On Fixpoint Logics and Equivalences for Processes with Restricted Nondeterminism

Julian Gutierrez

Department of Computer Science
University of Oxford

Abstract. In concurrency, processes can be studied using a partial order or an interleaving semantics. In partial order semantics, at least four different kinds of behaviour can be recognised: concurrency, causality, conflict, and confusion. In interleaving semantics, only conflicts can be observed. All these features can be characterised in logical terms, and various logics have been defined for this purpose. For instance, Hennessy–Milner logic is a modal language that captures strong bisimilarity, the standard bisimulation equivalence for processes with interleaving semantics. However, when considering processes with partial order semantics, stronger equivalences are used and more discriminating logics are needed.

In the present paper, we study conditions to ease the definition of such *logics and equivalences* for processes with partial order semantics. More specifically, we study the impact that *nondeterminism* can have on some fixpoint modal logics and bisimulation equivalences for concurrent and multi-agent systems, when it is systematically restricted within the four kinds of behaviour mentioned above. Our results show that when the concurrency and confusion relations are taken to be deterministic, then the main equivalence for causal behaviour can be completely captured (even in logical and game-theoretic terms) by a simpler, weaker, more local bisimulation relation. We also provide key examples of the kinds of processes that can be modelled using deterministic confusion in order to illustrate the expressive power of the general framework defined here.

1 Introduction

Concurrency theory studies the logical and mathematical foundations of parallel processes, i.e., of systems composed of independent components which can interact with each other and with an environment. These systems can be analysed by studying the formalisms (logics and methodologies) employed to specify and verify their properties as well as the mathematical structures used to represent their behaviour. Such formalisms and structures make use of models of two different kinds: *interleaving* or *partially ordered*. This semantic feature is particularly important as most logics, tools, and verification techniques for analysing the behaviour of concurrent processes have to take this difference into account.

On the one hand, the interleaving semantics approach reduces concurrency to the *nondeterministic, sequential computation* of the events the processes can

perform independently. On the other hand, partial order semantics represent concurrency *explicitly by means of an independence relation* on the set of events that the processes can execute in parallel; following this approach, the so-called 'true concurrency' approach, independence or concurrency is a *primitive* notion (as in, e.g., Petri nets or event structures) rather than a *derived* concept, as in the interleaving framework (e.g., as in Kripke structures or transition systems).

Choosing either kind of semantics is, however, more than a matter of taste. Each kind of semantics endows concurrency models with different computational properties; see [22, 27], and the references therein, for two comprehensive studies on the subject. Indeed, choosing one kind of semantics over the other has important implications—both from theoretical and practical viewpoints—as making such a choice raises different issues, one of which we study here and explain next.

When considering partial order—also called noninterleaving—semantics, at least four kinds of behaviour can be studied: *concurrency* (parallel, independent execution of events), *causality* (sequential, dependent execution of events), *conflict* (mutually exclusive choice of events), and *confusion* (synchronous and asynchronous combinations of the first three). On the other hand, when considering interleaving semantics the picture is semantically much simpler, as only conflicts can be observed. From a partial order point of view, a process with an interleaving semantics has no concurrency, trivial causality, and no confusion.

Because a way to understand the behaviour of concurrent processes is by understanding the interplay between the kinds of behaviour above mentioned, several *logics and equivalences* have been proposed. For instance, Hennessy–Milner logic (HML [14]) is a modal logic that captures strong bisimilarity [14], the standard bisimulation equivalence for processes with interleaving semantics. However, when considering systems with partial order semantics, stronger equivalences are used—*e.g.*, history-preserving bisimilarity (HPB [10])—and more discriminating logics are needed. It is known that *nondeterminism* can have an impact on the logics and equivalences for concurrent processes. For instance, strong bisimilarity coincides with trace equivalence [15] over deterministic concurrent systems with interleaving semantics. As a consequence, HML reduces in discriminating power to the (strictly simpler) fragment that logically characterises trace equivalence. Here, we investigate a similar phenomenon in the noninterleaving framework: *we study the impact that nondeterminism can have on some logics and equivalences for concurrent processes with partial order semantics.*

Nondeterminism can appear when there are conflicts or there is concurrency, and hence also when there is confusion; as nondeterminism requires multiple events enabled at the same time, the concept makes no sense w.r.t. causality. The restriction to processes where concurrency is deterministic is well-known, and called *no auto-concurrency* in the literature. It is a restriction rather easy to justify, especially when considering multi-agent systems since one can assume that agents have disjoint sets of labelled events or that they execute equally labelled events which can be differentiated by adjoining to them the "identifier" of the process/agent that executes them. Conflicts, on the other hand, are usually regarded to allow nondeterminism—i.e., having equally labelled choices for a se-

quential process—as in the interleaving framework. Finally, nondeterminism can also appear in processes with confusion, even if no auto-concurrency is assumed.

Deterministic confusion is a much less studied restriction. Instead, concurrent systems with no confusion have received much attention in the past [8]. However, it is known [28] that confusion is necessary to properly model communication and mutual exclusion protocols, and thus confusion should be taken into account to model interesting concurrent processes. In fact, a motivation of our work is the following observation: that in such kinds of systems, both no auto-concurrency and deterministic confusion can be assumed without incurring a high cost with respect to expressivity. The restriction can also be imposed when considering the behaviour of many multi-agent systems. Some examples are given later.

Specifically, in this paper, we show that *if nondeterminism is restricted to conflicts*, then the definition of history-preserving bisimilarity (HPB)—which is regarded as the standard bisimulation equivalence for causality—coincides with a simpler, weaker, and local definition, which admits a logical characterisation given by a *causal* extension of HML. This logic can, in turn, be extended with fixpoints, in the same way that the modal μ-calculus [17] extends HML, so that complex causal properties of concurrent processes can be specified. These results basically show that the analyses of causal behaviour in concurrent and multi-agent systems can be carried out in a logically simpler, *local* manner, provided that nondeterminism is solely used to model the choices of the sequential agents, or components, of the concurrent system—and their synchronization mechanisms (where confusion is needed [28]) are kept deterministic. We proceed as follows.

We first study the relationships between logics and equivalences for concurrent processes with partial order semantics purely based on observable *local dualities* between concurrency and conflict, on the one hand, and concurrency and causality, on the other. These dualities, which can be found across several noninterleaving models of concurrency, are mathematically supported in a beautiful way by a simple *axiomatisation of concurrent behaviour*. Although the dualities and axiomatisation are defined with respect to noninterleaving models, such dualities and axiomatisation have an easy interpretation when considering processes with interleaving semantics since they are a particular instance of the general framework. We then define a logical notion of equivalence tailored to be model independent. We do so by defining a number of fixpoint modal logics whose semantics are given by an intermediate structure called a *process space*, which is a mathematical structure designed to be used as a common bridge between the particular models of concurrency under consideration (namely, Petri nets, event structures, and different kinds of transition systems). Roughly speaking, a process space is a structure that represents the local partial order behaviour of a concurrent system, and is built using the local dualities mentioned above. We then compare the logical equivalences induced by these logics with some of the bisimulation equivalences in the literature. In particular, we show that over *processes with deterministic confusion and no auto-concurrency*, some of the standard bisimulation equivalences in the literature coincide with the weaker and simpler logical equivalence relations induced by some of the logics we study.

We also give a game-theoretic analysis/interpretation of the main results, and end with examples of the kinds of systems that the main theorems apply to.

2 Models, Logics, and Games

In this section we provide some background material on the models, logics, and games for concurrency that will be used throughout the paper. For further information on these topics we refer the reader to, e.g., [22], [7], and [29], respectively.

2.1 Models

Petri Nets. A *net* $\mathcal{N}$ is a tuple $(P, C, R, \theta, \Sigma)$, where P is a set of places, C is a set of actions, $R \subseteq (P \times C) \cup (C \times P)$ is a relation between places and actions, and θ is a labelling function $\theta : C \to \Sigma$ from actions to a set Σ of action labels. Places and actions are called nodes; given a node $n \in P \cup C$, the set ${}^\bullet n = \{x \mid (x, n) \in R\}$ is the preset of n and the set $n^\bullet = \{y \mid (n, y) \in R\}$ is the postset of n.[1] States in a net are called markings, which are sets or multisets of places; in the former case—that is, when markings are sets of places—such nets are called *safe*. Safe nets are expressive enough to model any finite-state system. As we are mainly interested in finite-state process, hereafter, we only consider safe nets. Thus, formally, a *Petri net* $\mathfrak{N}$ is a tuple $(\mathcal{N}, M_0)$, where $\mathcal{N} = (P, C, R, \theta, \Sigma)$ is a net and $M_0 \subseteq P$ is its initial marking.

Markings are used to define the dynamics of nets; they do so in the following way. We say that a marking M enables an action t iff ${}^\bullet t \subseteq M$. If t is enabled at M, then t can occur and its occurrence leads to a successor marking M', where $M' = (M \setminus {}^\bullet t) \cup t^\bullet$, written as $M \xrightarrow{t} M'$. Let $\xrightarrow{t}$ be the relation between successor markings and let $\longrightarrow^*$ be its transitive closure. Given a Petri net $\mathfrak{N} = (\mathcal{N}, M_0)$, the relation $\longrightarrow^*$ defines the set of reachable markings in the system $\mathfrak{N}$; such a set of reachable markings is fixed for any pair $(\mathcal{N}, M_0)$, and can be constructed with the occurrence net unfolding construction in [21].

Finally, let par be the symmetric independence relation on actions such that $t_1 \mathsf{par}\ t_2$ iff ${}^\bullet t_1^\bullet \cap {}^\bullet t_2^\bullet = \emptyset$, where ${}^\bullet t^\bullet$ stands for the set ${}^\bullet t \cup t^\bullet$, and there exists a reachable marking M such that both ${}^\bullet t_1 \subseteq M$ and ${}^\bullet t_2 \subseteq M$. Then, if two actions t_1 and t_2 can occur concurrently they must be independent, i.e., $(t_1, t_2) \in \mathsf{par}$. Actions in Petri nets determine event structures [27, 30], a model presented next.

Event Structures. An event structure is a possibly labelled partially ordered set (poset) together with a binary relation on such a set. Formally, a labelled *event structure* $\mathcal{E}$ is a tuple $(E, \preccurlyeq, \sharp, \eta, \Sigma)$, where E is a set of events that are partially ordered by $\preccurlyeq$, the causal dependency relation on events; events in a labelled event structure are 'occurrences' of actions in a system. Additionally,

[1] The reader acquainted with net theory may have noticed that we use the word 'action' instead of 'transition', more common in the literature on (Petri) nets. We have made this choice of notation in order to avoid confusion later on in the document.

$\eta : E \to \Sigma$ is a labelling function from events to a set of labels Σ, and $\sharp \subseteq E \times E$ is an irreflexive and symmetric conflict relation such that the following two conditions hold: (1) if $e_1, e_2, e_3 \in E$ and $e_1 \sharp e_2 \preccurlyeq e_3$, then $e_1 \sharp e_3$; and, moreover, (2) for all events e in E, the set $\{e' \in E \mid e' \preccurlyeq e\}$ is finite.

The independence relation on events is defined with respect to the causal relation $\preccurlyeq$ and conflict relation $\sharp$ on events. Two events e_1 and e_2 are said to be concurrent with each other, denoted by $e_1 \mathsf{co}\ e_2$, iff $e_1 \not\preccurlyeq e_2$ and $e_2 \not\preccurlyeq e_1$ and $\neg(e_1 \sharp e_2)$. States within the model of event structures are called configurations. A configuration C is a conflict-free set of events—that is, if events $e_1, e_2 \in C$, then $\neg(e_1 \sharp e_2)$—such that if $e \in C$ and $e' \preccurlyeq e$, then $e' \in C$. The initial configuration (or initial state) of an event structure $\mathcal{E}$ is by definition the empty configuration $\{\}$. Finally, a successor configuration C' of a configuration C is given by $C' = C \cup \{e\}$ such that $e \notin C$. Write $C \xrightarrow{e} C'$ for this relation, and let $\longrightarrow^*$ be defined similar to the Petri net case. Both Petri nets and event structures have a natural transition system representation, which we now introduce.

Transition Systems with Independence. A *labelled transition system* (LTS) is an edge-labelled graph structure. Formally, an LTS is a tuple (S, T, Σ), where S is a set of vertices called states, Σ is a set of labels, and $T \subseteq S \times \Sigma \times S$ is a set of Σ-labelled edges, which are called transitions. A *rooted* LTS is an LTS with a designated initial state $s_0 \in S$. A transition system with independence is a rooted LTS where independent transitions can be explicitly recognised. Formally, a *transition system with independence* (TSI) $\mathfrak{T}$ is a tuple (S, s_0, T, I, Σ), where S is a set of states with initial state s_0, $T \subseteq S \times \Sigma \times S$ is a transition relation, Σ is a set of labels, and $I \subseteq T \times T$ is an irreflexive and symmetric relation on independent transitions. The relation $\prec$ on transitions defined by

$$(s, a, s_1) \prec (s_2, a, q) \Leftrightarrow$$
$$\exists b.(s, a, s_1) I (s, b, s_2) \wedge (s, a, s_1) I (s_1, b, q) \wedge (s, b, s_2) I (s_2, a, q)$$

expresses that two transitions are 'instances' of the same action, but in two different interleavings. We let $\sim$ be the least equivalence relation that includes $\prec$, i.e., the reflexive, symmetric, and transitive closure of $\prec$. The equivalence relation $\sim$ is used to group all transitions that are instances of the same action in all its possible interleavings. Additionally, I is subject to the following axioms:

- **A1**. $(s, a, s_1) \sim (s, a, s_2) \Rightarrow s_1 = s_2$
- **A2**. $(s, a, s_1)\ I\ (s, b, s_2) \Rightarrow \exists q.(s, a, s_1)\ I\ (s_1, b, q) \wedge (s, b, s_2)\ I\ (s_2, a, q)$
- **A3**. $(s, a, s_1)\ I\ (s_1, b, q) \Rightarrow \exists s_2.(s, a, s_1)\ I\ (s, b, s_2) \wedge (s, b, s_2)\ I\ (s_2, a, q)$
- **A4**. $(s, a, s_1)(\prec \cup \succ)(s_2, a, q)\ I\ (w, b, w') \Rightarrow (s, a, s_1)\ I\ (w, b, w')$

Axiom **A1** says that, from any state, the execution of a transition leads always to a unique state. This axiom represents a determinacy condition. Axioms **A2** and **A3** ensure that independent transitions can be executed in either order. Finally, **A4** ensures that relation I is well defined. More precisely, **A4** says that if t and t' are independent, then all other transitions in the equivalence class $[t]_\sim$ (i.e., all other transitions that are instances of the same action but in different

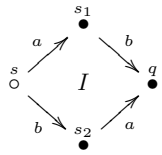

Fig. 1. The 'concurrency diamond' for $t\ I\ t'$, where $t = (s, a, s_1)$ and $t' = (s, b, s_2)$. Concurrency is depicted with the symbol I inside the square. The initial state is $\circ$.

interleavings) are independent of t' as well, and vice versa. Having said that, an alternative, and possibly more intuitive, definition for axiom **A4** can be given. Let $\mathfrak{I}(t)$ be the set $\{t' \mid t\ I\ t'\}$. Then, axiom **A4** is equivalent to this expression: **A4**'. $t \sim t_2 \Rightarrow \mathfrak{I}(t) = \mathfrak{I}(t_2)$. This axiomatisation of concurrent behaviour was defined by Nielsen and Winskel in [22], and can be used to generate a 'concurrency diamond' for any two independent transitions t and t' (see, e.g., Figure 1).

Two interleaving models, namely Kripke structures and trees, are closely related to TSIs. Whereas the former (Kripke structures) are state-labelled TSIs—rather than edge-labelled structures—with an empty independence relation I, the latter (trees) are unfoldings of rooted labelled transitions systems.

A Uniform Representation. Despite being different informatic structures, the three models for concurrency just presented have a number of fundamental relationships between them, as well as with some models for interleaving concurrency. More precisely, TSI are noninterleaving transition-based representations of Petri nets, whereas event structures are unfoldings of TSI. This is analogous to the fact that LTS are interleaving transition-based representations of Petri nets while trees are unfoldings of LTS. There are also simple relationships between TSI and LTS as well as between event structures and trees as follows: LTS are exactly those TSI with an empty independence relation I on transitions, and trees are isomorphic to event structures with an empty relation co on events. In this way, partial order models can generalise the most important interleaving models in concurrency, namely LTS, trees, and Kripke structures. A formal (categorical) presentation of these relationships can be found, e.g., in [22, 27].

Our results will be valid across all the models previously mentioned. Thus, it is convenient to fix some notations to refer unambiguously to any of them. To this end, we will mainly use the TSI notation (in the next subsection we present the maps to determine a TSI based on Petri nets and event structures). Also, with no further distinctions we use the word 'system' when referring to any of these models or to sub-models of them, e.g., to LTS or Kripke structures. The main reason for our choice of notation is that the basic components of a TSI can be easily and uniformly recognised in all the other models studied here. Thus, the translations to TSI are simple and direct.

We would like to note that our generic setting based on the TSI model allows one to see more clearly that the axiomatization (of concurrent behaviour) presented for TSI also holds for the other partial order models when analysing

their local behaviour. The maps presented below can be used to construct a TSI for any given (safe) Petri net or event structure with a finite set of configurations. Then, when analysing either Petri nets or event structures, one first must use such maps to construct their TSI representation, and then perform all analyses in such a uniform framework. The adjunctions between these three models, which can be found in [22, 27], ensure the preservation of behaviour of the maps.

Petri Nets and Event Structures as TSI Models. A Petri net $\mathfrak{N} = (\mathcal{N}, M_0)$, where $\mathcal{N} = (P, C, R, \theta, \Sigma)$ is a net as defined before and M_0 is its initial marking, can be represented as a TSI $\mathfrak{T} = (S, s_0, T, I, \Sigma)$ in the following way:

$$\begin{aligned} S &= \{M \subseteq P \mid M_0 \longrightarrow^* M\} \\ T &= \{(M, a, M') \mid \exists t \in C.\ a = \theta(t), M \xrightarrow{t} M'\} \\ I &= \{((M_1, a, M_1'), (M_2, b, M_2')) \mid \exists (t_1, t_2) \in \mathsf{par}. \\ &\qquad a = \theta(t_1), b = \theta(t_2), M_1 \xrightarrow{t_1} M_1', M_2 \xrightarrow{t_2} M_2'\} \end{aligned}$$

where the set of states S of the TSI $\mathfrak{T}$ represents the set of reachable markings of $\mathfrak{N}$, the initial state s_0 is the initial marking M_0, the set of labels Σ remains the same, and T and I have the expected derived interpretations. Similarly, an event structure $\mathcal{E} = (E, \preccurlyeq, \sharp, \eta, \Sigma)$ determines a TSI $\mathfrak{T} = (S, s_0, T, I, \Sigma)$ by means of the following mapping:

$$\begin{aligned} S &= \{C \subseteq E \mid \{\} \longrightarrow^* C\} \\ T &= \{(C, a, C') \mid \exists e \in E.\ a = \eta(e), C \xrightarrow{e} C'\} \\ I &= \{((C_1, a, C_1'), (C_2, b, C_2')) \mid \exists (e_1, e_2) \in \mathsf{co}. \\ &\qquad a = \eta(e_1), b = \eta(e_2), C_1 \xrightarrow{e_1} C_1', C_2 \xrightarrow{e_2} C_2'\} \end{aligned}$$

where the set of states S is the set of configurations of $\mathfrak{E}$, the initial state s_0 is the initial configuration $\{\}$, and, as before, the set of labels Σ remains the same in both models, and T and I have the expected derived TSI interpretations.

It is important to note that *actions* in a Petri net, *transitions* in a TSI and *events* in an event structure are all different. Namely, transitions are *instances* of actions, i.e., are actions relative to a particular interleaving. Take, e.g., a Petri net with two independent actions ($a \parallel b$ in CCS notation [18]); this net is represented by a TSI with four different transitions, since there are two possible interleavings in such a system, namely $a_1.b_2$ and $b_1.a_2$. Thus, each action in the net for $a \parallel b$ becomes two different transitions in the corresponding TSI.

On the other hand, events are *occurrences* of actions, i.e., are actions relative to the causality relation. For instance, the Petri net representing the system defined by $(a + b).c$, where $a + b$ is the nondeterministic choice between actions a and b, and . is the sequential composition of such a choice with the action c, is represented by four events, instead of only three, because there are two different causal lines for the execution of action c, namely $a.c_1$ and $b.c_2$. Then, the Petri net action c becomes two events c_1 and c_2 in the corresponding event structure.

Before we continue with our presentation, let us introduce some notation. Given $t = (s, a, s')$, also written as $s \xrightarrow{a} s'$ or $s \xrightarrow{t} s'$, we have that: state s is the source of t, and write $\sigma(t) = s$; state s' is the target of t, and write $\tau(t) = s'$; and a is the label of t, and write $\delta(t) = a$.

2.2 Logics

The logics we will study later on are fixpoint modal logics that refine/extend the modal μ-calculus with operators to reason about concurrency and causality.

The Modal μ-Calculus. The *μ-calculus* ($\mathcal{L}_\mu$ [17]) has formulae ϕ built from a set Var of variables $Y, Z, ...$ and a set Σ of labels $a, b, ...$ by the following grammar:

$$\phi ::= Z \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \langle a \rangle \phi_1 \mid [a]\,\phi_1 \mid \mu Z.\phi_1 \mid \nu Z.\phi_1$$

Also, define the Boolean constants as $\mathrm{ff} \stackrel{\text{def}}{=} \mu Z.Z$ and $\mathrm{tt} \stackrel{\text{def}}{=} \nu Z.Z$; and assume these abbreviations: $\langle K \rangle \phi_1$ for $\bigvee_{a \in K} \langle a \rangle \phi_1$ and $[K]\,\phi_1$ for $\bigwedge_{a \in K} [a]\,\phi_1$, where $K \subseteq \Sigma$, as well as $[-]\,\phi_1$ for $[\Sigma]\,\phi_1$ and $[-K]\,\phi_1$ for $[\Sigma \setminus K]\,\phi_1$, and similarly for the 'diamond' modality $\langle a \rangle \phi_1$. The meanings of ff, tt, $\wedge$, and $\vee$ are the usual ones. The semantics of the 'diamond' modality $\langle a \rangle \phi_1$ is, informally, that it is possible to perform an a-labelled action to a state where ϕ_1 holds; and dually for the 'box' modality $[a]\,\phi_1$. Finally, $\mu Z.\phi$ and $\nu Z.\phi$ are, respectively, the minimal (finite looping) and maximal (infinite looping) fixpoint operators of the logic. The denotations of μ-calculus formulae are as follows.

A *μ-calculus model* $\mathfrak{M} = (\mathfrak{T}, \mathcal{V})$ is an LTS $\mathfrak{T} = (S, T, \Sigma)$ together with a valuation $\mathcal{V} : \mathrm{Var} \to 2^S$. The denotation $\|\phi\|^{\mathfrak{T}}_{\mathcal{V}}$ of a formula ϕ in the model $\mathfrak{M}$ is a subset of S given as follows (omitting the superscript $\mathfrak{T}$):

$$\begin{array}{ll}
\|Z\|_{\mathcal{V}} & = \mathcal{V}(Z) \\
\|\phi_1 \wedge \phi_2\|_{\mathcal{V}} & = \|\phi_1\|_{\mathcal{V}} \cap \|\phi_2\|_{\mathcal{V}} \\
\|\phi_1 \vee \phi_2\|_{\mathcal{V}} & = \|\phi_1\|_{\mathcal{V}} \cup \|\phi_2\|_{\mathcal{V}} \\
\|\langle a \rangle \phi_1\|_{\mathcal{V}} & = \{s \in S \mid \exists s' \in S.\ s \xrightarrow{a} s' \wedge s' \in \|\phi_1\|_{\mathcal{V}}\} \\
\|[a]\,\phi_1\|_{\mathcal{V}} & = \{s \in S \mid \forall s' \in S.\ s \xrightarrow{a} s' \Rightarrow s' \in \|\phi_1\|_{\mathcal{V}}\} \\
\|\mu Z.\phi\|_{\mathcal{V}} & = \bigcap \{Q \in 2^S \mid \|\phi\|_{\mathcal{V}[Z:=Q]} \subseteq Q\} \\
\|\nu Z.\phi\|_{\mathcal{V}} & = \bigcup \{Q \in 2^S \mid Q \subseteq \|\phi\|_{\mathcal{V}[Z:=Q]}\}
\end{array}$$

where $\mathcal{V}[Z := Q]$ is the valuation $\mathcal{V}'$ that agrees with $\mathcal{V}$ save that $\mathcal{V}'(Z) = Q$, and the denotation of the Boolean and modal operators is as for HML.

Not surprisingly, the main source of the immense expressiveness of $\mathcal{L}_\mu$ comes from the freedom to mix (or alternate) minimal and maximal fixpoint operators arbitrarily. In fact, one of the most important features of $\mathcal{L}_\mu$ is that many interesting temporal logics, such as LTL or CTL, which are widely used for software and hardware specification and verification, can be embedded into $\mathcal{L}_\mu$.

Another important fragment of $\mathcal{L}_\mu$ is Hennessy–Milner logic (HML), which corresponds to the fixpoint-free language. One of the reasons why HML is an important fragment is that it characterises *bisimilarity* [14], the equivalence relation induced by modal logic. A bisimulation between two rooted systems $\mathfrak{T}_1$ and $\mathfrak{T}_2$ with initial states s_0 and q_0, respectively, is an equivalence relation $\sim_{SB}$ such that $s_0 \sim_{SB} q_0$ if, and only if, they satisfy the same set of HML formulae. Then, we say that the two states s_0 and q_0 (or equivalently the two rooted systems $\mathfrak{T}_1$ and $\mathfrak{T}_2$) are bisimilar iff there is a bisimulation equivalence between them. Additional to the logical characterisation given by modal logics, bisimilarities can also be characterised in a game-theoretic way—shown next.

2.3 Logic Games for Bisimulation

A logic game [4] is played by two *players*, a "Verifier" ($\exists$) and a "Falsifier" ($\forall$), in order to verify or refute the truth or falsity of a given property. In these games the Verifier tries to show that the property holds, whereas the Falsifier wants to refute such an assertion. Solving these games amounts to answering the question of whether the Verifier has a *strategy* to win all plays in the game. Usually the *board* where the game is played is a graph (or labelled transition-based) structure in which each position of the board belongs to only one of the two players, which hereafter will be called Eve (the Verifier) and Adam (the Falsifier).

Bisimulation Games. Bisimulation games are formal and interactive characterisations of a family of equivalence relations called bisimulation equivalences. One of the simplest bisimulation equivalences is *bisimilarity*, the equivalence relation induced by modal logic. More precisely, a bisimulation game $\mathcal{G}(\mathfrak{T}_1, \mathfrak{T}_2)$ is a formal representation of a bisimulation equivalence $\sim_{eq}$ between two systems $\mathfrak{T}_1$ and $\mathfrak{T}_2$. Whereas Eve believes that $\mathfrak{T}_1 \sim_{eq} \mathfrak{T}_2$, Adam wants to show that $\mathfrak{T}_1 \not\sim_{eq} \mathfrak{T}_2$. All plays start in the initial position (s_0, q_0) consisting of the initial states of the systems, and the players take alternating turns—although Adam always plays first and chooses where to play. Thus, in every round of the game Adam makes the first move in either system according to a set of rules, and then Eve must make a corresponding $\sim_{eq}$-equivalent move on the other system; the game can proceed in this way indefinitely. Thus, the plays of the game can be of finite or infinite length. All plays of infinite length are winning for Eve; in the case of plays of finite length, the player who cannot make a move loses the game. These winning conditions apply to all the bisimulation games we study here. The best known bisimulation game for interleaving concurrency is the one that characterises *strong bisimilarity* (SB [14]), the bisimulation equivalence induced by HML (i.e., the fixpoint-free fragment of $\mathcal{L}_\mu$), which we present next.

Strong Bisimulation Games. A bisimulation game for strong bisimilarity is played on a board $\mathfrak{B}$ composed of pairs (s, q) of states s and q of two systems $\mathfrak{T}_1$ and $\mathfrak{T}_2$, respectively. Such a pair is a position of the board $\mathfrak{B}$ and is called a 'configuration' of the game. The position (s_0, q_0), where s_0 and q_0 are the initial states of $\mathfrak{T}_1$ and $\mathfrak{T}_2$, is the initial configuration. Since the strategies λ of the game are history-free, i.e., $\lambda : \mathfrak{B} \to \mathfrak{B}$, then a strategy λ is a partial function on $\mathfrak{B} \subseteq S \times Q$, where S and Q are the state sets of $\mathfrak{T}_1$ and $\mathfrak{T}_2$, respectively.

Because a system has a unique initial state, a bisimulation game can be presented as $\mathcal{G}(\mathfrak{T}_1, \mathfrak{T}_2)$ or $\mathcal{G}(s_0, q_0)$ if the two systems are obvious from the context. Also, since bisimulation games are symmetric, we omit the subscript in $\mathfrak{T}$ whenever referring to either system. Then, define a *strong bisimulation game* as follows. Let (s, q) be a configuration of the game $\mathcal{G}(\mathfrak{T}_1, \mathfrak{T}_2)$. There are two players, Adam and Eve, and Adam always plays first and chooses where to play. The relation R_{SB} is a strong bisimulation, $\sim_{SB}$, between $\mathfrak{T}_1$ and $\mathfrak{T}_2$ if:

– (Base case) The initial configuration (s_0, q_0) is in R_{SB}.

– ($\sim_{SB}$ rule) If (s, q) is in R_{SB} and Adam chooses a transition in $\mathfrak{T}$, say a transition $s \xrightarrow{a} s'$ of $\mathfrak{T}_1$, then Eve must choose a transition in the other system (any $q \xrightarrow{a} q'$ of $\mathfrak{T}_2$ in this case), such that $(s', q') \in R_{SB}$ as well.

We say that $\mathfrak{T}_1 \sim_{SB} \mathfrak{T}_2$ iff Eve has a winning strategy for the SB game $\mathcal{G}(\mathfrak{T}_1, \mathfrak{T}_2)$. This bisimulation game does not capture any information of partial order models that is not already present in their interleaving counterparts. For this reason, games for strong bisimilarity are games for interleaving concurrency rather than for partial order concurrency. In order to capture properties of partial order models, one has to recognise at least when two transitions of a system are independent and hence executable in parallel. This feature is captured, for instance, by the following finer game which characterises causality in a *global* way.

History-Preserving Bisimulation Games. A game for history-preserving bisimilarity (HPB) is a bisimulation game as presented before with a further global 'synchronisation' requirement on transitions. Such a synchronisation requirement makes the selection of transitions by Eve more restricted. Let us first define this notion of synchronisation before giving the formal presentation of the game.

A possibly empty sequence of transitions $\pi = [t_1, ..., t_k]$ is a *run* of a system $\mathfrak{T}$. Let $\Pi_{\mathfrak{T}}$ be the set of runs of $\mathfrak{T}$ and $\varrho(\pi)$ be the last transition of π. Define $\epsilon = \varrho([\,]) $ and $s_0 = \sigma(\epsilon) = \tau(\epsilon)$, for an empty sequence $[\,]$. Given a run π and a transition t, the sequence $\pi.t$ denotes the run π extended with t. Let $\pi_1 \in \Pi_{\mathfrak{T}_1}$ and $\pi_2 \in \Pi_{\mathfrak{T}_2}$ for two systems $\mathfrak{T}_1$ and $\mathfrak{T}_2$. We say that the pair of runs $(\pi_1.u, \pi_2.v)$ is *synchronous* iff $(\varrho(\pi_1), u) \in I_1 \Leftrightarrow (\varrho(\pi_2), v) \in I_2$, where I_1 and I_2 are the independence relations of $\mathfrak{T}_1$ and $\mathfrak{T}_2$, and the posets induced by $\pi_1.u$ with I_1 and $\pi_2.v$ with I_2 (a global condition) are isomorphic.[2] By definition (ϵ, ϵ) is synchronous. As it is more convenient to define HPB games on pairs of runs rather than states, a configuration of the game will be a pair of runs.

Then, define a *history-preserving bisimulation game* as follows. Let (π_1, π_2) be a configuration of the game $\mathcal{G}(\mathfrak{T}_1, \mathfrak{T}_2)$. The initial configuration of the game is (ϵ, ϵ). The relation R_{HPB} is a history-preserving bisimulation, $\sim_{HPB}$, between $\mathfrak{T}_1$ and $\mathfrak{T}_2$ iff it is a strong bisimulation relation between $\mathfrak{T}_1$ and $\mathfrak{T}_2$ and:

– (Base case) The initial configuration (ϵ, ϵ) is in R_{HPB}.
– ($\sim_{HPB}$ rule) If (π_1, π_2) is in R_{HPB} and Adam chooses a transition u in either system, say in $\mathfrak{T}_1$, such that $u = \tau(\varrho(\pi_1)) \xrightarrow{a} s'$, then Eve must choose a transition v in the other system such that $v = \tau(\varrho(\pi_2)) \xrightarrow{a} q'$ and the new configuration $(\pi_1.u, \pi_2.v)$ is synchronous, i.e., $(\pi_1.u, \pi_2.v) \in R_{HPB}$ as well.

Say that $\mathfrak{T}_1 \sim_{HPB} \mathfrak{T}_2$ iff Eve has a winning strategy in the HPB game $\mathcal{G}(\mathfrak{T}_1, \mathfrak{T}_2)$. Unlike the game for $\sim_{SB}$, an HPB game can capture properties of partial order models. An example is, e.g., two processes $a \parallel b$ and $a.b + b.a$, which are equivalent from an interleaving viewpoint, but different w.r.t. their partial order semantics.

[2] Given a run π and an independence relation I, there is a poset $(E, \leq_E)$ induced by π with I such that E has as elements the event occurrences associated with the transitions in π and where the partial order relation $\leq_E$ is defined by the event structure unfolding of the system whose independence relation is I.

3 A Logic for Local Causality and Independence

In this section we study the underlying mathematical properties of the partial order models of concurrency presented before, and show that the behaviour of these systems can be captured in a uniform way by two simple and general dualities of local behaviour. We use these dualities of local behaviour to define a μ-calculus (a fixpoint logic) with a noninterleaving semantics where causality and independence are captured/defined in a *local* way. The logic provides a logical definition and characterisation of *local, causal, and independent* behaviour.

3.1 Local Dualities in Partial Order Models

We present two ways in which concurrency can be regarded as a dual concept to *conflict* and *causality*, respectively. These two ways of observing concurrency will be called *immediate concurrency* and *linearised concurrency*. Whereas immediate concurrency is dual to conflict, linearised concurrency is dual to causality.

The intuitions behind these two observations are the following. Consider a concurrent system and any two different transitions t and t' with the same source node, i.e., $\sigma(t) = \sigma(t')$. These two transitions are either immediately concurrent, and therefore independent, i.e., $(t, t') \in I$, or dependent, in which case they must be in conflict. Similarly, consider any two transitions t and t' where $\tau(t) = \sigma(t')$. Again, the pair of transitions (t, t') can either belong to I, in which case the two transitions are concurrent, yet have been linearised, or the pair does not belong to I, and therefore the two transitions are causally dependent. In both cases, the two conditions are mutually exclusive and there are no other possibilities.

The local dualities just described are formally defined in the following way:

$$\begin{aligned}
\otimes &\stackrel{\text{def}}{=} \{(t, t') \in T \times T \mid \sigma(t) = \sigma(t') \wedge t \; I \; t'\} \\
\# &\stackrel{\text{def}}{=} \{(t, t') \in T \times T \mid \sigma(t) = \sigma(t') \wedge \neg(t \; I \; t')\} \\
\ominus &\stackrel{\text{def}}{=} \{(t, t') \in T \times T \mid \tau(t) = \sigma(t') \wedge t \; I \; t'\} \\
\leq &\stackrel{\text{def}}{=} \{(t, t') \in T \times T \mid \tau(t) = \sigma(t') \wedge \neg(t \; I \; t')\}
\end{aligned}$$

Notice the dual conditions between $\otimes$ and $\#$ and between $\ominus$ and $\leq$ with respect to the independence relation, if assuming valid the locality requirement. Let t and t' be two transitions. We say that t and t' are *immediately concurrent* iff $(t, t') \in \otimes$, *in conflict* iff $(t, t') \in \#$, *linearly concurrent* iff $(t, t') \in \ominus$, or *causally dependent* iff $(t, t') \in \leq$.

Sets in a Local Context. The relation $\otimes$ on pairs of transitions, can be used to recognise *sets* where every transition is independent of each other and hence can all be executed concurrently. Such sets are said to be *conflict-free* and belong to the same 'trace'. Formally, a *conflict-free* set of transitions P is a set of transitions with the same source, where $t \otimes t'$ for each two elements in P.

Notice that by definition empty sets and singleton sets are trivially conflict-free. Given a system $\mathfrak{T}$, all conflict-free sets of transitions at a state s can be

defined locally from the *maximal set* of transitions $\mathfrak{X}(s)$, where $\mathfrak{X}(s)$ is the set of all transitions t such that $\sigma(t) = s$. We simply write $\mathfrak{X}$ when the state s is defined elsewhere or is implicit from the context. Moreover, all maximal sets and conflict-free sets of transitions are fixed given a particular system $\mathfrak{T}$.

Definition 3.1 Given a system $\mathfrak{T}$, a *support* set R in $\mathfrak{T}$ is either a maximal set of transitions in $\mathfrak{T}$ or a non-empty conflict-free set of transitions in $\mathfrak{T}$. $\triangleleft$

Given a system $\mathfrak{T}$, the set of all its support sets is denoted by $\mathfrak{P}$. As can be seen from the definition, support sets can be of two kinds, and one of them provides us with a way of doing local reasoning. More precisely, doing local reasoning on sets of independent transitions becomes possible when considering conflict-free sets since they can be decomposed into smaller sets, where every transition is, as well, independent of each other. Using standard notation on sets, we write $P_1 \uplus P_2$ to denote the disjoint union of sets P_1 and P_2. If both P_1 and P_2 are support sets then we have that $P_1 \neq \emptyset$ and $P_2 \neq \emptyset$, and hence $P_1 \uplus P_2 \neq \emptyset$.

Definition 3.2 Given a support set R, a *complete trace* M of R, denoted by $M \sqsubseteq R$, is a support set $M \subseteq R$ such that $\neg\exists t \in R \setminus M.\ \forall t' \in M.\ t \otimes t'$. $\triangleleft$

Note that if R is a conflict-free support set, then $M = R$. Otherwise, R necessarily is a maximal set $\mathfrak{X}$ and M must be a proper subset of R. Therefore, if $R = \mathfrak{X}$, then the sets M such that $M \sqsubseteq \mathfrak{X}$ are the biggest conflict-free support sets that can be recognised in a particular state s of a system $\mathfrak{T}$; we call them *maximal traces*. Since all complete and maximal traces are support sets, then they are also fixed and computable given a particular finite system $\mathfrak{T}$.

Example 3.3 (Sets) Consider a system with a state s and transitions a, b, c where $a \otimes b$ and both $a\#c$ and $b\#c$. The following sets are determined in s:

- maximal set $\{a, b, c\}$ and non-empty conflict-free sets $\{\{a, b\}, \{a\}, \{b\}, \{c\}\}$;
- maximal traces $\{\{a, b\}, \{c\}\}$. $\triangleleft$

3.2 A Fixpoint Logic for Local Causality and Independence

The local dualities and sets defined before are used to build the semantics of a fixpoint logic which captures *causal and independent* (i.e., partial order) behaviour in noninterleaving models. The semantics of this logic is based on the recognition of the dualities that can be defined in a partial order model for concurrency. The logic is called Trace Fixpoint Logic ($\mathbb{L}_\mu$). As defined by its semantics, $\mathbb{L}_\mu$ captures the duality between concurrency and causality by refining the modal operator of $\mathcal{L}_\mu$. On the other hand, the duality between concurrency and conflict is captured by a modality that recognises maximal traces in the system.

Process Spaces. In order to define the semantics of $\mathbb{L}_\mu$ we construct an intermediate structure into which any of the systems we consider here can be mapped. Such a structure determines a 'space of processes', which are simple abstract entities representing pieces of isolated (i.e., local and independent) behaviour.

Definition 3.4 Let $\mathfrak{T} = (S, s_0, T, \Sigma, I)$ be a system. A *Process Space* $\mathbb{S}$ is the lattice $\mathfrak{P} \times \mathfrak{A}$, where $\mathfrak{P}$ is the set of support sets of $\mathfrak{T}$ and $\mathfrak{A}$ is the set of transitions $T \cup \{t_\epsilon\}$, such that t_ϵ is the empty transition satisfying that for all $t \in T$, if $s_0 = \sigma(t)$ then $t_\epsilon \leq t$. A tuple $(R, t) \in \mathbb{S}$ is called a process, and the initial process of $\mathbb{S}$ is the tuple $(\mathfrak{X}_0, t_\epsilon)$, where $\mathfrak{X}_0 = \mathfrak{X}(s_0)$. $\triangleleft$

Notice that for any process it is always possible to infer the particular state in $\mathfrak{T}$ to which such a process relates. Since a process does not represent explicitly the state of a system we say that a process space is *stateless*. Also, let $\mathcal{X}$ be the subset of $\mathfrak{P}$ that contains only *maximal sets* $\mathfrak{X}$ and *maximal traces* M. Call $\mathfrak{S} = \mathcal{X} \times \mathfrak{A}$ a *stateless maximal* process space.

Trace Fixpoint Logic. Having defined local dualities in partial order models and a process space upon them, we are now ready to present a modal logic that is sensitive to local causal and independent information and that allows for reasoning on the traces of concurrent systems with partial order semantics.

Definition 3.5 *Trace Fixpoint Logic* ($\mathbb{L}_\mu$) has formulae ϕ built from a set Var of variables $Y, Z, ...$ and a set Σ of labels $a, b, ...$ by the following grammar:

$$\phi ::= Z \mid \neg\phi_1 \mid \phi_1 \wedge \phi_2 \mid \langle a \rangle_c \phi_1 \mid \langle a \rangle_{nc} \phi_1 \mid \langle \otimes \rangle \phi_1 \mid \mu Z.\phi_1$$

where $Z \in \text{Var}$ and $\mu Z.\phi_1$ has the restriction that any free occurrence of Z in ϕ_1 must be within the scope of an even number of negations. $\triangleleft$

Dual operators, namely "$\vee$", "$[a]_c$", "$[a]_{nc}$", "$[\otimes]$", and "ν", respectively, are defined in the usual way for fixpoint and modal logics. Moreover, it is easy to show [12] that 'plain' modalities, i.e., HML modalities, can be represented as follows: $\langle a \rangle \phi_1 \stackrel{\text{def}}{=} \langle a \rangle_c \phi_1 \vee \langle a \rangle_{nc} \phi_1$ and $[a]\,\phi_1 \stackrel{\text{def}}{=} [a]_c\,\phi_1 \wedge [a]_{nc}\,\phi_1$.

Informally, the meaning of the basic $\mathbb{L}_\mu$ operators is the following: Boolean constants and operators are interpreted in the usual sense; the semantics of the 'causal' diamond modality $\langle a \rangle_c \phi_1$ (resp. of the 'non-causal' diamond modality $\langle a \rangle_{nc} \phi_1$) is that a process (R, t) satisfies $\langle a \rangle_c \phi_1$ (resp. $\langle a \rangle_{nc} \phi_1$) if it can perform an a-labelled action r that causally depends on t (resp. that is independent of t) and move through r into a process where ϕ_1 holds; and dually for the causal and non-causal box modalities $[a]_c\,\phi_1$ and $[a]_{nc}\,\phi_1$. The modality $\langle \otimes \rangle \phi_1$ provides local second-order power on conflict-free sets of transitions, i.e., on sets of independent transitions. This modality allows one to restrict, locally, the behaviour of a system to those executions that can actually happen concurrently at a given state. Finally, the meaning of the fixpoint operators is as for $\mathcal{L}_\mu$.

Formally, the semantics of $\mathbb{L}_\mu$ is as follows.

Definition 3.6 A $\mathbb{L}_\mu$ *model* $\mathfrak{M}$ is a system $\mathfrak{T} = (S, s_0, T, I, \Sigma)$ together with a valuation $\mathcal{V} : \mathrm{Var} \to 2^{\mathfrak{S}}$, where $\mathfrak{S} = \mathcal{X} \times \mathfrak{A}$ is the stateless maximal process space associated with $\mathfrak{T}$. The denotation $\|\phi\|^{\mathfrak{T}}_{\mathcal{V}}$ of a formula ϕ in the model $\mathfrak{M} = (\mathfrak{T}, \mathcal{V})$ is a subset of $\mathfrak{S}$, given by the following rules (omitting the superscript $\mathfrak{T}$):

$$\begin{array}{ll}
\|Z\|_{\mathcal{V}} & = \mathcal{V}(Z) \\
\|\neg\phi_1\|_{\mathcal{V}} & = \mathfrak{S} - \|\phi_1\|_{\mathcal{V}} \\
\|\phi_1 \wedge \phi_2\|_{\mathcal{V}} & = \|\phi_1\|_{\mathcal{V}} \cap \|\phi_2\|_{\mathcal{V}} \\
\|\langle a\rangle_c \phi_1\|_{\mathcal{V}} & = \{(R,t) \in \mathfrak{S} \mid \exists r \in R.\ t \leq r \wedge (\mathfrak{X}, r) \in \|\phi_1\|_{\mathcal{V}}\} \\
\|\langle a\rangle_{nc} \phi_1\|_{\mathcal{V}} & = \{(R,t) \in \mathfrak{S} \mid \exists r \in R.\ t \ominus r \wedge (\mathfrak{X}, r) \in \|\phi_1\|_{\mathcal{V}}\} \\
\|\langle \otimes\rangle \phi_1\|_{\mathcal{V}} & = \{(R,t) \in \mathfrak{S} \mid \exists M \in \mathcal{X}.\ M \sqsubseteq R \wedge (M, t) \in \|\phi_1\|_{\mathcal{V}}\}
\end{array}$$

such that $a = \delta(r)$ and $\mathfrak{X}$ is the maximal set at $\tau(r)$. Also, given the usual restriction on free occurrences of variables imposed in order to obtain monotone operators in the complete lattice $(2^{\mathfrak{S}}, \subseteq)$, the powerset lattice of $\mathfrak{S}$, it is possible to define the denotation of the least fixpoint operator in the standard way:

$$\|\mu Z.\phi\|_{\mathcal{V}} = \bigcap\{Q \subseteq \mathfrak{S} \mid \|\phi\|_{\mathcal{V}[Z:=Q]} \subseteq Q\}$$

where $\mathcal{V}[Z := Q]$ is the valuation $\mathcal{V}'$ which agrees with $\mathcal{V}$, with $\mathcal{V}'(Z) = Q$. ◁

The semantics of the dual Boolean, modal, and fixpoint operators is given in the usual way. Finally, the *satisfaction* relation $\models$ is defined in the standard way: given a process P and a formula ϕ, we have that $P \models \phi$ iff $P \in \|\phi\|$.

So much so far of formal definitions. Let us now present a few examples.

Example 3.7 (Critical regions) Let $\phi = \nu Z.\,[\otimes]\,([wrA]\,\mathrm{ff} \vee [wrB]\,\mathrm{ff}) \wedge [-]\,Z$. This $\mathbb{L}_\mu$ formula says that *always* it is impossible for a system to execute in parallel two actions wrA and wrB. If, for instance, 'wrA' and 'wrB' refer to actions—of two parallel processes A and B—that modify (write) a particular critical region, then one can be sure that the access to such a critical section is safe if the temporal specification ϕ is satisfied by the system. ◁

Example 3.8 (Secure synchronisation) When using a process algebra like CCS, one would like to specify the property that whenever some action, say a, is executed there exists a parallel action $\bar{a}$ that can also be executed in order for them to synchronise. Then, one can be sure that the temporal property that whenever some component of a system performs a always another component is ready to respond with $\bar{a}$ is satisfied iff $\phi = \nu Z.[a]_c\,\langle\bar{a}\rangle_{nc}\mathrm{tt} \wedge [-]\,Z$ holds. ◁

Example 3.9 (Causal reachability) Let ϕ be the following reachability logical formula: $\phi = [h]\,\mu Z.([\otimes]\,(\langle a\rangle_c \mathrm{tt} \wedge \langle b\rangle_c \mathrm{tt})) \vee \langle -\rangle_c Z$. This $\mathbb{L}_\mu$ formula expresses that after executing any initial action h (if any) there exists at least one execution of causally dependent actions such that *eventually* two actions a and b can be executed in parallel (since they must be independent). ◁

Example 3.10 (Response properties) Another interesting property, which requires the combination of least and greatest fixpoints, is a response to a request of a service. Suppose that whenever a system component (an agent) executes an a-labelled action, there must exist a sequential system component (another agent) that eventually executes a b-labelled action, which is meant to be the allocation of the service being requested through a. Such a property is expressed in $\mathbb{L}_\mu$ with the following fixpoint formula: $\phi = \nu Z.\,[a]\,(\mu Y.\langle b\rangle_c \mathrm{tt} \wedge \langle -\rangle_c Y) \wedge [-]\,Z$. If, moreover, one wants to specify that b is necessarily reachable in all causal lines, i.e., that it is causally unavoidable, then ϕ can be modified as follows: $\phi = \nu Z.\,[a]\,(\mu Y.\langle b\rangle_c \mathrm{tt} \wedge [-]_c\, Y \wedge \langle -\rangle_c \mathrm{tt}) \wedge [-]\,Z$. ◁

4 Nondeterminism in Causal and Concurrent Processes

We now study the logical equivalences induced by different $\mathbb{L}_\mu$ sublogics by comparing them with, arguably, two of the most important bisimulation equivalences for concurrency, namely with strong bisimilarity (SB) and with history-preserving bisimilarity (HPB). We do so either by *syntactically* restricting the interplay between concurrency and conflict, and concurrency and causality, in the logical side, or by *semantically* restricting the degree of nondeterminism that is allowed in the models upon which the logic is interpreted.

Definition 4.1 ($\mathfrak{L}$ equivalence $\sim_{\mathfrak{L}}$) Given a logic $\mathfrak{L}$, two processes P and Q associated with two systems $\mathfrak{T}_1$ and $\mathfrak{T}_2$, respectively, are $\mathfrak{L}$-equivalent, $P \sim_{\mathfrak{L}} Q$, if and only if, for every $\mathfrak{L}$ formula ϕ in $\mathfrak{F}_{\mathfrak{L}}$, $P \models^{\mathfrak{T}_1} \phi \Leftrightarrow Q \models^{\mathfrak{T}_2} \phi$, where $\mathfrak{F}_{\mathfrak{L}}$ is the set of all fixpoint-free closed formulae of $\mathfrak{L}$. ◁

Remark 4.2 (Equivalence in logical form) The previous definition delivers a logical, abstract notion of equivalence that can be used across different models for concurrency, i.e., tailored to be model independent. With this logical notion of equivalence two systems $\mathfrak{T}_1$ and $\mathfrak{T}_2$, possibly of different kinds, are equivalent with respect to some equivalence $\sim_{\mathfrak{L}}$ if, and only if, their associated process spaces cannot be differentiated by any $\mathfrak{L}$-logical formula. ◁

Recall that in order to obtain an exact match between finitary modal logic and bisimulation, all models under consideration are image-finite [14], i.e., of finite branching. Moreover, since the semantics of $\mathbb{L}_\mu$ is based on action labels, we only consider models without 'auto-concurrency' [22], a common restriction when studying either modal logics or equivalences for (labelled) partial order models. More precisely, auto-concurrency is the phenomenon by which multiple instances of various concurrent transitions are equally labelled. In other words, auto-concurrency can be seen as nondeterminism inside a set of independent transitions. In many cases auto-concurrency is regarded as an undesirable situation on partial order models since it can be easily avoided in practice and makes slightly counter-intuitive the analysis of behavioural properties of concurrent processes with partial order semantics. In fact, on finite systems, auto-concurrency is formally, but not actually, a further restriction since any bounded branching

system that has auto-concurrency can be effectively converted into a system that does not have auto-concurrency by a suitable relabelling of auto-concurrent transitions without changing the concurrent behaviour of the model.

Let us now turn to the study of some syntactic fragments of $\mathbb{L}_\mu$. They are called the *natural* syntactic fragments of $\mathbb{L}_\mu$ because such sub-logics arise as the languages where the dualities between concurrency and causality as well as concurrency and conflict are syntactically manipulated. As we will see, the equivalences induced by some of such fragments coincide with known bisimilarities for interleaving and for causal behaviour, in the latter case provided that the degree of nondeterminism in the systems is restricted. We start by analysing a syntactic fragment of $\mathbb{L}_\mu$ that is oblivious to causal information.

The Modal μ-Calculus. The first sublogic is obtained from $\mathbb{L}_\mu$ by disabling the sensitivity of this logic to both dualities. On the one hand, insensitivity to the duality between concurrency and causality can be captured by considering only modalities without subscript, i.e., HML modalities, using the abbreviations given previously. On the other hand, insensitivity to the duality between concurrency and conflict can be captured by considering the $[\langle\otimes\rangle]$-free $\mathbb{L}_\mu$ sublanguage, where $[\langle\otimes\rangle]$ means $\{\langle\otimes\rangle, [\otimes]\}$. The resulting logic has the same syntax of $\mathcal{L}_\mu$. This fragment is the purely-modal $[\langle\otimes\rangle]$-free fragment of $\mathbb{L}_\mu$. Then, it is easy to show:

Proposition 4.3 *The syntactic purely-modal $[\langle\otimes\rangle]$-free fragment of $\mathbb{L}_\mu$ is semantically equivalent to the modal μ-calculus.*

Remark 4.4 Regarding logical and concurrent equivalences, it is now easy to see that strong bisimilarity, SB, the equivalence induced by modal logic, is captured by the fixpoint-free fragment of this sublogic, which we denote by $\sim_{\mathcal{L}_\mu}$. Hence, the *logical correspondence* $\sim_{\mathcal{L}_\mu} \equiv \sim_{SB}$ follows from Proposition 4.3 and the fact that modal logic characterises bisimulation on finite models. $\triangleleft$

The Trace Modal μ-Calculus. The second sublogic we consider is the 'trace modal mu-calculus', $\mathcal{L}_\mu^\otimes$. This logic is obtained from $\mathbb{L}_\mu$ by allowing only the recognition of the duality between concurrency and conflict using $\langle\otimes\rangle$. The syntax of $\mathcal{L}_\mu^\otimes$ is: $\phi ::= Z \mid \neg\phi_1 \mid \phi_1 \wedge \phi_2 \mid \langle a\rangle\phi_1 \mid \langle\otimes\rangle\phi_1 \mid \mu Z.\phi_1$. We write $\sim_{\mathcal{L}_\mu^\otimes}$ for the equivalence induced by this sublogic. Observe that $\mathcal{L}_\mu^\otimes$ is more expressive than $\mathcal{L}_\mu$ in because $\mathcal{L}_\mu^\otimes$ includes $\mathcal{L}_\mu$ and can differentiate concurrency from nondeterminism—take, e.g., formula $\langle\otimes\rangle(\langle a\rangle\mathrm{tt} \wedge \langle b\rangle\mathrm{tt})$ and a system where $a \parallel b$. The following counter-example shows that $\sim_{\mathcal{L}_\mu^\otimes}$ and $\sim_{HPB}$ do not coincide.

Proposition 4.5 *Neither $\sim_{HPB} \subseteq \sim_{\mathcal{L}_\mu^\otimes}$ nor $\sim_{\mathcal{L}_\mu^\otimes} \subseteq \sim_{HPB}$.*

Proof. The two systems at the top in Figure 2 are HPB and yet can be distinguished by the formula $\phi = \langle\otimes\rangle(\langle a\rangle\langle c\rangle\mathrm{tt} \wedge \langle b\rangle\langle d\rangle\mathrm{tt})$. On the other hand, the systems at the bottom are not HPB and cannot be differentiated by any $\mathcal{L}_\mu^\otimes$ formula. This can be verified by exhaustively checking $\mathcal{L}_\mu^\otimes$ formulae in state $\circ$. $\square$

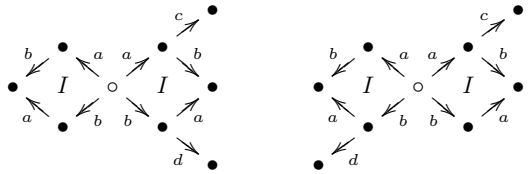

Fig. 2. Counter-examples of the coincidence between $\sim_{HPB}$ and $\sim_{\mathcal{L}_\mu^\otimes}$

There is a fundamental reason for the mismatch between $\sim_{HPB}$ and $\sim_{\mathcal{L}_\mu^\otimes}$. It has to do with a special 'sharing' of resources between some of the transitions in the model. This special kind of sharing of resources is characterised by a phenomenon called *confusion*, which is a concept first studied in the theory of concurrency with nets, and thus, it is useful to think of it directly on nets.

Confusion can be symmetric or asymmetric. Roughly speaking, it is a phenomenon that arises between at least three different actions, say between t_1, t_2, and t_3. In the symmetric case, two of them are independent, e.g., t_1 par t_2, and at the same time are in conflict with the third action, i.e., ${}^\bullet t_1 \cap {}^\bullet t_3 \neq \emptyset$ and ${}^\bullet t_2 \cap {}^\bullet t_3 \neq \emptyset$. On the other hand, in the asymmetric case, two of the actions are independent, e.g., t_1 par t_2 as before, whereas the third one is in conflict with one of the independent actions, say with t_1, and causally depends on the other, i.e., ${}^\bullet t_1 \cap {}^\bullet t_3 \neq \emptyset$ and $t_2^\bullet \cap {}^\bullet t_3 \neq \emptyset$, respectively. Confusion is important because, although it is undesirable when analysing concurrent behaviour, it is also "inherent to any reasonable net model of a mutual exclusion module" [28]. Confusion is also present when modelling race conditions in concurrent and distributed systems with shared memory models. These facts show the ubiquity of this phenomenon when analysing real-life models of communicating concurrent systems. Although confusion is a natural concept in net theory, it can also be defined for TSI and event structures, though in these cases the definition is more complicated because it involves sets of transitions and sets of events, respectively, rather than single actions as in the Petri net case. Confusion appears in the two counter-examples shown in Figure 2, in both cases in its asymmetric variant. The problem is that both $\sim_{HPB}$ and $\sim_{\mathcal{L}_\mu^\otimes}$ can recognise some forms of confusion, but not all of them. However, there is a class of nets called *free-choice*

where confusion never arises, and for which coincidence results between $\sim_{HPB}$ and $\sim_{\mathcal{L}_\mu^\otimes}$ may be possible. Define a free-choice system as follows:

Definition 4.6 A system $\mathfrak{T}$ is *free-choice* iff whenever there are three transitions t_1, t_2, and t_3 such that t_1 is in conflict with t_2 and any of them is independent of t_3, i.e., either $(t_1, t_3) \in I$ or $(t_2, t_3) \in I$, then there must be two transitions t_4 and t_5, where $t_1 \sim t_4$ and $t_2 \sim t_5$, such that t_4 and t_5 are also in conflict, and t_3 is independent of both t_4 and t_5. $\triangleleft$

Informally, the previous definition means that a choice, i.e., a conflict, cannot be globally affected by the execution of a concurrent transition, since equivalent choices are always possible both before and after that. These facts, along with the observations made before, led us to conjecture that, in fact, the statement $\sim_{\mathcal{L}_\mu^\otimes} \equiv \sim_{HPB}$ holds on free-choice systems without auto-concurrency.

Now, let us move to the study of a modal logic that is sensitive to the causal information embodied in partial order systems. In particular, it will be shown that for some classes of systems the *local* duality between concurrency and causality is good enough to capture the full (more complex and discriminating) notion of *global* causality defined by history-preserving bisimilarity, HPB.

From Local to Global Causality

In this section we show the most interesting coincidence result of the equivalence induced by one of the sublogics of $\mathbb{L}_\mu$ with a bisimilarity for partial order systems. The result holds for a class of systems whose expressive power lies between that of so-called 'free-choice' nets [8] and that of safe nets, as before with the usual restrictions to systems that have *finite branching* and *no auto-concurrency*.

The coincidence result is with respect to HPB. This equivalence is considered to be the standard bisimulation equivalence for causality since it fully captures the interplay between branching and causal behaviour. The interesting feature of this coincidence result is that HPB provides a *global* notion of causality whereas the logic we are about to study induces a *local* one, as shown later on. Then, the question we answer here is that of the class of systems for which *local causality* fully captures the standard notion of *global causality*. Such an answer is given by the following modal logic, which is a refinement of HML.

The Causal Modal μ-Calculus. The third sublogic to be considered is the 'causal modal mu-calculus', $\mathcal{L}_\mu^c$. This sublogic is obtained from $\mathbb{L}_\mu$ by allowing only the recognition of the duality between concurrency and causality throughout the modal operators on transitions of $\mathbb{L}_\mu$. The syntax of this syntactic fragment is the following: $\phi ::= Z \mid \neg\phi_1 \mid \phi_1 \wedge \phi_2 \mid \langle a \rangle_c \phi_1 \mid \langle a \rangle_{nc} \phi_1 \mid \mu Z.\phi_1$.

Clearly, $\mathcal{L}_\mu^c$ is also more expressive than $\mathcal{L}_\mu$ in partial order models because of the same reasons given for $\mathcal{L}_\mu^\otimes$. The naturality of $\mathcal{L}_\mu^c$ for expressing causal properties is demonstrated by the equivalence it induces, written as $\sim_{\mathcal{L}_\mu^c}$, which coincides with $\sim_{HPB}$, the standard bisimulation equivalence for causal processes,

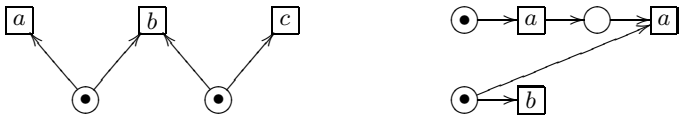

Fig. 3. Confusion: the Petri net on the left has symmetric confusion and the Petri net on the right has asymmetric confusion. In both cases it is deterministic.

when interpreted over systems without auto-concurrency where any 3-tuple of transitions (t_1, t_2, t_3) in *confusion is deterministic*. Let us define confusion, a ternary relation on transitions, and its deterministic case over labelled models.

Definition 4.7 (Confusion) Let cfs be a ternary relation on transitions of a system $\mathfrak{T}$ such that $(t_1, t_2, t_3) \in \mathsf{cfs}$ iff $t_1 \otimes t_2$ and either $t_1 \# t_3$ and $t_2 \# t_3$ (the symmetric case) or $t_1 \leq t_3$ and $\exists r_2.\ t_2 \sim r_2 \wedge r_2 \# t_3$ (the asymmetric case). Naturally, a tuple $(t_1, t_2, t_3) \in \mathsf{cfs}$ is *deterministic* iff it holds that if two different transitions are in conflict, then they are not equally labelled. $\triangleleft$

Observe that if we consider systems with deterministic confusion and without auto-concurrency, then the following statement immediately follows:

Fact 4.8 Let $\mathfrak{T}$ have deterministic confusion and no auto-concurrency. Then, a tuple $(t_1, t_2, t_3) \in \mathsf{cfs}$ is deterministic iff either the three transitions have different labels (symmetric case) or if $\delta(t_1) = \delta(t_3)$ then $t_1 \leq t_3$ (asymmetric case). $\triangleleft$

Perhaps because confusion is a concept that originated in net theory, it is very easy to depict confusion using Petri nets. Figure 3 shows the two simplest nets featuring confusion, both in their symmetric and asymmetric variants.

Any Petri net that has confusion must have either of these two nets as a subsystem. The statement equivalently holds for TSI and event structures if considering, respectively, the TSI and event structure models corresponding to such nets. This property allows one to define a class of systems for which the logical equivalence induced by $\mathcal{L}^c_\mu$ captures HPB. Such a class contains those systems without auto-concurrency that either are *free-choice* or have a deterministic confusion relation. Thus, let us now define the class of free-choice systems. For simplicity, we do so indirectly via the standard definition of free-choice nets, which is well-known in the literature [8]. Formally, let $\mathcal{N}$ be a net. We say that $\mathcal{N}$ is *free-choice* iff for all $s \in P$ we have that $\mid s^\bullet \mid \leq 1$ or $\forall t \in s^\bullet.\mid {}^\bullet t \mid = 1$.

A free-choice Petri net is a free-choice net with an initial marking. A free-choice event structure is an event structure unfolding [21] of a free-choice Petri net and a free-choice TSI is the TSI model obtained from a free-choice Petri net. We are almost ready to show that the two equivalences $\sim_{\mathcal{L}^c_\mu}$ and $\sim_{HPB}$ coincide over a class of concurrent systems that we call with *deterministic communication*, and denote by Ξ. The reason of this name is that non-trivial communication mechanisms require confusion to be properly modelled (in a noninterleaving framework), and in what follows we will consider systems with *deterministic confusion*—and hence with deterministic communication mechanisms.

Definition 4.9 (Ξ systems) The family of Ξ systems is the class of systems without auto-concurrency that has a deterministic confusion relation. $\triangleleft$

Remark 4.10 (Expressivity) The family of Ξ systems contains, at least, the following classes of models (by definition, without auto-concurrency and with a deterministic conflict relation): Moore and Mealy machines, labelled graphs, synchronous and asynchronous products of sequential processes, and free-choice systems (as their confusion relation is empty—trivially deterministic). More importantly, via a natural re-labelling of actions, Ξ systems can model mutual exclusion protocols and the usual synchronization mechanisms of some process calculi. Some examples to illustrate systems of these kinds are given later. $\triangleleft$

Now, back to the issue of relating HPB and $\sim_{\mathcal{L}^c_\mu}$, the proof that HPB and $\sim_{\mathcal{L}^c_\mu}$ coincide for the class of Ξ systems goes by showing that the two inclusions $\sim_{HPB} \subseteq \sim_{\mathcal{L}^c_\mu}$ and $\sim_{\mathcal{L}^c_\mu} \subseteq \sim_{HPB}$ hold. In fact, the first inclusion holds for any class of systems while the second one requires the restriction to Ξ systems. This last observation shows that, in fact, the logical equivalence $\sim_{\mathcal{L}^c_\mu}$ for (local) causal behaviour is, in general, weaker than history-preserving bisimilarity (indeed, HPB makes more demanding global requirements to hold), whenever the systems under consideration do not have deterministic confusion.

Lemma 4.11 (Logical soundness) $\sim_{HPB} \subseteq \sim_{\mathcal{L}^c_\mu}$.

Proof. This inclusion can be shown by induction on $\mathcal{L}^c_\mu$ formulae, which we denote by $\mathfrak{F}_{\mathcal{L}^c_\mu}$. Let $\mathfrak{T}_1$ and $\mathfrak{T}_0$ be two systems and $P \in \mathfrak{S}_1$ and $Q \in \mathfrak{S}_0$ two processes that belong to the process spaces $\mathfrak{S}_1$ and $\mathfrak{S}_0$ associated with $\mathfrak{T}_1$ and $\mathfrak{T}_0$, respectively. If we have that $P \sim_{HPB} Q$ then for all $\phi \in \mathfrak{F}_{\mathcal{L}^c_\mu}$ we have that $P \models^{\mathfrak{T}_1}_{\mathcal{V}_1} \phi \Leftrightarrow Q \models^{\mathfrak{T}_0}_{\mathcal{V}_0} \phi$ given two models $\mathfrak{M}_1 = (\mathfrak{T}_1, \mathcal{V}_1)$ and $\mathfrak{M}_0 = (\mathfrak{T}_0, \mathcal{V}_0)$. Since $\mathcal{L}^c_\mu$ only considers maximal sets, the process $P = (p, t)$ (resp. the process $Q = (q, t)$) is actually a binary tuple in $S_1 \times \mathfrak{A}_1$ (resp. in $S_0 \times \mathfrak{A}_0$) rather than a tuple in $\mathcal{X}_1 \times \mathfrak{A}_1$ (resp. in $\mathcal{X}_0 \times \mathfrak{A}_0$). Henceforth, let us write $\models$ instead of $\models^{\mathfrak{T}_i}_{\mathcal{V}_i}$, for $i \in \{0, 1\}$, since the models will be clear from the context.

The base case of the induction is when $\phi = \mathtt{tt}$ or when $\phi = \mathtt{ff}$ which is trivial since tt and ff are always true and false, respectively. Now, consider the cases for the Boolean operators $\wedge$ and $\vee$; first suppose that $\phi = \psi_1 \wedge \psi_2$ and assume that the result holds for both ψ_1 and ψ_2. By the definition of the satisfaction relation $P \models \phi$ iff $P \models \psi_1$ and $P \models \psi_2$ iff by the inductive hypothesis $Q \models \psi_1$ and $Q \models \psi_2$, and hence, by the definition of the satisfaction relation $Q \models \phi$. The case for $\vee$ is similar.

Now, consider the cases for the four modalities. First, suppose $\phi = [a]_{nc}\,\psi$ and $P \models \phi$. Therefore, for any $P' = (p', t')$, such that $a = \delta(t')$ and $P \xrightarrow{a} P'$ and $t \ominus t'$, it follows that $P' \models \psi$. Now, let $Q \xrightarrow{a} Q'$ such that $a = \delta(t')$ and $t \ominus t'$ since the bisimulation must remain synchronous. Just to recall, synchrony in HPB means that the last transition chosen in $\mathfrak{T}_1$ (resp. in $\mathfrak{T}_0$) is concurrent with the former transition also chosen in $\mathfrak{T}_1$ (resp. in $\mathfrak{T}_0$) iff the same pattern holds in the last two transitions chosen in $\mathfrak{T}_0$ (resp. in $\mathfrak{T}_1$), and moreover the

two sequences of transitions (i.e., runs) that are generated in this way are the linearisations of isomorphic posets. So, as we know that for some P' there is a $P \xrightarrow{a} P'$, where $t \ominus t'$, and by the inductive hypothesis $P' \sim_{HPB} Q'$, then $Q' \models \psi$, where $t \ominus t'$, and so by the definition of the satisfaction relation $Q \models \phi$. The case when Q satisfies ϕ is symmetric, and the case when $\phi = [a]_c \psi$ is similar (only changing $\ominus$ for $\leq$). The cases for the operators $\langle a \rangle_c$ and $\langle a \rangle_{nc}$ are analogous. $\square$

In order to show the second inclusion, namely $\sim_{\mathcal{L}^c_\mu} \subseteq \sim_{HPB}$, we first require some lemmas that characterise the set of runs that can be identified by $\mathcal{L}^c_\mu$ in a partial order model. More specifically, a proof that if two systems $\mathfrak{T}_0$ and $\mathfrak{T}_1$ are $\mathcal{L}^c_\mu$-equivalent, then for each run of one of the systems there exists a 'locally synchronous' run (which is defined below) in the other system. Then, one can use this result to show that for any two Ξ systems $\mathfrak{T}_0$ and $\mathfrak{T}_1$ such that $\mathfrak{T}_0 \sim_{\mathcal{L}^c_\mu} \mathfrak{T}_1$, each pair of locally synchronous runs is moreover induced by two isomorphic posets (the global condition of HPB), and hence, $\mathfrak{T}_0 \sim_{HPB} \mathfrak{T}_1$ must hold too since, in such a case, the pair of runs is synchronous. The formalisation of the observations just made is given by Lemmas 4.12, 4.13, and 4.14, presented below.

Recall the definition of runs and of synchronous runs given before, and let $\pi_0 \in \Pi_{\mathfrak{T}_0}$ and $\pi_1 \in \Pi_{\mathfrak{T}_1}$ be two runs of two systems $\mathfrak{T}_0$ and $\mathfrak{T}_1$, and u, v two transitions. A pair of runs $(\pi_0.u, \pi_1.v)$ is inductively defined as *locally synchronous* iff (π_0, π_1) is locally synchronous and $(\varrho(\pi_0), u) \in I_0 \Leftrightarrow (\varrho(\pi_1), v) \in I_1$, where I_0 and I_1 are the independence relations of $\mathfrak{T}_0$ and $\mathfrak{T}_1$, respectively. By definition, the pair of empty runs (ϵ, ϵ) is locally synchronous. Note that the definitions of locally synchronous runs and synchronous runs is quite similar; the only difference is that synchronous runs must be the linearisation of isomorphic posets whereas locally synchronous runs need not be. Then, we can show:

Lemma 4.12 (Logic-based local causality) *Let $\mathfrak{T}_0$ and $\mathfrak{T}_1$ be two systems and $\Pi_{\mathfrak{T}_0}$ and $\Pi_{\mathfrak{T}_1}$ their sets of runs. If $\mathfrak{T}_0 \sim_{\mathcal{L}^c_\mu} \mathfrak{T}_1$ then for each $\pi_0 \in \Pi_{\mathfrak{T}_0}$ (resp. $\pi_1 \in \Pi_{\mathfrak{T}_1}$) there exists a run $\pi_1 \in \Pi_{\mathfrak{T}_1}$ (resp. $\pi_0 \in \Pi_{\mathfrak{T}_0}$) such that the pair of runs (π_0, π_1) is locally synchronous.*

Proof. The proof goes by a contradiction argument. Suppose that for all ϕ in $\mathfrak{F}_{\mathcal{L}^c_\mu}$ we have that $P \models \phi \Leftrightarrow Q \models \phi$ and there exists a run in one of the systems that is not locally synchronous to any of the runs in the other system. The case where P and Q are the initial processes of $\mathfrak{T}_0$ and $\mathfrak{T}_1$, respectively, is trivially false since, by definition, the pair of empty runs (ϵ, ϵ) is locally synchronous.

Then, suppose now that (π_0, π_1) is locally synchronous and that P and Q are two processes reached, respectively, in $\mathfrak{T}_0$ and in $\mathfrak{T}_1$ after following π_0 and π_1 in each system (starting from their initial processes). Additionally, suppose that there exists a transition u in one of the systems, say in $\mathfrak{T}_0$, such that there is no transition v in the other system for which the pair of runs $(\pi_0.u, \pi_1.v)$ is locally synchronous. Note that P and Q are strongly bisimilar, since $\mathcal{L}^c_\mu$ includes $\mathcal{L}_\mu$, and thus, the case in which a process can perform a transition (regardless of its label) and the other cannot do so is impossible as this contradicts the hypothesis that $P \sim_{SB} Q$.

So, suppose that for some transition u with label a, $P = (p, \varrho(\pi_0)) \xrightarrow{a} P' = (p', u)$ and $\varrho(\pi_0) \ominus u$ (resp. $\varrho(\pi_0) \leq u$), but for all transitions v such that $a = \delta(v)$ it holds that $Q = (q, \varrho(\pi_1)) \xrightarrow{a} Q' = (q', v)$ and $\varrho(\pi_1) \leq v$ (resp. $\varrho(\pi_1) \ominus v$) only. However, we know that, by hypothesis, $P \sim_{\mathcal{L}^c_\mu} Q$ and so, it must be true that if $P \models \langle a \rangle_{nc} \phi$ (resp. if $P \models \langle a \rangle_c \phi$) then $Q \models \langle a \rangle_{nc} \phi$ (resp. if $Q \models \langle a \rangle_c \phi$), which is a contradiction. Thus, one must be able to match pairs of independent transitions in one of the systems whenever the same happens in the other system for all pairs of processes P and Q satisfying that $P \sim_{\mathcal{L}^c_\mu} Q$. □

The previous lemma ensures that if two systems satisfy the same set of $\mathcal{L}^c_\mu$ formulae, then, *locally, they have the same causal behaviour*. However, in order to show that, globally, they also have the same causal behaviour, one needs some additional information, which is given by the following lemma.

Lemma 4.13 *Let $\mathfrak{T}$ be a Ξ system whose conflict relation is* cfs *and let $\pi \in \Pi_{\mathfrak{T}}$. If, after executing the run π in $\mathfrak{T}$, there are two different enabled transitions u and v such that $\delta(u) = \delta(v)$, then the following two statements hold:*

1. *$u \# v$.*
2. *For every $a \in \Sigma$, there is at most one t in π such that $\tau(t) = \sigma(u') = \sigma(v')$ and $a = \delta(t)$, for which $t \leq u'$ and $t \leq v'$ and $u \sim u'$ and $v \sim v'$.*

Proof. In the same order as in the statement of the lemma:

1. Because there is no auto-concurrency.
2. As the confusion relation is deterministic there is no $c \in$ cfs such that both u and v belong to c; moreover, if there is transition t' different from t such that $\tau(t') = \sigma(u') = \sigma(v')$, then t and t' must be independent, and because of no auto-concurrency, it cannot be the case that $\delta(t') = a = \delta(t)$. □

Finally, the following lemma ensures that for the class of Ξ systems, the notion of locally synchronous runs (associated with *local causality*) is powerful enough to capture the stronger notion of synchronous runs (associated with *global causality*), so long the two systems satisfy the same set of $\mathcal{L}^c_\mu$ formulae.

Lemma 4.14 (System-based local causality) *Let $\mathfrak{T}_0$ and $\mathfrak{T}_1$ be Ξ systems whose sets of runs are $\Pi_{\mathfrak{T}_0}$ and $\Pi_{\mathfrak{T}_1}$. If for each $\pi_0 \in \Pi_{\mathfrak{T}_0}$ (resp. $\pi_1 \in \Pi_{\mathfrak{T}_1}$) there exists some $\pi_1 \in \Pi_{\mathfrak{T}_1}$ (resp. $\pi_0 \in \Pi_{\mathfrak{T}_0}$) such that (π_0, π_1) is locally synchronous, then (π_0, π_1) is, moreover, synchronous.*

Proof. The proof is based on the fact that if (π_0, π_1) is a locally synchronous pair, then the posets induced by such locally synchronous runs induce isomorphic posets if the systems are Ξ, and hence, the pair of runs is also 'globally' synchronous. We proceed by induction on the length of runs. The base case, i.e., when the pair of runs is $(\pi_0, \pi_1) = (\epsilon, \epsilon)$, is trivial since in this case the two posets are empty.

Then, for the induction step, suppose that there is a non-empty run π_0 of size k that is locally synchronous to some run π_1; moreover, suppose that π_0 and π_1

induce isomorphic posets. We show that there is not a run $\pi_0.u$ which induces a poset that is not isomorphic to any of the posets induced by those runs $\pi_1.v$ for which the pairs of extended runs $(\pi_0.u, \pi_1.v)$ are locally synchronous.

Due to the definition of Ξ systems, one can consider the following three cases: (1) the transition u is the instance of a net action e such that $|\ {}^\bullet e\ | > 1$ and u is not in the conflict relation of $\mathfrak{T}_0$; (2) the transition u is the instance of a net action e such that for some net place s we have that $e \in s^\bullet$ and $\forall e \in s^\bullet.|\ {}^\bullet e\ | = 1$ and u is not in the conflict relation of $\mathfrak{T}_0$; or (3) the transition u is an instance of a net action of either type and is in the conflict relation of $\mathfrak{T}_0$.

For the first case, let π_0 be any run such that $\varrho(\pi_0) \leq u$. By hypothesis we have that the posets induced by π_0 and π_1 are isomorphic, that u depends only on one transition (namely, on $\varrho(\pi_0)$), and that $\varrho(\pi_1) \leq v$ as well. Then, the only possibility for this case to fail is if v, unlike u, causally depends on more than only one transition (since it already depends on $\varrho(\pi_1)$). Suppose this could happen; then, there is at least one transition e_j in π_1 on which v also causally depends and that is independent of $\varrho(\pi_1)$. Then there must exist a run π_1^- of length $k-1$ that do not contain e_j and where $\varrho(\pi_1^-) \leq v'$ for some v' such that $\delta(v') = \delta(v)$. Since v and v' cannot be two instances of the same net action, then they must be in conflict (because of no auto-concurrency) and moreover belong to some tuple c of the confusion relation cfs of $\mathfrak{T}_1$, which is impossible since Ξ systems have a deterministic confusion relation. As a consequence any transition u of this kind can be matched only by a transition v which, due to Lemma 4.13, for every label extends a unique transition of any run, keeping the two extended runs $\pi_0.u$ and $\pi_1.v$ not only locally synchronous but also globally synchronous.

For the second case, suppose that u depends on a set $\{e_0^i, ..., e_0^k, ..., e_0^m\}$ of elements of the poset induced by π_0, i.e., $\forall e \in \{e_0^i, ..., e_0^k, ..., e_0^m\}.(e, u) \notin I_0$, and there is at least one e_0^k that was related to some e_1^k of π_1 while constructing the two locally synchronous runs, i.e., $e_0^k = \pi_0(k)$ and $e_1^k = \pi_1(k)$ for some natural number k, but that is not extended in π_0 with respect to u as e_1^k is extended in π_1 with respect to v, i.e., which makes the two induced posets not isomorphic because $(e_0^k, u) \notin I_0$ whereas $(e_1^k, v) \in I_1$.

For the same reasons given in the first case, v cannot depend on only one transition in π_1. On the contrary it must depend on at least two transitions, one of which must have the same label as e_0^k and e_1^k; let e_1^n be such a transition. As in the first case, w.l.o.g., the other transition can be $\varrho(\pi_1)$. Then, we have that v causally depends on e_1^n and is independent of e_1^k, which is independent of e_1^n. But this is impossible since $\delta(e_1^n) = \delta(e_1^k)$ and there is no auto-concurrency. Therefore, both runs must be extended in a synchronous way in this case as well.

Finally, for the third case notice that the arguments given before apply here as well, regardless of the kind of transition under consideration since the two properties in the former cases still hold: on the one hand, any two transitions equally labelled are always in conflict and causally depend (locally) on only one transition of any run; and, on the other hand, whenever is enabled a transition that is an instance of a net action whose preset is not a singleton, then that transition is the only one enabled with such a label.

As a consequence, any transition v must extend the poset induced by π_1 in the same way as u extends the poset induced by π_0, i.e., $\forall k \in \{1, ..., \mid \pi_0 \mid\}$ one has that $(\pi_0(k), u) \notin I_0$ iff $(\pi_1(k), v) \notin I_1$, making the two posets isomorphic in all cases and for all pairs (π_0, π_1) of locally synchronous runs of any length. $\square$

Informally, one can say that the arguments in the proof just given go through because any 'extra-concurrency' in one of the systems with respect to the other can be recognised since there is no auto-concurrency, and any 'extra-causality' can be recognised since, in Ξ systems, any two transitions enabled at the same time and equally labelled must be in conflict and causally depend on the same transitions in any run. Then, using Lemmas 4.12 to 4.14, we immediately get:

Corollary 4.15 (Logical completeness) $\sim_{\mathcal{L}^c_\mu} \subseteq \sim_{HPB}$ *on* Ξ *systems.*

It then follows from Lemma 4.11 and Corollary 4.15 the main result of the paper regarding the relationship between local and global causality, namely:

Theorem 4.16 (Full logical definability) $\sim_{\mathcal{L}^c_\mu} \equiv \sim_{HPB}$ *on* Ξ *systems.*

The previous theorem shows that for the class of Ξ systems the notion of 'local' causality defined by $\mathcal{L}^c_\mu$ captures the stronger notion of 'global' causality, which is captured by HPB in arbitrary classes of models of concurrency. Moreover, observe that the logical equivalence induced by $\sim_{\mathcal{L}^c_\mu}$ is *decidable*. This result immediately follows from Theorem 4.16 since HPB is decidable [16].

These last results can have interesting practical applications. For instance, the complexity of deciding whether two systems are HPB, i.e., that they posses the same causal properties, is EXPTIME-complete [16]; since verifying that two partial order systems satisfy the same set of $\mathcal{L}^c_\mu$ properties requires one to check only related 'localities' then the problem may be computationally easier.

4.1 A Bisimulation Game for $\mathbb{L}_\mu$ on Ξ Systems

The logic games for bisimulation presented in Section 2 provide a *first-order* power on the transitions that are picked when playing the game. In this section we present a bisimulation game that gives the players *higher-order* power on the sets of transitions in the game board. Since such games may be too powerful without restrictions, we consider higher-order games for bisimulation where this higher-order power is restricted to simple characteristic sets of transitions in the board. Moreover, since these games are intended to be used in the analysis of modal logics, then such a higher-order power is also restricted to a *local setting*.

In particular, in this section we define a higher-order logic game for bisimulation that helps us understand the bisimulation equivalence induced by $\mathbb{L}_\mu$, and how this logical equivalence relates to HPB, the best-known history-preserving bisimilarity for causal behaviour. To this end, we consider games with *monadic* second-order power on conflict-free sets of transitions, and show that such games can both capture the logical equivalence induced by $\mathbb{L}_\mu$ and be related in a natural way to the bisimulation game that characterises HPB.

Logical Correspondence. We now give a game-theoretical characterisation of the equivalence that $\mathbb{L}_\mu$ induces over Ξ Systems, by defining a characteristic bisimulation game for it. As we know that the game for $\mathbb{L}_\mu$ is at least as powerful as the HPB game over Ξ systems—since in such a case HPB is captured by a $\mathbb{L}_\mu$ fragment—then we can design a game that extends the (first-order) bisimulation game for HPB, but where the global requirement of HPB is replaced by a simpler local condition. Here we do so. Specifically, we show that this bisimulation game, which we call 'trace history-preserving bisimulation' (THPB) game, characterises the logical equivalence induced by $\mathbb{L}_\mu$. But, before presenting the game for $\mathbb{L}_\mu$, let us give a definition that is related to the role of support sets as locally identifiable sets of concurrent transitions, i.e., of conflict-free sets of transitions.

Definition 4.17 (Local HP isomorphism) Two sets of transitions R_1 and R_2 are said to be *locally history-preserving isomorphic* with respect to a pair of transitions (t_m, t_n) iff there exists a bijection $\mathcal{B}$ between them such that for every $(t_1, t_2) \in \mathcal{B}$, if $t_m \leq t_1$ (resp. $t_m \ominus t_1$) then $t_n \leq t_2$ (resp. $t_n \ominus t_2$). $\triangleleft$

Notice that any infinite play of an HPB game where Eve wins always induces a sequence of locally history-preserving isomorphic sets, where each set is a singleton. This follows from the fact that if this were not the case then Adam could win by choosing a transition in either R_1 or R_2 such that the runs in the bisimulation would no longer be synchronous. Recall that the local analyses (and definitions) we make in this section are supported by Lemma 4.14, namely, by the fact that the existence of locally synchronous runs implies the existence of (globally) synchronous runs—and vice-versa. We can now define THPB games.

Definition 4.18 (Trace history-preserving bisimulation games) Let the pair (π_1, π_2) be a configuration of the game $\mathcal{G}(\mathfrak{T}_1, \mathfrak{T}_2)$. The initial configuration of the game is (ϵ, ϵ). There are two players, Eve and Adam, and Adam always plays first and chooses where to play before using any rule of the game. The equivalence relation R_{THPB} is a trace history-preserving bisimulation (THPB), $\sim_{THPB}$, between $\mathfrak{T}_1$ and $\mathfrak{T}_2$ iff it is an HPB relation between $\mathfrak{T}_1$ and $\mathfrak{T}_2$ and:

- (Base case) The initial configuration (ϵ, ϵ) is in R_{THPB}.
- ($\sim_{THPB}$ rule). Before Adam chooses a transition using the $\sim_{HPB}$ rule, he can also restrict the set of available transitions by choosing either in π_1 or π_2 a maximal trace to be the new set of available choices. Then, Eve must choose a maximal set in the other component of the configuration.

$\mathfrak{T}_1 \sim_{THPB} \mathfrak{T}_2$ iff Eve has a winning strategy for the THPB game $\mathcal{G}(\mathfrak{T}_1, \mathfrak{T}_2)$. $\triangleleft$

Lemma 4.19 *If Eve has a winning strategy in the THPB game $\mathcal{G}(\mathfrak{T}_1, \mathfrak{T}_2)$, then $\mathfrak{T}_1 \sim_{\mathbb{L}_\mu} \mathfrak{T}_2$.*

Proof. By contradiction suppose that Eve has a winning strategy and $P \not\sim_{\mathbb{L}_\mu} Q$, where $P = (M, t)$ and $Q = (N, r)$ are two processes of $\mathfrak{T}_1$ and $\mathfrak{T}_2$, respectively. There are two cases. Suppose that Adam cannot make a move. This means that

both $P \models^{\mathfrak{T}_1} [-]\,\mathrm{ff}$ and $Q \models^{\mathfrak{T}_2} [-]\,\mathrm{ff}$ only, which is a contradiction. The other case is when Eve wins in an infinite play. Since $\mathcal{L}_\mu^c$ induces an HPB equivalence on Ξ systems, and the THPB game extends the HPB game in that case, then, w.l.o.g., we can consider only the case when the rule $\sim_{THPB}$ is necessarily played.

Then, let $P \models^{\mathfrak{T}_1} \langle\otimes\rangle\phi_1$ that, by hypothesis, is not satisfied by Q. By the satisfaction relation either M is already a maximal trace or there is a maximal trace M' such that $M' \sqsubseteq M$. Additionally, such a maximal trace cannot be recognised from N. However, this is not possible since Eve can always find such a support set by hypothesis. Thus, the only other possibility is that the support set can be constructed but a synchronous transition in it cannot be found. But this also leads to a contradiction because the support sets that Eve chooses are, additionally, history-preserving isomorphic to the ones that Adam chooses. Therefore all properties that include $\langle\otimes\rangle$ must be satisfied at this stage and the game has to proceed to the next round. However, since the play will continue forever, this holds for all reachable processes, and therefore, all formulae containing $\langle\otimes\rangle$ that are satisfied in P must also be satisfied in Q, which is again a contradiction. □

Corollary 4.20 (Soundness). *If $\mathfrak{T}_1 \not\sim_{\mathbb{L}_\mu} \mathfrak{T}_2$, then Adam has a winning strategy in the THPB game $\mathcal{G}(\mathfrak{T}_1, \mathfrak{T}_2)$.*

Lemma 4.21 (Completeness). *If $\mathfrak{T}_1 \sim_{\mathbb{L}_\mu} \mathfrak{T}_2$, then Eve has a winning strategy in the THPB game $\mathcal{G}(\mathfrak{T}_1, \mathfrak{T}_2)$.*

Proof. By constructing a winning strategy for Eve assuming that $\mathfrak{T}_1 \sim_{\mathbb{L}_\mu} \mathfrak{T}_2$. For the same reasons given previously, without loss of any generality, it is possible to consider only the case when Adam uses the $\sim_{THPB}$ rule.

Then, suppose that Adam is able to choose a maximal set M enabled at $P = (M, t)$, where P is a process in the stateless maximal process space $\mathfrak{S}$ associated with $\mathfrak{T}_1$. This implies that $P \models^{\mathfrak{T}_1} \phi$, where $\phi = \langle\otimes\rangle\phi_1$ for some formula ϕ with support set M. By the hypothesis, for some process $Q = (N, r)$ that is trace history-preserving bisimilar to P, it must be true that $Q \models^{\mathfrak{T}_2} \phi$ as well, and therefore Eve can choose a maximal set N which is the support set for ϕ in $Q = (N, r)$. Since we have that $P \sim_{\mathbb{L}_\mu} Q$, then M and N must be locally history-preserving isomorphic sets with respect to (t, r); otherwise, there would be a simple modal formula differentiating them.

Then Adam must choose an element of either set of transitions using the $\sim_{HPB}$ rule, say a transition $t' \in M$. But since M and N are locally history-preserving isomorphic sets with respect to (t, r), then it is always possible for Eve to find a transition $r' \in N$ that synchronises as t', forcing the game to proceed to the next round. The play, therefore, must either go on forever or stop because Adam cannot make a move. In either case Eve wins the game. The dual case is similar since Adam can always choose where to play, i.e., in which structure, before applying any rule of the game. □

These soundness and completeness results just presented give a full game-theoretical characterisation to the equivalence induced by $\mathbb{L}_\mu$.

Theorem 4.22 (Game abstraction) $\mathfrak{T}_1 \sim_{\mathbb{L}_\mu} \mathfrak{T}_2$ *iff Eve has a winning strategy for the THPB game* $\mathcal{G}(\mathfrak{T}_1, \mathfrak{T}_2)$*; conversely,* $\mathfrak{T}_1 \not\sim_{\mathbb{L}_\mu} \mathfrak{T}_2$ *iff Adam has a winning strategy for the THPB game* $\mathcal{G}(\mathfrak{T}_1, \mathfrak{T}_2)$*.*

And, thus, we also immediately obtain:

Corollary 4.23 $\sim_{\mathbb{L}_\mu} \equiv \sim_{THPB}$*.*

Remark 4.24 (Local vs. global causality) Before finishing this section it is important to note that even though the games for HPB (for global causality) and THPB (for local causality) look very similar, they are fundamentally different: whereas the former requires the construction of synchronous runs of transitions when playing the game, the former requires instead the construction of locally synchronous runs of conflict-free sets of transitions. In other words, the game for THPB checks that some sets of transitions causally depend on others, but only in a local way, which one hopes would be easier to check. ◁

5 Concurrent Systems with Deterministic Confusion

Many distributed and reactive processes can be modelled as Ξ systems, that is, as concurrent and multi-agent systems with no auto-concurrency and deterministic confusion—arguably, systems with a *deterministic* communication mechanism. In what follows, we will present three such process models of this kind.

5.1 Synchronous Products of Sequential Agents

Logics for multi-agent systems (MAS [32]) such as ATL [1] are interpreted over models that may be seen as synchronous products of sequential processes/agents. Because in this kind of systems all processes synchronise after performing one computation step, their communication mechanism is simple (yet powerful). Indeed, MAS are Ξ systems easily modelled using a TSI representation.

Specifically, in order to model a MAS, an explicit notion of an agent must be defined. Since in this case our basic model is based on transitions representing instances of actions in the system, such transitions should belong to uniquely defined agents. Thus, the following set of agents and corresponding "ownership" mapping on transitions are defined. Let Γ be a finite set of sequential agents and $\mathcal{A} : \mathfrak{A} \to \Gamma$ be a mapping that assigns transitions to agents. In this way, one knows the transitions that an agent can execute. However, as transitions in a partial order model represent instances of actions in a system, rather than actions that an agent can execute, a consistency restriction is defined so that all transitions that are instances of the same action are performed by the same agent and without interference of any other agents. This is captured by the constraints: (i) if $t_1 \sim t_2$ then $\mathcal{A}(t_1) = \mathcal{A}(t_2)$,[3] and (ii) if $t_1 \# t_2$ then $\mathcal{A}(t_1) = \mathcal{A}(t_2)$.

[3] Recall that $\sim$ is the equivalence relation on TSI transitions as defined earlier.

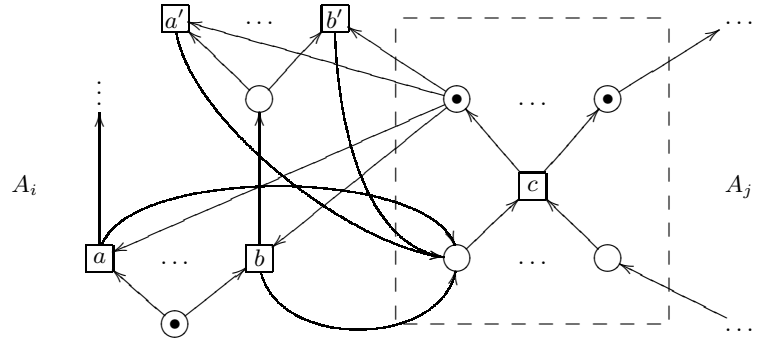

Fig. 4. Petri net representation of the communication mechanism in a synchronous product of sequential agents. Only one extra action c is required and $2 * |\Gamma|$ places.

Also, since modal logics can make distinctions between transitions with different labels, a consistency relation on labels of transitions should also be defined: if $\mathcal{A}(t_1) \neq \mathcal{A}(t_2)$ then $\delta(t_1) \neq \delta(t_2)$. Imposing these restrictions is equivalent to defining a distributed alphabet Σ (hence no auto-concurrency) over the set of agents Γ. A partial order model extended with these definitions and properties is called 'consistent' for MAS. Observe that since all agents synchronise after exactly one computation step, then asynchronous confusion does not arise. Synchronous confusion does not arise either: if two transitions are concurrent, then (because agents are sequential—i.e., single threaded) such transitions are owned by different agents and hence the statement if $\mathcal{A}(t_1) \neq \mathcal{A}(t_2)$ then $\delta(t_1) \neq \delta(t_2)$ holds; and because of (ii), any three transitions in confusion must belong to the same agent, which then leads to a contradiction. Diagrammatically, the synchronisation mechanism used in this kind of systems is as depicted in Figure 4.

We can add syntactic sugar to $\mathbb{L}_\mu$ to express properties of MAS. For instance, say a modality $\langle a \rangle^\alpha \phi$ is 'well-defined' for MAS iff for all transitions $t \in \mathfrak{A}$ if $a = \sigma(t)$ then $\alpha = \mathcal{A}(t)$. Modalities of the form $\langle K \rangle^\alpha \phi$, where $K \subseteq \Sigma$, can be defined by $\langle K \rangle^\alpha \phi = \bigvee_{a \in K} \langle a \rangle^\alpha \phi$ using the standard notation for modalities with sets of labels. We write $\langle - \rangle^\alpha \phi$ to mean the diamond modality on the set of labels restricted to those of α. Assume similar definitions for the other modalities.

Observe, based on the model in Figure 4, that the system also has deterministic confusion if each sequential process always eventually communicates with the other agents, as those other transitions would be local to each agent. In those kinds of cases, the following property, in Example 5.1, would be useful to check.

Example 5.1 (MAS) The $\mathbb{L}_\mu$ formula $\psi = [-]^\beta \langle - \rangle^\alpha_{nc} \mu Z. \phi \vee \langle - \rangle^\alpha_c Z$ says that there is a sequential agent α (the system) that can satisfy or achieve goal ϕ regardless the behaviour of an adversarial agent β (the environment). Informally, ψ says "whatever you (the environment) do, I (the system) can get to ϕ, though I may have to do some things that do not depend on what you just did." $\triangleleft$

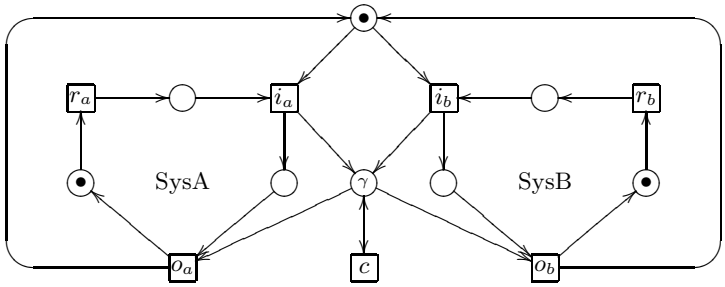

Fig. 5. A model of a mutual exclusion protocol for two (sub)systems SysA and SysB.

5.2 Asynchronous Mutual-Exclusion Protocols

We now present a net-based mutual-exclusion protocol. The system exhibits confusion, but since each agent/process requesting to have permission to access the critical (private/controlled) region can be differentiated from others by indexing its request with a "tag" or process identifier, then the protocol can be ensured to have a communication mechanism with deterministic confusion, as shown next.

The system in Figure 5 is a Petri net representation of a mutual exclusion protocol, which cannot be represented using a free-choice system, but can be modelled using a Ξ system. In the figure, the actions r_a and r_b are requests for entering a critical region denoted by γ; moreover, i_a and i_b (resp. o_a and o_b) are actions for entering (resp. leaving) γ. The c-labelled action abstracts away from the interaction of either subsystem—SysA or SysB—with the rest of the system once permission to access the critical region γ has been granted.

Note that in this case the communication mechanism is even simpler than that for products of sequential agents since in the mutual-exclusion protocol case no new actions are needed: more precisely, communication, i.e., synchronisation, is implemented by the addition of a couple of shared places to the system.

5.3 Noninterleaving Semantics of Finite Process Calculi

Complex systems of concurrent processes can be described using process calculi, such as Milner's CCS [18] or Hoare's CSP [15]. In the case of CCS and related process languages, a mathematically simple and elegant partial order semantics can be given using the model of event structures (see, e.g., [31]). Such a partial order semantics makes use of a 'synchronisation algebra' that defines when two transitions (of two processes) can synchronize with one another. This particular synchronisation mechanism is implemented with a sub-process that comprises three events (one of which is a fresh synchronisation event) exhibiting confusion, as depicted in Figure 6, which can be designed to be deterministic.

If we let each sequential process P_i, which can be easily represented as a CCS process, tag the transitions it owns with its unique identifier/name i, then, as before, we obtain a distributed alphabet—thus, ensuring no auto-concurrency.

However, unlike our previous examples, we now have to add an event for every two transitions that synchronise in the system. The label of such a transition can be defined to be given by an *injective* function f on the set of transitions of each sequential process. Because we only consider finite-state processes, this injective function—which, intuitively, generates fresh names for the new transitions—can be implemented with a finite alphabet of synchronisation events.

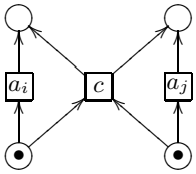

Fig. 6. A net model of the CCS process $P = a_i \parallel a_j$, i.e., of the parallel composition of two processes $P_i = a_i.0$ and $P_j = a_j.0$. The 'synchronization algebra' of P allows a_i and a_j either to synchronise and produce fresh c or to be executed independently. Note that deterministic confusion is ensured because f is injective on transitions.

6 Concluding Remarks and Related Work

Many logics have been defined to reason about the behaviour of processes with noninterleaving (causal, partial order, true-concurrency) semantics. For instance, see [2, 3, 5, 6, 12, 13, 20, 23–26], and the references therein. The main questions addressed in these papers include satisfiability, model checking, and equivalence checking. In particular, the work in [3, 20, 26] focused on coincidence results between logical equivalences and equivalences for concurrency in very general settings. Positive results can be found in some of the papers cited mentioned, especially in [3] where the logical equivalences induced by many logics are shown to correspond to some of the best-known bisimulation equivalences for 'true-concurrency' (i.e., for concurrent processes with partial order semantics).

In this paper, we too were interested in the equivalence checking problem but, more specifically, in the discovery of interesting/natural situations where nondeterminism could be reasonably restricted so that the standard equivalence relation for causal behaviour (namely, HPB) could be captured by a simpler, weaker, more local relation. Our approach was logic-based and game-theoretic. In that sense, our work is more closely related to that in [5, 6], although the restrictions imposed in [5, 6] are in the structure of the models themselves rather than in their degree of nondeterminism, as is the case here. Unfortunately, the bisimulation equivalences induced by the logics studied in [5, 6] do not coincide, in most cases, with the standard bisimilarities for noninterleaving concurrency, not even when restricted to particular classes of concurrent processes.

Not only our results show that it may be more fruitful to restrict the degree of *nondeterminism* in the processes under consideration—instead of the structure

of the processes themselves—but also they give strong indications that it may be nondeterministic confusion which differentiates the expressive power of most relevant modal logics whose induced equivalences are strictly between strong and history-preserving bisimilarity. In fact, all the examples (we are aware of) in the literature given to illustrate differences of equivalences strictly between SB and HPB are with respect to concurrent processes with nondeterministic confusion. This is not the case for other equivalences; for instance, there are examples to differentiate, e.g., SB from history-preserving bisimilarities [10] or 'plain' HPB from stronger history-preserving bisimilarties [9], that do not require nondeterministic confusion. In fact, there are counter-examples to the coincidence between all of these equivalences over concurrent systems without any confusion at all [9].

Our results mainly apply to Ξ systems, informally, concurrent processes with deterministic communication mechanisms (formally, multi-agent systems with deterministic confusion). Some key observations are the following: over concurrent and multi-agent systems, where sequential agents are explicitly represented, it becomes natural to define deterministic confusion relations on the models representing their *partial order* behaviour. The question we have addressed in this paper has also been studied in the *interleaving* context. For instance, coincidence results between the logical equivalence induced by a number of modal logics and bisimulation equivalences for interleaving concurrency can be found in work by Milner and Hennessy [14] and by De Nicola and Vaandrager [19].

Another important feature of the work reported in this paper is the use of logical and mathematical reasoning based on two natural dualities of local behaviour in concurrent processes, which are provided by a simple axiomatisation of noninterleaving behaviour. This kind of local reasoning is at the heart of the work developed in [12] and, to the best of our knowledge, it had not been used anywhere else to analyse the relationship between different logical and concurrent equivalences. We believe that the observations made about such local dualities of concurrent and causal behaviour deserve to be explored even much further.

Acknowledgements. I thank the support of the ERC Advance Grant RACE and Michael Wooldridge for useful comments on an initial version of the paper. The first part of this paper is an extended and revised version of [11]; the second and third parts are based on unpublished results initially studied in [12].

References

1. R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *J. ACM*, **49**(5):672–713, 2002.
2. R. Alur, D. Peled, and W. Penczek. Model-checking of causality properties. In *LICS*, 90–100. IEEE Computer Society, 1995.
3. P. Baldan and S. Crafa. A logic for true concurrency. *J. ACM*, **61**(4):24, 2014.
4. J. v. Benthem. Logic games, from tools to models of interaction. In *Logic at the Crossroads*, 283–317. Allied Publishers, 2007.
5. J. C. Bradfield. Independence: logics and concurrency. In *Truth and Games: Essays in Honour of Gabriel Sandu*, Acta Phil. Fennica **78**, 47–70. Soc. Phil. Fen., 2006.

6. J. C. Bradfield and S. B. Fröschle. Independence-friendly modal logic and true concurrency. *Nordic Journal of Computing*, **9**(1):102–117, 2002.
7. J. C. Bradfield and C. Stirling. Modal mu-calculi. In *Handbook of Modal Logic*, 721–756. Elsevier, 2007.
8. J. Desel and J. Esparza. *Free Choice Petri Nets*. Cambridge Tracts in Theoretical Computer Science **40**. Cambridge University Press, 1995.
9. S. B. Fröschle. *Decidability and Coincidence of Equivalences for Concurrency*. PhD thesis, Univeristy of Edinburgh, 2004.
10. R. J. v. Glabbeek and U. Goltz. Refinement of actions and equivalence notions for concurrent systems. *Acta Informatica*, **37**(4/5):229–327, 2001.
11. J. Gutierrez. Logics and bisimulation games for concurrency, causality and conflict. In *FOSSACS*, LNCS **5504**, 48–62. Springer, 2009.
12. J. Gutierrez. *On Bisimulation and Model-Checking for Concurrent Systems with Partial Order Semantics*. PhD thesis, University of Edinburgh, 2011.
13. J. Gutierrez and J. C. Bradfield. Model-checking games for fixpoint logics with partial order models. *Information and Computation*, **209**(5):766–781, 2011.
14. M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *J. ACM*, **32**(1):137–161, 1985.
15. C. A. R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 2004.
16. L. Jategaonkar and A. R. Meyer. Deciding true concurrency equivalences on safe, finite nets. *Theoretical Computer Science*, **154**(1):107–143, 1996.
17. D. Kozen. Results on the propositional mu-calculus. *Theoretical Computer Science*, **27**:333–354, 1983.
18. R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.
19. R. D. Nicola and F. W. Vaandrager. Three logics for branching bisimulation. *J. ACM*, **42**(2):458–487, 1995.
20. M. Nielsen and C. Clausen. Games and logics for a noninterleaving bisimulation. *Nordic Journal of Computing*, **2**(2):221–249, 1995.
21. M. Nielsen, G. D. Plotkin, and G. Winskel. Petri nets, event structures and domains, Part I. *Theoretical Computer Science*, **13**:85–108, 1981.
22. M. Nielsen and G. Winskel. Models for concurrency. In *Handbook of Logic in Computer Science*, 1–148. Oxford University Press, 1995.
23. W. Penczek. Branching time and partial order in temporal logics. In *Time and Logic: A Computational Approach*, 179–228. UCL Press, 1995.
24. W. Penczek and R. Kuiper. Traces and logic. In *The Book of Traces*, 307–390. World Scientific, 1995.
25. I. Phillips and I. Ulidowski. A hierarchy of reverse bisimulations on stable configuration structures. *Mathematical Structures in Computer Science*, **22**(2):333–372, 2012.
26. I. Phillips and I. Ulidowski. Event identifier logic. *Mathematical Structures in Computer Science*, **24**(2), 2014.
27. V. Sassone, M. Nielsen, and G. Winskel. Models for concurrency: Towards a classification. *Theoretical Computer Science*, **170**(1-2):297–348, 1996.
28. E. Smith. On the border of causality: Contact and confusion. *Theoretical Computer Science*, **153**(1&2):245–270, 1996.
29. C. Stirling. Bisimulation, modal logic and model checking games. *Logic J. IGPL*, **7**(1):103–124, 1999.
30. G. Winskel. *Events in Computation*. PhD thesis, University of Edinburgh, 1980.
31. G. Winskel. Event structure semantics for CCS and related languages. In *ICALP*, LNCS **140**, 561–576. Springer, 1982.
32. M. Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley & Sons, 2009.