Reversibility and Models for Concurrency

Iain Phillips[1]

Department of Computing
Imperial College London
180 Queen's Gate, London, SW7 2AZ, United Kingdom

Irek Ulidowski[2]

Department of Computer Science
University of Leicester
University Road, Leicester, LE1 7RH, United Kingdom

Abstract

There is a growing interest in models of reversible computation driven by exciting application areas such as bio-systems and quantum computing. Reversible process algebras RCCS [2] and CCSK [8] were developed and general techniques for reversing other process operators were proposed. The paper shows that the notion of reversibility can bridge the gap between some interleaving models and non-interleaving models of concurrency, and makes them interchangeable. We prove that transition systems associated with reversible process algebras are equivalent as models to labelled prime event structures. Furthermore, we show that forward-reverse bisimulation corresponds to hereditary history-preserving bisimulation in the setting with no auto-concurrency and no auto-causation.

Keywords: Reversible computation, labelled transition systems, prime event structures, hereditary history-preserving bisimulation

1 Introduction

CCS with Communication Keys (CCSK) [8] is a reversible version of CCS which can be used to model and analyse bidirectional behaviour of systems, for example the binding and unbinding of molecules in biochemical reactions. The definition of CCSK is given in the *Structural Operational Semantics* (SOS) style, and the SOS approach is also employed to give a procedure for converting operators of other process algebras into reversible operators [8]. The main idea is that dynamic operators (which can be destroyed in the course of a transition) are converted by the procedure into static operators (which are preserved), using new auxiliary operators. For instance, both sides of a choice $P+Q$ are retained. As a result process terms in the

[1] Email: iccp@doc.ic.ac.uk
[2] Email: iu3@mcs.le.ac.uk

This is a preliminary version. The final version will be published in Electronic Notes in Theoretical Computer Science

new reversible language do not change their overall structure during computation. A crucial component of the procedure is the notion of *communication keys*. These are unique identifiers that are assigned to, and are recorded in the syntax of, past actions and are vital for the communication mechanism to work correctly in the forward and reverse directions. In contrast, in RCCS [2] past behaviour, including communication, is stored on external devices such as memories.

A process algebra of reversible operators produced by the procedure from [8] gives rise to a forward labelled transition relation (ltr) $\rightarrow$ and a reverse ltr $\rightsquigarrow$, which is the inverse of $\rightarrow$. These ltrs enjoy a number of intuitive properties that arise directly in the presence of reversibility. Some of the properties, for example the Reverse Diamond property (see Definition 2.3), can be thought of as intrinsic properties of reversible transition systems. Other properties, for example the Non-repeating property (see Section 2.1), are more specific and a consequence of using the communication keys mechanism. In this paper we investigate properties of this sort and propose an abstract definition of reversible transition systems by identifying several properties that must be satisfied. We find that reversible transition systems are rich enough to express such true concurrency (non-interleaving) notions as events, concurrency, causality and conflict. Consequently, we show that reversible transition systems are equivalent as models to prime event structures. Hence, we can study and work with true concurrency semantics within the purely interleaving setting of reversible transition systems.

This result leads naturally to questions regarding the distinguishing power of the standard interleaving equivalences defined over reversible transition systems. For instance: What true concurrency bisimulation is the *forward-reverse (FR) bisimulation* relation [8] "closest" to? FR bisimulation, a natural notion of bisimulation over reversible transition systems, is clearly finer than the standard bisimulation as it distinguishes between $a \mid b$ and $ab + ba$. It is also finer than the two versions of *history-preserving bisimulation* as given in [5] since it does not satisfy the absorption law: $(a \mid (b+c)) + (a \mid b) + ((a+c) \mid b) = (a \mid (b+c)) + ((a+c) \mid b)$. We show that FR bisimulation coincides with *hereditary history-preserving (HHP) bisimulation*, which is regarded as the canonical true concurrency equivalence [1,5,7,3]. The result holds for reversible transition systems with no auto-concurrency and with no auto-causation, and since CCSK gives rise to such transition systems the result holds for CCSK.

2 Prime ltrs and prime graphs

We propose a class of transition systems that are equivalent to labelled prime event structures. Previously, Sassone, Nielsen and Winskel [9] introduced *occurrence transition systems with independence relation* (oTSI). The independence relation is defined on transitions by giving the conditions it satisfies; it says which transitions are concurrent. It is rich enough to express the causality and conflict relations of event structures. Sassone *et al.* show that prime event structures with binary conflict are equivalent to oTSIs which satisfy an extra property (E) [9]. Independently, van Glabbeek defines *history-preserving* process graphs [4] by setting a number of properties relating to concurrent history of processes that must be satisfied. These

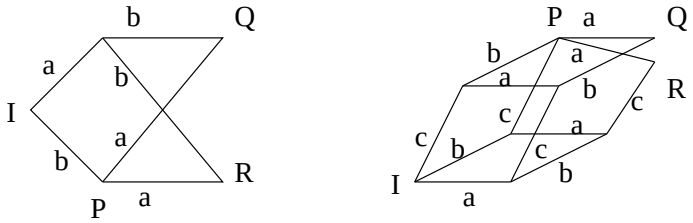


Fig. 1.

graphs are as expressive as event structures. In this work we follow van Glabbeek's approach and do not use independence relations.

In the literature, a labelled transition system, or lts for short, may or may not have an initial state. Let us deem an ltr to be a structure that specifies a transition relation with no initial state and an lts to be an ltr with an initial state. Thus an lts can be seen as a rooted ltr.

Definition 2.1 A *labelled transition relation* is a structure $(\mathsf{Proc}, \mathsf{Lab}, \rightarrow)$, where Proc is the set of processes, Lab is the set of action labels and $\rightarrow \subseteq \mathsf{Proc} \times \mathsf{Lab} \times \mathsf{Proc}$ is a *transition relation*. Given $\rightarrow$, a *reverse transition relation* $\rightsquigarrow$ is the inverse of $\rightarrow$. A *labelled transition system* is a structure $(\mathsf{Proc}, \mathsf{Lab}, \rightarrow, I)$ where $I \in \mathsf{Proc}$ is the initial state.

We let $P, Q, S, T, \ldots$ be typical processes and a, b, c, d be typical action labels. We shall keep Lab fixed throughout the paper.

Definition 2.2 [6] Let $(\mathsf{Proc}, \mathsf{Lab}, \rightarrow)$ be an ltr. Let $\sim$ be the smallest equivalence relation satisfying: if $P \xrightarrow{a} Q \xrightarrow{b} S$ and $P \xrightarrow{b} R \xrightarrow{a} S$ and $Q \neq R$ then $(P, a, Q) \sim (R, a, S)$. The equivalence classes, written $[P, a, Q]$, are the *events*. Define a labelling function ℓ from $\rightarrow / \sim$ to Lab by setting $\ell([P, a, Q]) = a$.

Definition 2.3 A *prime ltr* is an ltr $(\mathsf{Proc}, \mathsf{Lab}, \rightarrow)$ satisfying the following:

- **WF** (well-founded) there is no infinite reverse computation;
- **UT** (unique transition) if $P \xrightarrow{a} Q$ and $P \xrightarrow{b} Q$ then $a = b$;
- **ED** (event-deterministic [9,4]) if $P \xrightarrow{a} Q$ and $P \xrightarrow{a} R$, and $(P, a, Q) \sim (P, a, R)$, then $Q = R$;
- **RD** (reverse diamond) if $Q \xrightarrow{a} P$, $R \xrightarrow{b} P$ and $Q \neq R$, then there is S such that $S \xrightarrow{a} R$, $S \xrightarrow{b} Q$;
- **FD** (forward diamond) if $P \xrightarrow{a} Q \rightarrow^* T$, $P \xrightarrow{b} R \rightarrow^* T$, and $Q \neq R$, then there is S such that $Q \xrightarrow{b} S$, $R \xrightarrow{a} S$ and $S \rightarrow^* T$.

It can be checked that all five properties are independent of each other. The most interesting cases are showing that ED and FD are not derivable from the others: see Figures 1 and 2 respectively. Note that the transitions in all figures ought to be read from left to right when the arrowheads are not displayed. In Figure 1 we see that in each graph $P \xrightarrow{a} Q$, $P \xrightarrow{a} R$ and the two transitions represent the same event. Note that reverse ED can be derived from RD and (forward) ED.

We now show that the ltr of [8] which we briefly outlined in the introduction

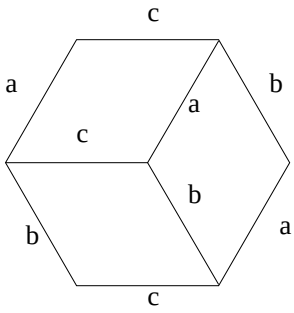


Fig. 2.

is an example of a prime ltr. Unfortunately space does not permit us to give the fairly lengthy definition of the ltr here.

Proposition 2.4 *The ltr of [8] is a prime ltr.*

Proof. This is shown in [8] for RD and FD. WF is clear. For UT: it is clear that if $P \xrightarrow{a} Q$ and $P \xrightarrow{b} Q$ then the transitions have the same key.

For ED, note that terms do not change structure during a computation, except that some static operators f become auxiliary operators f_r. So any $f_r[m]$ must have been created in exactly the same position by two transitions on opposite sides of a diamond. This must also hold for $\sim$. Hence the result. □

Next, we give several definitions and results that will be crucial in proving our main results.

Definition 2.5 Let $(\mathsf{Proc}, \mathsf{Lab}, \rightarrow)$ be an ltr. A *path* from P to Q is a finite forward computation from P to Q.

Definition 2.6 [4, Definition 3.1] Say that two paths in a ltr are *adjacent* if one can be got from the other by replacing some segment $P \xrightarrow{a} R \xrightarrow{b} Q$ by $P \xrightarrow{b} S \xrightarrow{a} Q$. Say that two paths are *homotopic* if they are related by the reflexive and transitive closure of adjacency.

Note that we can assume that $R \neq S$ in the definition of adjacency when dealing with prime ltrs, by UT. So homotopic paths have the same events.

Let $(\mathsf{Proc}, \mathsf{Lab}, \rightarrow)$ be a prime ltr. Then the irreversible processes are $\mathsf{Irr} = \{P \in \mathsf{Proc} : P \not\rightarrow\}$; let $\mathsf{der}(P) = \{Q \in \mathsf{Proc} : P \rightarrow^* Q\}$.

Lemma 2.7 *Suppose that* $(\mathsf{Proc}, \mathsf{Lab}, \rightarrow)$ *is a prime ltr. Suppose that* $P \in \mathsf{Proc}$, $Q, R \in \mathsf{Irr}$ *and* s, t *are paths from* Q, R *to* P. *Then* $Q = R$ *and* s, t *are homotopic.*

Proof. By RD and UT. □

Proposition 2.8 *Any two paths in a prime ltr with the same endpoints are homotopic.*

Proof. ¿From Lemma 2.7 by extending backwards to an irreversible process, using WF. □

Proposition 2.8 tells us that any two paths with the same endpoints have the same multiset of events. We want to show that any path cannot have repeated

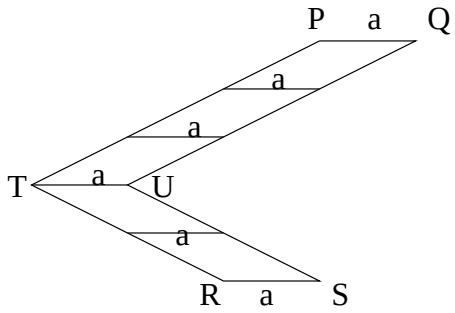


Fig. 3.

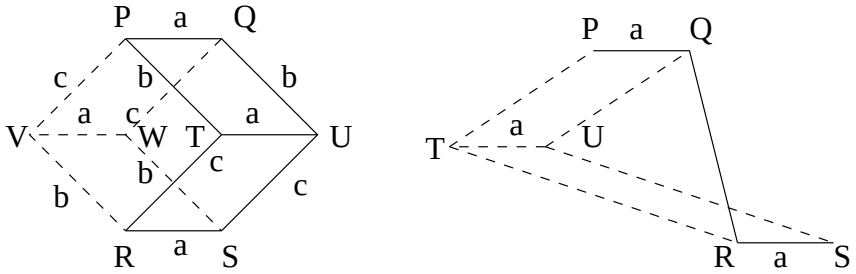


Fig. 4.

events. If this is the case then any $P \in \mathsf{Proc}$ is associated with a well-defined set of events, namely the set of events in any path from some $Q \in \mathsf{Irr}$ to P.

Lemma 2.9 *Let* $(\mathsf{Proc}, \mathsf{Lab}, \rightarrow)$ *be a prime ltr. If* $(P, a, Q) \sim (R, a, S)$ *then there are* T, U *as in Figure 3 such that the processes on opposite sides of each diamond are unequal, and the events in the paths from* T *to* P, R *and from* U *to* Q, S *are all different from* $[P, a, Q]$.

Proof. The essence of the proof is to show that if one has P, Q, R, S, T, U as in the first graph in Figure 4, with the transitions from P, R to T and from Q, S to U, then there exist V, W and the appropriate dotted transitions (for example from V to P, R). We use reverse ED and RD. □

Proposition 2.10 *Let* $(\mathsf{Proc}, \mathsf{Lab}, \rightarrow)$ *be a prime ltr. In any path there are no repeated events.*

Proof. Suppose we have a path from P to S via Q, R as in the second graph in Figure 4, with $(P, a, Q) \sim (R, a, S)$. Then by Lemma 2.9 we have T, U as in Figure 3. But now there are two different paths from T to R, one containing $[P, a, Q]$ and one not. This contradicts Proposition 2.8. □

A *rooted path* is a forward computation $P_0 \xrightarrow{a_1} P_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} P_n$ where $P_0 \in \mathsf{Irr}$. Define $e < e'$ if and only if $e \neq e'$ and all rooted paths that contain a representative of e' also contain a representative of e. (Note that we are generalising the [6] definition to ltrs rather than process graphs with a single initial state.)

Lemma 2.11 (Sideways diamond) *In a prime ltr, if* $P \xrightarrow{a} Q \xrightarrow{b} R$ *and* $[P, a, Q] \not< [Q, b, R]$ *then there is* S *such that* $P \xrightarrow{b} S \xrightarrow{a} R$.

Proof. (Sketch, with abuse of notation between events and labels of events) Since $[P, a, Q] \not< [Q, b, R]$ there is a rooted path that contains b but no a before b. Using Lemma 2.9, we connect the two instances of b back to their common earliest b. On the path back from R via the earliest b there must be an occurrence of a (extend back to the coomon root and use Proposition 2.8). Hence, there is a path π with b before a. Now we apply RD to $P \xrightarrow{a} Q \xrightarrow{b} R$ and the transitions of the top segment of π starting with the a transition. Then we use FD to promote the a along π towards R, eventually obtaining the result. □

Definition 2.12 A *process graph*, or *graph*, is an lts where every process is reachable via the transition relation from the initial state.

Proposition 2.13 *Suppose that* $(\mathsf{Proc}, \mathsf{Lab}, \rightarrow)$ *is a prime ltr. If* $P \in \mathsf{Irr}$ *then* $\mathsf{der}(P)$ *is closed under* $\rightsquigarrow$.

Proof. By Lemma 2.7. □

Finally, we are ready to define the central structure of the paper.

Definition 2.14 A *prime (process) graph* is a graph $\mathcal{G} = (G, \mathsf{Lab}, \rightarrow, I)$ such that $(G, \mathsf{Lab}, \rightarrow)$ is a prime ltr.

The components of a prime graph $\mathcal{G}$ will be denoted by $G_{\mathcal{G}}$, $\rightarrow_{\mathcal{G}}$, and $I_{\mathcal{G}}$. We shall omit the subscript $\mathcal{G}$ when it is clear from the context, and we shall use this naming convention with other structures given as tuples.

The property WF guarantees that prime graphs are acyclic.

Proposition 2.15 *Let* $(G, \mathsf{Lab}, \rightarrow)$ *be a prime ltr, and let* $I \in \mathsf{Irr}$. *Then* $(\mathsf{der}(I), \mathsf{Lab}, (\rightarrow \cap \mathsf{der}(I)^2), I)$ *is a prime graph.*

Proof. We use Proposition 2.13. It is easy to check that the properties of prime ltrs hold for $\mathsf{der}(I)$. □

2.1 Process graphs for CCSK

We proved in Proposition 2.4 that the process graphs that we get for CCSK are prime graphs. Moreover, CCSK process graphs satisfy an additional property:

NR (non-repeating) there are no repeated labels in forward computations.

NR and WF imply what is called the *nonrepetitive* property in [4]. CCSK graphs satisfy NR as a consequence of using the keys mechanism mentioned in the introduction:

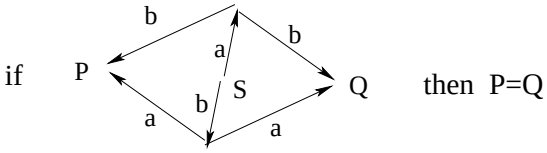
Proposition 2.16 *The ltr of [8] satisfies NR.*

Prime graphs that satisfy NR are called *non-repeating* prime graphs. We shall show in Section 4 that prime event structures that correspond to non-repeating prime graphs enjoy the properties of no auto-concurrency [1] and no auto-causation.

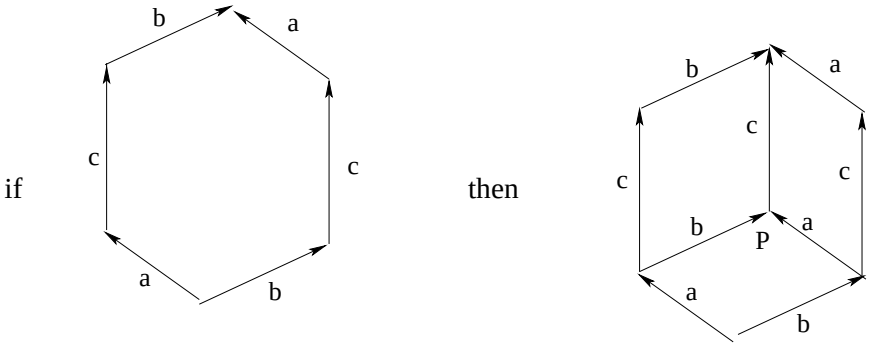
2.2 Local characterisation of prime graphs

Several of the properties of prime graphs, notably ED and FD, are global in character: in order to verify them one may need to inspect an arbitrarily large portion of a graph. The global character, however, makes them very useful in proving other properties and results for prime graphs. An open question was posed by Sassone *et al.* in [9]: Can one find a set of properties that involve only local information that characterise a form of transition systems that are equivalent to prime event structures (with binary conflict)? We answer this question for prime event structures with general (not necessarily binary) conflict by proposing local versions of the ED and FD properties:

- **ED2** (event-deterministic 2)



- **FD2** (forward diamond 2)



Note that ED2 could be seen as a form of a forward diamond property: if $S \xrightarrow{a}\xrightarrow{b} P$ and $S \xrightarrow{b}\xrightarrow{a} P$ is a forward diamond and there is another ab diamond from S to Q, then $P = Q$. In other words, if there a forward diamond from S to P, then such P is unique.

Definition 2.17 A *prime2 graph* is defined as a prime graph except that its ltr component satisfies ED2 and FD2 instead of ED and FD, respectively.

Note that the graph in the definition of ED2 is isomorphic to the first graph in Figure 1. Looking back at RD, the state S is the unique such S for prime2 graphs. This can be proved using RD three times and then ED2. We shall denote RD with unique S by **RD2**. We also have that P in FD2 is the unique such P, as can be proved using FD2 and ED2. We can define reverse Event Determinism (**reverse ED2**) property by reversing the arrows in ED2. Then reverse ED2 follows easily from RD2.

Proposition 2.18 *Prime graphs are prime2 graphs and vice versa.*

Proof. Omitted. □

3 Prime event structures

We would like to map prime graphs into prime event structures, and vice versa, so that we can then compare our FR bisimulation with HHP bisimulation.

The definition below of a prime event structure is taken from [6] and is consistent with the original definition by Winskel [10]. It generalises *prime event structures with binary conflict relation*.

Definition 3.1 A *labelled prime event structure* is a tuple $\mathcal{E} = (E, <, \sharp, \ell)$, where E is the set of *events*, $< \subseteq E \times E$ is a partial order, called the *causality relation*, satisfying the principle of finite causes: $\{e' \in E \mid e' < e\}$ is finite for each $e \in E$, and $\sharp \subseteq \mathcal{P}(E)$ is a set of finite, nonempty, non-singleton subsets of E, called the *conflict relation*, satisfying the principles of extension:

$$\sharp X \text{ and } Y \subseteq_{fin} E \text{ implies } \sharp(X \cup Y)$$

and *conflict heredity*:

$$\sharp(X \cup \{e\}) \text{ and } \ e < e' \text{ implies } \sharp(X \cup \{e'\}),$$

and $\ell : E \to \mathsf{Lab}$ is a labelling function.

A set of events X is conflict-free if $X \notin \sharp$. A prime event structure has a binary conflict if for every X such that $\sharp X$ there is Y with only two elements, $Y \subseteq X$ and $\sharp Y$.

Given a set of events X of E, the set of events below X, written as $X{\downarrow}$, is defined as $\{e' \in E \mid \exists e \in X.\, e' \leq e\}$. X is downwards-closed if $X = X{\downarrow}$.

Two prime event structures are *isomorphic* if and only if there exists a bijection between their sets of events which preserves and reflects the causality relation, the conflict relation and the labelling.

Example 3.2 Consider the event structure with events $\{a, b, c\}$ and where $\sharp\{a, b, c\}$, with no other conflict and no causation. This event structure is equivalent to no prime event structure with binary conflict.

The behaviour of a prime event structure is represented in terms of *configurations*: finite subsets of events which are conflict-free and downwards-closed. Given a prime event structure $\mathcal{E}$, $\mathsf{cfs}(E_{\mathcal{E}})$ is the set of its configurations. The *configuration* graph for $\mathcal{E}$, written as $\mathsf{cg}(\mathcal{E})$, has configurations of $E_{\mathcal{E}}$ as its states and is got by taking $\emptyset$ as the initial state and setting $X \xrightarrow{a}_{\mathsf{cg}(\mathcal{E})} Y$ if $Y \setminus X = \{e\}$ with $\ell(e) = a$. Such configuration graphs are ranged over by $\mathcal{C}, \mathcal{D}$ and have the sets of configurations $C_{\mathcal{C}}, D_{\mathcal{D}}$; the associated transition relations are $\to_{\mathcal{C}}, \to_{\mathcal{D}}$, respectively. The reverse transition relations are $\rightsquigarrow_{\mathcal{C}}, \rightsquigarrow_{\mathcal{D}}$, respectively.

Proposition 3.3 *The configuration graph of a prime event structure is a prime graph.*

Proof. Let $\mathcal{C}$ be the configuration graph of a prime event structure. It suffices to check all the properties of prime graphs hold for $\mathcal{C}$.

(i) WF follows from the fact that configurations are finite subsets of events, and when we compute in reverse we decrease the size of configurations.

(ii) UT follows from the definition of $\rightarrow_{\mathcal{C}}$: $X \xrightarrow{a}_{\mathcal{C}} Y$ implies $Y \setminus X$ has just one event and its label is a.

(iii) ED follows by the definition of configurations and $\rightarrow_{\mathcal{C}}$.

(iv) RD is due to the fact that intersection of two configurations is also a configuration for prime event structures [4]. For configurations P, Q, R and S we have $P = Q \cup \{a\}$, $P = R \cup \{b\}$, so there is S such that $Q = S \cup \{b\}$ and $R = S \cup \{a\}$. The intersection of Q and R is a configuration, meaning that S is a configuration.

(v) FD follows from a reachability property of configurations [4], namely that if X, Y, Z are configurations and $X \cup Y \subseteq Z$ then $X \cup Y$ is a configuration. For configurations P, Q, R and T we have $Q = P \cup \{a\}$, $R = P \cup \{b\}$ and $Q \cup R \subseteq T$, so $Q \cup R$ is a configuration and $S = Q \cup R$.

□

The configuration graph of the event structure in Example 3.2 forms three faces of a cube. It is a prime graph.

The configuration graph corresponding to an event structure in full generality (i.e. not necessarily prime) may not satisfy some of the properties of prime graphs:

Example 3.4 Consider the parallel switch of [10, Example 1.1.7], where the bulb will light if either of two switches is closed. This is an example of so-called or-causation where a disjunction of events causes an event. We have event b (bulb) caused by either of the switch events 0 or 1, and $\{0, b\} \xrightarrow{1} \{0, 1, b\}$, $\{1, b\} \xrightarrow{0} \{0, 1, b\}$. But $\{b\}$ is not a configuration, so that we cannot complete the reverse diamond and, hence, RD does not hold.

A "non-repeating" condition for prime event structures concerns events:

Definition 3.5 A labelled prime event structure is *non-repeating* if for any events e, e', if $\ell(e) = \ell(e')$ then $e = e'$ or $\sharp\{e, e'\}$.

The reason for the name is that no two events that occur in a configuration can have the same label. Another way to look at this is that a configuration gives rise to a set of labels, rather than a multiset of labels, as would otherwise be the case in general. It is equivalent to no auto-concurrency [1] and no auto-causation: if $e < e'$ then $\ell(e) \neq \ell(e')$.

Proposition 3.6 *No two events of a configuration of a non-repeating prime event structure have the same label.*

4 Correspondence of prime graphs and prime event structures

Van Glabbeek and Vaandrager [6] give a method for assigning to a process graph $\mathcal{G}$ (with initial state) a labelled prime event structure $\mathsf{es}(\mathcal{G})$. The method is as

follows. Take any process graph $\mathcal{G} = (G, \mathsf{Lab}, \rightarrow, I)$. The relation $\sim$ and the events $[P, a, Q]$ are defined as in Section 2. Let $E = \rightarrow/\sim$. Define a labelling function $\ell : E \rightarrow \mathsf{Lab}$ by setting $\ell([P, a, Q]) = a$. A rooted path is a forward computation $I = P_0 \stackrel{a_1}{\rightarrow} P_1 \stackrel{a_2}{\rightarrow} \cdots \stackrel{a_n}{\rightarrow} P_n$ starting at the root. Define $e < e'$ if and only if $e \neq e'$ and all rooted paths that contain a representative of e' also contain a representative of e. Let $X \subseteq E$ be finite. We define $\sharp X$ if and only if $\mathcal{G}$ does not have a rooted path containing representatives of all events in X. Then $\mathcal{E} = (E, <, \sharp, \ell)$ is a prime event structure.

Van Glabbeek and Vaandrager show in [6] that if $\mathcal{E}$ is a prime event structure then $\mathsf{es}(\mathsf{cg}(\mathcal{E})) = \mathcal{E}$. Here we do not distinguish isomorphic process graphs. This shows that prime event structures are embeddable in process graphs.

Proposition 4.1 *Given a prime event structure $\mathcal{E}$, $\mathsf{es}(\mathsf{cg}(\mathcal{E})) = \mathcal{E}$.*

We shall show that any prime graph is the configuration graph of the event structure got from the prime graph by the procedure described above, namely:

Theorem 4.2 *Given a prime graph $\mathcal{G}$, $\mathsf{cg}(\mathsf{es}(\mathcal{G})) = \mathcal{G}$.*

The rest of this section is devoted to proving the theorem.

Consider $\mathsf{es}(\mathcal{G})$. Clearly the configuration corresponding to $P \in G$ is just the set of events executed up to P. Let $I = P_0 \stackrel{a_1}{\rightarrow} P_1 \stackrel{a_2}{\rightarrow} \cdots \stackrel{a_n}{\rightarrow} P_n = P$ be a path with endpoint P. So we define the configuration associated with P as $\mathsf{cf}(P) = \{[P_i, a_{i+1}, P_{i+1}] : i < n\}$. Clearly $\mathsf{cf}(P) \in \mathsf{cfs}(\mathsf{es}(\mathcal{G}))$. We have that $\mathsf{cf}(P)$ is well-defined: since any two paths between I and P are homotopic they contain the same sets of events. We must show that $\mathsf{cf}(\,)$ is a bijection from G to $\mathsf{cfs}(\mathsf{es}(\mathcal{G}))$, and then $P \stackrel{a}{\rightarrow} Q$ if and only if $\mathsf{cf}(P) \stackrel{a}{\rightarrow}_{\mathsf{cg}(\mathsf{es}(\mathcal{G}))} \mathsf{cf}(Q)$.

Proposition 4.3 *Let $(G, \mathsf{Lab}, \rightarrow, I)$ be a prime graph. Suppose that $\mathsf{cf}(P) = \mathsf{cf}(Q)$. Then $P = Q$.*

Proof. (Sketch, with abuse of notation between events and labels of events) Suppose we have $I \stackrel{a}{\rightarrow} P' \stackrel{s}{\rightarrow} P$. Then we must have $I \stackrel{t}{\rightarrow} Q' \stackrel{a}{\rightarrow} Q'' \stackrel{t'}{\rightarrow} Q$ with $(I, a, P') \sim (Q', a, Q'')$. Clearly any events in t cannot be below a, and so by repeated use of SD (Lemma 2.11) we can "promote" a to the front of t to get $I \stackrel{a}{\rightarrow} Q''' \stackrel{t}{\rightarrow} Q'' \stackrel{t'}{\rightarrow} Q$. Now we use ED to deduce that $P' = Q'''$. Continuing this process along s we deduce that $P = Q$. □

Proposition 4.4 *Let $\mathcal{G} = (G, \mathsf{Lab}, \rightarrow, I)$ be a prime graph. If $X \in \mathsf{cfs}(\mathsf{es}(\mathcal{G}))$ then there is P such that $\mathsf{cf}(P) = X$*

Proof. (Sketch, with abuse of notation between events and labels of events) Let $X \in \mathsf{cfs}(\mathsf{es}(\mathcal{G}))$. We show that if $I \stackrel{s}{\rightarrow} Q$ and $X \subseteq \mathsf{cf}(Q)$ with X downwards-closed under $<$ then there exist t, P such that $\mathsf{cf}(P) = X$ and $I \stackrel{t}{\rightarrow} P \rightarrow^* Q$. Suppose that a comes immediately before b in s, and $a \notin X$, $b \in X$. Then $a \not< b$ since X is downwards-closed. We can swap a and b by SD (Lemma 2.11). Continue this until all events in X occur before all events not in X. We have the desired t, P. □

Proposition 4.5 *Let $\mathcal{G} = (G, \mathsf{Lab}, \rightarrow, I)$ be a prime graph. Then $P \stackrel{a}{\rightarrow} Q$ if and only if $\mathsf{cf}(P) \stackrel{a}{\rightarrow}_{\mathsf{cg}(\mathsf{es}(\mathcal{G}))} \mathsf{cf}(Q)$.*

Proof. (Sketch, with abuse of notation between events and labels of events) ($\Rightarrow$) Clearly $\mathsf{cf}(Q) = \mathsf{cf}(P) \cup \{[P, a, Q]\}$. Here we use the result that events cannot be repeated along any path (Proposition 2.10). Then $\mathsf{cf}(P) \xrightarrow{a}_{\mathsf{cg}(\mathsf{es}(\mathcal{G}))} \mathsf{cf}(Q)$ by definition.

($\Leftarrow$) We have $\mathsf{cf}(Q) = \mathsf{cf}(P) \cup \{e\}$ where $\ell(e) = a$. Suppose $I \xrightarrow{s} P$ and $I \xrightarrow{tat'} Q$. Then for each $b \in t'$ we have $a \not< b$ since $b \in s$ and $a \notin s$. So by SD (Lemma 2.11), a can be permuted with each successive member of t' and we have $I \xrightarrow{tt'} R \xrightarrow{a} Q$ for some R. But then $R = P$ by Proposition 4.3. □

Finally, we show that if we add a condition on prime graphs that no path can include repeated labels, then we have a correspondence with event structures where events with the same label must be the same or in conflict.

Proposition 4.6 *Let $\mathcal{G}$ be a non-repeating prime graph. Then $\mathsf{es}(\mathcal{G})$ is a non-repeating prime event structure.*

Proof. Assume for contradiction that $\mathsf{es}(\mathcal{G})$ is not non-repeating: there are different events e, e' with $\ell(e) = \ell(e')$ and $\{e, e'\} \notin \sharp$. The last implies, by the definition of $\sharp$, that e, e' appear on a path. Hence, this path has repeated labels $\ell(e)$: contradiction. □

Proposition 4.7 *Let $\mathcal{E}$ be a non-repeating prime event structure. Then $\mathsf{cg}(\mathcal{E})$ is a non-repeating prime graph.*

Proof. A straightforward proof by contradiction using the fact that no path of a prime graph has repeated events (Proposition 2.10). □

5 Correspondence of bisimulations

We recall a definition of a bisimulation relation for process graphs that takes both forward and reverse transitions into account [8]:

Definition 5.1 Let $\mathcal{G}$ and $\mathcal{H}$ be process graphs. A symmetric relation $\mathcal{S} \subseteq G_{\mathcal{G}} \times H_{\mathcal{H}}$ is a *forward-reverse (FR) bisimulation* between $\mathcal{G}$ and $\mathcal{H}$ if $\mathcal{S}(I_{\mathcal{G}}, I_{\mathcal{H}})$ and whenever $\mathcal{S}(P, Q)$ then

- if $P \xrightarrow{\mu}_{\mathcal{G}} P'$ then there is Q' such that $Q \xrightarrow{\mu}_{\mathcal{H}} Q'$ and $\mathcal{S}(P', Q')$;
- if $P \overset{\mu}{\rightsquigarrow}_{\mathcal{G}} P'$ then there is Q' such that $Q \overset{\mu}{\rightsquigarrow}_{\mathcal{H}} Q'$ and $\mathcal{S}(P', Q')$.

We define $\mathcal{G} \sim_{\mathsf{FR}} \mathcal{H}$ if and only if there is an FR bisimulation between $\mathcal{G}$ and $\mathcal{H}$.

By abuse of notation we shall write $P \sim_{\mathsf{FR}} Q$ if P, Q are states of $\mathcal{G}, \mathcal{H}$, respectively, and there exists an FR bisimulation $\mathcal{S}$ between $\mathcal{G}$ and $\mathcal{H}$ such that $\mathcal{S}(P, Q)$.

A similar relation, called *back-and-forth bisimulation* was proposed by Bednarczyk [1]. It uses forward transitions to mimic the effect of reverse transitions. Since in this paper $\rightsquigarrow$ is the inverse of $\rightarrow$, the two bisimulations coincide. However, this does not hold for general transition systems where some transitions are irreversible.

Example 5.2 FR bisimulation satisfies an intuitive equation $a = a + a$. Note that the graph of $a + a$ is a non-repeating prime graph.

FR bisimulation is insensitive to auto-concurrency: Let $I \xrightarrow{a} P \xrightarrow{a} R$, $I \xrightarrow{a} Q \xrightarrow{a} R$ with $P \neq Q$, and $I' \xrightarrow{a} P' \xrightarrow{a} R'$. Consider the graphs with initial states I and I', respectively. They are clearly FR bisimilar, and they have auto-concurrency and auto-causation, respectively. Note that CCSK processes $a \mid a$ and $a.a$ are not FR bisimilar: using the notation from [8] we have that after performing a, a we get to $a[m] \mid a[n]$ and $a[m].a[n]$. Now, $a[m] \mid a[n] \overset{a[m]}{\rightsquigarrow}$ and $a[m].a[n] \not\overset{a[m]}{\rightsquigarrow}$.

Finally, FR bisimulation does not satisfy the absorption law:

$$(a \mid (b + c)) + (a \mid b) + ((a + c) \mid b) = (a \mid (b + c)) + ((a + c) \mid b)$$

If one performs a and then b with the $a \mid b$ component on the left, then these must be matched by the a and then the b of the $((a + c) \mid b)$ summand on the right. (Matching it with the a of $(a \mid (b + c))$ is wrong as after this a action is performed, no c is possible after a in $a \mid b$.) The right hand side can now reverse a and do a c (still using the same summand as all other summands are disabled). The left hand side cannot match this: the component $a[m] \mid b[n]$ can regress by $a[m]$ to $a \mid b[n]$ but it cannot perform any c.

We have shown that the configuration graphs associated with prime graphs are prime graphs themselves (Theorem 4.2), and that the configuration graphs for prime event structures are prime graphs (Proposition 3.3). Hence, in the setting of prime graphs and prime event structures the definition of FR bisimulation generalises trivially to configuration graphs.

Hereditary history-preserving bisimulation was proposed by Bednarczyk [1], and also appeared under a different name in an earlier version of [5] and in [7]. Here, we present a reformulation of the original definition due to van Glabbeek and Goltz [5] (but without termination). This particular bisimulation is defined over configuration graphs and, in addition to matching configurations and the transitions between configurations, it also keeps a history of the matched events along matching computations. This is achieved by means of a label-preserving and order-preserving isomorphism between the events of the two configuration graphs.

Given two prime event structures $\mathcal{E}$ and $\mathcal{F}$, their configuration graphs are $\mathsf{cg}(\mathcal{E})$ and $\mathsf{cg}(\mathcal{F})$. Let these graphs be called $\mathcal{C}$ and $\mathcal{D}$, respectively, with $C = \mathsf{cfs}(E)$ and $D = \mathsf{cfs}(F)$. In order to define the so-called "history" isomorphisms we need to consider the full sub-event structures determined by configurations. We shall write such event structures as $(X, <_X, \sharp_X, \ell \restriction X)$ for a configuration X of $\mathcal{E}$ where $<_X = <_{\mathcal{E}} \cap (X \times X)$ and $\ell \restriction X$ is the restriction of the domain of ℓ to X. Since each X is conflict-free the restriction of the conflict relation $\sharp$ to X is empty, i.e. $\sharp_X = \emptyset$. Hence, it suffices to define the required isomorphisms over the structures $(X, <_X, \ell \restriction X)$.

Definition 5.3 Let $\mathcal{C}$ and $\mathcal{D}$ be configuration graphs. A relation $\mathcal{R} \subseteq C \times D \times \mathcal{P}(E_{\mathcal{C}} \times E_{\mathcal{D}})$ is a *hereditary history-preserving (HHP) bisimulation* between $\mathcal{C}$ and $\mathcal{D}$ if $\mathcal{R}(\emptyset, \emptyset, \emptyset)$, and whenever $\mathcal{R}(X, Y, f)$ then

- f is an isomorphism between $(X, <_X, \ell \restriction X)$ and $(Y, <_Y, \ell \restriction Y)$;
- if $X \xrightarrow{\mu}_{\mathcal{C}} X'$ then $\exists\, Y', f'$ such that $Y \xrightarrow{\mu}_{\mathcal{D}} Y'$, $\mathcal{R}(X', Y', f')$ and $f' \restriction X = f$;

- if $Y \stackrel{\mu}{\rightarrow}_{\mathcal{D}} Y'$ then $\exists\, X', f'$ such that $X \stackrel{\mu}{\rightarrow}_{\mathcal{C}} X'$, $\mathcal{R}(X', Y', f')$ and $f' \restriction X = f$;
- if $X \stackrel{\mu}{\rightsquigarrow}_{\mathcal{C}} X'$ then $\exists\, Y', f'$ such that $Y \stackrel{\mu}{\rightsquigarrow}_{\mathcal{D}} Y'$, $\mathcal{R}(X', Y', f')$ and $f \restriction X' = f'$;
- if $Y \stackrel{\mu}{\rightsquigarrow}_{\mathcal{D}} Y'$ then $\exists\, X', f'$ such that $X \stackrel{\mu}{\rightsquigarrow}_{\mathcal{C}} X'$, $\mathcal{R}(X', Y', f')$ and $f \restriction X' = f'$.

We define $\mathcal{C} \sim_{\mathsf{HHP}} \mathcal{D}$ if and only if there is an HHP bisimulation between $\mathcal{C}$ and $\mathcal{D}$.

The main result of this section is that FR bisimulation coincides with HHP bisimulation on configuration graphs for non-repeating prime event structures. A similar result was proved by Bednarczyk [1]. He considered configuration structures which arise from a smaller family of prime event structure with *binary conflict*. But instead of the non-repeating property his result holds under a less restrictive no auto-concurrency condition.

Theorem 5.4 *Let $\mathcal{E}$ and $\mathcal{F}$ be non-repeating prime event structures, and let $\mathcal{C}$ and $\mathcal{D}$ be their configuration graphs. Then, $\mathcal{C} \sim_{\mathsf{HHP}} \mathcal{D}$ if and only if $\mathcal{C} \sim_{\mathsf{FR}} \mathcal{D}$.*

Example 5.5 The graphs for a and $a + a$ are HHP bisimilar. Consider the graphs with initial states I and I' from Example 5.2. Although they are FR bisimilar they are not HHP bisimilar: If we match the two a transitions $I \stackrel{a}{\rightarrow} P \stackrel{a}{\rightarrow} R$ with the respective transitions in $I' \stackrel{a}{\rightarrow} P' \stackrel{a}{\rightarrow} R'$, we obtain an order isomorphism $\{([(I, \stackrel{a}{\rightarrow}, P)], [(I', \stackrel{a}{\rightarrow}, P')]),\ ([(P, \stackrel{a}{\rightarrow}, R)], [(P', \stackrel{a}{\rightarrow}, R')])$. Then, reversing a from R to Q cannot be matched by reversing the isomorphic transition from R' since $[(Q, \stackrel{a}{\rightarrow}, R)] = [(I, \stackrel{a}{\rightarrow}, P)]$.

The proof of Theorem 5.4 from left to right is by definition of FR bisimulation. The other direction requires first some auxiliary results.

Since configurations of non-repeating prime event structures have no repeated events (Proposition 2.10) and no two events share the same label, the events in a configuration can be identified uniquely by their labels. And, the causality order on the events in a configuration reduces to the corresponding order on events' labels. Hence, in the remainder the isomorphisms required in Definition 5.3 shall be called simply *order isomorphisms*.

Proposition 5.6 *Let $\mathcal{C}$ and $\mathcal{D}$ be configurations graphs of non-repeating prime event structures $\mathcal{E}$ and $\mathcal{F}$. Let $\mathcal{S}$ be an FR bisimulation between $\mathcal{C}$ and $\mathcal{D}$, and let $\mathcal{S}(C, D)$ for $C \in \mathcal{C}$ and $D \in \mathcal{D}$. Then a map $is(C, D)$, defined by $is(C, D)(e) = e'$ if $\ell(e) = \ell(e')$ and $e \in C, e' \in D$, is an order isomorphism between C and D.*

Proof. The map *is* is clearly well-defined. since configurations of non-repeating prime event structures have no repeated events (Proposition 2.10) and no two events have the same label. One shows that the map is injective by using the above mentioned results. To show that *is* is surjective, namely for every $e' \in D$ there is $e \in C$ such that $is(C, D)(e) = e'$, we consider $\mathcal{S}(C, D)$. A reverse computation $D \stackrel{t}{\rightsquigarrow}_{\mathcal{D}} \stackrel{a}{\rightsquigarrow}_{\mathcal{D}} D'$, where $\ell(e') = a$ and t is some sequence of labels for events in D, implies $C \stackrel{t}{\rightsquigarrow}_{\mathcal{C}} \stackrel{a}{\rightsquigarrow}_{\mathcal{C}} C'$ by $\mathcal{S}(C, D)$. Because of the non-repeating property, there is a unique event $e \in C$ with the label a, so $is(C, D)(e) = e'$.

It remains to prove that *is* is order-preserving. We shall show $a <_C b$ if and only if $a <_D b$ for all appropriate events (identified by their labels) a, b. Assume for contradiction that $a <_C b$ and not $a <_D b$. The last means that there is a path

involving transitions for all events of D such that it contains no a before b. Hence, this path has a after b. We reverse the transitions in this path and, since $\mathcal{S}(C,D)$, we match them with the corresponding transitions from C. Therefore, we get a path in C with a after b. Since $a <_C b$ implies that every path that has b contains also a, we have that a path in C has two occurrences of a: contradiction. □

The following result is a consequence of the non-repeating property for both prime event structures and prime graphs. It is proved similarly as the previous proposition.

Proposition 5.7 *Let $\mathcal{C}$ and $\mathcal{D}$ be configuration graphs of non-repeating prime event structures $\mathcal{E}$ and $\mathcal{F}$. Let $\mathcal{S}$ be an FR bisimulation between $\mathcal{C}$ and $\mathcal{D}$, and let $\mathcal{S}(C,D)$ for $C \in \mathcal{C}$ and $D \in \mathcal{D}$.*

(i) *If $C \xrightarrow{a}_{\mathcal{C}} C'$ with $C' \setminus C = \{e\}$ and $\ell(e) = a$ then there is D' such that $D \xrightarrow{a}_{\mathcal{D}} D'$ with $D' \setminus D = \{e'\}$ and $\ell(e') = a$, and $\mathcal{S}(C',D')$ and $is(C',D')$ is an order isomorphism between C' and D';*

(ii) *If $D \xrightarrow{a}_{\mathcal{D}} D'$ with $D' \setminus D = \{e'\}$ and $\ell(e') = a$ then there is C' such that $C \xrightarrow{a}_{\mathcal{C}} C'$ with $C' \setminus C = \{e\}$ and $\ell(e) = a$, and $\mathcal{S}(C',D')$ and $is(C',D')$ is an order isomorphism between C' and D'.*

Proof of Theorem 5.4. Let $\mathcal{E}$ and $\mathcal{F}$ be non-repeating prime event structures, and let $\mathcal{C}$ and $\mathcal{D}$ be their configuration graphs. We show $\mathcal{C} \sim_{\mathsf{FR}} \mathcal{D}$ implies $\mathcal{C} \sim_{\mathsf{HHP}} \mathcal{D}$.

Let $\mathcal{S}$ be an FR bisimulation that relates the configurations of $\mathcal{C}$ and $\mathcal{D}$. Define a relation $\mathcal{R} \subseteq C_{\mathcal{C}} \times D_{\mathcal{D}} \times \mathcal{P}(E_{\mathcal{C}} \times E_{\mathcal{D}})$ as follows: for all configurations C, D we let $\mathcal{R}(C,D,f)$ if $\mathcal{S}(C,D)$ and $f = is(C,D)$. It remains to show that $\mathcal{R}$ is a hereditary history-preserving bisimulation between $\mathcal{C}$ and $\mathcal{D}$.

Clearly, $\mathcal{R}(\emptyset,\emptyset,\emptyset)$. Assume $\mathcal{R}(C,D,is(C,D))$. Then $is(C,D)$ is an order isomorphism, and the two conditions for the forward transitions follow from Proposition 5.7. For the reverse transition conditions we consider the last one from Definition 5.3. Let $D \overset{\mu}{\rightsquigarrow}_{\mathcal{D}} D'$. Then by $\mathcal{S}(C,D)$ there is C' such that $C \overset{\mu}{\rightsquigarrow}_{\mathcal{C}} C'$ and $\mathcal{S}(C',D')$. Clearly, $C' \setminus C = \{e\}$ with $\ell(e) = \mu$ and $D' \setminus D = \{e'\}$ with $\ell(e') = \mu$. Since $is(C',D')(e) = is(C,D)(e)$ if $e \in C$ and $is(C',D')(e) = e'$ if $e \in C' \setminus C$, we deduce that $is(C',D')$ is an order isomorphism. And we obtain $\mathcal{R}(C',D',is(C',D'))$.□

6 Conclusions and future directions

We have proposed prime graphs as an alternative and equivalent model to labelled prime event structures (with general conflict). Prime graphs are a subclass of reversible transition systems that satisfy several properties relating to concurrency and reversibility. A concrete example of prime graphs are process graphs for CCSK processes. The communication mechanism of CCSK works correctly in both directions thanks to the use of communications keys, and the keys, in turn, force an additional property of non-repeating on prime graphs for CCSK processes.

The non-repeating property does not seem to be intrinsic to reversible transition systems. However, it seems to guarantee compositionality and it deals with autoconcurrency. It raises the question of whether non-repeating is necessary or not for concrete formulations.

In the second part of the paper we have investigated the distinguishing power of FR bisimulation and have shown that for non-repeating models, either prime graphs or prime event structures, FR bisimulation coincides with HHP bisimulation. Since CCSK gives rise naturally to such models, HHP bisimulation, and hopefully other true concurrency equivalences, can be verified by standard interleaving-based techniques. In future it would be interesting to study trace-based equivalences in reversible transition systems and identify the corresponding equivalences in true concurrency models.

Acknowledgements

We would like to thank Daniele Varacca, Sibylle Fröschle, Reiko Heckel and the anonymous referees for helpful comments and suggestions.

References

[1] Bednarczyk, M., *Hereditary history preserving bisimulations or what is the power of the future perfect in program logics*, Technical Report ICS PAS, Polish Academy of Sciences (1991).

[2] Danos, V. and J. Krivine, *Reversible communicating systems*, in: P. Gardner and N. Yoshida, editors, *Proceedings of the 15th International Conference on Concurrency Theory CONCUR 2004*, LNCS **3170** (2004), pp. 292–307.

[3] Fiore, M. P., G. L. Cattani and G. Winskel, *Weak bisimulation and open maps*, in: *Proceedings of the 14th Annual IEEE Symposium on Logic in Computer Science*, IEEE (1999), pp. 67–76.

[4] van Glabbeek, R.J., *History preserving process graphs* (1996), http://boole.stanford.edu/~rvg/pub/history.draft.dvi.

[5] van Glabbeek, R.J. and U. Goltz, *Refinement of actions and equivalence notions for concurrent systems*, Acta Informatica **37** (2001), pp. 229–327.

[6] van Glabbeek, R.J. and F. Vaandrager, *The difference between splitting in n and n + 1*, Information and Computation **136** (1997), pp. 109–142.

[7] Joyal, A., M. Nielsen and G. Winskel, *Bisimulation from open maps*, Information and Computation **127** (1996), pp. 164–185.

[8] Phillips, I.C.C. and I. Ulidowski, *Reversing algebraic process calculi*, in: *Proceedings of 9th International Conference on Foundations of Software Science and Computation Structures, FOSSACS 2006*, LNCS **3921** (2006), pp. 246–260. Extended version accepted by Journal of Logic and Algebraic Programming.

[9] Sassone, V., M. Nielsen and G. Winskel, *Models of concurrency: Towards a classification*, Theoretical Computer Science **170** (1996), pp. 297–348.

[10] Winskel, G., *Event structures*, in: *Advances in Petri Nets 86*, LNCS **255** (1987), pp. 325–392.