Józef Winkowski

TOWARDS A UNIVERSAL MODEL OF ACTION [1]

Nr 1035

Warsaw, October 2016

[1]Report Nr 1035 of the Institute of Computer Science of the Polish Academy of Sciences. The work has been supported by the Institute of Computer Science of the Polish Academy of Sciences. Updated 2017.5.6.

Abstract: In the paper a model of action is described that is universal in the sense that it may serve to represent actions of any kind: discrete, continuous, or partially discrete and partially continuous. The model is founded on the assumption that an action is executed in a universe of objects. It describes how the possible executions change the situation of involved objects. It exploits the fact that executions are represented such that fragments of executions are represented such that their closed segments admit only trivial automorphisms. The model has an algebraic strucure and it is a directed complete partial order.

Keywords: Action, object, object instance, occurrence of object instance, execution, execution segment, execution fragment, execution structure, lposet, pomset, sequential composition, parallel composition, independent component, prefix order.

PEWNA PROPOZYCJA UNIWERSALNEGO MODELU AKCJI

Streszczenie: Praca zawiera opis pewnego modelu akcji, który jest uniwersalny w tym sensie, że może służyć do reprezentowania akcji dowolnego rodzaju: dyskretnych, ciągłych, lub częściowo dyskretnych i częściowo ciągłych. Model ten opiera się na założeniu, że akcja jest wykonywana w pewnym środowisku obiektów. Opisuje jak możliwe wykonania akcji zmieniają sytuacje zaangażowanych obiektów. Wykorzystuje fakt, że fragmenty wykonań akcji są reprezentowane tak, że ich ograniczone segmenty mają jedynie trywialne automorfizmy. Model ma pewną strukturę algebraiczną i częściowy porządek przy którym podzbiory skierowane mają kresy górne.

Słowa kluczowe: Akcja, obiekt, instancja obiektu, wystąpienie instancji obiektu, wykonanie, segment wykonania, fragment wykonania, struktura wykonań, lposet, pomset, złożenie szeregowe, złożenie równoległe, niezależna składowa, porządek prefiksowy.

1 Introduction

For discrete actions like behaviours of discrete concurrent systems there are models such as labelled event structures which reflect well concurrency and nondeterminism (cf. [7], [18] and [25]). Such models are based on intrinsic properties of modelled actions and they are compositional in the sense that the model of a compound action can be obtained by combining models of component actions (cf. [24]). Usually, they represent actions up to an equivalence (cf. [9]). However, they do not offer means to represent continuous actions.

Continuous actions like those considered in control theory are usually modelled by differential equations which reflect the relations between variables representing the input of a system and variables representing the system state and output (cf. [10]).

Behaviours of systems working in a continuous way can also be represented as the behaviours of continuous Petri nets such as those described in [2] and [3]. In such nets transitions are regarded as continuous activities transforming some products into some other products and the behaviour consists of execution of such activities with intesities given by a control path.

Behaviours of hybrid systems like those including real-world components, which work in a continuous way, and controlling components, which operate in discrete steps, are modelled as a combination of continuous and discrete activities (cf. [8], [16], and [17]).

However, a typical way of representing in such models the possible continuity of work is to use a reference to a global time which is not an intrinsic property and may be artificial or uncertain.

An attempt of obtaining a global time independent model of actions with continuous components has been described in [11]. The idea presented in [11] relies on a generalization of labelled event structures by dividing the set of possible events into a subset of discrete events and a family of subsets of remaining events, each subset of the family provided with a local time. However, this idea does not seem to be further developed.

In this paper we propose a model of action, a model which uses only intrinsic properties of the modelled actions, is compositional, and is universal enough to represent in the same way actions of any kind: discrete, continuous, or partially discrete and partially continuous. In particular, we want to avoid a reference to a global time. So, instead of representing the continuity in actions with the aid of such a reference, we develop algebraic

tools allowing to represent the possible continuity of action executions as infinite divisibility of such executions into segments.

A model of action of this type is needed for representing and relating behaviours of systems without a need of inventing a special model in every particular case.

Our model of action is founded on the assumption that an action is executed in a universe of objects, each object with a set of possible instances corresponding to its states and other temporary features, each fragment of execution transforming instances of objects (cf. [23]). Formally it is a specific labelled partially ordered set (lposet), called execution structure. Each labelled element of such an lposet represents an occurrence of object instance represented by the label during a possible execution. The partial order represents the causal dependency of occurrences of object instances, i.e., how occurrences of object instances arise from occurrences of object instances. Maximal downwards-closed subsets without incomparable occurrences of instances of the same object represent the possible executions.

Execution structures are similar to labelled event structures, but instead of occurrences of atomic actions they represent occurrences of object instances, and instead of conflict relations which define the possible executions indirectly they represent the possible executions directly.

As in the case of labelled event structures, different execution structures may represent the same action, and formally it can be decided with the aid of the respective notion of equivalence. In particular, for execution structures we have a notion of history preserving equivalence which is similar to that for event structures.

In the case of execution structures this notion is especially attractive since it leads to a very simple representation of equivalence classes and, consequently, to a very simple model of action. This model, called a reduced execution structure, consists of isomorphism classes of execution fragments and has an algebraic structure.

The representation of actions by reduced execution structures simplifies their studies. As reduced execution structures are sets of abstract execution fragments, they are partially ordered by inclusion. Consequently, a simulation of an action by another action can be defined as the usual inclusion of one reduced execution structure in another reduced execution structure.

The paper is organized as follows. In section 2 we introduce the basics on partially ordered and partially ordered labelled sets. In section 3 we define execution structures of actions and describe their properties. In section 4 we define a history preserving equivalence of execution structures. In section 5 we describe executions and their fragments as independent enti-

ties and we define operations on execution fragments and the corresponding partial algebras. In section 6 we describe how classes of history preserving equivalent execution structures can be represented by subsets of such algebras. In Appendix we present those proofs of presented statements which cannot be seen from the context.

2 Preliminaries

In this section we represent the necessary tools related to order relations exploited in the paper.

Given a partial order $\leq$ on a set X, i.e. a binary relation which is reflexive, anti-symmetric and transitive, $P = (X, \leq)$ is said to be a *partially ordered set*, or briefly a *poset*, a subset $Y \subseteq X$ is said to be *downwards-closed* iff $x \leq y$ for some $y \in Y$ implies $x \in Y$, *upward closed* iff $y \leq x$ for some $y \in Y$ implies $x \in Y$, *bounded* iff it has an upper bound, i.e. an element $z \in X$ such that $y \leq z$ for all $y \in Y$, *directed* iff for every $x, y \in Y$ there exists in Y an upper bound z of $\{x, y\}$, a *chain* iff $x \leq y$ or $y \leq x$ for all $x, y \in Y$, an *antichain* iff $x < y$ does not hold for any $x, y \in Y$, and P is said to be *directed complete* or a *directed complete partial order* (a DCPO) iff every of its directed subsets has the least upper bound. A Scott topology on the underlying set X of such poset is the topology in which a subset $U \subseteq X$ is open iff it is upward closed and does not contain the least upper bound of any directed subset of $X - U$.

A *cross-section* of P is a maximal antichain Z of P such that, for every $x, y \in X$ for which $x \leq y$ and $x \leq z'$ and $z'' \leq y$ with some $z', z'' \in Z$, there exists $z \in Z$ such that $x \leq z \leq y$.

Note that if Z is a cross-section of P then the relation $\leq$ is the transitive closure of the union of the restrictions of the relation $\leq$ to the subsets $Z^- = \{x \in X : x \leq z \text{ for some } z \in Z\}$ and $Z^+ = \{x \in X : z \leq x \text{ for some } z \in Z\}$.

A cross-section Z' is said to *precede* a cross-section Z'', written as $Z' \preceq Z''$, iff $Z'^- \subseteq Z''^-$.

The set of cross-sections of P has an important property.

2.1. Proposition. The relation $\preceq$ is a partial order on the set of cross-sections of P. For every two cross-sections Z' and Z'' of P there exist the greatest lower bound $Z' \wedge Z''$ and the least upper bound $Z' \vee Z''$ of Z' and Z'' with respect to $\preceq$, where
$Z' \wedge Z'' = \{z \in Z' \cup Z'' : z \leq z' \text{ for some } z' \in Z' \text{ and } z \leq z'' \text{ for some } z'' \in Z''\}$,

$Z' \vee Z'' = \{z \in Z' \cup Z'' : z' \leq z$ for some $z' \in Z'$ and $z'' \leq z$ for some $z'' \in Z''\}$.
Moreover, the set of cross-sections of P with the operations thus defined is a distributive lattice. ♯

Given cross-sections Z' and Z'' of P such that $Z' \preceq Z''$, a *closed segment* or briefly a *segment* of P from Z' to Z'' is defined as the restriction of P to the set $[Z', Z''] = Z''^{-} - Z'^{-}$, written as $P|[Z', Z'']$. If Z' (resp., Z'') is not specified then $[., Z'']$ (resp., $[Z', .]$) denotes the set Z''^{-} (resp., Z'^{+}), $P|[., Z'']$ denotes the restriction of P to the subset Z''^{-}, $P|[Z', .]$ denotes the restriction of P to the subset Z'^{+}, and these restrictions are also called segments of P. A segment of P that is contained in a segment Q of P is called a *subsegment* of Q. In particular, a segment of $P|[Y', Y'']$ such that $Z' \preceq Y' \preceq Y'' \preceq Z''$ is a subsegment of $P|[Z', Z'']$. If $Z' \neq Y'$ or $Y'' \neq Z''$ (resp.: if $Z' = Y'$, or if $Y'' = Z''$) then we call $P|[Y', Y'']$ a *proper* (resp.: an *initial*, or a *final*) subsegment of $P|[Z', Z'']$.

Given a function f defined on P, an *initial segment* of f is defined as the restriction of f an initial segment of P.

Given a cross-section c of P, the restrictions of P to the subsets $c^{-} = \{x \in X : x \leq z$ for some $z \in c\}$ and $c^{+} = \{x \in X : z \leq x$ for some $z \in c\}$ are called respectively the *head* and the *tail* of P with respect to c, and written respectively as $head(P, c)$ and $tail(P, c)$.

The *sequential decomposition* of P at a cross-section c is the pair $(head(P, c), tail(P, c)$ and P is said to *consist* of $head(P, c)$ *followed* by $tail(P, c)$.

A *parallel decomposition* of $P = (X, \leq)$ is a pair $s = (s^F, s^S)$ of disjoint subsets s^F and s^S of X such that $s^F \cup s^S = X$ and $x' \leq x''$ only if x' and x'' are both in one of these subsets. Given such a decomposition, the restriction of P to each of the sets s^F and s^S is said to be an *independent component* of P. Note that P itself is an independent component of P.

A *fragment* or a *component* of P is an independent component C of a segment S of P such that the set of minimal elements of C is a cross-section of C and it is contained in the cross-section of P that consists of minimal elements of S.

Given a partial order $\leq$ on a set X and a function $l : X \to W$ that assigns to every $x \in X$ a label $l(x)$ from a set W, the triple $L = (X, \leq, l)$ is called a *labelled partially ordered set*, or briefly an *lposet*.

If the set of elements of $L = (X, \leq, l)$ that are are maximal (resp., maximal) with respect to $\leq$ is a cross-section of $(X, \leq)$ then we call the restriction of L to this set the *origin* (resp., the *end*) of L, write it as

origin(L) (resp., as *end*(L)). If *origin*(L) and *end*(L) exist then L is said to be *closed*.

A *sequential decomposition* (resp., a *parallel decomposition*) of L is defined as the partition of L into two parts corresponding to a sequential decomposition (resp., to a parallel decomposition) of the underlying poset $P = (X, \leq)$. A *chain* (resp., an *anti-chain*, a *cross-section*, a *segment*, an *independent component*, a *fragment*) of L, is defined as the restriction of L to a chain (resp., to an antichain, to a cross-section, to a segment, to an independent component, to a fragment) of P. A cross-section c of L is said to *precede* a cross-section c' of L, and it is written as $c \preceq c'$, iff $Z \preceq Z'$ for Z being the restriction of c to P and Z' being the restriction of c' to P.

Given a cross-section c of L, the restrictions of L to the subsets $c^- = \{x \in X : x \leq z \text{ for some } z \in c\}$ and $c^+ = \{x \in X : z \leq x \text{ for some } z \in c\}$ are called respectively the *head* and the *tail* of L with respect to c, and written respectively as *head*(L, c) and *tail*(L, c).

Given a parallel decomposition $s = (s^F, s^S)$ of L, the restrictions of L to the subsets s^F and s^S are called respectively the *first independent component* and the *second independent component* of L according to s, and written respectively as *first*(E, s) and *second*(E, s). Every lposet that is the first or the second independent component of L according to a parallel decomposition of L is called an *independent component* of L.

An lposet L' is said to *occur* in L if it is a fragment of L.

By **LPOSETS** we denote the category of lposets and their morphisms, where a *morphism* from an lposet $L = (X, \leq, l)$ to an lposet $L' = (X', \leq', l')$ is defined as a mapping $b : X \to X'$ such that, for all x and y, $x \leq y$ iff $b(x) \leq' b(y)$, and, for all x, $l(x) = l'(b(x))$. In the category **LPOSETS** a morphism from $L = (X, \leq, l)$ to $L' = (X', \leq', l')$ is an *isomorphism* iff it is bijective, and it is an *automorphism* iff it is bijective and $L = L'$. If there exists an isomorphism from an lposet L to an lposet L' then we say that L and $L'$are *isomorphic*. A *partially ordered multiset*, or briefly a *pomset*, is defined as an isomorphism class ξ of lposets. Each lposet that belongs to such a class ξ is called an *instance* of ξ. The pomset corresponding to an lposet L is written as $[L]$.

3 Execution structures

In general, an action is a manner in which a mechanism or instrument operates. In this paper we think of actions as of activities executed in a universe of objects (memory locations, messages, etc.), each object with a set of

possible instances corresponding to its states and other temporary features (contents, positions, etc.), each execution and its fragments transforming instances of objects.

A universe of objects is defined as follows.

3.1. Definition. A *universe of objects* is $\mathbf{U} = (V, W, ob)$, where V is a set of *objects*, W is a set of *instances* of objects from V (a set of *object instances*), and ob is a mapping $ob : W \to V$ that assigns the respective object to each of its instances. ♯

3.2. Example. Suppose that V' is the union of the one-element set containing a clock to indicate time, denoted *clock*, and of a set of tanks to keep a liquid. Define objects to be elements of V'. Define instances of the clock to be pairs $(clock, t)$ where t is a real number representing the indicated time. Define instances of $v \in V'$ to be pairs $w = (v, s)$, where $s \geq 0$ is the quantity of liquid in v. Define W' to be the set of possible instances of objects $v \in V'$. Define $ob' : W' \to V'$ to be the mapping such that $(clock, t) \mapsto clock$ and $(v, s) \mapsto v$ for every tank $v \in V'$. Then $\mathbf{U}' = (V', W', ob')$ is a universe of objects. ♯

In order to define structures which will be used to represent actions it is convenient to define first structures which will be used to represent possible concrete execution fragments of some unspecified actions.

3.3. Definition. A *concrete execution fragment* in a universe $\mathbf{U} = (V, W, ob)$ of objects is an lposet $E = (X, \leq, l)$, where X is a set (of *occurrences* in E of (instances of) objects), $l : X \to W$ is a mapping (a *labelling* that assigns the respective object instance to each occurrence of this object instance), and $\leq$ is a partial order (the *causal dependency relation* of E) such that

(1) for every object $v \in V$, the set $X|v = \{x \in X : ob(l(x)) = v\}$ is either empty or it is a maximal chain and has an element in every cross-secton of E,

(2) every element of X belongs to a cross-section of E,

(3) no closed segment of E is isomorphic to its proper closed subsegment,

(4) the set of minimal elements of E is a cross-section. ♯

Condition (1) characterizes concrete execution fragments and their properties. A concrete execution fragment E is a partially ordered set of occurrences of object instances. Each object may have many instances and each of them may have in E many occurrences. A cross-section of E represents a possible state of E. Condition (1) says that the occurrences of instances of every object which takes part in E form a maximal chain, that E contains all information on such object, and that every possible state of E contains a part of this information. Condition (2) says that every occurrence of an object in E belongs to a possible state of E. Condition (3) says that all what happens to the involved objects is sufficient to distinguish every closed segment of E from its proper closed subsegments and, consequently, to reflect the progress of E. It implies that even in the case of continuous execution fragments the lposets representing closed segments of execution fragments admit only trivial automorphisms, a property that will be crucial for simplifying a natural equivalence of models of actions. Condition (4) means that E has an initial state.

3.4. Definition. An *abstract execution fragment*, or briefly an *execution fragment* in a universe $\mathbf{U} = (V, W, ob)$ of objects is an isomorphism class ξ of concrete execution fragments in this universe. ♯

Actions can be represented by branching structures called execution structures. Branches of such structures will represent concrete executions of represented actions.

3.5. Definition. An *execution structure* in $\mathbf{U}$ is an lposet $L = (X, \leq, l)$, where X is a set (of *occurrences* of (instances of) objects), $l : X \to W$ is a mapping (a *labelling* that assigns the respective object instance to each occurrence of this object instance), and $\leq$ is a partial order (the *causal dependency relation* of L), such that

(1) every restriction E of L to a maximal downwards-closed subset which does not contain incomparable occurrences of instances of an object is an execution fragment in $\mathbf{U}$, called a *concrete execution* of L,

(2) if a cross-section c of a concrete execution E of L contains the origin of a fragment F of concrete execution of L then there exists a concrete execution E' of L such that $head(E', c) = head(E, c)$ and F is isomorphic to an independent component of an initial segment of $tail(E', c)$,

(3) every concrete execution E in $\mathbf{U}$ such that every initial segment of E is isomorphic to an initial segment of a concrete execution of L is an initial segment of a concrete execution of L.

Concrete executions of L and their fragments are called *concrete execution fragments* of L. Concrete executions of L and their segments and fragments with the origins consisting of minimal elements of L are said to be *initial*. ♯

Condition (1) characterizes those subsets of X which are underlying sets of concrete executions of the represented action. Condition (2) means that the consequences of each bounded initial execution segment $head(E, c)$ of a concrete execution E depend only on object instances which occur in the reached cross-section $end(head(E, c)) = c$. Condition (3) describes the structure of initial segments of L.

Execution structures are similar to labelled trees used in [15] to represent behaviours of communicating systems. They more sophisticated because they reflect explicitly the concurrency existing in actions. Representation of concrete executions of actions in terms of object instances and their occurrences allows us to describe in the same way both discrete and continuous execution fragments.

3.6. Example. Consider an occurrence Petri net (B, E, F) in which every $e \in E$ represents an occurrence of an event $h(e)$ of an elementary net system H as described in [20], every $b \in B$ represents a holding of $h(b)$, where $h(b)$ is a condition of H or the negation of a condition of H, and $F \subseteq B \times E \cup E \times B$ represents the causality relation. In such net the reflexive and transitive closure F^* of F is a partial order on $B \cup E$.

The behaviour of H is an action which can be represented by the execution structure $L(H) = (X_{L(H)}, \leq_{L(H)}, l_{L(H)})$ in the universe of conditions of H where $X_{L(H)}$ is the set B, $b \leq_{L(H)} b'$ iff bF^*b', and $l_{L(H)}(b) = h(b)$.

Every $\{b\}^{\downarrow} = \{x \in B \cup E : xF^*b\}$ repreents the history of reaching the holding b of $h(b)$ (cf. [4]). Every concrete execution G of $L(H)$ is the restriction of the occurrence net (B, E, F) to a maximal subset of B without branching at elements of B. ♯

3.7. Example. Suppose that we want to represent the behaviour of a tank v as in Example 3.2 and the relation of this behaviour to the flow of real time. Then we consider it as the behaviour of the system consisting of

clock and v and represent it by the execution structure $L(clock, v)$ in the universe $\mathbf{U}'$ such that
$L(clock, v) = (X_{L(clock,v)}, \leq_{L(clock,v)}, l_{L(clock,v)})$ where
$X_{L(clock,v)} = \{clock\} \times [0, \infty) \cup \{v\} \times F$ with the set F of real-valued functions defined on intervals $[0, t]$, each function describing how the quantity of liquid in v depends on the real time,
$\leq_{L(clock,v)}$ is a partial order on $X_{L(clock,v)}$ such that
$(clock, t) \leq_{L(clock,v)} (clock, t')$ for $t \leq t'$,
$(v, f) \leq_{L(clock,v)} (clock, t)$ for f with the domain $[0, t']$ such that $t' < t$,
$(v, f) \leq_{L(clock,v)} (v, f')$ for f being an initial segment of f',
$l_{L(clock,v)}(clock, t) = (clock, t)$ and $l_{L(clock,v)}(v, f) = f(t)$ for f with the domain $[0, t]$.

Every concrete execution E of $L(clock, v)$ is the restriction of the lposet $L(clock, v)$ to a subset $\{clock\} \times [0, \infty) \cup \{v\} \times \{f\}^{\downarrow}$ of $X_{L(clock,v)}$, where $\{f\}^{\downarrow}$ is the set of bounded initial segments of a function $f : [0, \infty) \to [0, \infty)$.
♯

3.8. Example. Suppose that we want to represent the behaviour of a tank v without relating it to the flow of real time which cannot be observed. Then the role of time scale must be played by an intrinsic time scale that can be derived from what happens in v. To this end it suffices to replace every function f which describes how the quantity of liquid in v depends on the non-observable real time t by its modified version $\widehat{f}$ which describes how the quantity of liquid in v depends on the observable variation of f in the corresponding interval $[0, t]$, where the variation of f in the interval $[0, t]$, $var(f; 0, t)$, is defined as the least upper bound of the set of quantities $|f(t_1) - f(t_0)| + ... + |f(t_n) - f(t_{n-1})|$, each quantity corresponding to a partition $t_0 = 0 < t_1 < ... < t_n = t$ of the interval $[0, t]$.

The behaviour of v can be represented by the execution structure
$L(v) = (X_{L(v)}, \leq_{L(v)}, l_{L(v)})$ in $\mathbf{U}'$ where
$X_{L(v)} = \{v\} \times F'$ with the set F' of real-valued functions, each function being the modified version $\widehat{f}$ of a function f defined on an interval $[0, t]$ and describing how the quantity of liquid in v depends on the non-observable real time,
$\leq_{L(v)}$ is the partial order on $X_{L(v)}$ such that $(v, \widehat{f}) \leq_{L(v)} (v, \widehat{f'})$ for $\widehat{f}$ being an initial segment of $\widehat{f'}$,
$l_{L(v)}(v, \widehat{f}) = (v, f(t))$ for f with the domain $[0, t]$.

Every concrete execution E of $L(v)$ is the restriction of the lposet $L(v)$ to a subset $\{v\} \times \{\widehat{f}\}^{\downarrow}$ of $X_{L(v)}$, where $\{\widehat{f}\}^{\downarrow}$ is the set of bounded initial

segments of the modified version $\widehat{f}$ of some $f : [0, \infty) \to [0, \infty)$. One-element subsets of E are its cross-sections. The ordered set of elements of E represents the intrinsic time of E whatever is its nature (discrete, continuous, or partially discrete and partially continuous). In the case of a continuous function f it reduces to an interval.

Note that the replacement of f by its modified version $\widehat{f}$ is necessary since for f with a constant value on an interval the condition (3) of Definition 3.3 is not satisfied for the corresponding lposet.

The abstract execution fragments corresponding to a concrete execution fragment Q of $L(v)$ and to a concrete execution fragment R of $L(v')$ are illustrated in Figure 3.1. Thick lines illustrate continuous state changes. ♯

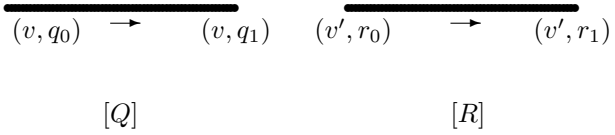


Figure 3.1: $[Q]$, $[R]$

3.9. Example. The behaviour of two tanks v and v' such that there is no pouring of liquid from v to v' or from v' to v can be represented by the execution structure $L(v, v')$ in $\mathbf{U}'$ such that
$X_{L(v,v')} = \{v\} \times F' \cup \{v'\} \times F'$ with the set F' of real-valued functions, each function being the modified version $\widehat{f}$ of a function f defined on an interval $[0, t]$ and describing how the quantity of liquid in a tank depends on the non-observable real time,
$\leq_{L(v,v')}$ is the partial order on $X_{L(v,v')}$ such that $(v, \widehat{f}) \leq_{L(v,v')} (v, \widehat{f'})$ for $\widehat{f}$ being the initial segment of $\widehat{f'}$ and $(v', \widehat{f}) \leq_{L(v,v')} (v', \widehat{f})$ for $\widehat{f}$ being an initial segment of $\widehat{f'}$,
$l_{L(v,v')}(v, \widehat{f}) = (v, f(t))$ and $l_{L(v,v')}(v', \widehat{f}) = (v', f(t))$ for f with the domain $[0, t]$.

Every concrete execution E of $L(v, v')$ is the restriction of $L(v, v')$ to a subset $\{v\} \times \{\widehat{f}\}^{\downarrow} \cup \{v'\} \times \{\widehat{f'}\}^{\downarrow}$ of $X_{L(v,v')}$, where $\{\widehat{f}\}^{\downarrow}$ is the set of bounded initial segments of the modified version $\widehat{f}$ of some $f : [0, \infty) \to [0, \infty)$ and $\{\widehat{f'}\}^{\downarrow}$ is the set of bounded initial segments of the modified version $\widehat{f'}$ of some $f' : [0, \infty) \to [0, \infty)$.

The abstract execution fragment $[T]$ which corresponds to a concrete execution fragment T of $L(v, v')$ is illustrated in Figure 3.2. ♯

3.10. Example. Pouring of the surplus of the liquid in a tank v over a given quantity $m > 0$ to a tank v' can be represented by the execution structure in $\mathbf{U}'$ that can be defined as $L'(v, v') = (X_{L'(v,v')}, \leq_{L'(v,v')}, l_{L'(v,v')})$ where
$X_{L'(v,v')} = \{x\} \times [m, \infty) \cup \{y\} \times [0, \infty) \cup \{x'\} \times \{m\} \cup \{y'\} \times [0, \infty)$ with different x, x', y, y',
$\leq_{L'(v,v')}$ is the least partial order on $X_{L'(v,v')}$ such that
$(x, q_1),\ (x', r_1)\ \leq_{L'(v,v')}\ (y, m),\ (y', r_2)$ for $m \geq 0$ and q_1, r_1, r_2 such that $q_1 - m = r_2 - r_1 \geq 0$, and
$l_{L'(v,v')}(x, q) = l_{L'(v,v')}(y, q) = (v, q),\ l_{L'(v,v')}(x', r) = l_{L'(v,v')}(y', r) = (v', r)$.

Concrete executions of $L'(v, v')$ are its restrictions to the subsets $\{(x, q_1), (y, m), (x', r_1), (y', r_2)\}$ such that $m \geq 0$ and $q_1 - m = r_2 - r_1 \geq 0$. The abstract execution fragment $[S]$ which corresponds to a concrete execution fragment S of $L'(v, v')$ is illustrated in Figure 3.2. ♯

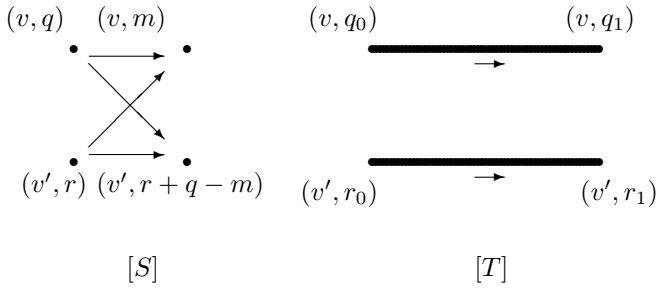


Figure 3.2: $[S]$, $[T]$

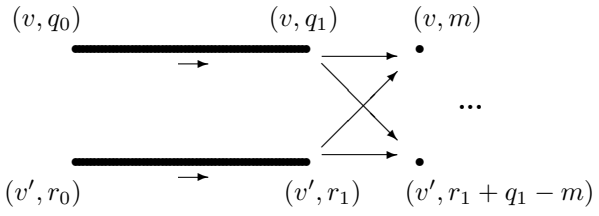
3.11. Example. The behaviour of the system P of tanks v and v' such that from time to time a quantity of liquid is poured from v to v' can be represented by the execution structure $L(P)$ in $\mathbf{U}'$ that can be defined as follows.

We can assume without a loss of generality that every concrete execution of $L(P)$ is isomorphic to an lposet $k = (X_k, \leq_k, l_k)$ that consists

of a sequence $T_1, S_1, T_2, S_2, ...$ of segments $T_1, T_2, ...$ of concrete executions of $L(v, v')$ and segments $S_1, S_2, ...$ of concrete executions of $L'(v, v')$ such that, for every $i \in \{1, 2, ...\}$, x is a maximal element of T_i iff it is a minimal element of S_i and y is a maximal element of S_i iff it is a minimal element of T_{i+1}. Let K denote the set of all such lposets k.

We define $L(P)$ as the lposet $(X_{L(P)}, \leq_{L(P)}, l_{L(P)})$ where
$X_{L(P)}$ is the set of labelled subsets of members k of K that can be represented as $\{x\}_k^- = \{y \in X_k : y \leq_k x\}$,
$\leq_{L(P)}$ is the partial order such that
$\{x\}_k^- \leq_{L(P)} \{x'\}_{k'}^-$ iff $x, x' \in X_{k'}$ and $\{x\}_k^- = \{x\}_{k'}^- \subseteq \{x'\}_k^-$,
$l_{L(P)}(\{x\}_k^-) = l_k(x)$.

The abstract execution fragment corresponding to a concrete execution N of $L(P)$ is illustrated in Figure 3.3. ♯



$[N]$

Figure 3.3: $[N]$

From Definition 3.5 and from the properties of cross-sections of lposets we obtain the following properties of execution structures.

3.12. Proposition. The relation *prefix*, where E' *prefix* E'' iff E' is an independent component of an initial segment of E'', is a partial order on the set of initial concrete execution fragments of L, called the *prefix order*. If initial concrete execution fragments E' and E'' of L are in this relation then we say that E' is a *prefix* of E''. ♯

3.13. Proposition. The set of initial concrete execution fragments of an execution structure L in **U** with the prefix order is a directed complete partially ordered set (a DCPO), written as $initial(L)$. ♯

3.14. Proposition. Let $(E_1, E_2) \mapsto E_1; E_2$ be the partial operation defined for concrete execution fragments E_1 and E_2 of L such that $end(E_1) = origin(E_2)$, where $E_1; E_2$ is defined as the unique concrete execution fragment E of L such that $head(E, c) = E_1$ and $tail(E, c) = E_2$ for a cross-section c of E. Let $(E_1, E_2) \mapsto E_1 \parallel E_2$ be the partial operation defined for concrete execution fragments E_1 and E_2 of L such that the sets of objects having occurrences in E_1 and in E_2 are disjoint, where $E_1 \parallel E_2$ is defined as the unique concrete execution fragment E of L such that $first(E, s) = E_1$ and $second(E, s) = E_2$ for a parallel distribution s of E. The set $cexe(L)$ of all concrete execution fragments of L with these operations is a partial algebra $\mathbf{cexe}(L) = (cexe(L), ;, \parallel)$. ♯

4 Equivalence of execution structures

Different execution structures may be equivalent in the sense that they may be regarded as representing the same action. For execution structures we have a notion of history preserving equivalence which is similar to the strong history preserving equivalence for event structures (cf. [9]).

4.1. Definition. A *history preserving bisimulation* between execution structures $L = (X, \leq, l)$ and $L' = (X', \leq', l')$ in a universe $\mathbf{U} = (V, W, ob)$ of objects is a set R of triples (A, A', f) which consist of a closed initial concrete execution segment A of L, of a closed initial concrete execution segment A' of L', and of an isomorphism $f : A \to A'$, where

(1) if $(A, A', f) \in R$ and A is a prefix of a closed initial concrete execution segment B of L then there exist a closed initial concrete execution segment B' of L' such that A' is a prefix of B' and an isomorphism $g : B \to B'$ such that $(B, B', g) \in R$ and g is an extention of f,

(2) if $(A, A', f) \in R$ and A' is a prefix of a closed initial concrete execution segment B' of L' then there exist a closed initial concrete execution segment B of L such that A is a prefix of B and an isomorphism $g : B \to B'$ such that $(B, B', g) \in R$ and g is an extention of f,

(3) if $(A, A', f) \in R$, C is a prefix of A, C' is the image of C under f, and g is the restriction of f to C, then $(C, C', g) \in R$.

If such a bisimulation exists then we say that L and L' are *history preserving equivalent* and write $L \approx L'$. ♯

4.2. Example. Let L' be the restriction of the execution structure $L(v) = (X_{L(v)}, \leq_{L(v)}, l_{L(v)})$ from Example 3.8 to a subset $z^+ = \{x \in X_{L(v)} : z \leq_{L(v)} x\}$ with the obvious isomorphism $i : L(v) \to L'$. The set R of triples (A, A', f), where A is a closed initial concrete execution segment of $L(v)$, A' is the image of A under i, and f is the unique isomorphism from A to A', is a history preserving bisimulation. Consequently, $L(v)$ and L' are equivalent with respect to R, $L(v) \approx L'$. ♯

4.3. Example. Suppose that $L' = (X', \leq', l')$ with $X' = X \times \{1\} \cup X \times \{2\}$ consists of two disjoint copies of an execution structure $L = (X, \leq, l)$, one copy with the underlying set $X \times \{1\}$ and the other copy with the underlying set $X \times \{2\}$. The set R of triples (A, A', f), where A is a closed initial concrete execution segment of L and A' is the image of A under the isomorphism $f_1 : x \mapsto (x, 1)$ and $f = f_1$, or A' is the image of A under the isomorphism $f_2 : x \mapsto (x, 2)$ and $f = f_2$, is a history preserving bisimulation between L and L'. Consequently, $L \approx L'$. ♯

The history preserving equivalence of execution structures may be highly complex since execution structures may branch without restrictions and unfold in a continuous way. Nevertheless, it is still very interesting since it leads to a very simple representation of equivalence classes and, consequently, to a very simple model of action. More precisely, its equivalence classes can be regarded as some sets of abstract execution fragments, and identity of such classes can be regarded as the identity of sets. This follows from a theorem that will be formulated in section 6. In order to formulate this theorem and derive its consequences it is convenient to consider execution of actions and their fragments as independent entities.

5 Algebras of execution fragments

Executions of actions and their fragments can be regarded as independent entities which can be considered without specifying what actions are executed. This will allow us to characterize the sets of initial abstract execution fragments of actions as specific subsets of certain partial algebras of abstract execution fragments. In order to define the respective partial algebras we describe some properties of abstract execution fragments and define natural partial operations on abstract execution fragments. Then we show that these operations can be used to define a partial order in every algebra of abstract execution fragments.

Let $\mathbf{U} = (V, W, ob)$ be a universe of objects and let $E = (X, \leq, l)$ be a concrete execution fragment in $\mathbf{U}$.

The following proposition is a direct consequence of definitions.

5.1. Proposition. Every fragment of E is an concrete execution fragment. ♯

In particular, for every cross-section c of E, the lposets $head(E, c)$ and $tail(E, c)$ are concrete execution fragments, and for every parallel decomposition $s = (s^F, s^S)$ of E, the lposets $first(E, s)$ and $second(E, s)$ are concrete execution fragments.

The following proposition reflects an important property of concrete execution fragments.

5.2. Proposition. For every cross-section c of a concrete execution fragment E, every isomorphism between closed initial segments of $tail(E, c)$ (resp.: between closed final segments of $head(E, c)$) is an identity. ♯

5.3. Corollary. For every closed segment Q of a concrete execution fragment E, every automorphism of Q is an identity. ♯

5.4. Corollary. For every closed concrete execution fragment E' there exists at most one isomorphism from E' to an initial segment of a concrete execution fragment E. ♯

5.5. Corollary. If a concrete execution fragment E is closed then for every closed concrete execution fragment E' there may be at most one isomorphism from E to E'. ♯

For every isomorphic concrete execution fragments E and E' we have $objects(E) = objects(E')$, where $objects(E)$ and $objects(E')$ denote the sets of objects having occurrences of instances in E and in E', respectively. Consequently, for the abstract execution fragment $[E]$ that corresponds to an execution fragment E we can define $objects([E]) = objects(E)$.

Collecting execution fragmentss into isomorphism classes, i.e. making abstract execution fragments, is convenient because it allows us to define some natural operations on the latter.

Let $EXE(\mathbf{U})$ denote the set of abstract execution fragments in $\mathbf{U}$. In the set $EXE(\mathbf{U})$ there exists the abstract execution fragment with the empty underlying set of its instance, called the *empty abstract execution fragment*, and written as 0. For each abstract execution fragment α from $EXE(\mathbf{U})$ with an instance $E \in \alpha$ and its cross-section $origin(E)$ there exists the unique abstract execution fragment $[origin(E)]$, called the *initial state* or the *source* of α and written as $dom(\alpha)$. For each closed abstract execution fragment α from $EXE(\mathbf{U})$ with an instance $E \in \alpha$ and its cross-section $end(E)$ there exists the unique abstract execution fragment $[end(E)]$, called the *final state* or the *target* of α and written as $cod(\alpha)$.

The abstract execution fragments belonging to $EXE(\mathbf{U})$ can be combined with the aid of two partial operations: a sequential composition and a parallel composition.

5.6. Definition. An abstract execution fragment α is said to *consist* of an abstract execution fragment α_1 *followed* by an abstract execution fragment α_2 iff an instance E of α has a cross-section c such that $head(E, c)$ is an instance of α_1 and $tail(E, c)$ is an instance of α_2. ♯

5.7. Proposition. For every two abstract execution fragments α_1 and α_2 such that $cod(\alpha_1)$ is defined and $cod(\alpha_1) = dom(\alpha_2)$ there exists a unique abstract execution fragment, written as $\alpha_1; \alpha_2$, or as $\alpha_1\alpha_2$, that consists of α_1 followed by α_2. ♯

5.8. Definition. The operation $(\alpha_1, \alpha_2) \mapsto \alpha_1\alpha_2$ is called the *sequential composition* of abstract execution fragments. ♯

Each abstract execution fragment which is a source or a target of an abstract execution fragment is an identity, i.e. an abstract execution fragment ι such that $\iota\phi = \phi$ whenever $\iota\phi$ is defined and $\psi\iota = \psi$ whenever $\psi\iota$ is defined. Moreover, $dom(\alpha)$ is the unique identity ι such that $\iota\alpha$ is defined, and if $cod(\alpha)$ is defined then it is the unique identity κ such that $\alpha\kappa$ is defined. Consequently, $\alpha \mapsto dom(\alpha)$ and $\alpha \mapsto cod(\alpha)$ are definable partial operations on abstract execution fragments. Identities are closed abstract execution fragments with causal dependency relations reducing to identity relations. They are called *states*, or *identities*, and we can identify them with the sets of occurring instances of objects.

5.9. Definition. An abstract execution fragment α is said to *consist* of two *parallel* abstract execution fragments α_1 and α_2 iff an instance E of α has a parallel decomposition s such that $\mathit{first}(E, s)$ is an instance of α_1 and $\mathit{second}(E, s)$ is an instance of α_2. ♯

5.10. Proposition. For every two abstract execution fragments α_1 and α_2 such that $\mathit{objects}(\alpha_1) \cap \mathit{objects}(\alpha_2) = \emptyset$ there exists an abstract execution fragment α with an instance E that has a parallel decomposition s such that $\mathit{first}(E, s)$ is an instance of α_1 and $\mathit{second}(E, s)$ is an instance of α_2. If such an abstract execution fragment α exists then it is unique, we write it as $\alpha_1 \parallel \alpha_2$, and we say that the abstract execution fragments α_1 and α_2 are *parallel*. ♯

For a proof it suffices to take $E_1 = (X_1, \leq_1, l_1) \in \alpha_1$ and $E_2 = (X_2, \leq_2, l_2) \in \alpha_2$ with $X_1 \cap X_2 = \emptyset$, and to provide $X_1 \cup X_2$ with the least common extensions of the causal dependency relations and labellings of E_1 and E_2.

5.11. Definition. The operation $(\alpha_1, \alpha_2) \mapsto \alpha_1 \parallel \alpha_2$ is called the *parallel composition* of abstract execution fragments. ♯

The operations on executions allow one to represent complex abstract execution fragments in terms of their components.

5.12. Example. In the case of abstract execution fragments in examples 3.8 - 3.11 we can represent $[T]$ as $[Q] \parallel [R]$, and an initial segment $[N_i]$ of $[N]$ with an instance consisting of $T_1, S_1, ..., T_i, S_i$ as $[T_1][S_1]...[T_i][S_i]$. ♯

The operations of composing abstract execution fragments allow one to turn the set $\mathit{EXE}(\mathbf{U})$ into a partial algebra.

5.13. Definition. The partial algebra $\mathbf{EXE}(\mathbf{U}) = (\mathit{EXE}(\mathbf{U}), ;, \parallel)$ is called the *algebra of execution fragments* in $\mathbf{U}$. ♯

The restriction of the algebra of execution fragments to the subset of closed abstract execution fragments is an arrows-only category (cf. [13]).

The operations of the algebra of execution fragments can be used to define in this algebra a partial order.

5.14. Proposition. The relation *pref*, where α *pref* β iff $\beta = (\alpha \parallel \gamma)\delta$ for some γ and δ, is a partial order on $EXE(\mathbf{U})$. If α and β are such that α *pref* β then we say that α is a *prefix* of β. ♯

5.15. Proposition. The extention $\sqsubseteq$ of the relation *pref*, where $\alpha \sqsubseteq \beta$ iff every prefix of α is a prefix of β, is a partial order on $EXE(\mathbf{U})$. The poset $(EXE(\mathbf{U}), \sqsubseteq)$ is a DCPO. Every element of $EXE(\mathbf{U})$ is the least upper bound of the directed set of its prefixes. ♯

5.16. Definition. The relation $\sqsubseteq$ on $EXE(\mathbf{U})$ is called the *prefix order*. The least upper bound of a directed subset D of the partially ordered set $(EXE(\mathbf{U}), \sqsubseteq)$ is called the *limit* of D. ♯

Note that the least upper bounds of directed subsets of the poset $(EXE(\mathbf{U}), \sqsubseteq)$ are limits of the corresponding filters in $EXE(\mathbf{U})$ with the Scott topology induced by the partial order $\sqsubseteq$.

6 Reduced execution structures

Let $\mathbf{U} = (V, W, ob)$ be a universe of objects.

The history preserving equivalence of execution structures is particularly interesting due to a very simple representation of its equivalence classes. More precisely, its equivalence classes can be regarded as some sets of abstract execution fragments, and identity of such classes can be regarded as the identity of the corresponding sets. This follows from the following theorem.

6.1. Theorem. Two execution structures L and L' in $\mathbf{U}$ are history preserving equivalent if and only if they have the same set of closed initial abstract execution fragments. ♯

For a proof it suffices to take into account Corollary 5.4 and Corollary 5.5 and consider the set R of triples (A, A', f) which consist of a closed initial abstract execution segment A of L, of a closed initial abstract execution segment A' of L', and of the unique isomorphism $f : A \to A'$, where A and A' are isomorphic and f is the unique isomorphism from A to A'.

6.2. Example. In order to see that the execution structures $L(v)$ and L' in Example 4.2 are equivalent with respect to the history preserving equivalence it suffices to notice that they have the same set of closed abstract execution fragments. ♯

The fact that execution structures in a universe of objects are history preserving equivalent if they have the same set of closed initial abstract execution fragments implies that each equivalence class of execution structures is determined uniquely by the set of closed initial abstract execution fragments of its members. Now we are going to show that each such a set determines a specific partially ordered set (a poset) of abstract execution fragments, a poset with an algebraic structure, called a reduced execution structure, and that every such a poset corresponds to an usual execution structure. To this end we use algebras of execution fragments and their prefix order and define reduced execution structures as specific subsets of such algebras. The posets thus obtained inherit some algebraic structure from the algebras in which they are defined, and there is a natural concept of a morphism from one such poset to another.

The definition of an execution structure implies the following property of the set of its abstract execution fragments.

6.3. Theorem. The set of initial abstract execution fragments of an execution structure in $\mathbf{U}$ is a subset B of the partial algebra $\mathbf{EXE}(\mathbf{U})$ of abstract execution fragments in $\mathbf{U}$ such that:

(1) B is downwards-closed with respect to $\sqsubseteq$,

(2) if α and β are initial segments of abstract execution fragments in $\mathbf{U}$ that are maximal elements of B then $\alpha(\gamma \parallel s) \in B$ iff $\beta(\gamma \parallel t) \in B$ for every γ such that $dom(\gamma) \parallel s = cod(\alpha)$ and $dom(\gamma) \parallel t = cod(\beta)$,

(3) if the least upper bound $\bigsqcup D$ of a subset D of B exists then it belongs to B. ♯

Due to Theorem 6.1 it is possible to represent actions by considering only their abstract executions. More precisely, every action considered up to the history preserving equivalence can be represented by a subset of the algebra of abstract execution fragments that can be defined as follows.

6.4. Definition. A *reduced execution structure* in $\mathbf{U}$ is a subset B of the partial algebra $\mathbf{EXE}(\mathbf{U})$ of abstract execution fragments in $\mathbf{U}$ that satisfies the conditions (1), (2), and (3) of Theorem 6.3. The abstract

execution fragments in **U** that are maximal elements of B are said to be abstract executions of B. By $seg(B)$ we denote the set of segments of abstract executions of B. By $\mathbf{seg}(B)$ we denote the restriction of the algebra $\mathbf{EXE}(\mathbf{U})$ to the set $seg(B)$. By $exe(B)$ we denote the set of elements of B. By $\mathbf{exe}(B)$ we denote the restriction of the algebra $\mathbf{EXE}(\mathbf{U})$ to the set $exe(B)$. ♯

Note that according to Proposition 5.15 every reduced execution structure is a DCPO. Note also that the partial algebra $\mathbf{EXE}(\mathbf{U})$ could be replaced by some of its subalgebras provided that the requirement of downwards-closedness of B with respect to $\sqsubseteq$ would be reduced to those abstract execution fragments which belong to the respective subalgebra.

6.5. Definition. A *morphism* from a reduced execution strucure B in **U** to a reduced execution structure B' in $\mathbf{U}'$
is a homomorphism $h : \mathbf{exe}(B) \to \mathbf{exe}(B')$. ♯

Reduced execution structures play a role are similar to that of languages in the theory of automata and in the theory of Petri nets. However, they consist of pomsets rather than of strings, and their elements are combined with the aid of operations different from concatenation.

By considering instances of abstract execution fragments of a reduced execution structure and by making a construction similar to those in examples 3.6 - 3.11 we can convert such structure into an execution structure. From this observation and from Propositions 3.12 - 3.14 and Proposition 5.15 we obtain the following property.

6.6. Theorem. A subset B the partial algebra $\mathbf{EXE}(\mathbf{U})$ of abstract execution fragments in **U** is a reduced execution structure in **U** iff it is the image of the set of initial execution fragments of an execution structure L in **U** under the correspondence $E \mapsto [E]$. ♯

Algebraic properties of $\mathbf{seg}(B)$ are related to those of $\mathbf{EXE}(\mathbf{U})$ due to the following theorem.

6.7. Theorem. For every reduced execution structure B in **U** the restriction $\mathbf{seg}(B)$ of $\mathbf{EXE}(\mathbf{U})$ to $seg(B)$ is a subalgebra of $\mathbf{EXE}(\mathbf{U})$. ♯

This result cannot be extended on the set $\mathbf{exe}(B)$. The algebraic properties of this set are as follows.

6.8. Theorem. For every reduced execution structure B in $\mathbf{U}$ the set $\mathbf{exe}(B)$ is closed under the sequential composition. The resut $\alpha \parallel \beta$ of the parallel composition of $\alpha \in \mathbf{exe}(B)$ and $\beta \in \mathbf{exe}(B)$ belongs to $\mathbf{exe}(B)$ iff $dom(\alpha) \parallel dom(\beta)$ is defined and belongs to $\mathbf{exe}(B)$. ♯

Taking into account Theorem 6.8 and the results of section 5 we obtain the following result.

6.9. Theorem. Every reduced execution structure B in $\mathbf{U}$ is a set of abstract execution fragments which can be obtained by combining abstract execution fragments from the set $exe(B)$ with the aid of compositions and construction of limits. ♯

This theorem has some consequences for applications of the model. Namely, it suggests how to construct an algorithm for symbolic generation of a reduced execution structure from a finite set of given initial states and from a finite set of given abstract execution fragments in a finite universe of objects. Such reduced execution structure can be generated starting from the given initial states and applyiung to each state which can be reached abstract execution fragments of the considered action in a way similar to that of generating unfoldings of Petri nets (cf. [5]). In some cases it can be used to investigate states which can be reached.

6.10. Example. Consider a tank v and a tank v' as in Example 3.2 and abstract execution fragments as in examples 3.8 - 3.11. By combining the abstract execution fragments as $[T]$ in Example 3.9 with the aid of sequential composition and construction of limits, we obtain a set A_1 of abstract execution fragments in the universe $\mathbf{U}'$. The set B_1 of abstract execution fragments from A_1 and their prefixes is a reduced execution structure in $\mathbf{U}'$. It represents an action that consists of independent actions of the tank v and the tank v'.

By combining the abstract execution fragments as $[T]$ in Example 3.9 and $[S]$ in Example 3.10 with the aid of sequential composition such that every two segments corresponding to components of type $[S]$ are separated by a segment corresponding to a component of type $[T]$, and by construction of limits, we obtain a set A_2 of abstract execution fragments in the universe

$\mathbf{U}'$. The set B_2 of abstract execution fragments from A_2 and their prefixes is a reduced execution structure in $\mathbf{U}'$. It represents an action that consists of actions of the tank v and the tank v' that are mainly independent, but from time to time are interrupted by the joint action of pouring of an amount of liquid from the tank v to the tank v'.

Each of the reduced execution structures B_1 and B_2 is a DCPO. ♯

7 Concluding remarks

We have described a model of action that is based on intrinsic action properties and is universal in the sense that it allows to represent in the same way discrete, continuous and hybrid actions. The model is derived from execution structures that are similar to labelled event structures, but are not restricted to discrete actions only, and reflect in a more subtle way the possible action executions and their components. In particular, the structures representing closed execution components admit only trivial automorphisms and unique isomorphisms. This leads to a simple characterization of history preserving equivalence of structures representing actions and to a simple characterization of its equivalence classes. More precisely, the equivalence classes of history preserving equivalence can be identified with reduced execution structures and there exists a bijective correspondence between equivalence classes of concrete execution structures and reduced execution structures. This allows us to represent actions by reduced execution structures rather than by concrete execution structures.

The representation of actions by reduced execution structures simplifies their studies.

As reduced execution structures are sets of abstract execution fragments, they are partially ordered by inclusion. Consequently, a simulation of an action by another action can be defined as the usual inclusion of one reduced execution structure in another reduced execution structure. In particular, a bisimulation reduces to identity.

When partially ordered by inclusion the set of reduced execution structures representing actions in a universe of objects is a complete lattice.

Structures representing abstract execution fragments can be combined with the aid of natural partial operations. This leads to partial algebras. In particular, the set of execution fragments of each reduced execution structure can be regarded as a partial algebra of abstract execution fragments in a universe of objects, and a homomorphism from such a partial algebra to another such a partial algebra of abstract execution fragments can be used

to represent a refinement of the represented action.

Every set of abstract execution fragments of a reduced execution structure with a partial prefix order is a directed complete partial order (a DCPO). Consequently, it can be provided with the Scott topology and the ideas described in [1], [12], and [21], can be applied to provide it with a probability measure.

The representation of actions by reduced execution structures leads to simple operations on actions and can be used to develop a calculus of actions playing a role similar to that of CCS (cf. [14], [15], and [22]).

When partially ordered by inclusion the set of reduced execution structures representing actions in a universe of objects is a complete lattice. Consequently, we can speak of the *greatest lower bound* and the *least upper bound* of a family of reduced execution structures and the represented actions. The greatest lower bound of a nonempty family of actions is the action represented by the intersection of the reduced execution structures representing the members of the family. The least upper bound of a nonempty family of actions is the action represented by the reduced execution structure that consists of the abstract execution fragments of the members of the family and of the abstract execution fragments whose existence follows from the requirements of the definition of reduced execution structures.

These operations can be used to define compound actions as results of combining their component actions.

In order to illustrate this consider tanks as in Example 3.2 and actions represented by reduced execution structures described in Example 6.10.

According to Example 3.8, the behaviour of a tank v can be represented by the reduced execution structure $B(v)$ that consists of the possible abstract execution fragments such as $[Q]$ in Figure 3.1.

According to Example 6.10, the action that consists of independent actions of the tank v and the tank v' can be represented by the reduced execution structure B_1. On the other hand, this action can be defined as the least upper bound of actions represented by $B(v)$ and $B(v')$ because B_1 is the least upper bound of $B(v)$ and $B(v')$.

According to Example 6.10, the action that consists of actions of the tank v and the tank v' that are mainly independent but from time to time are interrupted by the joint action of pouring an amount of liquid from the tank v to the tank v' can be represented by the reduced execution structure B_2. On the other hand, this action can be defined as the least upper bound of actions represented by B_1 and the least reduced execution structure containing the abstract execution fragments correponding to the possible concrete execution fragments as S in Example 3.10.

So, the proposed model of action is compositional in the sense that it allows to define complex actions as results of applying natural operations to models of simple component actions. In particular, it can be used to formulate finite definitions of actions whose components are infinite but can be described in a finite way in a logic.

The lattice theoretical operations on actions are not the only operations we can consider. In general, operations on actions can be defined like operations on data flows or operations of calculi like CCS. Such operations should be continuous in the sense that they should preserve the least upper bounds of chains of reduced execution structures. Then the list of operations on actions can be extended with the aid of fixed point equations, and a powerful calculus of actions can be developed.

References

[1] M. Alvarez-Manilla, A. Edalat, N. Saheb-Djahromi, *An Extension Result for Continuous Valuations*, J. London math. Soc. (2) 61 (2000) 629-640

[2] R. David, *Modeling of Dynamic Systems by Petri Nets*, Proc. of ECC 91, European Control Conference, Grenoble, France, July 2-5, 1991, 136-147

[3] M. Droste, R. M. Shortt, *Continuous Petri Nets and Transition Systems*, H. Ehrig et al. (Eds.): Unifying Petri Nets, Springer LNCS 2128, 2001, 457-484

[4] J. Engelfriet, *Branching Processes of Petri Nets*, Acta Informatica 28 (1991) 575-591

[5] J. Esparza, *Model Checking Using Net Unfoldings*, Science of Computer Programming 23 (1994), 151-195

[6] R. van Glabbeek, U. Goltz, *Equivalence Notions for Concurrent Systems and Refinement of Actions*, Acta Informatica, Vol. 37, 2001, 229-327

[7] R. van Glabbeek, G. D. Plotkin, *Configuration Structures*, Proceedings of LICS'95, Kozen, D., (Ed.), IEEE Computer Society Press (1995) 199-209

[8] T. Henzinger, *The Theory of Hybrid Automata*, Proc. of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS 96), 278-292

[9] A. Joyal, M. Nielsen, G. Winskel, *Bisimulation from Open Maps*, Proceedings of LICS 93, 1993, 418-427

[10] R. E. Kalman, P. L. Falb, M. A. Arbib, *Topics in Mathematical System Theory*, Mc Graw-Hill Book Company, New York, San Francisco, St. Louis, Toronto, London, Sydney, 1969

[11] Padmanabhan Krishnan, *Hybrid Event Structures*, Proc. of Computing: The Australasian Theory Symposium, Melbourne, Australia, January 29-January 30 1996

[12] N. Lynch, R. Segala, F. Vaandrager, *Observing Branching Structure Through Probabilistic Contexts*, Siam Journal on Computing 37 (4), 977-1013, September 2007

[13] S. Mac Lane, *Categories for the Working Mathematician*, Springer-Verlag New York Heidelberg Berlin 1971

[14] R. Milner, *Synthesis of Communicating Behaviour*, Proc. of MFCS'78, J. Winkowski (Ed.), Springer LNCS 64 (1978) 71-83

[15] R. Milner, *A Calculus of Communicating Systems*, Springer LNCS 92 (1980)

[16] S. Mitra, N. Lynch, *Trace-based Semantics for Probabilistic Timed I/O Automata*, Hybrid Systems: Computation and Control (HSCC 2007), Pisa, Italy, April 3-5, 2007, Springer LNCS 4416, Full version: http://theory.lcs.mit.edu/∼mitras/research/PTIOA-066-full.pdf (1980)

[17] Anil Nerode, Wolf Kohn, *Models for Hybrid Systems: Automata, Topologies, Controllability, Observability*, Springer LNCS 736, 1993, 317-357

[18] M. Nielsen, G. D. Plotkin, G. Winskel, *Petri Nets, Event Structures and Domains, Part I*, Theoretical Computer Science, Vol. 13, No. 1 (1981), 85-108

[19] C. A. Petri, *Introduction to General Net Theory*, in W. Brauer (Ed.): Net Theory and Applications, Springer LNCS 84 (1980) 1-19

[20] G. Rozenberg, P. S. Thiagarajan, *Petri Nets: Basic Notions, Structure, Behaviour*, in J. W. de Bakker, W. P. de Roever and G. Rozenberg (Eds.), Current Trends in Concurrency, Springer LNCS 224 (1986) 585-668

[21] D. Varacca, H. Völzer, G. Winskel, *Probabilistic Event Structures and Domains*, in P. Gardner and N. Yoshida (eds.), CONCUR 2004, Springer LNCS 3170 (2004), 497-511

[22] J. Winkowski, A. Maggiolo-Schettini, *An Algebra of Processes*, Journal of Comp. and System Sciences, Vol. 35, No. 2, October 1987, 206-228

[23] J. Winkowski, *An Algebraic Framework for Defining Behaviours of Concurrent Systems. Part 1: The Constructive Presentation*, Fundamenta Informaticae 97 (2009), 235-273

[24] G. Winskel, *Event Structure Semantics for CCS and Related Languages*, in M. Nielsen and E. M. Schmidt (Eds.): Springer LNCS 140 (1982), 561-567

[25] G. Winskel, M. Nielsen, *Models for Concurrency*, in S. Abramsky, Dov M. Gabbay and T. S. E. Maibaum (Eds.): Handbook of Logic in Computer Science 4 (1995), 1-148

Appendix

Proof of Proposition 2.1.
The set $Z' \wedge Z''$ is an antichain since otherwise there would be $x < y$ for some x and y in this set. If $x \in Z'$ then there would be $y \in Z''$ and there would exist $z' \in Z'$ such that $y \leq z'$. However, this is impossible since Z' is an antichain. Similarly for $x \in Z''$.

The set $Z' \wedge Z''$ is a maximal antichain since otherwise there would exist x that would be incomparable with all the elements of this set. Consequently, there would not exist $z' \in Z'$ and $z'' \in Z''$ such that $z' \leq x \leq z''$, or $z'' \leq x \leq z'$, or $z', z'' \leq x$, and thus there would be $x \leq z'$ and $x \leq z''$ for some $z' \in Z'$ and $z'' \in Z''$ that are not in $Z' \wedge Z''$. Consequently, there would exist z, say in Z'', such that $x \leq z \leq z'$. Moreover, $z \in Z' \wedge Z''$ since otherwise there would be $t \in Z'$ such that $t \leq z \leq z'$, what is impossible.

In order to see that $Z' \wedge Z''$ is a cross-section we consider $x \leq y$ such that $x \leq t$ and $u \leq y$ for some $t \in Z' \wedge Z''$ and $u \in Z' \wedge Z''$, where $t \in Z'$ and $u \in Z''$. Without a loss of generality we can assume that $y \leq y'$ for some $y' \in Z'$ since otherwise we could replace y by an element of Z'. Consequently, there exists $z \in Z''$ such that $x \leq z \leq y$. On the other hand, $z \in Z' \wedge Z''$ since otherwise there would be $z' \in Z'$ such that $z' \leq z \leq y$, what is impossible. In a similar manner we can find $z \in Z' \wedge Z''$ for the other cases of t and u.

In order to see that $Z' \wedge Z''$ is the greatest lower bound of Z' and Z'' consider a cross-section Y which precedes Z' and Z'' and observe that $y \leq z' \in Z'$ and $y \leq z'' \in Z''$ with z' and z'' not in $Z' \wedge Z''$ and $y \in Y$ implies the existence of $t \in Z'$ such that $y \leq t \leq z'$ or $u \in Z''$ such that $y \leq u \leq z''$.

Similarly, $Z' \vee Z''$ is the least upper bound of Z' and Z''.

The last part of the proposition follows from the fact that the correspondence $Z \mapsto Z^-$ is an isomorphism from the lattice of cross-sections of P to a sublattice of the lattice of subsets of P. ♯

Proof of Proposition 5.2.
Let Q be the restriction of E to c^+ and let R and S be two initial segments of Q. Suppose that $f : R \to S$ is an isomorphism that it is not an identity. Then there exists an initial subsegment T of R such that the image of T under f, say T', is different from T. By (3) of definition 3.3 neither T' is a subsegment of T nor T is a subsegment of T'. Define T'' to be the least segment containing both T and T', and consider $f' : T \to T''$, where $f'(x) = f(x)$ for $x \leq f(x)$ and $f'(x) = x$ for $f(x) < x$. In order to derive a contradiction, and thus to prove that f is an identity, it suffices to verify, that f' is an isomorphism. It can be done as follows.

For injectivity suppose that $f'(x) = f'(y)$. If $x \leq f(x)$ and $y \leq f(y)$ then $f(x) = f'(x) = f'(y) = f(y)$ and thus $x = y$. If $f(x) < x$ and $f(y) < y$ then $x = f'(x) = f'(y) = y$. The case $x \leq f(x)$ and $f(y) < y$ is excluded by $f'(x) = f'(y)$ since $x \leq f(x) = f'(x) = f'(y) = y$ and, on the other hand, $f(y) < y = f(x)$ implies $y < x$. Similarly, the case $f(x) < x$ and $y \leq f(y)$ is excluded. Consequently, f' is injective.

For surjectivity suppose that y is in T''. If $y \leq f(y)$ then, by surjectivity of f and condition (1) of Definition 3.3, there exists $t \leq y$ such that $y = f(t)$ and thus $y = f(t) = f'(t)$ since $t \leq y = f(t)$. If $f(y) < y$ then $y = f'(y)$. Consequently, f' is surjective.

For monotonicity suppose that $x \leq y$. If $x \leq f(x)$ and $y \leq f(y)$ then $f'(x) = f(x) \leq f(y) = f'(y)$. If $f(x) < x$ and $f(y) < y$ then $f'(x) = x \leq$

$y = f'(y)$. If $x \leq f(x)$ and $f(y) < y$ then $f'(x) = f(x) \leq f(y) < y = f'(y)$. If $f(x) < x$ and $y \leq f(y)$ then $f'(x) = x \leq y \leq f(y) = f'(y)$. Consequently, f' is monotonic.

For monotonicity of the inverse suppose that $f'(x) < f'(y)$. If $x \leq f(x)$ and $y \leq f(y)$ then $f(x) = f'(x) < f'(y) = f(y)$ and thus $x < y$. If $f(x) < x$ and $f(y) < y$ then $x = f'(x) < f'(y) = y$. If $x \leq f(x)$ and $f(y) < y$ then $x \leq f(x) = f'(x) < f'(y) = y$. If $f(x) < x$ and $y \leq f(y)$ then $f(x) < x = f'(x) < f'(y) = f(y)$ and thus $x < y$. Consequently, the inverse of f' is monotonic.

A proof for final subsegments of E restricted to c^- is similar. ♯

Proof of Proposition 5.7.
Take $E_1 = (X_1, \leq_1, l_1) \in \alpha_1$ and $E_2 = (X_2, \leq_2, l_2) \in \alpha_2$ with $\mathit{end}(E_1) = \mathit{origin}(E_2)$ and with the underlying set of $\mathit{end}(E_1)$ and the underlying set of $\mathit{origin}(E_2)$ equal to $X_1 \cap X_2$, and provide $X = X_1 \cup X_2$ with the least common extensions of the causal dependency relations and labellings of E_1 and E_2.

Let E be the lposet thus obtained. It suffices to prove that E is an execution fragment and notice that $\mathit{head}(E, c) = E_1$ and $\mathit{tail}(E, c) = E_2$. In order to prove that E is an execution fragment it suffices to show that E does not contain a segment with isomorphic proper subsegment. To this end suppose the contrary. Suppose that $f : Q \to R$ is an isomorphism from a segment Q of E to a proper subsegment R of Q, where Q consists of a part Q_1 contained in E_1 and a part Q_2 contained in E_2. By applying twice the method described in the proof of Proposition 5.2 we can modify f to an isomorphism $f' : Q \to R$ such that the image of Q_1 under f', say R_1, is contained in Q_1, and the image of Q_2 under f', say R_2, is contained in Q_2. As R is a proper subsegment of Q, one of these images, say R_1, is a proper part of the respective Q_i. By taking the greatest lower bounds and the least upper bounds of appropriate cross-sections we can extend Q_1 and R_1 to segments Q_1' and R_1' of P_1 such that R_1' is a proper subsegment of Q_1' and there exists an isomorphism from Q_1' to R_1'. This is in a contradiction with the fact that E_1 is an execution fragment. Consequently, E is an execution fragment. ♯

Proof of Proposition 5.14.
For transitivity suppose that $\beta = (\alpha \parallel \gamma)\delta$ and $\beta' = (\beta \parallel \gamma')\delta'$. If $E_{\beta'}$ is an instance of β' then there exists c such that $\mathit{head}(E_{\beta'}, c)$ is an instance $E_{\beta \parallel \gamma'}$ of $\beta \parallel \gamma'$ and $\mathit{head}(\mathit{first}(E_{\beta \parallel \gamma'}, s), c_1)$ is an instance E_β of β for some s and a component c_1 of c. Moreover, there exists d such that $\mathit{head}(E_\beta, d)$ is an

instance $E_{\alpha\|\gamma}$ of $\alpha \parallel \gamma$ and $head(first(E_{\alpha\|\gamma}, t), d_1)$ is an instance of E_α for some t and a component d_1 of d. Consequently, $head(E_{\beta'}, c')$ is an instance of $\alpha \parallel \gamma \parallel \gamma'$ for c' consisting of d and of the complement of c_1 to c, and $\beta' = (\alpha \parallel \gamma \parallel \gamma')\delta''$ for $\delta'' = [tail(E_{\beta'}, c')]$. For antisymmetry suppose that $\beta = (\alpha \parallel \gamma)\delta$ and $\alpha = (\beta \parallel \gamma')\delta'$. As objects with instances occurring in α cannot occur in γ and objects with instances occurring in β cannot occur in γ', there must be $\gamma = \gamma' = 0$. Consequently, $\alpha = \alpha\delta\delta'$ and, by Corollary 5.5, δ and δ' must be identities. ♯

Proof of Proposition 5.15.
Given a directed subset D of the poset $(EXE(\mathbf{U}), \sqsubseteq)$, the pefixes of elements of D form a directed set D'. For every element of D' we choose a concrete instance, and we consider α and $\beta = (\alpha \parallel \gamma)\delta$ such that E is the chosen instance of α, E_1 is the chosen instance of β, E_2 is the chosen instance of $\alpha \parallel \gamma$ and $E_3 = head(E_1, c)$ is an instance of $\alpha \parallel \gamma$. Then there exists a unique isomorphism f from E_2 to E_3 since otherwise there would be another isomorphism g and the correspondence $f(x) \mapsto g(x)$ would be different from the identity isomorphism between two initial segments of E_1. On the other hand, f determines a unique isomorphism between E and $first(E_2, s)$ with a parallel decomposition s due to the fact that the first part of E_2 is determined uniquely by the set of objects which occur in it. Consequently, we can construct a direct system of instances of elements of D' such that the colimit of this system in the category **LPOSETS** is an instance of the least upper bound of D' and of D. The last part of the proposition is a simple consequence of the condition (2) of Definition 3.3. ♯

Proof of Proposition 6.9.
As no segments of executions of B can be composed in parallel, it suffices to prove that $\alpha\beta \in seg(B)$ whenever α and β are segments of executions of B. To this end consider an execution structure L such that B is the image of the set of initial execution segments of L under the correspondence $E \mapsto [E]$. Consider in L an execution E such that $head(tail(E, c), d)$ is an instance of β, and an execution E' such that $tail(head(E', c'), a)$ with c' isomorphic to c is an instance of α. According to condition (2) of Definition 3.5 there exists an execution E'' with $head(E'', c') = head(E', c')$ such that $tail(E'', c')$ is isomorphic to $tail(E, c)$. Consequently, there exists d' such that $head(tail(E'', c'), d')$ is an instance of β. On the other hand, $tail(head(E'', c'), a) = tail(head(E', c'), a)$ is an instance of α. Hence there exists d'' such that $tail(head(E'', d''), a)$ is an instance of $\alpha\beta$ and, consequently, $\alpha\beta \in seg(B)$. ♯

Pracę zgłosił Wojciech Penczek

Adres autora

Józef Winkowski
Instytut Podstaw Informatyki PAN
01-248 Warszawa, Jana Kazimierza 5,
e-mail: wink@ipipan.waw.pl

Klasyfikacja rzeczowa: F.1.1, F.1.2

Printed as manuscript
Na prawach rękopisu

Nakład 100 egzemplarzy. Oddano do druku w październiku 2016 r. Wydawnictwo IPI PAN. ISSN: 0138-0648