Characterizing Security in Synchronization Graphs

Mark-Oliver Stehr
Universität Hamburg, Fachbereich Informatik
Vogt-Kölln-Straße 30, D-22527 Hamburg
stehr@informatik.uni-hamburg.de

Abstract: Synchronization graphs, marked graphs, or more precisely T-systems, constitute an important class of Petri nets well suited for the description and analysis of concurrent but deterministic synchronization schemes. T-systems are a well investigated model of net theory with a rich collection of theoretical results.

Nevertheless it has recently been found by the author that a fundamental problem concerning the nature of concurrency and causality has not been addressed so far, namely the question if there exists another binary relation between net elements not covered by concurrency and causality in safe and live T-systems. Here we will give a negative answer by proving that every two elements are either concurrent or causally dependent (tertium non datur).

Another issue that has not received much attention so far is the notion of security introduced by C. A. Petri with the motivation that it provides a more adequate abstraction of technical safety than the usual notion of safety in net theory. As an application of the first result on the nature of concurrency and causality we will prove that a surprisingly simple structural criterion for security proposed by C. A. Petri and C. Y. Yuan actually provides a characterization of security in safe and live T-systems.

Keywords: Petri nets, synchronization graphs, marked graphs, T-systems, causality, concurrency, safety, security

1 Introduction

Abstract notions of concurrency and causality have been studied by C. A. Petri in the context of concurrency theory [9] [10] [11]. Following [11] its structures are triples (X,li,co) where li and co are irreflexive, symmetric binary relations over a set X. The basic axioms include $co \cap li = \emptyset$ and $co \cup li \cup \mathrm{id}_X = X \times X$. The standard interpretations for li and co are (undirected) causality and concurrency, respectively, although further interpretations are suggested in [9].

Since the introduction of the first ideas of concurrency theory in [7] and [8] different sets of axioms, interactions between them, extensions and refinements of the theory have been studied [9] [2] [16] [10] [11] [1] [13] [6]. A general motivation was to provide a foundation for the theory of Petri nets and their processes in terms of more elementary concepts. In addition to the investigation of partial-order-based models [1] [2] where li is generated from a partial order by $li = (<) \cup (<)^{-1}$ there are also some results on models which do not rely on the assumption of an underlying partial order [17] [5] [6] [18]. As an example for the latter class we have studied concurrency structures associated with T-systems as models of concurrency theory [6].

Although definitions of causality and concurrency directly on T-systems instead of on their processes were originally motivated by our interest in T-system models of concurrency theory, they will play a different role in the present work: The binary relations of concurrency and causality will be used as a means to express important aspects of arbitrary safe and live T-systems.

The notion of security has been introduced for elementary net systems in [12], since safety has been found to be not an appropriate abstraction of technical safety if the objects modeled by tokens exhibit a spatial or temporal extension (as it is usually the case). Whereas safety is defined as the absence of *contacts*, security requires additionally the absence of *transjunctions*. Intuitively, a transjunction is a situation where "successive" tokens are not separated by an empty place. So security ensures that a well-defined separation between tokens is maintained explicitly in the net model. There are different ways to achieve security, either by net transformation or, more economically, by choosing an appropriate synchronization scheme as a basis of the system design. For the latter possibility a parameterized class of security structures, called cycloids, has been proposed in [14].

The present work is organized as follows: After some preliminaries fixing terminology and recapitulating folklore results about T-systems, we provide some facts about flow reversal in Section 3 that will be important for the proof of our main result. Next we provide definitions of causality (li) and concurrency (co) which are appropriate for safe and live T-systems. Formally, we will define irreflexive, symmetric relations $li, co \subseteq X \times X$, where X is the set of net elements. In Section 5 we prove the main result, namely that co and li do not only satisfy the equation $co \cap li = \emptyset$ but also $co \cup li \cup \mathrm{id}_X = X \times X$. Finally, in Section 6 this result is applied to prove a structural characterization of security in T-systems which is of remarkable simplicity.

2 Preliminaries

A possibly infinite *sequence* w is a function such that $\mathrm{dom}(w)$ is a subset of the integers and $i,k \in \mathrm{dom}(w) \wedge i < j < k \Rightarrow j \in \mathrm{dom}(w)$. The elements of $\mathrm{dom}(w)$ are called *indices* of w. We will use sequences together with the obvious definition of *concatenation* denoted by juxtaposition. Elements are conceived as singleton sequences.

As in [14] a *net* N is a triple (S,T,F) with $S \cap T = \emptyset$, $F \subseteq (S \times T) \cup (T \times S)$, $F \cap F^{-1} = \emptyset$, $\mathrm{dom}(F) \cup \mathrm{ran}(F) = S \cup T$. S, T, and F are the set of *places*, the set of *transitions*, and the *flow relation*, respectively. $X := S \cup T$ is also called the set of *net elements*. For a net element $x \in X$ the *preset* and *postset* of x are ${}^\bullet x := F^{-1}[x]$ and $x^\bullet := F[x]$, respectively. Two distinct transitions $t,t' \in T$ are said to be independent iff $({}^\bullet t \cup t^\bullet) \cap ({}^\bullet t' \cup t'^\bullet) = \emptyset$. Notice that the definition of a net excludes side-conditions, i.e. places in ${}^\bullet t \cap t^\bullet$, and isolated net elements, i.e. elements $x \in X$ with ${}^\bullet x \cup x^\bullet = \emptyset$.

A set $M \in \mathcal{P}(S)$ is called a *marking* of N. An element $x \in M$ is said to be *marked* at M. The occurrence relation $(\xrightarrow{t}) \subseteq \mathcal{P}(S) \times \mathcal{P}(S)$ for $t \in T$ is the smallest relation satisfying $M \cup {}^\bullet t \xrightarrow{t} M \cup t^\bullet$ for some marking M disjoint from ${}^\bullet t$ and $t^\bullet$. In case of $M \xrightarrow{t} M'$ we say that t is *(forward) enabled* at M, t is *backward enabled* at M' and M' is reachable from M by the occurrence of t. We also write $M \xrightarrow{w} M'$ for a sequence $w = (w_0 \ldots w_{n-1})$ iff there is a sequence $m = (m_0 \ldots m_n) = (M \ldots M')$ with $m_i \xrightarrow{w_i} m_{i+1}$ for all indices $i \in \{0 \ldots n{-}1\}$. Moreover, we define a relation $(\rightarrow) \subseteq \mathcal{P}(S) \times \mathcal{P}(S)$ by $M \rightarrow M' \Leftrightarrow \exists\, t \in T : M \xrightarrow{t} M'$ and an equivalence relation $(\leftrightarrow^*) := (\rightarrow \cup \rightarrow^{-1})^*$. For a marking M the set $\{M' \mid M \rightarrow^* M'\}$ is called the *forward case class* generated by M and $\{M' \mid M \leftrightarrow^* M'\}$ is the *full case class* generated by M.

T-nets are nets without branching places, i.e. $|{}^\bullet s|, |s^\bullet| \leq 1$ for all $s \in S$. An *(elementary) T-system* (N,C_0) is a T-net equipped with a marking C_0. The forward case class of (N,C_0) is the forward case class generated by C_0 and is denoted by $\mathcal{C}$.

Let N be a net and $\mathcal{M}$ be a set of markings. $\mathcal{M}$ is *(forward) safe* iff there is no marking $M \in \mathcal{M}$ and transition $t \in T$ such that t has a *(forward) contact* at M, i.e. ${}^\bullet t \subseteq M$ and $t^\bullet \cap M \neq \emptyset$. $\mathcal{M}$ is *backward safe* iff there is no marking $M \in \mathcal{M}$ and transition $t \in T$ such that t has a *backward contact* at M, i.e. $t^\bullet \subseteq M$ and ${}^\bullet t \cap M \neq \emptyset$. $\mathcal{M}$ is *(forward) live* iff for every marking $M \in \mathcal{M}$ and transition $t \in T$ there is a marking $M' \in \mathcal{M}$ with $M \rightarrow^* M'$ such that t is enabled at M'. $\mathcal{M}$ is *backward live* iff for every marking $M \in \mathcal{M}$ and transition $t \in T$ there is a marking $M' \in \mathcal{M}$ with $M' \rightarrow^* M$ such that t is backward enabled at M'. $\mathcal{M}$ has a *(forward) deadlock* iff there is a marking $M \in \mathcal{M}$ such that no transition is enabled at M.

Throughout this work we assume that (N,C_0) is a T-system with a net $N = (S,T,F)$ that has the following properties: $S \cup T$ is non-empty and finite, N is connected, i.e. $(F \cup F^{-1})^* = X \times X$.

A sequence w is said to be a *chain* iff $w_i \; F \; w_{i+1}$ for all indices i, $i{+}1 \in \mathrm{dom}(w)$. A chain w is *simple* iff $w_i \neq w_j$ for all indices $i \neq j$ of w. A finite chain $w = (w_0 \ldots w_{n-1})$ is a *cycle* iff it is nonempty and $w_{n-1} \; F \; w_0$. A *circuit* is a simple cycle. A sequence w *carries* $|\{i \in \mathrm{dom}(w) \mid w_i \in M\}|$ *tokens* at a marking M. Since N is a T-net, the number of tokens carried by a cycle/circuit is invariant under occurrence of transitions. We say that a sequence w is *marked* at M iff it carries at least one token at M. A cycle/circuit w is a *basic cycle/basic circuit* iff it carries exactly one token at C_0.

We will implicitly use the following folklore results, which can be adapted to our terminology e.g. from [3] or [4]: If $\mathcal{C}$ is safe and live then N is strongly connected. $\mathcal{C}$ is safe and live iff every circuit carries at least one token and every net element is contained in a basic circuit. If $\mathcal{C}$ is safe but not live then $\mathcal{C}$ has a deadlock.

The concept of basic cycle is usually not considered in the context of safe and live T-systems, since it is equivalent to the notion of basic circuit. This fact will be exploited in the proof of our main result.

Remark 2.1 If $\mathcal{C}$ is live then w is a basic circuit iff w is a basic cycle.

Proof Clearly every basic circuit is a basic cycle. It remains to prove the converse. Let w be a basic cycle. Assume w is not a circuit. Then there is a net element x occurring at least twice in w. W.l.o.g. we can assume that x is a transition, because N is a T-net. Now w has the form $(w'\ x\ w''\ x\ w''')$ and carries a single token. Hence, it can be decomposed into two cycles $(x\ w'')$ and $(x\ w'''\ w')$, but at most one of them can be marked, contradicting liveness. □

3 Flow Reversal

Flow reversal is a simple operation associating to each net $N = (S,T,F)$ the reverse net $N^{-1} := (S,T,F^{-1})$. In general, flow reversal can destroy important properties, however, it is rather well-behaved for safe and live T-systems, as demonstrated subsequently. This will enable us to use flow reversal as a technical means in the proof of our main result. Since the effect of flow reversal has not been treated in the Petri net literature, we have included the proofs. Remember that $\mathcal{C}$ is the forward case class of (N,C_0).

Remark 3.1 If $\mathcal{C}$ is live and (forward) safe then $\mathcal{C}$ is backward safe.

Proof Assume $t \in T$ has a backward contact at some case $C \in \mathcal{C}$. Since N is strongly connected (due to safety and liveness) we have $t^\bullet \subseteq C$ and there is some $s \in {}^\bullet t \cap C$. Again due to safety and liveness, s must be contained in a basic circuit w, but then w contains some $s' \in t^\bullet$. Now we have $s \neq s$, since N has no side-conditions, and $s,s' \in C$ contradicting the fact that w is a basic circuit. □

The next lemma states that backward occurrence can be simulated by forward occurrence in safe and live T-systems.

Lemma 3.2 Let $\mathcal{C}$ be safe and live and assume a marking $C \in \mathcal{C}$ and an arbitrary marking M.
Then $M \xrightarrow{t} C$ implies $C \rightarrow^* M$ for each $t \in T$.

Proof There is a finite sequence w of transitions such that $C \xrightarrow{w} C$ and w

contains each transition exactly once (see e.g. [3]). So we have $C \overset{(u\ t\ v)}{\to} C$ with finite sequences u and v. From $M \overset{t}{\to} C$ we obtain ${}^\bullet t \subseteq M$ and $t^\bullet \subseteq C$. We will first prove the claim that t and t' are independent for all t' occurring in v. Assume there is a transition t' in v such that t and t' are not independent. It follows that there is a place $s \in S$ such that $s \in {}^\bullet t \cap t'^\bullet$ or $s \in t^\bullet \cap {}^\bullet t'$. In the first case s is marked after the occurrence of t' and, since $s \in {}^\bullet t$ and t is not contained in v, s remains marked in final marking C. But then t would have a backward contact at C contradicting safety and liveness (see previous remark). In the second case s is marked after the occurrence of t. Then the token is removed by t' in v. Hence s is unmarked at C. But this contradicts $s \in t^\bullet \subseteq C$. So the claim about independence holds, and we can reshuffle $C \overset{(u\ t\ v)}{\to} C$ to obtain $C \overset{(u\ v\ t)}{\to} C$ by exchanging independent neighbors of the transition sequence. Due to $M \overset{t}{\to} C$ the last relation can be decomposed into $C \overset{(u\ v)}{\to} M \overset{t}{\to} C$. □

Remark 3.3 The forward case class $\mathcal{C}$ is equal to the full case class generated by C and is invariant under flow reversal.

Proof Equality follows from the previous lemma. Invariance under flow reversal is obvious from the definition of the full case class. □

By the previous remark the forward case class and full case class generated by C coincide. *Hence we will refer to them simply as the case class $\mathcal{C}$ in the remainder of this work.*

Lemma 3.4 If $\mathcal{C}$ is safe and live then for all $C,C' \in \mathcal{C}$ we have $C \to^* C'$.

Proof $C,C' \in \mathcal{C}$ implies $C_0 \to^* C$ and $C_0 \to^* C'$. By Lemma 3.2 we obtain $C \to^* C'$. □

Finally we observe that safety and liveness *together* are preserved by flow reversal.

Remark 3.5 If $\mathcal{C}$ is safe and live w.r.t. N then $\mathcal{C}$ is also safe and live w.r.t. N^{-1}.

Proof Safety of $\mathcal{C}$ w.r.t. N^{-1} follows from backward safety w.r.t. N which holds by Remark 3.1. Liveness of $\mathcal{C}$ w.r.t. N^{-1} follows from backward liveness w.r.t. N, which can be proved as follows: Due to liveness of $\mathcal{C}$ w.r.t. N for every marking $C \in \mathcal{C}$ and transition $t \in T$ there is a marking C' with $C \to^* C'$ such that t is enabled at C'. Let C'' be a marking with $C' \overset{t}{\to} C''$. Clearly, t is backward enabled at C''. Moreover, $C'' \to^* C$ because of $C,C'' \in \mathcal{C}$ and Lemma 3.4. □

For the remainder of this work we restrict our attention to *safe and live T-systems* (N,C_0), i.e. T-systems with a safe and live case class $\mathcal{C}$ generated by C_0.

4 Causality and Concurrency

In this section we will define causality and concurrency for T-systems using a simple technical device, namely T-splitting. In [14] T-splitting has been used to obtain security from safety. Here T-splitting allows us consider the original net N together with a refined version $\tilde{N}$ where the activity of transitions can be observed from the state. It may be helpful to think of N as a net where transitions can be marked (cf. [6]). Technically, the refined net will enable us to speak about net elements in a uniform way.

The *refined net* $\tilde{N} := (\tilde{S},\tilde{T},\tilde{F})$ obtained by *T-splitting* from N. It is defined by $\tilde{S} := \{\tilde{x} \mid x \in X\}$, $\tilde{T} := \{\acute{t}, \grave{t} \mid t \in T\}$ and $\tilde{F}$ being the smallest relation satisfying (a) $\acute{t}\ \tilde{F}\ \tilde{t}\ \tilde{F}\ \grave{t}$, (b) $\tilde{s}\ \tilde{F}\ \acute{t}$ if $s\ F\ t$, (c) $\grave{t}\ \tilde{F}\ \tilde{s}$ if $t\ F\ s$ for $s \in S$ and $t \in T$. This refinement induces a net morphism $\phi : \tilde{N} \to N$ mapping $\acute{x}$, $\tilde{x}$, $\grave{x}$ to x in the sense of [14]. For convenience we identify x and $\tilde{x}$ for all $x \in X$ throughout the work.

Observe that T-splitting does not change the behavior of a T-system essentially. More precisely, the *refined T-system* $(\tilde{N},\tilde{C}_0)$ has a safe and live case class $\tilde{\mathcal{C}}$ generated by $\tilde{C}_0$ which satisfies $\tilde{\mathcal{C}} \cap \mathcal{P}(S) = \mathcal{C}$. $\tilde{\mathcal{C}}$ is also called the *refined case class* of (N,C_0).

Now it is easy to introduce appropriate notions of causality and concurrency for T-systems: Let x and y be different net elements of N. x and y are *causally dependent* ($x\ li\ y$) iff there is a basic circuit of N containing both x and y. x and y are *concurrent* ($x\ co\ y$) iff there is a marking in the refined case class $\tilde{\mathcal{C}}$ containing both x and y. Intuitively, two net elements x and y are concurrent iff they *can* be active/marked concurrently. x and y are causally dependent on the other hand iff they *must* be active/marked in strict alternation.

Remark 4.1 li and co are invariant under flow reversal.

Proof Holds by definition for li and follows from Remark 3.3 for co. □

Actually, the relation li is the binary aspect of a more general view of T-systems as cyclic orders [19] but as demonstrated in [17] already a single binary relation, either li or co, is sufficient to determine the structure and dynamics of a rather general class of T-systems uniquely up to flow reversal.

5 The Main Result

We will first give a technical lemma prepared by the following proposition, which is of independent interest, since it conveys a fundamental aspect of the topological nature of safe and live T-systems: We can always reach one net element from another one via the flow relation such that there is a marking in the refined case class that has not to be passed.

Proposition 5.1 Let x and y be distinct net elements of N. Then there is a marking $\tilde{D} \in \tilde{\mathcal{C}}$ containing x and a simple chain $v = (x\ v'\ y)$ such that v' is unmarked at $\tilde{D}$.

Proof Due to liveness and strong connectedness there is a marking $\tilde{C} \in \tilde{\mathcal{C}}$ containing x. Let w be a basic circuit containing x. Consider the marking $\tilde{C}' := \tilde{C} - \{x\}$ and the refined case class $\tilde{\mathcal{C}}'$ generated by it. As w carries no token at $\tilde{C}'$, $\tilde{\mathcal{C}}'$ is not live. Hence, from $\tilde{C}'$ we can reach a deadlock at some marking $\tilde{D}' \in \tilde{\mathcal{C}}'$. Due to safety every transition has at least one empty input place at $\tilde{D}'$. Starting from y and proceeding along F^{-1} we can construct a backward-infinite chain $(u\ y)$ such that u carries no token at $\tilde{D}'$. Due to finiteness of X the infinite chain u must contain a circuit w.r.t. $\tilde{\mathcal{C}}'$. This unmarked circuit was impossible w.r.t. $\tilde{\mathcal{C}}$, because $\tilde{\mathcal{C}}$ is live. It follows that x must be contained in this circuit and consequently it is contained in u. Hence, $(u\ y)$ is of the form $(u'\ x\ v'\ y\ u'')$ and w.l.o.g. we can assume that the length of v' is minimal. Observe that $v = (x\ v'\ y)$ is a chain with v' being unmarked at $\tilde{D}'$. Due to minimality of v' and $x \neq y$ the chain v is simple, in particular x is not contained in v'. So v' is also unmarked at $\tilde{D} := \tilde{D}' \cup \{x\}$, which is a marking of $\tilde{\mathcal{C}}$. □

For the proof of the main result we will need a modification of the previous proposition. It is given by the following lemma.

Lemma 5.2 Let x and y be distinct net elements of N such that $\neg\ (x\ co\ y)$. Let $u = (x\ u'\ y)$ be a chain such that x is not contained in u' and let $\tilde{C} \in \tilde{\mathcal{C}}$ such that x and u' are marked at $\tilde{C}$. Then there is a marking $\tilde{D} \in \tilde{\mathcal{C}}$ containing x and a simple chain $v = (x\ v'\ y)$ such that v' is unmarked and u' is marked at $\tilde{D}$.

Proof Extend the previous proof as follows: Proceed as above using the supplied marking $\tilde{C} \in \tilde{\mathcal{C}}$. Clearly, u' is marked at $\tilde{C}' := \tilde{C} - \{x\}$. Observe that we cannot have an intermediate marking $\tilde{C}''$ containing y and reachable from $\tilde{C}'$ (in particular y is not marked at the deadlock $\tilde{D}'$). Otherwise $\tilde{C}'' \cup \{x\}$ would be a marking of $\tilde{\mathcal{C}}$ violating $\neg\ (x\ co\ y)$. Consequently, the number of tokens in u' cannot decrease while going from $\tilde{C}'$ to $\tilde{D}'$, as this is only possible via y. Hence u' remains marked at $\tilde{D}'$ and also at $\tilde{D} := \tilde{D}' \cup \{x\}$. □

Theorem 5.3 For net elements $x \neq y$ we have either $x\ li\ y$ or $x\ co\ y$ but not both.

Proof The fact that $x\ li\ y$ implies $\neg\ (x\ co\ y)$ follows immediately from the definitions of li and co. It remains to prove that $\neg\ (x\ co\ y)$ implies $x\ li\ y$. So assume $\neg\ (x\ co\ y)$. Due to strong connectedness x and y are contained in at least one cycle w. Let $u = (x\ u'\ y\ u'')$ be a cycle containing x and y carrying a minimum number of tokens. By liveness u carries at least one token. If u carries exactly one token then it is a basic cycle and also a basic circuit (see Remark 2.1) proving $x\ li\ y$. Otherwise u carries at least two tokens. Consider a marking $\tilde{C} \in \tilde{\mathcal{C}}$ containing x. As x and y cannot be marked simultaneously (here we use $\neg\ (x\ co\ y)$), u' or u'' is marked. First we consider the case where u' is marked: Observe that x is not contained in u', otherwise a cycle containing x and y carrying fewer tokens could be constructed. Now the preceding lemma provides a marking $\tilde{D} \in \tilde{\mathcal{C}}$ and a simple chain $(x\ v'\ y)$ such that v' is unmarked and u' remains marked at $\tilde{D}$. So we can construct a new cycle $(x\ v'\ y\ u'')$ which carries fewer tokens than u at $\tilde{D}$, contradicting the minimality assumption. Finally, we deal with the case where u'' is marked by considering the reverse T-system (N^{-1},C_0) instead of (N,C_0). This can be justified by the following facts: $\mathcal{C}$ is also the case class generated by C_0 w.r.t. N^{-1} (by Remark 3.3). $\mathcal{C}$ is safe and live (by Remark 3.5) w.r.t. N^{-1}. *co* is invariant under flow reversal (by Remark 4.1). The number of tokens carried by cycles is invariant under flow reversal. Exploiting these facts we can apply the same argument as in the previous case (replacing F by F^{-1} and exchanging u' and u'') to derive a contradiction. □

6 Characterizing Security

As an easy application of the previous theorem we obtain a security characterization that has already been proposed in [15] as a security criterion but without proving that it actually characterizes security in safe and live T-systems.

For a net N a set $\mathcal{M}$ of markings is said to be *secure* iff $\mathcal{M}$ is (forward) safe and backward safe and there is no marking $M \in \mathcal{M}$ and transition $t \in T$ such that t is in transjunction at M. Here a transition t is said to be in *transjunction* at M iff $^\bullet t \cap M \neq \emptyset \wedge t^\bullet \cap M \neq \emptyset$. In contrast to the definitions of contact the notion of transjunction is invariant under flow reversal.

Remember that $\mathcal{C}$ has been assumed to be safe and live and it coincides with the full case class. Indeed it is the full case class which has been used in the original definition of security in [12] and [14] for $\mathcal{M}$.

Corollary 6.1 $\mathcal{C}$ is secure iff for every $t \in T$ every pair $x \in {}^\bullet t$, $y \in t^\bullet$ is contained in a basic circuit.

Proof ($\Leftarrow$) Backward safety follows from safety by Remark 3.1. Moreover, $\mathcal{C}$ is obviously free of transjunctions due to the structural condition. ($\Rightarrow$) Let $t \in T$ and $x \in {}^\bullet t$, $y \in t^\bullet$. We have $x \neq y$, since N has no side-conditions. From security it follows that there is no marking $C \in \mathcal{C}$ with $x,y \in C$. By definition of *co* we obtain $\neg\ (x\ co\ y)$ and using Theorem 5.3 we conclude $x\ li\ y$. Now x and y are contained in some basic circuit according to the definition of *li*. □

From the non-trivial part of this corollary it follows that there is only one way to achieve security, namely by structural means of a very special kind.

7 Final Remarks

The main theorem about the structure of concurrency and causality presented in Section 5 has been derived as a by-product while investigating the close relationship between cyclic orders and T-systems [18]. Nevertheless we believe that this result is of independent interest due to its generality and its close connection to the notion of security. From the results of this work we conclude that security can be characterized structurally in terms of a single binary relation, either causality *or* concurrency.

From a foundational point of view these results shed new light on the local axioms of concurrency theory (cf. [17]), stating that the immediate neighborhood of each element consists of two *co*-equivalence classes related by *li*. Interpreting the two *co*-equivalence classes as the immediate past and the immediate future of an element, respectively, the local axioms require that the immediate past and immediate future of a single element are causally dependent. From this perspective security is just a reformulation of this natural assumption on the level of net systems.

References

[1] E. Best and C. Fernandez. *Nonsequential Processes—A Petri Net View*, volume 13 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1988.

[2] E. Best and A. Merceron. Concurrency axioms and D-continuous posets. In G. Rozenberg, editor, *Advances in Petri Nets 1984*, LNCS 188, pages 32–47. Springer-Verlag, 1985.

[3] F. Commoner, A. W. Holt, S. Even, and A. Pnueli. Marked directed graphs. *Journal of Computer and System Sciences*, 5:511–523, 1971.

[4] J. Desel and J. Esparza. *Free Choice Petri nets*. Number 40 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1995.

[5] O. Kummer. *Axiomensysteme für die Theorie der Nebenläufigkeit*. Logos, Berlin, 1996.

[6] O. Kummer and M.-O. Stehr. Petri's axioms of concurrency – A selection of recent results. In *Proc. 18th Int. Conf. on Appl. and Theory of Petri Nets*, LNCS 1248. Springer, 1997.

[7] C. A. Petri. Nicht-sequentielle Prozesse. ISF-Bericht ISF-76-6, third edition, GMD, St. Augustin, 1977.

[8] C. A. Petri. Concurrency as a basis of systems thinking. In F. V. Jensen, B. H. Mayoh, and K. K. Moller, editors, *Proc. from 5th Scandinavian Logic Symposium, Jan. 1979, Aalborg*, pages 143–162, Aalborg, 1979. Universitetsforlag.

[9] C. A. Petri. Concurrency. In *Net Theory and Applications – Proc. Adv. Course on General Net Theory of Processes and Systems*, LNCS 84, pages 251–260. Springer, 1980.

[10] C. A. Petri. Concurrency theory. In W. Brauer, W. Reisig, and G. Rozenberg, editors, *Advances in Petri Nets 1986*, LNCS 254, pages 4–24. Springer-Verlag, 1987.

[11] C. A. Petri. Concurrency Theory. Lecture notes, Universität Hamburg, Fachbereich Informatik, 1988.

[12] C. A. Petri. On technical safety and security. *Petri Net Newsletter*, 33:25–30, August 1989.

[13] C. A. Petri. Vollständige Signalordnung. Lecture notes, Universität Hamburg, Fachbereich Informatik, 1989.

[14] C. A. Petri. Nets, time and space. *Theoretical Computer Science*, 153(1–2):3–48, 1996.

[15] C. A. Petri and C. Y. Yuan. On technical safety and security (continued). *Petri Net Newsletter*, 35:8–15, April 1990.

[16] W. Reisig. A strong part of concurrency. In G. Rozenberg, editor, *Advances in Petri Nets 1987*, LNCS 266, pages 238–272. Springer-Verlag, 1987.

[17] M.-O. Stehr. Concurrency Theory of Cyclic and Acyclic Processes. Fachbereichsbericht FBI-HH-B-190/96, Univ. Hamburg, FB Informatik, 1996.

[18] M.-O. Stehr. System Specification by Cyclic Causality Contraints. Fachbereichsbericht FBI-HH-B-210/98, Univ. Hamburg, FB Informatik, 1998.

[19] M.-O. Stehr. Thinking in cycles. In *Proc. 19th Int. Conf. on Appl. and Theory of Petri Nets*, LNCS 1420. Springer-Verlag, 1998.

[6] and [17] are available via `http://www.informatik.uni-hamburg.de/TGI`.