Concurrent Automata vs. Asynchronous Systems

Rémi Morin

Laboratoire d'Informatique Fondamentale de Marseille,
39 rue F. Joliot-Curie, F-13453 Marseille cedex 13, France
remi.morin@lif.univ-mrs.fr

Abstract. We compare the expressive power of two automata-based finite-state models of concurrency. We show that Droste's and Kuske's coherent stably concurrent automata and Bednarczyk's forward-stable asynchronous systems describe the same class of regular event structures. This connection subsumes a previous study by Schmitt which relates Stark's trace automata to asynchronous systems. This work relies on Zielonka's theorem and some unrecognized result due to Arnold.

1 Introduction

In a seminal paper [12] Nielsen, Plotkin and Winskel introduced prime event structures as natural unfoldings of 1-safe Petri nets. This semantics can be decomposed into several steps by means of intermediate models which are prefix-closed Mazurkiewicz trace languages [9] and asynchronous systems [2]. More general automata-based models of concurrency were later related to more general event structures [19], namely trace automata [15] and concurrent automata [5]. Interestingly three classes of automata-based models are known to describe exactly prime event structures: Forward-stable asynchronous systems, stable trace automata, and the more general model of coherent stably concurrent automata.

More recently the problem of characterizing the unfoldings of *finite* concurrent automata has been investigated [14,18,11]. In [14], Schmitt established that all unfoldings of finite stable trace automata are also unfoldings of finite forward-stable asynchronous systems. In [18], Thiagarajan proved with the help of Zielonka's theorem [20] that all unfoldings of finite forward-stable asynchronous systems are also unfoldings of finite 1-safe Petri nets.

In this paper we improve both approaches and show that all unfoldings of finite coherent stably concurrent automata are unfoldings of finite 1-safe Petri nets. We proceed in two steps. With the help of some unrecognized difficult work by Arnold [1] we prove that if a prime event structure is the unfolding of a finite coherent stably concurrent automaton then it is also the unfolding of a finite coherent asynchronous system. Next we use Zielonka's theorem to establish that if a prime event structure is the unfolding of a finite coherent asynchronous system then it is also the unfolding of a finite forward-stable asynchronous system. This step is more technical so we sketch the construction in more details.

To simplify the presentation of this paper we consider particular domains as semantical objects instead of prime event structures. These domains are known to be equivalent to prime event structures so that we shall sketch in the conclusion how our results can be rephrased in that setting.

J. Jędrzejowicz and A. Szepietowski (Eds.): MFCS 2005, LNCS 3618, pp. 686–698, 2005.

2 Background and Results

In this section we present the main framework of this study and compare the contribution of this paper to some known results from the literature.

First we introduce the very general automata-based model of concurrency known as automata with concurrency relations [5]. The latter appear as a generalization of several other models such as asynchronous systems, trace automata, and Mazurkiewicz traces.

Definition 2.1. *An* automaton with concurrency relations *over the alphabet* Σ *is a structure* $\mathcal{A} = (Q, \imath, \Sigma, \longrightarrow, (\|_q)_{q\in Q})$ *such that*

1. Q *is a non-empty (possibly infinite) set of states, with an initial state* $\imath \in Q$*;*
2. $\longrightarrow \subseteq Q \times \Sigma \times Q$ *is a set of transitions;*
3. *if* $q \xrightarrow{a} q_1$ *and* $q \xrightarrow{a} q_2$ *then* $q_1 = q_2$*;*
4. $(\|_q)_{q\in Q}$ *is a family of binary, irreflexive, and symmetric relations on* Σ*;*
5. *if* $a\|_q b$ *then there exist* $q \xrightarrow{a} q_1$, $q \xrightarrow{b} q_2$, $q_1 \xrightarrow{b} q_3$ *and* $q_2 \xrightarrow{a} q_3$.

We say that $\mathcal{A}$ *is* finite *if* Q *and* Σ *are finite.*

The *language* $L(\mathcal{A})$ *of sequential computations* of $\mathcal{A}$ consists of all words $u = a_1...a_n \in \Sigma^\star$ for which there are some states $q_0, ..., q_n \in Q$ such that $\imath = q_0$ and for each $i \in [1, n]$, $q_{i-1} \xrightarrow{a_i} q_i$. For short, these conditions will be denoted by $q_0 \xrightarrow{u} q_n$. Now the independence relations $\|_q$ yield a natural equivalence relation over the set of sequential computations $L(\mathcal{A})$ as follows. The *trace equivalence* $\sim_{\mathcal{A}}$ associated with $\mathcal{A}$ is the least equivalence over $L(\mathcal{A})$ such that for all words $u, v \in \Sigma^\star$ and all actions $a, b \in \Sigma$ if $\imath \xrightarrow{u} p \xrightarrow{ab} q \xrightarrow{v} r$ and $a\|_p b$ then $u.ab.v \sim_{\mathcal{A}} u.ba.v$. Conditions 3 and 5 ensure that if w and w' are two trace equivalent words then they lead from the initial state to the same state.

For any word $u \in L(\mathcal{A})$, the trace $[u]$ consists of all words $v \in L(\mathcal{A})$ that are trace equivalent to u: Formally we put $[u] = \{v \in \Sigma^\star \mid v \sim_{\mathcal{A}} u\}$. The *trace language* $\mathcal{L}(\mathcal{A}) = L(\mathcal{A})/\sim_{\mathcal{A}}$ consists of all traces. The latter are partially ordered in the following way: We put $[u] \sqsubseteq [v]$ if there exists some word $z \in \Sigma^\star$ such that $u.z \sim_{\mathcal{A}} v$. The *trace domain* of $\mathcal{A}$ is the partial order $(\mathcal{L}(\mathcal{A}), \sqsubseteq)$.

Example 2.2. Let $D = \{(n, m) \in \mathbb{N}^2 \mid n \leqslant m\}$ be equipped with the partial order $\sqsubseteq$ for which $(n, m) \sqsubseteq (n', m')$ if $n \leqslant n'$ and $m \leqslant m'$. Then $(D, \sqsubseteq)$ is (isomorphic to) the trace domain of the automaton with concurrency relations $\mathcal{A} = (D, (0, 0), \{a, b\}, \longrightarrow, (\|_q)_{q\in Q})$ where $(n, m) \xrightarrow{a} (n', m')$ if $n' = n + 1$ and $m' = m$; $(n, m) \xrightarrow{b} (n', m')$ if $n' = n$ and $m' = m + 1$; and $a\|_{(n,m)} b$ if $n < m$. Noteworthy it is easy to see that no *finite* automaton with concurrency relations admits the partial order $(D, \sqsubseteq)$ as trace domain.

2.1 Coherent Stably Concurrent Automata

In the literature, a particular attention was devoted to automata whose concurrency relations $\|_q$ depend locally on each other. In the two following definitions, for all actions $a, b, c \in \Sigma$ and all states q, we write $a\|_{q.c} b$ if there exists a state $q' \in Q$ such that $q \xrightarrow{c} q'$ and $a\|_{q'} b$.

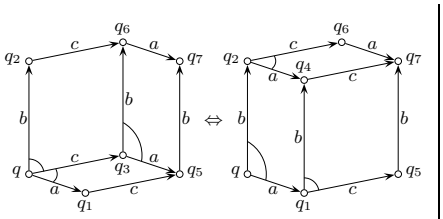

Fig. 1. Stably concurrent automata

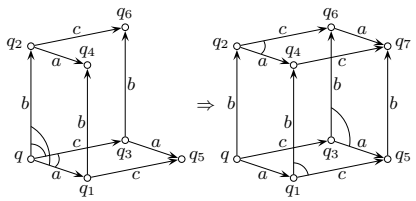

Fig. 2. Coherence property

Definition 2.3. *An automaton with concurrency relations $\mathcal{A}$ is called a* stably concurrent automaton *if for all states $q \in Q$ and all actions $a, b, c \in \Sigma$:*

$$a\|_q c \wedge b\|_q c \wedge a\|_{q.c} b \text{ if and only if } a\|_q b \wedge b\|_{q.a} c \wedge a\|_{q.b} c$$

This requirement is depicted in Fig. 1. In this paper we are interested in stably concurrent automata that satisfy an additionnal coherence condition that ressembles the requirement (C) of the generalized trace languages from [13].

Definition 2.4. *A stably concurrent automaton is* coherent *if for all states $q \in Q$ and all actions $a, b, c \in \Sigma$: $a\|_q b \wedge a\|_q c \wedge b\|_q c$ implies $a\|_{q.c} b$.*

This requirement is depicted in Fig. 2. The trace language of such a coherent stably concurrent automaton can be viewed as a generalized trace language [13]. Consequently, it corresponds also to a labeled event structure with a binary conflict. As we will explain in the conclusion, most results used or established in this paper can be rephrased and applied in the framework of event structures.

2.2 Forward-Stable Asynchronous Systems

Automata with concurrency relations are a generalization of several other models of concurrency, in particular Bednarczyk's asynchronous automata [2] and Stark's trace automata [15]. These models are known to have close relationships to event structures, too. As opposed to automata with concurrency relations, both models involve a single independence relation.

Definition 2.5. *Let Σ be some alphabet and $\|$ be a binary, symmetric, and irreflexive relation over Σ. Let $\mathcal{A} = (Q, \imath, \Sigma, \longrightarrow, \|)$ be a structure satisfying Conditions 1, 2, and 3 of Definition 2.1. Then $\mathcal{A}$ is called an* asynchronous system *[2] if we have*

ID*: $q_1 \xrightarrow{a} q_2 \wedge q_2 \xrightarrow{b} q_3 \wedge a\|b$ implies $q_1 \xrightarrow{b} q_4 \wedge q_4 \xrightarrow{a} q_3$ for some $q_4 \in Q$.*
On the other hand, $\mathcal{A}$ is called a trace automaton *[15] if we have*

FD*: $q_1 \xrightarrow{a} q_2 \wedge q_1 \xrightarrow{b} q_3 \wedge a\|b$ implies $q_2 \xrightarrow{b} q_4 \wedge q_3 \xrightarrow{a} q_4$ for some $q_4 \in Q$.*

Finally, if $\mathcal{A}$ satisfies both conditions ID *and* FD *then it is called a* forward-stable *asynchronous system [2].*

Asynchronous systems and trace automata can be seen as automata with concurrency relations by putting $a \|_q b$ if $a \| b$, $q \xrightarrow{a} q_1 \xrightarrow{b} q_2$, and $q \xrightarrow{b} q_3 \xrightarrow{a} q_2$. Consequently they are associated with a set of sequential computations $L(\mathcal{A})$, a trace language $\mathcal{L}(\mathcal{A})$, and a trace domain $(\mathcal{L}(\mathcal{A}), \sqsubseteq)$.

Observe that any trace automaton satisfies the coherence property (Fig. 2). Moreover it fulfills also half of the requirement to be a stably concurrent automaton (Fig. 1) namely the implication from left to right. We say that a trace automaton is *stable* if it is a stably concurrent automaton.

2.3 Comparisons of Expressive Power

Observe now that any asynchronous automaton is a stably concurrent automaton. Furthermore, it is coherent as soon as it is forward-stable. Thus the trace domain of any forward-stable asynchronous automaton is obviously the trace domain of a coherent stably concurrent automaton. The next theorem expresses the converse property.

THEOREM 2.6. *For any coherent stably concurrent automaton $\mathcal{A}$, there exists a forward-stable asynchronous system $\mathcal{A}'$ such that the trace domains $(\mathcal{L}(\mathcal{A}), \sqsubseteq)$ and $(\mathcal{L}(\mathcal{A}'), \sqsubseteq)$ are isomorphic.*

This theorem summarizes some results from [2,8]. In [2] it is shown that the trace domains of forward-stable asynchronous systems can be identified with event structures by means of a coreflection between the two models. This connection can be extended to coherent stably concurrent automata [8].

Consider now again the trace domain $(D, \sqsubseteq)$ of Example 2.2. As explained above, this partial order is isomorphic to the trace domain of some coherent stably concurrent automaton over the finite alphabet $\{a, b\}$. However, as noticed by Husson [7], any asynchronous automaton whose trace domain is isomorphic to $(D, \sqsubseteq)$ admits some *infinite* alphabet. This shows that Theorem 2.6 fails if one considers finite alphabets only.

In this paper, we focus on finite automata. We show in Corollary 4.6 that Theorem 2.6 remains valid if we restrict to *finite* stably concurrent automata. Since stable trace automata are coherent stably concurrent automata, Corollary 4.6 subsumes the following theorem due to Schmitt.

THEOREM 2.7. *[14, Th. 3.12] For any finite stable trace automaton $\mathcal{A}$, there exists a finite forward-stable asynchronous system $\mathcal{A}'$ such that the trace domains $(\mathcal{L}(\mathcal{A}), \sqsubseteq)$ and $(\mathcal{L}(\mathcal{A}'), \sqsubseteq)$ are isomorphic.*

Organization of the Paper. In the rest of this paper we consider only *finite* alphabets. In the following section we relate stably concurrent automata to the theory of regular consistent sets of pomsets [1] and Mazurkiewicz traces [4]. This first step allows us to transform a finite stably concurrent automaton into a finite asynchronous system with the same trace domain. Moreover this process preserves coherence. Next we state our main result (Cor. 4.6) and sketch its proof by means of Zielonka's theorem [20]. This second step shows how to build

a forward-stable finite asynchronous system from a coherent one while preserving the trace domain. In order to improve [14] and [18], the main difficulty here is to ensure Property FD from the assumption of coherence. In the conclusion we explain how this work applies to the setting of regular event structures [18].

3 Consistent Sets of Pomsets

A *pomset* over an alphabet Σ is a triple $t = (E, \preccurlyeq, \xi)$ where $(E, \preccurlyeq)$ is a finite partial order and ξ is a mapping from E to Σ *without autoconcurrency*: $\xi(x) = \xi(y)$ implies $x \preccurlyeq y$ or $y \preccurlyeq x$ for all $x, y \in E$. A pomset can be seen as an abstraction of an execution of a concurrent system. In this view, the elements e of E are *events* and their label $\xi(e)$ describes the action that is performed in the system by the event $e \in E$. Furthermore, the order $\preccurlyeq$ describes the causal dependence between events. We denote by $\mathbb{P}(\Sigma)$ the class of all pomsets over Σ.

An *order extension* of a pomset $t = (E, \preccurlyeq, \xi)$ is a pomset $t' = (E, \preccurlyeq', \xi)$ such that $\preccurlyeq \subseteq \preccurlyeq'$. A *linear extension* of t is an order extension that is linearly ordered. It corresponds to a sequential view of the concurrent execution t. Linear extensions of a pomset t over Σ can naturally be regarded as words over Σ. By $\mathrm{LE}(t) \subseteq \Sigma^\star$, we denote the set of linear extensions of a pomset t over Σ. For any subset of pomsets $\mathcal{L} \subseteq \mathbb{P}(\Sigma)$, we put $\mathrm{LE}(\mathcal{L}) = \bigcup_{t \in \mathcal{L}} \mathrm{LE}(t)$.

Two isomorphic pomsets admit the same set of linear extensions. Noteworthy the converse property holds [17]: If $\mathrm{LE}(t) = \mathrm{LE}(t')$ then t and t' are two isomorphic pomsets. In the sequel of this paper we do not distinguish between isomorphic pomsets any longer because they are used as representative of sets of words. In particular, $\mathrm{LE}(t) = \mathrm{LE}(t')$ implies $t = t'$.

An *ideal* of a pomset $t = (E, \preccurlyeq, \xi)$ is a subset $H \subseteq E$ such that $x \in H \wedge y \preccurlyeq x \Rightarrow y \in H$. The restriction $t' = (H, \preccurlyeq \cap (H \times H), \xi \cap (H \times \Sigma))$ is then called a *prefix* of t and we write $t' \leqslant t$. For any set of pomsets $\mathcal{L}$, $\mathrm{Pref}(\mathcal{L})$ denotes the set of prefixes of pomsets from $\mathcal{L}$. We say that $\mathcal{L}$ is *prefix-closed* if $\mathrm{Pref}(\mathcal{L}) = \mathcal{L}$.

3.1 Regular Consistent Sets of Pomsets

We borrow now the notion of consistent sets of pomsets from [1]. Although we deal mainly with prefix-closed sets of pomsets, we present here the general definition of a regular consistent set of pomsets. Intuitively, consistency means that concurrency is determined by any sequential ordering of events.

Definition 3.1. *A set of pomsets $\mathcal{L}$ is called* consistent *if*

$$\forall t_1, t_2 \in \mathrm{Pref}(\mathcal{L}) : \mathrm{LE}(t_1) \cap \mathrm{LE}(t_2) \neq \emptyset \Rightarrow t_1 = t_2.$$

Let $\mathcal{L}$ be a consistent set of pomsets. The *pomset equivalence* $\sim_{\mathcal{L}}$ over $\mathrm{LE}(\mathcal{L})$ is such that $w \sim_{\mathcal{L}} w'$ if and only if $\{w, w'\} \subseteq \mathrm{LE}(t)$ for some $t \in \mathcal{L}$. Note that $\sim_{\mathcal{L}}$ is an equivalence relation over $\mathrm{LE}(\mathcal{L})$ because $\mathcal{L}$ is consistent.

For any two words $w, w' \in \Sigma^\star$, we put $w \equiv_{\mathcal{L}} w'$ if for all words u,$v \in \Sigma^\star$ it holds: $w.u \sim_{\mathcal{L}} w.v \Leftrightarrow w'.u \sim_{\mathcal{L}} w'.v$. It is easy to see that $\equiv_{\mathcal{L}}$ is a right-congruence over $\Sigma^\star$. Relation $\equiv_{\mathcal{L}}$ appeared in [1] in the definition of regular, complete, consistent, and prefix-closed sets of pomsets.

Definition 3.2. *A consistent set of pomsets $\mathcal{L}$ is* regular *if $\equiv_{\mathcal{L}}$ is of finite index.*

Regularity satisfies several natural properties. In particular if $\mathcal{L}$ is a regular consistent set of pomsets then $\mathrm{Pref}(\mathcal{L})$ is consistent and regular, too.

3.2 Relationships with Stably Concurrent Automata

The connection between prefix-closed consistent sets of pomsets and stably concurrent automata originates from a pomset description of traces.

Theorem 3.3. *[3, Th. 4.6] Let $\mathcal{A}$ be a stably concurrent automaton over Σ. Each trace $[u] \in \mathcal{L}(\mathcal{A})$ is the set of linear extensions of a (unique) pomset.*

This result shows that the trace language $\mathcal{L}(\mathcal{A})$ of a stably concurrent automaton can be represented by a set of pomsets. *We adopt this dual view in the rest of this paper.* Clearly $L(\mathcal{A}) = \mathrm{LE}(\mathcal{L}(\mathcal{A}))$ and $\sim_{\mathcal{A}} = \sim_{\mathcal{L}(\mathcal{A})}$. Noteworthy $[u] \sqsubseteq [v]$ means in this setting that the pomset $[u]$ is a prefix of the pomset $[v]$. Moreover the trace language of any stably concurrent automaton is prefix-closed [3, Cor. 4.11]. It follows that $\mathcal{L}(\mathcal{A})$ is a consistent set of pomsets.

The mapping from stably concurrent automata to prefix-closed and consistent sets of pomsets is actually onto: *Any prefix-closed and consistent set of pomsets is the trace language of a stably concurrent automaton.* This connection specializes into a correspondance between *finite* stably concurrent automata and *regular*, prefix-closed, and consistent sets of pomsets.

3.3 From Consistent Sets of Pomsets to Mazurkiewicz Traces

Basically Arnold's result relates regular consistent sets of pomsets to regular Mazurkiewicz trace languages [4]. The latter can be viewed as the trace language of a particular asynchronous system. Consider some independence relation $(\Sigma, \|)$ and some asynchronous system $\mathcal{A}$ that has a single state q such that $q \xrightarrow{a} q$ for all $a \in \Sigma$. Then the set of Mazurkiewicz traces $\mathbb{M}(\Sigma, \|)$ may be defined as the trace language $\mathcal{L}(\mathcal{A})$. Since $\mathcal{A}$ is a stably concurrent automaton, for each $u \in \Sigma^\star$ there exists a (unique) pomset t over Σ such that $[u] = \mathrm{LE}(t)$ (Th. 3.3). That is why subsets of Mazurkiewicz traces are particular cases of consistent sets of pomsets. Noteworthy a subset of Mazurkiewicz traces $\mathcal{L} \subseteq \mathbb{M}(\Sigma, \|)$ is regular (Def. 3.2) if and only if $\mathrm{LE}(\mathcal{L})$ is a regular set of words.

Let Σ and Γ be two alphabets and $\pi : \Gamma \to \Sigma$ a mapping from Γ to Σ. This mapping extends in a natural way into a function that maps each pomset $t = (E, \preccurlyeq, \xi)$ over Γ to the structure $\pi(t) = (E, \preccurlyeq, \pi \circ \xi)$. The latter might not be a pomset over Σ in case some autoconcurrency appears in it. This situation can occur if $\pi(a) = \pi(b)$ for two distinct actions $a, b \in \Sigma$ while there are two events e and f that are labelled by a and b and that are not causally related. The next notion of refinement allows to relate two sets of pomsets that are identical up to some relabeling.

Definition 3.4. *Let $\mathcal{L}$ and $\mathcal{L}'$ be two consistent sets of pomsets over Σ and Γ respectively. A mapping $\pi : \Gamma \to \Sigma$ from Γ to Σ is a* refinement *from $\mathcal{L}$ to $\mathcal{L}'$ if π is a bijection from $\mathcal{L}'$ onto $\mathcal{L}$ and from $\mathrm{Pref}(\mathcal{L}')$ onto $\mathrm{Pref}(\mathcal{L})$.*

Now one main contribution of [1] can be slightly extended as follows.

THEOREM 3.5. *[1, Th. 6.16] For any regular consistent set of pomsets $\mathcal{L}$ over Σ, there is a refinement from $\mathcal{L}$ to a regular set of Mazurkiewicz traces $\mathcal{L}' \subseteq \mathbb{M}(\Gamma, \|)$.*

4 From Coherence to Forward-Stability

In this section we want to apply Theorem 3.5 in order to build a finite *forward-stable* asynchronous system from a finite coherent stably concurrent automaton (Cor. 4.6). The requirement that the asynchronous system should be forward-stable is the main difficulty tackled in this section: Without this requirement, the result would follow directly from Th. 3.5 because any regular prefix-closed set of Mazurkiewicz traces is the trace language of a finite asynchronous system.

4.1 Coherent and Forward-Stable Mazurkiewicz Trace Languages

It is easy to characterize the trace languages associated with the stably concurrent automata we are intererested in.

DEFINITION 4.1. *A prefix-closed and consistent set of pomsets $\mathcal{L}$ over Σ is* coherent *if for all words $u \in \Sigma^\star$, all distinct actions $a, b, c \in \Sigma$:*
$u.ab \sim_{\mathcal{L}} u.ba \wedge u.bc \sim_{\mathcal{L}} u.cb \wedge u.ca \sim_{\mathcal{L}} u.ac$ *implies* $u.abc \sim_{\mathcal{L}} u.acb \sim_{\mathcal{L}} u.cab$.

Clearly, the trace language of a coherent stably concurrent automaton is coherent. Conversely, we can show that any coherent prefix-closed consistent set of pomsets is the trace language of some coherent stably concurrent automaton.

Since we deal also with forward-stable asynchronous systems (Def. 2.5 and Cor. 4.6), we focus also on forward-stable Mazurkiewicz trace languages.

DEFINITION 4.2. *A prefix-closed set of Mazurkiewicz traces $\mathcal{L} \subseteq \mathbb{M}(\Sigma, \|)$ is* forward-stable *w.r.t. $(\Sigma, \|)$ if for all words $u, v \in \Sigma^\star$ and all actions $a, b \in \Sigma$:*
$$[u.a] \in \mathcal{L} \wedge [u.b] \in \mathcal{L} \wedge a \| b \text{ implies } [u.ab] \in \mathcal{L}.$$

This condition is well-known. A forward-stable Mazurkiewicz trace language is called *safe-branching* in [16], *forward independence closed* in [10], *ideal* in [2], and *proper* in [9]. Clearly the trace language of a forward-stable asynchronous system is forward-stable. Actually, the converse property holds. As expressed by the next basic lemma, this connection specializes into a correspondance between finite asynchronous systems and regular sets of Mazurkiewicz traces.

LEMMA 4.3. *Any regular, forward-stable, and prefix-closed set of Mazurkiewicz traces is the trace language of some finite forward-stable asynchronous system.*

Observe now that any forward-stable prefix-closed set of Mazurkiewicz traces is coherent. It is easy to see that the converse property does not hold. However we can represent coherent set of Mazurkiewicz traces by forward-stable sets of Mazurkiewicz traces by means of a refinement (Def. 3.4). This is expressed for regular languages in the next useful result whose proof will be sketched in Subsection 4.3 and relies on Zielonka's theorem.

THEOREM 4.4. *Let $\mathcal{L}$ be a regular,* coherent*, and prefix-closed set of Mazurkiewicz traces. There exists a refinement from $\mathcal{L}$ to a regular,* forward-stable*, and prefix-closed set of Mazurkiewicz traces.*

In order to apply Theorem 4.4 together with Theorem 3.5, we observe that coherence of consistent sets of pomsets is preserved by refinements.

LEMMA 4.5. *Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be two consistent sets of pomsets over Σ_1 and Σ_2 respectively. Let $\pi : \Sigma_1 \to \Sigma_2$ be a refinement from $\mathcal{L}_2$ to $\mathcal{L}_1$. If $\mathcal{L}_2$ is prefix-closed and coherent then $\mathcal{L}_1$ is prefix-closed and coherent, too.*

We come now to the statement of our main result.

COROLLARY 4.6. *For any finite coherent stably concurrent automaton $\mathcal{A}$, there exists some finite forward-stable asynchronous system $\mathcal{A}'$ such that the trace domains $(\mathcal{L}(\mathcal{A}), \sqsubseteq)$ and $(\mathcal{L}(\mathcal{A}'), \sqsubseteq)$ are isomorphic.*

Proof. The trace language $\mathcal{L}(\mathcal{A})$ is a regular, coherent, prefix-closed, and consistent set of pomsets. By Th. 3.5 there exists a refinement from $\mathcal{L}(\mathcal{A})$ to a regular and prefix-closed set of Mazurkiewicz traces $\mathcal{L}'$. By Lemma 4.5, $\mathcal{L}'$ is coherent, too. By Theorem 4.4, we get a refinement from $\mathcal{L}'$ to a regular, forward-stable, and prefix-closed set of Mazurkiewicz traces $\mathcal{L}''$. By Lemma 4.3, $\mathcal{L}''$ is the trace language of a finite forward-stable asynchronous system $\mathcal{A}''$. Since we can compose refinements, we get a refinement from $\mathcal{L}(\mathcal{A})$ to $\mathcal{L}(\mathcal{A}'')$. It follows that the trace domains $(\mathcal{L}(\mathcal{A}), \sqsubseteq)$ and $(\mathcal{L}(\mathcal{A}''), \sqsubseteq)$ are isomorphic. ■

4.2 Zielonka's Theorem

Let $\mathbb{S} = (\mathcal{P}_i)_{i\in I}$ be a family of finite automata $\mathcal{P}_i = (Q_i, \imath_i, \Sigma_i, \longrightarrow_i)$ where Q_i is a non-empty finite set of states, $\imath_i \in Q_i$ is the initial state, Σ_i is an alphabet of actions and $\longrightarrow_i \subseteq Q_i \times \Sigma_i \times Q_i$ is a set of *deterministic* transitions: If $q \xrightarrow{a} q'$ and $q \xrightarrow{a} q''$ then $q' = q''$. The global behaviour of such a system can be modelled by a single automaton which is the *mixed product* of its components [6]: $\prod \mathbb{S} = (\prod_{i\in I} Q_i, (\imath_i)_{i\in I}, \bigcup_{i\in I} \Sigma_i, \longrightarrow)$ where $(q_i)_{i\in I} \xrightarrow{a} (q'_i)_{i\in I}$ if and only if for all $i \in I$ it holds $a \in \Sigma_i \Rightarrow q_i \xrightarrow{a}_i q'_i$ and $a \notin \Sigma_i \Rightarrow q_i = q'_i$. We can enrich the mixed product of $\mathbb{S}$ by explicitly modelling concurrency: We put $a \| b$ if $\{a, b\} \not\subseteq \Sigma_i$ for all $i \in I$. In that way we provide the mixed product $\prod \mathbb{S}$ with an independence relation $\|$ and turn it into a *forward-stable* asynchronous system. The latter is associated with a trace language $\mathcal{L}(\prod \mathbb{S})$.

Let us now formulate a particular version of Zielonka's theorem [20,10,16] in terms of mixed products and refinements. Let $\mathcal{A}$ be an asynchronous system over the independence alphabet $(\Sigma, \|)$ with set of states Q and initial state $\imath \in Q$. A finite family $\delta = (\Sigma_i)_{i\in I}$ of subsets of Σ is called a *distribution of* $(\Sigma, \|)$ if for all actions $a, b \in \Sigma$ we have $a \not\| b \Leftrightarrow \exists i \in I, \{a, b\} \subseteq \Sigma_i$. Given a subset of states $F \subseteq Q$, we let $\mathcal{L}_F(\mathcal{A})$ denote the subset of traces $[u]$ such that $\imath \xrightarrow{u} q \in F$.

THEOREM 4.7. *Let $\delta = (\Delta_i)_{i\in I}$ be a distribution of some independence alphabet $(\Sigma, \|)$. Let $\mathcal{L} \subseteq \mathbb{M}(\Sigma, \|)$ be a regular, forward-stable, and prefix-closed set of*

Mazurkiewicz traces. There exists a family of finite automata $\mathbb{S} = (\mathcal{P}_i)_{i\in I}$ *with local alphabets* $(\Sigma_i)_{i\in I}$ *and a refinement* $\pi : \bigcup_{i\in I} \Sigma_i \to \Sigma$ *from* $\mathcal{L}$ *to* $\mathcal{L}(\prod \mathbb{S})$ *such that for all* $i \in I$ *and all* $a \in \bigcup_{i\in I} \Sigma_i$ *it holds* $a \in \Sigma_i \Leftrightarrow \pi(a) \in \Delta_i$.

4.3 Proof of Theorem 4.4

In this section we fix a regular, coherent, and prefix-closed set of Mazurkiewicz traces $\mathcal{L} \subseteq \mathbb{M}(\Gamma_1, \|_1)$. There exists a finite forward-stable asynchronous system $\mathcal{A}_1 = (Q_1, \imath_1, \Gamma_1, \longrightarrow_1, \|_1)$ together with a subset of states $F \subseteq Q_1$ such that $\mathcal{L} = \mathcal{L}_F(\mathcal{A}_1)$. Clearly we can assume that all states of $q \in Q_1$ are reachable from the initial state $\imath_1$. Consequently if $q \xrightarrow{a}_1 q' \in F$ then $q \in F$ because $\mathcal{L}$ is prefix-closed.

We build from $(\Gamma_1, \|_1)$ an extended independence alphabet $(\Gamma_2, \|_2)$ such that $\Gamma_2 = \Gamma_1 \uplus \{\{a, b\} \subseteq \Gamma_1 \mid a\|_1 b\}$ and the independence relation $\|_2$ is defined as follows:

- for all $a, b \in \Gamma_1$, $a\|_2 b$ if $a\|_1 b$;
- for all $\{a, b\} \in \Gamma_2 \setminus \Gamma_1$, for all $c \in \Gamma_1$, $c\|_2\{a, b\}$ if $c\|_1 a$ and $c\|_1 b$;
- for all $\{a, b\}, \{c, d\} \in \Gamma_2 \setminus \Gamma_1$, $\{a, b\}\|_2\{c, d\}$ if $a\|_1 c$, $a\|_1 d$, $b\|_1 c$, and $b\|_1 d$.

We fix some arbitrary distribution $\delta = (\Delta_i)_{i\in I}$ of $(\Gamma_1, \|_1)$. For each $i \in I$, we define an extended subset of actions $\Delta'_i = \Delta_i \uplus \{x \in \Gamma_2 \setminus \Gamma_1 \mid x \cap \Delta_i \neq \emptyset\}$. We can check easily that $\delta' = (\Delta'_i)_{i\in I}$ is a distribution of $(\Gamma_2, \|_2)$.

We build also a new structure $\mathcal{A}_2 = (Q_2, \imath_2, \Gamma_2, \longrightarrow_2, \|_2)$ where $Q_2 \subseteq (Q_1)^{2^{\Gamma_1}}$, that is, a state $\sigma \in Q_2$ is a map that associates each subset of actions $A \subseteq \Gamma_1$ with some state $\sigma(A) \in Q_1$. The initial state $\imath_2 \in Q_2$ maps each subset $A \subseteq \Gamma_1$ to the initial state $\imath_1$. The transition relation $\longrightarrow_2$ is defined as follows: Consider two states $\sigma : 2^{\Gamma_1} \to Q_1$ and $\sigma' : 2^{\Gamma_1} \to Q_1$

- we put $\sigma \xrightarrow{a}_2 \sigma'$ for some action $a \in \Gamma_1$ if for all $A \subseteq \Gamma_1$,
  - if $a \in A$ then $\sigma'(A) = \sigma(A)$;
  - if $a \notin A$ then $\sigma(B) \xrightarrow{a}_1 \sigma'(A)$ where $B = \{c \in A \mid c\|_1 a\}$.
- we put $\sigma \xrightarrow{x}_2 \sigma'$ with $x = \{a, b\} \in \Gamma_2 \setminus \Gamma_1$ if $\sigma = \sigma'$, $\sigma(A) \xrightarrow{a}_1 q_a \in F$, $\sigma(A) \xrightarrow{b}_1 q_b \in F$, and $\sigma(A) \xrightarrow{ab}_1 q \notin F$ where $A = \{c \in \Gamma_1 \mid c\|_1 a \wedge c\|_1 b\}$.

Finally let Q_2 be the subset of states that are reachable from $\imath_2$. By an immediate induction, it is clear that for all words $u \in \Gamma_1^\star$ and all states $\sigma \in Q_2$, if $\imath_2 \xrightarrow{u}_2 \sigma$ then $\imath_1 \xrightarrow{u}_1 \sigma(\emptyset)$. We shall prove a useful converse property in Lemma 4.8.

The product of two Mazurkiewicz traces $[w], [w'] \in \mathbb{M}(\Gamma_1, \|_1)$ is defined as usual by $[w] \cdot [w'] = [w.w']$. For all $A \subseteq \Gamma_1$ and all traces $[u] \in \mathbb{M}(\Gamma_1, \|_1)$ we denote by $[u]/A$ the least trace $[v]$ such that $[u] = [v] \cdot [z]$ for some $z \in A^\star$. If $u \in L(\mathcal{A}_1)$ then we define the map $\sigma_u : 2^{\Gamma_1} \to Q_1$ as follows: For all $A \subseteq \Gamma_1$, we let $\sigma_u(A)$ be the state from Q_1 such that $\imath_1 \xrightarrow{v}_1 \sigma_u(A)$ for all $v \in [u]/A$. In particular $\imath_1 \xrightarrow{u}_1 \sigma_u(\emptyset)$. Note also that $\imath_2 = \sigma_\varepsilon$ and $u \sim v$ implies $\sigma_u = \sigma_v$.

LEMMA 4.8. *For all* $u \in \Gamma_1^\star$, $\imath_1 \xrightarrow{u}_1 q_1$ *in* $\mathcal{A}_1$ *if and only if* $\imath_2 \xrightarrow{u}_2 \sigma_u$ *in* $\mathcal{A}_2$.

Proof. The proof follows by induction by means of the next two key properties. For all words $u \in \Gamma_1^\star$, for all actions $a \in \Gamma_1$, and for all subsets $A \subseteq \Gamma_1$

1. if $u.a \in L(\mathcal{A}_1)$ and $a \notin A$ then $[u.a]/A = [v.a]$ where $[v] = [u]/\{c \in A \mid c \|_1 a\}$;
2. if $u.a \in L(\mathcal{A}_1)$ and $a \in A$ then $\sigma_{u.a}(A) = \sigma(A)$. ∎

This lemma enables us to check easily that the structure $\mathcal{A}_2$ is a forward-stable asynchronous system. Now $\mathcal{L}(\mathcal{A}_2)$ is a regular, prefix-closed, and forward-stable set of Mazurkiewicz traces $\mathcal{L}(\mathcal{A}_2) \subseteq \mathbb{M}(\Gamma_2, \|_2)$. We can apply Zielonka's theorem (Th. 4.7) to $\mathcal{L}(\mathcal{A}_2)$ with the distribution $\delta' = (\Delta'_i)_{i \in I}$ from above. We get a new independence alphabet $(\Gamma_3, \|_3)$, a mapping $\pi : \Gamma_3 \to \Gamma_2$, and a finite system of finite automata $\mathcal{S}_3 = (\mathcal{P}_i)_{i \in I}$ with alphabets Σ_i such that $(\Sigma_i)_{i \in I}$ is a distribution of $(\Gamma_3, \|_3)$, π is a refinement from $\mathcal{L}(\mathcal{A}_2)$ to $\mathcal{L}(\prod \mathcal{S}_3)$, and for all $i \in I$ and all $a \in \bigcup_{i \in I} \Sigma_i$ we have $a \in \Sigma_i \Leftrightarrow \pi(a) \in \Delta_i$. Then for all $a, b \in \Gamma_3$, we have $a \not\|_3 b \Leftrightarrow \pi(a) \not\|_1 \pi(b)$. We put $\mathcal{A}_3 = (Q_3, \imath_3, \Gamma_3, \longrightarrow_3, \|_3) = \prod \mathcal{S}_3$. We can assume that in each component automaton $\mathcal{P}_i$, each action occurs in at most one transition.

We consider now a new forward-stable asynchronous system $\mathcal{A}_4$ by restricting the alphabet and the independence relation over $\mathcal{A}_3$. We let $(\Gamma_4, \|_4)$ be the independence alphabet such that $\Gamma_4 = \Gamma_3 \cap \pi^{-1}(\Gamma_1)$ and for all $a, b \in \Gamma_4$, $a \not\|_4 b$ if $a \not\|_3 b$ or there exists some action $x \in \Gamma_3 \setminus \Gamma_4$ such that $\pi(x) = \{\pi(a), \pi(b)\}$ and the next condition is satisfied for all $i \in I$ and all $q_i \in Q_i$:

$$\forall c \in \{a, b\} : \Big(c \in \Sigma_i \wedge \exists q'_i \in Q_i, q_i \xrightarrow{c}_i q'_i\Big) \Rightarrow \Big(x \in \Sigma_i \wedge \exists \widetilde{q_i} \in Q_i, q_i \xrightarrow{x}_i \widetilde{q_i}\Big)$$

As transitions, we simply restrict to transitions carrying actions from Γ_4: We put $q \xrightarrow{a}_4 q'$ if $q \xrightarrow{a}_3 q'$ and $\pi(a) \in \Gamma_1$. Obviously $\mathcal{A}_4 = (Q_3, \imath_3, \Gamma_4, \longrightarrow_4, \|_4)$ is also a forward-stable asynchronous system.

We build a last structure $\mathcal{A}_5$ by synchronizing $\mathcal{A}_1$ and $\mathcal{A}_4$ with a restriction to the global states $F \subseteq Q_1$. We put $\mathcal{A}_5 = (Q_5, (\imath_1, \imath_3), \Gamma_4, \longrightarrow_5, \|_4)$ where $Q_5 \subseteq F \times Q_3$ and the transition relation is defined as follows: We put $(q_1, q_3) \xrightarrow{a}_5 (q'_1, q'_3)$ if $q_1 \xrightarrow{\pi(a)}_1 q'_1$ and $q_3 \xrightarrow{a}_3 q'_3$. Now a pair $(q_1, q_3) \in F \times Q_3$ belongs to Q_5 if it is reachable, that is: There exists some $u \in \Gamma_4^\star$ such that $(\imath_1, \imath_3) \xrightarrow{u}_5 (q_1, q_3)$. It is easy to check that $\mathcal{A}_5$ is a finite asynchronous system.

We can use now the hypotheses that $\mathcal{L}$ is coherent and each action appears locally in at most one transition to show the crucial following fact.

LEMMA 4.9. *The finite asynchronous system $\mathcal{A}_5$ is forward-stable.*

Proof. Assume $(\imath_1, \imath_3) \xrightarrow{u}_5 (q_1, q_3) \xrightarrow{a}_5 (q'_1, q'_3)$ and $(q_1, q_3) \xrightarrow{b}_5 (q''_1, q''_3)$ with $a \| b$. Since $a \|_3 b$, $q'_3 \xrightarrow{b}_3 q'''_3$ and $q''_3 \xrightarrow{a}_3 q'''_3$ for some state $q'''_3 \in Q_3$ because $\mathcal{A}_3$ is forward-stable. Similarly there exists some state $q'''_1 \in Q_1$ such that $q'_1 \xrightarrow{\pi(b)}_1 q'''_1$ and $q''_1 \xrightarrow{\pi(a)}_1 q'''_1$. It is sufficient to prove that $q'''_1 \in F$. We proceed by contradiction and assume $q'''_1 \notin F$. Let $A = \{c \in \Gamma_1 \mid c \| _1 \pi(a) \wedge c \|_1 \pi(b)\}$. We consider $[v] = [\pi(u)]/A$. We have $\imath_1 \xrightarrow{v}_1 \sigma_{\pi(u)}(A) \xrightarrow{z}_1 q_1 = \sigma_{\pi(u)}(\emptyset)$ with $z \in A^\star$. Since $q'_1 \in F$, $q''_1 \in F$, $q'''_1 \notin F$, and $\mathcal{L}$ is coherent we have $\sigma_{\pi(u)} \xrightarrow{x}_2 \sigma_{\pi(u)}$ with $x = \{\pi(a), \pi(b)\}$. There exists $x' \in \Gamma_3 \setminus \Gamma_4$ such that $u.x' \in L(\mathcal{A}_3)$ and $\pi(x') = x$. We show now that $a \not\|_4 b$ (which is the expected

contradiction). Let $i \in I$ and $q_i \in Q_i$ be such that $a \in \Sigma_i$ and $q_i \xrightarrow{a} q_i'$. Then $\pi(a) \in \Delta_i'$ and $x \in \Delta_i'$. It follows that $x' \in \Sigma_i$ hence $q_i \xrightarrow{x'}_i \widetilde{q}_i$ for some local state $\widetilde{q}_i \in Q_i$ because $u.a \in L(\mathcal{A}_3)$, $u.x' \in L(\mathcal{A}_3)$, and each action appears locally in at most one transition. Similarly, for all $j \in I$ and all $q_j \in Q_j$ if $b \in \Sigma_j$ and $q_j \xrightarrow{b} q_j''$ then $x' \in \Sigma_j$ and $q_j \xrightarrow{x'}_j \widetilde{q}_j$ for some state $\widetilde{q}_j \in Q_j$. Thus $a \not\|_4 b$. ■

We can easily check that the mapping $\pi : \Gamma_4 \to \Gamma_1$ induces a bijection from $L(\mathcal{A}_5)$ to $\mathrm{LE}(\mathcal{L})$. Moreover $u \sim_{\mathcal{A}_5} v$ implies $\pi(u) \sim_{\mathcal{L}} \pi(v)$. To complete the proof and show that $\pi : \Gamma_4 \to \Gamma_1$ is a refinement from $\mathcal{L}$ to $\mathcal{L}(\mathcal{A}_5)$ it is sufficient to establish the converse property. Assume $u.ab.v \sim_{\mathcal{L}} u.ba.v$ where $u, v \in \Gamma_1^\star$ and $a \|_1 b$. There are $u', v' \in \Gamma_3^\star$ and $a', b' \in \Gamma_3$ such that $\pi(u') = u$, $\pi(v') = v$, $\pi(a') = a$, $\pi(b') = b$, and $u'.a'b'.v' \in L(\mathcal{A}_3)$. Moreover $u'.a'b'.v' \sim_{\mathcal{A}_3} u'.b'a'.v'$ because $a' \|_3 b'$. We need just to show that $a' \|_4 b'$. We proceed by contradiction and assume $a' \not\|_4 b'$. There exists some $x \in \Gamma_3 \setminus \Gamma_4$ such that $\pi(x) = \{a, b\}$ and the next condition is satisfied for all $i \in I$ and all $q_i \in Q_i$:

$$\forall c \in \{a', b'\} : \Big(c \in \Sigma_i \wedge \exists q_i' \in Q_i, q_i \xrightarrow{c}_i q_i'\Big) \Rightarrow \Big(x \in \Sigma_i \wedge \exists \widetilde{q}_i \in Q_i, q_i \xrightarrow{x}_i \widetilde{q}_i\Big)$$

We have $\imath_3 \xrightarrow{u'}_3 q_3 \xrightarrow{a'}_3 q_3' \xrightarrow{b'}_3 q_3''$ and $q_3 \xrightarrow{b'}_3 q_3''' \xrightarrow{a'}_3 q_3''$. If $x \in \Sigma_i$ then $\pi(x) \in \Delta_i'$ hence it holds $a \in \Delta_i$ or $b \in \Delta_i$, which implies that $a' \in \Sigma_i$ or $b' \in \Sigma_i$. Therefore $q_3 \xrightarrow{x}_3 \widetilde{q}_3$ in $\mathcal{A}_3$. It follows that $u'.x \in L(\mathcal{A}_3)$ and $u.\pi(x) \in L(\mathcal{A}_2)$. Hence $u.ab \notin \mathrm{LE}(\mathcal{L}_F(\mathcal{A}_1)) = \mathrm{LE}(\mathcal{L})$, a contradiction.

5 Related Works

To conclude we wish to sketch some connections between this paper and the theory of regular event structures [18]. Due to the page limit, most definitions are omitted here. However we believe that the following arguments can clarify how our results apply to that setting.

Prefix-closed, coherent, and consistent sets of pomsets are the trace languages of coherent stably concurrent automata. They can be regarded as generalized trace languages [13]. The latter are closely related to event structures by means of a coreflection whose units are labelings [13]. Event structures are a classical semantical model in concurrency theory since they appeared as the unfoldings of (possibly infinite) 1-safe Petri nets [12]. This strong relationship can be extended to forward-stable asynchronous systems [2] and coherent stably concurrent automata [8].

The results presented in this paper allow us to claim that for an event structure $\mathcal{E}$ the following conditions are equivalent:

- (i) $\mathcal{E}$ is the unfolding of a finite 1-safe Petri net;
- (ii) $\mathcal{E}$ admits a regular forward-stable Mazurkiewicz labeling;
- (iii) $\mathcal{E}$ is the unfolding of a finite forward-stable asynchronous system;
- (iv) $\mathcal{E}$ is the unfolding of a finite coherent asynchronous system;
- (v) $\mathcal{E}$ is the unfolding of a finite coherent stably concurrent automaton;
- (vi) $\mathcal{E}$ admits a regular labeling.

In [18], the equivalence between (i) and (ii) is established by means of Zielonka's theorem. The equivalences (ii) $\Leftrightarrow$ (iii) and (v) $\Leftrightarrow$ (vi) are easy consequences of the corresponding definitions. The implications (iii) $\Rightarrow$ (iv) and (iv) $\Rightarrow$ (v) are trivial. As explained in the proof of Corollary 4.6, Arnold's result (Th. 3.5) was used here to get (v) $\Rightarrow$ (iv). Then we used Zielonka's theorem to prove (iv) $\Rightarrow$ (iii) by means of Theorem 4.4. This work subsumes a difficult work by Schmitt who established that (iii) holds if and only if $\mathcal{E}$ is the unfolding of a finite stable trace automaton [14]. Note here that a direct proof of (vi) $\Rightarrow$ (ii) would provide us easily with an alternative proof of Theorem 3.5 which is a difficult result.

A very interesting conjecture by Thiagarajan [18] characterizes the unfoldings of finite 1-safe Petri nets. It asserts that *any regular event structure is the unfolding of a finite 1-safe Petri net.* Since (vi) $\Rightarrow$ (i), our contribution reduces this conjecture to proving that *any regular event structure admits a regular labelling.* We are investigating at present this issue by adapting to the general setting the techniques developped in [11]. We stress finally that any direct proof of Thiagarajan's conjecture would show that (vi) $\Rightarrow$ (ii) because (vi) implies that $\mathcal{E}$ is a regular event structure. It would lead to a new proof of Theorem 3.5. The latter is thus a natural ingredient to answer this question.

References

1. Arnold A.: *An extension of the notion of traces and asynchronous automata.* RAIRO, Theoretical Informatics and Applications **25** (Gauthiers-Villars, 1991) 355–393
2. Bednarczyk M.A.: *Categories of Asynchronous Systems.* PhD thesis in Computer Science (University of Sussex, 1988)
3. Bracho F., Droste M., Kuske D.: *Representations of computations in concurrent automata by dependence orders.* Theoretical Computer Science **174** (1997) 67–96
4. Diekert V. and Rozenberg G.: *The Book of Traces.* (World Scientific, 1995)
5. Droste M.: *Concurrency, automata and domains.* LNCS **443** (1990) 195–208
6. Duboc C.: *Mixed product and asynchronous automata.* Theoretical Computer Science **48** (1986) 183–199
7. Husson J.-Fr.: *Modélisation de la causalité par des relations d'indépendances.* PhD thesis (Université Paul Sabatier de Toulouse, 1996)
8. Kuske D.: *Nondeterministic automata with concurrency relations and domains.* CAAP, LNCS **787** (1994) 202–217
9. Mazurkiewicz A.: *Trace theory.* LNCS **255** (1987) 279–324
10. Mukund M.: *From global specifications to distributed implementations.* In *Synthesis and Control of Discrete Event Systems*, Kluwer (2002) 19–34
11. Nielsen M., Thiagarajan P.S.: *Regular Event Structures and Finite Petri Nets: The Conflict-Free Case.* ICATPN, LNCS **2260** (2002) 335–351
12. Nielsen M., Plotkin G and Winskel G.: *Petri nets, event structures and domains I.* Theoretical Computer Science **13** (1980) 86–108
13. Sassone V., Nielsen M., Winskel G.: *Deterministic Behavioural Models for Concurrency (Extended Abstract).* MFCS, LNCS **711** (1993) 682–692
14. Schmitt V.: *Stable trace automata vs. full trace automata.* Theoretical Computer Science **200** (1998) 45–100

15. Stark E.W.: *Connections between a Concrete and an Abstract Model of Concurrent Systems*. LNCS **442** (1990) 53–79
16. Ştefănescu A., Esparza J., and Muscholl A.: *Synthesis of distributed algorithms using asynchronous automata*. CONCUR, LNCS **2761** (2003) 20–34
17. Szpilrajn E.: *Sur l'extension de l'ordre partiel*. Fund. Math. **16** (1930) 386–389
18. Thiagarajan P.S.: *Regular Event Structures and Finite Petri Nets: A Conjecture*. Formal and Natural Computing, LNCS **2300** (2002) 244–256
19. Winskel G.: *Event structures*. Advances in Petri Nets, LNCS **255** (1987) 325–392
20. Zielonka W.: *Notes on finite asynchronous automata*. RAIRO, Theoretical Informatics and Applications **21** (Gauthiers-Villars, 1987) 99–135