A Logic with Reverse Modalities for History-preserving Bisimulations

Iain Phillips
Department of Computing, Imperial College London, England
iccp@doc.ic.ac.uk

Irek Ulidowski
Department of Computer Science, University of Leicester, England
iu3@mcs.le.ac.uk

We introduce event identifier logic (EIL) which extends Hennessy-Milner logic by the addition of (1) reverse as well as forward modalities, and (2) identifiers to keep track of events. We show that this logic corresponds to hereditary history-preserving (HH) bisimulation equivalence within a particular true-concurrency model, namely stable configuration structures. We furthermore show how natural sublogics of EIL correspond to coarser equivalences. In particular we provide logical characterisations of weak history-preserving (WH) and history-preserving (H) bisimulation. Logics corresponding to HH and H bisimulation have been given previously, but not to WH bisimulation (when autoconcurrency is allowed), as far as we are aware. We also present characteristic formulas which characterise individual structures with respect to history-preserving equivalences.

1 Introduction

The paper presents a modal logic that can express simple properties of computation in the true concurrency setting of stable configuration structures. We aim, like Hennessy-Milner logic (HML) [19] in the interleaving setting, to characterise the main true concurrency equivalences and to develop characteristic formulas for them. We focus in this paper on history-preserving bisimulation equivalences.

HML has a "diamond" modality $\langle a\rangle\phi$ which says that an event labelled a can be performed, taking us to a new state which satisfies ϕ. The logic also contains negation ($\neg$), conjunction ($\wedge$) and a base formula which always holds (tt). HML is strong enough to distinguish any two processes which are not bisimilar.

We are interested in making true concurrency distinctions between processes. These processes will be *event structures*, where the current state is represented by the set of events which have occurred so far. Such sets are called *configurations*. Events have labels (ranged over by $a, b, \ldots$), and different events may have the same label. We shall refer to example event structures using a CCS-like notation, with $a \,|\, b$ denoting an event labelled with a in parallel with another labelled with b, $a.b$ denoting two events labelled a and b where the first causes the second, and $a + b$ denoting two events labelled a and b which conflict.

In the true concurrency setting bisimulation is referred to as *interleaving bisimulation*, or IB for short. The processes $a \,|\, b$ and $a.b + b.a$ are interleaving bisimilar, but from the point of view of true concurrency they should be distinguished, and HML is not powerful enough to do this.

We therefore look for a more powerful logic, and we base this logic on adding reverse moves. Instead of the one modality $\langle a\rangle\phi$ we have two: *forward diamond* $\langle a\rangle\rangle\phi$ (which is just a new notation for the $\langle a\rangle\phi$ of HML) and *reverse diamond* $\langle\langle a\rangle\phi$. The latter is satisfied if we can reverse some event labelled with

a and get to a configuration where ϕ holds. Such an event would have to be *maximal* to enable us to reverse it, i.e. it could not be causing some other event that has already occurred.

With this new reverse modality we can now distinguish $a\,|\,b$ and $a.b+b.a$: $a\,|\,b$ satisfies $\langle a\rangle\rangle\langle b\rangle\rangle\langle\langle a\rangle\mathtt{tt}$, while $a.b+b.a$ does not. The formula expresses the idea that a and b are *concurrent*. Alternatively we see that $a.b+b.a$ satisfies $\langle a\rangle\rangle\langle b\rangle\rangle\neg\langle\langle a\rangle\mathtt{tt}$, while $a\,|\,b$ does not. This latter formula expresses the idea that *a causes b*.

The new logic corresponds to *reverse interleaving bisimulation* [31], or RI-IB for short. In the absence of autoconcurrency, Bednarczyk [3] showed that this is as strong as *hereditary history-preserving bisimulation* [3], or HH for short, which is usually regarded as the strongest desirable true concurrency equivalence. HH was independently proposed in [21], under the name of strong history-preserving bisimulation.

Auto-concurrency is where events can occur concurrently and have the same label. To allow for this, we need to strengthen the logic. For instance, we want to distinguish $a\,|\,a$ from $a.a$, which is not possible with the logic as it stands: $\langle a\rangle\rangle\langle a\rangle\rangle\langle\langle a\rangle\mathtt{tt}$ is satisfied by both processes. We need some way of distinguishing the two events labelled with a. We change our modalities so that when we make a forward move we *declare* an *identifier* (ranged over by $x,y,\ldots$) which stands for that event, allowing us to refer to it again when reversing it. Now we can write $\langle x:a\rangle\rangle\langle y:a\rangle\rangle\langle\langle x\rangle\mathtt{tt}$, and this is satisfied by $a\,|\,a$, but not by $a.a$. Declaration is an identifier-binding operation, so that x and y are both bound in the formula. Baldan and Crafa [2] also used such declarations in their forward-only logic.

With this simple change we now have a logic which is as strong as HH, even with autoconcurrency.

We have to be careful that our logic does not become too strong. For instance, we want to ensure that processes a and $a+a$ are indistinguishable. One might think that $a+a$ satisfies $\langle x:a\rangle\rangle\langle\langle x\rangle\langle y:a\rangle\rangle\neg\langle\langle x\rangle\mathtt{tt}$, while a does not. To avoid this, we need to ensure that x is forgotten about once it is reversed, and so cannot be used again. One could make a syntactic restriction that in a formula $\langle\langle x\rangle\phi$ the identifier x is not allowed to occur (free) in ϕ. However this is not actually necessary, as our semantics will ensure that all identifiers must be assigned to events in the current configuration. So in fact $\langle x:a\rangle\rangle\langle\langle x\rangle\langle y:a\rangle\rangle\neg\langle\langle x\rangle\mathtt{tt}$ is not satisfied by $a+a$, since we are not allowed to reverse x as it would take us to a configuration where x is mentioned in $\langle y:a\rangle\rangle\neg\langle\langle x\rangle\mathtt{tt}$ but x is assigned to an event outside the current configuration. Baldan and Crafa [2] also had to deal with this issue.

Our logic is not quite complete, since we wish to express certain further properties. For instance, we would like to express a reverse move labelled with a, i.e. $\langle\langle a\rangle\phi$. Instead of adding this directly, we add *declarations* $(x:a)\phi$. We can now express $\langle\langle a\rangle\phi$ by the formula $(x:a)\langle\langle x\rangle\phi$ (where x does not occur (free) in ϕ).

We also wish to express so-called *step transitions*, which are transitions consisting of multiple events occurring concurrently. For instance a forward step $\langle a,a\rangle\rangle\phi$ of two events labelled with a can be achieved by $\langle x:a\rangle\rangle\langle y:a\rangle\rangle(\phi\wedge\langle\langle x\rangle\mathtt{tt})$ and a reverse step $\langle\langle a,a\rangle\phi$ can be achieved by $(x:a)(y:a)(\langle\langle x\rangle\langle\langle y\rangle\phi\wedge\langle\langle y\rangle\mathtt{tt})$ (both formulas with x and y not free in ϕ). Thus the reverse steps employ declarations. As well as expressing reverse steps, declarations allow us to obtain a sublogic which corresponds to *weak history-preserving bisimulation* (WH).

This completes a brief introduction of our logic, which we call *Event Identifier Logic*, or EIL for short. Apart from corresponding to HH, EIL has natural sublogics for several other true concurrency equivalences. Figure 1 shows a hierarchy of equivalences that we are able to characterise, where arrows denote proper set inclusion. Apart from the mentioned HH and WH, *history-preserving bisimulation* (H) is a widely studied equivalence that employs history isomorphism. *Hereditary weak-history preserving bisimulation* (HWH) is WH with the hereditary property [3] that deals with reversing of events. The definitions of these equivalences can be found in [12, 31], and are outlined in Section 3.2.

$$\begin{tikzcd}
 & \mathrm{HH} \arrow[dl] \arrow[dr] & \\
\mathrm{H} \arrow[dr] & & \mathrm{HWH} \arrow[dl] \\
 & \mathrm{WH} &
\end{tikzcd}$$

Figure 1: The hierarchy of history-preserving equivalences.

It is natural to ask if, at least for a finite structure, there is a single logical formula which captures all of its behaviour, up to a certain equivalence. Such formulas are called *characteristic* formulas. They have been investigated previously for HML and other logics [16, 35, 1]. We look at characteristic formulas with respect to three of the equivalences we consider, namely HH, H and WH.

The main contribution of the paper is a logic EIL. It could be argued that EIL is a natural and canonical logic for the true concurrency equivalences considered here in the following sense. Firstly, its forward and reverse modalities capture faithfully the information of the forward and reverse transitions in the definitions of the equivalences, Secondly, event identifier environments and event declarations give rise naturally to order isomorphisms for HH, H, HWH and WH. Finally, EIL extends HML and keeps with its spirit of having simple modalities defined seamlessly over a general computation model.

Other contributions include the first to our knowledge logics for WH and HWH. Finally, we present the first to our knowledge characteristic formulas for HH, H and WH.

The paper is organised as follows. We look at related work in Section 2. Then we recall the definitions of configuration structures and the bisimulation-based equivalences that we shall need in Section 3. We then introduce EIL in Section 4, giving examples of its usage. Next we look at how to characterise various equivalences using EIL and its sublogics (Section 5). In Section 6 we investigate characteristic formulas. We finish with conclusions and future work.

2 Related work

Previous work on logics for true concurrency can be categorised loosely according to the type of semantic structure (model) that the satisfaction relation of the logic is defined for. There are logics over configurations (sets of consistent events) [15, 2] and logics over paths (or computations) [5, 27, 28, 29, 32], although logics in [27, 28, 29] can be seen also as logics over configurations. Other structures such as trees, graphs and Kripke frames are used as models in, for example, [26, 25, 17, 18].

The logic in this paper uses simple forward and reverse event identifier modalities that are sufficient to characterise HH. In contrast, Baldan and Crafa [2] achieved an alternative characterisation of HH with a different modal logic that uses solely forward-only event identifier modalities $\langle x \rangle$ and $(\boldsymbol{x}, \overline{\boldsymbol{y}} < \mathsf{a}\,z)$. The formula $(\boldsymbol{x}, \overline{\boldsymbol{y}} < \mathsf{a}\,z)\phi$ holds in a configuration if in its future there is an a-labelled event e that can be bound to z, and ϕ holds. Additionally, e must be (1) caused at least by the events already bound to the events in $\boldsymbol{x}$ and (2) concurrent with at least the events already bound to the events in $\boldsymbol{y}$. Several interesting sublogics were also identified in [2] that characterise H, pomset bisimulation [4, 12] and step bisimulation [33, 12] respectively.

Goltz, Kuiper and Penczek [15] researched configurations of prime event structures *without autoconcurrency*. In such a setting HH coincides with reverse interleaving bisimulation RI-IB (shown in [3]). Moreover, H coincides with WH. *Partial Order Logic* (POL) is proposed in [15]. POL contains past modalities and the authors stated that it characterises RI-IB (and thus HH). Also, it is conjectured that if one restricts POL in such a way that no forward modalities can be nested in a past modality, then such a logic characterises H (and thus WH).

Cherief [5] defined a pomset bisimulation relation over paths and shows that it coincides with H (defined over configurations). The author then predicted that an extension of HML with forward and reverse pomset modalities characterises H. This idea was then developed further by Pinchinat, Laroussinie and Schnoebelen in [32].

Nielsen and Clausen defined a δ-bisimulation relation (δb) over paths [27, 29]. Unlike in [5, 32], one is allowed to reverse independent maximal events in any order. This seemingly small change has a profound effect on the strength of the equivalence: δb coincides with HH. It was shown that an extension of HML with a reverse modality characterises HH when there is no autoconcurrency [27, 29]. Additionally, it was stated (without a proof) [28] that an extension of HML with a reverse *event index* modality characterises HH even in the presence of autoconcurrency. The notion of paths used in [27, 28, 29] induces a notion of configuration. Hence, their logics could be understood as logics over configurations and reverse index modality could be seen as a form of our reverse event identifier modality. We would argue, however, that many properties of configurations related to causality and concurrency between events are expressed more naturally with reverse identifier modalities.

Past or reverse modalities, which are central to our logic, were used before in a number of modal logics and temporal logics [20, 7, 6, 26, 15, 23, 24, 30] but only [26, 15] proposed logical characterisations of true concurrency equivalences. Among the rest, HML with backward modalities in [7, 6] defined over paths is shown to characterise branching bisimulation. Finally, Gutierrez introduced a modal logic for transition systems with independence [17, 18] that has two diamond modalities: one for causally dependent transitions and the other for concurrent transitions with respect to a given transition.

3 Configuration structures and equivalences

In this section we define our computational model (stable configuration structures) and the various bisimulation equivalences for which we shall present logical characterisations.

3.1 Configuration structures

We work with stable configuration structures [13, 14, 12], which are equivalent to stable event structures [36].

Definition 3.1. A *configuration structure* (over an alphabet Act) is a pair $\mathscr{C} = (C, \ell)$ where C is a family of finite sets (configurations) and $\ell : \bigcup_{X \in C} X \to \mathsf{Act}$ is a labelling function.

We use $C_{\mathscr{C}}, \ell_{\mathscr{C}}$ to refer to the two components of a configuration structure $\mathscr{C}$. Also we let $E_{\mathscr{C}} = \bigcup_{X \in C} X$, the *events* of $\mathscr{C}$. We let $e, \ldots$ range over events, and $E, F, \ldots$ over sets of events. We let $a, b, c, \ldots$ range over labels in Act.

Definition 3.2 ([12])**.** A configuration structure $\mathscr{C} = (C, \ell)$ is *stable* if it is

- rooted: $\emptyset \in C$; connected: $\emptyset \neq X \in C$ implies $\exists e \in X : X \setminus \{e\} \in C$;
- closed under bounded unions: if $X, Y, Z \in C$ then $X \cup Y \subseteq Z$ implies $X \cup Y \in C$;
- closed under bounded intersections: if $X, Y, Z \in C$ then $X \cup Y \subseteq Z$ implies $X \cap Y \in C$.

Any stable configuration structure is the set of configurations of a stable event structure [12, Thm 5.3].

Definition 3.3. Let $\mathscr{C} = (C, \ell)$ be a stable configuration structure, and let $X \in C$.

- Causality: $d \leq_X e$ iff for all $Y \in C$ with $Y \subseteq X$ we have $e \in Y$ implies $d \in Y$. Furthermore $d <_X e$ iff $d \leq_X e$ and $d \neq e$.
- Concurrency: $d\ co_X\ e$ iff $d \not<_X e$ and $e \not<_X d$.

It is shown in [12] that $<_X$ is a partial order and that the sub-configurations of X are precisely those subsets Y which are left-closed w.r.t. $<_X$, i.e. if $d <_X e \in Y$ then $d \in Y$. Furthermore, if $X, Y \in C$ with $Y \subseteq X$, then $<_Y = <_X \restriction Y$.

Recall that a prime event structure is a set of events with a labelling function, together with a causality relation and a conflict relation (between events that cannot be members of the same configuration) [36]. The set of configurations of a prime event structure forms a stable configuration structure; prime event structures are a proper subclass of stable event structures. All of our examples are given as prime event structures or the corresponding CCS expressions. When drawing diagrams of prime event structures we shall, as usual, depict the causal relation with arrows, and the conflict relation with dotted lines. We shall also suppress the actual events and write their labels instead. Thus if we have two events e_1 and e_2, both labelled with a, in diagrams we shall denote them as a_1 and a_2, respectively, when we wish to distinguish between them. This is justified, since all the notions of equivalence we shall discuss depend on the labels of the events, rather than the events themselves.

Example 3.4. Consider a prime event structure with events e_1, e_2, e_3 all labelled with a, where e_1 causes e_2 and e_1, e_2 are concurrent with e_3. The corresponding CCS expression is $(a.a) \mid a$. The set of configurations consists of $\emptyset$, $\{e_1\}, \{e_3\}, \{e_1, e_2\}, \{e_1, e_3\}$ and $\{e_1, e_2, e_3\}$.

Definition 3.5. Let $\mathscr{C} = (C, \ell)$ be a stable configuration structure and let $a \in \mathsf{Act}$. We let $X \xrightarrow{e}_{\mathscr{C}} X'$ iff $X, X' \in C$, $X \subseteq X'$ and $X' \setminus X = \{e\}$. Furthermore we let $X \xrightarrow{a}_{\mathscr{C}} X'$ iff $X \xrightarrow{e}_{\mathscr{C}} X'$ for some e with $\ell(e) = a$. We also define reverse transitions: $X \overset{e}{\rightsquigarrow}_{\mathscr{C}} X'$ iff $X' \xrightarrow{e}_{\mathscr{C}} X$, and $X \overset{a}{\rightsquigarrow}_{\mathscr{C}} X'$ iff $X' \xrightarrow{a}_{\mathscr{C}} X$. The overloading of notation whereby transitions can be labelled with events or with event labels should not cause confusion.

For a set of events E, let $\ell(E)$ be the multiset of labels of events in E. We define a *step* transition relation where concurrent events are executed in a single step:

Definition 3.6. Let $\mathscr{C} = (C, \ell)$ be a stable configuration structure and let $A \in \mathbb{N}^{\mathsf{Act}}$ (A is a multiset over Act). We let $X \xrightarrow{A}_{\mathscr{C}} X'$ iff $X, X' \in C$, $X \subseteq X'$, and $X' \setminus X = E$ with $d\ co_{X'}\ e$ for all $d, e \in E$ and $\ell(E) = A$.

We shall assume in what follows that stable configuration structures are *image finite* with respect to forward transitions, i.e. for any configuration X and any label a, the set $\{X' : X \xrightarrow{a}_{\mathscr{C}} X'\}$ is finite.

3.2 Equivalences

We define history-preserving bisimulations and illustrate the differences between them with examples.

Definition 3.7. Let $\mathscr{X} = (X, <_X, \ell_X)$ and $\mathscr{Y} = (Y, <_Y, \ell_Y)$ be partial orders which are labelled over Act. We say that $\mathscr{X}$ and $\mathscr{Y}$ are *isomorphic* ($X \cong Y$) iff there is a bijection from X to Y respecting the ordering and the labelling. The isomorphism class $[\mathscr{X}]_{\cong}$ of a partial order labelled over Act is called a *pomset* over Act.

Definition 3.8 ([8, 12])**.** Let $\mathscr{C}, \mathscr{D}$ be stable configuration structures. A relation $\mathscr{R} \subseteq C_{\mathscr{C}} \times C_{\mathscr{D}}$ is a *weak history-preserving (WH) bisimulation* between $\mathscr{C}$ and $\mathscr{D}$ if $\mathscr{R}(\emptyset, \emptyset)$ and if $\mathscr{R}(X, Y)$ and $a \in \mathsf{Act}$ then:

- $(X, <_X, \ell_{\mathscr{C}} \restriction X) \cong (Y, <_Y, \ell_{\mathscr{D}} \restriction Y)$;
- if $X \xrightarrow{a}_{\mathscr{C}} X'$ then $\exists Y'.\ Y \xrightarrow{a}_{\mathscr{D}} Y'$ and $\mathscr{R}(X', Y')$;
- if $Y \xrightarrow{a}_{\mathscr{D}} Y'$ then $\exists X'.\ X \xrightarrow{a}_{\mathscr{C}} X'$ and $\mathscr{R}(X', Y')$.

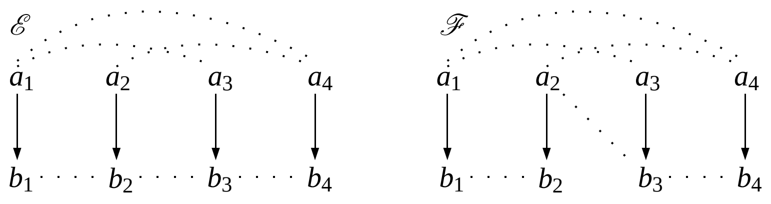

Figure 2: Example 3.12.

We say that $\mathscr{C}$ and $\mathscr{D}$ are WH equivalent ($\mathscr{C} \approx_{\mathsf{wh}} \mathscr{D}$) iff there is a WH bisimulation between $\mathscr{C}$ and $\mathscr{D}$.

Definition 3.9 ([34, 12])**.** Let $\mathscr{C},\mathscr{D}$ be stable configuration structures. A relation $\mathscr{R} \subseteq C_{\mathscr{C}} \times C_{\mathscr{D}} \times \mathscr{P}(E_{\mathscr{C}} \times E_{\mathscr{D}})$ is a *history-preserving (H) bisimulation* between $\mathscr{C}$ and $\mathscr{D}$ iff $\mathscr{R}(\emptyset,\emptyset,\emptyset)$ and if $\mathscr{R}(X,Y,f)$ and $a \in \mathsf{Act}$

- f is an isomorphism between $(X,<_X,\ell_{\mathscr{C}} \restriction X)$ and $(Y,<_Y,\ell_{\mathscr{D}} \restriction Y)$;
- if $X \xrightarrow{a}_{\mathscr{C}} X'$ then $\exists Y', f'.\ Y \xrightarrow{a}_{\mathscr{D}} Y'$, $\mathscr{R}(X',Y',f')$ and $f' \restriction X = f$;
- if $Y \xrightarrow{a}_{\mathscr{D}} Y'$ then $\exists X', f'.\ X \xrightarrow{a}_{\mathscr{C}} X'$, $\mathscr{R}(X',Y',f')$ and $f' \restriction X = f$.

We say that $\mathscr{C}$ and $\mathscr{D}$ are H equivalent ($\mathscr{C} \approx_{\mathsf{h}} \mathscr{D}$) iff there is an H bisimulation between $\mathscr{C}$ and $\mathscr{D}$.

Both H and WH have associated hereditary versions:

Definition 3.10 ([3, 21, 12])**.** Let $\mathscr{C},\mathscr{D}$ be stable configuration structures and let $a \in \mathsf{Act}$. Then $\mathscr{R} \subseteq C_{\mathscr{C}} \times C_{\mathscr{D}} \times \mathscr{P}(E_{\mathscr{C}} \times E_{\mathscr{D}})$ is a hereditary H (HH) bisimulation iff $\mathscr{R}$ is an H bisimulation and if $\mathscr{R}(X,Y,f)$ then for any $a \in \mathsf{Act}$,

- if $X \overset{a}{\rightsquigarrow}_{\mathscr{C}} X'$ then $\exists Y', f'.\ Y \overset{a}{\rightsquigarrow}_{\mathscr{D}} Y'$, $\mathscr{R}(X',Y',f')$ and $f \restriction X' = f'$;
- if $Y \overset{a}{\rightsquigarrow}_{\mathscr{D}} Y'$ then $\exists X', f'.\ X \overset{a}{\rightsquigarrow}_{\mathscr{C}} X'$, $\mathscr{R}(X',Y',f')$ and $f \restriction X' = f'$.

We say that $\mathscr{C}$ and $\mathscr{D}$ are HH equivalent ($\mathscr{C} \approx_{\mathsf{hh}} \mathscr{D}$) iff there is an HH bisimulation between $\mathscr{C}$ and $\mathscr{D}$.

Definition 3.11. Let $\mathscr{C},\mathscr{D}$ be stable configuration structures and let $a \in \mathsf{Act}$. Then $\mathscr{R} \subseteq C_{\mathscr{C}} \times C_{\mathscr{D}} \times \mathscr{P}(E_{\mathscr{C}} \times E_{\mathscr{D}})$ is a hereditary WH (HWH) bisimulation if $\mathscr{R}(\emptyset,\emptyset,\emptyset)$ and if $\mathscr{R}(X,Y,f)$ and $a \in \mathsf{Act}$ then:

- f is an isomorphism between $(X,<_X,\ell_{\mathscr{C}} \restriction X)$ and $(Y,<_Y,\ell_{\mathscr{D}} \restriction Y)$;
- if $X \xrightarrow{a}_{\mathscr{C}} X'$ then $\exists Y', f'.\ Y \xrightarrow{a}_{\mathscr{D}} Y'$ and $\mathscr{R}(X',Y',f')$;
- if $Y \xrightarrow{a}_{\mathscr{D}} Y'$ then $\exists X', f'.\ X \xrightarrow{a}_{\mathscr{C}} X'$ and $\mathscr{R}(X',Y',f')$;
- if $X \overset{a}{\rightsquigarrow}_{\mathscr{C}} X'$ then $\exists Y', f'.\ Y \overset{a}{\rightsquigarrow}_{\mathscr{D}} Y'$, $\mathscr{R}(X',Y',f')$ and $f \restriction X' = f'$;
- if $Y \overset{a}{\rightsquigarrow}_{\mathscr{D}} Y'$ then $\exists X', f'.\ X \overset{a}{\rightsquigarrow}_{\mathscr{C}} X'$, $\mathscr{R}(X',Y',f')$ and $f \restriction X' = f'$.

Also $\mathscr{C}$ and $\mathscr{D}$ are HWH equivalent ($\mathscr{C} \approx_{\mathsf{hwh}} \mathscr{D}$) iff there is an HWH bisimulation between $\mathscr{C}$ and $\mathscr{D}$.

The inclusions in Figure 1 are immediate from the definitions. They are strict inclusions:

Example 3.12 ([31])**.** Consider event structures $\mathscr{E}$, $\mathscr{F}$ in Figure 2, where each event structure has four a-labelled and four b-labelled events. $\mathscr{E} = \mathscr{F}$ holds for $\approx_{\mathsf{hwh}}$, and hence for $\approx_{\mathsf{wh}}$, but not for $\approx_{\mathsf{h}}$, and hence not for $\approx_{\mathsf{hh}}$. We now show this. $\mathscr{E}$, $\mathscr{F}$ have the same configurations except that $\{a_2,a_3,b_3\}$ is missing in $\mathscr{F}$. We define a bisimulation by relating all isomorphic states, and check that it is an HWH. To see that $\mathscr{E}$ and $\mathscr{F}$ are not H-equivalent, consider $\emptyset \xrightarrow{a_2}\xrightarrow{a_3} \{a_2,a_3\}$ in $\mathscr{F}$. This must be matched by moving to configuration $\{a_i,a_{i+1}\}$ in $\mathscr{E}$, where $i \in \{1,2,3\}$. But then both b_i and b_{i+1} are possible. However $\{a_2,a_3\}$ in $\mathscr{F}$ can only do b_2. Hence one of the b_i and b_{i+1} in $\mathscr{E}$ cannot be matched to b_2 in such way that the resulting isomorphism contains the already established pairs (either $(a_2,a_i),(a_3,a_{i+1})$ or $(a_2,a_{i+1}),(a_3,a_i)$) and is history-preserving.

Example 3.13. The Absorption Law [4, 3, 12]

$$(a\,|\,(b+c)) + (a\,|\,b) + ((a+c)\,|\,b) = (a\,|\,(b+c)) + ((a+c)\,|\,b)$$

holds for $\approx_{\mathsf{h}}$, and thus for $\approx_{\mathsf{wh}}$, but not for $\approx_{\mathsf{hwh}}$.

4 Event Identifier Logic

We now introduce our logic, which we call Event Identifier Logic (EIL). We assume an infinite set of identifiers Id, ranged over by $x, y, z, \ldots$. The syntax of EIL is as follows:

$$\phi ::= \mathtt{tt} \mid \neg\phi \mid \phi \wedge \phi' \mid \langle x : a \rangle\rangle \phi \mid (x : a)\phi \mid \langle\langle x \rangle \phi$$

We include the usual operators of propositional logic: truth $\mathtt{tt}$, negation $\neg\phi$ and conjunction $\phi \wedge \phi'$. We then have *forward diamond* $\langle x : a \rangle\rangle \phi$, which says that it is possible to perform an event labelled with a and reach a new configuration where ϕ holds. In the formula $\langle x : a \rangle\rangle \phi$, the modality $\langle x : a \rangle\rangle$ binds all free occurrences of x in ϕ. Next we have *declaration* $(x : a)\phi$. This says that there is some event with label a in the current configuration which can be bound to x, in such a way that ϕ holds. Here the declaration $(x : a)$ binds all free occurrences of x in ϕ. Finally we have *reverse diamond* $\langle\langle x \rangle \phi$. This says that it is possible to perform the reverse event bound to identifier x, and reach a configuration where ϕ holds. Note that $\langle\langle x \rangle$ does not bind x. Clearly any occurrences of x that get bound by $(x : a)$ must be of the form $\langle\langle x \rangle$. We allow alpha-conversion of bound names. We use $\phi, \psi, \ldots$ to range over formulas of EIL.

Example 4.1. The formula $\langle x : a \rangle\rangle \langle y : a \rangle\rangle \langle\langle x \rangle \mathtt{tt}$ says that there are events with label a, say e_1 and e_2, that can be bound to x and y such that, after performing e_1 and then e_2, we can reverse e_1. Obviously, after performing e_1 followed by e_2, we can always reverse e_2. This formula could be interpreted as saying that an event bound to x is *concurrent* with an event bound to y. Next, consider $\langle x : a \rangle\rangle \langle y : a \rangle\rangle \neg \langle\langle x \rangle \mathtt{tt}$. The formula expresses that an event bound to x *causes* an event bound to y (because if we could reverse x before y, we would reach a configuration containing y and not x, which contradicts x being a cause of y).

Definition 4.2. We define $\mathsf{fi}(\phi)$, the set of free identifiers of ϕ, by induction on formulas:.

$$\begin{array}{lll} \mathsf{fi}(\mathtt{tt}) = \emptyset & \mathsf{fi}(\phi_1 \wedge \phi_2) = \mathsf{fi}(\phi_1) \cup \mathsf{fi}(\phi_2) & \mathsf{fi}((x : a)\phi) = \mathsf{fi}(\phi) \setminus \{x\} \\ \mathsf{fi}(\neg\phi) = \mathsf{fi}(\phi) & \mathsf{fi}(\langle x : a \rangle\rangle \phi) = \mathsf{fi}(\phi) \setminus \{x\} & \mathsf{fi}(\langle\langle x \rangle \phi) = \mathsf{fi}(\phi) \cup \{x\} \end{array}$$

We say that ϕ is *closed* if $\mathsf{fi}(\phi) = \emptyset$; otherwise ϕ is *open*.

In order to assign meaning to open formulas, as usual we employ environments which tell us what events the free identifiers are bound to.

Definition 4.3. An *environment* ρ is a partial mapping from Id to events. We say that ρ *is a permissible environment for* ϕ *and* X if $\mathsf{fi}(\phi) \subseteq \mathsf{dom}(\rho)$ and $\mathsf{rge}(\rho \upharpoonright \mathsf{fi}(\phi)) \subseteq X$.

We let $\emptyset$ denote the empty environment. We let $\rho[x \mapsto e]$ denote the environment ρ' which agrees with ρ except possibly on x, where $\rho'(x) = e$ (and $\rho(x)$ may or may not be defined). We abbreviate $\emptyset[x \mapsto e]$ by $[x \mapsto e]$. We let $\rho \setminus x$ denote ρ with the assignment to x deleted (if defined in ρ).

Now we can formally define the semantics of EIL:

Definition 4.4. Let $\mathscr{C}$ be a stable configuration structure. We define a satisfaction relation $\mathscr{C}, X, \rho \models \phi$ where X is a configuration of $\mathscr{C}$, and ρ is a permissible environment for ϕ and X, by induction on formulas as follows (we suppress the $\mathscr{C}$ where it is clear from the context):

- $X,\rho \models \mathrm{tt}$ always
- $X,\rho \models \neg\phi$ iff $X,\rho \not\models \phi$
- $X,\rho \models \phi_1 \wedge \phi_2$ iff $X,\rho \models \phi_1$ and $X,\rho \models \phi_2$
- $X,\rho \models \langle x:a\rangle\rangle\phi$ iff $\exists X', e$ such that $X \xrightarrow{e}_{\mathscr{C}} X'$ with $\ell(e)=a$ and $X',\rho[x\mapsto e] \models \phi$
- $X,\rho \models (x:a)\phi$ iff $\exists e \in X$ such that $\ell(e)=a$ and $X,\rho[x\mapsto e] \models \phi$
- $X,\rho \models \langle\langle x\rangle\phi$ iff $\exists X', e$ such that $X \overset{e}{\rightsquigarrow}_{\mathscr{C}} X'$ with $\rho(x)=e$ and $X',\rho \models \phi$ (and ρ is a permissible environment for ϕ and X')

For closed ϕ we further define $\mathscr{C},X \models \phi$ iff $\mathscr{C},X,\emptyset \models \phi$, and $\mathscr{C} \models \phi$ iff $\mathscr{C},\emptyset \models \phi$.

In the case of $\langle\langle x\rangle\phi$, note that even though according to the syntax x is allowed to occur free in ϕ, if x does occur free in ϕ then $X,\rho \models \langle\langle x\rangle\phi$ can never hold: if $\rho(x)=e$ and $X \overset{e}{\rightsquigarrow}_{\mathscr{C}} X'$ then $X',\rho \models \phi$ cannot hold, since ρ is not a permissible environment for ϕ and X', as ρ assigns a free identifier of ϕ to an event outside X'.

Example 4.5. Consider the configuration structure from Example 3.4. The empty configuration satisfies $\langle x:a\rangle\rangle\langle y:a\rangle\rangle\langle\langle x\rangle\mathrm{tt}$: we have $\emptyset,\emptyset \models \langle x:a\rangle\rangle\langle y:a\rangle\rangle\langle\langle x\rangle\mathrm{tt}$ since $\{e_1,e_3\},[x\mapsto e_1, y\mapsto e_3] \models \langle\langle x\rangle\mathrm{tt}$; the latter holds because $\{e_1,e_3\} \overset{e_1}{\rightsquigarrow} \{e_3\}$ and $\rho(x)=e_1$. Also, $\emptyset,\emptyset \models \langle x:a\rangle\rangle\langle y:a\rangle\rangle\neg\langle\langle x\rangle\mathrm{tt}$. We have $\emptyset,\emptyset \models \langle x:a\rangle\rangle\langle y:a\rangle\rangle\neg\langle\langle x\rangle\mathrm{tt}$ since $\{e_1,e_2\},[x\mapsto e_1, y\mapsto e_2] \models \neg\langle\langle x\rangle\mathrm{tt}$. This is because $\{e_1,e_2\} \overset{e_1}{\not\rightsquigarrow} \{e_2\}$ as $\{e_2\}$ is not a configuration.

The closed formula $(x:a)\mathrm{tt}$ says that there is some event labelled with a in the current configuration: $X \models (x:a)\mathrm{tt}$ iff $\exists e \in X.\ \ell(e)=a$. Returning to Example 3.4, note that as well as $\{e_1,e_2\},[x\mapsto e_1, y\mapsto e_2] \models \neg\langle\langle x\rangle\mathrm{tt}$ this also holds: $\{e_1,e_2\},[x\mapsto e_1, y\mapsto e_2] \models (x:a)\langle\langle x\rangle\mathrm{tt}$. By the definition of $(x:a)$, the current environment is updated to $[x\mapsto e_2, y\mapsto e_2]$ and we obtain $\{e_1,e_2\},[x\mapsto e_2, y\mapsto e_2] \models \langle\langle x\rangle\mathrm{tt}$. Correspondingly, $\{e_1,e_2\},[x\mapsto e_1, y\mapsto e_2] \models (x:a)\langle\langle x\rangle(y:a)\langle\langle y\rangle\mathrm{tt}$. However, $\{e_1,e_2\},[x\mapsto e_1, y\mapsto e_2] \not\models (x:a)\langle\langle x\rangle\langle\langle y\rangle\mathrm{tt}$ since $\{e_1\},[x\mapsto e_2, y\mapsto e_2] \not\models \langle\langle y\rangle\mathrm{tt}$.

We introduce further operators as derived operators of EIL:

Notation 4.6 (Derived operators). Let $A=\{a_1,\dots,a_n\}$ be a multiset of labels.

- $\mathrm{ff} \overset{\mathrm{df}}{=} \neg\mathrm{tt}$, $\quad [x:a]]\,\phi \overset{\mathrm{df}}{=} \neg\langle x:a\rangle\rangle\neg\phi$, $\quad \phi_1 \vee \phi_2 \overset{\mathrm{df}}{=} \neg(\neg\phi_1 \wedge \neg\phi_2)$
- Forward step $\langle A\rangle\rangle\phi \overset{\mathrm{df}}{=} \langle x_1:a_1\rangle\rangle\cdots\langle x_n:a_n\rangle\rangle(\phi \wedge \bigwedge_{i=1}^{n-1}\langle\langle x_i\rangle\mathrm{tt})$ where $x_1,\dots,x_n$ are fresh and distinct (and in particular are not free in ϕ). We write $\langle a_1,\dots,a_n\rangle\rangle\phi$ instead of $\langle\{a_1,\dots,a_n\}\rangle\rangle\phi$. In the case $n=1$ we have $\langle a\rangle\rangle\phi \overset{\mathrm{df}}{=} \langle x:a\rangle\rangle\phi$ where x is fresh.
- Reverse step $\langle\langle A\rangle\phi \overset{\mathrm{df}}{=} (x_1:a_1)\cdots(x_n:a_n)(\langle\langle x_1\rangle\cdots\langle\langle x_n\rangle\phi \wedge \bigwedge_{i=2}^{n}\langle\langle x_i\rangle\mathrm{tt})$ where $x_1,\dots,x_n$ are fresh and distinct (and in particular are not free in ϕ). We write $\langle\langle a_1,\dots,a_n\rangle\phi$ instead of $\langle\langle\{a_1,\dots,a_n\}\rangle\phi$. In the case $n=1$ we have $\langle\langle a\rangle\phi \overset{\mathrm{df}}{=} (x:a)\langle\langle x\rangle\phi$ where x is fresh.

Example 4.7. Consider $\mathscr{E}$, $\mathscr{F}$ in Figure 2 and $\phi \equiv [x:a]]\,[y:a]]\,(\langle z:b\rangle\rangle\neg\langle\langle x\rangle\mathrm{tt} \wedge \langle w:b\rangle\rangle\neg\langle\langle y\rangle\mathrm{tt})$. We easily check that $\mathscr{E}$ satisfies ϕ and $\mathscr{F}$ does not. Next, consider $\psi \equiv \langle x:a\rangle\rangle([w:c]]\,\mathrm{ff} \wedge \langle y:b\rangle\rangle\langle\langle x\rangle\ [z:c]]\,\mathrm{ff})$. Then the LHS structure of the Absorption Law in Example 3.13 satisfies ψ and the RHS does not. Strictly speaking, event identifiers are not necessary to distinguish the two pairs of configuration structures. A formula with simple label modalities $\langle a\rangle\rangle([c]]\,\mathrm{ff} \wedge \langle b\rangle\rangle\langle\langle a\rangle\ [c]]\,\mathrm{ff})$ is sufficient for the the Absorption Law, and $\mathscr{E}$, $\mathscr{F}$ in Figure 2 can be distinguished by a logic with pomset modalities (both reverse and forward) defined over runs [5, 32].

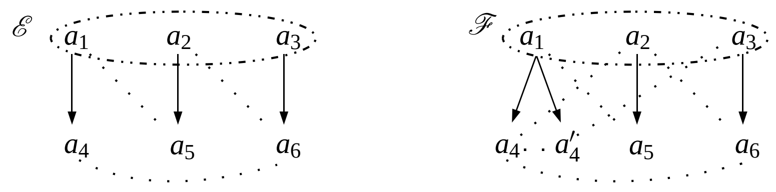

Figure 3: Example 4.8.

Example 4.8. Consider $\mathscr{E}$, $\mathscr{F}$ in Figure 3. There is a non-binary conflict among the three initial a-events (indicated by a dashed ellipsis) defined by requiring that at most two of these events can appear in any configuration. $\mathscr{E}$ and $\mathscr{F}$ are H equivalent: we define a bisimulation by relating configurations of identically labelled events (including where a_4 is matched with a'_4) and check that it is an H. The structures are also HWH equivalent. This time we define a bisimulation between order isomorphic configurations (of which there only five isomorphism classes: $\emptyset$, $\{a\}$, $\{a,a\}$, $\{a < a\}$ and $\{a < a, a\}$, where events separated by commas are concurrent) and check that it is an HWH. However, $\mathscr{E}$ and $\mathscr{F}$ are not HH equivalent and event identifiers are indeed necessary to distinguish them. The formula $\langle x : a\rangle\rangle\langle y : a\rangle\rangle(\neg\langle\langle x\rangle \mathtt{tt} \wedge \langle z : a\rangle\rangle\langle\langle y\rangle\langle w : a\rangle\rangle\neg\langle\langle z\rangle \mathtt{tt} \wedge \langle z' : a\rangle\rangle\langle\langle y\rangle\neg\langle w' : a\rangle\rangle\neg\langle\langle z'\rangle \mathtt{tt})$ is only satisfied by $\mathscr{E}$. It requires that x causes y and that z and z' are bound to different events because $\langle z : a\rangle\rangle$ and $\langle z' : a\rangle\rangle$ are followed by mutually contradictory behaviours. This is possible in $\mathscr{E}$ (a_1, a_4 can be followed by either a_3 or a_2) but not in $\mathscr{F}$: none of the pairs of causally dependent events offers two different a-events.

5 Using EIL to characterise equivalences

We wish to show that EIL and its various sublogics characterise the equivalences defined in Section 3.2. Each sublogic of EIL induces an equivalence on configuration structures in a standard fashion:

Definition 5.1. Let L be any sublogic of EIL. Then L induces an equivalence on stable configuration structures as follows: $\mathscr{C} \sim_L \mathscr{D}$ iff for all closed $\phi \in L$ we have $\mathscr{C} \models \phi$ iff $\mathscr{D} \models \phi$.

First we introduce a simple sublogic that allows us to characterise order isomorphism.

5.1 Reverse-only logic and order isomorphism

We define sublogics of EIL, consisting of formulas where only reverse transitions are allowed.

Definition 5.2. *Reverse-only* logic $\mathrm{EIL_{ro}}$:

$$\phi ::= \mathtt{tt} \mid \neg\phi \mid \phi \wedge \phi' \mid (x : a)\phi \mid \langle\langle x\rangle\phi$$

We further define *declaration-free* reverse-only logic $\mathrm{EIL_{dfro}}$:

$$\phi ::= \mathtt{tt} \mid \neg\phi \mid \phi \wedge \phi' \mid \langle\langle x\rangle\phi$$

These logics are preserved between isomorphic configurations, and characterise configurations up to isomorphism.

Lemma 5.3. *Let $\mathscr{C}, \mathscr{D}$ be stable configuration structures, and let X, Y be configurations of $\mathscr{C}, \mathscr{D}$ respectively. Suppose that $f : X \cong Y$. Then for any $\phi \in \mathrm{EIL_{ro}}$, and any ρ (permissible environment for ϕ and X), we have $X, \rho \models \phi$ iff $Y, f \circ \rho_\phi \models \phi$.*

Recall that ρ_ϕ is an abbreviation for $\rho \restriction \mathsf{fi}(\phi)$. Function composition is in applicative rather than diagrammatic order.

Given any configuration X we can create a closed formula $\theta_X \in \mathrm{EIL}_{\mathrm{ro}}$ which gives the order structure of X. We make this precise in the following lemma:

Lemma 5.4. *Let X be a configuration of a stable configuration structure $\mathscr{C}$. There is a* closed *formula $\theta_X \in \mathrm{EIL}_{\mathrm{ro}}$, such that if Y is any configuration of a stable configuration structure $\mathscr{D}$ and $|Y| = |X|$, then $Y \cong X$ iff $Y \models \theta_X$.*

The next lemma follows fairly immediately from the proof of Lemma 5.4 and from Lemma 5.3:

Lemma 5.5. *Let X be a configuration of a stable configuration structure $\mathscr{C}$. Let $\{z_e : e \in X\}$ be distinct identifiers. Let the environment ρ_X be defined by $\rho_X(z_e) = e$ $(e \in X)$. There is a formula $\theta'_X \in \mathrm{EIL}_{\mathrm{dfro}}$ with $\mathsf{fi}(\theta') = \{z_e : e \in X\}$, such that $X, \rho_X \models \theta'_X$ and if Y is any configuration of a stable configuration structure $\mathscr{D}$ and $|Y| = |X|$, then $Y \cong X$ iff $\exists \rho . Y, \rho \models \theta'_X$.*

5.2 Logics for history-preserving bisimulations

We start by showing that EIL characterises HH-bisimulation. We then present sublogics of EIL which correspond to H-bisimulation, WH-bisimulation and HWH-bisimulation.

Our first result is related to the result of [28] that a logic with reverse event index modality (discussed above in Section 2) characterises HH.

Theorem 5.6. *Let $\mathscr{C}, \mathscr{D}$ be stable configuration structures. Then, $\mathscr{C} \approx_{\mathsf{hh}} \mathscr{D}$ if and only if $\mathscr{C} \sim_{\mathrm{EIL}} \mathscr{D}$.*

Remark 5.7. In fact Theorem 5.6 would hold with the logic restricted by not using declarations $(x : a)\phi$. However we include declarations in EIL because they are useful in defining sublogics for WH, among other things.

We define a sublogic of EIL which characterises history-preserving bisimulation:

Definition 5.8. $\mathrm{EIL}_{\mathrm{h}}$ is given as follows, where ϕ_r is a formula of $\mathrm{EIL}_{\mathrm{ro}}$:

$$\phi ::= \mathsf{tt} \mid \neg\phi \mid \phi \wedge \phi' \mid \langle x : a\rangle\rangle\phi \mid (x : a)\phi \mid \phi_r$$

$\mathrm{EIL}_{\mathrm{h}}$ is just EIL with $\langle\langle x : a\rangle\phi$ replaced by $\phi_r \in \mathrm{EIL}_{\mathrm{ro}}$. Thus one is not allowed to go forward after going in reverse. This concept of disallowing forward moves embedded inside reverse moves appears in [15].

Theorem 5.9. *Let $\mathscr{C}, \mathscr{D}$ be stable configuration structures. Then, $\mathscr{C} \approx_{\mathsf{h}} \mathscr{D}$ if and only if $\mathscr{C} \sim_{\mathrm{EIL}_{\mathrm{h}}} \mathscr{D}$.*

Remark 5.10. Just as for Theorem 5.6, Theorem 5.9 would still hold if we disallow declarations $(x : a)\phi$. This gives the following more minimal logic, where $\phi_r \in \mathrm{EIL}_{\mathrm{dfro}}$.

$$\phi ::= \mathsf{tt} \mid \neg\phi \mid \phi \wedge \phi' \mid \langle x : a\rangle\rangle\phi \mid \phi_r$$

We define a sublogic $\mathrm{EIL}_{\mathrm{wh}}$ of $\mathrm{EIL}_{\mathrm{h}}$ which characterises weak history-preserving bisimulation. We get from $\mathrm{EIL}_{\mathrm{h}}$ to $\mathrm{EIL}_{\mathrm{wh}}$ by simply requiring that all formulas of $\mathrm{EIL}_{\mathrm{wh}}$ are *closed*.

Definition 5.11. $\mathrm{EIL}_{\mathrm{wh}}$ is given as follows, where ϕ_{rc} is a *closed* formula of $\mathrm{EIL}_{\mathrm{ro}}$ (Definition 5.2):

$$\phi ::= \mathsf{tt} \mid \neg\phi \mid \phi \wedge \phi' \mid \langle a\rangle\rangle\phi \mid \phi_{rc}$$

In the above definition we write $\langle a\rangle\rangle\phi$ rather than $\langle x : a\rangle\rangle\phi$ since ϕ is closed and in particular x does not occur free in ϕ (Notation 4.6). Also we omit declarations $(x : a)\phi$ since they have no effect when ϕ is closed. Of course declarations can occur in ϕ_{rc}.

Theorem 5.12. *Let $\mathscr{C},\mathscr{D}$ be stable configuration structures. Then, $\mathscr{C} \approx_{\mathsf{wh}} \mathscr{D}$ iff $\mathscr{C} \sim_{\mathrm{EIL}_{\mathrm{wh}}} \mathscr{D}$.*

We believe that $\mathrm{EIL}_{\mathrm{wh}}$ is the first logic proposed for weak history-preserving bisimulation with autoconcurrency allowed. Goltz et al. [15] described a logic for weak history-preserving bisimulation with no autoconcurrency allowed, but in this case, weak history-preserving bisimulation is as strong as history-preserving bisimulation [12].

Just as we weakened $\mathrm{EIL}_{\mathrm{h}}$ to get $\mathrm{EIL}_{\mathrm{wh}}$ we can weaken EIL by requiring that forward transitions $\langle x : a\rangle\rangle\phi$ are only allowed if ϕ is closed. Again instead of $\langle x : a\rangle\rangle\phi$ we write $\langle a\rangle\rangle\phi$. This gives us $\mathrm{EIL}_{\mathrm{hwh}}$:

Definition 5.13. $\mathrm{EIL}_{\mathrm{hwh}}$ is given below, where ϕ_c ranges over closed formulas of $\mathrm{EIL}_{\mathrm{hwh}}$.

$$\phi ::= \mathrm{tt} \mid \neg\phi \mid \phi \wedge \phi' \mid \langle a\rangle\rangle\phi_c \mid (x : a)\phi \mid \langle\langle x\rangle\phi$$

Plainly $\mathrm{EIL}_{\mathrm{wh}}$ is a sublogic of $\mathrm{EIL}_{\mathrm{hwh}}$ as well as of $\mathrm{EIL}_{\mathrm{h}}$.

Theorem 5.14. *Let $\mathscr{C},\mathscr{D}$ be stable configuration structures. Then, $\mathscr{C} \approx_{\mathsf{hwh}} \mathscr{D}$ iff $\mathscr{C} \sim_{\mathrm{EIL}_{\mathrm{hwh}}} \mathscr{D}$.*

With no (equidepth) autoconcurrency, we know that $\approx_{\mathsf{hwh}}$ is as strong as $\approx_{\mathsf{hh}}$ [3, 31]. So $\mathrm{EIL}_{\mathrm{hwh}}$ is as strong as EIL in this case.

6 Characteristic formulas

In this section we investigate characteristic formulas for three of the equivalences we have considered, namely HH, H and WH. The idea is that we reduce checking whether $\mathscr{C}$ and $\mathscr{D}$ satisfy the same formulas in a logic such as EIL to the question of whether $\mathscr{D}$ satisfies a particular formula $\chi_{\mathscr{C}}$, the *characteristic formula* of $\mathscr{C}$, which completely expresses the behaviour of $\mathscr{C}$, at least as far as the particular logic is concerned. As pointed out in [1], this means that checking whether two structures are equivalent is changed from the problem of potentially having to check infinitely many formulas into a single model-checking problem $\mathscr{D} \models \chi_{\mathscr{C}}$.

Characteristic formulas for models of concurrent systems were first investigated in [16], and subsequently in [35] and other papers—see [1] for further references. As far as we are aware, characteristic formulas have not previously been investigated for any true concurrency logic, although we should mention that in [1] characteristic formulas are studied for a logic with both forward and reverse modalities, related to the back and forth simulation of [6].

We shall confine ourselves to *finite* stable configuration structures in this section. Even with this assumption, it is not obvious that an equivalence such as HH, which employs both forward and reverse transitions, can be captured by a single finite-depth formula. To show that forward and reverse transitions need not alternate for ever, we first relate HH to a simple game.

Definition 6.1. Let $\mathscr{C},\mathscr{D}$ be finite stable configuration structures. The game $G(\mathscr{C},\mathscr{D})$ has two players: A (attacker) and D (defender). The set of game states is $S(\mathscr{C},\mathscr{D}) \overset{\mathrm{df}}{=} \{(X,Y,f) : X \in C_{\mathscr{C}}, Y \in C_{\mathscr{D}}, f : X \cong Y\}$. The start state is $(\emptyset,\emptyset,\emptyset)$. At each state of the game A chooses a forward (resp. reverse) move e of either $\mathscr{C}$ or $\mathscr{D}$. Then D must reply with a corresponding forward (resp. reverse) move e' by the other structure. Going forwards we extend f to f' and going in reverse we restrict f to f', as in the definition of HH. The two moves produce a new game state (X',Y',f'). Then D wins if we get to a previously visited state. Conversely, A wins if D cannot find a move. (Also D wins if A cannot find a move, but that can only happen if both $\mathscr{C}$ and $\mathscr{D}$ have only the empty configuration.)

It is reasonable that D wins if a state is repeated, since if A then chooses a different and better move at the repeated state, A could have chosen that on the previous occasion.

Definition 6.2. Given finite stable configuration structures $\mathscr{C},\mathscr{D}$, let $s(\mathscr{C},\mathscr{D}) \overset{\mathrm{df}}{=} |S(\mathscr{C},\mathscr{D})|$, let $c(\mathscr{C}) = \max\{|X| : X \in C_{\mathscr{C}}\}$, and let $c(\mathscr{C},\mathscr{D}) = \min\{c(\mathscr{C}),c(\mathscr{D})\}$.

Clearly any play of the game $G(\mathscr{C},\mathscr{D})$ finishes after no more than $s(\mathscr{C},\mathscr{D})$ moves. We can place an upper bound on $s(\mathscr{C},\mathscr{D})$ as follows:

Proposition 6.3. *Let $\mathscr{C},\mathscr{D}$ be finite stable configuration structures. Then $s(\mathscr{C},\mathscr{D}) \leq |C_{\mathscr{C}}|.|C_{\mathscr{D}}|.c(\mathscr{C},\mathscr{D})!$.*

Note that if there is no autoconcurrency, any isomorphism $f : X \cong Y$ is unique, and so we can improve the upper bound on the number of states to $s(\mathscr{C},\mathscr{D}) \leq |C_{\mathscr{C}}|.|C_{\mathscr{D}}|$.

Proposition 6.4. *Let $\mathscr{C},\mathscr{D}$ be finite stable configuration structures. Then $\mathscr{C} \approx_{\mathsf{hh}} \mathscr{D}$ iff defender D has a winning strategy for the game $G(\mathscr{C},\mathscr{D})$.*

Remark 6.5. Certainly game characterisations of HH equivalence have been used many times before; see e.g. [9, 10, 11, 22, 17]. However defender is usually said to win if the play continues for ever, whereas we say that defender wins if a state is repeated. This is because we are working with finite configuration structures, rather than, say, Petri nets.

Definition 6.6. Let $\phi \in \mathrm{EIL}$. The *modal depth* $\mathsf{md}(\phi)$ of ϕ is defined as follows:

$$\begin{array}{lll} \mathsf{md}(\mathtt{tt}) \overset{\mathrm{df}}{=} 0 & \mathsf{md}(\phi \wedge \phi') \overset{\mathrm{df}}{=} \max(\mathsf{md}(\phi),\mathsf{md}(\phi')) & \mathsf{md}((x:a)\phi) \overset{\mathrm{df}}{=} \mathsf{md}(\phi) \\ \mathsf{md}(\neg\phi) \overset{\mathrm{df}}{=} \mathsf{md}(\phi) & \mathsf{md}(\langle x:a\rangle\rangle\phi) \overset{\mathrm{df}}{=} 1+\mathsf{md}(\phi) & \mathsf{md}(\langle\langle x:a\rangle\phi) \overset{\mathrm{df}}{=} 1+\mathsf{md}(\phi) \end{array}$$

We can use the game characterisation of HH to bound the modal depth of EIL formulas needed to check whether finite structures are HH equivalent:

Theorem 6.7. *Let $\mathscr{C},\mathscr{D}$ be finite stable configuration structures. Then $\mathscr{C} \approx_{\mathsf{hh}} \mathscr{D}$ iff $\mathscr{C}$ and $\mathscr{D}$ satisfy the same* EIL *formulas of modal depth no more than $s(\mathscr{C},\mathscr{D})+c(\mathscr{C},\mathscr{D})$.*

We now define a family of characteristic formulas for HH equivalence, parametrised on modal depth.

Definition 6.8. Suppose that Act is finite. Let $\mathscr{C}$ be a finite stable configuration structure. We define formulas $\chi^{\mathrm{hh}}_{X,n}$ (X a configuration of $\mathscr{C}$) by induction on n:

$$\begin{array}{rcl} \chi^{\mathrm{hh}}_{X,0} & \overset{\mathrm{df}}{=} & \theta'_X \\ \chi^{\mathrm{hh}}_{X,n+1} & \overset{\mathrm{df}}{=} & \theta'_X \wedge (\bigwedge\limits_{X \xrightarrow{e}_{\mathscr{C}} X'} \langle z_e : \ell(e)\rangle\rangle \chi^{\mathrm{hh}}_{X',n}) \wedge (\bigwedge\limits_{a \in \mathsf{Act}} [x:a]] \bigvee\limits_{X \xrightarrow{e}_{\mathscr{C}} X', \ell(e)=a} \chi^{\mathrm{hh}}_{X',n}[x/z_e]) \wedge (\bigwedge\limits_{X \overset{e}{\rightsquigarrow}_{\mathscr{C}} X'} \langle\langle z_e\rangle \chi^{\mathrm{hh}}_{X',n}) \end{array}$$

Here $\theta'_X \in \mathrm{EIL}_{\mathsf{dfro}}$ is as in Lemma 5.5 and $\mathsf{fi}(\chi^{\mathrm{hh}}_{X,n}) = \{z_e : e \in X\}$. We further let $\chi^{\mathrm{hh}}_{\mathscr{C},n} \overset{\mathrm{df}}{=} \chi^{\mathrm{hh}}_{\emptyset,n}$.

Note that $\chi^{\mathrm{hh}}_{X,n} \in \mathrm{EIL}$ and $\mathsf{md}(\chi^{\mathrm{hh}}_{X,n}) \leq n + c(\mathscr{C})$.

Theorem 6.9. *Suppose that* Act *is finite. Let $\mathscr{C},\mathscr{D}$ be finite stable configuration structures. Let $s \overset{\mathrm{df}}{=} s(\mathscr{C},\mathscr{D})$. Then $\mathscr{C} \approx_{\mathsf{hh}} \mathscr{D}$ iff $\mathscr{D} \models \chi^{\mathrm{hh}}_{\mathscr{C},s}$.*

Thus we do not have a single characteristic formula for $\mathscr{C}$, but we can deal uniformly with all $\mathscr{D}$ up to a certain size. This is almost as good as having a single characteristic formula for $\mathscr{C}$, since we can generate a formula of the appropriate size once we have settled on $\mathscr{D}$, so that we have still reduced equivalence checking to checking a single formula. Single characteristic formulas are certainly possible for some $\mathscr{C}$s; there remains an open question of whether for all finite $\mathscr{C}$ there is a single formula $\chi^{\mathrm{hh}}_{\mathscr{C}}$ which works for all $\mathscr{D}$.

Matters are simpler for H and WH equivalences, since only forward transitions are employed.

Definition 6.10. Suppose that Act is finite. Let $\mathscr{C}$ be a finite stable configuration structure. We define formulas χ_X^{h} (X a configuration of $\mathscr{C}$) as follows:

$$\chi_X^{\mathrm{h}} \stackrel{\mathrm{df}}{=} \theta'_X \wedge (\bigwedge_{X \xrightarrow{e}_{\mathscr{C}} X'} \langle z_e : \ell(e) \rangle\rangle \chi_{X'}^{\mathrm{h}}) \wedge (\bigwedge_{a \in \mathsf{Act}} [x:a]] \bigvee_{X \xrightarrow{e}_{\mathscr{C}} X', \ell(e)=a} \chi_{X'}^{\mathrm{h}}[x/z_e])$$

Here $\theta'_X \in \mathrm{EIL}_{\mathrm{dfro}}$ is as in Lemma 5.5. We further let $\chi_{\mathscr{C}}^{\mathrm{h}} \stackrel{\mathrm{df}}{=} \chi_{\emptyset}^{\mathrm{h}}$.

Note that $\chi_{\mathscr{C}}^{\mathrm{h}} \in \mathrm{EIL}_{\mathrm{h}}$; it is well-defined, since maximal configurations form the base cases of the recursion. Also $\mathsf{md}(\chi_X^{\mathrm{h}}) \leq 2.c(\mathscr{C})$.

Proposition 6.11. *Suppose that* Act *is finite. Let* $\mathscr{C}, \mathscr{D}$ *be finite stable configuration structures. Then* $\mathscr{D} \approx_{\mathsf{h}} \mathscr{C}$ *iff* $\mathscr{D} \models \chi_{\mathscr{C}}^{\mathrm{h}}$.

WH is even easier as formulas are closed:

Definition 6.12. Suppose that Act is finite. Let $\mathscr{C}$ be a finite stable configuration structure. We define formulas χ_X^{wh} (X a configuration of $\mathscr{C}$) as follows:

$$\chi_X^{\mathrm{wh}} \stackrel{\mathrm{df}}{=} \theta_X \wedge (\bigwedge_{X \xrightarrow{a}_{\mathscr{C}} X'} \langle a \rangle\rangle \chi_{X'}^{\mathrm{wh}}) \wedge (\bigwedge_{a \in \mathsf{Act}} [a]] \bigvee_{X \xrightarrow{a}_{\mathscr{C}} X'} \chi_{X'}^{\mathrm{wh}})$$

Here $\theta_X \in \mathrm{EIL}_{\mathrm{ro}}$ is as in Lemma 5.4. We further let $\chi_{\mathscr{C}}^{\mathrm{wh}} \stackrel{\mathrm{df}}{=} \chi_{\emptyset}^{\mathrm{wh}}$.

Note that $\chi_{\mathscr{C}}^{\mathrm{wh}} \in \mathrm{EIL}_{\mathrm{wh}}$ and $\mathsf{md}(\chi_X^{\mathrm{wh}}) \leq 2.c(\mathscr{C})$.

Proposition 6.13. *Suppose that* Act *is finite. Let* $\mathscr{C}, \mathscr{D}$ *be finite stable configuration structures. Then* $\mathscr{D} \approx_{\mathsf{wh}} \mathscr{C}$ *iff* $\mathscr{D} \models \chi_{\mathscr{C}}^{\mathrm{wh}}$.

7 Conclusions and future work

We have introduced a logic which uses event identifiers to track events in both forwards and reverse directions. As we have seen, this enables it to express causality and concurrency between events. The logic is strong enough to characterise hereditary history-preserving (HH) bisimulation equivalence. We are also able to characterise weaker equivalences using sublogics. In particular we can characterise weak history-preserving bisimulation, which has not been done previously as far as we are aware. We also investigated characteristic formulas for our logic with respect to HH and other equivalences. Again we are not aware of previous work on characteristic formulas for logics for true concurrency.

Baldan and Crafa [2] gave logics for pomset bisimulation and step bisimulation; we have also been able to characterise these equivalences in our setting, but we had to omit this material for reasons of space.

In future work we would like to (1) investigate general laws which hold for the logic, (2) look at sublogics characterising other true concurrency equivalences, including equivalences involving reverse transitions from [3, 31], and (3) answer the open question raised in Section 6 about whether there is a single characteristic formula for a finite structure with respect to HH equivalence.

Acknowledgements. We are grateful to Ian Hodkinson and the anonymous referees for helpful comments and suggestions.

References

[1] L. Aceto, A. Ingólfsdóttir & J. Sack (2009): *Characteristic Formulae for Fixed-Point Semantics: A General Framework*. In: *Proceedings 16th International Workshop on Expressiveness in Concurrency, EXPRESS 2009*, *Electronic Proceedings in Theoretical Computer Science* 8, pp. 1–15, doi:10.4204/EPTCS.8.1.

[2] P. Baldan & S. Crafa (2010): *A Logic for True Concurrency*. In: *Proceedings of 21st International Conference on Concurrency Theory, CONCUR 2010*, *Lecture Notes in Computer Science* 6269, Springer-Verlag, pp. 147–161, doi:10.1007/978-3-642-15375-4_11.

[3] M.A. Bednarczyk (1991): *Hereditary history preserving bisimulations or what is the power of the future perfect in program logics*. Technical Report, Institute of Computer Science, Polish Academy of Sciences, Gdańsk.

[4] G. Boudol & I. Castellani (1987): *On the semantics of concurrency: partial orders and transition systems*. In: *Proceedings of TAPSOFT'87*, *Lecture Notes in Computer Science* 249, Springer-Verlag, pp. 123–137, doi:10.1007/3-540-17660-8_52.

[5] F. Cherief (1992): *Back and forth bisimulations on prime event structures*. In: *Proceedings of PARLE '92*, *Lecture Notes in Computer Science* 605, Springer-Verlag, pp. 843–858, doi:10.1007/3-540-55599-4_128.

[6] R. De Nicola, U. Montanari & F. Vaandrager (1990): *Back and forth bisimulations*. In: *Proceedings of CONCUR '90, Theories of Concurrency: Unification and Extension*, *Lecture Notes in Computer Science* 458, Springer-Verlag, pp. 152–165, doi:10.1007/BFb0039058.

[7] R. De Nicola & F. Vaandrager (1990): *Three Logics for Branching Bisimulation (Extended Abstract)*. In: *Proceedings, Fifth Annual IEEE Symposium on Logic in Computer Science*, IEEE, Computer Society Press, pp. 118–129.

[8] P. Degano, R. De Nicola & U. Montanari (1987): *Observational equivalences for concurrency models*. In M. Wirsing, editor: *Formal Descriptions of Programming Concepts – III, Proceedings of the 3rd IFIP WG 2.2 Conference*, North-Holland, pp. 105–129.

[9] S.B. Fröschle (1999): *Decidability of Plain and Hereditary History-Preserving Bisimilarity for BPP*. In: *Proceedings of Express'99*, *Electronic Notes in Theoretical Computer Science* 27, Elsevier, doi:10.1016/S1571-0661(05)80297-X.

[10] S.B. Fröschle (2005): *Composition and Decomposition in True-Concurrency*. In: *Foundations of Software Science and Computational Structures, 8th International Conference, FOSSACS 2005*, *Lecture Notes in Computer Science* 3441, Springer-Verlag, pp. 333–347, doi:10.1007/978-3-540-31982-5_21.

[11] S.B. Fröschle & S. Lasota (2005): *Decomposition and Complexity of Hereditary History Preserving Bisimulation on BPP*. In: *CONCUR 2005*, *Lecture Notes in Computer Science* 3653, Springer-Verlag, pp. 263–277, doi:10.1007/11539452_22.

[12] R.J. van Glabbeek & U. Goltz (2001): *Refinement of actions and equivalence notions for concurrent systems*. *Acta Informatica* 37(4/5), pp. 229–327, doi:10.1007/s002360000041.

[13] R.J. van Glabbeek & G.D. Plotkin (1995): *Configuration structures*. In: *Proceedings of 10th Annual IEEE Symposium on Logic in Computer Science, LICS 1995*, IEEE Computer Society Press, pp. 199–209, doi:10.1109/LICS.1995.523257.

[14] R.J. van Glabbeek & G.D. Plotkin (2009): *Configuration structures, event structures and Petri nets*. *Theoretical Computer Science* 410(41), pp. 4111–4159, doi:10.1016/j.tcs.2009.06.014.

[15] U. Goltz, R. Kuiper & W. Penczek (1992): *Propositional temporal logics and equivalences*. In: *Proceedings of 3rd International Conference on Concurrency Theory, CONCUR 1992*, *Lecture Notes in Computer Science* 630, Springer-Verlag, pp. 222–236, doi:10.1007/BFb0084794.

[16] S. Graf & J. Sifakis (1986): *A Modal Characterization of Observational Congruence on Finite Terms of CCS*. *Information and Control* 68(1-3), pp. 125–145, doi:10.1016/S0019-9958(86)80031-6.

[17] J. Gutierrez (2009): *Logics and Bisimulation Games for Concurrency, Causality and Conflict*. In: *Proceedings of the 12th International Conference on Foundations of Software Science and Computation Structures, FOSSACS 09*, *Lecture Notes in Computer Science* 5504, Springer-Verlag, pp. 48–62, doi:`10.1007/978-3-642-00596-1_5`.

[18] J. Gutierrez & J.C. Bradfield (2009): *Model-Checking Games for Fixpoint Logics with Partial Order Models*. In: *Proceedings of the 20th International Conference on Concurrency Theory, CONCUR 2009*, *Lecture Notes in Computer Science* 5710, Springer-Verlag, pp. 354–368, doi:`10.1007/978-3-642-04081-8_24`.

[19] M.C.B. Hennessy & R. Milner (1985): *Algebraic laws for nondeterminism and concurrency*. *Journal of the Association for Computing Machinery* 32(1), pp. 137–161, doi:`10.1145/2455.2460`.

[20] M.C.B. Hennessy & C. Stirling (1985): *The power of the future perfect in program logics*. *Infomation and Control* 67, pp. 23–52, doi:`10.1016/S0019-9958(85)80025-5`.

[21] A. Joyal, M. Nielsen & G. Winskel (1996): *Bisimulation from Open Maps*. *Information and Computation* 127(2), pp. 164–185, doi:`10.1006/inco.1996.0057`.

[22] M. Jurdzinski, M. Nielsen & J. Srba (2003): *Undecidability of domino games and hhp-bisimilarity*. *Information and Computation* 184(2), pp. 343–368, doi:`10.1016/S0890-5401(03)00064-6`.

[23] F. Laroussinie, S. Pinchinat & Ph. Schnoebelen (1995): *Translations between modal logics of reactive systems*. *Theoretical Computer Science* 140(1), pp. 53–71, doi:`10.1016/0304-3975(94)00204-V`.

[24] F. Laroussinie & Ph. Schnoebelen (1995): *A hierarchy of temporal logics with past*. *Theoretical Computer Science* 148, pp. 303–324, doi:`10.1016/0304-3975(95)00035-U`.

[25] M. Mukund & P.S. Thiagarajan (1992): *A logical characterization of well branching event structures*. *Theoretical Computer Science* 96(1), pp. 35–72, doi:`10.1016/0304-3975(92)90181-E`.

[26] R. De Nicola & G.L. Ferrari (1990): *Observational Logics and Concurrency Models*. In: *FSTTCS*, *Lecture Notes in Computer Science* 472, Springer-Verlag, pp. 301–315, doi:`10.1007/3-540-53487-3_53`.

[27] M. Nielsen & C. Clausen (1994): *Bisimulation for Models in Concurrency*. In: *Proceedings of 5th International Conference on Concurrency Theory, CONCUR'94*, *Lecture Notes in Computer Science* 836, Springer-Verlag, pp. 385–400, doi:`10.1007/BFb0015021`.

[28] M. Nielsen & C. Clausen (1994): *Bisimulation, games, and logic*. In: *Results and Trends in Theoretical Computer Science*, *Lecture Notes in Computer Science* 812, Springer-Verlag, pp. 289–306, doi:`10.1007/3-540-58131-6_54`.

[29] M. Nielsen & C. Clausen (1995): *Games and logics for a noninterleaving bisimulation*. *Nordic Journal of Computing* 2(2), pp. 221–249.

[30] W. Penczek (1995): *Branching time and partial order in temporal logics*. In: *Time and Logic: A Computational Approach*, UCL Press Ltd., pp. 179–228.

[31] I.C.C. Phillips & I. Ulidowski (2011): *A Hierarchy of Reverse Bisimulations on Stable Configuration Structures*. *Mathematical Structures in Computer Science* Available at `http://www.doc.ic.ac.uk/~iccp/papers/hierarchymscs.pdf`. To appear.

[32] S. Pinchinat, F. Laroussinie & Ph. Schnoebelen (1994): *Logical characterizations of truly concurrent bisimulation*. Technical Report 114, Grenoble.

[33] L. Pomello (1986): *Some equivalence notions for concurrent systems – An overview*. In: *Advances in Petri Nets 1985*, *Lecture Notes in Computer Science* 222, Springer-Verlag, pp. 381–400, doi:`10.1007/BFb0016222`.

[34] A. Rabinovich & B.A. Trakhtenbrot (1988): *Behavior structures and nets*. *Fundamenta Informaticae* 11(4), pp. 357–403.

[35] B. Steffen & A. Ingólfsdóttir (1994): *Characteristic Formulae for Processes with Divergence*. *Information and Computation* 110(1), pp. 149–163, doi:`10.1006/inco.1994.1028`.

[36] G. Winskel (1987): *Event structures*. In: *Advances in Petri Nets 1986*, *Lecture Notes in Computer Science* 255, Springer-Verlag, pp. 325–392, doi:`10.1007/3-540-17906-2_31`.