On Synthesising Step Alphabets for Acyclic Invariant Structures

Ryszard Janicki[1], Jetty Kleijn[2], Maciej Koutny[3], and Łukasz Mikulski[4]

[1] Department of Computing and Software, McMaster University
Hamilton, ON, L8S 4K1, Canada
janicki@mcmaster.ca
[2] LIACS, Leiden University, P.O.Box 9512
NL-2300 RA Leiden, The Netherlands
h.c.m.kleijn@liacs.leidenuniv.nll
[3] School of Computing Science, Newcastle University
Newcastle upon Tyne NE1 7RU, United Kingdom
maciej.koutny@ncl.ac.uk
[4] Faculty of Mathematics and Computer Science, Nicolaus Copernicus University
Toruń, Chopina 12/18, Poland
lukasz.mikulski@mat.umk.pl

Abstract. A step alphabet describes dependencies between the actions of a concurrent system in the form of two relations expressing potential simultaneity and sequentialisability. These form the basis for the identification of step sequences as observations of the same run of the system. The resulting equivalence classes – step traces – can be represented by (labelled) invariant structures with two relations, viz. mutex and weak causality.
In this paper, we address the following synthesis problem for acyclic invariant structures which are structures corresponding to step traces with sequential realisations: given an acyclic invariant structure that can represent a step trace, construct a suitable step alphabet, i.e., provide suitable simultaneity and sequentialisability relations for its actions. The main result is that the set of all suitable alphabets forms a complete lattice with the ordering derived from the relative 'strength' of the dependencies between individual actions.

Keywords: step alphabet, step trace, invariant structure, simultaneity, sequentialisability, mutual exclusion, weak causality, acyclicity, synthesis

1 Introduction

An observational semantics of concurrent systems is usually defined either in terms of sequences (total orders), or step sequences, (stratified orders). When concurrent histories are fully described by causality relations, i.e., by a *partial order*, Mazurkiewicz traces [14, 15] allow a representation of the entire partial order by a single sequence (plus *independency* relation), which provides a simple

and elegant connection between observational and process semantics (i.e., the semantics in terms of concurrent histories) of concurrent systems.

A trace is an equivalence class of sequences comprising all (sequential) observations of a single concurrent run. The dependencies between the events of a trace are invariant among (common to) all elements of the trace. They define an acyclic dependence graph which — through its transitive closure — determines the underlying causality structure of the trace as a (labelled) partial order [18]. Then each trace is represented by a labelled partial order; see, e.g., [3, 6].

However, the concurrency paradigm of Mazurkiewicz traces with the corresponding partial order interpretation of concurrency is rather restricted, as it cannot, for example, handle the 'no later than relationship' [12].

In [9], a generalisation of the theory of Mazurkiewicz traces is presented for the case that actions could occur and could be observed as occurring simultaneously (a common assumption made, e.g., by concurrency models inspired by bio-chemical reactions as in [4]; see also [12] for other examples). This leads to step sequences, i.e., sequences of steps rather than single actions where a step is a set of one or more actions occurring simultaneously. To retain the philosophy underlying Mazurkiewicz traces, the extended set-up is based on a few explicit and simple design choices. Instead of the independence relation, *step alphabets* use two basic relations between pairs of actions: *simultaneity* indicating actions that may occur together in a step, and *sequentialisation* indicating equivalent orders of executing two different actions. The two relations are applied to identify step sequences as observations of the same concurrent run, and the resulting equivalence classes of step sequences are called *step traces*. Step traces can be represented by invariant structures with two relations: the mutual exclusion and (possibly cyclic) weak causality. Step sequences have been used to represent operational semantics of concurrent systems since a long time [5, 19, 1]. The fundamental difference between models like those in [5, 19, 1] and the approach of this paper is that step sequences that are considered equivalent are grouped into step traces. Moreover, each step trace uniquely defines a relational structure, similar to how each trace defines a unique causal partial order.

A key algorithmic issue here is to decide whether an invariant structure represents a step trace over a given step alphabet. This problem was considered and solved for the general case in [7]. In [10], a companion paper to the present paper, we seek further efficiency improvements by restricting the class of order structures under consideration to those with an *acyclic* weak causality relation. There it is shown that these invariant structures represent *linearisable step traces*, i.e., step traces where each step $A = \{a_1, \ldots, a_n\}$ can be represented by some equivalent sequence of singletons $\{a_{i_1}\}\{a_{i_2}\}\ldots\{a_{i_n}\}$. In other words, each step has a linear representation. (Here it is worthwhile to remark that the requirement that concurrent runs can be sequentialised, is not an uncommon assumption, see e.g., [1, 2, 16, 20].) The acyclity property allows one to structurally reduce an invariant structure to a structure which can then be used to solve the problem in a purely local way, by considering pairs of events, rather than the whole

structures. Furthermore, a characterisation is given of those step alphabets that can give rise to a given acyclic invariant structure.

In this paper, we show how one can check whether an acyclic invariant structure represents a step trace over any step alphabet. The main result here is that the set of all step alphabets that would qualify, forms a complete lattice with the ordering derived from relative 'strength' of the dependencies between pairs of individual actions.

2 Background

In this section, we outline the main concepts of the extension of Mazurkiewicz trace theory to step traces. Complete definitions and results, as well as illustrative examples, can be found in, e.g., [7, 9, 10].

Step alphabets. A *step alphabet* is a triple $\theta = \langle \Sigma, \mathsf{sim}, \mathsf{seq} \rangle$, where Σ is a finite nonempty *alphabet of actions*, and sim (*simultaneity*) and seq (*sequentialisability*) are irreflexive binary relations over Σ such that sim and $\mathsf{seq} \setminus \mathsf{sim}$ are symmetric relations.

A *step* over θ is a clique of the simultaneity relation sim, and *step sequences* over θ are all finite sequences SSEQ_θ of steps. A step sequence is *linear* if it consists of singleton sets.

Step traces. Two step sequences over θ, u and v, are *equivalent* if one can be obtained from the other by (repeatedly): swapping two consecutive steps AB consisting of actions which can be sequentialised in any order ($A \times B \subseteq \mathsf{seq} \cap \mathsf{seq}^{-1}$), splitting a step A into two consecutive steps BC provided that the resulting ordering of actions respects the sequentialisability given by seq ($B \times C \subseteq \mathsf{seq}$), and joining adjacent steps BC into a single step $B \cup C$ if their order respects seq and all actions can be simultaneous ($B \times C \subseteq \mathsf{sim} \cap \mathsf{seq}$). The equivalence classes STR_θ of the resulting equivalence relation on step sequences are *step traces* over θ.

Derived dependencies. Next to simultaneity and serialisability, it helps to use six further *(static) dependencies* between actions which together partition $\Sigma \times \Sigma$:

- $\mathsf{rig} = (\Sigma \times \Sigma) \setminus (\mathsf{sim} \cup (\mathsf{seq} \cap \mathsf{seq}^{-1}))$ is *rigid order* allowing neither simultaneity nor changing of the order of actions.
- $\mathsf{inl} = (\mathsf{seq} \cap \mathsf{seq}^{-1}) \setminus \mathsf{sim}$ is *interleaving* allowing to change the order of actions that cannot occur (simultaneously) together in a step.
- $\mathsf{ssi} = \mathsf{sim} \setminus (\mathsf{seq} \cup \mathsf{seq}^{-1})$ is *strong simultaneity* allowing a pair of actions to occur together in a step but disallowing serialisation and interleaving.
- $\mathsf{sse} = (\mathsf{seq} \setminus \mathsf{seq}^{-1}) \cap \mathsf{sim}$ is *semi-serialisability* allowing a pair of simultaneous occurrences of actions to be serialized in the order given, but not in the reverse order.
- $\mathsf{wdp} = (\mathsf{seq}^{-1} \setminus \mathsf{seq}) \cap \mathsf{sim}$ is *weak dependence* which is the inverse of semi-serialisability.

– $\mathsf{con} = \mathsf{sim} \cap \mathsf{seq} \cap \mathsf{seq}^{-1}$ is *concurrency* identifying actions which can occur simultaneously as well as in any order.

For example, let $u = \{\{c\}\{ab\}\{a\}\{d\}$ be a step sequence over the step alphabet $\theta = \langle \Sigma, \mathsf{sim}, \mathsf{seq} \rangle$, where $\Sigma = \{a, b, c, d\}$ and

$$\begin{array}{l} \mathsf{sim} = \{\langle b,c\rangle, \langle c,b\rangle, \langle b,d\rangle, \langle d,b\rangle, \langle a,b\rangle, \langle b,a\rangle, \langle a,d\rangle, \langle d,a\rangle, \langle a,c\rangle, \langle c,a\rangle\} \\ \mathsf{seq} = \{\langle b,c\rangle, \langle c,b\rangle, \langle b,d\rangle, \langle d,b\rangle, \langle c,d\rangle, \langle d,c\rangle, \langle a,d\rangle, \langle c,a\rangle\} \ . \end{array}$$

Then $\tau = \{\{c\}\{ab\}\{a\}\{d\}, \{c\}\{ab\}\{ad\}, \{abc\}\{a\}\{d\}, \{abc\}\{ad\}\}$ is the trace over θ comprising u. Moreover, the derived dependencies are as follows:

$$\begin{array}{lll} \mathsf{rig} = id_{\{a,b,c,d\}} & \mathsf{inl} = \{\langle c,d\rangle, \langle d,c\rangle\} & \mathsf{ssi} = \{\langle a,b\rangle, \langle b,a\rangle\} \\ \mathsf{sse} = \{\langle a,d\rangle, \langle c,a\rangle\} & \mathsf{wdp} = \{\langle d,a\rangle, \langle a,c\rangle\} & \mathsf{con} = \{\langle b,c\rangle, \langle c,b\rangle, \langle b,d\rangle, \langle d,b\rangle\} \ . \end{array}$$

Events. An *event* is a pair $a^{(i)} = \langle a, i\rangle$ representing the i-th occurrence of action $a \in \Sigma$ in some execution scenario. The default labelling of $a^{(i)}$ is $\ell_{a^{(i)}} = a$. An *event domain* is a finite set of events Δ such that if $a^{(i)} \in \Delta$ and $i > 1$ then also $a^{(i-1)} \in \Delta$. If u is a step sequence, then $\mathsf{occ}(u)$ comprises all events $a^{(i)}$ such that i does not exceed the number of occurrences of a within u, and the position of $a^{(i)}$ within u is given by $\mathsf{pos}_u(a^{(i)}) = j$ if the i-th occurrence of a happens in the j-th step of u.

Order structures. In Mazurkiewicz trace theory, a central role is played by acyclic relations. In the treatment of step traces, a similar role is played by *order structures* $or = \langle \Delta, \rightleftharpoons, \sqsubset \rangle$ such that Δ is an event domain, and the remaining two components are irreflexive binary relations on Δ. An order structure or is supposed to represent an execution scenario involving events in Δ. The notation $x \rightleftharpoons y$ means that x cannot occur simultaneously with y, and $x \sqsubset y$ that x cannot occur later than y, i.e., only before or simultaneously with y. The relation $\rightleftharpoons$ is called *mutex* and $\sqsubset$ *weak causality* (or weak precedence). Moreover, $\prec$, defined as the intersection of mutex and weak causality, is interpreted as *causality* (or precedence). It is assumed that events on a weak causality cycle cannot be in the mutex relation, and that all occurrences of a given action are totally ordered by $\prec^+$.

Order structures with the same domain can be compared, with $or \lhd or'$ denoting that or' *extends* or, as well as intersected, with $\bigcap OS$ denoting the intersection of a nonempty set of order structures OS.

There are three classes of order structures corresponding to the three main classes of acyclic relations used in Mazurkiewicz trace theory.

Saturated order structures (corresponding to total orders). An order structure or is *saturated* if there is no order structure $or' \neq or$ such that $or \lhd or'$, and $\mathsf{or2sr}(or)$ denotes all saturated extensions of an order structure or. Each step sequence u defines a saturated structure $\mathsf{sseq2sr}(u) = \langle \mathsf{occ}(u), \rightleftharpoons, \sqsubset \rangle$ such that, for all $x \neq y \in \mathsf{occ}(u)$:

$$\begin{array}{lll} x \rightleftharpoons y & \textit{if} & \mathsf{pos}_u(x) \neq \mathsf{pos}_u(y) \\ x \sqsubset y & \textit{if} & \mathsf{pos}_u(x) \leq \mathsf{pos}_u(y). \end{array}$$

For a step alphabet θ, this establishes a *one-to-one* correspondence between SSEQ_θ and $\mathsf{sseq2sr}(\mathsf{SSEQ}_\theta)$, with the inverse mapping being denoted by $\mathsf{sr2sseq}$.

Invariant structures (corresponding to causal partial orders). Each step trace τ defines an *invariant (order) structure* $\mathsf{str2ir}(\tau) = \bigcap \mathsf{sseq2sr}(\tau)$. For a step alphabet θ, this establishes a *one-to-one* correspondence between STR_θ and $\mathsf{IR}_\theta = \mathsf{str2ir}(\mathsf{STR}_\theta)$, called the *invariant structures over* θ. The inverse mapping is $\mathsf{ir2str}(ir) = \mathsf{sr2sseq}(\mathsf{or2sr}(ir))$. The set of all step alphabets θ such that $ir \in \mathsf{IR}_\theta$ is denoted by Θ_{ir}.

Dependence structures (corresponding to dependence graphs). Deriving an invariant structure for a step trace following the definition of $\mathsf{str2ir}(\tau)$ is inefficient as the number of step sequences in τ can be huge. Fortunately, by taking a single step sequence in τ, one can extract all the *essential* causal relationships between events in $\mathsf{str2ir}(\tau)$. For a step sequence u over a step alphabet θ, the resulting *dependence (order) structure* is defined as $\mathsf{sseq2or}_\theta(u) = \langle \mathsf{occ}(u), \rightleftharpoons, \sqsubset \rangle$, where, for all $x, y \in \mathsf{occ}(u)$:

$$\begin{array}{lllll} x \rightleftharpoons y & \textit{if} & \langle \ell_x, \ell_y \rangle \in \mathsf{ssi} \cup \mathsf{rig} \cup \mathsf{inl} \cup \mathsf{wdp} & \wedge & \mathsf{pos}_u(x) < \mathsf{pos}_u(y) \\ & \textit{or} & \langle \ell_x, \ell_y \rangle \in \mathsf{ssi} \cup \mathsf{rig} \cup \mathsf{inl} \cup \mathsf{sse} & \wedge & \mathsf{pos}_u(x) > \mathsf{pos}_u(y) \\ x \sqsubset y & \textit{if} & \langle \ell_x, \ell_y \rangle \in \mathsf{ssi} \cup \mathsf{sse} \cup \mathsf{wdp} \cup \mathsf{rig} & \wedge & \mathsf{pos}_u(x) < \mathsf{pos}_u(y) \\ & \textit{or} & \langle \ell_x, \ell_y \rangle \in \mathsf{ssi} \cup \mathsf{sse} & \wedge & \mathsf{pos}_u(x) = \mathsf{pos}_u(y). \end{array} \tag{1}$$

The above definition determines if two events are weakly causally related and/or in the mutex relationship or neither, by looking at their relative order in the step sequence and their dependence as given in θ. For example, the first two lines of (1) mean that if two events that are not in the same step and have labels that cannot be sequentialised when in the same step, are in the mutex relationship.

All step sequences belonging to τ have the same dependence structure, denoted by $\mathsf{sseq2or}_\theta(\tau)$. The *closure* of the latter is the invariant structure induced by τ, $\mathsf{clo}(\mathsf{sseq2or}_\theta(\tau)) = \mathsf{str2ir}(\tau)$, where clo is an operation defined for order structures generalising the transitive closure of acyclic relations. A unique dependence graph *underlying* an invariant structure $ir \in \mathsf{IR}_\theta$, is given by $\mathsf{ir2or}_\theta(ir) = \mathsf{sseq2or}_\theta(\mathsf{ir2str}(ir))$.

Invariant and dependence structures can be used to develop effective representations and algorithmic treatment for step traces.

Linearising alphabets and acyclic invariant structures. Assuming that a non-sequential execution has an equivalent sequential representation amounts, in the current setting, to requiring that a step trace contains a linear step sequence.

A step alphabet θ is *linearising* if every step trace over θ contains a linear step sequence. (In particular, this means that $\mathsf{ssi}_\theta = \varnothing$.) As the next result demonstrates, linearising step alphabets can be characterised by *acyclic* invariant structures defined as those which have acyclic weak causality relation.

Fact 1 *[10] A step alphabet θ is linearising* if and only if *all invariant structures in IR_θ are acyclic.*

Transitive reduction. A Hasse diagram is the most 'efficient' representation of a partial order. The corresponding notion for an acyclic invariant structure $ir = \langle \Delta, \rightleftharpoons, \sqsubset \rangle$ is the *transitive reduction* $\mathsf{red}(ir) = \langle \Delta, \rightleftharpoons \setminus (\rightleftharpoons_r \cup \rightleftharpoons_r^{-1}), \sqsubset \setminus (\sqsubset \circ \sqsubset) \rangle$, where:

$$\rightleftharpoons_r = \{\langle x, y\rangle \in \Delta \times \Delta \mid \exists \langle w, z\rangle : w \rightleftharpoons z \ \wedge \ ((x \sqsubseteq w \sqsubset y \wedge x \sqsubseteq z \sqsubset y) \vee (x \sqsubset w \sqsubseteq y \wedge x \sqsubset z \sqsubseteq y)) \ \}.$$

Then $\mathsf{clo}(\mathsf{red}(ir)) = ir$ and one cannot remove any mutex or weak causality relationship from $\mathsf{red}(ir)$ without losing information about ir.

Alphabets of acyclic invariant structures. Transitive reduction can be used to efficiently check whether an invariant structure represents a step trace over a given step alphabet. To formulate this result, we first introduce auxiliary notation.

For an order structure $or = \langle \Delta, \rightleftharpoons, \sqsubset, \ell \rangle$, we introduce *evidence relations* that capture all possible (seven) combinations of mutex and weak causality between two distinct events:

$$\begin{array}{lcllcl}
\longrightarrow_{or} & = & \{\langle x, y\rangle \mid x \sqsubset y \not\sqsubset x \rightleftharpoons y\} & \longleftarrow_{or} & = & \{\langle x, y\rangle \mid x \not\sqsubset y \sqsubset x \rightleftharpoons y\} \\
\dashrightarrow_{or} & = & \{\langle x, y\rangle \mid x \sqsubset y \not\sqsubset x \not\rightleftharpoons y\} & \dashleftarrow_{or} & = & \{\langle x, y\rangle \mid x \not\sqsubset y \sqsubset x \not\rightleftharpoons y\} \\
\mathrel{---}_{or} & = & \{\langle x, y\rangle \mid x \not\sqsubset y \not\sqsubset x \rightleftharpoons y\} & \dashleftarrow\!\dashrightarrow_{or} & = & \{\langle x, y\rangle \mid x \sqsubset y \sqsubset x \not\rightleftharpoons y\} \\
 & & & \cdots_{or} & = & \{\langle x, y\rangle \mid x \not\sqsubset y \not\sqsubset x \not\rightleftharpoons y \neq x\}
\end{array}$$

In addition we use abstract symbols to represent these relationships between events and for the relations between actions:

$$\mathsf{Evi} = \{\longrightarrow, \longleftarrow, \dashleftarrow\!\dashrightarrow, \mathrel{---}, \dashrightarrow, \dashleftarrow, \cdots\}. \quad \textit{and} \quad \mathsf{Dep} = \{\mathsf{rig}, \mathsf{inl}, \mathsf{ssi}, \mathsf{sse}, \mathsf{wdp}, \mathsf{con}\}$$

Let $x \neq y$ be events of an acyclic order structure or. Then the *evidence* of $\langle x, y\rangle$ is defined as $\mathsf{evi}_{or}(x, y) = \mathsf{e}$, where $\mathsf{e} \in \mathsf{Evi}$ if $\langle x, y\rangle \in \mathsf{e}_{or}$. For a step alphabet θ and two distinct actions, a and b, we define their *dependency* in θ as $\mathsf{dep}_\theta(a, b) = \mathsf{d}$, where $\mathsf{d} \in \mathsf{Dep}$ if $\langle a, b\rangle \in \mathsf{d}_\theta$. Finally, we introduce the *possible dependencies* $\mathsf{PD}_{ir}(x, y)$ which could relate pairs of events in an acyclic invariant structure ir.

$\mathsf{evi}_{ir}(x,y)$	$\longrightarrow$	$\longrightarrow$	$\longrightarrow$	$\dashrightarrow$	$\dashrightarrow$	$\mathrel{--}$
$\mathsf{evi}_{\mathsf{red}(ir)}(x,y)$	$\longrightarrow$	$\mathrel{---}$	$\mathrel{--}$	$\dashrightarrow$	$\mathrel{--}$	$\mathrel{--}$
$\mathsf{PD}_{ir}(x,y)$	wdp rig	inl rig wdp	inl rig sse wdp con	sse	sse con	con
$\mathsf{evi}_{ir}(x,y)$	$\longleftarrow$	$\longleftarrow$	$\longleftarrow$	$\dashleftarrow$	$\dashleftarrow$	$\mathrel{---}$
$\mathsf{evi}_{\mathsf{red}(ir)}(x,y)$	$\longleftarrow$	$\mathrel{---}$	$\mathrel{--}$	$\dashleftarrow$	$\mathrel{--}$	$\mathrel{---}$
$\mathsf{PD}_{ir}(x,y)$	sse rig	inl rig sse	inl rig wdp sse con	wdp	wdp con	inl

Table 1. Possible dependencies of $\langle x, y\rangle$ in an acyclic invariant structure.

We then obtain a characterisation of all step alphabets which match an acyclic invariant structure.

Fact 2 *[10] Let $ir = (\Delta, \rightleftharpoons, \sqsubset)$ be an acyclic invariant structure, and θ be a step alphabet such that ℓ_Δ is included in the action set of θ. Then $ir \in \mathsf{IR}_\theta$* if and only if $\mathsf{dep}_\theta(a,b) \in \mathsf{PD}_{ir}(a,b)$, *for all $a \neq b \in \ell_\Delta$, where:*

$$\mathsf{PD}_{ir}(a,b) = \bigcap \{\mathsf{PD}_{ir}(x,y) \mid x,y \in \Delta \wedge \ell_x = a \wedge \ell_y = b\} \ .$$

3 Synthesising alphabets for acyclic invariant structures

We now extend the characterisation of possible dependencies of events in acyclic invariant structures provided for individual pairs of events in Table 1.

Proposition 1. *Let $ir = (\Delta, \rightleftharpoons, \sqsubset)$ be an acyclic invariant structure, and $a \neq b \in \Delta$. Then $\mathsf{PD}_{ir}(a,b)$ is one of the following 14 sets:*

$\varnothing$	$\{\mathsf{sse}\}$	$\{\mathsf{rig}, \mathsf{inl}\}$	$\{\mathsf{wdp}, \mathsf{con}\}$	$\{\mathsf{rig}, \mathsf{inl}, \mathsf{sse}\}$
$\{\mathsf{rig}\}$	$\{\mathsf{wdp}\}$	$\{\mathsf{rig}, \mathsf{sse}\}$	$\{\mathsf{sse}, \mathsf{con}\}$	$\{\mathsf{rig}, \mathsf{inl}, \mathsf{sse}, \mathsf{wdp}, \mathsf{con}\}$
$\{\mathsf{con}\}$	$\{\mathsf{inl}\}$	$\{\mathsf{rig}, \mathsf{wdp}\}$	$\{\mathsf{rig}, \mathsf{inl}, \mathsf{wdp}\}$	

Proof. From Fact 2 it follows that, for all $x \neq y \in \Delta$, we have $\mathsf{PD}_{ir}(x,y) \in \{R_1, \ldots, R_{11}\}$, where:

$R_1 = \{\mathsf{inl}\}$	$R_2 = \{\mathsf{sse}\}$	$R_3 = \{\mathsf{wdp}\}$
$R_4 = \{\mathsf{con}\}$	$R_5 = \{\mathsf{rig}, \mathsf{wdp}\}$	$R_6 = \{\mathsf{rig}, \mathsf{sse}\}$
$R_7 = \{\mathsf{wdp}, \mathsf{con}\}$	$R_8 = \{\mathsf{con}, \mathsf{sse}\}$	$R_9 = \{\mathsf{inl}, \mathsf{rig}, \mathsf{wdp}\}$
$R_{10} = \{\mathsf{inl}, \mathsf{rig}, \mathsf{sse}\}$	$R_{11} = \{\mathsf{inl}, \mathsf{rig}, \mathsf{sse}, \mathsf{wdp}, \mathsf{con}\}$.	

Since different pairs of events labelled in the same way may generate different R_i's, the set $\mathsf{PD}_{ir}(a,b)$ is, in general, be an intersection $\bigcap_{i \in K} R_i$, where K is any nonempty subset of $\{1, 2, \ldots, 11\}$. Thus, in principle, we have to consider $2^{11} - 1$ intersections. Fortunately, we can simplify this task greatly, as the result follows from the straightforward observations made below:

1. The sets $R_1, \ldots, R_{11}$ are potential values for $\mathsf{PD}_{ir}(a,b)$.
2. The following are also are potential values for $\mathsf{PD}_{ir}(a,b)$:
   - $R_{12} = \varnothing = R_1 \cap R_2$.
   - $R_{13} = \{\mathsf{rig}\} = R_5 \cap R_6$.
   - $R_{14} = \{\mathsf{rig}, \mathsf{inl}\} = R_9 \cap R_{10}$.
3. No new potential values for $\mathsf{PD}_{ir}(a,b)$ can be obtained through intersections $R = \bigcap_{i \in K} R_i$, for nonempty $K \subseteq \{1, 2, \ldots, 11\}$, since we have the following:
   - If $i \in K \cap \{1, \ldots, 4\}$ then $R = R_i$ or $R = R_{12}$.
   - If $11 \in K$ and $R \neq R_{11}$ then $R = \bigcap_{i \in K \setminus \{11\}} R_i$.
   - $|R_5 \cap R_6| = |R_5 \cap R_7| = |R_5 \cap R_{10}| = |R_6 \cap R_9| = |R_6 \cap R_8| = |R_7 \cap R_9| = |R_8 \cap R_{10}| = 1$.
   - $R_5 \cap R_8 = R_6 \cap R_7 = R_7 \cap R_{10} = R_8 \cap R_9 = \varnothing$.
   - $R_5 \cap R_9 = R_5$, $R_6 \cap R_{10} = R_6$, and $R_9 \cap R_{10} = R_{14}$. □

Alphabet synthesis for a single structure. Suppose that we are given an acyclic invariant structure $ir = \langle \Delta, \rightleftharpoons, \sqsubset \rangle$ and asked to find (or *synthesise*) a step alphabet in Θ_{ir}, or conclude that such an alphabet does not exist. We can solve this problem by first computing $\mathsf{PD}_{ir}(a,b)$, for all pairs of distinct actions $\langle a,b \rangle \in \ell_\Delta \times \ell_\Delta$. If any of the computed values is equal to $\varnothing$, we conclude that $\Theta_{ir} = \varnothing$, and so the alphabet synthesis fails. Otherwise, we can find a suitable step alphabet, in the following way. For any pair of distinct actions $\langle a,b \rangle \in \ell_\Delta \times \ell_\Delta$ we can arbitrarily choose a dependency $\mathsf{d}(a,b) \in \mathsf{PD}_{ir}(a,b)$, and then take $\theta = \langle \ell_\Delta, \mathsf{sim}, \mathsf{ser} \rangle$, where $\mathsf{sim} = \mathsf{dep}^{-1}(\{\mathsf{con}, \mathsf{sse}, \mathsf{wdp}\})$ and $\mathsf{seq} = \mathsf{dep}^{-1}(\{\mathsf{con}, \mathsf{sse}, \mathsf{inl}\})$. It is then easily seen that $ir \in \mathsf{IR}_\theta$.

Alphabet synthesis for multiple structures. The solution presented above for a single structure, can be readily generalised to sets of acyclic invariant structures. Suppose now that $IR = \{ir_j \mid j \in J\}$ is a possibly infinite nonempty set of acyclic invariant structures ir_j, each such structure having the domain Δ_j and using actions $\Sigma_j = \ell_{\Delta_j}$. We do not assume that the Σ_j's are the same, but assume that $\Sigma = \bigcup_{j \in J} \Sigma_j$ is finite. The problem is to synthesise a step alphabet in $\bigcap_{j \in J} \Theta_{ir_j}$, or to conclude that such an alphabet does not exist. We can solve this problem by first computing, for all pairs of distinct actions $\langle a,b \rangle \in \Sigma \times \Sigma$:

$$\mathsf{PD}_{IR}(a,b) = \begin{cases} \{\mathsf{inl}, \mathsf{rig}, \mathsf{sse}, \mathsf{wdp}, \mathsf{con}\} & \textit{if } J(a,b) = \varnothing \\ \bigcap_{j \in J(a,b)} \mathsf{PD}_{ir_j}(a,b) & \textit{otherwise}, \end{cases}$$

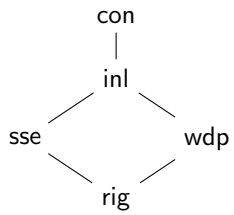
where $J(a,b) = \{j \in J \mid a,b \in \Sigma_j\}$. If any of the computed values is equal to $\varnothing$, we conclude that $\bigcap_{j \in J} \Theta_{ir_j} = \varnothing$, and so the alphabet synthesis fails. Otherwise, we can find a suitable step alphabet, in the following way. For any pair of distinct actions $\langle a,b \rangle \in \Sigma \times \Sigma$ we can arbitrarily choose a dependency $\mathsf{d}(a,b) \in \mathsf{PD}_{ir}(a,b)$, and then take $\theta = \langle \Sigma, \mathsf{sim}, \mathsf{ser} \rangle$, where $\mathsf{sim} = \mathsf{dep}^{-1}(\{\mathsf{con}, \mathsf{sse}, \mathsf{wdp}\})$ and $\mathsf{seq} = \mathsf{dep}^{-1}(\{\mathsf{con}, \mathsf{sse}, \mathsf{inl}\})$. It is then easily seen that $ir_j \in \mathsf{IR}_\theta$, for every $j \in J$.

4 A complete lattice of step alphabets

In this section, we assume that $ir = \langle \Delta, \rightleftharpoons, \sqsubset, \ell \rangle$ is a fixed acyclic invariant structure such that $\Theta_{ir} \neq \varnothing$.

Fact 2 provides a complete characterisation of the step alphabets in Θ_{ir}. Since, in general, there may be several such alphabets, one might ask for a method of assessing their relative 'quality'. The results so far do not provide us with a way of addressing this issue, and in what follows we will show that the step alphabets in Θ_{ir} can, in fact, be ordered in a manner reflecting their potential for creating causal relationships.

We first introduce a weak partial order $\preccurlyeq$ on the dependence symbols in $\mathsf{Dep} \setminus \{\mathsf{ssi}\}$. It is given by the (Hasse) diagram below, where $\preccurlyeq$ is directed from lower to higher nodes, e.g., $\mathsf{rig} \preccurlyeq \mathsf{con}$. Intuitively, $\mathsf{d}' \preccurlyeq \mathsf{d}$ means that d' introduces at least as much causal relationships as d.



The order on dependencies can then be used to order step alphabets.

Definition 1 (alphabet ordering). *Let $\theta, \theta' \in \Theta_{ir}$. Then $\theta \preccurlyeq \theta'$ if, for all $a \neq b \in \ell_\Delta$, $\mathsf{dep}_\theta(a,b) \preccurlyeq \mathsf{dep}_{\theta'}(a,b)$.*

The above definition is purely *syntactical*. However, it can be justified by compelling *semantical* arguments as well.

Proposition 2. *If $\theta', \theta'' \in \Theta_{ir}$, then there exists $\theta \in \Theta_{ir}$ such that:*

$$\mathsf{ir2or}_\theta(ir) \lhd \mathsf{ir2or}_{\theta'}(ir) \quad \textit{and} \quad \mathsf{ir2or}_\theta(ir) \lhd \mathsf{ir2or}_{\theta''}(ir).$$

Proof. For any $a \neq b \in \ell_\Delta$ we can choose from $\mathsf{PD}_{ir}(a,b)$ a dependence which is maximal from the point of view of $\preccurlyeq$. Then the result is straightforward except for the case that $\mathsf{wdp} \preccurlyeq \mathsf{inl}$ (and, symmetrically, $\mathsf{sse} \preccurlyeq \mathsf{inl}$) since wdp can generate ‹- - in a dependence graph. However, when both inl and wdp are possible dependencies but con is not, then wdp can only generate $\longrightarrow$, and so inl generates strictly less causal relationships and can be chosen for θ. □

That is, for each pair of step alphabets in Θ_{ir}, one can use $\preccurlyeq$ to find a step alphabet which imposes fewer underlying causal relationships than the original two alphabets. It is worth observing that a stronger result, like "$\theta \preccurlyeq \theta'$ implies $\mathsf{ir2or}_{\theta'}(ir) \lhd \mathsf{ir2or}_\theta(ir)$", does not in general hold. Consider, for example, an acyclic invariant structure ir shown in Figure 1, and five step alphabets θ_i such that:

$$\begin{array}{lll} \mathsf{dep}_{\theta_1}(a,b) = \mathsf{rig} & \mathsf{dep}_{\theta_2}(a,b) = \mathsf{wdp} & \mathsf{dep}_{\theta_3}(a,b) = \mathsf{con} \\ \mathsf{dep}_{\theta_4}(a,b) = \mathsf{sse} & \mathsf{dep}_{\theta_5}(a,b) = \mathsf{inl}. & \end{array}$$

In every set of possible dependencies there exist the most restrictive as well as the most permissive choice (given by $\preccurlyeq$). However, this does not mean that $\theta \preccurlyeq \theta'$ implies that the underlying dependence structures corresponding to the same invariant structure are comparable w.r.t. $\lhd$ (see $\mathsf{ir2or}_{\theta_4}(ir)$ and $\mathsf{ir2or}_{\theta_5}(ir)$ depicted in Figure 1).

We then obtain as the *main result* of this paper that there is always the most restrictive as well as the most permissive step alphabet with which a given acyclic invariant structure is consistent.

Theorem 1. *$\langle \Theta_{ir}, \preccurlyeq \rangle$ is a complete lattice.*

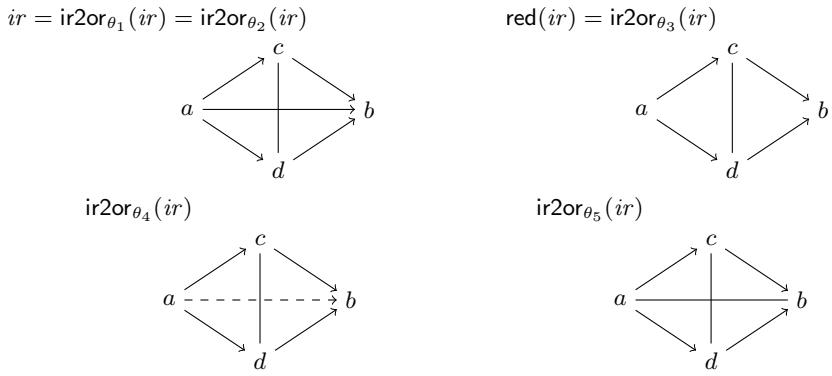


Fig. 1. An acyclic invariant structure ir together with its transitive reduction and underlying dependence structures w.r.t. five different step alphabets.

Proof. Note that all 13 nonempty sets of dependence symbols from Proposition 1 form complete lattices with the relation $\preccurlyeq$.

Indeed, $\{\mathsf{sse}\}$, $\{\mathsf{rig}\}$, $\{\mathsf{wdp}\}$, $\{\mathsf{con}\}$, and $\{\mathsf{inl}\}$ are singletons (and so form trivial lattices), and in $\{\mathsf{rig}, \mathsf{inl}\}$, $\{\mathsf{wdp}, \mathsf{con}\}$, $\{\mathsf{rig}, \mathsf{sse}\}$, $\{\mathsf{sse}, \mathsf{con}\}$, and $\{\mathsf{rig}, \mathsf{wdp}\}$, we have the first element in the relation $\preccurlyeq$ with the second one.

Only three larger sets remain to be considered: $\{\mathsf{rig}, \mathsf{inl}, \mathsf{sse}\}$, where $\mathsf{rig} \preccurlyeq \mathsf{inl} \preccurlyeq \mathsf{sse}$; $\{\mathsf{rig}, \mathsf{inl}, \mathsf{wdp}\}$, where $\mathsf{rig} \preccurlyeq \mathsf{inl} \preccurlyeq \mathsf{wdp}$; and $\{\mathsf{rig}, \mathsf{inl}, \mathsf{sse}, \mathsf{wdp}, \mathsf{con}\}$ which is the whole $\mathsf{Dep} \setminus \{\mathsf{ssi}\}$ used in the definition of $\preccurlyeq$.

Since, for a fixed finite set of actions Σ present in ir, the relation $\preccurlyeq$ on step alphabets is defined component-wise (where every pair of actions is a single component), we are dealing with a finite product of complete lattices. Such a product obviously forms a complete lattice, which ends the proof. □

To conclude, as $\langle \Theta_{ir}, \preccurlyeq \rangle$ is a finite complete lattice, it has a maximal element θ_{ir}^{max} and a minimal element θ_{ir}^{min}. They are given by respectively taking, for all $a \neq b \in \ell_\Delta$:

$$\mathsf{dep}_{\theta_{ir}^{max}}(a,b) = \max_{\preccurlyeq} \mathsf{PD}_{ir}(a,b) \quad \textit{and} \quad \mathsf{dep}_{\theta_{ir}^{min}}(a,b) = \min_{\preccurlyeq} \mathsf{PD}_{ir}(a,b)\ .$$

5 Mazurkiewicz traces

As noted in [8], Mazurkiewicz traces with their original sequential semantics and partial orders as structural counterparts, may be seen as a special subclass of step traces. The manner in which we obtain the corresponding step traces is to consider a subclass of step alphabets Θ_{sim}. Each such step alphabet has an empty relation sim (hence all steps are singletons). This implies that the only two interesting dependence symbols are inl and rig. As a matter of fact, they

play the role of the independence relation and the dependence relation in the setting of Mazurkiewicz traces.

Thus, in this case, we consider an acyclic invariant structure $ir = \langle \Delta, \rightleftharpoons, \sqsubset, \ell \rangle$ such that $\rightleftharpoons$ is equal to $(\Delta \times \Delta) \setminus id_\Delta$, and obtain:

Theorem 2. *$\theta \in \Theta_{ir} \cap \Theta_{\mathsf{sim}}$ if and only if $\mathsf{dep}_\theta(a,b) \in \mathsf{PD}_{ir}(a,b) \cap \{\mathsf{rig}, \mathsf{inl}\}$, for all $a \neq b \in \ell_\Delta$.*

Proof. Follows from Fact 2 and the definition of Θ_{sim}. □

Recalling from [11] the relation $\leq$ defined on step alphabets ($\theta' \leq \theta$ if at the same time $\mathsf{sim}_{\theta'} \subseteq \mathsf{sim}_\theta$ and $\mathsf{seq}_{\theta'} \subseteq \mathsf{seq}_\theta$) we also get

Theorem 3. *$\langle \Theta_{ir} \cap \Theta_{\mathsf{sim}}, \leq \rangle$ is a complete lattice.*

Proof. Since $\mathsf{rig} \preccurlyeq \mathsf{inl}$ and, in this case $\theta \preccurlyeq \theta'$ iff $\theta \leq \theta'$, this is a direct consequence of Theorems 1 and 2. □

6 Conclusions

In this paper we carried out a deeper discussion on the structure of step alphabets suitable for a given labelled acyclic invariant structure. The main result proven here states that one can always choose not only the most restrictive alphabet, i.e., maximal in the sense of the extension relation $\lhd$ (proven to hold in the general case in [11]), but also the most permissive one, i.e., minimal in the sense of $\lhd$ (taking advantage of the acyclicity of an invariant structure in a significant way).

Related work. The papers [13, 17] discuss relationships between individual events related to causality and simultaneity (mutual exclusion). The main difference between these works and that presented here is the assumed semantics (step semantics, in our case, rather than a sequential semantics). There is also a difference of the meaning of some relationships. The observations discussed in [13, 17] are (complete or incomplete) sets of processes/sequences produced by a system. On this basis, one attempts to classify causality and mutual exclusion in all/some executions of the system. In our case, this concerns causality in a single process (which is projected to the whole system), and mutual exclusion between two events means that such events do not occur in the same step rather than not in the same process as in the approach of [13, 17]. Moreover, in order to determine relationships in [13] an idea similar to ours (comparison of arcs in original/reduced order structure with those appearing in the closed one) is employed. Adding more concrete observations implies going up (in the direction of more restrictive ones) in lattice of determined relationships. On the other hand, [17] defines the strength of a relationship by quantifying its presence in the whole history of the model processes. This corresponds to our procedure of determining possible dependencies (using intersection, i.e., through universal quantification).

Future work. We plan to investigate efficient methods of step alphabet synthesis for invariant structures without specified labelling. A possible solution is to use the method from this paper after taking an injective labelling. However, such an attempt does not really yield interesting results as it is clearly desirable to ask for an optimal (from the point of view of the step alphabet size) solution, and investigate whether assuming acyclicity of the initial structure makes any difference.

Acknowledgement. We are grateful to the reviewers for their useful comments and suggestions for improvement. This research was supported by EPSRC (grant EP/K001698/1 UNCOVER), the Polish National Science Center (grant No.2013/-09/D/ST6/03928), and NSERC of Canada (grant RGPIN6466-15).

References

1. Baldan, P., Busi, N., Corradini, A., Pinna, G.M.: Domain and event structure semantics for Petri nets with read and inhibitor arcs. Theoretical Computer Science **323**, 129–189 (2004)
2. Bergstra, J., Ponse, A., Smolka, S. (eds.): Handbook of Process Algebra. Elsevier, Amsterdam (2001)
3. Diekert, V., Rozenberg, G. (eds.): The Book of Traces. World Scientific, River Edge, NJ, USA (1995)
4. Ehrenfeucht, A., Rozenberg, G.: Reaction systems. Fundamenta Informaticae **75**(1-4), 263–280 (2007)
5. Grabowski, J.: On partial languages. Fundamenta Informaticae **4**(2), 125–147 (1983)
6. Hoogeboom, H.J., Rozenberg, G.: Dependence graphs. In: The Book of Traces, pp. 43–67. World Scientific (1995)
7. Janicki, R., Kleijn, J., Koutny, M., Mikulski, Ł.: Characterising concurrent histories. Fundamenta Informaticae **139**, 21–42 (2015)
8. Janicki, R., Kleijn, J., Koutny, M., Mikulski, Ł.: Order structures for subclasses of generalised traces. In: Language and Automata Theory and Applications, *Lecture Notes in Computer Science*, vol. 8977, pp. 689–700. Springer (2015)
9. Janicki, R., Kleijn, J., Koutny, M., Mikulski, Ł.: Step traces. Acta Informatica **53**, 35–65 (2016)
10. Janicki, R., Kleijn, J., Koutny, M., Mikulski, Ł.: Alphabets of acyclic invariant structures. Fundamenta Informaticae (accepted) pp. 1–12 (2017). See http://folco.mat.umk.pl/papers/JKKM-2017.pdf
11. Janicki, R., Kleijn, J., Koutny, M., Mikulski, Ł.: Invariant structures and dependence relations. Fundamenta Informaticae (accepted) pp. 1–27 (2017)
12. Janicki, R., Koutny, M.: Structure of concurrency. Theoretical Computer Science **112**(1), 5–52 (1993)
13. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering block-structured process models from incomplete event logs. In: Application and Theory of Petri Nets and Concurrency, pp. 91–110 (2014)
14. Mazurkiewicz, A.: Concurrent program schemes and their interpretations. DAIMI Rep. PB 78, Aarhus University (1977)

15. Mazurkiewicz, A.: Basic notions of trace theory. In: Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency, *Lecture Notes in Computer Science*, vol. 354, pp. 285–363. Springer (1988)
16. Montanari, U., Rossi, F.: Contextual nets. Acta Informatica **32**, 545–596 (1995)
17. Polyvyanyy, A., Weidlich, M., Conforti, R., Rosa, M.L., ter Hofstede, A.H.M.: The 4c spectrum of fundamental behavioral relations for concurrent systems. In: Application and Theory of Petri Nets and Concurrency, pp. 210–232 (2014)
18. Pratt, V.: Modeling concurrency with partial orders. International Journal of Parallel Programming **15**(1), 33–71 (1986)
19. Rozenberg, G., Verraedt, R.: Subset languages of Petri nets. Part I. Theoretical Computer Science **26**, 301–326 (1983)
20. Vogler, W.: Partial order semantics and read arcs. Theoretical Computer Science **286**(1), 33–63 (2002)