v there holds: $m_{\mathcal{S}'}(x_p) \leqslant I((p, l(v)))$.

Figure 5 shows one of the executions from Figure 1 (four times) with annotated token flow function x_p w.r.t. the place p. Here, we omitted to draw the initial and maximal event and corresponding arcs, because there are no tokens in p in the initial marking and in the final marking after the execution of the LSO. The node labeled by a has intoken flow $0 = W((p,a))$ and outtoken flow $2 = W((a,p))$. The same holds for the node labeled by b. The nodes labeled by c have intoken flow $2 = W((p,c))$ and outtoken flow $0 = W((c,p))$. Finally the node labeled by d has intoken flow $0 = W((p,d))$ and outtoken flow $0 = W((d,p))$. Therefore, x_p satisfies (IN) and (OUT) w.r.t. p. To examine condition (FIN) we must only consider the node labeled by d (node filled by black color): The execution is depicted four times showing all prefixes (nodes filled by grey color) of this node. The arcs which count for the final marking of a prefix are highlighted: The first and third prefix have a final marking of $2 \leqslant 2 = I((p,d))$, the second and fourth prefix have a final marking of $0 \leqslant 2 = I((p,d))$. Thus, also (FIN) is satisfied. The maximum over all final markings of prefixes of the d-labelled node is displayed inside this node.

Observe that also the definition of the token flow property is inherent exponential in the size of the LSO since it involves in general exponential many prefixes of the LSO (condition (FIN)). Nonetheless, as will be explained in Section 4, the test of condition (FIN) can be transformed into a flow optimization problem which can be solved in polynomial time.

3.3 Enabledness vs. token flow property

In this Subsection we will prove the first main result of this paper given by the following Theorem. In the subsequent Subsection we will finally present a polynomial test of the token flow property as the second main result.

Theorem 14. *Let (N, m_0), $N = (P, T, F, W, I)$, be a marked PTI-net and let $\mathcal{S} = (V, \prec, \sqsubset, l)$ be an LSO with $l(V) = T$. Then $\mathcal{S}$ is enabled w.r.t. (N, m_0) if and only if it fulfills the token flow property w.r.t. (N, m_0).*

Proof. **only if**: Let $\mathcal{S}$ be enabled w.r.t. (N, m_0). Then, by Lemma 8 $(V, \prec, l)$ is enabled w.r.t. (P, T, F, W, m_0), that means for each $p \in P$ there is a token flow function $x_p : \prec^\star \to \mathbb{N}$ of $(V, \prec, l)$ satisfying (IN) and (OUT).

We claim, that each x_p also fulfills (FIN). For this let $v \in V$ and $\mathcal{S}'$ be a prefix of v defined by V'. By Lemma 4 there is a linearization $\mathcal{S}_{lin}$ of $\mathcal{S}$ such that V' also defines a prefix $\mathcal{S}'_{lin}$ of v w.r.t. $\mathcal{S}_{lin}$. There is a step occurrence sequence $\sigma = \tau_1 \dots \tau_n$ of (N, m_0) whose underlying LSO $\mathcal{S}_\sigma$ equals $\mathcal{S}_{lin}$. Since prefixes are downward $\sqsubset$-closed a prefix $\sigma' = \tau_1 \dots \tau_m$ $(m < n)$ of σ with $l(v) \in \tau_{m+1}$ must exist which corresponds to $\mathcal{S}'_{lin}$. In other words, the LSO $\mathcal{S}_{\sigma'}$ underlying σ' equals $\mathcal{S}'_{lin}$. It is enough to show now that $m(p) = m_{\mathcal{S}'}(x_p)$ for the marking m reached after the execution of σ', since $m(p) \leq I((p,t))$ for each place p and each transition $t \in \tau_{m+1}$ by Definition 6.

We finally compute: $m(p) = m_0(p) - \sum_{i=1}^{m} \sum_{t \in \tau_i} \tau(t)(W((p,t)) - W((t,p))) = Out(v_0, x_p) - \sum_{v \in V'} (In(v, x_p) - Out(v, x_p)) = \sum_{v \in V' \cup \{v_0\}} (\sum_{v \prec^\star w} x_p((v,w)) - \sum_{w \prec^\star v} x_p((w,v))) = m_{\mathcal{S}'}(x_p)$, since the values on edges within V' cancel each other out.

if: Let $\mathcal{S}$ fulfill the token flow property w.r.t. (N, m_0) and let x_p be a token flow function satisfying (IN), (OUT) and (FIN) w.r.t. the place p. Consider a sequence of transition steps $\sigma = \tau_1 \dots \tau_n$, whose underlying LSO $\mathcal{S}_\sigma$ is a linearization of $\mathcal{S}$. We have to show that σ is a step occurrence sequence of (N, m_0). For this, we show inductively that if $\sigma_k = \tau_1 \dots \tau_k$ is a step occurrence sequence then τ_{k+1} is a transition step enabled in the marking m reached after the execution of σ_k for $0 \leqslant k \leqslant n-1$.

First observe that σ is a step occurrence sequence of (P, T, F, W, m_0), since $(V, \prec, l)$ satisfies the token flow property on the p/t-net level and the *LPO* underlying σ clearly sequentializes $(V, \prec, l)$. That means the first condition of Definition 6 that $m(p) \geq \sum_{t \in \tau_{k+1}} \tau_{k+1}(t) W((p,t))$ for every place $p \in {\bullet}\tau_{k+1}$ is always satisfied. We still have to verify the condition of Definition 6 that $m(p) \leq I((p,t))$ for each place p and each transition $t \in \tau_{k+1}$. If $\mathcal{S}_{\sigma_k}$ is the LSO underlying σ_k, then $\mathcal{S}_{\sigma_k}$ is a prefix of $\mathcal{S}_\sigma$. Denoting $\mathcal{S}_{\sigma_k} = (V_k, \prec_k, \sqsubset_k, l_k)$, by Lemma 4, V_k also defines a prefix $\mathcal{S}_k$ of $\mathcal{S}$. Fix $t \in \tau_{k+1}$ and $p \in P$ and let $v \in V$ with $l(v) = t$ such that $\mathcal{S}_{\sigma_k}$ is a prefix of v. Then also (Lemma 4) $\mathcal{S}_k$ is a prefix of v. It is enough to show now that $m(p) = m_{\mathcal{S}_k}(x_p)$, since $m_{\mathcal{S}_k}(x_p) \leqslant I((p, l(v))$ by (FIN). The necessary computation is as above. □

4 Testing the token flow property

In this section we give a polynomial algorithm to test whether an LSO $\mathcal{S} = (V, \prec, \sqsubset, l)$ with $l(V) = T$ fulfills the token flow property w.r.t. a marked PTI-net (N, m_0). In the case that $\mathcal{S}$ fulfills the flow property, the algorithm constructs respective token flow functions for every place satisfying (IN), (OUT), and (FIN).

From [5] we have a polynomial test whether for each place there is a token flow function satisfying (IN) and (OUT). If such token flow functions do not exist, then clearly the LSO does not fulfill the token flow property. In the positive case the algorithm from [5] generates such token flow functions. We claim that either these token flow functions also satisfy (FIN) or the LSO does not fulfill the token flow property (i.e. there are no such token flow functions). This observation is based on the following lemma stating that the final marking of a prefix w.r.t. a token flow function x_p satisfying (IN) and (OUT) for p only depends on the initial making $m_0(p)$ and the arc weights $W((p,t))$ and $W((t,p))$ for $t \in T$, but not on the concrete distribution of the token flow given by x_p. This follows directly from the fact that the final marking can be computed as the marking reached after the execution of the prefix (the corresponding computation can be found in the proof of Theorem 14).

Lemma 15. *Let p be a place and let x_p be a token flow function satisfying (IN) and (OUT). Then it holds for each prefix $\mathcal{S}' = (V', \prec', \sqsubset', l')$ of $\mathcal{S}$: $m_{\mathcal{S}'}(x_p) = m_0(p) + \sum_{t \in T} |V'|_l(t)(W((t,p)) - W((p,t)))$.*

Thus, for different token flow functions x_p and x'_p satisfying (IN) and (OUT) for a place p the values $m_{\mathcal{S}'}(x_p)$ and $m_{\mathcal{S}'}(x'_p)$ coincide and thus either both fulfill (FIN) or both do not fulfill (FIN). It remains to test property (FIN) for the computed token flow functions x_p satisfying (IN) and (OUT). For this it is enough to compute for each node v the maximum of the values $m_{\mathcal{S}'}(x_p)$ over all prefixes $\mathcal{S}'$ of v and to compare this maximum with the value $I((p, l(v)))$.

Definition 16 (Inhibitor value)**.** *The* inhibitor value $Inh(v,x)$ *of an event v w.r.t. a token flow function x is defined by $Inh(v,x) = \max\{m_{\mathcal{S}'}(x) \mid \mathcal{S}' \text{ is prefix of } v\}$.*

A straightforward way to compute the inhibitor value of some node v is to enumerate all prefixes of this node and compute the final markings of all these prefixes according to Lemma 15. Unfortunately this is not efficient since there may be exponential many prefixes in the number of nodes (as already stated).

Another possible formalization of the problem is as follows: The final marking of a prefix is defined as the sum over the values of the token flow function on edges leaving the prefix. These edges separate the node set of the prefix from the subsequent nodes. Formally, this separation can be seen as a cut through $\mathcal{S}$ (resp. $\mathrm{lpo}_\mathcal{S}$) partitioning the nodes of $\mathcal{S}$ into two node sets. Such cuts are considered in flow theory and to avoid confusion we use the term *flow cuts* for this kind of cuts from now on. In flow theory one searches for maximal or minimal flows through flow networks with upper and/or lower capacities on edges. Thereto, one considers capacities of flow cuts. Interpreting $\mathrm{lpo}_\mathcal{S}$ as a flow network and the values of the token flow function as lower capacity bounds for flows through this network, the final marking of a prefix is given as the capacity of some flow cut and the inhibitor value of some node can be seen as the maximum capacity of flow cuts of the network. This maximum then can be computed efficiently through its correspondence to minimal flows (see the Appendix).

Definition 17 (Flow network, flow, flow cut, capacity)**.** *A* flow network (with lower capacities) *is a tuple (G, c, s, t) where $G = (W, E)$ is a directed graph, $c : E \to \mathbb{N}_0$ is a capacity function, $s \in W$ is a node with $\forall v \in W : (v, s) \notin E$ called* source *and $t \in W$ is a node with $\forall v \in W : (t, v) \notin E$ called* sink.

The capacity is interpreted as a lower bound for flows, that means a flow *is a function $f : E \to \mathbb{N}_0$ such that*

(a) $\forall (v,v') \in E : f((v,v')) \geqslant c((v,v'))$ *and (b)* $\forall v \in W : \sum_{(w,v)\in E} f((w,v)) = \sum_{(v,w)\in E} f((v,w))$. *The* value $|f|$ *of a flow* f *is defined as the outgoing flow of the source (or equivalently the ingoing flow of the sink)* $\sum_{(s,v)\in E} f((s,v))$. *A* minimal flow *is the flow with minimal value among all flows.*

A flow cut *is a tuple* $(S,T) \subseteq W \times W$ *such that* $s \in S$, $t \in T$, $S \cap T = \emptyset$ *and* $S \cup T = W$. *The capacity of a flow cut is defined by* $c((S,T)) = \sum_{v\in S,\, w\in T,\, (v,w)\in E} c((v,w))$ *if* $(T \times S) \cap E = \emptyset$ *and* $c((S,T)) = 0$ *else.*

In the following we describe how the inhibitor value of a node v can be interpreted as the maximal capacity of some flow cut in a flow network. For this we interpret, loosely speaking, $\mathcal{S}$ as a flow network. Therefore, we first have to omit the not later than relation. Clearly, we can glue events of $\mathcal{S}$ which are in a symmetric not later than relation. If $u \sqsubset v$ but $v \not\sqsubset u$, then there might be prefixes containing u but not v and there might be prefixes which contain or do not contain both events u and v together. Since the same holds if $u \prec v$, we replace remaining not later than relations by earlier than relations. An additional difficulty is that we do not want to consider all flow cuts of this flow network, but only those corresponding to prefixes of v. Therefore, we only consider (lower) capacity constraints on edges leaving some prefix of v.

Definition 18 (Associated flow network)**.** *Let* $\mathcal{S} = (V, \prec, \sqsubset, l)$ *be an LSO,* $v \in V$, $\mathcal{S}^\star = (V^\star, \prec^\star, \sqsubset^\star, l^\star)$ *be a* $\star$*-extension of* $\mathcal{S}$ *with initial event* v_0 *and maximal event* $v_{\max}$ *and* x *be a token flow function of* $\mathcal{S}$. *Let further* U *be the set of all nodes occurring in some prefix of* v. *Define the* flow network (G,c,s,t), $G=(W,E)$, associated to x and v *by*

- *For* $u \in V^\star$ *denote* $[u] = [u]_\sqsubset = \{w \in V^\star \mid w = u \vee (w \sqsubset^\star u \wedge u \sqsubset^\star w)\}$. *Define* $W = \{[u] \mid u \in V^\star\}$, $s = [v_0]$ *(*$= \{v_0\}$*) and* $t = [v_{\max}]$ *(*$= \{v_{\max}\}$*).*
- *Set* $E = \{([u],[w]) \mid u \sqsubset^\star w\}$.
- *Set* $c(([u],[w])) = \sum_{u'\in[u],\, w'\in[w],\, u'\prec^\star w'} x((u',w'))$ *if* $u \in U \wedge w \not\prec v$ *and* $c(([u],[w])) = 0$ *else.*

Observe that the associated flow network is well-defined, that means for $u' \in [u]$ and $w' \in [w]$ we have $u \sqsubset^\star w \Longrightarrow u' \sqsubset^\star w'$ and $c(([u],[w])) = c(([u'],[w']))$. The following lemma states that the final marking of prefixes can be computed by capacities of flow cuts in the associated flow network.

Lemma 19. *Let* $\mathcal{S}' = (V', \prec', \sqsubset', l')$ *be a prefix of a node* v. *Let further* x *be a token flow function of* $\mathcal{S}$ *and* (G,c,s,t), $G=(W,E)$, *be the flow network associated to* x *and* v. *Denote* $S = \{[v] \mid v = v_0 \vee v \in V'\}$ *and* $T = W \setminus S$. *Then* $m_{\mathcal{S}'}(x) = c((S,T))$.

Proof. Since $V' \subseteq U$ for the set U of all nodes occurring in some prefix of v we have for each $u \in V' \cup \{v_0\}$ and $w \notin V' \cup \{v_0\}$ that $c(([u],[w])) = \sum_{u'\in[u],\, w'\in[w],\, u'\prec^\star w'} x((u',w'))$. The statement is now an easy computation. Just observe that $(T \times S) \cap E = \emptyset$ since $w \not\sqsubset^\star u$ for $[u] \in S$, $[w] \in T$. □

Since flow cuts which do not correspond to prefixes of v do not have bigger capacities than flow cuts corresponding to such prefixes we get:

Theorem 20. *Let* v *be a node and* x *be a token flow function of an LSO* $\mathcal{S}$. *Let further* (G,c,s,t), $G=(W,E)$, *be the flow network associated to* x *and* v. *Then* $Inh(v,x) = \max\{c((S,T)) \mid (S,T)$ *flow cut of* $(G,c,s,t)\}$.

Proof. Let (S,T) be a flow cut of (G,c,s,t) which does not correspond to a prefix of v in the sense that $S \neq \{[u] \mid u = v_0 \vee u \in V'\}$ for each prefix $\mathcal{S}' = (V', \prec', \sqsubset', l')$ of v.

We first claim that if (S,T) does not correspond to a prefix of $\mathcal{S} = (V, \prec, \sqsubset, l)$ then $c((S,T)) = 0$ since there is $[u] \in S$ and $[w] \in T$ with $([w],[u]) \in E$. Indeed, in this case $V_S = \bigcup_{[u]\in S\setminus[v_0]} [u]$ does not define a prefix of $\mathcal{S}$. That means that there is $u \in V_S$ and $w \notin V_S$ with $w \sqsubset u$. By the definition of V_S it is not possible that also $u \sqsubset w$ (because then $[w] = [u]$). Therefore, by the definition of E we get $([w],[u]) \in E$.

Let finally (S,T) correspond to a prefix $\mathcal{S}' = (V', \prec', \sqsubset', l')$ of $\mathcal{S} = (V, \prec, \sqsubset, l)$ which is not a prefix of v. We claim that then there is a prefix $\mathcal{S}'' = (V'', \prec'', \sqsubset'', l'')$ of v such that $c((S,T)) \leqslant c((S'',T''))$ for the flow cut $S'' = \{[u] \mid u = v_0 \vee u \in V''\}$ and $T'' = W \setminus S''$.

Observe that the intersection and the union of two node sets defining two prefixes always defines a prefix again. This implies that there is a maximal prefix of v which is defined exactly by the set U of all nodes occurring in some prefix of v and that there is also a minimal prefix of v defined by the set $U' = \{u \in V \mid u \prec v\}$.

In particular, the intersection $V'' = V' \cap U$ defines a prefix $\mathcal{S}''$. Let $S'' = \{[v] \mid v = v_0 \vee v \in V''\}$ and $T'' = W \setminus S''$ be the corresponding flow cut. Then clearly $c((S,T)) \leqslant c((S'',T''))$ since $c(([u],[w])) = 0$ if $u \notin U$ and there may be edges $([u],[w]) \in E$ with $u \in V''$ and $w \in V' \setminus V''$ which only count in the second case. Thus, if $\mathcal{S}''$ is a prefix of v, we are done. Assume that $\mathcal{S}''$ is not a prefix of v. Then $V''' = V'' \cup U'$ defines a prefix of v and $c((S'',T'')) \leqslant c((S''',T'''))$ for $S''' = \{[v] \mid v = v_0 \vee v \in V'''\}$ and $T''' = W \setminus S'''$ (since $c(([u],[w])) = 0$ if $w \prec v$ and there may be edges $([u],[w]) \in E$ with $u \in V''' \setminus V''$ which only count in the second case). □

Thus inhibitor values can be computed through the maximal capacity of a flow cut in the associated flow network. This maximal capacity equals the minimal flow through this network. The proof for this statement is analogous to the

proof of the better known *maximal flow equals minimal cut* theorem of Ford/Fulkerson [3] in flow networks with upper capacities. As for maximal flows in flow networks with upper capacities there are polynomial algorithms to compute minimal flows in flow networks with lower capacities running in $O(n^3)$ time where n is the number of nodes of the flow network resp. the given LSO (we give a short explanation of the main arguments in the Appendix).

If p is a place for which there is a token flow function of the given LSO satisfying (IN) and (OUT) then the inhibitor value w.r.t. this token flow function must be computed for each node of the LSO. Altogether, the polynomial test of the token flow property takes $O(|P|n^4)$ time and looks formally as follows:

1: $test \leftarrow true$
2: **for all** ($p \in P$) **do**
3: **if** ($(V, \prec, l)$ does not fulfill token flow property w.r.t. (P, T, F, W, m_0) and p) **then**
4: $test \leftarrow false$
5: **else**
6: $x_p \leftarrow$ token flow function of $\mathcal{S}$ satisfying (IN) and (OUT) w.r.t. p
7: **for all** ($v \in V$) **do**
8: $(G, c, s, t) \leftarrow$ flow network associated to x_p and v
9: $M \leftarrow$ value of a minimal flow in (G, c, s, t)
10: **if** ($M > I((p, l(v)))$) **then**
11: $test \leftarrow false$
12: **end if**
13: **end for**
14: **end if**
15: **end for**
16: **return** test

Algorithm 1: Tests whether $\mathcal{S} = (V, \prec, \sqsubset, l)$ fulfills the token flow property w.r.t. (N, m_0), $N = (P, T, F, W, I)$.

5 Conclusion

We defined executions of PTI-nets w.r.t. the a priori semantics as enabled LSOs. This definition of enabled LSOs is a proper generalization of the definition of enabled LPOs and allows the representation of executions of PTI-nets with minimal causal dependencies between transition occurrences. As the first main result we showed that enabled LSOs can be characterized through the so called token flow property which we lifted from LPOs to LSOs, thus establishing a part of the semantical framework of p/t-net-executions to PTI-nets (Figure 2). As the second main result we developed a polynomial test of the token flow property.

These results are also valid for the a posteriori semantics of PTI-nets. In this case, the test of inhibitor conditions need not precede the execution of transitions, therefore synchronicity and concurrency are not distinguished and executions are represented by enabled LPOs. Nonetheless, the generalized notion of the token flow property (definition 13) can be used for such LPOs, where in condition (FIN) classical prefixes of LPOs and a modified notion of final markings corresponding to the a posteriori occurrence rule must be considered. Then the equivalence of enabledness and token flow property follows by construction. For the efficient test of the token flow property one needs to consider a modified flow network to compute the inhibitor value of nodes.

References

[1] J. Billington. Protocol specification using p-graphs, a technique based on coloured petri nets. In W. Reisig; G. Rozenberg [12], pages 293–330.

[2] S. Donatelli and G. Franceschinis. Modelling and analysis of distributed software using gspns. In W. Reisig; G. Rozenberg [12], pages 438–476.

[3] L. Ford and D. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8:399–404, 1956.

[4] A. Goldberg and S. Rao. Beyond the flow decomposition barrier. *Journal of the ACM*, 45(5):783–797, 1998.

[5] G. Juhás, R. Lorenz, and J. Desel. Can i execute my scenario in your net?. In G. Ciardo and P. Darondeau, editors, *ICATPN*, volume 3536 of *Lecture Notes in Computer Science*, pages 289–308. Springer, 2005.

[6] A. Kiehn. On the interrelation between synchronized and non-synchronized behaviour of petri nets. *Elektronische Informationsverarbeitung und Kybernetik*, 24(1/2):3–18, 1988.

[7] H. C. M. Kleijn and M. Koutny. Process semantics of general inhibitor nets. *Inf. Comput.*, 190(1):18–69, 2004.

[8] V. Malhotra, M. Kumar, and S. Maheshwari. An $O(|V|^3)$ algorithm for finding maximum flows in networks. *Information Processing Letters*, 7(6):277–278, 1978.

[9] J. Peterson. *Petri Net Theory and the Modeling of Systems*. Prentice-Hall, 1981.

[10] J. van Leeuwen, editor. *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*. Elsevier and MIT Press, 1990.

[11] W. Vogler. *Modular Construction and Partial Order Semantics of Petri Nets.*, volume 625 of *Lecture Notes in Computer Science*. Springer, 1992.

[12] W. Reisig; G. Rozenberg, editor. *Lectures on Petri Nets II: Applications, Advances in Petri Nets, the volumes are based on the Advanced Course on Petri Nets, held in Dagstuhl, September 1996*, volume 1492 of *Lecture Notes in Computer Science*. Springer, 1998.

Appendix (only for the reviewing procedure)

We finally present briefly the ideas how to prove a *minimal flow equals maximal cut* theorem in flow networks with lower capacities and how to compute efficiently such minimal flows.

It is easy to see that the following statements are equivalent for flow networks with lower capacities:

(i) f is a minimal flow.

(ii) There is no flow reducing path in the *residual network w.r.t.* f.

(iii) There is a flow cut (S,T) with $c((S,T)) = |f|$.

Here the residual network (G, c_f, s, t), $G = (W, E_f)$, w.r.t. f is defined as follows: For $(v,v') \in E$ define $c_f((v,v')) = f((v,v')) - c((v,v'))$ and denote $E_f = \{(v,v') \in W \times W \mid ((v,v') \in E \wedge c_f((v,v')) > 0) \vee ((v',v) \in E)\}$. A *flow reducing path w.r.t. a flow* f in the residual network is a simple path from source to sink in the residual network w.r.t. f.

Clearly, if f is a minimal flow then there is no flow reducing path in the residual network w.r.t. f. This is because along a flow reducing path the flow f can be reduced as follows: for edges $(v,v') \in E$, if (v,v') belongs to the path then reduce the flow on this edge by 1, if (v',v) belongs to the path then augment the flow on this edge by 1. Then by construction the reduced flow still satisfies the capacity constraint (a) and also constraint (b) since either the flow ingoing (outgoing) a node is once reduced and once augmented by 1 (along the path) or ingoing and outgoing flow of a node are both reduced or both augmented by 1 (along the path). Moreover $|f|$ is reduced by 1 since this is the case for the flow ingoing the sink.

If there is no flow reducing path in the flow network w.r.t. f then we define a flow cut (S,T) as follows: $S = \{w \in W \mid$ there is a simple path from s to w in the residual network$\}$ and $T = W \setminus S$. Then $f((u,v)) = c((u,v))$ for each edge $(u,v) \in E \cap (S \times T)$ (otherwise $v \in S$) and $E \cap (T \times S) = \emptyset$ (otherwise $u \in S$ for $(u,v) \in E \cap (T \times S)$). It is easy to see that $|f| = \sum_{e \in E \cap (S \times T)} f(e) - \sum_{e \in E \cap (T \times S)} f(e)$ (for each flow cut (S,T)). This gives $c((S,T)) = |f|$.

Finally, if there is a flow cut (S,T) with $|f| = c((S,T))$ then f must be minimal since $|f| = \sum_{e \in E \cap (S' \times T')} f(e) - \sum_{e \in E \cap (T' \times S')} f(e) \geqslant c((S',T'))$ for all flow cuts (S',T') (because $c((S',T')) = 0$ in the case $E \cap (T' \times S') \neq \emptyset$).

In particular, $|f| = c((S,T))$ if and only if (S,T) is a flow cut with maximal capacity and f is a minimal flow in the flow network.

We end up with a polynomial algorithm for the computation of minimal flows in flow networks with lower capacities therewith offering the possibility to efficiently compute the maximal capacity of flow cuts in such flow networks: Compute an arbitrary (feasible) flow of the flow network satisfying the lower capacity constraint (a) by a transformation into a maximal flow problem ([10]). There are maximal flow algorithms running in $O(n^3)$ time and faster ([8]) where n is the number of nodes of the LSO. Then iteratively reduce this flow along (shortest) flow reducing paths. This takes again maximal $O(n^3)$ time (a proof is analogous to the case of computing maximal flows in flow networks with upper capacities along so called flow augmenting paths ([3, 8])). This gives a overall time complexity of $O(n^3)$. See also [4] for an overview on flow theory and efficient algorithms.