Minimisation of Event Structures

Paolo Baldan
University of Padova, Italy
baldan@math.unipd.it

Alessandra Raffaetà
Ca' Foscari University of Venice, Italy
raffaeta@unive.it

Abstract

Event structures are fundamental models in concurrency theory, providing a representation of events in computation and of their relations, notably concurrency, conflict and causality. In this paper we present a theory of minimisation for event structures. Working in a class of event structures that generalises many stable event structure models in the literature, (e.g., prime, asymmetric, flow and bundle event structures) we study a notion of behaviour-preserving quotient, taking hereditary history preserving bisimilarity as a reference behavioural equivalence. We show that for any event structure a uniquely determined minimal quotient always exists. We observe that each event structure can be seen as the quotient of a prime event structure, and that quotients of general event structures arise from quotients of (suitably defined) corresponding prime event structures. This gives a special relevance to quotients in the class of prime event structures, which are then studied in detail, providing a characterisation and showing that also prime event structures always admit a unique minimal quotient.

2012 ACM Subject Classification Theory of computation → Concurrency; Software and its engineering → Formal methods

Keywords and phrases Event structures, minimisation, history-preserving bisimilarity, behaviour-preserving quotient

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2019.30

Related Version Extended version of the paper available at `http://arxiv.org/abs/1907.07042`.

Funding *Paolo Baldan*: Supported by the PRIN Project Analysis of Program Analyses (ASPRA).

1 Introduction

When dealing with formal models of computational systems, a classical problem is that of minimisation, i.e., for a given system, define and possibly construct a compact version of the system which, very roughly speaking, exhibits the same behaviour as the original one, avoiding unnecessary duplications. The minimisation procedure depends on the notion of behaviour of interest and also on the expressive power of the formalism at hand, which determines its capability of describing succinctly some behaviour. A classical example is that of finite state automata, where one is typically interested in the accepted language. Given a deterministic finite state automaton, a uniquely determined minimal automaton accepting the same language can be constructed, e.g., as a quotient of the original automaton via a partition/refinement algorithm (see, e.g., [16]). Moving to non-deterministic finite automata, minimal automata become smaller, at the price of a computationally more expensive minimisation procedure and non-uniqueness of the minimal automaton [22].

In this paper we study the problem of minimisation for event structures, a fundamental model in concurrency theory [33, 34]. Event structures are a natural semantic model when one is interested in modelling the dynamics of a system by providing an explicit representation of the events in computations (occurrence of atomic actions) and of the relations between

39th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2019).
Editors: Arkadev Chattopadhyay and Paul Gastin; Article No. 30; pp. 30:1–30:15
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

events, like causal dependencies, choices, possibility of parallel execution, i.e., in what is referred to as a true concurrent (non-interleaving) semantics. Prime event structures [24], probably the most widely used event structure model, capture dependencies between events in terms of causality and conflict. A number of generalisations of prime event structures have been introduced in the literature. For instance, flow [9, 8] and bundle [20] event structures add the possibility of directly modelling disjunctive causes. Asymmetric event structures [5] and extended bundle event structures [20] include an asymmetric form of conflict which allows one to model concurrent readings and precedences between actions. Event structures have been used for defining a concurrent semantics of several formalisms, like Petri nets [24], graph rewriting systems [4, 3, 27] and process calculi [32, 31, 10]. Recent applications are in the field of weak memory models [11, 25, 17, 13] and of process mining and differencing [15].

Behavioural equivalences, defined in a true concurrent setting, take into account not only the possibility of performing steps, but also the way in which such steps relate with each other. We will focus on hereditary history preserving (hhp-)bisimilarity [7], the finest equivalence in the true concurrent spectrum in [29], which, via the concept of open map, has been shown to arise as a canonical behavioural equivalence when considering partially ordered computations as observations [18].

The motivation for the present paper originally stems from some work on business process models. The idea, advocated in [15, 1], is to use event structures as a foundation for representing, analysing and comparing process models. Processes are mined in the form of event structures and then translated into a suitable process modeling language. The processes, in their graphical presentation, should be simple and understandable, as much as possible, by a human user, who should be able, e.g., to interpret the differences between two processes diagnosed by a comparison tool. For this aim it is important to avoid "redundancies" in the representation and thus to reduce the number of events, without altering the behaviour (modifications that "extend" the behaviour could be sensible in model generalisation, but this is not of interest here). The paper [2] explores the use of asymmetric and flow event structures and, for such models, it introduces some reduction techniques that allow one to merge events without changing the true concurrent behaviour. A notion of behaviour-preserving quotient, referred to as a folding, is introduced over an abstract class of event structures, having asymmetric and flow event structures as subclasses. However, no general theory is developed. The mentioned paper focuses on a special class of foldings, the so-called elementary foldings, which can only merge a single set of events into one event, and these are studied separately on each specific subclass of event structures (asymmetric and flow event structures), providing only sufficient conditions ensuring that a function is an elementary folding.

A general theory of behaviour-preserving quotients for event structures is thus called for, settling some natural foundational questions. Is the notion of folding adequate, i.e., are all behaviour-preserving quotients expressible in terms of foldings? Is there a minimal quotient in some suitably defined general class of event structures? Does it exist in specific subclasses? (for asymmetric and flow event structures the answer is known to be negative, but for prime event structures the question is open). Can we have a characterisation of foldings for specific subclasses of event structures, providing not only sufficient but also necessary conditions?

In this paper we start addressing the above questions. We work in a general class of event structures based on the idea of family of posets [26] (also called rigid families in [14]). Although this is not discussed in deep due to space limitations, poset event structures are sufficiently expressive to generalise most stable event structures models in the literature, including prime [24], asymmetric [5], flow [9] and bundle [20] event structures (a wider discussion can be found in [6].)

As a first step we study, in this general setting, the notion of folding. A folding is a surjective function that identifies some events while keeping the behaviour unchanged. Formally, it establishes a hhp-bisimilarity between the source and target event structure. It turns out that not all behaviour-preserving quotients arise as foldings, but we show that for any behaviour-preserving quotient, there is a folding that induces a coarser equivalence. Additionally, given two possible foldings of an event structure we show that it is always possible to "join" them. This allows us to prove that for each event structure a maximally folded version, namely a uniquely determined minimal quotient, always exists.

Relying on the order-theoretic properties of the set of configurations of event structures [26], and on the correspondence between prime event structures and domains [24], we derive that each event structure in the considered class arises as the folding of a canonical prime event structure. Moreover, all foldings between general event structures arise from foldings of the corresponding canonical prime event structures. Interestingly, this result can be derived from a characterisation of foldings as open maps in the sense of [18].

The results above give a special relevance to foldings in the class of prime event structures, which thus are studied in detail. We provide necessary and sufficient conditions characterising foldings for prime event structures. This allows us to establish a clear connection with the so-called abstraction morphisms, introduced in [12] for similar purposes. The characterisation of foldings provided can guide, at least in the case of finite structures, the effective construction of behaviour preserving quotients. Moreover we show that also prime event structures always admit a minimal quotient.

Most results have a natural categorical interpretation, which is only hinted at in the paper. In order to keep the presentation simple, the categorical references are inserted in side remarks that can be safely skipped by the non-interested reader. This applies, in particular, to the possibility of viewing foldings as open maps in the sense of [18]. This correspondence, which in the present paper only surfaces, suggests the possibility of understanding and developing our results in a more abstract categorical setting. More details are provided in the extended version [6].

The rest of the paper is structured as follows. In § 2 we introduce the class of event structures we work with and hereditary history preserving bisimilarity. Moreover, we discuss how some event structure models in the literature embed into the considered class. In § 3 we study the notion of folding, we prove the existence of a minimal quotient and we show the tight relation between general foldings and those on prime event structures. In § 4 we present folding criteria on prime event structures, and discuss the existence of minimal quotients. Finally, in § 5 we draw some conclusions, discuss connections with related literature and outline future work venues. Due to space limitations all proofs have been omitted. They can be found in the extended version [6], which also contains some additional results.

2 Event Structures and History Preserving Bisimilarity

In this section we define *hereditary history-preserving bisimilarity*, the reference behavioural equivalence in the paper. This is done for an abstract notion of event structure, introduced in [26], in a way that various stable event structure models in the literature can be seen as special subclasses. We will explicitly discuss prime [24] and flow [9, 8] event structures.

Notation. We first fix some basic notation on sets, relations and functions. Let $r \subseteq X \times X$ be a binary relation. The relation r is *acyclic* on Y if there is no $\{y_0, y_1, \ldots, y_n\} \subseteq Y$ such that $y_0 \ r \ y_1 \ r \ldots r \ y_n \ r \ y_0$. Relation r is a *partial order* if it is reflexive, antisymmetric and

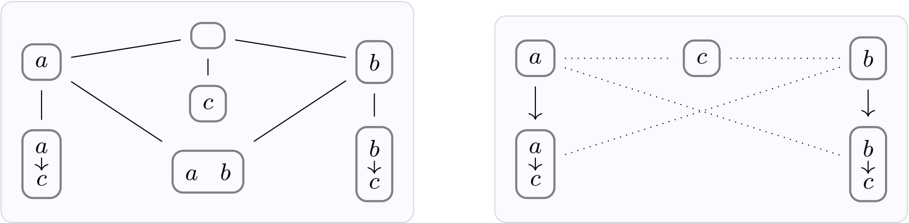

Figure 1 An event structure E and the canonical PES $\mathbb{P}(\mathsf{E})$.

transitive. Given a function $f : X \to Y$ we will denote by $f[x \mapsto y] : X \cup \{x\} \to Y \cup \{y\}$ the function defined by $f[x \mapsto y](x) = y$ and $f[x \mapsto y](z) = f(z)$ for $z \in X \setminus \{x\}$. Note that the same notation can represent an update of f, when $x \in X$, or an extension of its domain, otherwise. For $Z \subseteq X$, we denote by $f_{|Z} : Z \to Y$ the restriction of f to Z.

2.1 Event Structures

Following [26, 28, 30, 2, 14], we work on a class of event structures where configurations are given as a primitive notion. More precisely, we borrow the idea of family of posets from [26], more recently considered also under the name of rigid family in [14].

▶ **Definition 2.1** (family of posets). *A* poset *is a pair* $(C, \le_C)$ *where* C *is a set and* $\le_C$ *is a partial order on* C*. A poset will be often denoted simply as* C*, leaving the partial order relation* $\le_C$ *implicit. Given two posets* C_1 *and* C_2 *we say that* C_1 *is a* prefix *of* C_2 *and write* $C_1 \sqsubseteq C_2$ *if* $C_1 \subseteq C_2$ *and* $\le_{C_1} = \le_{C_2} \cap (C_2 \times C_1)$*. A* family of posets F *is a prefix-closed set of finite posets i.e., a set of finite posets such that if* $C_2 \in F$ *and* $C_1 \sqsubseteq C_2$ *then* $C_1 \in F$*. We say that two posets* $C_1, C_2 \in F$ *are compatible, written* $C_1 \frown C_2$*, if they have an upper bound, i.e., there is* $C \in F$ *such that* $C_1, C_2 \sqsubseteq C$*. The family of posets* F *is called* coherent *if each subset of* F *whose elements are pairwise compatible has an upper bound.*

Posets C will be used to represent configurations, i.e., sets of events executed in a computation of an event structure. The order $\le_C$ intuitively represents the order in which the events in C can occur. This motivates the prefix order that can be read as a computational extension: when $C_1 \sqsubseteq C_2$ we have that $C_1 \subseteq C_2$, events in C_1 are ordered exactly as in C_2, and the new events in $C_2 \setminus C_1$ cannot precede events already in C_1 (i.e., for all $x_1 \in C_1$, $x_2 \in C_2$, if $x_2 \le_{C_2} x_1$ then $x_2 \in C_1$).

An example of family of posets can be found in Fig. 1 (left). Observe, for instance, that the configuration with set of events $\{c\}$ is not a prefix of the one with set of events $\{a, c\}$, since in the latter $a \le c$.

An event structure is then defined simply as a coherent family of posets where events carry a label. Hereafter Λ denotes a fixed set of labels.

▶ **Definition 2.2** ((poset) event structure). *A* (poset) event structure *is a tuple* $\mathsf{E} = \langle E, \mathit{Conf}(\mathsf{E}), \lambda\rangle$ *where* E *is a set of events,* $\mathit{Conf}(\mathsf{E})$ *is a coherent family of posets such that* $E = \bigcup \mathit{Conf}(\mathsf{E})$ *and* $\lambda : E \to \Lambda$ *is a labelling function. For a configuration* $C \in \mathit{Conf}(\mathsf{E})$ *the order* $\le_C$ *is referred to as the* local order*.*

In [2] abstract event structures are defined as a collection of ordered configurations, without any further constraint. This is sufficient for giving some general definitions which are then studied in specific subclasses of event structures. Here, in order to develop a theory of foldings at the level of general event structures, we need to assume stronger properties, those of a family of posets from [26] (e.g, the fact that Definition 3.14 is well-given relies on this). This motivates the name poset event structure. Also note that, differently from what happens in other general concurrency models, like configuration structures [30], configurations are endowed explicitly with a partial order, which in turn intervenes in the definition of the prefix order between configurations. This is essential to view as subclasses some kinds of event structures, like asymmetric event structures [5] or extended bundle event structures [21], where the order on configuration is not simply subset inclusion. Since we only deal with poset event structures and their subclasses, we will often omit the qualification "poset" and refer to them just as event structures. Moreover, we will often identify an event structure E with the underlying set E of events and write, e.g., $x \in \mathsf{E}$ for $x \in E$.

An *isomorphism of configurations* $f : C \to C'$ is an isomorphism of posets that respects the labelling, namely for all $x, y \in C$, we have $\lambda(x) = \lambda(f(x))$ and $x \leq_C y$ iff $f(x) \leq_{C'} f(y)$. When configurations C, C' are isomorphic we write $C \simeq C'$.

Given an event x in a configuration C, it will be useful to refer to the prefix of C including only those events that necessarily precede x in C (and x itself). This motivates the following definition.

▶ **Definition 2.3** (history). *Let E be an event structure, let $C \in \mathit{Conf}(\mathsf{E})$ and let $x \in C$. The history of x in C is defined as the set $C[x] = \{y \in C \mid y \leq_C x\}$ endowed with the restriction of $\leq_C$ to $C[x]$, i.e., $\leq_{C[x]} = \leq_C \cap (C[x] \times C[x])$. The set of histories in E is $\mathit{Hist}(\mathsf{E}) = \{C[x] \mid C \in \mathit{Conf}(\mathsf{E}) \ \wedge \ x \in C\}$. The set of histories of a specific event $x \in \mathsf{E}$ will be denoted by $\mathit{Hist}(x)$.*

We mentioned that various generalisations of PESs in the literature can be naturally viewed as subclasses of poset event structures. Verifying that the corresponding families of configurations satisfy the properties of Definition 2.2 is easy. We briefly discuss prime and flow event structures (more details are in [6], where also other models are discussed).

Prime event structures. Prime event structures [24] capture the dependencies between events in terms of causality and conflict.

▶ **Definition 2.4** (prime event structure). *A* prime event structure *(PES, for short) is a tuple $\mathsf{P} = \langle E, \leq, \#, \lambda \rangle$, where E is a set of events, $\leq$ and $\#$ are binary relations on E called* causality *and* conflict*, respectively, and $\lambda : E \to \Lambda$ is a labelling function, such that*

- *$\leq$ is a partial order and $\lfloor x \rfloor = \{y \in E \mid y \leq x\}$ is finite for all $x \in E$;*
- *$\#$ is irreflexive, symmetric and hereditary with respect to causality, i.e., for all $x, y, z \in E$, if $x \# y$ and $y \leq z$ then $x \# z$.*

Configurations are sets of events without conflicts and closed with respect to causality. For later use, we also introduce a notation for the absence of conflicts, referred to as consistency.

▶ **Definition 2.5** (consistency, configuration). *Given a PES $\mathsf{P} = \langle E, \leq, \#, \lambda \rangle$, say that $x, y \in E$ are* consistent*, written $x \frown y$, when $\neg(x \# y)$. A subset $X \subseteq E$ is called* consistent*, written ${\frown} X$, when its elements are pairwise consistent. A* configuration *of P is a finite set of events $C \subseteq E$ such that (i) ${\frown} C$ and (ii) for all $x \in C$, $\lfloor x \rfloor \subseteq C$.*

Some examples of PESs can be found in Fig. 2. Causality is represented as a solid arrow, while conflict is represented as a dotted line. For instance, in P_0, event a_1 is a cause of b_1 and it is in conflict both with a_2 and b_3. Only direct causalities and non-inherited conflicts are represented. For instance, in P_0, the conflicts $a_1 \# b_2$, $a_2 \# b_1$ and $b_1 \# b_2$ are not represented since they are inherited. The labelling is implicitly represented by naming the events by their label, possibly with some index. For instance, a_1 and a_2 are events labelled by a.

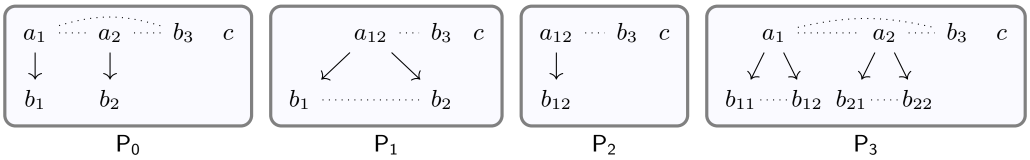

Figure 2 Some prime event structures.

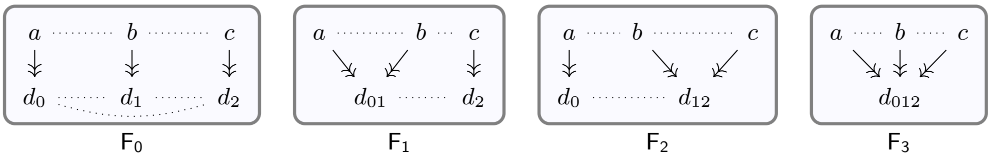

Figure 3 Some flow structures.

Clearly PESs can be seen as poset event structures. Given a PES $\mathsf{P} = \langle E, \leq, \#, \lambda\rangle$ and its set of configurations $\mathit{Conf}(\mathsf{P})$, the local order of a configuration $C \in \mathit{Conf}(\mathsf{P})$ is $\leq_C = \leq \cap (C \times C)$, i.e., the restriction of the causality relation to C. The extension order turns out to be simply subset inclusion. In fact, given $C_1 \subseteq C_2$, if $x_1 \in C_1$ and $x_2 \in C_2$, with $x_2 \leq_{C_2} x_1$, then necessarily $x_2 \in C_1$ since configurations are causally closed. Therefore, recalling that $\leq_{C_1} = \leq \cap (C_1 \times C_1)$ and $\leq_{C_2} = \leq \cap (C_2 \times C_2)$, we immediately conclude that $\leq_{C_2} \cap (C_2 \times C_1) = \leq \cap (C_2 \times C_1) = \leq \cap (C_1 \times C_1) = \leq_{C_1}$, as desired. As an example, the PES P_2 of Fig. 2, viewed as a poset event structure, can be found in Fig. 4.

Flow event structures. Flow event structures [9, 8] extend PESs with the possibility of modelling in a direct way multiple disjunctive and mutually exclusive causes for an event.

▶ **Definition 2.6** (flow event structure). *A* flow event structure *(FES) is a tuple $\langle E, \prec, \#, \lambda\rangle$, where E is a set of events, $\prec \subseteq E \times E$ is an irreflexive relation called the* flow relation*, $\# \subseteq E \times E$ is the* symmetric conflict *relation, and $\lambda : E \to \Lambda$ is a labelling function.*

Causality is replaced by an irreflexive (in general non transitive) flow relation $\prec$, intuitively representing immediate causal dependency. Moreover, conflict is no longer hereditary.

An event can have causes which are in conflict and these have a disjunctive interpretation, i.e., the event will be enabled by a maximal conflict-free subset of its causes.

▶ **Definition 2.7** (FES configuration). *Given a FES $\mathsf{F} = \langle E, \prec, \#, \lambda\rangle$, a configuration is a finite set of events $C \subseteq E$ such that (i) $\prec$ is acyclic on C, (ii) $\neg(x\#x')$ for all $x, x' \in C$ and (iii) for all $x \in C$ and $y \notin C$ with $y \prec x$, there exists $z \in C$ such that $y\#z$ and $z \prec x$.*

Some examples of FESs can be found in Fig. 3. Relation $\prec$ is represented by a double headed solid arrow. For instance, consider the FES F_1. The set $C = \{a, d_{01}\}$ is a configuration. We have $b \prec d_{01}$ and $b \notin C$, but this is fine since there is $a \in C$ such that $a\#b$ and $a \prec d_{01}$.

Under mild assumptions that exclude the presence of non-executable events (a condition referred to as fullness in [8]), FESs can be seen as poset event structures, by endowing each configuration C with a local order arising as the reflexive and transitive closure of the restriction of the flow relation to C, i.e., $\leq_C = (\prec \cap (C \times C))^*$.

2.2 Hereditary History Preserving Bisimilarity

In order to define hereditary history preserving bisimilarity, it is convenient to have an explicit representation of the transitions between configurations.

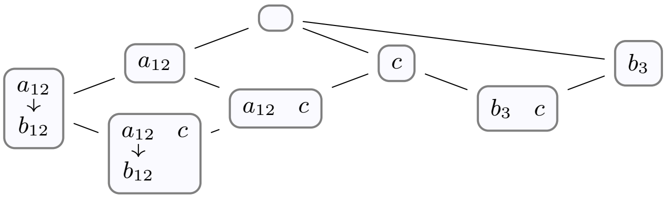

Figure 4 The configurations $\mathit{Conf}(\mathsf{P}_2)$ of the PES P_2 in Fig. 2 viewed as poset event structures.

▶ **Definition 2.8** (transition system). *Let* E *be an event structure. If* $C, C' \in \mathit{Conf}(\mathsf{E})$ *with* $C \sqsubseteq C'$ *we write* $C \xrightarrow{X} C'$ *where* $X = C' \setminus C$.

When X is a singleton, i.e., $X = \{x\}$, we will often write $C \xrightarrow{x} C'$ instead of $C \xrightarrow{\{x\}} C'$.

As it happens in the interleaving approach, a bisimulation between two event structures requires any event of an event structure to be simulated by an event of the other, with the same label. Additionally, the two events must have the same "causal history".

▶ **Definition 2.9** (hereditary history preserving bisimilarity). *Let* E, E' *be event structures. A* hereditary history preserving (hhp-)bisimulation *is a set* R *of triples* (C, f, C'), *where* $C \in \mathit{Conf}(\mathsf{E})$, $C' \in \mathit{Conf}(\mathsf{E}')$ *and* $f : C \to C'$ *is an isomorphism of configurations, such that* $(\emptyset, \emptyset, \emptyset) \in R$ *and for all* $(C_1, f, C_1') \in R$

1. *for all* $C_1 \xrightarrow{x} C_2$ *there exists some* $C_1' \xrightarrow{x'} C_2'$ *such that* $(C_2, f[x \mapsto x'], C_2') \in R$;
2. *for all* $C_1' \xrightarrow{x'} C_2'$ *there exists some* $C_1 \xrightarrow{x} C_2$ *such that* $(C_2, f[x \mapsto x'], C_2') \in R$;
3. *if* $C_2 \in \mathit{Conf}(\mathsf{E})$ *with* $C_2 \sqsubseteq C_1$ *then* $(C_2, f_{|C_2}, f(C_2)) \in R$ *(downward closure).*

Observe that, in the definition above, an event must be simulated by an event with the same label. In fact, in the triple $(C \cup \{x\}, f[x \mapsto x'], C' \cup \{x'\}) \in R$, the second component $f[x \mapsto x']$ must be an isomorphism of configurations, i.e., of labelled posets, and thus it preserves labels. Hhp-bisimilarity has been shown to arise as a canonical behavioural equivalence on prime event structures, as an instance of a general notion defined in terms of the concept of open map, when considering partially ordered computations as observations [18].

3 Foldings of Event Structures

In this section, we study a notion of folding, which is intended to formalise the intuition of a behaviour-preserving quotient for an event structure. We prove that there always exists a minimal quotient and we show that foldings between general poset event structures always arise, in a suitable formal sense, from foldings over prime event structures.

3.1 Morphisms and Foldings

We first endow event structures with a notion of morphism. In the sequel, given two event structures E, E', a function $f : E \to E'$ and a configuration $C \in \mathit{Conf}(\mathsf{E})$, we write $f(C)$ to refer to the poset whose underlying set is $\{f(x) \mid x \in C\}$, endowed with the order $f(x) \leq_{f(C)} f(y)$ iff $x \leq y$.

▶ **Definition 3.1** (morphism). *Let* E, E' *be event structures. A* (strong) morphism $f : \mathsf{E} \to \mathsf{E}'$ *is a function* $f : E \to E'$ *between the underlying sets of events such that* $\lambda = \lambda' \circ f$ *and for all configurations* $C \in \mathit{Conf}(\mathsf{E})$, *the function* f *is injective on* C *and* $f(C) \in \mathit{Conf}(\mathsf{E}')$.

Hereafter, the qualification "strong" will be omitted since this is the only kind of morphisms we deal with. It is motivated by the fact that normally morphisms on event structures are designed to represent simulations. If this were the purpose, then the requirement on preservation of configurations could have been weaker, i.e., we could have asked the order in the target configuration to be included in (not identical to) the image of the order of the source configuration and morphisms could have been partial. However, in our setting, for the objective of defining history-preserving quotients, the stronger notion works fine and simplifies the presentation.

▶ Remark 3.2. *The composition of morphisms is a morphism and the identity is a morphism. Hence the class of event structures and event structure morphisms form a category* $\mathbf{ES}$*.*

▶ **Definition 3.3** (folding). *Let* E *and* E' *be event structures. A* folding *is a morphism* $f : \mathsf{E} \to \mathsf{E}'$ *such that the relation* $R_f = \{(C, f_{|C}, f(C)) \mid C \in \mathit{Conf}(\mathsf{E})\}$ *is a hhp-bisimulation.*

In words, a folding is a function that "merges" some sets of events of an event structure into single events without altering the behaviour modulo hhp-bisimilarity. In [2] the notion of folding is given by requiring the preservation of hp-bisimilarity, a weaker behavioural equivalence defined as hhp-bisimilarity but omitting the requirement of downward-closure (condition 3 in Definition 2.9). Note that, as far as the notion of folding is concerned, this makes no difference: R_f is downward-closed by definition, hence it is a hhp-bisimulation whenever it is a hp-bisimulation. Instead, taking hhp-bisimilarity as the reference equivalence appears to be the right choice for the development of the theory. For instance, it allows one to prove Lemma 3.12 that plays an important role for arguing about the adequateness of the notion of folding (e.g., it is essential for Proposition 3.13). Interestingly, foldings can be characterised as open maps in the sense of [18], by taking conflict free prime event structures as subcategory of observations. This is explicitly worked out in [6].

As an example, consider the PESs in Fig. 2 and the function $f_{02} : \mathsf{P}_0 \to \mathsf{P}_2$ that maps events as suggested by the indices, i.e., $f_{02}(a_1) = f_{02}(a_2) = a_{12}$, $f_{02}(b_1) = f_{02}(b_2) = b_{12}$, $f_{02}(b_3) = b_3$ and $f_{02}(c) = c$. It is easy to see that f_{02} is a folding. Note that, instead, $f_{01} : \mathsf{P}_0 \to \mathsf{P}_1$, again mapping events according to their indices, is not a folding. In fact, $f_{01}(\{a_1\}) = \{a_{12}\} \xrightarrow{b_2} \{a_{12}, b_2\}$, but clearly there is no transition $\{a_1\} \xrightarrow{x}$ with $f_{01}(x) = b_2$, since the only preimage of b_2 in P_0 is b_2.

Observe that the greater expressiveness of FESs allows one to obtain smaller quotients. For instance, consider Fig. 3. The FES F_0 seen as a PES would be minimal. Instead, in the class of FESs it is not: the obvious functions from F_0 to F_1 and F_2 are foldings.

▶ Remark 3.4. *The composition of foldings is a folding and the identity is a folding. We can consider a subcategory* $\mathbf{ES_f}$ *of* $\mathbf{ES}$ *with the same objects and foldings as morphisms.*

Consider again the PESs in Fig. 2 and the morphisms $f_{30} : \mathsf{P}_3 \to \mathsf{P}_0$ and $f_{02} : \mathsf{P}_0 \to \mathsf{P}_2$. These are induced by the labelling, apart for the b_{ij} for which we let $f_{30}(b_{ij}) = b_i$. Both are foldings: the first merges b_{11} with b_{12} and b_{21} with b_{22}, while the second merges a_1 with a_2 and b_1 with b_2. Their composition $f_{32} = f_{30} \circ f_{02} : \mathsf{P}_3 \to \mathsf{P}_2$ is again a folding.

A simple but crucial result shows that the target event structure for a folding is completely determined by the mapping on events. We first define the quotient induced by a morphism.

▶ **Definition 3.5** (quotients from morphisms). *Let* E, E' *be event structures and let* $f : \mathsf{E} \to \mathsf{E}'$ *be a morphism. Let* $\equiv_f$ *be the equivalence relation on* E *defined by* $x \equiv_f y$ *if* $f(x) = f(y)$*. We denote by* $\mathsf{E}_{/\equiv_f}$ *the event structure with configurations* $\mathit{Conf}(\mathsf{E}_{/\equiv_f}) = \{[C]_{\equiv_f} \mid C \in \mathit{Conf}(\mathsf{E})\}$ *where* $[C]_{\equiv_f} = \{[x]_{\equiv_f} \mid x \in C\}$ *is ordered by* $[x]_{\equiv_f} \leq_{[C]_{\equiv_f}} [y]_{\equiv_f}$ *iff* $x \leq_C y$*.*

It is immediate to see that $\mathsf{E}_{/\equiv_f}$ is a well-defined event structure.

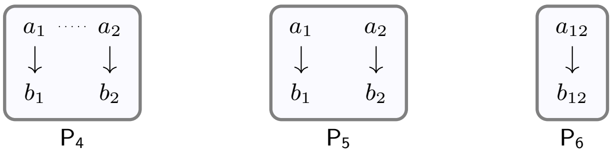

Figure 5 Non existence of pushout of general morphisms.

▶ **Lemma 3.6** (folding as equivalences). *Let E, E' be event structures and let $f : \mathsf{E} \to \mathsf{E}'$ be a morphism. If f is a folding then $\mathsf{E}_{/\equiv_f}$ is isomorphic to E'.*

The previous result allows us to identify foldings with the corresponding equivalences on the source event structures and motivates the following definition.

▶ **Definition 3.7** (folding equivalences). *Let E be an event structure. The set of folding equivalences over E is $FEq(\mathsf{E}) = \{\equiv_f \mid f : \mathsf{E} \to \mathsf{E}'$ folding for some $\mathsf{E}'\}$.*

Hereafter, we will freely switch between the two views of foldings as morphisms or as equivalences, since each will be convenient for some purposes.

We next observe that given two foldings we can always take their "join", providing a new folding that, roughly speaking, produces a quotient smaller than both the original ones.

▶ **Proposition 3.8** (joining foldings). *Let $\mathsf{E}, \mathsf{E}', \mathsf{E}''$ be event structures and let $f' : \mathsf{E} \to \mathsf{E}'$, $f'' : \mathsf{E} \to \mathsf{E}''$ be foldings. Define E''' as the quotient $\mathsf{E}_{/\equiv}$ where $\equiv$ is the transitive closure of $\equiv_{f'} \cup \equiv_{f''}$. Then $g' : \mathsf{E}' \to \mathsf{E}'''$ defined by $g'(x') = [x]_\equiv$ if $f'(x) = x'$ and $g'' : \mathsf{E}'' \to \mathsf{E}'''$ defined by $g''(x'') = [x]_\equiv$ if $f''(x) = x''$ are foldings.*

As an example, consider the PES in Fig. 2 and two morphisms $f_{30} : \mathsf{P}_3 \to \mathsf{P}_0$ and $f_{31} : \mathsf{P}_3 \to \mathsf{P}_1$. The way all events are mapped by f_{30} and f_{31} is naturally suggested by their labelling, apart for the b_{ij} for which we let $f_{30}(b_{ij}) = b_i$ while $f_{31}(b_{ij}) = b_j$. It can be seen that both are foldings. Their join, constructed as in Proposition 3.8, is P_2 with the folding morphisms $f_{02} : \mathsf{P}_0 \to \mathsf{P}_2$ and $f_{12} : \mathsf{P}_1 \to \mathsf{P}_2$.

▶ Remark 3.9. *Proposition 3.8 is a consequence of the fact that the category* **ES** *has pushouts of foldings. Indeed, E''' as defined above is the pushout of f' and f'' (in* **ES** *and also in* $\mathbf{ES_f}$*). It can be seen that, in general,* **ES** *does not have all pushouts.*

As a counterexample to the existence of pushouts in **ES** for general morphisms, consider the obvious mappings $f_{45} : \mathsf{P}_4 \to \mathsf{P}_5$ and $f_{46} : \mathsf{P}_4 \to \mathsf{P}_6$ in Fig. 5. It is easy to realise that, if a pushout existed, the mapping from P_5 into the pushout object should identify the concurrent events a_1 and a_2, failing to be an event structure morphism.

When interpreted in the setting of folding equivalences of an event structure, Proposition 3.8 has a clear meaning. Recall that the equivalences over some fixed set X, ordered by inclusion, form a complete lattice, where the top element is the universal equivalence $X \times X$ and the bottom is the identity on X. Then Proposition 3.8 implies that $FEq(\mathsf{E})$ is a sublattice of the lattice of equivalences. Actually, it can be shown that $FEq(\mathsf{E})$ is itself a complete lattice. Therefore each event structure E admits a maximally folded version.

▶ **Theorem 3.10** (lattice of foldings). *Let E be an event structure. Then $FEq(\mathsf{E})$ is a sublattice of the complete lattice of equivalence relations over E.*

▶ Remark 3.11. *The above result arises from a generalisation of Proposition 3.8 showing that, for any event structure E, each collection of foldings $f_i : \mathsf{E} \to \mathsf{E}_i$, with $i \in I$, admits a colimit in* **ES**. *Thus the coslice category $(\mathsf{E} \downarrow \mathbf{ES_f})$ has a terminal object, which is the maximally folded event structure.*

It is natural to ask whether behaviour-preserving quotients correspond to foldings. Strictly speaking, the answer is negative. More precisely, there can be morphisms $f : \mathsf{E} \to \mathsf{E}'$ such that $\mathsf{E}_{/\equiv_f}$ is hhp-bisimilar to E, but f is not a folding. For an example, consider the PESs P_0 and P_1 in Fig. 2 and the morphism $f_{01} : \mathsf{P}_0 \to \mathsf{P}_1$ suggested by the indexing. We already observed this is not a folding, but $\mathsf{P}_{0/\equiv_{f_{01}}}$, which is isomorphic to P_1, is hhp-bisimilar to P_0. However, we can show that for any behaviour-preserving quotient, there is a folding that produces a coarser equivalence, and thus a smaller quotient. For instance, in the example discussed above, there is the folding $f_{02} : \mathsf{P}_0 \to \mathsf{P}_2$, that “produces” a smaller quotient.

This follows from the possibility of joining foldings (Proposition 3.8) and the fact that a hhp-bisimulation can be always seen as an event structure, a result that generalises to our setting a property proved for PESs in [7].

▶ **Lemma 3.12** (hhp-bisimulation as an event structure). *Let E', E'' be event structures and let R be a hhp-bisimulation between them. Then there exists a (prime) event structure E_R and two foldings $\pi' : \mathsf{E}_R \to \mathsf{E}'$ and $\pi'' : \mathsf{E}_R \to \mathsf{E}''$.*

▶ **Proposition 3.13** (foldings subsume behavioural quotients). *Let E be an event structure and let $f : \mathsf{E} \to \mathsf{E}'$ be a morphism such that $\mathsf{E}_{/\equiv_f}$ is hhp-bisimilar to E. Then there exists a folding $g : \mathsf{E} \to \mathsf{E}''$ such that $\equiv_g$ is coarser than $\equiv_f$.*

The proof relies on the possibility of joining foldings (Proposition 3.8) and the fact that a hhp-bisimulation can be always seen as an event structure, a result that generalises to our setting a property proved for PESs in [7].

3.2 Folding through Prime Event Structures

We observe that each event structure is the folding of a corresponding canonical PES. We then prove that, interestingly enough, all foldings between event structures arise from foldings of the corresponding canonical PESs.

We start with the definition of the canonical PES associated with an event structure.

▶ **Definition 3.14** (PES for an event structure). *Let E be an event structure. Its canonical PES is $\mathbb{P}(\mathsf{E}) = \langle \mathit{Hist}(\mathsf{E}), \sqsubseteq, \#, \lambda' \rangle$ where $\sqsubseteq$ is prefix, $\#$ is incompatibility, i.e., for $H_1, H_2 \in \mathit{Hist}(\mathsf{E})$ we let $H_1 \# H_2$ if $\neg(H_1 \frown H_2)$, and $\lambda'(H) = \lambda(x)$ when $H \in \mathit{Hist}(x)$. Given a morphism $f : \mathsf{E} \to \mathsf{E}'$ we write $\mathbb{P}(f) : \mathbb{P}(\mathsf{E}) \to \mathbb{P}(\mathsf{E}')$ for the morphism defined by $\mathbb{P}(f)(H) = f(H)$.*

It can be easily seen that the definition above is well-given. In particular, $\mathbb{P}(\mathsf{E})$ is a well-defined PES because, as proved in [26], a family of posets ordered by prefix is finitary coherent prime algebraic domain. Then the tight relation between this class of domains and PES highlighted in [33] allows one to conclude the proof. For instance, in Fig. 1(right) one can find the canonical PES for the event structure on the left.

The canonical PES associated with an event structure can always be folded to the original event structure.

▶ **Lemma 3.15** (unfolding event structures to PES's). *Let E be an event structure. Define a function $\phi_{\mathsf{E}} : \mathbb{P}(\mathsf{E}) \to \mathsf{E}$, for all $H \in \mathit{Hist}(\mathsf{E})$ by $\phi_{\mathsf{E}}(H) = x$ if $H \in \mathit{Hist}(x)$ for $x \in \mathsf{E}$. Then ϕ_{E} is a folding.*

We next show that any morphism from a PES to an event structure E factorises uniquely through the PES $\mathbb{P}(\mathsf{E})$ associated with E (categorically, ϕ_{E} is cofree over E). The same applies to foldings and it will be useful to relate foldings in E with foldings in $\mathbb{P}(\mathsf{E})$.

▶ **Lemma 3.16** (cofreeness of ϕ_{E}). *Let* E *be an event structure, let* P' *be a* PES *and let* $f : \mathsf{P}' \to \mathsf{E}$ *be an event structure morphism. Then there exists a unique morphism* $g : \mathsf{P}' \to \mathbb{P}(\mathsf{E})$ *such that* $f = \phi_{\mathsf{E}} \circ g$*.*

$$\begin{tikzcd}
\mathbb{P}(\mathsf{E}) \arrow[r, "\phi_{\mathsf{E}}"] & \mathsf{E} \\
\mathsf{P}' \arrow[u, dotted, "g"] \arrow[ur, "f"'] &
\end{tikzcd}$$

Moreover, when f *is a folding then so is* g*.*

▶ Remark 3.17. *Lemma 3.16 means that the category* **PES** *of prime event structures is a coreflective subcategory of* **ES***, i.e.,* $\mathbb{P} : \mathbf{ES} \to \mathbf{PES}$ *can be seen as a functor, right adjoint to the inclusion* $\mathbb{I} : \mathbf{PES} \to \mathbf{ES}$*. Moreover,* $\mathbb{P}$ *restricts to a functor on the subcategory of foldings,* $\mathbb{P} : \mathbf{ES_f} \to \mathbf{PES_f}$*, where an analogous result holds.*

We conclude that all foldings between event structures arise from foldings of the associated PESs. Given that **PES** is a coreflective subcategory of **ES** and foldings can be seen as open maps, this result (and also the fact that morphisms ϕ_{E} are foldings) can be derived from [18, Lemma 6]. More details on this can be found in the extended version [6].

▶ **Proposition 3.18** (folding through PES's). *Let* E, E' *be event structures. For all morphisms* $f : \mathsf{E} \to \mathsf{E}'$ *consider* $\mathbb{P}(f) : \mathbb{P}(\mathsf{E}) \to \mathbb{P}(\mathsf{E}')$ *defined by* $\mathbb{P}(f)(H) = f(H)$*. Then* f *is a folding iff* $\mathbb{P}(f)$ *is a folding.*

4 Foldings for Prime Event Structures

Motivated by the fact that foldings on general poset event structures always arise from foldings of the corresponding canonical PESs, in this section we study foldings in the class of PESs. We provide a characterisation and show that also PESs always admit a least quotient.

Since foldings are special morphisms, we first provide a characterisation of PES morphisms.

▶ **Lemma 4.1** (PES morphisms). *Let* P *and* P' *be* PES*s and let* $f : P \to P'$ *be a function on the underlying sets of events. Then* f *is a morphism iff for all* $x, y \in \mathsf{P}$

1. $\lambda'(f(x)) = \lambda(x)$*;*
2. $f(\lfloor x \rfloor) = \lfloor f(x) \rfloor$*; namely (a) for all* $z' \in \mathsf{P}'$*, if* $z' \leq f(x)$ *there exists* $z \in \mathsf{P}$ *such that* $z \leq x$ *and* $f(z) = z'$ *(b) if* $z \leq x$ *then* $f(z) \leq f(x)$*;*
3. *(a) if* $f(x) = f(y)$ *and* $x \neq y$ *then* $x \# y$ *and (b) if* $f(x) \# f(y)$ *then* $x \# y$*.*

These are the standard conditions characterising (total) PES morphisms (see, e.g., [33]), with the addition of condition (2b) that is imposed to ensure that configurations are mapped to isomorphic configurations, as required by the notion of (strong) morphism (Definition 3.1).

We know that not all PES morphisms are foldings. We next identify some additional conditions characterising those morphisms which are foldings. Given a relation $r \subseteq X \times X$ and $Y, Z \subseteq X$ we write $Y\ r^{\forall}\ Z$ if for all $y \in Y$ and $z \in Z$ it holds $y\ r\ z$. Singletons are replaced by their only element, writing, e.g., $y\ r^{\forall}\ Z$ for $\{y\}\ r^{\forall}\ Z$.

▶ **Theorem 4.2** (PES foldings). *Let* P *and* P' *be* PES*s and let* $f : \mathsf{P} \to \mathsf{P}'$ *be a morphism. Then* f *is a folding if and only if it is surjective and for all* $X, Y \subseteq \mathsf{P}$*,* $x, y \in \mathsf{P}$*,* $y' \in \mathsf{P}'$

1. *if* $x \#^{\forall} f^{-1}(y')$ *then* $f(x) \# y'$*;*
2. *if* $\frown(X \cup \{x\})$*,* $\frown(Y \cup \{y\})$*,* $\frown(X \cup Y)$ *and* $f(x) = f(y)$ *then there exists* $z \in \mathsf{P}$ *such that* $f(z) = f(x)$ *and* $\frown(X \cup Y \cup \{z\})$*.*

The notion of folding on PESs turns out to be closely related to that of abstraction homomorphism for PESs introduced in [12] for similar purposes. More precisely, abstraction homomorphisms can be characterised as those PES morphisms additionally satisfying condition (1) of Theorem 4.2, while they do not necessarily satisfy condition (2). Their more liberal definition with respect to foldings can be explained by the fact that they are designed to work on a subclass of structured PESs (a formal comparison is in the extended version [6]).

The next result "transfers" the conditions characterising foldings to folding equivalences.

▶ **Corollary 4.3** (folding equivalences for PES's). *Let* P *be a* PES *and let* $\equiv$ *be an equivalence on* P*. Then* $\equiv$ *is a folding equivalence in* $FEq(\mathsf{P})$ *iff for all* $x, y \in \mathsf{P}$, $x \neq y$, *if* $x \equiv y$ *then*

1. $\lambda(x) = \lambda(y)$*;*
2. $[\lfloor x \rfloor]_{\equiv} = [\lfloor y \rfloor]_{\equiv}$*;*
3. $x \# y$*.*

Moreover, for all $x, y \in \mathsf{P}$, $X, Y \subseteq \mathsf{P}$

4. *if* $x \#^{\forall} [y]_{\equiv}$ *then* $[x]_{\equiv} \#^{\forall} [y]_{\equiv}$*;*
5. *if* $\frown(X \cup \{x\})$, $\frown(Y \cup \{y\})$, $\frown(X \cup Y)$ *there exists* $z \in [x]_{\equiv}$ *such that* $\frown(X \cup Y \cup \{z\})$*.*

For instance, in Fig. 2, consider the equivalence $\equiv_{01}$ over P_0 such that $a_1 \equiv_{01} a_2$. This produces P_1 as quotient. This is not a folding equivalence since condition (4) fails: $a_1 \#^{\forall} [b_2]_{\equiv_{01}}$, but $\neg(a_2 \# b_2)$ and thus $\neg([a_1]_{\equiv_{01}} \#^{\forall} [b_2]_{\equiv_{01}})$. Instead, the equivalence $\equiv_{02}$ over P_0 such that $a_1 \equiv_{02} a_2$ and $b_1 \equiv_{02} b_2$, producing P_2 as quotient, satisfies all five conditions.

When PESs are finite, the result above suggests a possible way of identifying foldings: one can pair candidate events to be folded on the basis of conditions (1)-(3) and then try to extend the sets with condition (4)-(5) when possible. The procedure can be inefficient due to the global flavor of the conditions. This will be further discussed in the conclusions.

We know, from Proposition 3.8, that all event structures admit a "maximally folded" version. We next observe that the same result continues to hold in the class of PESs.

▶ **Theorem 4.4** (joining foldings on PES's). *Let* $\mathsf{P}, \mathsf{P}', \mathsf{P}''$ *be* PES*s and let* $f' : \mathsf{P} \to \mathsf{P}'$, $f'' : \mathsf{P} \to \mathsf{P}''$ *be foldings. Define* E''' *along with* $g' : \mathsf{P}' \to \mathsf{E}'''$ *and* $g'' : \mathsf{P}'' \to \mathsf{E}'''$ *as in Proposition 3.8. Then* E''' *is a* PES*. As a consequence, each* PES *admits a uniquely determined minimal quotient in the class of* PES*s.*

▶ Remark 4.5. *Theorem 4.4 is a consequence of the fact that the subcategory* $\mathbf{PES_f}$ *is a coreflective subcategory of* $\mathbf{ES_f}$ *and thus it is closed under colimits.*

In passing, we note that in the class of FESs the existence of a unique minimal folding is lost. In fact, consider Fig. 3. It can be easily seen that F_1 and F_2 are different minimal foldings of F_0. In particular, merging the three d-labelled events as done in F_3 modifies the behaviour. In fact, in F_3, the event d_{012} is not enabled in $C = \{a\}$ since $c \prec d_{012}$ and no event in C is in conflict with c. Instead, in F_0, the event d_0 is clearly enabled from $\{a\}$. Existence of a unique minimal folding could be possibly recovered by strengthening the notion of folding. Note, however, that this would be against the spirit of our work where the notion of folding is not a choice. Rather, after having assumed hhp-bisimilarity as the reference behavioural equivalence, the notion of folding is essentially "determined" as a quotient (surjective function) that preserves the behaviour up to hhp-bisimilarity.

5 Conclusions

We studied the problem of minimisation for poset event structures, a class that encompasses many stable event structure models in the literature, taking hereditary history preserving bisimilarity as reference behavioural equivalence. We showed that a uniquely determined minimal quotient always exists for poset event structures and also in the subclass of PESs. We showed that foldings between general poset event structures arise from foldings of corresponding canonical PESs. Finally, we provided a characterisation of foldings of PESs.

We believe that, besides its original motivations from the setting of business process models and its foundational interest, this work can be of help in the study of minimisation, under a true concurrent equivalence, of operational models which can be mapped to event structures, like transition systems with independence or Petri nets.

As underlined throughout the paper, our theory of folding has many connections with the literature on event structures. The idea of "unfolding" more expressive models to prime algebraic domains and PESs has been studied by many authors (e.g., in [26, 24, 28, 30, 9]). The same can be said for the idea of refining a single action into a complex computation (see, e.g., [29] and references therein). Instead, the problem of characterising behaviour-preserving quotients of event structures has received less attention. We already commented on the relation with abstraction homomorphisms for PESs [12], which capture the idea of behaviour-preserving abstraction but only in a subclass of structured PESs. In some cases, given a Petri net or an event structure a special transition system can be extracted, on which minimisation is performed. In particular, the paper [23] proposes an encoding of safe Petri nets into causal automata, preserving hp-bisimilarity. Such automata can be transformed into a standard labelled transition systems, which in turn can be minimised. However, in this way, the correspondence with the original events is lost.

The notion of behaviour-preserving function has been given an elegant abstract characterisation in terms of open maps [18]. We mentioned the possibility of viewing our foldings as open maps and we observed that various results admit a categorical interpretation (see also [6]). This gives clear indications of the possibility of providing a general abstract view of the results in this paper, something which represents an interesting topic of future research.

The characterisation of foldings on PESs can be used as a basis to develop, at least in the case of finite structures, algorithms for the construction of behaviour preserving quotients. The fact that conditions for folding refer to sets of events might make the minimisation procedure very inefficient. Identifying suitable heuristics and investigating the possibility of having more "local" folding conditions are interesting directions of future work.

Although not explicitly discussed in the paper, by considering elementary foldings, i.e., foldings that just merge a single set of events, one can indeed determine some more efficient folding rules. This is essentially what is done for AESs and FESs in [2]. However, restricting to elementary foldings is limiting, since it can be seen that general foldings cannot be always decomposed in terms of elementary ones (e.g., it can be seen that in Fig. 2, the folding $f_{02} : \mathsf{P}_0 \to \mathsf{P}_2$ cannot be obtained as the composition of elementary foldings).

When dealing with possibly infinite event structures one could try to devise reduction rules acting on some finitary representation and inducing foldings on the corresponding event structure. Note, however, that working, e.g., on finite safe Petri nets, the minimisation procedure would be necessarily incomplete, given that hhp-bisimilarity is undecidable [19].

References

1 A. Armas-Cervantes, P. Baldan, M. Dumas, and L. García-Bañuelos. Diagnosing behavioral differences between business process models: An approach based on event structures. *Information Systems*, 56:304–325, 2016.

2 A. Armas-Cervantes, P. Baldan, and L. García-Bañuelos. Reduction of event structures under history preserving bisimulation. *Journal of Logical and Algebraic Methods in Programming*, 85(6):1110–1130, 2016.

3 P. Baldan. *Modelling concurrent computations: from contextual Petri nets to graph grammars*. PhD thesis, University of Pisa, 2000.

4 P. Baldan, A. Corradini, H. Ehrig, M. Löwe, U. Montanari, and F. Rossi. Concurrent Semantics of Algebraic Graph Transformation Systems. In G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformation*, volume III: Concurrency, pages 107–187. World Scientific, 1999.

5 P. Baldan, A. Corradini, and U. Montanari. Contextual Petri nets, asymmetric event structures and processes. *Information and Computation*, 171(1):1–49, 2001.

6 P. Baldan and A. Raffaetà. Minimising Event Structures, 2019. arXiv:1907.07042.

7 M.A. Bednarczyk. Hereditary History Preserving Bisimulations or What is the Power of the Future Perfect in Program Logics. Technical report, Polish Academy of Sciences, 1991.

8 G. Boudol. Flow Event Structures and Flow Nets. In *Semantics of System of Concurrent Processes*, volume 469 of *LNCS*, pages 62–95. Springer Verlag, 1990.

9 G. Boudol and I. Castellani. Permutation of transitions: an event structure semantics for CCS and SCCS. In *Linear Time, Branching Time and Partial Order Semantics in Logics and Models for Concurrency*, volume 354 of *LNCS*, pages 411–427. Springer Verlag, 1988.

10 R. Bruni, H.C. Melgratti, and U. Montanari. Event Structure Semantics for Nominal Calculi. In C. Baier and H. Hermanns, editors, *CONCUR 2006*, volume 4137 of *LNCS*, pages 295–309. Springer, 2006.

11 S. Castellan. Weak memory models using event structures. In *Proceedings of JFLA'16*, 2016.

12 I. Castellani. *Bisimulations for Concurrency*. PhD thesis, University of Edimburgh, 1988.

13 S. Chakraborty and V. Vafeiadis. Grounding thin-air reads with event structures. *PACMPL*, 3(POPL):70:1–70:28, 2019.

14 I. Cristescu, J. Krivine, and D. Varacca. Rigid families for CCS and the pi-calculus. In *Proceedings of ICTAC'15*, volume 9399 of *LNCS*, pages 223–240. Springer, 2015.

15 M. Dumas and L. García-Bañuelos. Process Mining Reloaded: Event Structures as a Unified Representation of Process Models and Event Logs. In R.R. Devillers and A. Valmari, editors, *Petri Nets 2015*, volume 9115 of *LNCS*, pages 33–48. Springer, 2015.

16 J.E. Hopcroft, R. Motwani, and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 2006.

17 A. Jeffrey and J. Riely. On Thin Air Reads: Towards an Event Structures Model of Relaxed Memory. In M. Grohe, E. Koskinen, and N. Shankar, editors, *LICS 2016*, pages 759–767. ACM, 2016.

18 A. Joyal, M. Nielsen, and G. Winskel. Bisimulation from Open Maps. *Information and Computation*, 127(2):164–185, 1996.

19 M. Jurdzinski, M. Nielsen, and J. Srba. Undecidability of domino games and hhp-bisimilarity. *Information and Computation*, 184(2):343–368, 2003.

20 R. Langerak. Bundle Event Structures: A Non-Interleaving Semantics for Lotos. In *5th Intl. Conf. on Formal Description Techniques (FORTE'92)*, pages 331–346. North-Holland, 1992.

21 R. Langerak. *Transformation and Semantics for LOTOS*. PhD thesis, Department of Computer Science, University of Twente, 1992.

22 A.R. Meyer and L.J. Stockmeyer. The Equivalence Problem for Regular Expressions with Squaring Requires Exponential Space. In *SWAT (FOCS)*, pages 125–129. IEEE Computer Society, 1972.

23 U. Montanari and M. Pistore. Minimal Transition Systems for History-Preserving Bisimulation. In *14th Annual Symposium on Theoretical Aspects of Computer Science*, volume 1200 of *LNCS*, pages 413–425. Springer Verlag, 1997.

24 M. Nielsen, G. Plotkin, and G. Winskel. Petri Nets, Event Structures and Domains, Part 1. *Theoretical Computer Science*, 13:85–108, 1981.

25 J. Pichon-Pharabod and P. Sewell. A concurrency semantics for relaxed atomics that permits optimisation and avoids thin-air executions. In R. Bodík and R. Majumdar, editors, *POPL 2016*, pages 622–633. ACM, 2016.

26 A. Rensink. Posets for Configurations! In W. R. Cleaveland, editor, *Proceedings of CONCUR'92*, volume 630 of *LNCS*, pages 269–285. Springer, 1992.

27 G. Schied. On relating Rewriting Systems and Graph Grammars to Event Structures. In H.-J. Schneider and H. Ehrig, editors, *Dagstuhl Seminar 9301 on Graph Transformations in Computer Science*, volume 776 of *LNCS*, pages 326–340. Springer, 1994.

28 R.J. van Glabbeek. History preserving process graphs. Draft available at `http://theory.stanford.edu/~rvg/abstracts.html#hppg`, 1996.

29 R.J. van Glabbeek and U. Goltz. Refinement of actions and equivalence notions for concurrent systems. *Acta Informatica*, 37(4/5):229–327, 2001.

30 R.J. van Glabbeek and G.D. Plotkin. Configuration structures, event structures and Petri nets. *Theoretical Computer Science*, 410(41):4111–4159, 2009.

31 D. Varacca and N. Yoshida. Typed event structures and the linear pi-calculus. *Theoretical Computer Science*, 411(19):1949–1973, 2010.

32 G. Winskel. Event Structure Semantics for CCS and Related Languages. Technical Report DAIMI PB-159, University of Aarhus, 1983.

33 G. Winskel. Event Structures. In *Petri Nets: Applications and Relationships to Other Models of Concurrency*, volume 255 of *LNCS*, pages 325–392. Springer, 1987.

34 G. Winskel. Events, Causality and Symmetry. *Computer Journal*, 54(1):42–57, 2011.