MODELLING TIMED BEHAVIOURS WITH THE AID OF EVENT AND CONFIGURATION STRUCTURES [1]

Józef Winkowski

Instytut Podstaw Informatyki PAN

01-237 Warszawa, ul. Ordona 21, Poland

[1]This work has been partially supported by the Polish State Committee for Scientific Research (Grant No. 2 2047 92 03). It has been published in the journal "Archiwum Informatyki Teoretycznej i Stosowanej" of the Polish Academy of Sciences, vol. 5, 1993, pp. 375-408.

Running head: MODELLING TIMED BEHAVIOURS

Abstract

A mathematical model of timed behaviours is offered and investigated. This model is presented in the form of event or configuration structures with an extra information about times and durations of events (with timing). It is shown that event and configuration structures thus enriched can be treated and composed as the usual event and configuration structures and that they can be considered up to a suitable equivalence. It is shown that the timing of event and configuration structures allows to define interesting characteristics of the modelled behaviours and that such characteristics for behaviours consisting of simpler components can be obtained by combining the characteristics of components.

Key words:

behaviour, timed behaviour, action, event, configuration, event structure, configuration structure, causality, branching, timing, compositionality, equivalence, eager configurations, stamped configurations, fast configurations.

1 Introduction

Executions of concurrent programs or communication protocols, activities of complex information processing systems as distributed data bases or operating systems, etc. are behaviours which develop by executing actions, where the development results indeterministically in a run from a set of possible runs. In order to precisely define and analyse such behaviours we have to model them by means of suitable mathematical structures. In a model of a behaviour we want to reflect how executions of actions (events) may follow each other (causality), how they may exclude each other (branching), and, possibly, how they occur in a global time (timing).

A popular way of modelling a behaviour is to consider only its causality and branching and to represent the behaviour by an event structure (a conflict event structure) of the form $E = (X, \leq, \sharp)$, where X is a set of events, $\leq$ is a causal order (or quasiorder) of events, and $\sharp$ is a conflict relation between events (cf. [6]). Simultaneous relations $x \leq y$ and $y \leq x$ in the case of causality given by a quasiorder represent a coincidence of events x and y. The causal quasiorder is usually assumed to be finitary in the sense that each event has only finitely many predecessors. The conflict relation is supposed to be irreflexive, symmetrical, disjoint with the causal quasiorder, and persistent in the sense that $x\sharp y$ and $y \leq z$ implies $x\sharp z$. A slightly more general model (a general event structure) can be obtained by replacing the conflict relation by a family F of sets of events (forbidden sets) such that Y belongs to F whenever the set of elements of Y and their predecessors, $\downarrow Y$, contains a member of F.

In the case of behaviours of our concern each event x represents a particular execution of an action α which may be written as $\mathbf{label}(x)$, and it may be identified with a pair (n, α), where n is a name of the execution ($n = \mathbf{name}(x)$). Consequently, we have to do with a labelled event structure, the labelling given by the correspondence $x \longmapsto \mathbf{label}(x)$.

An equivalent model of a behaviour can be obtained by considering the family of configurations of a representing event structure E, where a configuration is a finite conflict-free prefix of E or, more precisely, the set of events of such a prefix (cf. [9], [10], and [11]). Such a family, $\mathbf{conf}(E)$, is a member of an axiomatically defined class of structures

(prime configuration structures). Namely, it is closed w.r. to intersections of nonempty subfamilies and unions of subfamilies which have upper bounds in $\mathbf{conf}(E)$. Conversely, each family P of sets which satisfies these axioms is the set of configurations of an event structure $\mathbf{evs}(P) = (X, \leq, F)$, where $X = \bigcup P$, $x \leq y$ iff each $p \in P$ with $y \in p$ contains also x, and F is the family of subsets of X such that X cannot be covered by any subfamily $S \subseteq P$ which is closed w.r. to unions of finite subfamilies.

Sometimes it is convenient to work with a slightly more general concept of configuration structures which we obtain by replacing the condition of closedness w.r. to intersections of nonempty subfamilies of configurations by a weaker condition of closedness w.r. to intersections of nonempty bounded subfamilies (stability in the sense of [9]). In this case, for a configuration structure P, each $x \in \bigcup P$ may identify an event only locally (within a configuration) and instead of a global causal quasiorder we obtain a consistent family $(\leq_p: p \in P)$ of local quasiorders, where $x \leq_p y$ iff $x, y \in p$ and $y \in q$ implies $x \in q$ for all $q \subseteq p$ with $q \in P$. Moreover, P can be transformed into a prime configuration structure, $\mathbf{prime}(P)$, by splitting each $x \in \bigcup P$ into a family $(x_p : p \in P)$, where x_p is determined uniquely by $\{y \in p : y \leq_p x\}$, and by replacing each $p \in P$ by $\{x_p : x \in p\}$.

Event and configuration structures representing behaviours can be combined with the aid of operations similar to those of CCS (cf. [4], [9], and [1]). These operations are defined assuming that actions are qualified as internal or noninternal and that some of the noninternal actions are regarded to be complementary and can be combined into single actions with the aid of a special associative partial operation $(\alpha, \beta) \longmapsto \alpha \bullet \beta$ (cf. synchronization algebras in [9]). A parallel composition is crucial. For configuration structures P and Q it is defined as $P \parallel Q$ such that $r \in P \parallel Q$ iff r consists of some $p \in P$ and $q \in Q$ in the sense that it is obtained from a disjoint union of p and q by combining some pairs of events $x \in p$ and $y \in q$ into single events $z \in r$ with $\mathbf{label}(z) = \mathbf{label}(x) \bullet \mathbf{label}(y)$. For event structures a direct definition is more complicated (cf. [1]), but we can define $E_0 \parallel E_1$ as $\mathbf{evs}(\mathbf{prime}(\mathbf{conf}(E_0) \parallel \mathbf{conf}(E_1)))$.

The considered operations on configuration structures are continuous w.r. to the

chain-complete prefix order defined as $P \sqsubseteq Q$ iff $P \subseteq Q$ and $q \in P$ for all $q \in Q$ such that $q \subseteq \bigcup P$. This guarantees the existence of least fixed-points of mappings defined with the aid of such operations and allows one to use them for defining a broad class of continuous operations.

Concrete event and configuration structures may be overloaded with occasional details which have nothing to do with the represented behaviours. Hence they should be considered up to an equivalence. In order to facilitate constructing models of behaviours by combining models of behaviour components such an equivalence should be a congruence w.r. to the considered operations on configuration structures. There is a number of candidates for such an equivalence. Among them the so called history preserving equivalence (introduced in [7]) seems to be adequate. It can be defined with the aid of a concept of simulation, where a simulation of a configuration structure P in a configuration structure Q is a family $(\varrho_{pq} : p \in P, q \in Q)$ of sets of bijective, labels and order preserving correspondences between the respective configurations such that each $f \in \varrho_{pq}$ has an extension in $\varrho_{p'q'}$ with some q' such that $q \subseteq q'$ for all $p' \in P$ which contain p.

In this paper we consider behaviours with timing (timed behaviours as those in [5] but with possibly time-consuming actions as in [3]). We represent them by configuration structures with extra information about temporal features of events (timed configuration structures). Namely, with each event x of a representing configuration structure P we associate the enabling time of x, $\mathbf{entime}(x)$, the start time of x, $\mathbf{stime}(x)$, and the completion time of x, $\mathbf{cptime}(x)$, where $\mathbf{entime}(x) \leq \mathbf{stime}(x) \leq \mathbf{cptime}(x)$. The start times and the completion times of events are supposed to be consistent with the causal quasiorder in the sense that $\mathbf{cptime}(x) \leq \mathbf{stime}(y)$ whenever $x \leq_p y$ for some $p \in P$ and not $y \leq_p x$, and $\mathbf{stime}(x) = \mathbf{stime}(y)$ whenever x and y are coincident.

In order to remain in the framework of a unified theory of event and configuration structures, we associate the temporal features of events with actions rather than with events themselves. This can be achieved by regarding each event x as a particular execution of a specific (timed) action of the form $\alpha = (m, r, s, t)$, where $r \leq s \leq t$, m stands for the proper action, $r = \mathbf{entime}(x)$, $s = \mathbf{stime}(x)$, and $t = \mathbf{cptime}(x)$. Formally,

we define timed actions as quadruples $\alpha = (m, r, s, t)$ such that m is a proper action ($m = \mathbf{ACTION}(\alpha)$) and r, s, t are real numbers satisfying $r \leq s \leq t$ and called respectively the enabling time of α ($r = \mathbf{ENTIME}(\alpha)$), the start time of α ($s = \mathbf{STIME}(\alpha)$), and the completion time of α ($t = \mathbf{CPTIME}(\alpha)$), and for each event x we replace x by $\varphi(x)$ such that $\mathbf{name}(\varphi(x)) = \mathbf{name}(x)$, $\mathbf{entime}(x) = \mathbf{ENTIME}(\mathbf{label}(\varphi(x)))$, $\mathbf{stime}(x) = \mathbf{STIME}(\mathbf{label}(\varphi(x)))$, and $\mathbf{cptime}(x) = \mathbf{CPTIME}(\mathbf{label}(\varphi(x)))$. The proper executed action can be defined as $\mathbf{action}(x) = \mathbf{ACTION}(\mathbf{label}(\varphi(x)))$.

The noninternality of timed actions is defined as the noninternality of the respective proper actions. The timed actions with the proper action internal are regarded to be internal. Noninternal timed actions are regarded to be complementary (and then composable) iff their proper actions are composable, start times are identical, and completion times are also identical. The composition of complementary timed actions α and β is defined by assuming

$$\mathbf{ACTION}(\alpha \bullet \beta) = \mathbf{ACTION}(\alpha) \bullet \mathbf{ACTION}(\beta)$$

$$\mathbf{ENTIME}(\alpha \bullet \beta) = max(\mathbf{ENTIME}(\alpha), \mathbf{ENTIME}(\beta))$$

$$\mathbf{STIME}(\alpha \bullet \beta) = \mathbf{STIME}(\alpha) = \mathbf{STIME}(\beta)$$

$$\mathbf{CPTIME}(\alpha \bullet \beta) = \mathbf{CPTIME}(\alpha) = \mathbf{CPTIME}(\beta).$$

These modifications allow us to combine timed configuration structures with the aid of operations on usual configuration structures and to define equivalences of timed configuration structures as for usual configuration structures.

The information about temporal features of events allows us to define important characteristics of behaviours (cf. [8]). In particular, we can distinguish configurations of such runs in which events are executed without unnecessary delays (eager configurations). Similarly, we can transform configurations to a special (stamped) form exhibiting how they follow each other in time, and we can say which of them can occur due to the precedence in time in case of a possibility of indeterministic choice.

We show that the respective sets of configurations constitute timed configuration structures which reflect the corresponding features of behaviours but possibly ignore some

others. For example, in the timed configuration structure of stampeded configurations we have exhibited the potential transitory chronicles of a behaviour and how they evolve, but we have not a complete information about the possible causal independence of events.

We show that the characteristics of timed behaviours in the form of timed configuration structures exhibiting special configurations are compositional in the sense that the characteristics of a composed behaviour can be obtained from those of component behaviours without a need of dealing with configuration structures representing the respective behaviours as such.

The paper is organized as follows. In section 2 we discuss general configuration structures. In section 3 we introduce the concept of timed configuration structures and define characteristics of such structures. In section 4 we discuss the operations on general and timed configuration structures and on the considered characteristics. In section 5 we discuss the equivalence of general and timed configuration structures and the relations between the equivalence and operations on configuration structures.

The present paper is an extension of [5]. Its novelties are: admission of time-consuming actions and studies of the introduced characteristics of timed configuration structures.

2 The general model

For models of behaviours we want to choose structures universal enough to represent behaviours both with and without timing. In this paper we choose event and configuration structures.

2.1. Definition. A (*generalized, finitary*) *event structure* is $E = (X, \leq, F)$, where X is a set (of possible *events*), $\leq$ is a quasiorder on X (a *causal relation*) such that each $x \in X$ has at most finitely many predecessors (that is events $y \in X$ with $y \leq x$), and F is a family of subsets of X (a family of *forbidden* sets) which is *persistent* in the sense that $g \in F$ whenever $g \subseteq X$ and $\downarrow g = \{x \in X : x \leq y \text{ for some } y \in g\}$ contains some $f \in F$.

The relation defined by

$$x \equiv y \text{ iff } x \leq y \text{ and } y \leq x$$

is called a *coincidence*. That defined by

$$x < y \text{ iff } x \leq y \text{ but not } y \leq x$$

is called a *strict* causal relation. Each finite $c \subseteq X$ which is *downward closed* in the sense that $\downarrow c = c$ and *admissible* in the sense that it does not contain any forbidden set is called a *configuration*. A subset of a configuration c which is also a configuration is called a *subconfiguration* of c. By $\mathbf{conf}(E)$ we denote the set of configurations of E. We say that E is *complete* if in this structure a subset $g \subseteq X$ is forbidden iff $\downarrow g$ contains a finite forbidden set. We say that E is a *conflict event structure* if in this structure a subset $g \subseteq X$ is forbidden iff $\downarrow g$ contains a two-element forbidden set (in such a case the forbidden two-element sets define a *conflict relation* $\sharp$, where $x \sharp y$ iff $\{x, y\} \in F$, and we write the event structure as $E = (X, \leq, \sharp)$). We say that E is *coincidence-free* if the causal relation is a partial order. Finally, given a set A (of *actions*, each action qualified as *internal* or *noninternal*), we say that E is a *labelled event structure* over A if each event $x \in X$ is a pair (n, α) consisting of a *name* n ($n = \mathbf{name}(x)$) and of a *label* α ($\alpha = \mathbf{label}(x)$), where $\alpha \in A$. An event $x \in X$ of such an E with $\alpha = \mathbf{label}(x)$ is then called an *occurrence* of α in E. By $\mathbf{les}(A)$ we denote the universe of all labelled event structures over A. $\square$

We shall deal with generalized event structures rather than with conflict ones normally considered in the literature. Note that the conflict relation defined with the aid of two-element forbidden sets is symmetric and persistent in the sense that $x \sharp z$ whenever $x \sharp y$ and $y \leq z$.

The way of defining labelled event structures as event structures with events of a particular form has been chosen with the purpose of simplifying notation.

2.2. Example. Consider a system consisting of a producer which produces and offers in a store some objects, and of a consumer which consumes these objects, if available.

Suppose that at the beginning no object is available and that the producer may terminate its activity after having produced any number of objects. The behaviour of this system can be represented by the labelled event structure in fig. 1, where p stands for production, c for consumption, and t for termination (we qualify p, c, and t as internal actions; for transparency we exhibit only actions and we omit relations which follow from the transitivity of causal order and from the persistency of conflict relation). Formally, the set of possible events can be defined as

$$\{(1,p),(2,p),...,(1,c),(2,c),...,(1,t),(2,t),...\},$$

the causal order can be defined by assuming

$$(1,p) \leq (2,p) \leq ...,(1,c) \leq (2,c) \leq ...,(i,p) \leq (i,c),(i,p) \leq (i+1,t),$$

the conflict relation can be defined by assuming

$$(i,p)\sharp(i,t),$$

and we have $\mathbf{label}(i,x) = x$. Configurations can be characterized as sets of the form

$$\{(1,p),...(i,p),(1,c),...,(j,c)\}$$

or

$$\{(1,p),...,(i,p),(1,c),...,(j,c),(i+1,t)\},$$

where $j \leq i$. $\square$

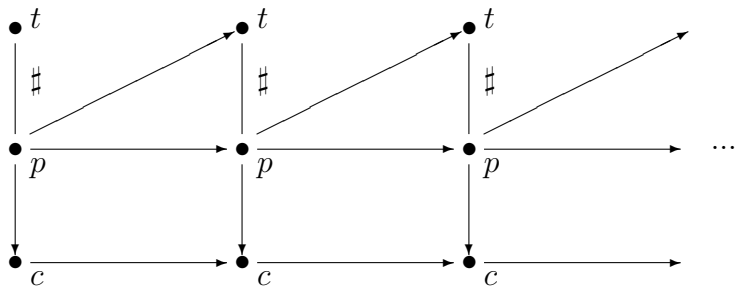

FIGURE 1: The behaviour of the producer-consumer system

2.3. Example. Consider a project as in Critical Path Method, that is a finite, acyclic, directed graph with edges representing activities and the order of edges representing the order of execution. Denote by U the set of edges and by $\prec$ the strict partial order of edges (with $u \prec u'$ iff there is a directed path, possibly of zero length, from the target node of u to the source node of u'). Suppose that each activity $u \in U$ is performed by executing an action $\alpha = \lambda(u)$ (internal or noninternal) from a set A, where each $\alpha \in A$ has a duration $d(\alpha) \geq 0$.

Such a project (see fig. 2 for an example) can be represented in the form of a labelled event structure $E = (X, \leq, \sharp)$. It suffices to define $X = \{(u, \lambda(u)) : u \in U\}$, $(u, \lambda(u)) \leq (u', \lambda(u'))$ iff $u \prec u'$ or $u = u'$, $\sharp = \emptyset$, and $\mathbf{label}(u, \lambda(u)) = \lambda(u)$ (in fig. 3 we show such a labelled event structure for the project in fig. 2). Configurations can be characterized as finite sets p of events such that $(u, \lambda(u)) \in p$ whenever $u \prec u'$ for some $(u', \lambda(u')) \in p$.

A project with indeterminate durations of actions can be represented by splitting each event $(u, \lambda(u))$ into the family of events of the form $(u, (\lambda(u), t))$ with t ranging over nonnegative real numbers (the possible durations of $\lambda(u)$), each event in conflict with the remaining events of the family, and by taking the causal order of original events (see fig. 4). Formally, the set of possible events can be defined as $\{(u, (\lambda(u), t)) : u \in U, t \geq 0\}$, the causal order as $(u, \lambda(u), t)) \leq (u', (\lambda(u'), t'))$ iff $u \prec u'$ or $u = u'$, and the conflict relation as the one with $(u, \lambda(u), t))\sharp(u', (\lambda(u'), t'))$ for $u = u'$ and $t \neq t'$. Here we have $\mathbf{label}(u, \lambda(u), t)) = (\lambda(u), t)$. Configurations can be characterized as finite sets p of events such that $(u, \lambda(u), t)) \in p$ whenever $u \prec u'$ for some $(u', (\lambda(u'), t')) \in p$ and $(u, \lambda(u), t)) \in p$ excludes $(u, \lambda(u), t')) \in p$ for $t \neq t'$.

The requirement of the lack of cycles in the graph of a project may be replaced by the weaker condition that all activities of each possible cycle are instantaneous. In such a case each cycle represents a set of coincident events, the strict partial order of edges must be defined as the existence of a suitable path in one only direction, and the causality is represented in the corresponding event structure by a quasiorder (see figures 5 and 6). □

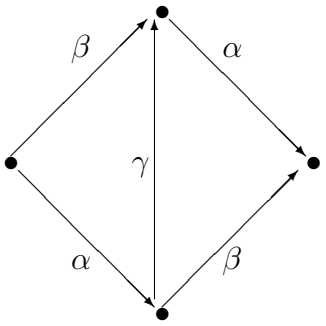

FIGURE 2: A project Π

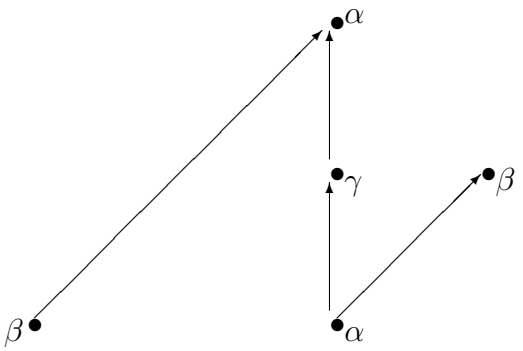

FIGURE 3: A labelled event structure for Π

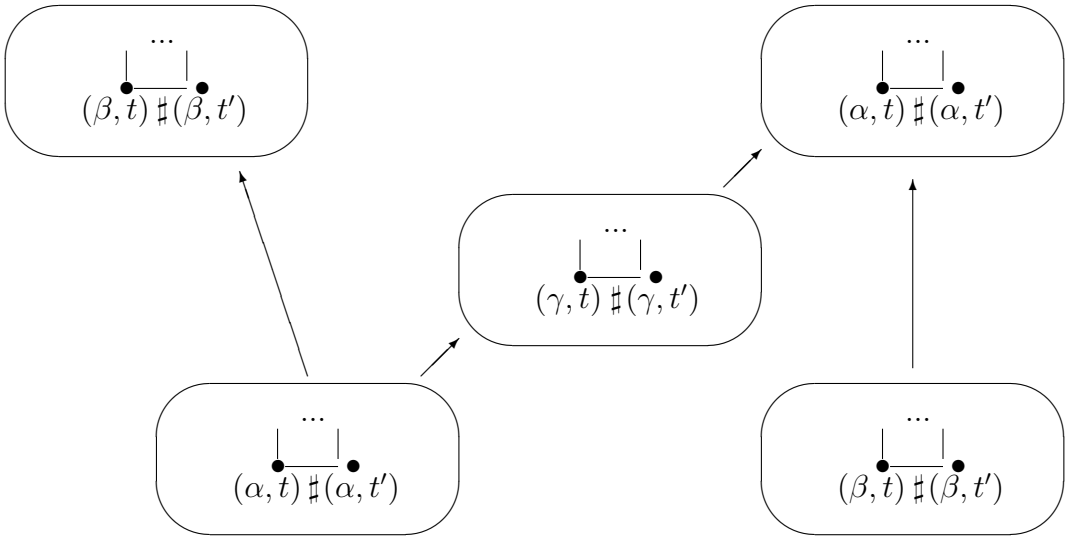

FIGURE 4: A labelled event structure for a project with indeterminate durations of actions

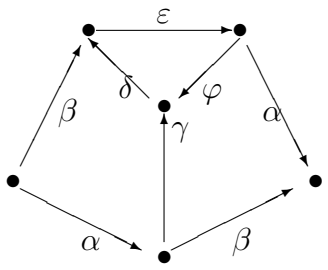

FIGURE 5: A project Π' with coincident activities

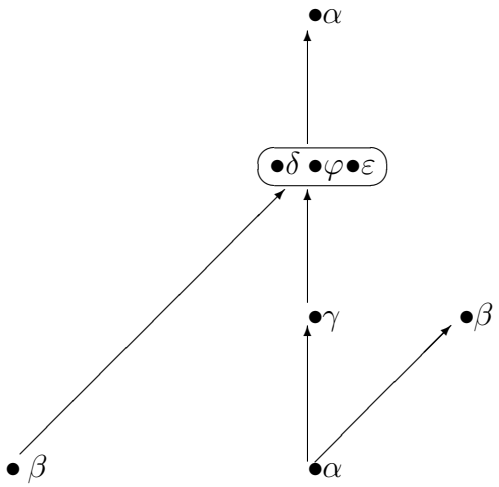

FIGURE 6: A labelled event structure for Π'

Families of configurations of event structures can be characterized as members of axiomatically defined class of set systems called configuration structures.

2.4. Definition. A (*general, finitary*) *configuration structure* is a nonempty family P of finite sets such that:

(1) $\bigcap S \in P$ for each nonempty subfamily $S \subseteq P$ which is *bounded* in P in the sense that all $s \in S$ are subsets of some $p \in P$,

(2) $\bigcup S \in P$ for each subfamily $S \subseteq P$ which is bounded in P.

Each $x \in \bigcup P$ is called an *event* of P. Each $p \in P$ is called a *configuration* of P. A subset of a configuration p which is also a configuration is called a *subconfiguration* of p. We say that P is *prime* if it is closed w.r. to intersections of arbitrary nonempty subfamilies $S \subseteq P$. A subfamily $S \subseteq P$ is said to be *finitely compatible* (resp.: *pairwise compatible*) in P if each finite (resp.: two-element) subfamily of S is bounded in P. We say that P is *coherent* if each finite subfamily $S \subseteq P$ which is pairwise compatible in P is also bounded in P. We say that P is *coincidence-free* if every two different events x, y of every configuration $p \in P$ can be *separated* by a subconfiguration p' of p in the sense that either $x \in p'$ and not $y \in p'$ or not $x \in p'$ and $y \in p'$. Finally, given a set A (of *actions*, each action qualified as *internal* or *noninternal*), we say that P is a *labelled*

configuration structure over A if each event $x \in \bigcup P$ is a pair (n, α) consisting of a *name* n ($n = \mathbf{name}(x)$) and of a *label* α ($\alpha = \mathbf{label}(x)$), where $\alpha \in A$. An event $x \in \bigcup P$ with $\mathbf{label}(x) = \alpha$ of such a structure P is then called an *occurrence* of α in P. By $\mathbf{lcs}(A)$ we denote the universe of all labelled configuration structures over A. By **nil** we denote the configuration structure $\{\emptyset\}$. □

2.5. Example. The configurations of the labelled event structure in 2.2 constitute a prime, coherent, and coincidence-free labelled configuration structure (see fig. 7). □

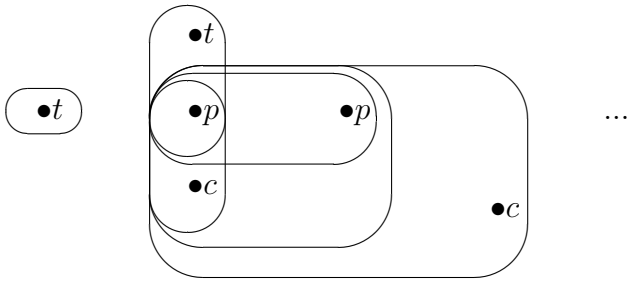

FIGURE 7: The labelled configuration structure of the producer-consumer system

2.6. Example. The configurations of the labelled event structures in 2.3 constitute prime, and coherent labelled configuration structures. Except for the case of the event structure shown in fig. 6, all these configuration structures are coincidence-free. □

The relationship between event structures and prime configuration structures is characterized by the following two propositions.

2.7. Proposition. For each event structure $E = (X, \leq, F)$, the set $\mathbf{conf}(E)$ is a prime configuration structure. If E is a conflict (resp.: coincidence-free) event structure then $\mathbf{conf}(E)$ is a coherent (resp.: coincidence-free) configuration structure. □

Proof: Suppose that $P = \mathbf{conf}(E)$. Let $S \subseteq P$ be nonempty. Then $\bigcap S$ is downward-closed since all members of S are downward-closed. Moreover, $\bigcap S$ does not contain any forbidden set since each $s \in S$ does not contain any forbidden set. Hence $\bigcap S$ is a configuration of E. Similarly, $\bigcup S \in P$ for each family S of subconfigurations of a

configuration of E. Thus $\mathbf{conf}(E)$ is a configuration structure.

Suppose that E is a conflict event structure and that some finite $S \subseteq P$ is pairwise compatible in P. Then $\bigcup S$ is downward-closed as the union of downward-closed sets $s \in S$. On the other hand, $\bigcup S$ cannot contain any two-element forbidden subset since otherwise S could not be pairwise compatible in P. Consequently, $\bigcup S$ is bounded in P. Thus $\mathbf{conf}(E)$ is a coherent configuration structure.

Suppose that E is coincidence-free. Then, for every two elements of a configutation c, either one of these elements is a strict cause of the other and the least subconfiguration of p which contains it does not contain the latter one, or these two elements are incomparable and the least subconfiguration of p which contains any of them does not contain the other. Thus $\mathbf{conf}(E)$ is a coincidence-free configuration structure. Q.E.D.

2.8. Proposition. For each prime configuration structure P there exists a (generalized, finitary) complete event structure $\mathbf{evs}(P)$ such that $\mathbf{conf}(\mathbf{evs}(P)) = P$. If P is coherent (resp.: coincidence-free) then $\mathbf{evs}(P)$ is a conflict (resp.: coincidence-free) event structure. $\Box$

Proof: We define $X = \bigcup P$, $x \leq y$ iff $y \in p$ implies $x \in p$ for all $p \in P$, and

$$F = \{f \subseteq X : f \text{ cannot be covered by any finitely compatible subfamily } S \subseteq P\}.$$

The relation $\leq$ is reflexive and transitive and thus a quasiorder. Moreover, each $x \in X$ has at most finitely many predecessors. Suppose that $g \subseteq X$ and that $\downarrow g$ contains some $f \in F$. Then $g \in F$ since otherwise g would be contained in some $p \in P$ and hence $\downarrow g$, and consequently f, would be contained in a configuration. Thus $E = (X, \leq, F)$ is an event structure.

It is straightforward that each $p \in P$ is a configuration of E, that is $P \subseteq \mathbf{conf}(E)$. In order to prove the converse inclusion suppose that d is a configuration of E. Then each $e \subseteq d$ must be contained in some $p \in P$. In particular, there exists a least $p_d \in P$ such that $d \subseteq p_d$. Moreover, d is the union of "prime" configurations of the form $\downarrow x$ over all $x \in d$, where each $\downarrow x$ is the intersection of all $p \in P$ such that $x \in p$. Without a loss of generality we may assume that all such $p \in P$ are contained in p_d, which implies that

$d \in P$. Hence $\mathbf{conf}(E) \subseteq P$. Thus $\mathbf{conf}(E) = P$ and consequently $\mathbf{evs}(P) = E$ satisfies the requirement of the proposition.

Suppose that P is coherent and that $f \in F$. Then f must contain a two-element forbidden set since otherwise it could be covered by a family $S \subseteq P$ which is pairwise compatible in P and thus, due to the coherence of P, finitely compatible in P.

Suppose that P is coincidence-free and that $x \leq y$ and $y \leq x$. Then x and y cannot be separated by any member of P and hence $x = y$. Consequently, the causal relation is a partial order, as required. Q.E.D.

In case of arbitrary (not necessarily prime) configuration structures we have not such a simple way of transforming them into event structures since instead of a single causal relation we obtain a family of local causal relations, each relation corresponding to a particular configuration.

2.9. Proposition(after [9])**.** Let P be a configuration structure. The family $(\leq_p : p \in P)$ of quasiorders (*local causal relations*) defined by

$$x \leq_p y \text{ iff } x, y \in p \text{ and } y \in q \text{ implies } x \in q \text{ for all subconfigurations } q \subseteq p,$$

is *consistent* in the sense that $\leq_q$ is the restriction of $\leq_p$ to q for all $p, q \in P$ such that $q \subseteq p$. Similarly for the family $(\equiv_p : p \in P)$ (of *local coincidences*), where

$$x \equiv_p y \text{ iff } x \leq_p y \text{ and } y \leq_p x$$

and for the family $(<_p : p \in P)$ (of *local strict causal relations*), where

$$x <_p y \text{ iff } x \leq_p y \text{ but not } y \leq_p x. \ \square$$

Proof: Suppose that $p, q \in P$, $q \subseteq p$, and $x, y \in q$. If $x \leq_q y$ and $y \in r$ for a subconfiguration of p then $y \in r \cap q$, where $r \cap q$ is a subconfiguration of q, and hence $x \in r$. Consequently, $x \leq_p y$. If $x \leq_p y$ and $y \in s$ for a subconfiguration s of q then $x \in s$ since $s \subseteq p$. Consequently, $x \leq_q y$. Q.E.D.

The idea of considering configuration structures which not necessarily are prime is due to [9] and it has been invented in order to avoid technical difficulties in modelling

the parallel composition of behaviours. It does not lead to essentially new type of models since each general configuration structure can easily be transformed into a prime one.

2.10. Proposition. For each configuration structure P there exists a prime configuration structure Q, a bijection $B : P \longrightarrow Q$, and a family $(b_{pq} : p \longrightarrow q \ : (p,q) \in B$ of bijections such that:

(1) $B(\bigcap S) = \bigcap(B(s) : s \in S)$ for each nonempty bounded $S \subseteq P$ and $B(\bigcup S) = \bigcup(B(s) : s \in S)$ for each bounded $S \subseteq P$,

(2) $b_{pq} = b_{p'q'} \cap (p \times q)$ whenever $p \subseteq p'$ and $q \subseteq q'$. $\Box$

Proof: It suffices to define $B(p)$ as the set of pairs (x, p_x) with $x \in p$ and p_x being the least subconfiguration of p that contains x, and to define $b_{pq}(x)$ with $q = B(p)$ as (x, p_x) (the fact that Q is a configuration structure follows easily from (1)). Q.E.D.

2.11. Example. By equipping each event of each configuration of the configuration structure in figure 8 with the least subconfiguration containing this event we obtain a prime configuration structure. $\Box$

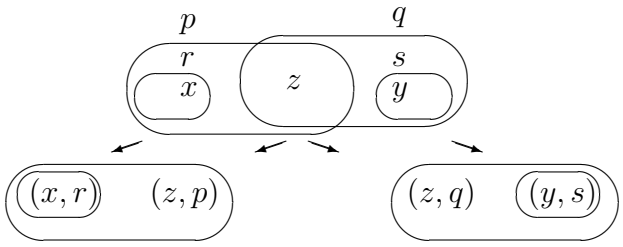

FIGURE 8: Making a configuration structure prime

In the case of labelled configuration structures which are of our interest in this paper the respective prime labelled configuration structure can be obtained as follows.

2.12. Definition. Given a labelled configuration structure P and its configuration p,

by a *rooted configuration* corresponding to p we mean the set of pairs of the form

$$\mathbf{copy}_p(x) = ((x, p_x), \mathbf{label}(x))$$

where $x \in p$ and p_x denotes the least subconfiguration of p that contains x. By $\mathbf{rooted}(P)$ we denote the set of rooted configurations corresponding to the configurations of P. $\Box$

The names of events of rooted configurations are equipped with the information from which one can reconstruct the causal order of the respective events. Moreover, this information allows one to identify uniquely the runs to which an event belongs.

2.13. Proposition. For each labelled configuration structure P the set $Q = \mathbf{rooted}(P)$ is a prime labelled configuration structure, $\mathbf{prime}(P)$. Moreover, there exist: a bijection $B : P \longrightarrow Q$ and a family $(b_{pq} : p \longrightarrow q : \ (p, q) \in B)$ which satisfy (1) and (2) of 2.10. Besides, if P is coherent (resp.: coincidence-free) then so is Q. $\Box$

Proof: One may follow the line of the proof of 2.10 replacing pairs of the form (x, p_x) by the respective $\mathbf{copy}_p(x)$.

As usual, we consider labelled event and configuration structures up to isomorphisms, where isomorphisms are defined as follows.

2.14. Definition. An *isomorphism* between two labelled configuration structures P and Q is a bijective correspondence $f : \bigcup P \longrightarrow \bigcup Q$ such that $\mathbf{label}(f(x)) = \mathbf{label}(x)$ for all $x \in \bigcup P$, $f(P) \in Q$ for all $p \in P$, and $f^{-1}(q) \in P$ for all $q \in Q$. If such an isomorphism exists then we say that P and Q are *isomorphic* and write $P \approx Q$. An isomorphism between labelled event structures E and E' is defined as an isomorphism between $\mathbf{conf}(E)$ and $\mathbf{conf}(E')$. $\Box$

In the rest of the present paper we consider only labelled configuration structures over a set A of actions. We recall that by $\mathbf{lcs}(A)$ we denote the universe of such configuration structures.

3 The model with timing

As we have said, event and configuration structures may reflect not only causality and branching of behaviours, but also their timing. We use for this purpose labelled event and configuration structures over a set of specific (timed) actions. Such actions are obtained by equipping each proper action with all the possible enabling times, start times, and completion times.

3.1. Definition. Given a set A of actions (each action qualified as *internal* or *noninternal*), a *timed action* over A is $\alpha = (m, r, s, t)$, where m is an action from A (the *proper action* of α, $m = \mathbf{ACTION}(\alpha)$) and r,s,t are real numbers (*temporal parameters* of α) such that $r \leq s \leq t$ (resp.: the *enabling time* of α, $r = \mathbf{ENTIME}(\alpha)$, the *start time* of α, $s = \mathbf{STIME}(\alpha)$, and the *completion time* of α, $t = \mathbf{CPTIME}(\alpha)$). If $\mathbf{ACTION}(\alpha)$ is internal (resp.: noninternal) then we say that α is *internal* (resp.: *noninternal*). If $\mathbf{ENTIME}(\alpha) = \mathbf{STIME}(\alpha)$ then we say that α is *critical*. By $\mathbf{ta}(A)$ we denote the set of timed actions over A. $\Box$.

Behaviours with timing (timed behaviours) can be represented by labelled event and configuration structures with labels from a set of timed actions, where the temporal parameters of actions agree with the causal order of action occurrences.

3.2. Definition. Let A be a set of actions. A *timed configuration structure* over A is a labelled configuration structure P with labels from the set $\mathbf{ta}(A)$ of timed actions over A (that is with $\mathbf{label}(x) \in \mathbf{ta}(A)$ for all $x \in \bigcup P$) such that, for all $x, y \in \bigcup P$:

(1) $\mathbf{CPTIME}(\mathbf{label}(x)) \leq \mathbf{STIME}(\mathbf{label}(y))$ whenever $x <_p y$ for some $p \in P$,

(2) $\mathbf{STIME}(\mathbf{label}(x)) = \mathbf{STIME}(\mathbf{label}(y))$ whenever $x \equiv_p y$ for some $p \in P$.

Similarly, a *timed event structure* over A is a labelled event structure E with labels from $\mathbf{ta}(A)$ such that the set $\mathbf{conf}(E)$ of configurations of E is a timed configuration structure over A. For each event x of a timed event or configuration structure we say that x is *internal* (resp.: *noninternal*, *critical*) if the timed action $\mathbf{label}(x)$ is internal

(resp.: noninternal, critical), and we define $\mathbf{action}(x) = \mathbf{ACTION}(\mathbf{label}(x))$ (the *proper action* of x), $\mathbf{entime}(x) = \mathbf{ENTIME}(\mathbf{label}(x))$ (the *enabling time* of x), $\mathbf{stime}(x) = \mathbf{STIME}(\mathbf{label}(x))$ (the *start time* of x), and $\mathbf{cptime}(x) = \mathbf{CPTIME}(\mathbf{label}(x))$ (the *completion time* of x). By $\mathbf{tcs}(A)$ (resp.: by $\mathbf{tes}(A)$) we denote the universe of timed configuration (resp.: event) structures over A. $\square$

According to this definition we have $\mathbf{tcs}(A) \subseteq \mathbf{lcs}(\mathbf{ta}(A))$.

3.3. Example. Consider a possible execution of a project as in 2.3. For each activity u, denote by $\mathbf{en}(u)$ the time of completion of all the strict predecessors of u, by $\mathbf{st}(u)$ the start time of u, and by $\mathbf{ct}(u)$ the completion time of u. Then

$$\max(\mathbf{ct}(u') : u' \prec u) \leq \mathbf{en}(u) \leq \mathbf{st}(u) \leq \mathbf{ct}(u) = \mathbf{st}(u) + d(\lambda(u))$$

and each $u \in U$ may be regarded as an event of executing the timed action $\mu(u) = (\lambda(u), \mathbf{en}(u), \mathbf{st}(u), \mathbf{ct}(u))$, and it may be represented as $x = (u, \mu(u))$. Consequently, the considered execution can be represented by the timed event structure $E' = (X', \leq', \sharp')$, where $X' = \{(u, \mu(u)) : u \in U\}$, $(u, \mu(u)) \leq' (u', \mu(u'))$ iff $u \prec u'$ or $u = u'$, and $\sharp' = \emptyset$. In this case configurations can be characterized as finite subsets of X' which contain $(u, \mu(u))$ whenever they contain $(u', \mu(u'))$ with $u \prec u'$. In fig. 9 we show a timed event structure which represents a possible execution of the project in fig. 2 for $d(\alpha) = 2$, $d(\beta) = 4$, $d(\gamma) = 1$.

For a project with indeterminate durations of actions it is convenient to consider activities together with sets of their predecessors, each of them with a possible duration. More precisely, we may consider sets v of pairs of the form (u, t), where $u \in U$, $t \geq 0$, $(u, t) \in v$, and $(u, t') \in v$ implies $t = t'$, and there is $u_v \in U$ such that $(u, t) \in v$ for some t whenever $u \prec u_v$ or $u = u_v$, and, for each such a set v, the corresponding actvity u_v, and an execution of the project with durations of activities as in v, we may regard v as an event of executing the timed action $\nu(v) = (\lambda(u_v), \mathbf{en}(u_v), \mathbf{st}(u_v), \mathbf{ct}(u_v))$ and we may represent such an event as $x = (v, \nu(v))$. Consequently, the behaviour consisting of the possible executions of the project can be represented by the timed event structure $E'' = (X'', \leq'', \sharp'')$, where X'' is the set of possible pairs $(v, \nu(v))$, $(v, \nu(v)) \leq'' (v', \nu(v'))$ iff

$v \subseteq v'$, and $(v, \nu(v)) \sharp'' (v', \nu(v'))$ whenever v and v' contain some (u, t), (u', t') with $u = u'$ and $t \neq t'$. $\square$

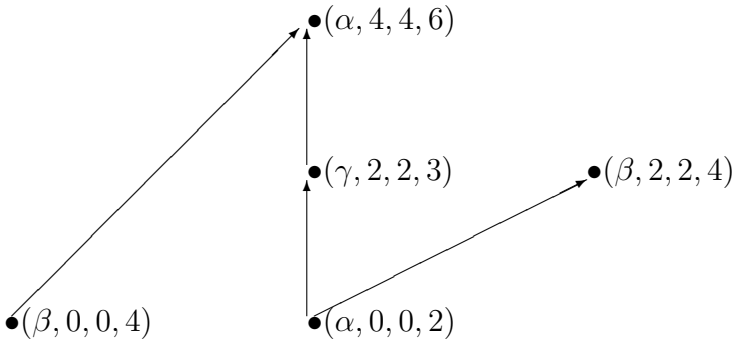

FIGURE 9: A timed configuration structure

This example not only illustrates the concept of a timed configuration structure, but it also suggests how to define such a structure for a behaviour with time-consuming actions from a representation in the form of a usual configuration structure. The respective result can be formulated as follows.

Let A be a set of time-consuming actions and δ a real-valued function assigning the corresponding non-negative duration to each $\alpha \in A$. A labelled configuration structure $P \in \mathbf{lcs}(A)$ is said to be *proper* for A and δ if $\delta(\alpha) = 0$ for each action which is a label of an event x which is coincident with some other event y. By $\mathbf{plcs}_\delta(A)$ we denote the universe of all $P \in \mathbf{lcs}(A)$ which are proper for A and δ. For each such P we define $\mathbf{timed}(P)$ as the family of sets of the form

$$\{(U(x), (m(x), r(x), s(x), t(x))) : x \in p\}$$

where $p \in P$, $m(x), r(x), s(x), t(x)$ are respectively an action symbol and non-negative real numbers such that the following condition $C(y)$ is satisfied for all $y \in p$:

$$m(y) = \mathbf{label}(y) \text{ and } t(y) = s(y) + \delta(m(y)) \text{ and } max(t(z) : z <_p y) \leq r(y) \leq s(y)\ ,$$

and $U(x) = \{(y, m(y), r(y), t(y)) : y <_p x)\}$. Then we obtain the following property.

3.4. Proposition. For each $P \in \mathbf{plcs}_\delta(A)$ the family $\mathbf{timed}(P)$ is a timed configuration structure. $\square$

Proof: The property follows from the fact that the inclusion $q \subseteq q'$ holds for $q, q' \in$ **timed**(P) iff q, q' correspond resp. to $p, p' \in P$ such that $p \subseteq p'$ and $U(x)$, $m(x)$, $r(x)$, $s(x)$, $t(x)$ are the same in q and q' for all $x \in p$. $\square$

Timed configuration structures obtained in the way just described due to assigning durations to actions are very particular members of the universe of all timed configuration structures. The only conditions they respect are the prescribed order and durations of actions whereas in general we may have to do with conditions of any nature.

The information about about timing of a behaviour (that is about temporal features of its events) allows us to define characteristics of behaviours which cannot be expressed only in terms of causality and branching.

One characteristic of a behaviour which can be defined due to timing is the set of configurations of runs in which events are executed without unnecessary delays. Configurations of this type can be defined as follows.

3.5. Definition. Given a timed configuration structure P, a configuration $p \in P$ is said to be *eager* if in each coincidence class contained in p there is a noninternal or critical event. By **eager**(P) we denote the set of eager configurations of P. $\square$

The following property of the sets of eager configurations is a direct consequence of the definition.

3.6. Proposition. For each timed configuration structure P, **eager**(P) is a timed configuration structure. Moreover, **eager**(**eager**(P)) = **eager**(P). $\square$

3.7. Example. The behaviour of the timed Petri net in fig. 10 can be represented by a timed configuration structure as in fig. 11, where γ,δ, φ, ψ represent the respective transfers of tokens and α, β, γ, δ, φ, ψ, χ are regarded to be internal actions. The configurations of this event structure are eager since the events of executing α, β, γ, δ, χ are critical and in the two-element coincidence class consisting of the events of executing φ and ψ there is a critical event (that of executing ψ). $\square$

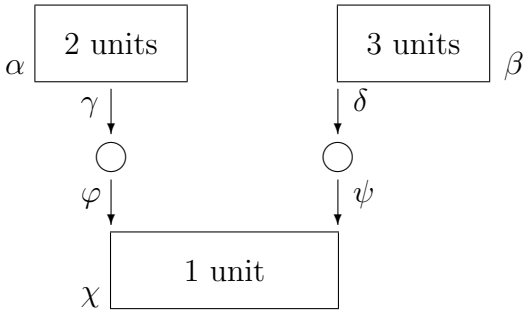

FIGURE 10: A timed Petri net N

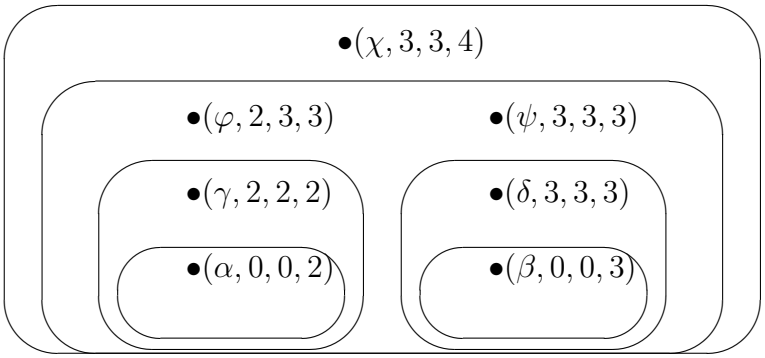

FIGURE 11: A timed configuration structure for N

Another characteristic of a behaviour with timing can be obtained by transforming configurations to a form exhibiting how they follow each other in time.

3.8. Proposition. Given a timed configuration structure P, its configuration p, and an event $x \in p$, the set of events $y \in p$ such that $\mathbf{stime}(y) < \mathbf{stime}(x)$, written as $\mathbf{past}_p(x)$, is a subconfiguration of p. □

Proof: We have $\mathbf{past}_p(x) = \{y \in p : \mathbf{stime}(y) < \mathbf{stime}(x)\}$. For each $y \in \mathbf{past}_p(x)$ the least subconfiguration p_y of p that contains y cannot contain x since it would imply $x \leq_p y$ and thus it could not be $\mathbf{stime}(y) < \mathbf{stime}(x)\}$. On the other hand, $\mathbf{past}_p(x)$ is the union of all such subconfigurations p_y and thus a subconfiguration of p. Q.E.D.

3.9. Definition. Given a timed configuration structure P and its configuration p, by a *stamped configuration*, or a *chronicle*, corresponding to p we mean the set of pairs of the form $\mathbf{std}_p(x) = ((x, \mathbf{past}_p(x)), \mathbf{label}(x))$. where $x \in p$. By $\mathbf{stamped}(P)$ we denote the set of stamped configurations corresponding to the configurations of P. □

An important role of stamped configurations follows from the fact that they are reached in time in the order of containment and thus may be interpreted as potential transitory chronicles. Namely, we have the following property.

3.10. Proposition. If $q, q' \in \mathbf{stamped}(P)$ are such that $q \subseteq q'$ then q contains all $x \in q'$ such that $\mathbf{stime}(x) < \mathbf{stime}(y)$ for some $y \in q$. □

Proof: Let q and q' be stamped configurations corresponding to p and p', respectively. As $q \subseteq q'$, we have $\mathbf{past}_p(y) = \mathbf{past}_{p'}(y)$. As $x, y \in q'$ and $\mathbf{stime}(x) < \mathbf{stime}(y)$, we have $x \in \mathbf{past}_{p'}(y)$. Hence $x \in \mathbf{past}_p(y)$. This implies $x \in p$. Q.E.D.

Noting that the stamped configurations which are intersections or unions of stamped configurations correspond precisely to the intersections or unions of the respective original configuations, and considering the bijections $\mathbf{std}_p(x) \longmapsto x$ between stamped and the respective original configurations, we obtain the following proposition (cf. 2.13).

3.11. Proposition. For each timed configuration structure P the set $Q = \mathbf{stamped}(P)$ is a timed configuration structure. Moreover, there exist a surjection $H : Q \longrightarrow P$ and a family $(h_{qp} : q \longrightarrow p : (q, p) \in H)$ of bijections such that:

(1) $H(\bigcap S) = \bigcap(H(s) : s \in S)$ for each nonempty bounded $S \subseteq Q$ and $H(\bigcup S) = \bigcup(H(s) : s \in S)$ for each bounded $S \subseteq Q$,

(2) $h_{qp} = h_{q'p'} \cap (q \times p)$ whenever $q \subseteq q'$ and $p \subseteq p'$. □

3.12. Example. By equipping the names of events of the timed configuration structure in fig. 12 with the respective subsets of events with earlier start times we obtain the timed configuration structure of stamped configurations. □

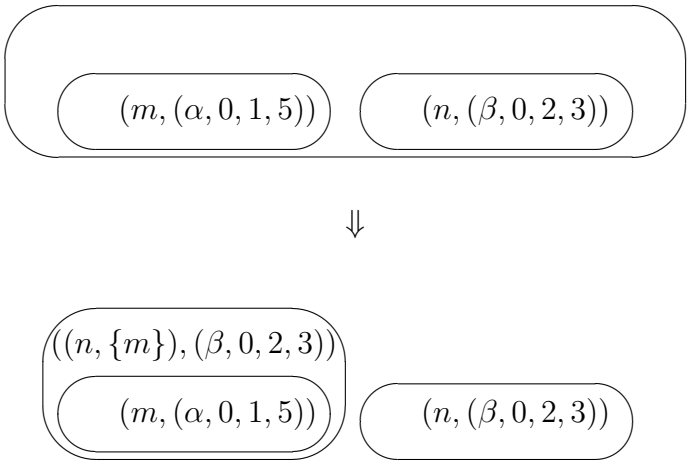

FIGURE 12: Stamping a timed configuration structure

The stamping of configurations of timed configuration structures leads to a linearization of runs and their parts. However, such a linearization results usually in a loss of information about the existing independency of events (see fig. 12, where after stamping the events of executing α and β are no longer independent).

The third (and may be most important) characteristic of a timed behaviour is the set of global states that are reached due to the choice of internal events according to the principle "first enabled first chosen". Such states are represented by stamped configurations which are fast in the following sense.

3.13. Definition. Given a timed configuration structure P, a stamped configuration $q \in \mathbf{stamped}(P)$ is said to be *fast* if, for each event $x \in q$ and each stamped configuration $q' \in \mathbf{stamped}(P)$ and each internal event $y \in q'$ such that $\mathbf{past}_{q'}(y) = \mathbf{past}_q(x)$, we have $\mathbf{stime}(x) \leq \mathbf{stime}(y)$. By $\mathbf{fast}(P)$ we denote the set of fast stamped configurations corresponding to the configurations of P. $\square$

Noting that each stamped configuration which is contained in a fast configuration is fast, and taking into account 3.10, we obtain the following property.

3.14. Proposition. For each timed configuration structure the set $\mathbf{fast}(P)$ is a timed

configuration structure. □

3.15. Example. The behaviour of the timed Petri net in fig. 13 can be represented by a timed configuration structure as in fig. 14, where α, β, γ, δ, ε are regarded to be internal actions. The configurations of this structure consisting of executions of α, β, γ, δ are fast whereas the configuration consisting of executions of α, β, γ, ε is not. This reflects the fact that δ is enabled before ε and then executed without any delay so that no conflict arises between α and ε. □

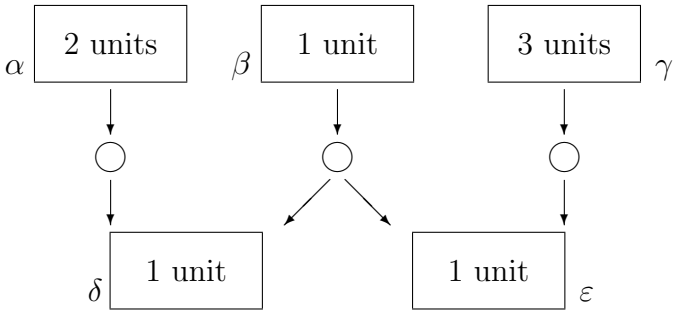

FIGURE 13: A timed Petri net N' with a choice

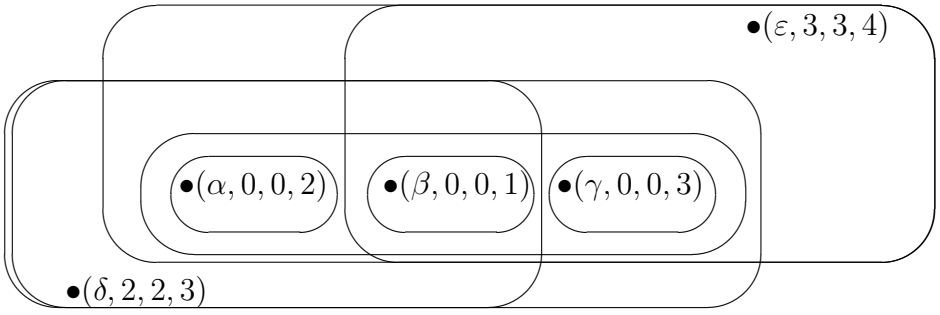

FIGURE 14: The timed configuration structure of N'

4 Compositionality

Behaviours one has usually to deal with in practice are rather complex. This creates a need of decomposing them into simpler components and then constructing the respective models by combining models of components with the aid of suitable operations.

Event and configuration structures representing behaviours (timed or not) can be combined with the aid of operations like those introduced in [9]. We define directly only some basic operations. Then we introduce a complete partial order of the respective structures such that the basic operations are continuous and we extend the collection of basic operations by superposing the already defined operations and by solving the fixed-point equations which can be formulated with the aid of such operations. This extension is done for labelled configuration structures since such structures seem to be most convenient and adequate for various applications.

The basic operations on labelled configuration structures are defined assuming that actions from a considered universe A (and the respective events) are qualified as *internal* or *noninternal* (cf. 2.1 and 2.4) and that some pairs of noninternal actions (and the respective pairs of events) are regarded as *complementary* and then can be combined into single actions with the aid of a special associative partial operation $(\alpha, \beta) \longmapsto \alpha \bullet \beta$ (this assumption is similar to that in [9], where actions are supposed to be elements of special monoids called synchronization algebras). The respective definitions can be obtained as follows.

4.1. Proposition. For all labelled configuration structures $P_0, P_1 \in \mathbf{lcs}(A)$, each $K \subseteq A$ which contains all internal actions, and each injective endomorphism h of A (that is an injection $h : A \longrightarrow A$ which preserves the internality and noninternality of actions, their complementarity, and the operation of composing actions), we have the following configuration structures in $\mathbf{lcs}(A)$:

(1) $P_0 \mid K$, the result of *restricting* P_0 to K, where $p \in P_0 \mid K$ iff $p \in P_0$ and $\mathbf{label}(x) \in K$ for all $x \in p$,

(2) $P_0 h$, the result of *relabelling* P_0 according to h, where $p \in P_0 h$ iff

$$p = \{(\mathbf{name}(x), h(\mathbf{label}(x))) : x \in p_0\}$$

for some $p_0 \in P_0$,

(3) $P_0; P_1$, the result of *prefixing* P_0 to P_1, where $p \in P_0; P_1$ iff

$$p = \{((0, x), \mathbf{label}(x)) : x \in p_0\}$$

for some $p_0 \in P_0$ or

$$p = \{((0, x), \mathbf{label}(x)) : x \in p_0\} \cup \{((1, y), \mathbf{label}(y)) : y \in p_1\}$$

for a maximal $p_0 \in P_0$ and some $p_1 \in P_1$,

(4) $P_0 + P_1$, the *sum* of P_0 and P_1, where $p \in P_0 + P_1$ iff

$$p = \{((0, x), \mathbf{label}(x)) : x \in p_0\}$$

for some $p_0 \in P_0$ or

$$p = \{((1, y), \mathbf{label}(y)) : y \in p_1\}$$

for some $p_1 \in P_1$,

(5) $P_0 \parallel P_1$, the *parallel composition* of P_0 and P_1, where $p \in P_0 \parallel P_1$ iff p *consists* of some $p_0 \in P_0$ and $p_1 \in P_1$ in the sense that $p = c_0(p_0) \cup c_1(p_1)$ with c_0 and c_1 defined by

$$c_0(x) = ((0, x), \mathbf{label}(x)) \text{ for } x \in p_0 - c^{-1}(p_1)$$

$$c_1(y) = ((1, y), \mathbf{label}(y)) \text{ for } x \in p_1 - c(p_0)$$

$$c_0(x) = c_1(y) = (((0, x), (1, y)), \mathbf{label}(x) \bullet \mathbf{label}(y)) \text{ for } (x, y) \in c$$

for a one-to-one correspondence $c \subseteq p_0 \times p_1$, called an *association*, such that

(5.1) $\mathbf{label}(x) \bullet \mathbf{label}(y)$ is defined for all $(x, y) \in c$,

(5.2) the reflexive and transitive closure $(R_c)^\star$ of the relation defined by

$$(\xi, \eta) \in R_c \text{ iff } c_0^{-1}(\xi) \leq_{p_0} c_0^{-1}(\eta) \text{ or } c_1^{-1}(\xi) \leq_{p_1} c_1^{-1}(\eta)$$

does not contain any pair (ξ, η) such that $c_0^{-1}(\eta) <_{p_0} c_0^{-1}(\xi)$ or $c_1^{-1}(\eta) <_{p_1} c_1^{-1}(\xi)$.

Moreover, in (5) the relation $(R_c)^\star$ coincides with the local causal relation $\leq_p$. $\Box$

Proof: For (1) - (4) proofs are trivial. For (5) we proceed as follows.

Let $P = P_0 \parallel P_1$. For each $p \in P$ we denote by p_0 and p_1 those configurations of P_0 and P_1, resp., of which p consists, and by c_p we denote the respective association. For each $S \subseteq P$ we define $S_0 = \{((c_s)_0)^{-1}(s) : s \in S\}$ and $S_1 = \{((c_s)_1)^{-1}(s) : s \in S\}$. If S is nonempty and bounded by some $p \in P$ then so are S_0 and S_1 with the upper bounds p_0 and p_1, resp.. Hence $((c_p)_0)^{-1}(\bigcap S) = \bigcap S_0 \in P_0$ and $((c_p)_1)^{-1}(\bigcap S) = \bigcap S_1 \in P_1$. On the other hand, each $q \subseteq p$ with $((c_p)_0)^{-1}(q) \in P_0$ and $((c_p)_1)^{-1}(q) \in P_1$ consists of $q_0 = ((c_p)_0)^{-1}(q)$ and $q_1 = ((c_p)_0)^{-1}(q)$ with the association $c_q = c_p \cap (q_0 \times q_1)$. Hence $\bigcap S$ consists of $\bigcap S_0$ and $\bigcap S_1$ with the association $\bigcap(c_s : s \in S)$ and, consequently, $\bigcap S \in P$. Similarly, $\bigcup S \in P$ for each bounded $S \subseteq P$.

In order to prove that $(R_c)^\star$ in (5) is contained in $\leq_p$ it suffices to consider $(\xi, \eta) \in R_c$ and note that each subconfiguration q of p with $\eta \in q$ contains also ξ.

In order to prove that $\leq_p$ is contained in $(R_c)^\star$ suppose that $\xi \leq_p \eta$. Note that, for each $\zeta \in p$, the least subconfiguration q of p with $\zeta \in q$ consists of q_0 and q_1, where q_i is the union of the least subconfigurations of q_i containing $(c_i)^{-1}(\zeta')$ over all $\zeta' \in p$ with $(\zeta', \zeta) \in (R_c)^\star$, and hence it contains exactly those $\zeta'' \in p$ for which $(\zeta'', \zeta) \in (R_c)^\star$. In particular, for $\zeta = \eta$ and $\zeta'' = \xi$ we obtain $(\xi, \eta) \in (R_c)^\star$, as required. Q.E.D.

4.2. Definition. The operations $P_0 \longmapsto P_0 \mid K$, $P_0 \longmapsto P_0 h$, $P_0 \longmapsto P_0; P_1$, $(P_0, P_1) \longmapsto P_0 + P_1$, $(P_0, P_1) \longmapsto P_0 \parallel P_1$, where P_0, P_1, K, h, are as in 4.1, are called *basic* operations on labelled configuration structures. $\Box$.

The operations just defined correspond to those of CCS (cf. [4] and [9]) except for the prefixing which allows to precede behaviours not only by a single action, but also by an arbitrary behaviour, the latter regarded as a constant. Being defined for configuration structures they can easily be transformed into the corresponding operations on event structures. For instance, the parallel composition of complete labelled event structures E_0 and E_1 can be defined as $\mathbf{evs}(\mathbf{prime}(\mathbf{conf}(E_0) \parallel \mathbf{conf}(E_1)))$ (see 2.8 and 2.13).

In order to apply basic operations on general labelled configuration structures to timed configuration structures over a universe A of actions it suffices to regard the latter ones as members of $\mathbf{lcs}(\mathbf{ta}(A))$, qualifying $\alpha \in \mathbf{ta}(A)$ as internal if $\mathbf{ACTION}(\alpha)$ is internal in A, qualifying $\alpha, \beta \in \mathbf{ta}(A)$ as complementary if $\mathbf{ACTION}(\alpha)$ and $\mathbf{ACTION}(\beta)$ are complementary in A and $\mathbf{STIME}(\alpha) = \mathbf{STIME}(\beta)$ and $\mathbf{CPTIME}(\alpha) = \mathbf{CPTIME}(\beta)$, and defining the composition of complementary actions $\alpha, \beta \in \mathbf{ta}(A)$ by

$$\mathbf{ACTION}(\alpha \bullet \beta) = \mathbf{ACTION}(\alpha) \bullet \mathbf{ACTION}(\beta)$$

$$\mathbf{ENTIME}(\alpha \bullet \beta) = max(\mathbf{ENTIME}(\alpha), \mathbf{ENTIME}(\beta))$$

$$\mathbf{STIME}(\alpha \bullet \beta) = \mathbf{STIME}(\alpha) = \mathbf{STIME}(\beta)$$

$$\mathbf{CPTIME}(\alpha \bullet \beta) = \mathbf{CPTIME}(\alpha) = \mathbf{CPTIME}(\beta).$$

However, in the case of timed configuration or event structures the basic operations must be applied with some care, or even slightly modified, if we want to guarantee the results to be timed configuration or event structures. All what we can say here is covered by the following property which is a direct consequence of the respective definitions.

4.3. Proposition. Let $P_0, P_1 \in \mathbf{tcs}(A)$, and let $K \subseteq A$ and $h : A \longrightarrow A$ be as in 4.1. Then $P_0 \mid K'$, $P_0 h'$, $P_0 + P_1$, $P_0 \parallel P_1$, where K' is the set of $\alpha \in \mathbf{ta}(A)$ with $\mathbf{ACTION}(\alpha) \in K$ and $h'(\alpha) = (h(\mathbf{ACTION}(\alpha)), \mathbf{ENTIME}(\alpha), \mathbf{STIME}(\alpha), \mathbf{CPTIME}(\alpha))$, are timed configuration structures in $\mathbf{tcs}(A)$. For each mapping of the form $\mathbf{shift}_d : \mathbf{ta}(A) \longrightarrow \mathbf{ta}(A)$, where

$$\mathbf{shift}_d(\alpha) = (\mathbf{ACTION}(\alpha), \mathbf{ENTIME}(\alpha) + d, \mathbf{STIME}(\alpha) + d, \mathbf{CPTIME}(\alpha) + d),$$

$P_0\mathbf{shift}_d$ is a timed configuration structure in $\mathbf{tcs}(A)$. Finally, $P_0.P_1$, the result of *timed prefixing* of P_0 to P_1, where $p \in P_0.P_1$ iff

$$p = \{((0, x), \mathbf{label}(x)) : x \in p_0\}$$

for some $p_0 \in P_0$ or

$$p = \{((0, x), \mathbf{label}(x)) : x \in p_0\} \cup \{((1, y), (m, r + d, s + d, t + d)) : y \in p_1\}$$

with $(m, r, s, t) = \mathbf{label}(y)$ and $d = max(\mathbf{cptime}(x) : x \in p_0) - min(\mathbf{stime}(y) : y \in p_1)$ for a maximal $p_0 \in P_0$ and some $p_1 \in P_1$, is a timed configuration structure in $\mathbf{tcs}(A)$. □

4.4. Example. For E' denoting the timed event structure as shown in fig. 15 the timed configuration structure $\mathbf{conf}(E')$ can be represented as $(P_0 \parallel P_1) \mid \mathbf{without}(\gamma^-, \gamma^+)$, where P_0, P_1, $P_0 \parallel P_1$ are as in fig. 15, γ^- and γ^+ are noninternal complementary actions with $\gamma^- \bullet \gamma^+ = \gamma$, and $\mathbf{without}(\gamma^-, \gamma^+)$ denotes the subuniverse of timed actions α with $\mathbf{ACTION}(\alpha)$ not in $\{\gamma^-, \gamma^+\}$. □

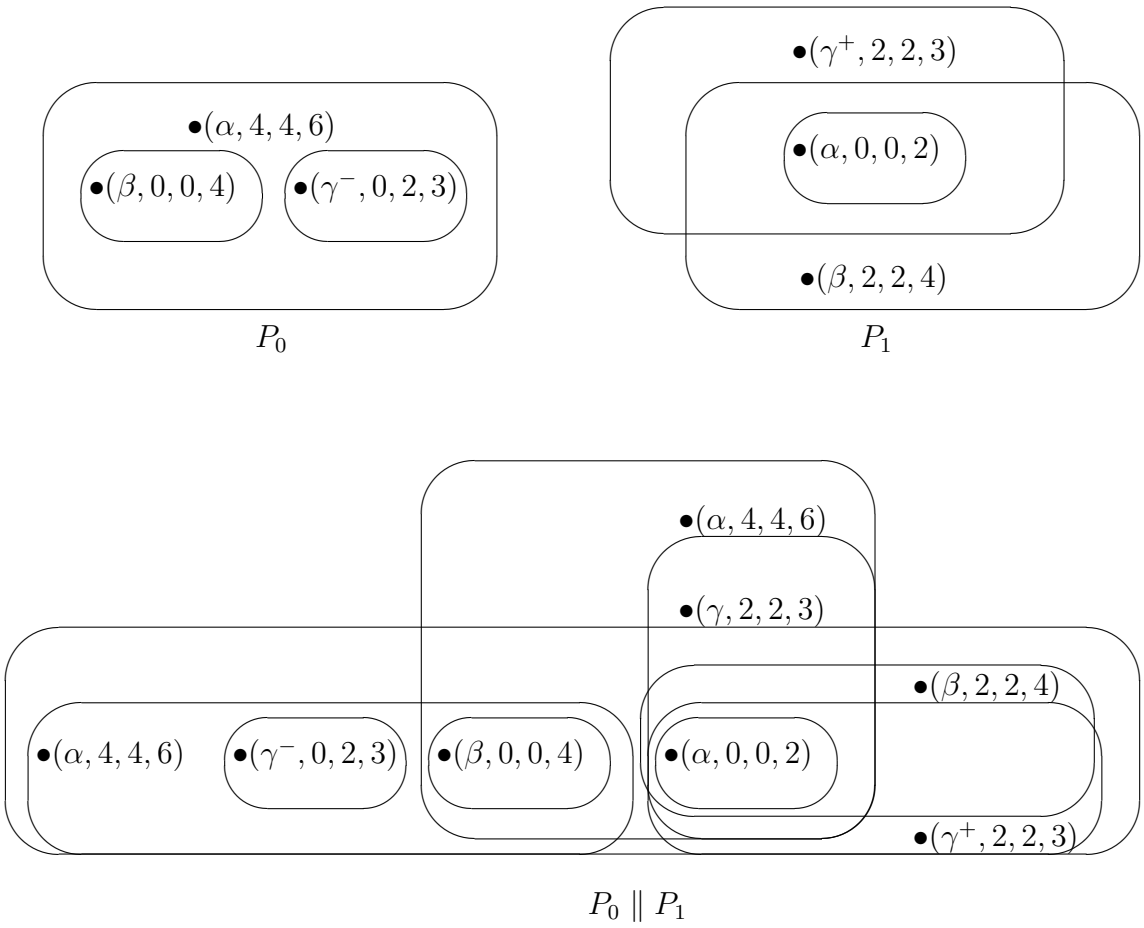

FIGURE 15: Composing timed configuration structures in parallel

4.5. Example. The timed behaviour which consists of choosing and executing a correspondingly delayed timed behaviour P if the choice is made with a delay not exceeding

Δ, or choosing and executing a correspondingly delayed timed behaviour Q if the choice is made with a delay greater than Δ can be defined as

$$\Sigma(P\mathbf{shift}_T : T \leq \Delta) + \Sigma(Q\mathbf{shift}_S : S > \Delta),$$

where $\Sigma(P_i : i \in I)$ denotes the sum of the family $(P_i : i \in I)$, a concept which can easily be obtained by generalizing the usual summation of timed configuration structures. $\Box$

The basic operations on labelled configuration structures constitute a basis for defining a broader class of operations. Such a class can be defined by introducing a complete partial order of configuration structures and considering the least solutions of fixed-point equations.

4.6. Definition. Given two arbitrary configuration structures P and Q, we say that P is a *prefix* of Q, written as $P \sqsubseteq Q$, if $P \subseteq Q$ and, for each $q \in Q$, $q \subseteq \bigcup P$ implies $q \in P$. $\Box$

Clearly, the relation of being a prefix is a partial order. It can be shown easily that this order is a chain-complete partial order.

4.7. Proposition. The prefix order in the universe of labelled configuration structures over a universe A of actions is a chain-complete partial order with **nil** playing the role of least element and the supremum of each countable chain $P_0 \sqsubseteq P_1 \sqsubseteq ...$ being the union $P = \bigcup(P_i : i \in \omega)$, written also as $\sqcup(P_i : i \in \omega)$, where ω denotes the set $\{0, 1, 2, ...\}$. If $P_0, P_1, ...$ are timed configuration structures then such a supremum is also a timed configuration structure. $\Box$

From the form of suprema of countable chains of labelled configuration structures we obtain easily the following property of basic operations on such structures.

4.8. Proposition. The basic operations on configuration structures in $\mathbf{lcs}(A)$ are continuous w.r. to the prefix order in the sense that they preserve the suprema of countable chains in the respective cartesian powers of $\mathbf{lcs}(A)$. $\Box$

Due to the well known properties of continuous operations in completely ordered sets, for continuous operations in $\mathbf{lcs}(A)$ we have the following result.

4.9. Proposition. Let $F : (\mathbf{lcs}(A))^{m+n} \longrightarrow (\mathbf{lcs}(A))^m$ be a continuous mapping which transforms each pair (P, Q) with $P \in (\mathbf{lcs}(A))^m$ and $Q \in (\mathbf{lcs}(A))^n$ into some $R = F(P, Q) \in (\mathbf{lcs}(A))^n$. Then we have:

(1) the fixed-point equation $P = F(P, Q)$ has a least solution, $\mathbf{fix}_P F(P, Q)$,

(2) this solution is given by $\sqcup(P_i : i \in \omega)$, where $P_0 = \mathbf{nil}^m$ and $P_{i+1} = F(P_i, Q)$ for $i \in \omega$,

(3) the correspondence $Q \longmapsto \mathbf{fix}_P F(P, Q)$ is a continuous mapping from $(\mathbf{lcs}(A)^n$ to $(\mathbf{lcs}(A)^m$.

A similar result holds true for timed configuration structures. We call the correspondence between F and $Q \longmapsto \mathbf{fix}_P F(P, Q)$ a *fixed-point operator*. □

The properties 4.8 and 4.9 and the fact that superpositions of continuous operations remain continuous allow us to obtain a broad class of continuous operations.

4.10. Proposition. The operations in $\mathbf{lcs}(A)$ which can be obtained by combining constant labelled configuration structures and basic operations with the aid of superpositions and fixed-point operators are continuous. We call them *definable* operations. □

For example, the operation $P \longmapsto P_0$ with a fixed $P_0 \in \mathbf{lcs}(A)$ is definable, the operation $Q \longmapsto \mathbf{fix}_P(P \parallel Q)$ is definable, etc.

The possibility of combining configuration structures allows one to define behaviours by describing how they consist of simpler components. In particular, this is true for timed behaviours. However, in the case of timed behaviours we are interested also in the characteristics which can be obtained due to timing, and we would like to find out such characteristics in a compositional way, that is by combining the characteristics of components and without the need of dealing with the behaviours themselves. Some

possibilities of this kind really exist due to the fact that the considered characteristics can be regarded as results of suitable operations on timed configuration structures, where the operations are related to the basic operations on timed configuration structures. The respective results are as follows.

4.11. Proposition. The correspondence $P \longmapsto \mathbf{eager}(P)$ is continuous. □

A proof is straightforward.

4.12. Proposition. Let $P_0, P_1 \in \mathbf{tcs}(A)$. Let K, h, K', h', d be as in 4.3. Then

(1) $\mathbf{eager}(P_0 \mid K') = \mathbf{eager}(P_0) \mid K'$,

(2) $\mathbf{eager}(P_0 h') = (\mathbf{eager}(P_0))h'$,

(3) $\mathbf{eager}(P_0\mathbf{shift}_d) = (\mathbf{eager}(P_0))\mathbf{shift}_d$,

(4) $\mathbf{eager}(P_0.P_1) = \mathbf{eager}(P_0).\mathbf{eager}(P_1)$,

(5) $\mathbf{eager}(P_0 + P_1) = \mathbf{eager}(P_0) + \mathbf{eager}(P_1)$,

(6) $\mathbf{eager}(P_0 \parallel P_1) = \mathbf{eager}(\mathbf{eager}(P_0) \parallel \mathbf{eager}(P_1))$. □

Proof: As the reasonning is similar in all the cases, we restrict ourselves to (6).

Suppose that $p \in \mathbf{eager}(P_0 \parallel P_1)$. Then p consists of $p_0 \in P_0$ and $p_1 \in P_1$ with an association c and it may be eager in $P_0 \parallel P_1$ only when p_0 is eager in P_0 and p_1 is eager in P_1. Hence $p \in \mathbf{eager}(P_0) \parallel \mathbf{eager}(P_1)$. Moreover, p must be eager in $\mathbf{eager}(P_0) \parallel \mathbf{eager}(P_1)$ since otherwise p could not be eager in $P_0 \parallel P_1$. Hence $\mathbf{eager}(P_0 \parallel P_1) \subseteq \mathbf{eager}(\mathbf{eager}(P_0) \parallel \mathbf{eager}(P_1))$. The converse inclusion is obvious. Q.E.D.

4.13. Proposition. The correspondence $P \longmapsto \mathbf{stamped}(P)$ is continuous. □

Proof: Let $P \sqsubseteq Q$ and let $H_P : P \to \mathbf{stamped}(P)$ and $H_Q : P \to \mathbf{stamped}(Q)$ be surjections as in 3.11.

Let $p = H_P(p')$ with $p' \in P$. Then we have $p' \in Q$ and the subconfigurations of p' in Q are exactly as in P. Hence $p = H_P(p') = H_Q(p')$. This implies $p \in \mathbf{stamped}(Q)$. Thus $\mathbf{stamped}(P) \subseteq \mathbf{stamped}(Q)$.

Let $q = H_Q(q')$ with $q \subseteq \bigcup \mathbf{stamped}(P)$. Then each $x \in q$ belongs to some $p = H_P(p')$ with $p' \in P$. Hence $x = \mathbf{std}_{q'}(y) \in H_P(p') = H_Q(p')$, that is $x = \mathbf{std}_{q'}(y)$ with $y \in p'$. As this holds for all $x \in q$, and thus for all $y \in q'$, we obtain $q' \subseteq \bigcup P$. Hence $q' \in P$, which implies $q = H_Q(q') = H_P(q') \in \mathbf{stamped}(P)$.

Consequently, $\mathbf{stamped}(P) \sqsubseteq \mathbf{stamped}(Q)$, that is the correspondence $P \longmapsto \mathbf{stamped}(P)$ is monotonic. Moreover, for each chain $P_0 \sqsubseteq P_1 \sqsubseteq ...$ we have $\sqcup(\mathbf{stamped}(P_i : i \in \omega) \sqsubseteq \mathbf{stamped}(\sqcup(P_i : i \in \omega))$ and each event of the right hand side can be found in some $\mathbf{stamped}(P_i)$. Consequently, the correspondence $P \longmapsto \mathbf{stamped}(P)$ is continuous. Q.E.D.

4.14. Proposition. For P_0, P_1, K', h', d as in 4.12 we have the following equivalences up to isomorphism and equalities:

(1) $\mathbf{stamped}(\mathbf{stamped}(P_0)) \approx \mathbf{stamped}(P_0)$,

(2) $\mathbf{stamped}(P_0 \mid K') = \mathbf{stamped}(P_0) \mid K'$,

(3) $\mathbf{stamped}(P_0 h') \approx (\mathbf{stamped}(P_0))h'$,

(4) $\mathbf{stamped}(P_0\mathbf{shift}_d) \approx (\mathbf{stamped}(P_0))\mathbf{shift}_d$,

(5) $\mathbf{stamped}(P_0.P_1) \approx \mathbf{stamped}(P_0).\mathbf{stamped}(P_1)$,

(6) $\mathbf{stamped}(P_0 + P_1) \approx \mathbf{stamped}(P_0) + \mathbf{stamped}(P_1)$,

(7) $\mathbf{stamped}(P_0 \parallel P_1) \approx \mathbf{stamped}(\mathbf{stamped}(P_0) \parallel \mathbf{stamped}(P_1))$. □

Proof: We restrict ourselves to (7). In this case the required isomorphism can be obtained from the following relation ϱ: $(\xi, \eta) \in \varrho$ iff $\xi = \mathbf{std}_p(x)$ for $x \in p$ and p consisting of $p_0 \in P_0$ and $p_1 \in P_1$ with an association c, and $\eta = \mathbf{std}_q(y)$ for $y \in q$ and q consisting of $q_0 \in \mathbf{stamped}(P_0)$ and $q_1 \in \mathbf{stamped}(P_1)$ with an association d, where $q_0 = \mathbf{std}_{p_0}(p_0)$,

$q_1 = \mathbf{std}_{p_1}(p_1)$, $d_0^{-1}(y) = \mathbf{std}_{p_0}(c_0^{-1}(x))$ whenever eiter side is defined, and $d_1^{-1}(y) = \mathbf{std}_{p_1}(c_1^{-1}(x))$ whenever either side is defined. A simple verification shows that ϱ defines really an isomorphism, as required. Q.E.D.

The correspondence $P \longmapsto \mathbf{fast}(P)$ is neither continuous nor even monotonic. In order to see this it suffices to consider the timed configuration structure in 3.15 (cf. figure 14) and its prefix obtained by removing the event of executing δ.

The only general properties of the correspondence $P \longmapsto \mathbf{fast}(P)$ we can prove are as follows.

4.15. Proposition. For P_0, P_1, K', h', d as in 4.12 and 4.14 we have the following relations:

(1) $\mathbf{fast}(\mathbf{fast}(P_0)) \approx \mathbf{fast}(P_0)$,

(2) $\mathbf{fast}(P_0) \mid K' \sqsubseteq \mathbf{fast}(P_0 \mid K')$,

(3) $\mathbf{fast}(P_0 h') \approx (\mathbf{fast}(P_0))h'$,

(4) $\mathbf{fast}(P_0\mathbf{shift}_d) \approx (\mathbf{fast}(P_0))\mathbf{shift}_d$,

(5) $\mathbf{fast}(P_0.P_1) \approx \mathbf{fast}(P_0).\mathbf{fast}(P_1)$ for P_0 of the form $\downarrow p_0$ with a finite $p_0 \in P_0$,

(6)
- $\mathbf{fast}(P_0 + P_1) \approx \mathbf{fast}(P_0)$ if

$$min(\mathbf{stime}(x) : x \in \bigcup P_0, \mathbf{action}(x) \text{ internal}) <$$
$$min(\mathbf{stime}(y) : y \in \bigcup P_1, \mathbf{action}(y) \text{ internal}),$$

- $\mathbf{fast}(P_0 + P_1) \approx \mathbf{fast}(P_1)$ if

$$min(\mathbf{stime}(x) : x \in \bigcup P_0, \mathbf{action}(x) \text{ internal}) >$$
$$min(\mathbf{stime}(y) : y \in \bigcup P_1, \mathbf{action}(y) \text{ internal}),$$

- $\mathbf{fast}(P_0 + P_1) \approx \mathbf{fast}(P_0) + \mathbf{fast}(P_1)$ if

$$min(\mathbf{stime}(x) : x \in \bigcup P_0, \mathbf{action}(x) \text{ internal}) =$$
$$min(\mathbf{stime}(y) : y \in \bigcup P_1, \mathbf{action}(y) \text{ internal}),$$

(7) $\mathbf{fast}(P_0 \parallel P_1)$ is isomorphic to a part of $\mathbf{fast}(\mathbf{fast}(P_0) \parallel \mathbf{fast}(P_1))$. □

Proof: Only (2) and (7) are not trivial. For (2) it suffices to notice that each $p \in \mathbf{stamped}(P) \mid K'$ which is fast in $\mathbf{stamped}(P)$ is fast also in $\mathbf{stamped}(P \mid K')$, and each $q \in \mathbf{fast}(P \mid K')$ which is contained in $p \in \mathbf{fast}(P) \mid K'$ must be fast in P. For (7) let us consider the restriction of the isomorphism ϱ from the proof of 4.14 to $\mathbf{fast}(P_0 \parallel P_1)$. Let $p' \in \mathbf{fast}(P_0 \parallel P_1)$ corresponds to p which consists of $p_0 \in P_0$ and $p_1 \in P_1$ with an association $\alpha \subseteq p_0 \times p_1$. If p'_0 which corresponds to p_0 is not in $\mathbf{fast}(P_0)$ then there exist $q_0 \in P_0$ and $x \in p_0$ and $y \in q_0$ such that $\mathbf{past}_{p_0}(x) = \mathbf{past}_{q_0}(y)$, y is an internal event, and $\mathbf{stime}(y) < \mathbf{stime}(x)$. By restricting α to the pairs (u, v) with $u \in \mathbf{past}_{p_0}(x) = \mathbf{past}_{q_0}(y)$ we obtain an association α' and $q \in P_0 \parallel P_1$ which consists of q_0 and p_1 with the association α' and has in $\mathbf{fast}(P_0 \parallel P_1)$ an image q'. But now q' and $x' \in p'$ and $y' \in q'$ which correspond respectively to x and y satisfy conditions which exclude $p' \in \mathbf{fast}(P_0 \parallel P_1)$. Thus each $p' \in \mathbf{fast}(P_0 \parallel P_1)$ consists of some $p'_0 \in \mathbf{fast}(P_0)$ and $p'_1 \in \mathbf{fast}(P_1)$, as required. Q.E.D.

The properties just described of the considered characteristics of timed behaviours allow a calculus for finding out such characteristics. In the case of sets of eager configurations results are unique. In other cases results can be obtained only up to isomorphism or they cannot be obtained in a direct way. The details will become clear in the next section, where we discuss a general concept of equivalence of timed configuration structures.

4.16. Example. For each timed behaviour Q which is eager in the sense that all its configurations are eager, i.e. $\mathbf{eager}(\mathrm{Q})=\mathrm{Q}$, we have

$$\mathbf{eager}(\mathbf{fix}_P(P \parallel Q)) = \mathbf{fix}_P(\mathbf{eager}(\mathbf{eager}(P) \parallel Q)).$$

Indeed, $\mathbf{fix}_P(\mathbf{eager}(P \parallel Q))$ is the supremum of the chain $P_0 \sqsubseteq P_1 \sqsubseteq ...$, where $P_0 = \mathbf{nil}$, $P_1 = \mathbf{eager}(\mathbf{nil} \parallel Q)$, $P_2 = \mathbf{eager}(\mathbf{eager}(\mathbf{nil} \parallel Q) \parallel Q) = \mathbf{eager}((\mathbf{nil} \parallel Q) \parallel Q)$, etc. □

In general, proceeding as in this example we obtain the following property.

4.17. Proposition. If $(P, Q) \longmapsto F(P, Q)$ is a definable operation on labelled config-

uration structures such that $\mathbf{eager}F(P,Q) = \mathbf{eager}F(\mathbf{eager}(P),\mathbf{eager}(Q))$ for all P, Q then $\mathbf{eager}(\mathbf{fix}_P F(P,Q)) = \mathbf{fix}_P(\mathbf{eager}(F(\mathbf{eager}(P),\mathbf{eager}(Q))))$ for all Q. □

5 Equivalence

For labelled event and configuration structures we have a number of equivalences which reflect some similarities of the represented behaviours (cf. [2]). All of them can easily be adapted for labelled configuration structures. Some of them are congruences for the considered operations on labelled event structures. We adapt one of such congruences for timed configuration structures. It is a variant of the so called history preserving equivalence (cf. [2] and [7]). As the internal events of timed behaviours have some effect on characteristics of such behaviours, we choose the strongest variant in which internal events are not neglected. Our definition of the history preserving equivalence is based on a concept of simulation.

5.1. Definition. A (*history preserving*) *simulation* of a labelled configuration structure P in a labelled configuration structure Q is a family $\varrho = (\varrho_{pq} : p \in P, q \in Q)$ where:

(1) each ϱ_{pq} is a set of isomorphisms $\varphi :\downarrow p \longrightarrow\downarrow q$,

(2) $\varrho_{\emptyset\emptyset}$ is the one-element set consisting of the empty isomorphism, $\emptyset : \{\emptyset\} \longrightarrow \{\emptyset\}$,

(3) for each $\varphi \in \varrho_{pq}$ and each $p' \in P$ such that $p \subseteq p'$ there exists $q' \in Q$ such that $q \subseteq q'$ and φ has an extension φ' in $\varrho_{p'q'}$.

We write such a simulation as $\varrho : P \longrightarrow Q$ or $P \xrightarrow{\varrho} Q$. If $\hat{\varrho} = (\hat{\varrho}_{qp} : q \in Q, p \in P)$, where $\hat{\varrho}_{qp} = \varrho_{pq}$, is also a simulation then we say that $\varrho : P \longrightarrow Q$ is a *bisimulation*. □

5.2. Example. Each isomorphism f from a labelled configuration structure P to a labelled configuration structure Q defines a bisimulation $\bar{f} : P \longrightarrow Q$, where $\bar{f}_{pq}$ is the one-element set consisting of the restriction of f to p for $q = f(p)$ and the empty set for $q \neq f(p)$. □

5.3. Example. For each labelled configuration structure P there exists a bisimulation

ϱ of P in the prime configuration structure $Q = \mathbf{prime}(P)(= \mathbf{rooted}(P))$. It suffices to consider the bijections described in 2.13 and define ϱ_{pq} as the one-element set consisting of b_{pq} for $(p,q) \in B$ and as the empty set for (p,q) not in B. $\Box$

5.4. Example. For each two labelled configuration structures P and Q such that $P \sqsubseteq Q$ there exists a simulation $\mathbf{in} : P \longrightarrow Q$, where each $\mathbf{in}_{pq}$ is the one-element set consisting of the identity mapping on p for $p = q$ and the empty set for $p \neq q$. $\Box$

The general definition of simulations and bisimulations is formulated such that it applies without any modification to timed configuration structures.

The following property of simulations allows one to compose them.

5.5. Proposition. If $\varrho = (\varrho_{pq} : p \in P, q \in Q)$ is a simulation of P in Q and $\sigma = (\sigma_{qr} : q \in Q, r \in R)$ is a simulation of Q in R then $\varrho\sigma = (\bigcup(\varrho_{pq}\sigma_{qr} : q \in Q) : p \in P, r \in R)$, where $\varrho_{pq}\sigma_{qr} = \{\varphi\psi : \varphi \in \varrho_{pq}, \psi \in \sigma_{qr}\}$ and $\varphi\psi$ is defined by $\varphi\psi(x) = \psi(\varphi(x))$, is a simulation of P in R. If ϱ and σ are bisimulations then so is $\varrho\sigma$. $\Box$

Proof: It is clear that $\varrho\sigma$ satisfies (1) and (2) of 5.1. For (3) suppose that $\chi \in (\varrho\sigma)_{pr}$ and that p is contained in $p' \in P$. Then $\chi = \varphi\psi$ for some $\varphi \in \varrho_{pq}$ and $\psi \in \sigma_{qr}$, and we have an extension φ' of φ in $\varrho_{p'q'}$ and an extension ψ' of ψ in $\sigma_{q'r'}$. Consequently, r is contained in some $r' \in R$ and we have an extension of χ in $(\varrho\sigma)_{p'r'}$, namely $\varphi'\psi'$. Q.E.D.

This proposition allows to define the history preserving equivalence.

5.6. Proposition. The following relation between labelled configuration systems is an equivalence:

$$P \sim Q \text{ iff there exists a bisimulation } \varrho : P \longrightarrow Q$$

We call it the *history preserving equivalence.* $\Box$

The possibility of composing simulations as described in 5.5 leads to a fact which plays an important role in studying the relation between the history preserving equivalence and definable operations on labelled configuration structures.

5.7. Proposition. The labelled configuration structures over a universe A of actions and their simulations constitute a category $\mathbf{LCS}(A)$. The timed configuration structures over A constitute a full subcategory $\mathbf{TCS}(A)$ of $\mathbf{LCS}(\mathbf{ta}(A))$. The category $\mathbf{LCS}(A)$ has colimits of countable chains. The colimit of each chain $P_0 \sqsubseteq P_1 \sqsubseteq ...$ coincides with its supremum $P = \sqcup(P_i : i \in \omega) = \bigcup(P_i : i \in \omega)$. Similarly for the cartesian powers of $\mathbf{LCS}(A)$. Moreover, for each commutative diagram as in fig. 16 with a unique simulation $\varrho : P \longrightarrow Q$ resulting from the universal properties of colimits, this unique simulation is determined by the property: $\varrho_{pq} = (\varrho_i)_{pq}$ for all $i \in \omega$ such that $p \in P_i$. $\Box$

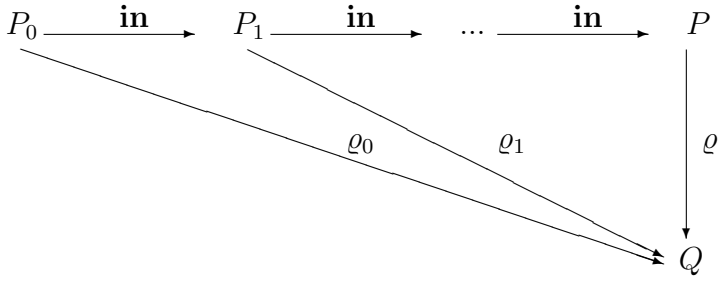

FIGURE 16: A colimit diagram

Proof: It suffices to prove the second part of the proposition. To this end it suffices to notice that the commutativity of the diagram in fig. 16 means that each $\varrho_i = ((\varrho_i)_{pq} : p \in P_i, q \in Q)$ is the restriction of $\varrho_{i+1} = ((\varrho_{i+1})_{pq} : p \in P_{i+1}, q \in Q)$, $\varrho_{i+2} = ((\varrho_{i+2})_{pq} : p \in P_{i+2}, q \in Q)$,..., $\varrho = (\varrho_{pq} : p \in P, q \in Q)$ to $p \in P_i$. Q.E.D.

The relation between the history preserving equivalence and the definable operations on labelled configuration structures can be studied with the aid of special functors between cartesian powers of the category $\mathbf{LCS}(A)$.

5.8. Definition. Let $F : (\mathbf{LCS}(A))^m \longrightarrow (\mathbf{LCS}(A))^n$ be a functor. We say that F in *continuous* if it preserves colimits. We say that F *preserves the prefix order* if, for all P, Q, $P \sqsubseteq Q$ implies $F(P) \sqsubseteq F(Q)$ and the coincidence of $F(\mathbf{in}) : F(P) \longrightarrow F(Q)$ with $\mathbf{in} : F(P) \longrightarrow F(Q)$. Finally, we say that F *preserves bisimulations* if $F(\varrho) : F(P) \longrightarrow F(Q)$ is a bisimulation whenever $\varrho : P \longrightarrow Q$ is a bisimulation. $\Box$

The following property of definable operations on labelled configuration structures is crucial for their effect on the history preserving equivalence.

5.9. Proposition. Each definable operation on labelled configuration structures from $\mathbf{lcs}(A)$ can be extended in a canonical way to a continuous functor which preserves the prefix order and bisimulations. □

Proof: For basic operations the proof is straightforward. For example, for $\varrho_0 : P_0 \longrightarrow Q_0$ and $\varrho_1 : P_1 \longrightarrow Q_1$ we consider $p \in P_0 \parallel P_1$ which consists of $p_0 \in P_0$ and $p_1 \in P_1$ with an association c, $q \in Q_0 \parallel Q_1$ which consists of $q_0 \in Q_0$ and $q_1 \in Q_1$ with an association d, $\varphi_0 \in (\varrho_0)_{p_0 q_0}$, $\varphi_1 \in (\varrho_1)_{p_1 q_1}$, and we define $(\varrho_0 \parallel \varrho_1)_{pq}$ as the set of bijections $\varphi : p \longrightarrow q$ such that $\varphi(c_0(x)) = d_0(\varphi_0(x))$ for $x \in p_0$ and $\varphi(c_1(y)) = d_1(\varphi_1(y))$ for $y \in p_1$. In this way we obtain a functor whose continuity and other required properties follow easily from 4.8 and 5.7.

In order to extend the proof on all definable operations it suffices to consider a continuous functor $F : (\mathbf{LCS}(A))^{m+n} \longrightarrow (\mathbf{LCS}(A))^m$ which preserves the prefix order and bisimulations and to prove that the operation $f : Q \longmapsto \mathbf{fix}_P F(P, Q)$ extends to a continuous functor which preserves the prefix order and bisimulations.

Suppose that $\varrho : Q \longrightarrow Q'$ is a simulation and consider the least solutions $f(Q)$ and $f(Q')$ of the respective fixed-point equations $P = F(P, Q)$ and $P = F(P, Q')$. As F is continuous, we obtain the commutative diagram in fig. 17 with a unique simulation $\sigma : f(Q) \longrightarrow f(Q')$. From the uniqueness of σ and 5.7 it follows that the correspondence $\varrho \longmapsto \sigma$ defines a functor f, where $f(\varrho)$ is the least common extension of the families defined inductively by the formulas $\varrho_0 = \mathbf{1}_{\mathbf{nil}^m}$ and $\varrho_{i+1} = F(\varrho_i, \varrho)$. It is easy to verify that this functor is continuous and that it preserves the prefix order and bisimulations. Q.E.D.

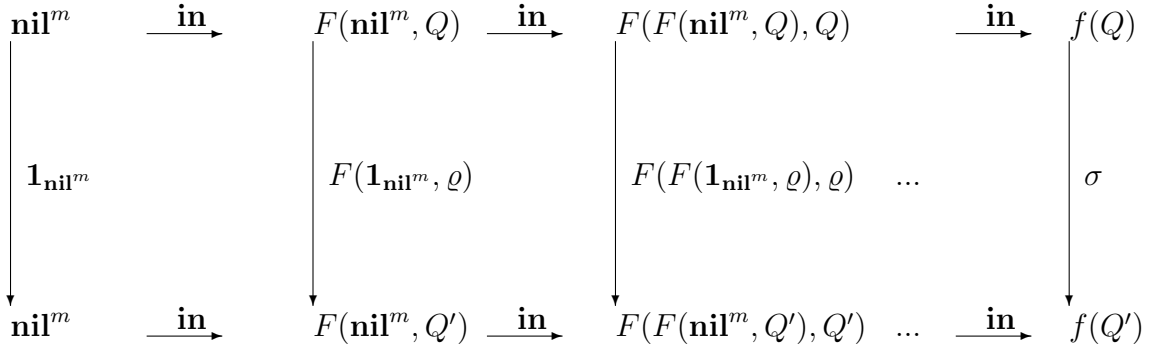

FIGURE 17: The effect of fixed-point operators on simulations

From 5.9 and the definition of the history preserving equivalence we obtain the needed result.

5.10. Proposition. The history preserving equivalence is a congruence for all definable operations on labelled configuration structures. □

The concept of equivalence of labelled configuration structures extends in a natural way to a concept of equivalence of operations on such structures.

5.11. Definition. Functors $F : (\mathbf{LCS}(A))^m \longrightarrow (\mathbf{LCS}(A))^n$ and $G : (\mathbf{LCS}(A))^m \longrightarrow (\mathbf{LCS}(A))^n$ are said to be *equivalent* (with respect to the history preserving equivalence) if there exists a natural transformation $\varrho : F \longrightarrow G$ which consists of bisimulations, that is a family $\varrho = (\varrho(P) : F(P) \longrightarrow G(P) : P \in (\mathbf{LCS}(A))^m)$ of bisimulations such that, for each simulation $\sigma : P \longrightarrow Q$, a diagram as in fig. 18 commutes. Two definable operations on labelled configuration structures are said to be *equivalent* if their canonical extensions to functors which preserve the prefix order and bisimulations are equivalent. □

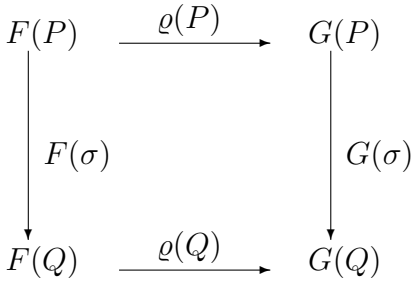

FIGURE 18: A commutative diagram for a natural transformation

5.12. Example. The operations $P \longmapsto P$ and $P \longmapsto P + P$ are equivalent with the equivalence given by the family $\beta = (\beta(P) : P \longrightarrow P + P : P \in \mathbf{LCS}(A)))$ of bisimulations, where $(\beta(P))_{pq}$ is the one-element set consisting of the bijection $\varphi : p \longrightarrow q$ defined by $\varphi(x) = ((0, x), \mathbf{label}(x))$ for all $p \in P$ and $q \in P + P$ such that $q = \varphi(p)$, $(\beta(P))_{pq}$ is the one-element set consisting of the bijection $\psi : p \longrightarrow q$ defined by $\psi(x) = ((1, y), \mathbf{label}(y))$ for all $p \in P$ and $q \in P + P$ such that $q = \psi(p)$, and $(\beta(P))_{pq}$ is the empty set in all the remaining cases. Similarly, the operations $(P, Q) \longmapsto P + Q$ and $(P, Q) \longmapsto Q + P$ are equivalent, the operations $(P, Q, R) \longmapsto (P + Q) + R$ and $(P, Q, R) \longmapsto P + (Q + R)$ are equivalent, the operations $(P, Q) \longmapsto (P \parallel Q)$ and $(P, Q) \longmapsto Q \parallel P$ are equivalent, and the operations $(P, Q, R) \longmapsto (P \parallel Q) \parallel R$ and $(P, Q, R) \longmapsto P \parallel (Q \parallel R)$ are equivalent. □

For the equivalence of definable operations on labelled configuration structures we have the following result.

5.13. Proposition. If two definable operations on labelled configuration structures are constructed in the same manner from equivalent definable operations then they are equivalent. □

Proof: The only nontrivial part of the proof is that about operations defined by fixed-point equations. Thus it suffices to consider two continuous, the prefix order and bisimula-

tions preserving functors $F : (\mathbf{LCS}(A))^{m+n} \longrightarrow (\mathbf{LCS}(A))^m$ and $G : (\mathbf{LCS}(A))^{m+n} \longrightarrow (\mathbf{LCS}(A))^m$ which are equivalent with the equivalence given by a family ϱ of bisimulations $\varrho(P,Q) : F(P,Q) \longrightarrow G(P,Q)$, and find a suitable family of bisimulations $\sigma(Q) : f(Q) \longrightarrow g(Q)$ for $f(Q) = \mathbf{fix}_P F(P,Q)$ and $g(Q) = \mathbf{fix}_P G(P,Q)$. To this end we consider the diagram in fig. 19, which is commutative due to the continuity and the other properties of F and G and due to the fact that ϱ is a natural transformation from F to G. By 5.7 we obtain a unique bisimulation $\sigma(Q) : f(Q) \longrightarrow g(Q)$ which completes this diagram to a commutative one. From the uniqueness of $\sigma(Q)$ we obtain that the family $\sigma = (\sigma(Q) : f(Q) \longrightarrow g(Q) : Q \in (\mathbf{LCS}(A))^n)$ is a natural transformation as required. Q.E.D.

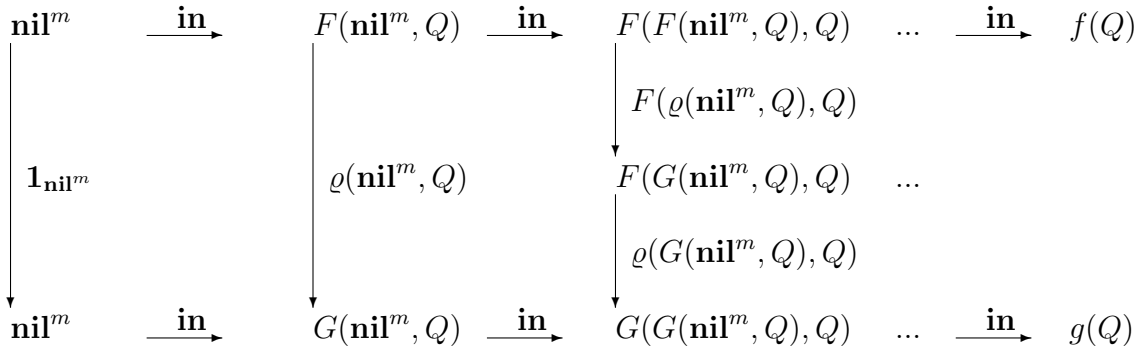

FIGURE 19: The effect of fixed-point operators on the equivalence of operations

For example, the operations $Q \longmapsto \mathbf{fix}_P(P \parallel Q)$ and $Q \longmapsto \mathbf{fix}_P(Q \parallel P)$ are equivalent.

The results about equivalence of labelled configuration structures and operations on such structures apply to timed configuration structures. In particular, together with the results of section 4 about compositionality and continuity of characteristics of timed configuration structures (cf. 4.11, 4.12, 4.13, 4.14, and 4.15), they give us some possibilities of finding out characteristics of timed behaviours. This is due to the fact that by slightly modifying the concepts of basic and definable operations we can adapt 5.9 and 5.13 to the

case of timed configuration structures. The modifications and the respective properties are as follows.

5.14. Definition. The operations $P_0 \longmapsto P_0 \mid K'$, $P_0 \longmapsto P_0 h'$, $P_0 \longmapsto P_0\mathbf{shift}_d$, $P_1 \longmapsto P_0.P_1$, $(P_0, P_1) \longmapsto P_0 + P_1$, $(P_0, P_1) \longmapsto P_0 \parallel P_1$, as defined in 4.3, and $P_0 \longmapsto \mathbf{eager}(P_0)$, $P_0 \longmapsto \mathbf{stamped}(P_0)$, are called *basic* operations on timed configuration structures. The operations in $\mathbf{tcs}(A)$ which can be obtained by combining constant timed configuration structures and basic operations on timed configuration structures with the aid of superpositions and fixed-point operators are called *definable* operations on timed configuration structures. □

Noting that all definable operations on timed configuration structures are continuous and proceeding as for labelled structures we obtain the following propositions.

5.15. Proposition. Each definable operation on timed configuration structures can be extended in a canonical way to a continuous functor which preserves the prefix order and bisimulations. □

5.16. Proposition. The history preserving equivalence is a congruence for all definable operations on timed configuration structures. □

5.17. Proposition. If two definable operations on timed configuration structures are constructed in the same manner from equivalent definable operations on timed configuration structures then they are equivalent. □

For example, the operations

$$Q \longmapsto \mathbf{fix}_P(\mathbf{stamped}(P \parallel Q))$$

and

$$Q \longmapsto \mathbf{fix}_P(\mathbf{stamped}(\mathbf{stamped}(P) \parallel \mathbf{stamped}(Q)))$$

are equivalent (cf. 4.14).

Proceeding as in the proof of 5.13 we obtain a general result allowing often to find

out the set of stamped configurations for timed configuration structures defined with the aid of fixed-point operators.

5.18.Proposition. If

$$(P, Q) \longmapsto \mathbf{stamped}(F(P, Q))$$

and

$$(P, Q) \longmapsto \mathbf{stamped}(F(\mathbf{stamped}(P), \mathbf{stamped}(Q)))$$

are two equivalent operations on timed configuration structures then the operations

$$Q \longmapsto \mathbf{stamped}(\mathbf{fix}_P F(P, Q))$$

and

$$Q \longmapsto \mathbf{fix}_P(\mathbf{stamped}(F(\mathbf{stamped}(P), \mathbf{stamped}(Q))))$$

are equivalent. □

For example, the operations

$$Q \longmapsto \mathbf{stamped}(\mathbf{fix}_P(P \parallel Q))$$

and

$$Q \longmapsto \mathbf{fix}_P(\mathbf{stamped}(\mathbf{stamped}(P) \parallel \mathbf{stamped}(Q)))$$

are equivalent.

We recall that similar but stronger properties of this kind have been obtained for the sets of eager configurations of timed configuration structures (cf. 4.16 and 4.17). Unfortunately, for the sets of fast configurations it is impossible to obtain such a result because of the lack of continuity of the correspondence $P \longmapsto \mathbf{fast}(P)$. However, it is easy to verify that the equivalence of timed configuration structures implies the equivalence of the corresponding sets of fast configurations.

6 Conclusions

The presented concepts and facts may be summarized as follows:

- the progress of a behaviour in time can be described by extending a given description of causality and branching of this behaviour by some extra information about how events of executing actions occur in time,

- a description in the form of a labelled configuration structure can be obtained by endowing events with the possible enabling times, start times, and execution times, and by considering the events thus endowed as executions of special timed actions,

- the respective labelled configuration structure, called a timed configuration structure, reflects the properties of a behaviour, including the existing causal independence of events, waiting, observable existence of configurations, and the possible influence of time on branching,

- important characteristics of behaviours can be represented by timed configuration structures formed of some particular configurations like the ones without unnecessary delays of enabled events, stamped configurations representing chronicles and the particular stamped configurations which are reached due to their sufficiently early enabling,

- the timed configuration structures representing behaviours and their characteristics can be defined in a modular way, that is by combining timed configuration structures representing simpler behaviours and their characteristics,

- except for several particular cases, the characteristics of composed behaviours can be obtained by combining only the characteristics of component behaviours and without a need of considering all information about the respective behaviours.

References

[1] P. DEGANO, R. DE NICOLA, U. MONTANARI, On the consistency of "truly concurrent" operational and denotational semantics, Proc. of the Symposium on Logic in Computer Science, Edinburgh, 1988, pp.133-141

[2] R. VAN GLABBEEK, U. GOLTZ, Equivalence notions for concurrent systems and refinement (extended abstract), Proc. of MFCS'89, Springer LNCS **379**, 1989, pp.237-248

[3] D. V. J. MURPHY, Approaching a Real-Time Concurrency Theory, Proc. of the International BCS-FACS Workshop "Semantics for Concurrency", University of Leicester, July 1990, M. Z. Kwiatkowska, M. W. Shields, R. M. Thomas, eds., Springer 1990, pp.295-310

[4] R. MILNER, A Calculus of Communicating Systems, Springer LNCS **92**, 1980

[5] A. MAGGIOLO-SCHETTINI, J. WINKOWSKI, Towards an Algebra for Timed Behaviours, Report **677**, Inst. of Computer Sc. of the Polish Academy of Sciences, January 1990, and *Theoret. Comput. Sci.* **103**, 1992, pp.335-363

[6] M. NIELSEN, G. D. PLOTKIN, G. WINSKEL, Petri Nets, Event Structures and Domains, Part I. *Theoret. Comput. Sci.* **13**, 1981, pp.85-108

[7] A. RABINOVICH, B. A. TRAKHTENBROT, Behaviour structure and nets, *Fundamenta Informaticae* **11**(4), 1988, pp.357-404

[8] J. WINKOWSKI, On some time-based characteristics of behaviours, Proc. of the 3rd Workshop on Concurrency and Compositionality, Goslar, March 5-8, 1991, E. Best, G. Rozenberg, eds., GMD-Studien Nr. **191**, May 1991, pp.222-224

[9] G. WINSKEL, Event Structure Semantics for CCS and Related Languages, Springer LNCS **140**, 1982, pp.561-576

[10] G. WINSKEL, Event Structure Semantics for CCS and Related Languages, Aarhus University, DAIMI PB-**159**, April 1983

[11] G. WINSKEL, Event Structures, Springer LNCS **255**, 1986, pp.325-392