A Logic for True Concurrency

PAOLO BALDAN, University of Padova
SILVIA CRAFA, University of Padova

We propose a logic for true concurrency whose formulae predicate about events in computations and their causal dependencies. The induced logical equivalence is hereditary history preserving bisimilarity, and fragments of the logic can be identified which correspond to other true concurrent behavioural equivalences in the literature: step, pomset and history preserving bisimilarity. Standard Hennessy-Milner logic, and thus (interleaving) bisimilarity, is also recovered as a fragment. We also propose an extension of the logic with fixpoint operators, thus allowing to describe causal and concurrency properties of infinite computations. This work contributes to a rational presentation of the true concurrent spectrum and to a deeper understanding of the relations between the involved behavioural equivalences.

Categories and Subject Descriptors: F.1.2 [**Theory of Computation**]: Modes of Computation—*Parallelism and concurrency*; F.4.1 [**Mathematical Logic and Formal Languages**]: Mathematical Logic—*Temporal logic*; F.3.2 [**Logics and Meanings of Programs**]: Semantics of Programming Languages—*Process models*

General Terms: Languages, Theory, Verification

Additional Key Words and Phrases: true concurrency, causality, mu-calculus, behavioural equivalences, history-preserving bisimilarity, event structures

ACM Reference Format:
Paolo Baldan and Silvia Crafa, 2014. A Logic for True Concurrency. *J. ACM* 9, 4, Article 39 (March 2010), 36 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

In the semantics of concurrent and distributed systems, a major dichotomy opposes the interleaving approaches, where concurrency of actions is reduced to the nondeterministic choice among their possible sequentialisations, to true concurrent approaches, where concurrency is taken as a primitive notion. In both cases, on top of the operational models a number of behavioural equivalences have been defined by abstracting from aspects which are considered unobservable [van Glabbeek 2001; van Glabbeek and Goltz 2001].

For the interleaving world, a systematic and impressive picture is taken in the linear-time branching-time spectrum [van Glabbeek 2001]. Quite interestingly, the equivalences in the spectrum can be uniformly characterised in logical terms. Bisimilarity, the finest equivalence, corresponds to Hennessy-Milner (HM) logic: two processes are bisimilar if and only if they satisfy the same HM logic formulae [Hennessy and Milner 1985]. Coarser equivalences correspond to suitable fragments of HM logic, as discussed in [van Glabbeek 2001].

This work is partially supported by the MIUR-PRIN Project CINA.
Author's addresses: P. Baldan and S. Crafa, Diaprtimento di Matematica, Università di Padova, Italy.

DOI: http://dx.doi.org/10.1145/0000000.0000000

In the true concurrent world, relying on models like event structures or transition systems with independence [Winskel and Nielsen 1995], several behavioural equivalences have been defined. Hereditary history preserving (hhp-)bisimilarity [Bednarczyk 1991], the finest equivalence in the spectrum of [van Glabbeek and Goltz 2001], has been shown to arise as a canonical behavioural equivalence when considering partially ordered computations [Joyal et al. 1996] (The abstract notion of bisimilarity threin instantiates to hhp-bisimilarity when taking the category of pomsets as the path category.) Coarser equivalences like history preserving (hp-)bisimilarity [Rabinovich and Trakhtenbrot 1988; Degano et al. 1988; Best et al. 1991], pomset and step bisimilarity have also been widely studied. Correspondingly, a number of logics have been studied, but, to the best of our knowledge, a unifying logical framework for the main true concurrent equivalences is still missing. The huge amount of work on the topic makes it impossible to give a complete account of related approaches. Just to give a few references (see Section 7 for a wider discussion), [De Nicola and Ferrari 1990] proposes a general framework encompassing a number of temporal and modal logics that characterise interleaving bisimilarity as well as pomset bisimilarity and weak hhp-bisimilarity, a weakening of hhp-bisimilarity studied, e.g., in [De Nicola and Ferrari 1990; Pinchinat et al. 1994; Cherief 1992]. However, finer equivalences are not considered and a single unitary logic is missing. Hp-bisimilarity has been studied in the setting of Petri nets and shown to be decidable for finite 1-safe Petri nets in [Vogler 1991]. A decidability result for finite-state Petri nets is obtained also in [Montanari and Pistore 1997] by means of an encoding of into history dependent (HD-)automata. Concerning hhp-bisimilarity, several logics with modalities corresponding to the "retraction" or "backward" execution of computations have been proposed [Hennessy and Stirling 1985; Bednarczyk 1991; Nielsen and Clausen 1995; Phillips and Ulidowski 2011]. When a system does not exhibit autoconcurrency, i.e., where events with the same label are never enabled concurrently, such logics are shown to capture hhp-bisimilarity. Relaxing this restriction requires to move to an event based logic, where specific events executed in the past can be retracted [Bednarczyk 1991; Nielsen and Clausen 1995; Phillips and Ulidowski 2011].

In this paper we propose a behavioural logic for concurrency and we show that it allows us to characterise a relevant part of the true concurrent spectrum. More specifically, the full logic $\mathcal{L}$ is shown to capture hhp-bisimilarity, the finest behavioural equivalence in the spectrum in [van Glabbeek and Goltz 2001]. Then suitable fragments of the logic are shown to scale down to the characterisation of coarser equivalences: history preserving, pomset and step bisimilarity. Standard HM logic, and thus (interleaving) bisimilarity, is also recovered as a fragment.

Our logic allows us to predicate about events in computations together with their causal and independence relations. It is interpreted over prime event structures [Nielsen et al. 1981; Winskel 1987], one of the most widely known event-based models of computation, where the dependencies between events are expressed in terms of causality and (binary) conflict. It could naturally be interpreted over any formalism with explicit notions of event, causality and consistency. A formula is evaluated in a configuration representing the current state of the computation, and it predicates on the possible future evolutions starting from that state. The logic is event-based in the sense that it contains an operator acting as a binder: it asserts the existence of an event satisfying suitable requirements and it binds the event to a variable so that the event can be referred to later in the formula. In this respect, it is reminiscent of the modal analogue of independence-friendly modal logic as considered in [Bradfield and Fröschle 2002].

The logic contains two main operators. The formula $(x, \overline{y} < \mathsf{a}\, z)\varphi$ declares that an a-labelled future event exists, which causally depends on the event bound to x, and is

independent from the event bound to y. Such an event is bound to variable z so that it can be later referred to in φ. In general, x and y can be replaced by tuples of variables. A second operator allows one to "execute" events previously bound to variables. The formula $\langle z \rangle \varphi$ says that the event bound to z is enabled in the current state, and after its execution φ holds.

Different behavioural equivalences are induced by fragments of the logics where we suitably restrict the set of possible futures the formulae are able to refer to. Namely, hhp-bisimilarity, that is captured by the full logic, corresponds to the ability of observing the existence of a number of legal but (possibly) incompatible futures. Such ability is strictly related to the capability of observing future events without executing them (in fact the execution of an event would rule out all the events in conflict with it). Interestingly, the definition of hhp-bisimilarity is normally given in terms of backward transitions, whereas our logical characterisation has a "forward flavour." By restricting to a fragment where future events can be observed only by executing them (any occurrence of the binding operator is immediately followed by a corresponding execution), we get hp-bisimilarity. Pomset bisimilarity is induced by a fragment of the logic obtained by further restricting that for hp-bisimilarity, with the requirement that propositional connectives are used only on closed (sub)formulae. Roughly speaking, this fragment predicates about the possibility of executing pomset transitions and the closedness requirement prevents pomset transitions from being causally linked to the events in the past. Finally, step bisimilarity corresponds to the possibility of observing only currently enabled concurrent events.

The logic $\mathcal{L}$ in its basic form is essentially a means to understand and compare different process equivalences, but its expressive power is rather weak. In fact, although events arbitrarily far in the future can be "observed", the logic only allows us to describe computations where a finite number of events are executed. In order to overcome this limitation and to provide a more powerful specification logic, well-suited for describing properties of unbounded, possibly infinite computations, we enrich the logic with a form of recursion. This is obtained by adding least (and dually greatest) fixpoint operators, thus obtaining a kind of first order modal μ-calculus similar to the μ-calculi in [Dam 1996; Dam et al. 1998] and [Groote and Willemse 2005], which are endowed with first order variables representing channels or data. Similarities exist also with the fixpoint extension of independence-friendly modal logic in [Bradfield and Kreutzer 2005]. In the resulting logic $\mu\mathcal{L}$ one can express non-trivial causal properties, like "any $\mathtt{a}$ action can always be followed by a causally related $\mathtt{b}$ action in at most three steps," or "an $\mathtt{a}$ action can always be executed in parallel with a $\mathtt{b}$ action." Moreover, we show that, as it happens in the interleaving case, the addition of the fixpoint operators does not alter the logical equivalence. The logical equivalence of $\mu\mathcal{L}$ is still hhp-bisimilarity and the same invariance result applies to the fixpoint extensions of the fragments of $\mathcal{L}$ characterising the coarser behavioural equivalences.

This work contributes to the definition of a logical counterpart of the true concurrent spectrum, shading further light on the relations between the involved behavioural equivalences and suggests interesting directions of investigations in the verification of true concurrent properties.

The rest of the paper is organised as follows. In Section 2 we introduce the basics of event structures and the concurrent equivalences we will work with in the paper. In Section 3 we present the syntax and semantics of our logic $\mathcal{L}$. In Section 4 we study the logical equivalence induced by $\mathcal{L}$, proving that it coincides with hhp-bisimilarity. In Section 5 we provide a characterisation of other concurrent equivalences in terms of fragments of our logic. In Section 6 we discuss the fixpoint extension of our logic. Finally, in Section 7 we discuss some related work and present directions of future

research. This is a revised and extended version of the conference paper [Baldan and Crafa 2010].

2. BACKGROUND

In this section we provide the basics of prime event structures which will be used as models for our logic. Then we define some common behavioural true concurrent equivalences which will play a basic role in the paper.

2.1. Event structures

Prime event structures [Nielsen et al. 1981; Winskel 1987] are a widely known model of concurrency. They describe the behaviour of a system in terms of events and dependency relations between such events. Throughout the paper Λ denotes a fixed set of labels ranged over by $\mathsf{a}, \mathsf{b}, \mathsf{c} \ldots$

Definition 2.1 (*prime event structure*). A (Λ-labelled) *prime event structure* (PES) is a tuple $\mathcal{E} = \langle E, \leq, \#, \lambda \rangle$, where E is a denumerable set of *events*, $\lambda : E \to \Lambda$ is a labelling function and $\leq$, $\#$ are binary relations on E, called *causality* and *conflict* respectively, such that:

(1) $\leq$ is a partial order and $\lceil e \rceil = \{e' \in E \mid e' \leq e\}$ is finite for all $e \in E$;
(2) $\#$ is irreflexive, symmetric and hereditary with respect to $\leq$, i.e., for all $e, e', e'' \in E$, if $e \# e' \leq e''$ then $e \# e''$.

In the following, we will assume that the components of an event structure $\mathcal{E}$ are named as in the definition above. Subscripts carry over the components.

Definition 2.2 (*consistency, concurrency*). Let $\mathcal{E}$ be a PES. We say that $e, e' \in E$ are *consistent*, written $e \frown e'$, if $\neg(e \# e')$. A subset $X \subseteq E$ is called *consistent* if $e \frown e'$ for all $e, e' \in X$. We say that e and e' are *concurrent*, written $e \,||\, e'$, if $\neg(e \leq e')$, $\neg(e' \leq e)$ and $\neg(e \# e')$.

Causality, concurrency and consistency will be sometimes used on sets of events. Given $X \subseteq E$ and $e \in E$, by $X < e$ we mean that for all $e' \in X$, $e' < e$. Similarly $X \,||\, e$, resp. $X \frown e$, means that for all $e' \in X$, $e' \,||\, e$, resp. $e' \frown e$. We write $\lceil X \rceil$ for $\bigcup_{e \in X} \lceil e \rceil$.

Configurations of event structures are intended to represent (concurrent) computations, which abstract from the order of execution of concurrent events.

Definition 2.3 (*configuration*). Let $\mathcal{E}$ be a PES. A *(finite) configuration* in $\mathcal{E}$ is a (finite) consistent subset of events $C \subseteq E$ closed w.r.t. causality (i.e., $\lceil C \rceil = C$). The set of finite configurations of $\mathcal{E}$ is denoted by $\mathcal{C}(\mathcal{E})$.

Observe that the empty set of events $\emptyset$ is always a configuration, which can be understood as the initial state of the computation.

Hereafter all configurations will be assumed to be finite. A consistent subset $X \subseteq E$ of events will always be seen as a *pomset* (partially ordered multiset) $(X, \leq_X, \lambda_X)$, where $\leq_X$ and λ_X are the restrictions of $\leq$ and λ to X. Given $X, Y \subseteq E$ we will write $X \sim Y$ if X and Y are isomorphic as pomsets.

Definition 2.4 (*pomset transition and step*). Let $\mathcal{E}$ be a PES and let $C \in \mathcal{C}(\mathcal{E})$. Given $\emptyset \neq X \subseteq E$, if $C \cap X = \emptyset$ and $C' = C \cup X \in \mathcal{C}(\mathcal{E})$ we write $C \xrightarrow{X} C'$ and call it a *pomset transition* from C to C'. When the events in X are pairwise concurrent, we say that $C \xrightarrow{X} C'$ is a *step*. When $X = \{e\}$ we write $C \xrightarrow{e} C'$ instead of $C \xrightarrow{\{e\}} C'$.

A PES $\mathcal{E}$ is called *image finite* if for any $C \in \mathcal{C}(\mathcal{E})$ and $\mathsf{a} \in \Lambda$, the set of events $\{e \in E \mid C \xrightarrow{e} C' \wedge \lambda(e) = \mathsf{a}\}$ is finite. *All* the PESs considered in this paper will be assumed to be image finite. As it commonly happens when relating modal logics and bisimilarities, this assumption is crucial for getting a logical characterisation of the various bisimulation equivalences in Sections 4 and 5, based on a finitary logic.

2.2. Concurrent behavioural equivalences

Behavioural equivalences which capture to some extent the concurrency features of a system, can be defined on the transition system where states are configurations and transitions are pomset transitions.

Definition 2.5 (*pomset, step bisimulation*). Let $\mathcal{E}_1$, $\mathcal{E}_2$ be PESs. A *pomset bisimulation* is a relation $R \subseteq \mathcal{C}(\mathcal{E}_1) \times \mathcal{C}(\mathcal{E}_2)$ such that if $(C_1, C_2) \in R$ and $C_1 \xrightarrow{X_1} C_1'$ then $C_2 \xrightarrow{X_2} C_2'$, with $X_1 \sim X_2$ and $(C_1', C_2') \in R$, and vice versa. We say that $\mathcal{E}_1$, $\mathcal{E}_2$ are *pomset bisimilar*, written $\mathcal{E}_1 \sim_p \mathcal{E}_2$, if there exists a pomset bisimulation R such that $(\emptyset, \emptyset) \in R$.

Step bisimulation is defined analogously, replacing general pomset transitions with steps. We write $\mathcal{E}_1 \sim_s \mathcal{E}_2$ when $\mathcal{E}_1$ and $\mathcal{E}_2$ are step bisimilar.

While pomset and step bisimilarity only consider the causal structure of the current step, (hereditary) history preserving bisimilarities are sensible to the way in which the executed events depend on events in the past. In order to define history preserving bisimilarities the following definition is helpful.

Definition 2.6 (*posetal product*). Given two PESs $\mathcal{E}_1$, $\mathcal{E}_2$, the *posetal product* of their configurations, denoted $\mathcal{C}(\mathcal{E}_1) \bar{\times} \mathcal{C}(\mathcal{E}_2)$, is defined as

$$\{(C_1, f, C_2) \mid C_1 \in \mathcal{C}(\mathcal{E}_1),\ C_2 \in \mathcal{C}(\mathcal{E}_2),\ f : C_1 \to C_2 \text{ isomorphism}\}$$

A subset $R \subseteq \mathcal{C}(\mathcal{E}_1) \bar{\times} \mathcal{C}(\mathcal{E}_2)$ is called a *posetal relation*. We say that R is downward closed when for any $(C_1, f, C_2), (C_1', f', C_2') \in \mathcal{C}(\mathcal{E}_1) \bar{\times} \mathcal{C}(\mathcal{E}_2)$, if $(C_1, f, C_2) \subseteq (C_1', f', C_2')$ pointwise and $(C_1', f', C_2') \in R$ then $(C_1, f, C_2) \in R$.

Given a function $f : X_1 \to X_2$ we will denote by $f[x_1 \mapsto x_2] : X_1 \cup \{x_1\} \to X_2 \cup \{x_2\}$ the function defined, for $z \in X_1 \cup \{x_1\}$, by

$$f[x_1 \mapsto x_2](z) = \begin{cases} x_2 & \text{if } z = x_1 \\ f(z) & \text{otherwise} \end{cases}$$

Definition 2.7 (*(hereditary) history preserving bisimulation*). A *history preserving (hp-)bisimulation* is a posetal relation $R \subseteq \mathcal{C}(\mathcal{E}_1) \bar{\times} \mathcal{C}(\mathcal{E}_2)$ such that if $(C_1, f, C_2) \in R$ and $C \xrightarrow{e_1} C_1'$ then $C_2 \xrightarrow{e_2} C_2'$, with $(C_1', f[e_1 \mapsto e_2], C_2') \in R$, and vice versa. We say that $\mathcal{E}_1$, $\mathcal{E}_2$ are *history preserving (hp-)bisimilar* and write $\mathcal{E}_1 \sim_{hp} \mathcal{E}_2$ if there exists a hp-bisimulation R such that $(\emptyset, \emptyset, \emptyset) \in R$.

A *hereditary history preserving (hhp-)bisimulation* is a downward closed hp-bisimulation. The fact that $\mathcal{E}_1$, $\mathcal{E}_2$ are *hereditary history preserving (hhp-)bisimilar* is denoted $\mathcal{E}_1 \sim_{hhp} \mathcal{E}_2$.

It is easy to show (see, e.g, [van Glabbeek and Goltz 2001]) that the definition of (h)hp-bisimilarity can be equivalently given by using pomset transitions instead of single event transitions, i.e., by asking that if $(C_1, f, C_2) \in R$ and $C \xrightarrow{X_1} C_1'$ then there exists $C_2 \xrightarrow{X_2} C_2'$ and $(C_1', f', C_2') \in R$, with $f'_{|C_1} = f$.

3. A LOGIC FOR TRUE CONCURRENCY

In this section we introduce the syntax and the semantics of our logic. Formulae predicate about events in computations and their dependencies as primitive concepts. The logic is interpreted over PESs. It could be interpreted, without any serious technical complication, over more general classes of event structures, as long as they are endowed with notions of causality and consistency (e.g., over stable event structures [Winskel 1987]). The choice of restricting to PESs is motivated by the fact that they are probably the most popular event structure model, easily accessible and, at the same time, quite expressive.

In order to keep the notation simple, tuples of variables like $x_1, \ldots, x_n$ will be denoted by $\vec{x}$ and, abusing the notation, tuples will be often used as sets.

Definition 3.1 (*syntax*). Let Var be a denumerable set of variables ranged over by $x, y, z, \ldots$. The syntax of the logic $\mathcal{L}$ over the set of labels Λ is defined as follows, where a ranges over Λ:

$$\varphi \ ::= \ \mathsf{T} \ \mid \ \varphi \wedge \varphi \ \mid \ \neg\varphi \ \mid \ (\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi \ \mid \ \langle z \rangle\, \varphi$$

The operator $(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)$ acts as a binder for the variable z, as clarified by the following notion of free variables in a formula.

Definition 3.2 (*free variables*). The set of free variables of a formula φ, denoted $fv(\varphi)$, is inductively defined by:

$$\begin{array}{ll} fv(\mathsf{T}) & = \emptyset \\ fv(\varphi_1 \wedge \varphi_2) & = fv(\varphi_1) \cup fv(\varphi_2) \\ fv(\neg\varphi) & = fv(\varphi) \\ fv((\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi) & = \vec{x} \cup \vec{y} \cup (fv(\varphi) \setminus \{z\}) \\ fv(\langle z \rangle\, \varphi) & = fv(\varphi) \cup \{z\} \end{array}$$

The satisfaction of a formula φ is defined with respect to a configuration $C \in \mathcal{C}(\mathcal{E})$, representing the state of the computation, and a (total) function $\eta : Var \to E$, called an *environment*, that binds free variables in φ to events in C or in the future of C. In particular, the events bound to free variables in a formula must be both pairwise consistent and consistent with the current state of the computation. Such a requirement is expressed by the following definition of legal pair.

Definition 3.3 (*environments, legal pairs*). Let $\mathcal{E}$ be a PES. We denote by $Env_{\mathcal{E}}$ the set of environments $\eta : Var \to E$. Given a formula φ in $\mathcal{L}$, a pair $(C, \eta) \in \mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}}$ is *legal* for φ if $C \cup \eta(fv(\varphi))$ is a consistent set of events. We denote by $lp_{\mathcal{E}}(\varphi)$ the set of legal pairs for φ in $\mathcal{E}$.

Remark. Observe that the legal pairs for a formula only depends on its set of free variables. Whenever $fv(\varphi) = fv(\psi)$ it holds that $lp_{\mathcal{E}}(\varphi) = lp_{\mathcal{E}}(\psi)$. More generally, if $fv(\varphi) \subseteq fv(\psi)$ then $lp_{\mathcal{E}}(\varphi) \supseteq lp_{\mathcal{E}}(\psi)$.

We simply write Env and $lp(\varphi)$, omitting the subscript, when the PES $\mathcal{E}$ is clear from the context. Moreover, in order to simplify the definition of the semantics, given a configuration C, we denote by $E[C]$ the *residual* of E after C, defined as $E[C] = \{e \mid e \in E \setminus C \ \wedge \ C \frown e\}$.

Definition 3.4 (*semantics*). Let $\mathcal{E}$ be a PES. The denotation of a formula φ, written $\{\!|\varphi|\!\}^{\mathcal{E}} \in 2^{\mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}}}$, is defined inductively as follows:

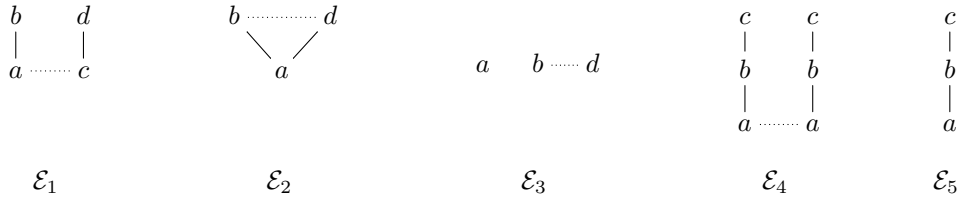


Fig. 1.

$$
\begin{aligned}
\{\!|\mathsf{T}|\!\}^{\mathcal{E}} &= \mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}} \\
\{\!|\varphi_1 \wedge \varphi_2|\!\}^{\mathcal{E}} &= \{\!|\varphi_1|\!\}^{\mathcal{E}} \cap \{\!|\varphi_2|\!\}^{\mathcal{E}} \cap lp(\varphi \wedge \psi) \\
\{\!|\neg\varphi|\!\}^{\mathcal{E}} &= lp(\varphi) \setminus \{\!|\varphi|\!\}^{\mathcal{E}} \\
\{\!|(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi|\!\}^{\mathcal{E}} &= \{(C,\eta) \mid\ (C,\eta) \in lp((\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi) \text{ and} \\
&\qquad \exists e \in E[C] \text{ such that } e \frown \eta(fv(\varphi) \setminus \{z\}) \\
&\qquad \wedge\ \lambda(e) = \mathsf{a}\ \wedge\ \eta(\vec{x}) < e\ \wedge\ \eta(\vec{y})\,||\,e \\
&\qquad \wedge\ (C, \eta[z \mapsto e]) \in \{\!|\varphi|\!\}^{\mathcal{E}} \qquad\qquad \}
\end{aligned}
$$

$$
\{\!|\langle z\rangle\ \varphi|\!\}^{\mathcal{E}} = \{(C,\eta) \mid\ C \xrightarrow{\eta(z)} C'\ \wedge\ (C',\eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}\ \}
$$

When $(C,\eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}$ we say that the PES $\mathcal{E}$ satisfies the formula φ in the configuration C and environment $\eta : Var \to E$, and write $\mathcal{E}, C \models_\eta \varphi$. For closed formulae φ, we write $\mathcal{E}, C \models \varphi$, when $\mathcal{E}, C \models_\eta \varphi$ for some η and $\mathcal{E} \models \varphi$, when $\mathcal{E}, \emptyset \models \varphi$.

Intuitively, the formula

$$(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi$$

holds in (C,η) when in the future of the configuration C there is an a-labelled event e, consistent with the events bound to free variables in φ, such that binding e to variable z, the formula φ holds. Such an event is required to be caused (at least) by the events already bound to variables in $\vec{x}$, and to be independent (at least) from those bound to variables in $\vec{y}$. We stress that the event e might not be currently enabled; it is only required to be consistent with the current configuration, meaning that it could be enabled in the future of the current configuration. The formula $\langle z\rangle\, \varphi$ says that the event bound to z is enabled by the current configuration, hence it can be executed producing a new configuration which satisfies the formula φ. To simplify the notation we write $(\mathsf{a}\, z)\, \varphi$ for $(\ < \mathsf{a}\, z)\, \varphi$.

As an example, consider the PES $\mathcal{E}_1$ in Fig. 1, corresponding to the CCS process $a.b + c.d$, where dotted lines represent immediate conflict and the causal order proceeds upwards along the straight lines. The empty configuration satisfies the closed formula $(\mathsf{b}\, x)\mathsf{T}$, i.e., $\mathcal{E}_1 \models (\mathsf{b}\, x)\mathsf{T}$, even if the b-labelled event is not immediately enabled. Also $\mathcal{E}_1 \models (\mathsf{b}\, x)\mathsf{T} \wedge (\mathsf{d}\, y)\mathsf{T}$, since there are two possible (incompatible) computations that start from the empty configuration and contain, respectively, a b-labelled and a d-labelled event. On the other hand, if $\varphi = (\mathsf{a}\, z)\langle z\rangle\, ((\mathsf{b}\, x)\mathsf{T} \wedge (\mathsf{d}\, y))\mathsf{T}$ then $\mathcal{E}_1 \not\models \varphi$ since after the execution of the a-labelled event, $\mathcal{E}_1$ reaches a configuration that does not admit a future containing an event labelled by d. As a further example, the formula φ above is satisfied by the PESs $\mathcal{E}_2$ and $\mathcal{E}_3$ in Fig. 1 corresponding respectively to the process $a.(b+d)$ and $a \mid (b+d)$, whereas the formula $(\mathsf{a}\, z)\langle z\rangle\, (\overline{z} < \mathsf{b}\, x)\mathsf{T}$ is satisfied only by $\mathcal{E}_3$.

It is worth noticing that the semantics of the binding operator does not prevent from choosing for z an event e that has been already bound to a different variable, i.e., the environment function η need not be injective. This is essential to avoid the direct observation of conflicts, a capability which would make the logical equivalence stronger than hhp-bisimilarity (and of any reasonable behavioural equivalence). Consider for instance the PESs associated to the hhp-equivalent processes $a + a$ and a: in order to be also logically equivalent, they both must satisfy the formula $(\mathsf{a}\, z)(\mathsf{a}\, z')\mathsf{T}$. Hence for the second PES, both z and z' must be bound to the unique a-labelled event. On the other hand, observe that both PESs falsify the formula $(\mathsf{a}\, z)(\mathsf{a}\, z')\langle z\rangle\, \langle z'\rangle\, \mathsf{T}$. In fact, z' must be bound to an event consistent with that associated to z (because z occurs free in $\langle z\rangle\, \langle z'\rangle\, \mathsf{T}$). Hence z and z' will be bound to the same event, which cannot be executed twice.

3.1. About legal pairs and environments

We remark that differently from other logics for event structures, whose semantics is given only with respect to the set of configurations, here legal pairs come into play in order to ensure that the events bound to free variables in a formula be consistent with the current state of the computation and pairwise consistent. The intuition is that, in a legal pair for a formula, the configuration identifies the current state of the computation and the environment should map variables free in the formula to events which have already occurred or which can occur in a possible future of the current state.

The use of legal pairs has some subtle effects on the semantics of the propositional connectives. In particular, concerning negation, it is immediate to see that a pair (C, η) is legal for φ if and only if it is legal for $\neg\varphi$. Hence, when a denotation (C, η) is not legal for φ, we have that neither $\mathcal{E}, C \models_\eta \varphi$ nor $\mathcal{E}, C \models_\eta \neg\varphi$. As a concrete example, take $\varphi = \langle x\rangle\, \langle y\rangle\, \mathsf{T}$. Then in the PES $\mathcal{E}_1$ of Fig. 1, if η binds x and y to the conflicting events labelled a and c, respectively, then $(\emptyset, \eta)$ is not legal for φ and we have $\mathcal{E}_1, \emptyset \not\models_\eta \varphi$ and $\mathcal{E}_1, \emptyset \not\models_\eta \neg\varphi$.

For closed formulae, we have the following:

LEMMA 3.5 (NEGATION). *Let φ be a closed formula in $\mathcal{L}$, let $\mathcal{E}$ be a* PES *and let $(C, \eta) \in \mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}}$. Then $\mathcal{E}, C \models_\eta \varphi$ iff $\mathcal{E}, C \not\models_\eta \neg\varphi$.*

PROOF. Immediately follows from the observation that for a closed formula any pair is legal. □

Concerning conjunction, observe that it is not the case that $lp(\varphi \wedge \psi) = lp(\varphi) \cap lp(\psi)$. Therefore it can happen that $\mathcal{E}, C \models_\eta \varphi$ and $\mathcal{E}, C \models_\eta \psi$, but $\mathcal{E}, C \not\models_\eta \varphi \wedge \psi$. As an example, consider again the PES $\mathcal{E}_1$ of Fig. 1, and the formulae $\varphi = \langle x\rangle\, \mathsf{T}$ and $\psi = \langle y\rangle\, \mathsf{T}$. If η binds x and y to the events labelled a and c, respectively, then $(\emptyset, \eta) \in lp(\varphi)$, $(\emptyset, \eta) \in lp(\varphi)$ and we have $\mathcal{E}_1, \emptyset \models_\eta \varphi$ and $\mathcal{E}_1, \emptyset \models_\eta \psi$. However, since the two events are in conflict, $(\emptyset, \eta) \notin lp(\varphi \wedge \psi)$, and thus $\mathcal{E}_1, \emptyset \not\models_\eta \varphi \wedge \psi$.

We next show that the denotation of a formula, given according to Definition 3.4, always consists of a set of legal pairs for the formula.

LEMMA 3.6 (DENOTATIONS CONSIST OF LEGAL PAIRS). *Let $\mathcal{E}$ be a* PES. *Then for any formula $\varphi \in \mathcal{L}$, it holds $\{\!|\varphi|\!\}^{\mathcal{E}} \subseteq lp_{\mathcal{E}}(\varphi)$*

PROOF. The proof is by routine induction on the structure of the formula φ. We only comment case $\varphi = \langle z\rangle\, \psi$. If $(C, \eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}$ then, by definition, if we let $e = \eta(z)$, it holds that $C \xrightarrow{e} C \cup \{e\}$ and $(C \cup \{e\}, \eta) \in \{\!|\psi|\!\}^{\mathcal{E}}$. Hence by inductive hypothesis $(C \cup \{e\}, \eta) \in lp_{\mathcal{E}}(\psi)$, i.e., $C \cup \{e\} \cup \eta(fv(\psi))$ is consistent. Since $fv(\varphi) = fv(\psi) \cup \{z\}$, we have that $C \cup \eta(fv(\varphi)) = C \cup \{e\} \cup \eta(fv(\psi))$, and thus we can conclude $(C, \eta) \in lp_{\mathcal{E}}(\varphi)$. □

The semantics of a formula only depends on the events that the environment associates to the free variables of the formula.

LEMMA 3.7. *Let $\mathcal{E}$ be a* PES *and let $C \in \mathcal{C}(E)$. Let $\varphi \in \mathcal{L}$ and let $\eta_1, \eta_2 : Var \to E$ be environments such that $\eta_1(x) = \eta_2(x)$ for any $x \in fv(\varphi)$. Then*

$$\mathcal{E}, C \models_{\eta_1} \varphi \qquad \textit{iff} \qquad \mathcal{E}, C \models_{\eta_2} \varphi$$

In particular, $(C, \eta_1) \in lp_{\mathcal{E}}(\varphi)$ if and only if $(C, \eta_2) \in lp_{\mathcal{E}}(\varphi)$.

PROOF. Routine induction on the structure of φ. □

Note that without restricting the semantics of formulae to legal pairs the logics would have been too powerful. In fact, it would have allowed us to observe conflicts through a combination of the binder and the execution modality. For instance, consider the PESs $\mathcal{E}_4$ and $\mathcal{E}_5$ in Fig. 1, corresponding to the processes $a.b.c + a.b.c$ and $a.b.c$, respectively, and take formula $\varphi = (\mathsf{a}\, x)(\mathsf{b}\, y)\langle x\rangle \neg\langle y\rangle \mathsf{T}$, saying that there are two events labelled by a and b such that after executing the first, the second cannot be executed. With the current definition neither $\mathcal{E}_4$ nor $\mathcal{E}_5$ satisfy φ, since after binding x to any a-labelled event e, in order to keep the denotation legal, y must be bound to the b-labelled event caused by e, that is executable after e. Without the restriction to legal pairs, instead, the formula would hold in $\mathcal{E}_4$, since variables x and y could be bound to conflicting events (e.g., x could be bound to the a-labelled event on the left and y to the b-labelled event on the right). Similarly, consider the formula $\psi = (\mathsf{a}\ x)(\mathsf{b}\ y)\neg(x, y < \mathsf{c}\ z)\mathsf{T}$, saying that there are two events, labelled by a and b, respectively, which are not common causes for any c-labelled event. Also ψ does not hold neither in $\mathcal{E}_4$ nor in $\mathcal{E}_5$. Omitting the restriction to legal pairs, ψ would be true only in $\mathcal{E}_4$ where x and y can be bound to conflicting events. This means that the logic would distinguish the PESs corresponding to a process from that corresponding to the non-deterministic choice between the process and itself, which instead are equated by virtually any behavioural equivalence.

Instead of restricting the semantics of formulae to legal pairs, one could envisage syntactic constraints which produce essentially the same effect, thus limiting the observation power of the logic. The idea is quite simple: in any formula, whenever we bind an event to a variable z, we require that the binder operator explicitly states the consistency of z with the free variables appearing in the remaining part of the formula. Specifically, for any subformula of the kind $(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \psi$, we could require the free variables of ψ to be a subset of $\vec{x} \cup \vec{y} \cup \{z\}$. In this way we are guaranteed that the event bound to z is either causally dependent or concurrent (hence consistent) with the events bound to the free variables of the formula. This essentially gives the same effect as restricting the semantics to legal pairs. It can be seen that restricting to the fragment of $\mathcal{L}$ consisting of well-formed formulae does not alter the logical equivalence which remains hhp-bisimilarity, as for the full logic. A more detailed account of this alternative approach is given in the Appendix A.

3.2. Dual operators

Relying on negation we can define operators which are dual to those primitive in the logic. As usual, disjunction $\varphi \vee \psi$ can be defined by the formula $\neg(\neg\varphi \wedge \neg\psi)$. Its semantics, according to Definition 3.4, turns out to be:

$$\{\!|\varphi \vee \psi|\!\}^{\mathcal{E}} = (\{\!|\varphi|\!\}^{\mathcal{E}} \cup \{\!|\psi|\!\}^{\mathcal{E}}) \cap lp(\varphi \vee \psi).$$

The formula F (false) is defined by $\neg\mathsf{T}$, with semantics:

$$\{\!|\mathsf{F}|\!\}^{\mathcal{E}} = \emptyset.$$

Moreover, we write

$$\{\vec{x},\overline{\vec{y}} < \mathsf{a}\, z\}\,\varphi \qquad \text{for the formula} \qquad \neg((\vec{x},\overline{\vec{y}} < \mathsf{a}\, z)\,\neg\varphi).$$
$$[\,z\,]\;\varphi \qquad \text{for the formula} \qquad \neg(\langle z\rangle\;\neg\varphi)$$

The dual of the binder has a universal flavour. In fact its semantics, given explicitly below, involves a universal quantification:

$$\begin{aligned}\{\!|\{\vec{x},\overline{\vec{y}} < \mathsf{a}\, z\}\,\varphi|\!\}^{\mathcal{E}} \;=\; \{(C,\eta) \mid\; & (C,\eta) \in lp(\{\vec{x},\overline{\vec{y}} < \mathsf{a}\, z\}\,\varphi) \text{ and}\\ & \forall e \in E[C] \text{ such that } e \frown \eta(fv(\varphi)\setminus\{z\})\\ & \wedge\, \lambda(e) = \mathsf{a} \;\wedge\; \eta(\vec{x}) < e \;\wedge\; \eta(\vec{y})\,||\,e\\ & \text{it holds } (C,\eta[z\mapsto e]) \in \{\!|\varphi|\!\}^{\mathcal{E}} \qquad\qquad \}\end{aligned}$$

i.e., $\mathcal{E},C \models_\eta \{\vec{x},\overline{\vec{y}} < \mathsf{a}\, z\}\,\varphi$ when for all a-labelled events e in the future of C, consistent with the events already bound to $fv(\varphi)$, caused by $\eta(\vec{x})$ and concurrent with $\eta(\vec{y})$, we have that binding e to z the formula φ holds.

The semantics of $[\,\cdot\,]$, instead, is:

$$\begin{aligned}\{\!|[\,z\,]\;\varphi|\!\}^{\mathcal{E}} \;=\; \{(C,\eta) \mid\; & (C,\eta) \in lp([\,z\,]\,\varphi) \text{ and}\\ & \text{if } C \xrightarrow{\eta(z)} C' \text{ then } (C',\eta) \in \{\!|\varphi|\!\}^{\mathcal{E}} \;\}\end{aligned}$$

namely, $\mathcal{E},C \models_\eta [\,z\,]\,\varphi$ if, either $\eta(z)$ is not executable from C or it is executable and in the reached configuration φ holds.

The logic $\mathcal{L}$ could be alternatively defined in positive form by including the dual operators and omitting negation. The syntax of the resulting logic, denoted $\mathcal{L}^+$, would be as follows:

$$\varphi \;::=\; \mathsf{T} \mid \mathsf{F} \mid \varphi\wedge\varphi \mid \varphi\vee\varphi \mid (\vec{x},\overline{\vec{y}} < \mathsf{a}\, z)\,\varphi \mid \{\vec{x},\overline{\vec{y}} < \mathsf{a}\, z\}\,\varphi \mid \langle z\rangle\,\varphi \mid [\,z\,]\,\varphi$$

Negation is then encodable in $\mathcal{L}^+$ by duality. Hereafter we will freely use the dual operators.

3.3. Examples and notation

In this subsection we provide some more examples illustrating the expressiveness of the logic. We start by introducing some handy notation, which will improve the readability of the formulae.

Immediate execution. We will write

$$\langle\!|\vec{x},\overline{\vec{y}} < \mathsf{a}\, z|\!\rangle\,\varphi \quad \text{for the formula} \quad (\vec{x},\overline{\vec{y}} < \mathsf{a}\, z)\langle z\rangle\,\varphi$$

that states the existence of an event e enabled by the current configuration, and thus which can be immediately executed, such that after executing e the formula φ holds (with e bound to variable z). Dually we introduce the notation $[\![\vec{x},\overline{\vec{y}} < \mathsf{a}\, z]\!]\,\varphi$, which stands for the formula $\{\vec{x},\overline{\vec{y}} < \mathsf{a}\, z\}[\,z\,]\,\varphi$.

Steps. We introduce a notation also to predicate the existence, resp., the immediate execution, of concurrent events, specifying also their dependencies. We will write

$$((\vec{x},\overline{\vec{y}} < \mathsf{a}\, z)\otimes(\vec{x}',\overline{\vec{y}'} < \mathsf{b}\, z'))\,\varphi \quad \text{for the formula} \quad (\vec{x},\overline{\vec{y}} < \mathsf{a}\, z)(\vec{x}',\overline{\vec{y}',z} < \mathsf{b}\, z')\varphi$$
$$(\langle\!|\vec{x},\overline{\vec{y}} < \mathsf{a}\, z|\!\rangle \otimes \langle\!|\vec{x}',\overline{\vec{y}'} < \mathsf{b}\, z'|\!\rangle)\varphi \quad \text{for the formula} \quad ((\vec{x},\overline{\vec{y}} < \mathsf{a}\, z)\otimes(\vec{x}',\overline{\vec{y}'} < \mathsf{b}\, z'))\langle z\rangle\,\langle z'\rangle\,\varphi$$

The first formula declares the existence of two concurrent events, labelled by a and b, respectively, such that if we bind such events to z and z', then φ holds. The second

Fig. 2.

formula states the existence of two concurrently enabled events, labelled by a and b, whose immediate execution leads to a state where φ holds. In particular, the ability to perform a step consisting of two concurrent events labelled by a and b is simply expressed by the formula $(\langle\!\langle \mathsf{a}\, x\rangle\!\rangle \otimes \langle\!\langle \mathsf{b}\, y\rangle\!\rangle)\mathsf{T}$.

Clearly, this notation can be generalised to the quantification and the immediate execution of any number of concurrent events.

An analogous notation will be used for the dual operators:

$$(\{\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z\} \otimes \{\vec{x}', \overline{\vec{y}'} < \mathsf{b}\, z'\})\,\varphi \qquad \text{and} \qquad ([\![\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z]\!] \otimes [\![\vec{x}', \overline{\vec{y}'} < \mathsf{b}\, z']\!])\varphi$$

The first formula asserts that considering any pair of concurrent events, labelled a and b, respectively, which are bound to z and z', the formula φ holds. The second formula states that the after the execution of all pairs of concurrent events, labelled a and b, respectively, the formula φ holds.

Example 3.8 (*interleaving vs. true concurrency*). Consider the PESs $\mathcal{E}_6$ and $\mathcal{E}_7$ in Fig. 2. They are equated by interleaving equivalences and distinguished by any true concurrent equivalence. The formula $\varphi_1 = \langle\!\langle \mathsf{a}\, x\rangle\!\rangle\langle\!\langle \overline{x} < \mathsf{b}\, y\rangle\!\rangle\mathsf{T} = (\langle\!\langle \mathsf{a}\, x\rangle\!\rangle \otimes \langle\!\langle \mathsf{b}\, y\rangle\!\rangle)\mathsf{T}$ is true only on $\mathcal{E}_7$, while $\varphi_2 = \langle\!\langle \mathsf{a}\, x\rangle\!\rangle\langle\!\langle x < \mathsf{b}\, y\rangle\!\rangle\mathsf{T}$ is true only on $\mathcal{E}_6$.

Wildcard operators. It is often useful to have a wildcard operator to refer to an event with an arbitrary label. When the set of labels Λ is finite, we write

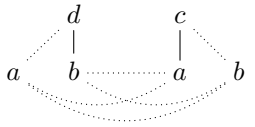
$$(\vec{x}, \overline{\vec{y}} < \_\, z)\varphi$$

to denote the formula $\bigvee_{\mathsf{a}\in\Lambda}(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\varphi$, and we use an analogous notation for the induced operators. For instance, the formula $(\langle\!\langle \_\, x_1\rangle\!\rangle \otimes \langle\!\langle \_\, x_2\rangle\!\rangle)\mathsf{T} \wedge \neg(\langle\!\langle \_\, y_1\rangle\!\rangle \otimes \langle\!\langle \_\, y_2\rangle\!\rangle \otimes \langle\!\langle \_\, y_3\rangle\!\rangle)\mathsf{T}$ states that in the current state there is a step consisting of two concurrent events and this is the maximal size for a step. When the set of labels Λ is infinite the same wildcard operators are no longer expressible in the finitary logic $\mathcal{L}$. However they can be added to $\mathcal{L}$ while retaining all the results in the paper. More precisely, logical equivalence for $\mathcal{L}$ would be still hhp-bisimilarity. In fact, by adding the wildcard operators logical equivalence becomes potentially finer and thus the fact that it implies hhp-bisimilarity (Proposition 4.2) clearly remains true. Conversely, finiteness of conjunctions plays no role in the proof of Proposition 4.4, hence it can be easily seen that hhp-bisimilarity implies logical equivalence even for an infinitary version of the logic $\mathcal{L}$ (explicitly introduced in Section 6.2 and denoted $\mathcal{L}^\infty$) where wildcard operators can be encoded. The same applies to the various fragments of $\mathcal{L}$ and to the logics with recursion.

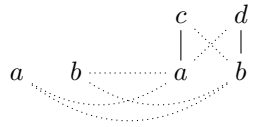
Example 3.9 (*causality and concurrency*). Consider the PESs $\mathcal{E}_6$ and $\mathcal{E}_8$ in Fig. 2. They are distinguished by all true concurrent equivalences, but since they share the same causal structure, in order to pinpoint how they differ, the logic must be able to express the presence of two concurrent events. Logic $\mathcal{L}$ can do this in a quite direct way, e.g., $\mathcal{E}_8 \models (\langle\!\langle \mathsf{a}\, x\rangle\!\rangle \otimes \langle\!\langle \mathsf{b}\, y\rangle\!\rangle)\mathsf{T}$, while $\mathcal{E}_6 \not\models (\langle\!\langle \mathsf{a}\, x\rangle\!\rangle \otimes \langle\!\langle \mathsf{b}\, y\rangle\!\rangle)\mathsf{T}$. On the other hand, PESs $\mathcal{E}_7$ and $\mathcal{E}_9$, roughly speaking, exhibit the same concurrency and indeed they are equated

by step bisimilarity. However they have a different causal structure and thus they are distinguished by any equivalence which observes causality, e.g., pomset bisimilarity. The logic can take them apart by predicating directly about causality, e.g., $\mathcal{E}_9$ satisfies $\langle\!\langle \mathsf{a}\, x\rangle\!\rangle\langle\!\langle x < \mathsf{b}\, y\rangle\!\rangle\mathsf{T}$, while $\mathcal{E}_7$ does not.

Example 3.10 (*conflicting futures*). Consider the PESs below which can be proved to be hp-bisimilar but not hhp-bisimilar (the example is taken from [Joyal et al. 1996]):



$\mathcal{E}_{10}$



$\mathcal{E}_{11}$

Intuitively, they differ since the causes of the events labelled by c and d, respectively, are in conflict in $\mathcal{E}_{10}$ and concurrent in $\mathcal{E}_{11}$. This difference can be captured by the formula $\varphi = ((\mathsf{a}\, x) \otimes (\mathsf{b}\, y))((x < \mathsf{c}\, z_1)\mathsf{T} \wedge (y < \mathsf{d}\, z_2)\mathsf{T})$, which is satisfied only by $\mathcal{E}_{11}$. Notice that the formula φ exploits the ability of the logic $\mathcal{L}$ of quantifying over events in conflict with previously bound events: formula φ is satisfied in $\mathcal{E}_{11}$ by binding x and y to the rightmost a-labelled and b-labelled events; then z_1 and z_2 are bound to events which are in conflict with either x or y. For this, the possibility of "observing" an event without executing it is essential: the formula $\varphi' = (\langle\!\langle \mathsf{a}\, x\rangle\!\rangle \otimes \langle\!\langle \mathsf{b}\, y\rangle\!\rangle)((x < \mathsf{c}\, z_1)\mathsf{T} \wedge (y < \mathsf{d}\, z_2)\mathsf{T})$ would be false for both PESs since the execution of the first two events leads to a configuration that is no further extensible.

As a last example, consider the CCS processes $P = a|(b+c) + a|b + b|(a+c)$ and $Q = a|(b+c) + b|(a+c)$, equated by the absorption law (see, e.g., [van Glabbeek and Goltz 2001]). They contain no causal dependencies, but they exhibit a different interplay between concurrency and branching. Accordingly, the corresponding PESs can be proved to be hp-bisimilar but not hhp-bisimilar. Intuitively, this difference arises from the fact that only the process P includes two concurrent events a and b such that, once their execution has started, by firing one of them, no c-labelled event will ever be enabled. Such a difference can be expressed in $\mathcal{L}$ by the formula $((\mathsf{a}\, x) \otimes (\mathsf{b}\, y))(\neg(\overline{x} < \mathsf{c}\, z)\mathsf{T} \wedge \neg(\overline{y} < \mathsf{c}\, z')\mathsf{T})$, which says that there are two concurrent events labelled a and b, respectively, such that none of them is concurrent with a c-labelled event. This is clearly satisfied only by the PES corresponding to P.

4. A LOGICAL CHARACTERISATION OF HHP-BISIMILARITY

We next study the logical equivalence induced by $\mathcal{L}$. We have already argued that no formula in $\mathcal{L}$ distinguishes the PESs a and $a\#a$, hence the logical equivalence induced by $\mathcal{L}$ is surely coarser than isomorphism. In this section we will show that it coincides with hhp-bisimilarity.

Since later we will also identify suitable fragments of $\mathcal{L}$ corresponding to coarser equivalences, we define logical equivalence for a generic fragment of $\mathcal{L}$.

Definition 4.1 (*logical equivalence*). Let $\mathcal{L}'$ be a fragment of $\mathcal{L}$. We say that two PES $\mathcal{E}_1, \mathcal{E}_2$ are *logically equivalent* in $\mathcal{L}'$, written $\mathcal{E}_1 \equiv_{\mathcal{L}'} \mathcal{E}_2$ when they satisfy the same closed formulae of $\mathcal{L}'$.

We first prove that two PES's satisfying the same formulae in $\mathcal{L}$ are hhp-bisimilar.

PROPOSITION 4.2. *Let $\mathcal{E}_1$ and $\mathcal{E}_2$ be* PES*s such that $\mathcal{E}_1 \equiv_{\mathcal{L}} \mathcal{E}_2$, then $\mathcal{E}_1 \sim_{hhp} \mathcal{E}_2$.*

PROOF. Let us start by introducing some notation. We fix a surjective environment $\eta_1 : Var \to E_1$. Then given an event $e \in E_1$, we write x_e to denote a fixed distinguished

variable such that $\eta_1(x_e) = e$. Similarly, for a configuration $C_1 = \{e_1, \ldots, e_n\}$ we denote by X_{C_1} the set of variables $\{x_{e_1}, \ldots, x_{e_n}\}$. Observe that $(\emptyset, \eta_1)$ is a legal pair for any formula $\varphi \in \mathcal{L}$ such that $fv(\varphi) \subseteq X_{C_1}$, since $\emptyset \cup \eta(fv(\varphi)) \subseteq C_1$, which is consistent. Consider the posetal relation $R \subseteq \mathcal{C}(\mathcal{E}_1) \bar{\times} \mathcal{C}(\mathcal{E}_2)$ defined by:

$$R \;=\; \{\, (C_1, f, C_2) \mid \forall \psi \in \mathcal{L}.\, fv(\psi) \subseteq X_{C_1} \quad (\mathcal{E}_1, \emptyset \models_{\eta_1} \psi \text{ iff } \mathcal{E}_2, \emptyset \models_{f \circ \eta_1} \psi) \,\} \tag{1}$$

where, for an isomorphism of pomsets $f : C_1 \to C_2$, we denote by $f \circ \eta_1$ an environment such that $f \circ \eta_1(x) = f(\eta_1(x))$ for $x \in X_{C_1}$ and $f \circ \eta_1(x)$ has any value, otherwise. Note that this does not introduce ambiguities since, by Lemma 3.7, the semantics of ψ only depends on the value of the environment on $fv(\psi)$ and $fv(\psi) \subseteq X_{C_1}$ by construction.

Observe that, since by hypothesis $\mathcal{E}_1 \equiv_{\mathcal{L}} \mathcal{E}_2$, we have that $(\emptyset, \emptyset, \emptyset) \in R$. Hence in order to conclude it is sufficient to show that R is a hhp-bisimulation.

— R is downward closed

Take $(C_1, f, C_2) \in R$ and consider $(C_1', f', C_2') \subseteq (C_1, f, C_2)$ pointwise. We have to show that $(C_1', f', C_2') \in R$.

Let ψ be any formula such that $fv(\psi) \subseteq X_{C_1'}$. Since $C_1' \subseteq C_1$, clearly $fv(\psi) \subseteq X_{C_1}$ and thus, since $(C_1, f, C_2) \in R$, by definition of R (1), we have that

$$\mathcal{E}_1, \emptyset \models_{\eta_1} \psi \quad \text{iff} \quad \mathcal{E}_2, \emptyset \models_{f \circ \eta_1} \psi,$$

Moreover, since $fv(\psi) \subseteq X_{C_1'}$, $\eta_1(X_{C_1'}) = C_1'$ and $f' = f_{|C_1'}$, we have that $(f \circ \eta_1)_{|fv(\psi)} = (f' \circ \eta_1)_{|fv(\psi)}$ and thus, by Lemma 3.7,

$$\mathcal{E}_2, \emptyset \models_{f \circ \eta_1} \psi \qquad \text{iff} \qquad \mathcal{E}_2, \emptyset \models_{f' \circ \eta_1} \psi$$

Summing up, for any ψ such that $fv(\psi) \subseteq X_{C_1'}$, it holds that $\mathcal{E}_1, \emptyset \models_{\eta_1} \psi$ iff $\mathcal{E}_2, \emptyset \models_{f' \circ \eta_1} \psi$. Therefore $(C_1', f', C_2') \in R$, as desired.

— R is a hp-bisimulation

We have to show that given $(C_1, f, C_2) \in R$, if $C_1 \xrightarrow{\;e\;} C_1'$ then there exists a transition $C_2 \xrightarrow{\;g\;} C_2'$ such that $f' = f[e \mapsto g] : C_1' \to C_2'$ is an isomorphism of pomsets (hence in particular $\lambda_1(e) = \lambda_2(g)$) and $(C_1', f', C_2') \in R$.

We proceed by contradiction. Since all PESs are assumed to be image finite, there are finitely many transitions $C_2 \xrightarrow{\;g^i\;} C_2^i$, with $i \in \{1, \ldots, n\}$, such that $C_1' \sim C_2^i$ (as pomsets). By contradiction assume that, for any $i \in \{1, \ldots, n\}$, it holds $(C_1', f^i, C_2^i) \notin R$. Hence, by definition of R (1), there exists a formula ψ^i such that

$$\mathcal{E}_1, \emptyset \models_{\eta_1} \psi^i \quad \text{and} \quad \mathcal{E}_2, \emptyset \not\models_{f^i \circ \eta_1} \psi^i$$

where $fv(\psi^i) \subseteq X_{C_1'} = X_{C_1} \cup \{x_e\}$ and $f^i = f[e \mapsto g^i]$. Observe that it could either be that $\mathcal{E}_1, \emptyset \not\models_{\eta_1} \psi^i$ and $\mathcal{E}_2, \emptyset \models_{f^i \circ \eta_1} \psi^i$, but we can reduce to the case above by taking the negation of ψ^i. In fact, since $fv(\psi^i) \subseteq X_{C_1'}$, we have that $(\emptyset, \eta_1) \in lp_{\mathcal{E}_1}(\psi^i)$, and thus from $\mathcal{E}_1, \emptyset \not\models_{\eta_1} \psi^i$ we deduce $\mathcal{E}_1, \emptyset \models_{\eta_1} \neg\psi^i$. Moreover, since $\mathcal{E}_2, \emptyset \models_{f^i \circ \eta_1} \psi^i$ we have $\mathcal{E}_2, \emptyset \not\models_{f^i \circ \eta_1} \neg\psi^i$.

Consider the formula

$$\varphi = (\vec{x}, \vec{\overline{y}} < \mathsf{a}\, x_e)(\langle X_{C_1} \rangle \langle x_e \rangle \mathsf{T} \wedge \psi^1 \wedge \ldots \wedge \psi^n)$$

where $\mathsf{a} = \lambda_1(e)$ and the $\vec{x}, \vec{y} \subseteq X_{C_1}$ are such that $\eta_1(\vec{x})$ is the set of causes of e in C_1 and $\eta_1(\vec{y})$ is the set of events in C_1 which are concurrent with e. Note that

$$fv(\varphi) = \vec{x} \cup \vec{y} \cup ((X_{C_1} \cup \{x_e\} \cup \textstyle\bigcup_{i=1}^n fv(\psi_i)) \setminus \{x_e\}) = X_{C_1}$$

In fact, by construction, $\vec{x} \cup \vec{y} = X_{C_1}$ and $fv(\psi^i) \subseteq X_{C_1'} = X_{C_1} \cup \{x_e\}$.
Now, it is easy to see that $\mathcal{E}_1, \emptyset \models_{\eta_1} \varphi$. Moreover $\mathcal{E}_2, \emptyset \not\models_{f\circ\eta_1} \varphi$. In fact, an event $g \in E_2$ such that $f \circ \eta_1(\vec{x}) < g$, $f \circ \eta_1(\vec{y}) \,||\, g$ and $\mathcal{E}_2, \emptyset \models_{f\circ\eta_1} \langle X_{C_1} \rangle \langle x_e \rangle$ is necessarily in the set $\{g^1, \ldots, g^n\}$, and thus, by construction, $\mathcal{E}_2, \emptyset \not\models_{f\circ\eta_1[x_e \mapsto g]} \psi^i$ for some $i \in \{1, \ldots, n\}$.
The existence of a formula φ which distinguishes C_1 and C_2 contradicts the hypothesis $(C_1, f, C_2) \in R$, as desired.

The fact that also the converse holds, i.e., if $C_2 \xrightarrow{g} C_2'$ then there exists a transition $C_1 \xrightarrow{e} C_1'$ such that $f' = f[e \mapsto g] : C_1' \to C_2'$ is an isomorphism of pomsets and $(C_1', f', C_2') \in R$, can be proved analogously. □

In order to prove that, conversely, hhp-bisimilar PESs satisfy the same $\mathcal{L}$ formulae, we first recall a lemma from [Bednarczyk 1991; van Glabbeek and Goltz 2001] which will be useful in the sequel.

LEMMA 4.3 (HHP-BISIMILARITY AS A PES). *Let $\mathcal{E}_1$, $\mathcal{E}_2$ be* PES*s such that $\mathcal{E}_1 \sim_{hhp} \mathcal{E}_2$ and let R be a hhp-bisimulation. Then there exists a* PES *$\mathcal{E}_R = \langle E_R, \leq_R, \#_R, \lambda_R \rangle$ such that for $i \in \{1, 2\}$*

— *$\mathcal{E}_i \sim_{hhp} \mathcal{E}_R$*
— *there are surjective maps $f_R^i : E_R \to E_i$ such that $\{ (C, f^i_{R|C}, f^i_R(C)) \mid C \in \mathcal{C}(\mathcal{E}_R)\}$ is a hhp-bisimulation.*

Additionally, each f_R^i preserves labels, causality $\leq$ and concurrency $||$, it maps configurations to configurations and it is injective on consistent sets of events.

PROOF SKETCH, FROM [BEDNARCZYK 1991; VAN GLABBEEK AND GOLTZ 2001]. We just recall the definition of $\mathcal{E}_R = \langle E_R, \leq_R, \#_R, \lambda_R \rangle$:

— $E_R = \{(e_1, f, e_2) \mid (\lceil e_1 \rceil, f, \lceil e_2 \rceil) \in R\}$,
— $(e_1, f, e_2) \leq_R (e_1', f', e_2')$ if $f \subseteq f'$,
— $(e_1, f, e_2) \#_R (e_1', f', e_2')$ if there exists no $(C, g, D) \in R$ such that $(\lceil e_1 \rceil, f, \lceil e_2 \rceil), (\lceil e_1' \rceil, f', \lceil e_2' \rceil) \subseteq (C, g, D)$ pointwise,
— $\lambda_R(e_1, f, e_2) = \lambda_1(e_1)$.

The maps $f_R^1 : E_R \to E_1$ and $f_R^2 : E_R \to E_2$ are just the projections on the first and third components, respectively. □

PROPOSITION 4.4. *Let $\mathcal{E}_1$ and $\mathcal{E}_2$ be* PES*s such that $\mathcal{E}_1 \sim_{hhp} \mathcal{E}_2$. Then $\mathcal{E}_1 \equiv_{\mathcal{L}} \mathcal{E}_2$.*

PROOF. Let R be a hhp-bisimulation relating $\mathcal{E}_1$ and $\mathcal{E}_2$. By Lemma 4.3, it is not restrictive to assume that $R = \{ (C_1, f_{|C_1}, f(C_1)) \}$, where $f : E_1 \to E_2$ is a surjective map satisfying the conditions in the statement of the lemma. Then it is sufficient to prove that for any formula $\varphi \in \mathcal{L}$, for any $(C_1, \eta_1) \in lp_{\mathcal{E}_1}(\varphi)$

$$\mathcal{E}_1, C_1 \models_{\eta_1} \varphi \quad \text{iff} \quad \mathcal{E}_2, f(C_1) \models_{f\circ\eta_1} \varphi \tag{2}$$

This implies, in particular, that $\mathcal{E}_1$ and $\mathcal{E}_2$ satisfy the same closed formulae, i.e., $\mathcal{E}_1 \equiv_{\mathcal{L}} \mathcal{E}_2$ as desired. In fact, given any closed formula φ, note that $(\emptyset, \eta_1) \in lp_{\mathcal{E}_1}(\varphi)$ for all environments η_1. Therefore if $\mathcal{E}_1 \models \varphi$, which means $\mathcal{E}_1, \emptyset \models_{\eta_1} \varphi$ for some η_1, we have $\mathcal{E}_2, \emptyset \models_{f\circ\eta_1} \varphi$, i.e., $\mathcal{E}_2 \models \varphi$. Vice versa, if $\mathcal{E}_2 \models \varphi$ then $\mathcal{E}_2, \emptyset \models_{\eta_2} \varphi$ for some $\eta_2 \in Env_{\mathcal{E}_2}$. Since φ is closed, by Lemma 3.7 the environment is irrelevant and thus, if we take any $\eta_1 \in Env_{\mathcal{E}_1}$, it holds $\mathcal{E}_2, \emptyset \models_{f\circ\eta_1} \varphi$. By this we get $\mathcal{E}_1, \emptyset \models_{\eta_1} \varphi$, which means $\mathcal{E}_1 \models \varphi$.
Now, in order to prove (2), first of all note that f preserves legal pairs, i.e., if $(C_1, \eta_1) \in lp_{\mathcal{E}_1}(\varphi)$ then $(f(C_1), f \circ \eta_1) \in lp_{\mathcal{E}_2}(\varphi)$ since f preserves consistency (as it preserves causality and concurrency).

The proof proceeds by induction on the formula φ:

— $\varphi = \mathsf{T}$
Immediate.

— $\varphi = \varphi_1 \wedge \varphi_2$
Let $(C_1, \eta_1) \in lp_{\mathcal{E}_1}(\varphi)$, hence $(C_1, \eta_1) \in lp_{\mathcal{E}_1}(\varphi_i)$ for $i \in \{1, 2\}$. If $\mathcal{E}_1, C_1 \models_{\eta_1} \varphi$, then, by definition of the semantics, we have $\mathcal{E}_1, C_1 \models_{\eta_1} \varphi_i$, for $i \in \{1, 2\}$. Thus we can use the inductive hypothesis to deduce that $\mathcal{E}_2, f(C_1) \models_{f\circ\eta_1} \varphi_i$, for $i \in \{1, 2\}$. Moreover, since f preserves legal pairs, we know that $(f(C_1), f \circ \eta_1) \in lp_{\mathcal{E}_2}(\varphi)$. Therefore $\mathcal{E}_2, f(C_1) \models_{f\circ\eta_1} \varphi$. The converse implication can be proved by just reverting all deductions.

— $\varphi = \neg\varphi_1$
Analogous to the previous case.

— $\varphi = (\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\psi$
Assume that $\mathcal{E}_1, C_1 \models_{\eta_1} \varphi$, with $(C_1, \eta_1) \in lp_{\mathcal{E}_1}(\varphi)$. Hence, by definition of the semantics, there exists an event $e \in E_1[C_1]$, such that $e \frown \eta_1(fv(\psi) \setminus \{z\})$, $\lambda_1(e) = \mathsf{a}$, $\eta_1(\vec{x}) \leq e$, $\eta_1(\vec{y}) \,||\, e$ and

$$\mathcal{E}_1, C_1 \models_{\eta_1'} \psi \tag{3}$$

where $\eta_1' = \eta_1[z \mapsto e]$.
By (3) and Lemma 3.6, $(C_1, \eta_1') \in lp_{\mathcal{E}_1}(\psi)$. Hence by inductive hypothesis $\mathcal{E}_2, f(C_1) \models_{f\circ\eta_1'} \psi$, with $f \circ \eta_1' = (f \circ \eta_1)[z \mapsto f(e)]$.
Since, by Lemma 4.3, f preserves consistency and it is injective on consistent sets of events, $f(e) \in E_2[f(C_1)]$. Additionally, again by Lemma 4.3, since f preserves labels, $\leq$ and $||$ (and hence $\frown$) we have that $f(e) \frown f \circ \eta_1(fv(\psi) \setminus \{z\})$, $\lambda_2(f(e)) = \lambda_1(e) = \mathsf{a}$ and $f(\eta_1(\vec{x})) \leq f(e)$, $f(\eta_1(\vec{y})) \,||\, f(e)$. Therefore we conclude that, as desired

$$\mathcal{E}_2, f(C_1) \models_{f\circ\eta_1} \varphi.$$

Conversely, let $\mathcal{E}_2, f(C_1) \models_{f\circ\eta_1} \varphi$, where $(C_1, \eta_1) \in lp_{\mathcal{E}_1}(\varphi)$. Therefore there exists an event $g \in E_2[f(C_1)]$, such that $g \frown f \circ \eta_1(fv(\psi) \setminus \{z\})$, $\lambda_2(g) = \mathsf{a}$, $f(\eta_1(\vec{x})) \leq g$ and $f(\eta_1(\vec{y})) \,||\, g$ and $\mathcal{E}_2, f(C_1) \models_{\eta_2'} \psi$, where $\eta_2' = (f \circ \eta_1)[z \mapsto g]$.
From the fact that $\mathcal{E}_2, f(C_1) \models_{f\circ\eta_1} \varphi$, by Lemma 3.6, we have that $(f(C_1), f \circ \eta_1) \in lp_{\mathcal{E}_2}(\varphi)$. This means that $f(C_1) \cup f \circ \eta_1(fv(\varphi))$ is consistent and thus $D_2 = f(C_1) \cup \lceil f \circ \eta_1(fv(\varphi))\rceil$ is a configuration. Since $fv(\varphi) = \vec{x} \cup \vec{y} \cup (fv(\psi) \setminus \{z\})$, the arguments above show that

$$D_2 \frown g. \tag{4}$$

Now, since by hypothesis $(C_1, \eta_1) \in lp_{\mathcal{E}_1}(\varphi)$, we know that $C_1 \cup \eta_1(fv(\varphi))$ is consistent. It follows that $D_1 = C_1 \cup \lceil \eta_1(fv(\varphi))\rceil$ is a configuration. Since, by Lemma 4.3, f is injective on consistent sets and preserves causality,

$$\begin{array}{rcl} D_2 & = & f(C_1) \cup \lceil f \circ \eta_1(fv(\varphi))\rceil \\ & = & f(C_1) \cup f(\lceil \eta_1(fv(\varphi))\rceil \\ & = & f(C_1 \cup \lceil \eta_1(fv(\varphi))\rceil) \\ & = & f(D_1) \end{array}$$

which means that $(D_1, f_{|D_1}, D_2) \in R$.
We distinguish two cases. If $g \in D_2$, since $f_{|D_1}$ is an isomorphism of pomsets between D_1 and D_2, we can take the (unique) $e \in D_1$ such that $f(e) = g$. By using the isomorphism property, we have immediately that $e \in E_1[C_1]$, $\eta_1(fv(\psi) \setminus \{z\}) \frown e$, $\lambda_1(e) = \lambda_2(g) = \mathsf{a}$, $\eta_1(\vec{x}) \leq e$ and $\eta_1(\vec{y}) \,||\, e$. Define the environment $\eta_1' = \eta_1[z \mapsto e]$.

Note that $(C_1, \eta_1') \in lp_{\mathcal{E}_1}(\psi)$ since $C_1 \cup \eta_1'(fv(\psi)) \subseteq C_1 \cup \eta_1'(fv(\varphi) \cup \{z\}) \subseteq D_1$. Therefore, since $\mathcal{E}_2, f(C_1) \models_{\eta_2'} \psi$, noticing that $f \circ \eta_1' = \eta_2'$, by inductive hypothesis we conclude $\mathcal{E}_1, C_1 \models_{\eta_1'} \psi$. Hence

$$\mathcal{E}_1, C_1 \models_{\eta_1} \varphi$$

Otherwise, if $g \notin D_2$, recalling (4), if we let $X_2 = \lceil g \rceil \setminus D_2$ we have a pomset transition in $\mathcal{E}_2$:

$$D_2 \xrightarrow{X_2} D_2' \tag{5}$$

Therefore, since R is a hhp-bisimulation, there is a pomset transition in $\mathcal{E}_1$ simulating (5):

$$D_1 \xrightarrow{X_1} D_1' \tag{6}$$

such that $(D_1', f_{|D_1'}, D_2') \in R$. Now, $g \in D_2'$ and thus we can replicate the argument above.

— $\varphi = \langle x \rangle \psi$
Assume that $\mathcal{E}_1, C_1 \models_{\eta_1} \varphi$, where $(C_1, \eta_1) \in lp_{\mathcal{E}_1}(\varphi)$. By definition of the semantics this means that

$$C_1 \xrightarrow{\eta_1(x)} C_1'$$

and $\mathcal{E}_1, C_1' \models_{\eta_1} \psi$.
Since R is a hhp-bisimulation, we have that

$$f(C_1) \xrightarrow{f(\eta_1(x))} f(C_1').$$

Now, since $C_1' = C_1 \cup \{\eta_1(x)\}$ and $fv(\psi) \subseteq fv(\varphi)$, we have that

$$C_1' \cup \eta_1(fv(\psi)) \subseteq C_1 \cup \{\eta_1(x)\} \cup \eta_1(fv(\varphi)) = C_1 \cup \eta_1(fv(\varphi)).$$

Since $(C_1, \eta_1) \in lp_{\mathcal{E}_1}(\psi)$ the set above is consistent and thus $(C_1', \eta_1) \in lp_{\mathcal{E}_1}(\psi)$. Therefore we can use the inductive hypothesis to deduce $\mathcal{E}_2, f(C_1') \models_{f \circ \eta_1} \psi$ and thus, as desired,

$$\mathcal{E}_2, f(C_1) \models_{f \circ \eta_1} \varphi.$$

Conversely, let $\mathcal{E}_2, f(C_1) \models_{f \circ \eta_1} \varphi$, where $(C_1, \eta_1) \in lp_{\mathcal{E}_1}(\varphi)$. By definition of the semantics this means that

$$f(C_1) \xrightarrow{f(\eta_1(x))} C_2'$$

and $\mathcal{E}_2, C_2' \models_{f \circ \eta_1} \psi$.
Since $(C_1, \eta_1) \in lp_{\mathcal{E}_1}(\psi)$, we know that $\eta_1(x)$ is consistent with C_1. Moreover, $C_1 \cup \{\eta_1(x)\}$ is causally closed, otherwise, since f preserves causality and it is injective on consistent sets, also $f(C_1 \cup \eta_1(x)) = C_2 \cup f(\eta_1(x)) = C_2'$ would not be causally closed.
Hence $C_1' = C_1 \cup \{\eta_1(x)\}$ is a configuration and thus

$$C_1 \xrightarrow{\eta_1(x)} C_1'$$

and clearly $f(C_1') = C_2'$. As above we can show that $(C_1', \eta_1) \in lp_{\mathcal{E}_1}(\psi)$ and thus, by inductive hypothesis, $\mathcal{E}_1, C_1' \models_{\eta_1} \psi$. Hence, as desired

$$\mathcal{E}_1, C_1 \models_{\eta_1} \varphi.$$

□

Propositions 4.4 and 4.2 together say that hhp-bisimilarity is the logical equivalence of $\mathcal{L}$.

THEOREM 4.5 (HHP-BISIMILARITY, LOGICALLY). *Let $\mathcal{E}_1$ and $\mathcal{E}_2$ be* PESs. *Then $\mathcal{E}_1 \sim_{hhp} \mathcal{E}_2$ iff $\mathcal{E}_1 \equiv_{\mathcal{L}} \mathcal{E}_2$.*

5. FROM HENNESSY-MILNER LOGIC TO HP-LOGIC

Hhp-bisimilarity is the finest equivalence in the spectrum of true concurrent equivalences proposed in [van Glabbeek and Goltz 2001]. Interestingly enough, coarser equivalences such as step, pomset and hp-bisimilarity, can be captured by suitable fragments of $\mathcal{L}$ summarised in Fig. 3, which can be viewed as the logical counterpart of the true concurrent spectrum.

Note that in each of these fragments after predicating the existence of an event we must execute it. As a consequence, differently from what happens in the full logic, in the fragments it is impossible to refer to events in conflict with already observed events. Intuitively, this means that behavioural equivalences up to hp-bisimilarity can observe events only by executing them. Hence they cannot fully capture the interplay between concurrency and branching, which is indeed distinctive of hhp-bisimilarity.

HM Logic	$\mathcal{L}_{HM}$	$\varphi ::= \langle\!\langle \mathsf{a}\, x \rangle\!\rangle \varphi \mid \varphi \wedge \varphi \mid \neg\varphi \mid \mathsf{T}$
Step Logic	$\mathcal{L}_s$	$\varphi ::= (\langle\!\langle \mathsf{a_1}\, x_1 \rangle\!\rangle \otimes \cdots \otimes \langle\!\langle \mathsf{a_n}\, x_n \rangle\!\rangle)\, \varphi \mid \varphi \wedge \varphi \mid \neg\varphi \mid \mathsf{T}$
Pomset Logic	$\mathcal{L}_p$	$\varphi ::= \langle\!\langle \vec{x}, \overline{\vec{y}} < \mathsf{a}\, z \rangle\!\rangle \varphi \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathsf{T}$ where $\neg$, $\wedge$ are used only on closed formulae.
HP Logic	$\mathcal{L}_{hp}$	$\varphi ::= \langle\!\langle \vec{x}, \overline{\vec{y}} < \mathsf{a}\, z \rangle\!\rangle \varphi \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathsf{T}$

Fig. 3. Fragments of $\mathcal{L}$ corresponding to various behavioural equivalences

5.1. Hennessy-Milner logic

A first simple observation is that standard Hennessy-Milner logic can be recovered as the fragment of $\mathcal{L}$ where only the derived modality $\langle\!\langle \mathsf{a}\, x \rangle\!\rangle \varphi$ (with no references to causally dependent/concurrent events) is allowed. In words, whenever we state the existence of an event we are forced to execute it. Note that, since no dependencies can be expressed, the bound variable x is irrelevant. The induced logical equivalence is thus (interleaving) bisimilarity [Hennessy and Milner 1985] (recall that we consider only image finite PES's).

5.2. Step logic

A fragment $\mathcal{L}_s$ corresponding to step bisimilarity naturally arises as a generalisation of HM logic where we can refer to sets of concurrently enabled events. More precisely, as shown in Fig. 3, $\mathcal{L}_s$ is the fragment of $\mathcal{L}$ where only the derived modality $\langle\!\langle \mathsf{a_1}\, x_1 \rangle\!\rangle \otimes \cdots \otimes \langle\!\langle \mathsf{a_n}\, x_n \rangle\!\rangle$ is used, allowing to predicate on the possibility of performing a parallel step, but without any reference to causal dependencies. Note that all formulae in $\mathcal{L}_s$ are closed, and thus environments (as well as variables) are irrelevant in their semantics.

As an example, consider the two PESs $\mathcal{E}_6$ and $\mathcal{E}_7$ in Fig. 2. They are bisimilar but not step bisimilar since only $\mathcal{E}_7$ can execute the step consisting of a and b in parallel. Accordingly, they are taken apart by the formula $(\langle\!|\mathsf{a}\,|\!\rangle \otimes \langle\!|\mathsf{b}\,|\!\rangle)\mathsf{T}$ in $\mathcal{L}_s$, which is true only on $\mathcal{E}_7$.

LEMMA 5.1. *Let $\mathcal{E}_1$ and $\mathcal{E}_2$ be* PESs *and let $C_i \in \mathcal{C}(\mathcal{E}_i)$, for $i \in \{1,2\}$, be configurations. There exists a step bisimulation R such that $(C_1, C_2) \in R$ iff for any $\varphi \in \mathcal{L}_s$, $\mathcal{E}_1, C_1 \models \varphi \Leftrightarrow \mathcal{E}_2, C_2 \models \varphi$.*

PROOF. ($\Rightarrow$) Assume that $(C_1, C_2) \in R$ for some step bisimulation R. The proof that for all $\varphi \in \mathcal{L}_s$, we have $\mathcal{E}_1, C_1 \models \varphi$ iff $\mathcal{E}_2, C_2 \models \varphi$ can be carried out by induction on the structure of φ.

We only discuss the non-trivial case where $\varphi = (\langle\!|\mathsf{a}_1\, x_1|\!\rangle \otimes \cdots \otimes \langle\!|\mathsf{a}_\mathsf{n}\, x_n|\!\rangle)\ \psi$. Assume that $\mathcal{E}_1, C_1 \models \varphi$. Hence there is a step $C_1 \xrightarrow{\{e_1,\ldots,e_n\}} C_1'$ where $\lambda_1(e_i) = \mathsf{a}_i$ for $i \in \{1,\ldots,n\}$ and

$$\mathcal{E}_1, C_1' \models \psi. \tag{7}$$

Since $(C_1, C_2) \in R$, also C_2 can perform an analogous step

$$C_2 \xrightarrow{\{g_1,\ldots,g_n\}} C_2'$$

with $\lambda_2(g_i) = \mathsf{a}_i$ for $i \in \{1,\ldots,n\}$ and $(C_1', C_2') \in R$. Additionally, by (7) and the induction hypothesis, we have that $\mathcal{E}_2, C_2' \models \psi$. Therefore we conclude $\mathcal{E}_2, C_2 \models \varphi$.

($\Leftarrow$) We prove that the relation

$$R = \{(C_1, C_2) \mid \forall \varphi \in \mathcal{L}_s\ \ (\mathcal{E}_1, C_1 \models \varphi \text{ iff } \mathcal{E}_2, C_2 \models \varphi)\}$$

is a step bisimulation.

We proceed by contradiction. Let $(C_1, C_2) \in R$, let $C_1 \xrightarrow{X} C_1'$ be a step in $\mathcal{E}_1$ and assume that for all Y such that $C_2 \xrightarrow{Y} C_2'$ and $X \sim Y$ as pomsets it does not hold that $(C_1', C_2') \in R$. Hence there exists a formula $\psi \in \mathcal{L}_s$ such that $\mathcal{E}_1, C_1' \models \psi$ and $\mathcal{E}_2, C_2' \not\models \psi$.

Since our PESs are assumed to be image finite, the number of possible steps $C_2 \xrightarrow{Y} C_2'$, with $X \sim Y$ is finite. Let $C_2 \xrightarrow{Y^i} C_2^i$, for $i \in \{1,\ldots,k\}$, be such steps and let ψ^i be the formulae such that $\mathcal{E}_1, C_1' \models \psi^i$ and $\mathcal{E}_2, C_2^i \not\models \psi^i$. If we define

$$\psi = (\langle\!|\mathsf{a}_1\, x_1|\!\rangle \otimes \cdots \otimes \langle\!|\mathsf{a}_\mathsf{n}\, x_n|\!\rangle)\ (\psi^1 \wedge \ldots \wedge \psi^k)$$

we have that $\mathcal{E}_1, C_1 \models \psi$ while $\mathcal{E}_2, C_2 \not\models \psi$. This gives the desired contradiction. □

Now it is immediate to conclude that the following holds.

THEOREM 5.2 (STEP BISIMILARITY, LOGICALLY). *Let $\mathcal{E}_1$ and $\mathcal{E}_2$ be* PESs. *Then $\mathcal{E}_1 \sim_s \mathcal{E}_2$ iff $\mathcal{E}_1 \equiv_{\mathcal{L}_s} \mathcal{E}_2$.*

5.3. Pomset logic

The logic $\mathcal{L}_p$ for pomset bisimilarity in Fig. 3 consists of the fragment of $\mathcal{L}$ where, still an event must be immediately executed when quantified, but it is possible to refer to dependencies between events. However, propositional connectives (negation and conjunction) can be used only on closed formulae.

Roughly speaking, in $\mathcal{L}_p$ closed subformulae characterise the execution of pomsets. The requirement that the propositional operators are used only on closed subformulae prevents pomset transitions from being causally linked to the events in the past. These ideas are formalised by the results below.

First observe that a closed formula in $\mathcal{L}_p$ has always the shape

$$\langle\!\langle \vec{x}_1, \overline{\vec{y}_1} < \mathsf{a}_1\, z_1 \rangle\!\rangle \ldots \langle\!\langle \vec{x}_n, \overline{\vec{y}_n} < \mathsf{a}_n\, z_n \rangle\!\rangle\ \psi$$

where, if we let $Z = \{z_1, \ldots, z_n\}$, then $\vec{x}_i, \vec{y}_i \subseteq Z$ for any $i \in \{1, \ldots, n\}$. We next prove that the prefix $\langle\!\langle \vec{x}_1, \overline{\vec{y}_1} < \mathsf{a}_1\, z_1 \rangle\!\rangle \ldots \langle\!\langle \vec{x}_n, \overline{\vec{y}_n} < \mathsf{a}_n\, z_n \rangle\!\rangle$ intuitively corresponds to the execution of a class of pomsets (not a single one, since the relation between some events might be unspecified). More precisely, in the situation above let $Pom(\langle\!\langle \vec{x}_1, \overline{\vec{y}_1} < \mathsf{a}_1\, z_1 \rangle\!\rangle \ldots \langle\!\langle \vec{x}_n, \overline{\vec{y}_n} < \mathsf{a}_n\, z_n \rangle\!\rangle)$ denote the class of pomsets $(Z, \leq, \lambda)$ such that $Z = \{z_1, \ldots, z_n\}$ and for $i \in \{1, \ldots, n\}$, $\lambda(z_1) = \mathsf{a}_i$ and given any $z \in Z$

— $z \in \vec{x}_i$ implies $z \leq z_i$,
— $z \in \vec{y}_i$ implies $z \not\leq z_i$.

With this definition it is immediate to show that the following result holds.

LEMMA 5.3. *Let $\varphi = \langle\!\langle \vec{x}_1, \overline{\vec{y}_1} < \mathsf{a}_1\, z_1 \rangle\!\rangle \ldots \langle\!\langle \vec{x}_n, \overline{\vec{y}_n} < \mathsf{a}_n\, z_n \rangle\!\rangle\ \psi$ be a closed formula in $\mathcal{L}_p$. Then*

$$\begin{array}{lll} \mathcal{E}, C \models_\eta \varphi & \textit{iff} & C \xrightarrow{X} C' \textit{ where } X = \{e_1, \ldots, e_n\} \textit{ is a pomset s.t. } X \sim (Z, \leq, \lambda) \\ & & \textit{for some } (Z, \leq, \lambda) \in Pom(\langle\!\langle \vec{x}_1, \overline{\vec{y}_1} < \mathsf{a}_1\, z_1 \rangle\!\rangle \ldots \langle\!\langle \vec{x}_n, \overline{\vec{y}_n} < \mathsf{a}_n\, z_n \rangle\!\rangle) \\ & & \textit{and } \mathcal{E}, C' \models_{\eta'} \psi, \textit{ with } \eta' = \eta[z_1 \mapsto e_1, \ldots, z_n \mapsto e_n]. \end{array}$$

PROOF. By induction on n. □

Next we observe that, in particular, the execution of a single pomset can be exactly characterised by a corresponding formula in $\mathcal{L}_p$.

Definition 5.4 (*pomsets as formulae in $\mathcal{L}_p$*). Let $Z = \{z_1, \ldots, z_n\}$ be a set of variables and let $p_Z = (Z, \leq_{p_Z}, \lambda_{p_Z})$ be a pomset. Given a formula $\varphi \in \mathcal{L}_p$, we denote by $\langle\!|\, p_Z \,|\!\rangle \varphi$ the formula inductively defined as follows. If Z is empty then $\langle\!|\, p_Z \,|\!\rangle \varphi = \varphi$. If $Z = Z' \cup \{z\}$, where z is maximal with respect to $\leq_{p_Z}$ (if there are many maximal z_i, choose the one with highest index), let $\vec{x} = \{z' \in Z' \mid z' \leq_{p_z} z\}$, $\vec{y} = Z' \setminus \vec{x}$, and $\mathsf{a} = \lambda_{p_Z}(z)$, then $\langle\!|\, p_Z \,|\!\rangle \varphi = \langle\!|\, p_{Z'} \,|\!\rangle \langle\!\langle \vec{x}, \overline{\vec{y}} < \mathsf{a}\, z \rangle\!\rangle \varphi$.

Note that if φ is a closed formula also $\langle\!|\, p_Z \,|\!\rangle \varphi$ is closed.

The fact that pomset formulae as defined above have exactly the intended semantics immediately follows from Lemma 5.3.

LEMMA 5.5 (POMSETS IN $\mathcal{L}_p$). *Let $\mathcal{E}$ be a* PES *and let $C \in \mathcal{C}(\mathcal{E})$ be a configuration. Given $\{z_1, \ldots, z_n\} \subseteq Var$ and a pomset $p_Z = (Z, \leq_{p_Z}, \lambda_{p_Z})$, then*

$$\begin{array}{lll} \mathcal{E}, C \models_\eta \langle\!|\, p_Z \,|\!\rangle \varphi & \textit{iff} & C \xrightarrow{X} C' \textit{ where } X = \{e_1, \ldots, e_n\} \textit{ is a pomset s.t. } X \sim p_Z \\ & & \textit{and } \mathcal{E}, C' \models_{\eta'} \varphi, \textit{ with } \eta' = \eta[z_1 \mapsto e_1, \ldots, z_n \mapsto e_n] \end{array}$$

PROOF. Just observe that $Pom(\langle\!|\, p_Z \,|\!\rangle) = \{p_Z\}$. Then the result is an instance of Lemma 5.3. □

LEMMA 5.6. *Let $\mathcal{E}_1$ and $\mathcal{E}_2$ be* PES*s and let $C_i \in \mathcal{C}(\mathcal{E}_i)$, for $i \in \{1, 2\}$, be configurations. There exists a pomset bisimulation R such that $(C_1, C_2) \in R$ iff for any $\varphi \in \mathcal{L}_p$, φ closed formula, $\mathcal{E}_1, C_1 \models \varphi \Leftrightarrow \mathcal{E}_2, C_2 \models \varphi$.*

PROOF. ($\Rightarrow$) Let R be a pomset bisimulation. We prove that if $(C_1, C_2) \in R$, then for all closed formulae $\varphi \in \mathcal{L}_p$, we have that $\mathcal{E}_1, C_1 \models \varphi$ iff $\mathcal{E}_2, C_2 \models \varphi$.

The proof proceeds by induction on the structure of the formula φ. The cases in which φ is a conjunction, negation or true are trivial. In the remaining cases φ is a

closed formula of the shape

$$\langle\!\langle \vec{x}_1, \overrightarrow{\vec{y}_1} < \mathsf{a}_1\, z_1 \rangle\!\rangle \ldots \langle\!\langle \vec{x}_n, \overrightarrow{\vec{y}_n} < \mathsf{a}_n\, z_n \rangle\!\rangle\ \psi. \tag{8}$$

where ψ is closed.

Assume that $\mathcal{E}_1, C_1 \models \varphi$, i.e., $\mathcal{E}_1, C_1 \models_{\eta_1} \varphi$ for some (irrelevant) η. Then, by Lemma 5.3, $C_1 \xrightarrow{X} C_1'$ where $X \sim (Z, \leq, \lambda)$ for some pomset $(Z, \leq, \lambda) \in Pom(\langle\!\langle \vec{x}_1, \overrightarrow{\vec{y}_1} < \mathsf{a}_1\, z_1 \rangle\!\rangle \ldots \langle\!\langle \vec{x}_n, \overrightarrow{\vec{y}_n} < \mathsf{a}_n\, z_n \rangle\!\rangle)$. Additionally $\mathcal{E}_1, C_1' \models_{\eta_1[z_1 \mapsto e_1, \ldots, z_n \mapsto e_n]} \psi$, which can be written $\mathcal{E}_1, C_1' \models \psi$, as ψ is closed.

Since $(C_1, C_2) \in R$ and R is a pomset bisimulation, there is a pomset $Y = \{g_1, \ldots, g_n\}$, isomorphic to X, and thus to $(Z, \leq, \lambda)$, such that

$$C_2 \xrightarrow{Y} C_2' \tag{9}$$

and $(C_1', C_2') \in R$. By inductive hypothesis, $\mathcal{E}_2, C_2' \models \psi$. Again, since ψ is closed, by Lemma 3.7 it also holds $\mathcal{E}_2, C_2' \models_{\eta_2[z_1 \mapsto g_1, \ldots, z_n \mapsto g_n]} \psi$, for any chosen η_2. This fact, together with (9), allows us to conclude, by Lemma 5.3, that $\mathcal{E}_2, C_2 \models_{\eta_2} \varphi$, i.e., since φ is closed, $\mathcal{E}_2, C_2 \models \varphi$ as desired.

($\Leftarrow$) The proof is similar to that of Lemma 5.1, i.e., we show that the relation

$$R = \{(C_1, C_2) \mid \forall \varphi \in \mathcal{L}_p, \varphi \text{ closed},\ \mathcal{E}_1, C_1 \models \varphi \text{ iff } \mathcal{E}_2, C_2 \models \varphi\}$$

is a pomset bisimulation.

We proceed by contradiction. Let $(C_1, C_2) \in R$, let $C_1 \xrightarrow{X} C_1'$, where X is a pomset, and assume that for all Y such that $C_2 \xrightarrow{Y} C_2'$ and $X \sim Y$ there exists a closed formula $\psi \in \mathcal{L}_p$ such that $\mathcal{E}_1, C_1' \models \psi$ and $\mathcal{E}_2, C_2' \not\models \psi$.

Since our PESs are assumed to be image finite, there are finitely many such pomset transitions $C_2 \xrightarrow{Y^i} C_2^i$, for $i \in \{1, \ldots, k\}$. Let ψ^i be the formulae such that $\mathcal{E}_1, C_1' \models \psi^i$ and $\mathcal{E}_2, C_2^i \not\models \psi^i$ for $i \in \{1, \ldots, k\}$. If p_Z is a pomset of variables, such that $p_Z \sim X$, let us define a formula in $\mathcal{L}_s$ as follows:

$$\psi = \langle\!\langle\, p_Z \rangle\!\rangle\ (\psi^1 \wedge \ldots \wedge \psi^k)$$

Then by Lemma 5.5, we have that $\mathcal{E}_1, C_1 \models \psi$ while $\mathcal{E}_2, C_2 \not\models \psi$. This gives the desired contradiction. □

The logical characterisation of pomset bisimilarity now immediately follows.

THEOREM 5.7 (POMSET BISIMILARITY, LOGICALLY). *Let $\mathcal{E}_1$ and $\mathcal{E}_2$ be* PESs. *Then $\mathcal{E}_1 \sim_p \mathcal{E}_2$ iff $\mathcal{E}_1 \equiv_{\mathcal{L}_p} \mathcal{E}_2$.*

As an example, consider the two PESs $\mathcal{E}_7$ and $\mathcal{E}_9$ in Fig. 2. They are step bisimilar but not pomset bisimilar since only the second one can execute the pomset $p_{a<b} = (\{\mathsf{a}, \mathsf{b}\}, \mathsf{a} < \mathsf{b}, \lambda)$, where λ is the obvious labelling. Accordingly, the formula $\varphi = \langle\!\langle\, p_{a<b} \rangle\!\rangle \mathsf{T} = \langle\!\langle \mathsf{a}\, x \rangle\!\rangle \langle\!\langle x < \mathsf{b}\, y \rangle\!\rangle \mathsf{T}$ in $\mathcal{L}_p$, is satisfied only by $\mathcal{E}_9$.

5.4. History preserving logic

The fragment $\mathcal{L}_{hp}$ corresponding to hp-bisimilarity is obtained from that for pomset bisimilarity by relaxing the condition asking that propositional connectives are applied only to closed formulae. Intuitively, in this way a formula $\varphi \in \mathcal{L}_{hp}$, besides expressing the possibility of executing a pomset p, also predicates about dependencies of events in the pomset with previously executed events (bound to the free variables of φ).

The following two PESs can be proved to be pomset equivalent but not hp-equivalent:

Intuitively, they allow the same pomset transitions, but they have a different "causal branching". Indeed, only in the left-most PESs, after the execution of an a-labelled event we can choose between a concurrent and a causally dependent b-labelled event. In the rightmost PES the choice is already determined by the execution of a. Formally, the formula $\langle\!|\mathsf{a}\,x|\!\rangle(\langle\!|\overline{x} < \mathsf{b}\,y|\!\rangle\mathsf{T} \wedge \langle\!|x < \mathsf{b}\,z|\!\rangle\mathsf{T})$ in $\mathcal{L}_{hp}$ is true only on the left-most PES.

We start with a simple lemma that makes explicit the semantics of the induced operator $\langle\!|\vec{x}, \overline{\vec{y}} < \mathsf{a}\,z|\!\rangle$.

LEMMA 5.8 (EVENTS WITH THEIR HISTORY IN THE LOGIC). *Given a* PES $\mathcal{E}$*, a formula* $\varphi \in \mathcal{L}_{hp}$ *and a legal pair* $(C, \eta) \in lp(\langle\!|\vec{x}, \overline{\vec{y}} < \mathsf{a}\,z|\!\rangle\,\varphi)$*:*

$$\mathcal{E}, C \models_\eta \langle\!|\vec{x}, \overline{\vec{y}} < \mathsf{a}\,z|\!\rangle\,\varphi \quad \textit{iff} \quad \begin{array}{l}\textit{there is an event } e \in E \textit{ such that } C \xrightarrow{\ e\ } C',\ \lambda(e) = \mathsf{a},\\ \eta(\vec{x}) \leq e,\ \eta(\vec{y})\,||\,e \textit{ and } C' \models_{\eta'} \varphi, \textit{ where } \eta' = \eta[z \mapsto e].\end{array}$$

PROOF. The result follows almost immediately from the definition of the semantics (Definition 3.4). □

LEMMA 5.9. *Let* $\mathcal{E}_1$ *and* $\mathcal{E}_2$ *be* PESs *and let* $(C_1, f, C_2) \in \mathcal{C}(\mathcal{E}_1)\bar{\times}\mathcal{C}(\mathcal{E}_2)$*, i.e.,* $C_i \in \mathcal{C}(\mathcal{E}_i)$*, for* $i \in \{1, 2\}$*, are configurations and* $f : C_1 \to C_2$ *is an isomorphism of pomsets. Then the following are equivalent:*

(*1*) *there is a hp-bisimulation* R *such that* $(C_1, f, C_2) \in R$*;*
(*2*) *for any* $\varphi \in \mathcal{L}_{hp}$ *and* $\eta_1 \in Env_{\mathcal{E}_1}$ *such that* $\eta_1(fv(\varphi)) \subseteq C_1$*, it holds that* $\mathcal{E}_1, C_1 \models_{\eta_1} \varphi \Leftrightarrow \mathcal{E}_2, C_2 \models_{f\circ\eta_1} \varphi$.

PROOF. (1 ⇒ 2) Let R be a hp-bisimulation. We show that for all formulae $\varphi \in \mathcal{L}_{hp}$, triples $(C_1, f, C_2) \in R$ and environments $\eta_1 \in Env_{\mathcal{E}_1}$ such that $\eta_1(fv(\varphi)) \subseteq C_1$ it holds

$$\mathcal{E}_1, C_1 \models_{\eta_1} \varphi \ \text{ iff } \ \mathcal{E}_2, C_2 \models_{f\circ\eta_1} \varphi.$$

We proceed by induction on the structure of the formula φ. We focus on the only non-trivial case where $\varphi = \langle\!|\vec{x}, \overline{\vec{y}} < \mathsf{a}\,z|\!\rangle\,\psi$. If $\mathcal{E}_1, C_1 \models_{\eta_1} \varphi$, then by Lemma 5.8 there is an event $e \in E_1$ such that

$$C_1 \xrightarrow{\ e\ } C_1' \tag{10}$$

with $\lambda_1(e) = \mathsf{a}$, $\eta_1(\vec{x}) \leq e$, $\eta_1(\vec{y})\,||\,e$ and $\mathcal{E}_1, C_1' \models_{\eta_1'} \psi$ where $\eta_1' = \eta_1[z \mapsto e]$.

Since $(C_1, f, C_2) \in R$, there exists an event $g \in E_2$ such that

$$C_2 \xrightarrow{\ g\ } C_2' \tag{11}$$

and $(C_1', f', C_2') \in R$, with $f' = f[e \mapsto g]$. Since f' is an isomorphism of configurations, we have that $\lambda_2(g) = \mathsf{a}$, $f(\eta_1(\vec{x})) \leq g$ and $f(\eta_1(\vec{y}))\,||\,g$.

Note that $\eta_1'(fv(\psi)) \subseteq \eta_1'(fv(\varphi) \cup \{z\}) = \eta_1(fv(\varphi)) \cup \{e\} \subseteq C_1 \cup \{e\} = C_1'$. Thus, we can use the induction hypothesis to deduce that $\mathcal{E}_2, C_2' \models_{f'\circ\eta_1'} \psi$. Therefore, by using again Lemma 5.8, we can conclude $\mathcal{E}_2, C_2 \models_{f\circ\eta_1} \varphi$.

The proof that $\mathcal{E}_2, C_2 \models_{f\circ\eta_1} \varphi$ implies $\mathcal{E}_1, C_1 \models_{\eta_1} \varphi$ is analogous and thus omitted.

(1 ⇐ 2) As in Proposition 4.2 we fix a surjective environment $\eta_1 : Var \to E_1$. Moreover, given an event $e \in E_1$, we write x_e to denote a fixed distinguished variable such that $\eta_1(x_e) = e$. Similarly, for a configuration $C_1 = \{e_1, \ldots, e_n\}$ we denote by X_{C_1} the set of

variables $\{x_{e_1}, \ldots, x_{e_n}\}$. Observe that (C_1, η_1) is a legal pair for any formula $\varphi \in \mathcal{L}$ such that $fv(\varphi) \subseteq X_{C_1}$.

Then we show that the posetal relation $R \subseteq \mathcal{C}(\mathcal{E}_1)\bar{\times}\mathcal{C}(\mathcal{E}_2)$ defined by

$$R = \{(C_1, f, C_2) \mid \forall \varphi \in \mathcal{L}_{hp}.\, fv(\varphi) \subseteq X_{C_1} \quad \mathcal{E}_1, C_1 \models_{\eta_1} \varphi \text{ iff } \mathcal{E}_2, C_2 \models_{f\circ\eta_1} \varphi\}$$

is a hp-bisimulation. Note that as in Proposition 4.2, with a slight abuse of notation, we denote by $f \circ \eta_1$ any environment η_2 such that $\eta_2(x) = f(\eta_1(x))$ for $x \in X_{C_1}$ and $\eta_2(x)$ has any value, otherwise. By Lemma 3.7, this arbitrariness has no impact on the satisfaction of φ in the definition of R since $fv(\varphi) \subseteq X_{C_1}$.

We proceed by contradiction. Assume that $(C_1, f, C_2) \in R$, let $C_1 \xrightarrow{e} C_1'$ and suppose that for all $g \in E_2$ such that $C_2 \xrightarrow{g} C_2'$ with $C_1' \sim C_2'$ as pomsets, we have $(C_1', f[e \mapsto g], C_2') \notin R$, i.e., there exists a formula ψ, with $fv(\psi) \subseteq X_{C_1'}$, such that $\mathcal{E}_1, C_1' \models_{\eta_1} \psi$ and $\mathcal{E}_2, C_2' \not\models_{f'\circ\eta_1} \psi$.

Since all PESs are assumed to be image finite, there are finitely many transitions

$$C_2 \xrightarrow{g^i} C_2^i, \quad i \in \{1, \ldots, k\}$$

such that $f^i = f[e \mapsto g^i] : C_1' \to C_2^i$ is an isomorphism of pomsets. Let ψ^i, for $i \in \{1, \ldots, k\}$ be formulae such that

$$\mathcal{E}_1, C_1' \models_{\eta_1} \psi^i \quad \text{and} \quad \mathcal{E}_2, C_2^i \not\models_{f^i\circ\eta_1} \psi^i$$

where $fv(\psi^i) \subseteq X_{C_1'} = X_{C_1} \cup \{x_e\}$. Now consider the formula

$$\varphi = \langle\!|\vec{x}, \overline{\vec{y}} < \mathsf{a}\, x_e|\!\rangle(\psi^1 \wedge \ldots \wedge \psi^k)$$

where $\mathsf{a} = \lambda_1(e)$ and the $\vec{x}, \vec{y} \subseteq X_{C_1}$ are such that $\eta_1(\vec{x})$ is the set of causes of e in C_1 and $\eta_1(\vec{y})$ is the set of events in C_1 which are concurrent with e. Note that $fv(\varphi) = \vec{x} \cup \vec{y} \cup ((\bigcup_{i=1}^k fv(\psi_i)) \setminus \{x_e\}) = X_{C_1}$.

Then by Lemma 5.8 we have that $\mathcal{E}_1, C_1 \models_{\eta_1} \varphi$ and $\mathcal{E}_2, C_2 \not\models_{f\circ\eta_1} \varphi$, which gives the desired contradiction.

The fact that R as defined above is a hp-bisimulation allows us to conclude. In fact, assume that $(C_1, f, C_2) \in \mathcal{C}(\mathcal{E}_1)\bar{\times}\mathcal{C}(\mathcal{E}_2)$ and (2) holds. Then for any $\varphi \in \mathcal{L}_{hp}$ such that $fv(\varphi) \subseteq X_{C_1}$, it holds that $\eta_1(fv(\varphi)) \subseteq \eta_1(X_{C_1}) = C_1$. Therefore we can use (2) and deduce that $\mathcal{E}_1, C_1 \models_{\eta_1} \varphi$ iff $\mathcal{E}_2, C_2 \models_{f\circ\eta_1} \varphi$. This implies that $(C_1, f, C_2) \in R$, i.e., we get (1). □

Remark. It is worth observing that the hp-bisimulation built in the previous proof relates two configurations C_1 and C_2 when they satisfy the same formulae, whereas the hhp-bisimulation built in the proof of Proposition 4.2 (which leads to Theorem 4.5) relates C_1 and C_2 when the same formulae are satisfied by the empty configuration (in an environment that binds free variables to C_1, resp. C_2). Intuitively, this corresponds to the fact that for hp-bisimilarity one has to check only the future of a configuration, while for hhp-bisimilarity also alternative evolutions (hence evolutions from the past) of a configuration must be considered.

THEOREM 5.10 (HP-BISIMILARITY, LOGICALLY). *Let $\mathcal{E}_1$ and $\mathcal{E}_2$ be* PESs. *Then $\mathcal{E}_1 \sim_{hp} \mathcal{E}_2$ iff $\mathcal{E}_1 \equiv_{\mathcal{L}_{hp}} \mathcal{E}_2$.*

PROOF. ($\Rightarrow$) Let $\mathcal{E}_1 \sim_{hp} \mathcal{E}_2$. Then there is a hp-bisimulation R such that $(\emptyset, \emptyset, \emptyset) \in R$. For all $\varphi \in \mathcal{L}_{hp}$, if φ is closed, i.e., $fv(\varphi) = \emptyset$, as an instance of Lemma 5.9, we obtain $\mathcal{E}_1, \emptyset \models_{\eta_1} \varphi$ iff $\mathcal{E}_2, \emptyset \models_{f\circ\eta} \varphi$, for any $\eta_1 \in Env_{\mathcal{E}_1}$. This amounts to $\mathcal{E}_1 \models \varphi$ iff $\mathcal{E}_2 \models \varphi$, i.e., $\mathcal{E}_1 \equiv_{\mathcal{L}_{hp}} \mathcal{E}_2$, as desired.

($\Leftarrow$) Let $\mathcal{E}_1 \equiv_{\mathcal{L}_{hp}} \mathcal{E}_2$. Then, for any closed formula $\varphi \in \mathcal{L}_{hp}$, it holds that $\mathcal{E}_1 \models \varphi$ iff $\mathcal{E}_2 \models \varphi$. Since φ is closed, satisfaction does not depend on the environment, hence $\mathcal{E}_1, \emptyset \models_{\eta_1} \varphi$ iff $\mathcal{E}_2, \emptyset \models_{\eta_2} \varphi$ for any $\eta_1 \in Env_{\mathcal{E}_1}$, $\eta_2 \in Env_{\mathcal{E}_2}$. In particular, we can consider $\emptyset : \emptyset \to \emptyset$, isomorphism between empty configurations and we have $\mathcal{E}_1, \emptyset \models_{\eta_1} \varphi$ iff $\mathcal{E}_2, \emptyset \models_{\emptyset \circ \eta_1} \varphi$ for any $\eta_1 \in Env_{\mathcal{E}_1}$. Therefore, we can apply Lemma 5.9 to conclude that there exists a hp-bisimulation R such that $(\emptyset, \emptyset, \emptyset) \in R$ and thus $\mathcal{E}_1 \sim_{hp} \mathcal{E}_2$. $\square$

6. A LOGIC WITH RECURSION

The logic $\mathcal{L}$ discussed in the previous section is theoretically interesting as it allows one to logically characterise the main true concurrent equivalences. However, as a specification language, it has a limited expressiveness: even if one can "observe" events arbitrarily far in the future, a single formula in $\mathcal{L}$ only describes properties where a finite number of events are executed. In order to overcome this limitation, in this section we study a fixpoint extension of the logic, where the use of recursion allows one to express causal and concurrency properties of infinite computations. The resulting logic, denoted $\mu\mathcal{L}$, is a kind of first-order μ-calculus similar to the μ-calculi in [Dam 1996; Dam et al. 1998] and [Groote and Willemse 2005], where first order variables are used to represent channels or data. Similarities exist also with the fixpoint extension of independence-friendly modal logic studied in [Bradfield and Kreutzer 2005]. In fact, in all of these papers fixpoints are added to a core logic which includes quantified first order variables. The solutions adopted to let the fixpoint operators and variables interact with first order variables are similar to that in our logic.

Let $\mathcal{X}^a$ be a set of *abstract propositions*, ranged over by $X, Y, \ldots$, that are intended to represent formulae possibly containing (unnamed) free event variables. Each abstract proposition has an arity $ar(X)$, which indicates the number of free event variables in X. An abstract proposition X can be turn into a formula by specifying a name for its free variables. For $\vec{x}$ such that $|\vec{x}| = ar(X)$, we write $X(\vec{x})$ to indicate the abstract proposition X whose free event variables are named $\vec{x}$. We call $X(\vec{x})$ a *proposition* and denote by $\mathcal{X}$ the set of all propositions.

Definition 6.1 (*syntax*). Let Var be a denumerable set of event variables and let $\mathcal{X}$ be a set of propositions, as explained above. The syntax of $\mu\mathcal{L}$ over the set of labels Λ is defined as follows:

$$\varphi \ ::= \ X(\vec{x}) \ \mid \ \mathsf{T} \ \mid \ \varphi \wedge \varphi \ \mid \ \neg\varphi \ \mid \ (\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi \ \mid \ \langle z \rangle\, \varphi \mid \ \mu X(\vec{x}).\varphi$$

where for formula $\mu X(\vec{x}).\varphi$, as usual, X must occur positively in φ and additionally, $fv(\varphi) = \vec{x}$.

The requirement that X occurs positively in the formula $\mu X(\vec{x}).\varphi$ is a standard one, later used in the definition of the semantics for ensuring the existence of the fixpoint.

Definition 6.2 (*free variables*). The *free variables* of a formula φ in $\mu\mathcal{L}$ are given as in Definition 3.2, with the addition of the following clauses:

$$fv(X(\vec{x})) = \vec{x} \qquad \text{and} \qquad fv(\mu X(\vec{x}).\varphi) = \vec{x}.$$

In the sequel we will often use the set of free variables of a formula as a tuple. Thus it is convenient to assume that $fv(\cdot)$ returns a fixed tuple of variables. Note that the fact that variables $\vec{x}$ are free in $X(\vec{x})$ and in $\mu X(\vec{x}).\varphi$ is reflected in the definition of free variable substitution. For instance $X(\vec{x})[\vec{y}/\vec{x}] = X(\vec{y})$ and $(\mu X(x).\varphi)[y/x] = \mu X(y).(\varphi[y/x])$.

A least fixpoint operator μ has been added. In a recursive formula $\mu X(\vec{x}).\varphi$ the abstract proposition X can occur in φ, possibly with a different tuple of variables which, intuitively, are used in the next iteration.

As usual a greatest fixpoint operator can be encoded, by duality, as

$$\nu X(\vec{x}).\varphi = \neg(\mu X(\vec{x}).\neg\tilde{\varphi})$$

where $\tilde{\varphi}$ is the formula obtained replacing any occurrence of X in φ with $\neg X$ (in order to keep the positivity of the occurrences of X).

As an example, the existence of a run consisting of an infinite causal chain of a-actions can be expressed by the following formula:

$$\langle\!|\mathsf{a}\, x|\!\rangle\ (\nu X(x).\langle\!|x < \mathsf{a}\, y|\!\rangle X(y))$$

The infinite causal chain is obtained by "passing" the event bound to y by the current execution to the next iteration so that it can be used as a cause in the corresponding execution. The execution outside the recursive formula binds x to an a-labelled event which will be the first in the causal chain.

In a fixpoint formula $\mu X(\vec{x}).\varphi$, the fixpoint operator binds all the free occurrences of the abstract proposition X in φ. This leads to the following notion of free abstract proposition.

Definition 6.3 (*free propositions, substitution*). The set of *free propositions* in a formula φ in $\mu\mathcal{L}$, denoted $fp(\varphi)$, is defined inductively by

$$\begin{array}{ll} fp(\mathsf{T}) = \emptyset & fp(X(\vec{x})) = \{X\} \\ fp(\varphi_1 \wedge \varphi_2) = fp(\varphi_1) \cup fp(\varphi_2) & \\ fp(\neg\varphi) = fp((\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi) = fp(\langle z\rangle\, \varphi) = fp(\varphi) & \\ fp(\mu X(\vec{x}).\varphi) = fp(\varphi) \setminus \{X\} & \end{array}$$

Let φ be a formula in $\mu\mathcal{L}$. For an abstract proposition X and formula ψ such that $fv(\psi) = \vec{x}$, $|\vec{x}| = ar(X)$, we denote by $\varphi[\psi/X]$ the formula obtained from φ by replacing any free occurrence of $X(\vec{y})$ by $\psi[\vec{y}/\vec{x}]$.

A formula $\varphi \in \mu\mathcal{L}$ is called *closed* when both $fv(\varphi)$ and $fp(\varphi)$ are empty.

Let us now move to the definition of the semantics. Legal pairs for a formula are defined exactly as in Definition 3.3. For instance the pair (C, η) is legal for the formula $X(\vec{x})$ if the set $C \cup \eta(\vec{x})$ is consistent. On the other hand, in addition to the (event variable) environment, the semantics of $\mu\mathcal{L}$ also requires an interpretation for the propositions, mapping each proposition $X(\vec{x})$ to a set of legal pairs for it.

Definition 6.4 (*proposition environments*). Let $\mathcal{E}$ be a PES. A *proposition environment* is a function $\pi : \mathcal{X} \to 2^{\mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}}}$ such that:

(1) $\pi(X(\vec{x})) \subseteq lp(X(\vec{x}))$ for any $X(\vec{x}) \in \mathcal{X}$, and
(2) if $(C, \eta) \in \pi(X(\vec{x}))$ and $\eta'(\vec{y}) = \eta(\vec{x})$ pointwise, then $(C, \eta') \in \pi(X(\vec{y}))$.

We denote by $PEnv_{\mathcal{E}}$ the set of proposition environments, ranged over by π.

The first condition requires that the denotation for $X(\vec{x})$ only consists of legal pairs for $X(\vec{x})$. The second condition requires that the semantics of a proposition only depends on the events that the environment associates to its free variables and that it does not depend on the naming of the variables. Such a condition allows us to generalise Lemma 3.7 to the logic with recursion.

Updates of a proposition environment must be properly defined in order to maintain the validity of properties 1 and 2 above. For $\pi \in PEnv_{\mathcal{E}}$ and $S \subseteq lp(X(\vec{x}))$, we write $\pi[X(\vec{x}) \mapsto S]$ for the proposition environment defined by

$$\begin{array}{lcl} \pi[X(\vec{x}) \mapsto S](X(\vec{y})) & = & \{(C, \eta') \mid (C, \eta) \in S \ \wedge\ \eta'(\vec{y}) = \eta(\vec{x})\} \\ \pi[X(\vec{x}) \mapsto S](Y(\vec{y})) & = & \pi(Y(\vec{y})) \quad \text{for } Y \neq X. \end{array}$$

LEMMA 6.5. *Let $\mathcal{E}$ be a* PES*, π a proposition environments, $\varphi \in \mu\mathcal{L}$ be a formula and let $\vec{x} = fv(\varphi)$ be the tuple of free variables in φ.*

(1) If $(C,\eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}$ and $\eta'(\vec{y}) = \eta(\vec{x})$ pointwise, then $(C,\eta') \in \{\!|\varphi[\vec{y}/\vec{x}]|\!\}^{\mathcal{E}}_{\pi}$.
(2) For any formula ψ and abstract proposition X such that $ar(X) = |fv(\varphi)|$ it holds $\{\!|\psi[\varphi/X]|\!\}^{\mathcal{E}}_{\pi} = \{\!|\psi|\!\}^{\mathcal{E}}_{\pi[X(\vec{x})\mapsto\{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}]}$.

PROOF. Both items can be proved by a routine induction (on φ for 1 and on ψ for 2). □

In particular, from 1 above it follows that, as already proved for logic $\mathcal{L}$ in Lemma 3.7, the semantics of a formula φ in $\mu\mathcal{L}$ only depends on the events that the environment associates to the free variables $\vec{x}$ of the formula, i.e., if $C \in \mathcal{C}(\mathcal{E})$ and η, η' are environments such that $\eta_{|\vec{x}} = \eta'_{|\vec{x}}$ then $(C,\eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}$ iff $(C,\eta') \in \{\!|\varphi|\!\}^{\mathcal{E}}$.

Definition 6.6 (*semantics*). Let $\mathcal{E}$ be a PES. The denotation of a formula is given by the function

$$\{\!|\cdot|\!\}^{\mathcal{E}} : \mu\mathcal{L} \to PEnv_{\mathcal{E}} \to 2^{\mathcal{C}(\mathcal{E})\times Env_{\mathcal{E}}}$$

defined inductively as follows, where we write $\{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}$ instead of $\{\!|\varphi|\!\}^{\mathcal{E}}(\pi)$:

$$\begin{aligned}
\{\!|\mathsf{T}|\!\}^{\mathcal{E}}_{\pi} &= \mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}} \\
\{\!|\varphi_1 \wedge \varphi_2|\!\}^{\mathcal{E}}_{\pi} &= \{\!|\varphi_1|\!\}^{\mathcal{E}}_{\pi} \cap \{\!|\varphi_2|\!\}^{\mathcal{E}}_{\pi} \cap lp(\varphi_1 \wedge \varphi_2) \\
\{\!|\neg\varphi|\!\}^{\mathcal{E}}_{\pi} &= lp(\varphi) \setminus \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi} \\
\{\!|(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi|\!\}^{\mathcal{E}}_{\pi} &= \{(C,\eta) \mid\ (C,\eta) \in lp((\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi) \text{ and} \\
&\qquad \exists e \in E[C] \text{ such that } e \frown \eta(fv(\varphi) \setminus \{z\}) \\
&\qquad \wedge\ \lambda(e) = \mathsf{a} \ \wedge\ \eta(\vec{x}) < e \ \wedge\ \eta(\vec{y}) \,||\, e \\
&\qquad \wedge\ (C, \eta[z \mapsto e]) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi} \qquad\qquad \} \\
\{\!|\langle z\rangle\ \varphi|\!\}^{\mathcal{E}}_{\pi} &= \{(C,\eta) \mid\ C \xrightarrow{\eta(z)} C' \ \wedge\ (C',\eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}\ \} \\
\{\!|X(\vec{x})|\!\}^{\mathcal{E}}_{\pi} &= \pi(X(\vec{x})) \\
\{\!|\mu X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi} &= lfp(f)
\end{aligned}$$

where $lfp(f)$ is the least fixed point of the function $f : 2^{lp(X(\vec{x}))} \to 2^{lp(X(\vec{x}))}$ that maps $S \subseteq lp(X(\vec{x}))$ into

$$f(S) = \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi[X(\vec{x})\mapsto S]}$$

When $(C,\eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}$ we say that the PES $\mathcal{E}$ satisfies the formula φ in the configuration C and environments η, π and write $\mathcal{E}, C \models_{\eta,\pi} \varphi$. For closed formulae φ, we write $\mathcal{E}, C \models \varphi$, when $\mathcal{E}, C \models_{\eta,\pi} \varphi$ for some η, π and $\mathcal{E} \models \varphi$ when $\mathcal{E}, \emptyset \models \varphi$.

It can be easily proved that Lemma 3.6 extends to $\mu\mathcal{L}$, i.e., for any formula $\varphi \in \mu\mathcal{L}$, its denotation only contains legal pairs, that is $\{\!|\varphi|\!\}^{\mathcal{E}}_{\pi} \subseteq lp_{\mathcal{E}}(\varphi)$. Note also that the semantics of recursive formulae is well-defined. In fact, $\pi[X(\vec{x}) \mapsto S]$ is a well-defined proposition environment, since $S \subseteq lp(X(\vec{x}))$. Moreover $f(S) = \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi[X(\vec{x})\mapsto S]} \subseteq lp(\varphi)$ by the previous observation, and $lp(X(\vec{x})) = lp(\varphi)$ since $fv(\varphi) = \vec{x}$ by definition of the syntax of $\mu\mathcal{L}$. Therefore, correctly, $f(S) \subseteq lp(X(\vec{x}))$. Moreover, the least fixed point of f exists by Knaster-Tarski theorem since the set $2^{lp(X(\vec{x}))}$ ordered by subset inclusion is a complete lattice and the function f used in the definition is monotone. This can be

easily checked by inspection of the definition of the semantics (Definition 6.6), keeping in mind that X is required to occur positively in φ.

As it happens for the non-recursive fragment $\mathcal{L}$, the logic $\mu\mathcal{L}$ could be defined in positive form. The corresponding syntax, given below, includes the dual operators and omits negation, which can then be encoded by duality.

$$\begin{array}{rcl} \varphi & ::= & X(\vec{x}) \mid \mathsf{T} \mid \varphi \wedge \varphi \mid (\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi \mid \langle z \rangle \varphi \mid \mu X(\vec{x}).\varphi \\ & & \mathsf{F} \mid \varphi \vee \varphi \mid \{\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z\}\, \varphi \mid [\, z \,]\, \varphi \mid \nu X(\vec{x}).\varphi \end{array}$$

In the following we will freely use the dual operators.

6.1. Examples

In the previous section we observed that standard HM logic can be viewed as a fragment of $\mathcal{L}$ where we only use the (derived) modality $\langle\!|\mathsf{a}\, x|\!\rangle$. Similarly, the propositional μ-calculus corresponds to a fragment of the the general logic $\mu\mathcal{L}$ where we avoid references to causally dependent/independent events. In particular, since in recursive formulae we do not express causal links between event variables used in different iterations, we can use only propositions without free variables (i.e., of arity 0). Therefore, the μ-calculus corresponds to the following fragment of $\mu\mathcal{L}$:

$$\varphi \ ::= \ X(\epsilon) \mid \mathsf{T} \mid \varphi \wedge \varphi \mid \neg\varphi \mid (\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi \mid \langle z \rangle \varphi \mid \mu X(\epsilon).\varphi$$

For simplicity in the following we omit trailing empty tuples of variables, writing X instead of $X(\epsilon)$.

As first examples of $\mu\mathcal{L}$ formulae we thus have some standard safety and liveness properties inherited from the μ-calculus (see, e.g., [Bradfield and Stirling 2006]). For a fixed closed formula ψ, representing a property of interest:

— ψ holds in every reachable state
$Inv(\psi) = \nu X.\ (\psi \wedge [\![\_\, z]\!] X)$;
— ψ eventually holds in some state
$Pos(\psi) = \mu X.\ (\psi \vee \langle\!|\_\, z|\!\rangle X)$;
— there is a complete (finite terminated or infinite) computation where ψ always holds
$Safe(\psi) = \nu X.\ (\psi \wedge ([\![\_\, z]\!]\mathsf{F} \vee \langle\!|\_\, x|\!\rangle X))$;
— in every complete computation eventually ψ holds
$Ev(\psi) = \mu X.\ (\psi \vee (\langle\!|\_\, z|\!\rangle \mathsf{T} \wedge [\![\_\, x]\!] X))$.

When moving to the full logic, property ψ can include concurrency and causal features. In case ψ is not closed, denoted by $\vec{x}$ the tuple of free variables in ψ, in order to respect the syntax any occurrence of X above must be replaced by $X(\vec{x})$. For instance, we can define $Ev((\langle\!|\mathsf{a}\, z|\!\rangle \otimes \langle\!|\mathsf{a}\, z'|\!\rangle)\mathsf{T})$ saying that eventually there will be a concurrent step consisting of two events, labelled a and b, respectively, or $Inv(\langle\!|\mathsf{r}\, z|\!\rangle Ev(\langle\!|z < \mathsf{s}\, z'|\!\rangle \mathsf{T}))$ saying that any r-labelled event will be eventually followed by an s-labelled event caused by it (e.g., any request will be eventually served).

More generally, logic $\mu\mathcal{L}$ allows one to express causal and concurrency properties of infinite computations, where events occurring in different fixpoint iterations are possibly related. We next provide a number of further examples.

— There is a causal chain of b-labelled events reaching a state where a can be fired:

$$\langle\!|\mathsf{a}\, y|\!\rangle \mathsf{T} \vee \langle\!|\mathsf{b}\, x|\!\rangle\, (\mu X(x).(\langle\!|\mathsf{a}\, z|\!\rangle \mathsf{T} \vee \langle\!|x < \mathsf{b}\, y|\!\rangle\, X(y)))$$

— There is an executable a-labelled event such that in every configuration reached by executing events which are concurrent with it, a c-labelled event can be executed:

$$(\mathsf{a}\,x)(\langle x\rangle\,\mathsf{T} \wedge \nu X(x).(\langle\!\langle \mathsf{c}\,z\rangle\!\rangle \mathsf{T} \wedge [\![\overline{x} < \_\, y]\!]\, X(x)))$$

— It is always possible to perform a step consisting of two concurrent events labelled by a and b, after executing any number of events labelled c:

$$\nu X.\ ((\langle\!\langle \mathsf{a}\,z\rangle\!\rangle \otimes \langle\!\langle \mathsf{b}\,z'\rangle\!\rangle)\mathsf{T} \wedge [\![\mathsf{c}\,w]\!] X)$$

— There is a finite sequence of (not necessarily related) steps, each consisting of two concurrent events labelled by a and b, respectively, leading to a state where a c-labelled event can be executed:

$$\mu X.(\langle\!\langle \mathsf{c}\,z\rangle\!\rangle \mathsf{T} \vee (\langle\!\langle \mathsf{a}\,z\rangle\!\rangle \otimes \langle\!\langle \mathsf{b}\,z'\rangle\!\rangle)\, X)$$

6.2. Invariance of logical equivalence

We show that the addition of fixpoints formulae does not alter the logical equivalence, that still coincides with hhp-bisimilarity, i.e., $\equiv_{\mathcal{L}} = \equiv_{\mu\mathcal{L}} = \sim_{hhp}$. (Recall that in the paper we are limiting ourselves to image-finite PESs.) This is done by adapting the proof of the fact that the μ-calculus induces the same equivalence as HM logic (see, e.g., [Bradfield and Stirling 2006]).

We start by introducing an infinitary version of the logic $\mu\mathcal{L}$, which is then exploited to define fixpoint approximants. Let $\mu\mathcal{L}^\infty$ denote an extension of $\mu\mathcal{L}$ with infinite conjunctions, i.e., formulae of $\mu\mathcal{L}^\infty$ are defined by the grammar

$$\varphi \ ::= \ X(\vec{x}) \ \mid \ \mathsf{T} \ \mid \ \textstyle\bigwedge_{i\in I} \varphi_i \ \mid \ \neg\varphi \ \mid \ (\vec{x}, \overline{\vec{y}} < \mathsf{a}\,z)\,\varphi \ \mid \ \langle z\rangle\,\varphi \mid \ \mu X(\vec{x}).\varphi$$

The semantics of $\mu\mathcal{L}^\infty$ is given as in Definition 6.6, replacing the clause for conjunction with $\{\!|\bigwedge_{i\in I}\varphi_i|\!\}^{\mathcal{E}}_{\pi} = \bigcap_{i\in I}\{\!|\varphi_i|\!\}^{\mathcal{E}}_{\pi} \cap lp(\bigwedge_{i\in I}\varphi_i)$. We denote by $\mathcal{L}^\infty$ the fragment of $\mu\mathcal{L}^\infty$ not including propositions and fixpoint operators.

Definition 6.7 (*approximants*). The α-th approximant of a fixpoint formula in $\mu\mathcal{L}^\infty$, for an ordinal α, is a formula in $\mathcal{L}^\infty$, inductively defined as follows:

$$\begin{aligned}
&\mu^0 X(\vec{x}).\varphi = \mathsf{F}\\
&\mu^{\alpha+1} X(\vec{x}).\varphi = \varphi[\mu^\alpha X(\vec{x}).\varphi/X]\\
&\mu^\lambda X(\vec{x}).\varphi = \textstyle\bigvee_{\alpha<\lambda} \mu^\alpha X(\vec{x}).\varphi \qquad \text{for } \lambda \text{ a limit ordinal}
\end{aligned}$$

A fixpoint formula $\mu X(\vec{x}).\varphi$ is intuitively equivalent to the (infinite) disjunction of its approximants. More formally:

LEMMA 6.8 (FIXPOINT UNFOLDING VIA APPROXIMANTS). *Let $\mathcal{E}$ be a* PES. *For any formula $\mu X(\vec{x}).\varphi$ in $\mu\mathcal{L}^\infty$ there exists an ordinal α such that*

$$\{\!|\mu X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi} = \{\!|\mu^\alpha X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi}.$$

PROOF. Recall that $\{\!|\mu X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi} = lfp(f)$ where $f : 2^{lp(X(\vec{x}))} \to 2^{lp(X(\vec{x}))}$ is the function defined by $f(S) = \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi[X(\vec{x})\mapsto S]}$.

We already noted that the function f is monotone in $2^{lp(X(\vec{x}))}$ ordered by subset inclusion. Hence its least fixpoint can be obtained by iterating f on $\emptyset$, the bottom element of the lattice, i.e., there exists an ordinal α such that $lfp(f) = f^\alpha(\emptyset)$, where $f^0(\emptyset) = \emptyset$, $f^{\alpha+1}(\emptyset) = f(f^\alpha(\emptyset))$ and $f^\lambda(\emptyset) = \bigcup_{\alpha<\lambda} f^\alpha(\emptyset)$ for λ a limit ordinal.

The observation that for any ordinal α it holds that $f^\alpha(\emptyset) = \{\!|\mu^\alpha X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi}$ allows us to conclude. The latter can be proved by transfinite induction on α.

($\alpha = 0$) $\{\!|\mu^0 X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi} = \{\!|\mathsf{F}|\!\}^{\mathcal{E}}_{\pi} = \emptyset = f^0(\emptyset)$

($\alpha \to \alpha+1$) We have that

$$
\begin{aligned}
&\{\!|\mu^{\alpha+1}X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi} = && \text{[definition of } \mu^{\alpha+1}X(\vec{x}).\varphi\text{]}\\
&= \{\!|\varphi[\mu^{\alpha}X(\vec{x}).\varphi/X]|\!\}^{\mathcal{E}}_{\pi} = && \text{[Lemma 6.5]}\\
&= \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi[X(\vec{x})\mapsto\{\!|\mu^{\alpha}X(\vec{x}).\varphi|\!\}_{\pi}]} = && \text{[definition of } f\text{]}\\
&= f(\{\!|\mu^{\alpha}X(\vec{x}).\varphi|\!\}_{\pi}) = && \text{[inductive hypothesis]}\\
&= f(f^{\alpha}(\emptyset))
\end{aligned}
$$

(λ limit ordinal) We have

$$
\begin{aligned}
&\{\!|\mu^{\lambda}X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi} = && \text{[definition of } \mu^{\lambda}X(\vec{x}).\varphi\text{]}\\
&= \{\!|\bigvee_{\alpha<\lambda}\mu^{\alpha}X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi} = && \text{[from Definition 6.6]}\\
&= (\bigcup_{\alpha<\lambda}\{\!|\mu^{\alpha}X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi}) \cap lp(\bigvee_{\alpha<\lambda}\mu^{\alpha}X(\vec{x}).\varphi) = && \text{[distributivity of } \cap \text{ w.r.t. } \cup\text{]}\\
&= \bigcup_{\alpha<\lambda}(\{\!|\mu^{\alpha}X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi} \cap lp(\bigvee_{\beta<\lambda}\mu^{\alpha}X(\vec{x}).\varphi)) = && [lp(\bigvee_{\beta<\lambda}\mu^{\alpha}X(\vec{x}).\varphi) = lp(\mu^{\beta}X(\vec{x}).\varphi)\\
& && \text{for any } \beta\text{, as all approximants have}\\
& && \text{the same free variables]}\\
&= \bigcup_{\beta<\lambda}(\{\!|\mu^{\alpha}X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi} \cap lp(\mu^{\alpha}X(\vec{x}).\varphi)) = && [\text{since } \{\!|\mu^{\alpha}X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi} \subseteq lp(\mu^{\alpha}X(\vec{x}).\varphi)]\\
&= \bigcup_{\beta<\lambda}\{\!|\mu^{\alpha}X(\vec{x}).\varphi|\!\}^{\mathcal{E}}_{\pi} = && \text{[by inductive hypothesis]}\\
&= \bigcup_{\alpha<\lambda} f^{\alpha}(\emptyset) =\\
&= f^{\lambda}(\emptyset)
\end{aligned}
$$

□

We can finally prove that the logical equivalences induced by $\mathcal{L}$ and $\mu\mathcal{L}$ are the same and they both coincide with $\sim_{hhp}$.

THEOREM 6.9 (INVARIANCE OF LOGICAL EQUIVALENCE). *The logical equivalences of $\mathcal{L}$ and $\mu\mathcal{L}$ coincide with $\sim_{hhp}$.*

PROOF. First of all, since $\mu\mathcal{L}$ extends $\mathcal{L}$, clearly $\equiv_{\mu\mathcal{L}}$ implies $\equiv_{\mathcal{L}}$ which in turn, by Proposition 4.2, implies $\sim_{hhp}$. Hence $\equiv_{\mu\mathcal{L}}$ implies $\sim_{hhp}$. For the opposite direction, note that Proposition 4.4 can be straightforwardly adapted to logic $\mathcal{L}^{\infty}$ (as finiteness of conjunction plays no role in the proof). Hence $\sim_{hhp}$ implies $\equiv_{\mathcal{L}^{\infty}}$. An inductive argument, using Lemma 6.8, allows one to show that for any closed formula in $\mu\mathcal{L}^{\infty}$ (and thus in particular any formula in $\mu\mathcal{L}$), there exists an equivalent formula in $\mathcal{L}^{\infty}$, obtained by replacing all fixpoint operators with suitable approximants. Therefore $\equiv_{\mathcal{L}^{\infty}}$ implies $\equiv_{\mu\mathcal{L}}$, hence $\sim_{hhp}$ implies $\equiv_{\mu\mathcal{L}}$ as desired. □

We conclude this section by mentioning that fragments of $\mu\mathcal{L}$ corresponding to fixpoint extension of step, pomset and history preserving logic can be defined in the obvious way. The invariance of logical equivalence for these fragments can be easily proved along the lines of the previous proof.

7. CONCLUSIONS: RELATED AND FUTURE WORK

We have introduced a logic for true concurrency, which allows us to predicate on events in computations and their mutual dependencies (causality and concurrency). The logic subsumes standard HM logic and provides a characterisation of the most widely known true concurrent behavioural equivalences: hhp-bisimilarity is the logical equivalence induced by the full logic, and suitable fragments are identified which induce hp-bisimilarity, pomset and step bisimilarity.

As we mentioned in the introduction, there is a vast literature relating logical and operational views of true concurrency, however, to the best of our knowledge, a uniform logical counterpart of the true concurrent spectrum was still missing. An exhaustive

account of the related literature is impossible; we just recall here the approaches that most closely relate to our work.

In [De Nicola and Ferrari 1990; Pinchinat et al. 1994; Cherief 1992] the causal structure of concurrent systems is pushed into the logic. The paper [De Nicola and Ferrari 1990] considers modalities which describe pomset transitions, thus providing an immediate characterisation of pomset bisimilarity. Moreover, [De Nicola and Ferrari 1990; Pinchinat et al. 1994; Cherief 1992] show that by tracing the history of states and adding the possibility of reverting pomset transitions, one obtains an equivalence coarser than hp-bisimilarity and incomparable with pomset bisimilarity, called weak hp-bisimilarity. Our logic intends to be more general by also capturing the interplay between concurrency and branching, which is not observable at the level of hp-bisimilarity.

The idea of studying logics for true concurrency, identifying suitable fragments which induce known or meaningful behavioural equivalences has been considered by several authors. In particular, a recent work [Gutierrez 2011] discusses a fixpoint modal logic for true concurrent models, called separation fixpoint logic (SFL), originally introduced in [Gutierrez 2009]. The logic SFL includes modalities which specify the execution of an action causally dependent/independent on the last executed one. Moreover, a "separation operator" deals with concurrently enabled actions. This line of work is in turn inspired by the so-called independence-friendly modal logic (IFML) [Bradfield and Fröschle 2002], which includes a modality that allows one to specify that the currently executed action is independent from a number of previously executed ones. In this sense IFML is similar in spirit to our logic. Equivalences induced by (fragments of) IFML, with alternative semantics, are investigated and shown to be often not standard in the true concurrent spectrum. The fragment of the logic in [Gutierrez 2011] without the separation operator captures a weakening of hp-bisimilarity, which coincides with hp-bisimilarity on a suitable subclass of safe Petri nets [Gutierrez 2011]. For similar reasons, the full logic induces an equivalence which is weaker than hhp-bisimilarity, and incomparable with hp-bisimilarity. Still a deeper comparison with this approach represents an interesting open issue.

Several classical papers have considered temporal logics with modalities corresponding to the "retraction" or "backward" execution of computations. In particular [Joyal et al. 1996; Nielsen and Clausen 1995; Bednarczyk 1991; Hennessy and Stirling 1985] study a so-called path logic with a past tense (also called future perfect) modality: the formula $@_a \varphi$ is true when φ holds in a state which can reach the current one with an a-transition. For systems that do not exhibit autoconcurrency, i.e., where events with the same label are never enabled concurrently, such a logic can be shown to characterise hhp-bisimilarity. The restriction to systems without autoconcurrency can be relaxed by modifying the past tense modality in a way which allows one to undo a specific event executed in the past [Nielsen and Clausen 1995]. With such a modification the logic becomes event-based logic, similar in spirit to our logic $\mathcal{L}$.

Compared to these works, the main novelty of our approach resides in the fact that the logic $\mathcal{L}$ provides a characterisation of the different standard true concurrent equivalences in a simple, unitary logical framework. In order to enforce this view, we intend to pursue a formal comparison with the logics for concurrency introduced in the literature. It is easy to see that the execution modalities of [Gutierrez 2011] can be encoded in $\mathcal{L}$ since they only refer to the last executed event, while the formulae in $\mathcal{L}$ can refer to any event executed in the past. On the other hand, the "separation operator" of [Gutierrez 2011], as well as the backward modalities mentioned above (past tense, future perfect, reverting pomset transitions) are not immediately encodable in $\mathcal{L}$. A deeper investigation would be of great help in shading further light on the true concurrent spectrum. Moreover $\mathcal{L}$ suggests an alternative, forward-only, operational

definition of hhp-bisimilarity which we would expect to be closely related to the characterisation of hhp-bisimilarity in [Fröschle and Hildebrandt 1999]. This approach could be inspiring also for other reverse bisimilarities [Phillips and Ulidowski 2010].

Interestingly, the idea of considering a logic with event variables is taken also in a very recent work [Phillips and Ulidowski 2011], which provides an elegant characterisation of (h)hp-bisimilarity via a logic, called event identifier logic (EIL), with a backward execution modality. The logic includes three operators: $\langle x{:}\mathsf{a}\rangle\rangle$, $(x{:}\mathsf{a})$ and $\langle\langle x\rangle$. The formula $\langle x{:}\mathsf{a}\rangle\rangle\varphi$ holds when, starting from the current configuration, an a-labelled event can be executed and, after the execution of such an event the formula φ holds. The formula $(x{:}\mathsf{a})\varphi$ states that the current configuration contains an a-labelled event (which has thus been executed in the past) and formula φ holds. In both cases, the a-labelled event is bound to variable x to be possibly referenced in φ. Finally, $\langle\langle x\rangle$ holds when the event bound to x can be undone and then φ holds. The reason why both logics capture hhp-bisimilarity is conceptually clear: the possibility of performing backward steps can be seen as a mean of exploring alternative different futures. The very same possibility is "primitive" in our logic where we can explore the future of a configuration, without executing the corresponding events. However, the formal relationships between EIL and our logic (e.g., the possibility of encoding backward steps in our logic) is still to be understood and represents a stimulating direction of future research.

As a byproduct of such an investigation, we foresee the identification of interesting extensions of the concurrent spectrum, both at the logical and at the operational side. For instance, it can be shown that the fragment of $\mathcal{L}$ where the operator $(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)$ is restricted to bind z to events consistent with those already quantified induces an equivalence which admits a natural operational definition, it is decidable and lies in between hp- and hhp-bisimilarity, still being different from the equivalences in [Gutierrez 2011].

Connected to this, model-checking and decidability issues are challenging directions of future investigation (see [Penczek 1995] for a survey of these issues over partial order temporal logics and logics based on event structures having explicit operators representing concurrency, causality and conflict). It is known that hhp-bisimilarity is undecidable, even for finite state systems [Jurdzinski et al. 2003], while hp-bisimilarity is decidable [Vogler 1991; Montanari and Pistore 1997]. Characterising decidable fragments of the logic could be helpful in drawing a clearer separation line between decidability and undecidability of concurrent equivalences. A promising direction is to impose a bound on the "causal depth" of the future which the logic can quantify on. In this way one gets a chain of equivalences, coarser than hhp-bisimilarity, which should be closely related with n-hhp bisimilarities introduced and shown to be decidable in [Fröschle and Hildebrandt 1999]. As for verification, we aim at investigating the automata-theoretic counterpart of the logic. In previous papers, hp-bisimilarity has been characterised in automata-theoretic terms using HD-automata [Montanari and Pistore 1997] or Petri nets [Vogler 1991]. It seems that HD-automata [Montanari and Pistore 1997] could provide a suitable automata counterpart of the fragment $\mathcal{L}_{hp}$. Also the game-theoretical approach proposed in [Gutierrez and Bradfield 2009; Gutierrez 2011] for the separation fixpoint logic as well as the model checking techniques developed in [Groote and Willemse 2005] for their first order μ-calculus can be sources of inspiration.

Just note that the model checking problem is not trivial since it may be the case that some formulae have infinite models only, even if we limit ourselves to the finite fragment of the logic. For instance, the formula $\langle\!|\mathsf{a}\, w|\!\rangle\mathsf{T} \wedge \neg(\mathsf{a}\, x)\neg(x < \mathsf{a}\, y)\mathsf{T}$ only holds in an PES which contains an infinite causal chain of a-labelled events. Preliminary

investigations lead us to conjecture that model-checking is decidable on finite state systems for the fixpoint extension of $\mathcal{L}_{hp}$, $\mathcal{L}_p$ and $\mathcal{L}_s$.

A. APPENDIX: WELL-FORMED FORMULAE

In this appendix we identify a fragment of the logic $\mathcal{L}$ where the restriction of the denotations to include only legal pairs is enforced syntactically. The idea is as follows: whenever we bind an event to a variable we declare how it relates to all the events bound to the free variables in the remaining part of the formula.

Definition A.1 (*well-formed formulae*). A formula $\varphi \in \mathcal{L}$ is called *well-formed* when, for any subformula of the kind $(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \psi$, we have that $fv(\psi) \subseteq \vec{x} \cup \vec{y} \cup \{z\}$. We denote by $\mathcal{L}_{wf}$ the fragment of $\mathcal{L}$ consisting of well-formed formulae.

Observe that any subformula of a well-formed formula is well-formed.

The semantics of well-formed formulae can be given as in Definition 3.4, without restricting to legal pairs. We refer to this "unrestricted" semantics as the well-formed denotation of a formula.

Definition A.2 (*semantics of well-formed formulae*). Let $\mathcal{E}$ be a PES. The well-formed denotation of a formula φ in $\mathcal{L}_{wf}$, written $\{\!|\varphi|\!\}^{\mathcal{E}}_{wf} \in 2^{\mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}}}$ is defined inductively as follow:

$$
\begin{aligned}
\{\!|\mathsf{T}|\!\}^{\mathcal{E}}_{wf} &= \mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}} \\
\{\!|\varphi_1 \wedge \varphi_2|\!\}^{\mathcal{E}}_{wf} &= \{\!|\varphi_1|\!\}^{\mathcal{E}}_{wf} \cap \{\!|\varphi_2|\!\}^{\mathcal{E}}_{wf} \\
\{\!|\neg\varphi|\!\}^{\mathcal{E}}_{wf} &= (\mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}}) \setminus \{\!|\varphi|\!\}^{\mathcal{E}}_{wf} \\
\{\!|(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi|\!\}^{\mathcal{E}}_{wf} &= \{(C,\eta) \mid \exists e \in E[C] \text{ such that } \lambda(e) = \mathsf{a} \wedge \eta(\vec{x}) < e \wedge \eta(\vec{y}) \,||\, e \\
&\qquad \wedge (C, \eta[z \mapsto e]) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{wf} \} \\
\{\!|\langle z\rangle\ \varphi|\!\}^{\mathcal{E}}_{wf} &= \{(C,\eta) \mid C \xrightarrow{\eta(z)} C' \wedge (C',\eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{wf} \}
\end{aligned}
$$

The claim that the "well-formedness" is a syntactic counterpart of the restriction to legal pairs is now formalised by proving that, for closed well-formed formulae, the well-formed denotation given above and the one based on legal pairs in Definition 3.4 do coincide.

PROPOSITION A.3 (SEMANTICS OF WELL-FORMED FORMULAE). *Let $\mathcal{E}$ be a* PES. *Then, for any closed well-formed formula φ*

$$\{\!|\varphi|\!\}^{\mathcal{E}} = \{\!|\varphi|\!\}^{\mathcal{E}}_{wf}$$

PROOF. We can prove more generally that for any well-formed formula φ, it holds that

$$\{\!|\varphi|\!\}^{\mathcal{E}} = \{\!|\varphi|\!\}^{\mathcal{E}}_{wf} \cap lp(\varphi).$$

From this the thesis immediately follows, since for a closed formula φ it holds that $lp(\varphi) = \mathcal{C}(\mathcal{E}) \times Env$. The proof can proceed by induction on φ.

(case T) Since $lp(\mathsf{T}) = \mathcal{C}(\mathcal{E}) \times Env$, we have

$$\{\!|\mathsf{T}|\!\}^{\mathcal{E}}_{wf} \cap lp(\mathsf{T}) = (\mathcal{C}(\mathcal{E}) \times Env) \cap (\mathcal{C}(\mathcal{E}) \times Env) = \mathcal{C}(\mathcal{E}) \times Env = \{\!|\mathsf{T}|\!\}^{\mathcal{E}}.$$

(case $\varphi \wedge \psi$) We have

$$\begin{array}{ll}
\{\!|\varphi \wedge \psi|\!\}^{\mathcal{E}}_{wf} \cap lp(\varphi \wedge \psi) & \text{[by Definition A.2]} \\
\quad = \{\!|\varphi|\!\}^{\mathcal{E}}_{wf} \cap \{\!|\psi|\!\}^{\mathcal{E}}_{wf} \cap lp(\varphi \wedge \psi) & \text{[by inductive hypothesis]} \\
\quad = \{\!|\varphi|\!\}^{\mathcal{E}} \cap lp(\varphi) \cap \{\!|\psi|\!\}^{\mathcal{E}} \cap lp(\psi) \cap lp(\varphi \wedge \psi) & \text{[since } lp(\varphi \wedge \psi) \subseteq lp(\varphi) \cap lp(\psi)\text{]} \\
\quad = \{\!|\varphi|\!\}^{\mathcal{E}} \cap \{\!|\psi|\!\}^{\mathcal{E}} \cap lp(\varphi \wedge \psi) & \text{[by Definition 3.4]} \\
\quad = \{\!|\varphi \wedge \psi|\!\}^{\mathcal{E}} &
\end{array}$$

(case $\neg\varphi$) We have

$$\begin{array}{ll}
\{\!|\neg\varphi|\!\}^{\mathcal{E}}_{wf} \cap lp(\neg\varphi) & \text{[by Definition A.2]} \\
\quad ((\mathcal{C}(\mathcal{E}) \times Env) \setminus \{\!|\varphi|\!\}^{\mathcal{E}}_{wf}) \cap lp(\neg\varphi) & \text{[since } lp(\neg\varphi) = lp(\varphi)\text{]} \\
\quad ((\mathcal{C}(\mathcal{E}) \times Env) \setminus \{\!|\varphi|\!\}^{\mathcal{E}}_{wf}) \cap lp(\varphi) & \text{[by calculation]} \\
\quad ((\mathcal{C}(\mathcal{E}) \times Env) \cap lp(\varphi)) \setminus (\{\!|\varphi|\!\}^{\mathcal{E}}_{wf} \cap lp(\varphi)) & \text{[by } lp(\varphi) \subseteq \mathcal{C}(\mathcal{E}) \times Env \text{ and ind. hypot.]} \\
\quad = lp(\varphi) \setminus \{\!|\varphi|\!\}^{\mathcal{E}} & \text{[by Definition 3.4]} \\
\quad = \{\!|\neg\varphi|\!\}^{\mathcal{E}} &
\end{array}$$

(case $(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi$) By Definition A.2 we have

$$\begin{array}{l}
\{\!|(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi|\!\}^{\mathcal{E}}_{wf} \cap lp((\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi) = \\
= \{(C, \eta) \mid\ (C, \eta) \in lp((\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi) \wedge \\
\qquad\qquad \exists e \in E[C].\ \lambda(e) = \mathsf{a}\ \wedge\ \eta(\vec{x}) < e\ \wedge\ \eta(\vec{y}) \,||\, e\ \wedge\ (C, \eta[z \mapsto e]) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{wf}\}
\end{array}$$

Now observe that, since the formula $(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi$ is well-formed, $fv(\varphi) \subseteq \vec{x} \cup \vec{y} \cup \{z\}$ and thus $fv((\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi) = \vec{x} \cup \vec{y}$. As a consequence, whenever $(C, \eta) \in lp((\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi)$ and $e \in E[C]$ with $\eta(\vec{x}) < e$ and $\eta(\vec{y}) \,||\, e$, we have

$$e \frown \eta(fv(\varphi) \setminus \{z\}) \quad \text{and} \quad (C, \eta[z \mapsto e]) \in lp(\varphi).$$

Therefore, we get

$$\begin{array}{l}
\{\!|(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi|\!\}^{\mathcal{E}}_{wf} \cap lp((\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi) = \\
= \{(C, \eta) \mid\ (C, \eta) \in lp((\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi) \wedge \\
\qquad\qquad \exists e \in E[C].\ \ e \frown \eta(fv(\varphi) \setminus \{z\})\ \wedge\ \lambda(e) = \mathsf{a}\ \wedge\ \eta(\vec{x}) < e\ \wedge\ \eta(\vec{y}) \,||\, e \\
\qquad\qquad\qquad\qquad \wedge\ (C, \eta[z \mapsto e]) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{wf} \cap lp(\varphi)\}
\end{array}$$

Since by inductive hypothesis $\{\!|\varphi|\!\}^{\mathcal{E}}_{wf} \cap lp(\varphi) = \{\!|\varphi|\!\}^{\mathcal{E}}$, we deduce that

$$\{\!|(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi|\!\}^{\mathcal{E}}_{wf} \cap lp((\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi) = \{\!|(\vec{x}, \overline{\vec{y}} < \mathsf{a}\, z)\, \varphi|\!\}^{\mathcal{E}}$$

as desired.

(case $\langle z \rangle\ \varphi$) We have

$$
\begin{aligned}
&\{\!|\langle z\rangle\ \varphi|\!\}^{\mathcal{E}}_{wf} \cap lp(\langle z\rangle\ \varphi) && \text{[by Definition A.2]}\\
&= \{(C,\eta) \mid C \xrightarrow{\eta(z)} C' \ \wedge\ (C',\eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{wf}\} \cap lp(\langle z\rangle\ \varphi) && \text{[by calculation]}\\
&= \{(C,\eta) \mid (C,\eta) \in lp(\langle z\rangle\ \varphi) \ \wedge\ C \xrightarrow{\eta(z)} C' \ \wedge\ (C',\eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{wf}\}\\
&\qquad\qquad \text{[since } (C,\eta) \in lp(\langle z\rangle\ \varphi) \ \wedge\ C \xrightarrow{\eta(z)} C' \text{ iff } (C',\eta) \in lp(\varphi) \ \wedge\ C \xrightarrow{\eta(z)} C'\text{]}\\
&= \{(C,\eta) \mid C \xrightarrow{\eta(z)} C' \ \wedge\ (C',\eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{wf} \cap lp(\varphi)\} && \text{[by inductive hypothesis]}\\
&= \{(C,\eta) \mid C \xrightarrow{\eta(z)} C' \ \wedge\ (C',\eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}\} && \text{[by Definition A.2]}\\
&= \{\!|\langle z\rangle\ \varphi|\!\}^{\mathcal{E}}
\end{aligned}
$$

□

Restricting to well-formed formulae does not alter the logical equivalence which remains hhp-bisimilarity.

PROPOSITION A.4 (WELL-FORMED FORMULAE INDUCE HHP-BISIMILARITY). *Let $\mathcal{E}_1$ and $\mathcal{E}_2$ be* PESs. *Then $\mathcal{E}_1 \sim_{hhp} \mathcal{E}_2$ iff $\mathcal{E}_1 \equiv_{\mathcal{L}_{wf}} \mathcal{E}_2$.*

PROOF. The fact that if $\mathcal{E}_1 \sim_{hhp} \mathcal{E}_2$ then $\mathcal{E}_1 \equiv_{\mathcal{L}_{wf}} \mathcal{E}_2$ follows immediately by Proposition 4.4, since $\mathcal{L}_{wf}$ is a fragment of $\mathcal{L}$.

The converse implication can be proved essentially as for the full logic $\mathcal{L}$ (Proposition 4.2) since the restriction to well-formed formulae smoothly integrates in the proof. More in detail, most of the proof of Proposition 4.2, remains unchanged. When showing that relation R is a hhp-bisimilarity, it is sufficient to note that if the formulae ψ^i are assumed to be well-formed then also the newly constructed formula $\varphi = (\vec{x},\vec{y} < \mathsf{a}\, x_e)(\langle X_{C_1}\rangle \langle x_e\rangle \mathsf{T} \wedge \psi^1 \wedge \ldots \wedge \psi^n)$ is well-formed. In fact, by construction, $\vec{x},\vec{y} \subseteq X_{C_1}$ are such that $\eta_1(\vec{x})$ is the set of causes of e in C_1 and $\eta_1(\vec{y})$ is the set of events in C_1, hence $\vec{x} \cup \vec{y} = X_{C_1}$. Moreover $fv(\psi^i) \subseteq X_{C'_1} = X_{C_1} \cup \{x_e\}$ and thus $fv(\langle X_{C_1}\rangle \langle x_e\rangle \mathsf{T} \wedge \psi^1 \wedge \ldots \wedge \psi^n) = X_{C_1} \cup \{x_e\}$. Hence φ is well-formed. □

The entire theory, including the fragments for step, pomset and hp-bisimilarity and the logic with recursion could be developed alternatively by focusing on the well-formed fragment of the logic, with the well-formed semantics.

ACKNOWLEDGMENT

We are grateful to Luca Aceto, Sibylle Fröschle and to the anonymous reviewers for their detailed comments and inspiring suggestions which helped us in improving the the paper. In particular a remark from the reviewers stimulated a more appropriate presentation of well-formed formulae.

REFERENCES

Paolo Baldan and Silvia Crafa. 2010. A Logic for True Concurrency. In *Proceedings of CONCUR'10 (Lecture Notes in Computer Science)*, Paul Gastin and François Laroussinie (Eds.), Vol. 6269. Springer, Heidelberg, DE, 147–161.

Marek A. Bednarczyk. 1991. *Hereditary History Preserving Bisimulations or What is the Power of the Future Perfect in Program Logics*. Technical Report. Polish Academy of Sciences.

Eike Best, Raymond Devillers, Astrid Kiehn, and Lucia Pomello. 1991. Fully Concurrent Bisimulation. *Acta Informatica* 28 (1991), 231–261.

Julian Bradfield and Sibylle B. Fröschle. 2002. Independence-Friendly Modal Logic and True Concurrency. *Nordic Journal of Computing* 9, 1 (2002), 102–117.

Julian Bradfield and Stephan Kreutzer. 2005. The Complexity of Independence-Friendly Fixpoint Logic. In *Proceedings of CLS'05 (Lecture Notes in Computer Science)*, C.-H. Luke Ong (Ed.), Vol. 3634. Springer, Heidelberg, DE, 355–368.

Julian Bradfield and Colin Stirling. 2006. Modal mu-calculi. In *Handbook of Modal Logic*, Patrick Blackburn, Johan van Benthem, and Franck Wolter (Eds.). Elsevier, Amsterdam, NL, 721–756.

Ferroudja Cherief. 1992. Back and Forth Bisimulations On Prime Event Structures. In *Proceedings of PARLE'92 (Lecture Notes in Computer Science)*, Daniel Etiemble and Jean-Claude Syre (Eds.), Vol. 605. Springer, Heidelberg, DE, 843–858.

Mads Dam. 1996. Model Checking Mobile Processes. *Information and Computation* 129, 1 (1996), 35–51.

Mads Dam, Lars-Åke Fredlund, and Dilian Gurov. 1998. Toward Parametric Verification of Open Distributed Systems. In *Proceedings of COMPOS'97 (Lecture Notes in Computer Science)*, Willem P. de Roever, Hans Langmaack, and Amir Pnueli (Eds.), Vol. 1536. Springer, Heidelberg, DE, 150–185.

Rocco De Nicola and Gianluigi Ferrari. 1990. Observational logics and concurrency models. In *Proceedings of FST-TCS'90 (Lecture Notes in Computer Science)*, Kesav V. Nori and C. E. Veni Madhavan (Eds.), Vol. 472. Springer, Heidelberg, DE, 301–315.

Pierpaolo Degano, Rocco De Nicola, and Ugo Montanari. 1988. Partial orderings descriptions and observations of nondeterministic concurrent processes.. In *REX Workshop (Lecture Notes in Computer Science)*, Jaco W. de Bakker, Willem P. de Roever, and Grzegorz Rozenberg (Eds.), Vol. 354. Springer, Heidelberg, DE, 438–466.

Sibylle B. Fröschle and Thomas T. Hildebrandt. 1999. On Plain and Hereditary History-Preserving Bisimulation. In *Proceedings of MFCS'99 (Lecture Notes in Computer Science)*, Miroslaw Kutylowski, Leszek Pacholski, and Tomasz Wierzbicki (Eds.), Vol. 1672. Springer, Heidelberg, DE, 354–365.

Jan Friso Groote and Tim A. C. Willemse. 2005. Model-checking processes with data. *Science of Computer Programming* 56, 3 (2005), 251–273.

Julian Gutierrez. 2009. Logics and Bisimulation Games for Concurrency, Causality and Conflict. In *Proceedings of FoSSaCS'09 (Lecture Notes in Computer Science)*, Luca de Alfaro (Ed.), Vol. 5504. Springer, Heidelberg, DE, 48–62.

Julian Gutierrez. 2011. *On bisimulation and model-checking for concurrent systems with partial order semantics*. Ph.D. Dissertation. LFCS - University of Edinburgh.

Julian Gutierrez and Julian C. Bradfield. 2009. Model-Checking Games for Fixpoint Logics with Partial Order Models. In *Proceedings of CONCUR'09 (Lecture Notes in Computer Science)*, Mario Bravetti and Gianluigi Zavattaro (Eds.), Vol. 5710. Springer, Heidelberg, DE, 354–368.

Matthew Hennessy and Robin Milner. 1985. Algebraic laws for nondeterminism and concurrency. *J. ACM* 32 (1985), 137–161.

Matthew Hennessy and Colin Stirling. 1985. The Power of the Future Perfect in Program Logics. *Information and Control* 67, 1-3 (1985), 23–52.

André Joyal, Mogens Nielsen, and Glynn Winskel. 1996. Bisimulation from Open Maps. *Information and Computation* 127, 2 (1996), 164–185. Originally BRICS Report Series RS-94-7.

Marcin Jurdzinski, Mogens Nielsen, and Jirí Srba. 2003. Undecidability of domino games and hhp-bisimilarity. *Information and Computation* 184, 2 (2003), 343–368.

Ugo Montanari and Marco Pistore. 1997. Minimal Transition Systems for History-Preserving Bisimulation. In *Proceedings of STACS'97 (Lecture Notes in Computer Science)*, Rüdiger Reischuk and Michel Morvan (Eds.), Vol. 1200. Springer, Heidelberg, DE, 413–425.

Mogens Nielsen and Christian Clausen. 1995. Games and logics for a noninterleaving bisimulation. *Nordic Journal of Computing* 2, 2 (1995), 221–249.

Mogens Nielsen, Gordon D. Plotkin, and Glynn Winskel. 1981. Petri Nets, Event Structures and Domains, Part I. *Theoretical Computer Science* 13 (1981), 85–108.

Wojciech Penczek. 1995. Branching Time and Partial Order in Temporal Logics. In *Time and Logic: A Computational Approach*, Leonard Bolc and Andrzej Szałas (Eds.). UCL Press, London, UK, 179–228.

Iain Phillips and Irek Ulidowski. 2010. Reverse Bisimulations on Stable Configuration Structures. In *Proceedings of SOS'09 (Electronic Proceedings in Theoretical Computer Science)*, B. Klin and P. Sobociǹski (Eds.), Vol. 18. Elsevier, Amsterdam, NL, 62–76.

Iain Phillips and Irek Ulidowski. 2011. A Logic with Reverse Modalities for History-preserving Bisimulations. In *Proceedings of EXPRESS 2011 (Electronic Proceedings in Theoretical Computer Science)*, Bas Luttik and Frank Valencia (Eds.), Vol. 64. Elsevier, Amsterdam, NL, 104–118.

Sophie Pinchinat, François Laroussinie, and Philippe Schnoebelen. 1994. *Logical Characterization of Truly Concurrent Bisimulation*. Technical Report 114. LIFIA-IMAG, Grenoble.

Alexander M.. Rabinovich and Boris A. Trakhtenbrot. 1988. Behaviour Structures and Nets. *Fundamenta Informaticae* 11 (1988), 357–404.

Rob J. van Glabbeek. 2001. The Linear Time – Branching Time Spectrum I; The Semantics of Concrete, Sequential Processes. In *Handbook of Process Algebra*, Jan A. Bergstra, Alban Ponse, and Scott A. Smolka (Eds.). Elsevier, Amsterdam, NL, Chapter 1, 3–99.

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

Walter Vogler. 1991. Deciding History Preserving Bisimilarity. In *Proceedings of ICALP'91 (Lecture Notes in Computer Science)*, Javier Leach Albert, Burkhard Monien, and Mario Rodríguez-Artalejo (Eds.), Vol. 510. Springer, Heidelberg, DE, 495–505.

Glynn Winskel. 1987. Event Structures. In *Petri Nets: Applications and Relationships to Other Models of Concurrency (Lecture Notes in Computer Science)*, Wilfried Brauer, Wolfgang Reisig, and Grzegorz Rozenberg (Eds.), Vol. 255. Springer, Heidelberg, DE, 325–392.

Glynn Winskel and Mogens Nielsen. 1995. Models for Concurrency. In *Handbook of logic in Computer Science*, Samson Abramsky, Dov M. Gabbay, and Thomas S. E. Maibaum (Eds.). Vol. 4. Clarendon Press, Oxford, UK.