Model Checking a Logic for True Concurrency

PAOLO BALDAN and TOMMASO PADOAN, Università di Padova, Italy

We study the model-checking problem for a logic for true concurrency, whose formulae predicate about events in computations and their causal dependencies. The logic, which represents the logical counterpart of history-preserving bisimilarity, is naturally interpreted over event structures or any formalism that can be given a causal semantics, like Petri nets. It includes least and greatest fixpoint operators and thus it can express properties of infinite computations. Since the event structure associated with a system is typically infinite (even if the system is finite state), already the decidability of model-checking is non-trivial. We first develop a local model-checking technique based on a tableau system, for which, over a class of event structures satisfying a suitable regularity condition, referred to as strong regularity, we prove termination, soundness, and completeness. The tableau system allows for a clean and intuitive proof of decidability, but a direct implementation of the procedure can be extremely inefficient. For easing the development of a more efficient model-checking technique, we move to an automata-theoretic framework. Given a formula and a strongly regular event structure, we show how to construct a parity tree automaton whose language is non-empty if and only if the event structure satisfies the formula. The automaton is usually infinite. We discuss how it can be quotiented to an equivalent finite automaton, where emptiness can be checked effectively. To show the applicability of the approach, we discuss how it instantiates to finite safe Petri nets, providing also a corresponding proof-of-concept model-checking tool.

CCS Concepts: • **Theory of computation → Verification by model checking**; *Parallel computing models*; *Program semantics*;

Additional Key Words and Phrases: True concurrency, event structures, Petri nets, model checking, tableaux, tree automata

ACM Reference format:
Paolo Baldan and Tommaso Padoan. 2020. Model Checking a Logic for True Concurrency. *ACM Trans. Comput. Logic* 21, 4, Article 34 (October 2020), 49 pages.
https://doi.org/10.1145/3412853

1 INTRODUCTION

When dealing with concurrent and distributed systems, a partial order approach to the semantics can be appropriate for providing a precise account of the computational steps and of their dependencies, such as causality and concurrency. This is normally referred to as the true concurrent approach to the semantics and opposed to the so-called interleaving approach where concurrency of actions is reduced to the non-deterministic choice among their possible sequentializations. True concurrent models can be convenient also because, thanks to an explicit representation of

Authors' addresses: P. Baldan and T. Padoan, Dipartimento di Matematica "Tullio Levi-Civita," Università di Padova, Via Trieste 63, Padova, 35121, Italy; emails: {baldan, padoan}@math.unipd.it.

1529-3785/2020/10-ART34 $15.00
https://doi.org/10.1145/3412853

concurrency, they provide some relief to the so-called state-space explosion problem in the analysis of concurrent systems, which instead occurs more severely in interleaving approaches (see, e.g., Reference [17]). A widely used foundational model in this class is given by Winskel's event structures [49]. They describe the behaviour of a system in terms of events in computations and two dependency relations: a partial order modelling causality and an additional relation modelling conflict. A survey on the use of such causal models can be found in Reference [50]. Recently they have been used in the study of concurrency in weak memory models [24, 40], for process mining and differencing [15], in the study of atomicity [18], and of information flow [3] properties.

Operational models can be abstracted by considering true concurrent equivalences that range from hereditary history-preserving bisimilarity to the coarser pomset and step equivalences (see, e.g., Reference [47]). On the logical side, various behavioural logics have been proposed capable of expressing causal properties of computations (see, e.g., References [7, 9, 13, 33, 36, 41, 42] just to mention a few and References [2, 19–21, 28] for some related verification techniques). The idea of having temporal and modal logics interpreted over event structures appears in some early work [27, 31, 32, 35], where sound and complete axiomatisations are identified for suitable subclasses of event structures.

Recently, the logical characterisation of true concurrent behavioural equivalences has received a renewed interest and corresponding event-based logics have been introduced [4, 39], interpreted over event structures. Logic formulae include variables that can be bound to events in computations and describe their dependencies. The expressiveness of such logics is sufficient to provide a logical characterisation of the main behavioural equivalences in the true concurrent spectrum [47]. Hereditary history-preserving (hhp-)bisimilarity [7], the finest equivalence in the spectrum, corresponds to the full logics, i.e., two systems are hhp-bisimilar if and only if they satisfy the same logical formulae, and fragments can be identified corresponding to coarser behavioural equivalences. While the relation between operational models, behavioural equivalences, and event-based true concurrent logics is well understood, the corresponding model-checking problem has received limited attention.

In this article, we focus on the logic referred to as $\mathcal{L}_{hp}$ in Reference [4], corresponding to a classical equivalence in the spectrum, i.e., history-preserving (hp-)bisimilarity [8, 14, 43]. The logic is endowed with least and greatest fixpoint operators, in mu-calculus style, to express interesting properties of infinite computations. Hp-bisimilarity is known to be decidable for finite safe Petri nets [23, 29, 48]. However, the question remains open on whether the corresponding model-checking problem for $\mathcal{L}_{hp}$ is decidable over some interesting class of systems. Note that the decidability of model-checking is non-trivial, even for finite state systems, since event structure models are typically infinite and the possibility of expressing properties that depends on the past often leads to undecidability [25].

The article develops an extensive study of this problem. First, relying on a tableau-based technique, we prove the decidability of model-checking for $\mathcal{L}_{hp}$ over a class of event structures satisfying a suitable regularity condition. Then, on the way to a concrete implementation, we devise an automata-theoretic technique for model-checking. Finally, we discuss how such technique can be implemented in practice on Petri net models, providing a proof-of-concept tool.

More in detail, inspired by the work in References [12, 45] for the mu-calculus, to tackle the decidability problem for $\mathcal{L}_{hp}$, we are naturally led to focus on local algorithms in the form of tableau systems. For checking whether a system model satisfies a given formula, a set of proof trees is constructed by applying a suitable set of rules that reduce the truth of a formula in a given state to the truth of suitably generated subformulae. This "local" approach that explores the state space "on demand" is particularly suited in our setting, characterised by the infiniteness of the event structure model of any non-trivial system.

The presence of fixpoint operators makes the issue of sound termination quite delicate and non-trivial already in the original approach dealing with finite transition systems. In our setting, where also the transition system is infinite, this is further complicated. A key choice that we take is the restriction to a class of event structures having a finitary flavour, which we call strongly regular event structures. Recall that regular event structures [46] are characterised by the fact that the number of sub-structures arising as residuals of the original event structure after some steps of computations is finite up to isomorphism. The intuition is that, after going sufficiently in depth, the event structure starts repeating cyclically. For strongly regular event structures the requirement is strengthened by asking the finiteness of the residuals extended with a bounded number of events from the past. This is important in our setting, since $\mathcal{L}_{hp}$ formulae can express history-based properties that depend not only on the future but also on events executed in the past.

A direct implementation of the tableau-based procedure can be extremely inefficient. Roughly speaking, the problem is that in the search of a successful tableau, branches that are, in some sense, equivalent are explored several times.

For this reason, we also devise an automata-theoretic technique, in the style of Reference [16], which reduces the truth of a formula in a model to the emptiness of the language accepted by a suitable automaton. Given a formula of $\mathcal{L}_{hp}$ and a strongly regular event structure, the procedure generates a parity tree automaton. Truth is reduced to emptiness in the sense that the event structure satisfies the formula if and only if the automaton accepts a non-empty language.

The result is not directly usable for practical purposes, since the automaton is infinite for any non-trivial event structure. However an equivalence on states can be defined such that the quotiented automaton accepts the same language as the original one. Whenever such equivalence is of finite index the quotiented automaton is finite, so satisfaction of the formula can be checked effectively on the quotient. We show that for all strongly regular event structures a canonical equivalence always exists that is of finite index.

The model-checking procedures are developed abstractly on event structures. A concrete algorithm on some formalism requires the effectiveness of the chosen equivalences on sequents/states and of the transition relation.

We discuss a concrete instantiation of the automata-theoretic procedure on finite safe Petri nets. It is implemented in a tool, wishfully called *True Concurrency Workbench* (TCWB), written in Haskell. Roughly, the search of an accepting run in the automaton can be seen as an optimisation of the procedure for building a successful tableau: The graph structure underlying the automaton helps in the reuse of the information discovered. Indeed, some tests reveal that the TCWB is way more efficient than the direct implementation of the tableau-based procedure (which could not manage most of the (simple) examples in the TCWB repository).

The rest of the article is structured as follows: In Section 2, we review event structures, strong regularity, and the logic $\mathcal{L}_{hp}$ of interest in the article. In Section 3, we introduce the true concurrent logic $\mathcal{L}_{hp}$ and its fixpoint extension. In Section 4, we give the model-checking procedure as a tableau system, and we prove its soundness, completeness and termination. In Section 5, we introduce (infinite state) parity tree automata and we show how the model-checking problem for $\mathcal{L}_{hp}$ on strongly regular PESs can be reduced to the non-emptiness of the language of such automata. In Section 6, we discuss the instantiation of the approach to Petri nets. Finally, in Section 7, we discuss some related work and outline directions of future research. To streamline the presentation, the proofs of some technical results are confined to an Appendix.

This article is a revised and extended version of the conference papers of References [5, 6]. The results of Reference [5], originally restricted to the alternation-free fragment of $\mathcal{L}_{hp}$, are extended here to the full logic. Moreover, full proofs, omitted in the conference versions, are provided. In

particular, the proofs of the results for the automata-theoretic technique have been vastly reworked to clarify and make formal the relation with the tableau system.

2 EVENT STRUCTURES AND REGULARITY

Prime event structures [49] are a widely known model of concurrency. They describe the behaviour of a system in terms of events and dependency relations between such events. Throughout the article, $\mathbb{E}$ is a fixed countable set of events from which all events are taken, Λ a finite set of labels ranged over by a, b, c, ..., and $\lambda : \mathbb{E} \to \Lambda$ a labelling function.

Definition 2.1 (Prime Event Structure). A (Λ-labelled) *prime event structure* (PES) is a tuple $\mathcal{E} = \langle E, \leq, \# \rangle$, where $E \subseteq \mathbb{E}$ is the set of *events* and $\leq$, $\#$ are binary relations on E, called *causality* and *conflict*, respectively, such that:

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

The PES $\mathcal{E}_1 = \langle E_1, \leq_1, \#_1 \rangle$, $\mathcal{E}_2 = \langle E_2, \leq_2, \#_2 \rangle$ are *isomorphic*, written $\mathcal{E}_1 \sim \mathcal{E}_2$, when there is a bijection $\iota : E_1 \to E_2$ such that for all $e_1, e_1' \in E_1$ it holds $e_1 \leq_1 e_1'$ iff $\iota(e_1) \leq_2 \iota(e_1')$ and $e_1 \#_1 e_1'$ iff $\iota(e_1) \#_2 \iota(e_1')$ and $\lambda(e_1) = \lambda(\iota(e_1))$.

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

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

Causality and concurrency will be sometimes used on set of events. Given $X \subseteq E$ and $e \in E$, by $X < e$, we mean that for all $e' \in X$, $e' < e$. Similarly $X \,||\, e$, respectively, $X \frown e$, means that for all $e' \in X$, $e' \,||\, e$, respectively, $e' \frown e$.

The concept of (concurrent) computation for event structures is captured by the notion of configuration.

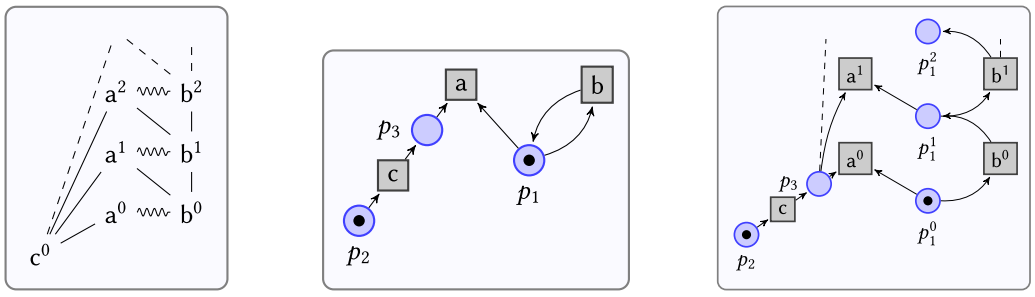
Definition 2.3 (Configuration). Let $\mathcal{E}$ be a PES. A *configuration* in $\mathcal{E}$ is a finite consistent subset of events $C \subseteq E$ closed w.r.t. causality (i.e., $\lceil e \rceil \subseteq C$ for all $e \in C$). The set of finite configurations of $\mathcal{E}$ is denoted by $\mathcal{C}(\mathcal{E})$.

The empty set of events $\emptyset$ is always a configuration, which can be interpreted as the initial state of the computation. The evolution of a system can be represented by a transition system where configurations are states.

Definition 2.4 (Transition System). Let $\mathcal{E}$ be a PES and let $C \in \mathcal{C}(\mathcal{E})$. Given $e \in E \setminus C$ such that $C \cup \{e\} \in \mathcal{C}(\mathcal{E})$, and $X, Y \subseteq C$ with $X < e$, $Y \,||\, e$, we write $C \xrightarrow{X, \overline{Y} < e}_{\lambda(e)} C \cup \{e\}$, possibly omitting X, Y, or the label $\lambda(e)$.

Transitions are labelled by the executed event e. In addition, they can report its label $\lambda(e)$, a subset of causes X, and a set of events $Y \subseteq C$ concurrent with e. When X or Y are empty they are normally omitted, e.g., we write $C \xrightarrow{X < e}_{\lambda(e)} C'$ for $C \xrightarrow{X, \overline{\emptyset} < e}_{\lambda(e)} C'$ and $C \xrightarrow{e}_{\lambda(e)} C'$ for $C \xrightarrow{\emptyset, \overline{\emptyset} < e}_{\lambda(e)} C'$.

Definition 2.5 (Branching). Let $\mathcal{E}$ be a PES. The set of *enabled* events at a configuration C is defined as $en(C) = \{e \in E \mid C \xrightarrow{e} C'\}$. We say that $\mathcal{E}$ is *k-boundedly branching* for some $k \in \mathbb{N}$ if



(a) $\mathcal{E}_{\mathcal{N}}$ (b) $\mathcal{N}$ (c) $\mathcal{U}(\mathcal{N})$

Fig. 1. (a) A PES $\mathcal{E}_{\mathcal{N}}$ associated with the net $\mathcal{N}$ in (b) via its unfolding (c).

$|en(C)| \leq k$ for all $C \in \mathcal{C}(\mathcal{E})$. We say that it is *boundedly branching* if it is k-boundedly branching for some $k \in \mathbb{N}$.

We already mentioned that the PES associated with a non-trivial system exhibiting a cyclic behaviour is infinite. We next introduce a class of PESs enjoying a finitary property that we call strong regularity.

Definition 2.6 (Residual). Let $\mathcal{E}$ be a PES. For a configuration $C \in \mathcal{C}(\mathcal{E})$, the *residual* of $\mathcal{E}$ after C, is defined as $\mathcal{E}[C] = \{e \mid e \in E \setminus C \ \wedge \ C \frown e\}$.

The residual can be seen as a PES, endowed with the restriction of causality and conflict of $\mathcal{E}$. Intuitively, it represents the PES that remains to be executed after the computation expressed by C.

Given $C \in \mathcal{C}(\mathcal{E})$ and $X \subseteq C$, we denote by $\mathcal{E}[C] \cup X$ the PES obtained from $\mathcal{E}[C]$ by adding the events in X with the causal dependencies they had in the original PES $\mathcal{E}$.

Definition 2.7 (Strong Regularity). A PES $\mathcal{E}$ is called *strongly regular* when it is boundedly branching and for each $k \in \mathbb{N}$ the set $\{\mathcal{E}[C] \cup \{e_1, \ldots, e_k\} \mid C \in \mathcal{C}(\mathcal{E}) \ \wedge \ e_1, \ldots, e_k \in C\}$ is finite up to isomorphism of PESs.

Strong regularity is obtained from the notion of regularity in Reference [46], by replacing residuals with residuals extended with a bounded number of events from the past. Roughly, the requirement is that the PES has a finite number of extended residuals over which the computation cycles. Intuitively, this is important, since we are interested in history-dependent properties. Clearly, each strongly regular PES is regular, since the property in Definition 2.7 must hold, in particular, for $k = 0$. We will later show in Section 6 that the PESs associated with finite safe Petri nets, i.e., the regular trace PESs [46], are strongly regular.

A simple PES is depicted in Figure 1(a). Graphically, curly lines represent immediate conflicts and the causal partial order proceeds upwards along the straight lines. Events are denoted by their labels, possibly with superscripts. For instance, in $\mathcal{E}_{\mathcal{N}}$, the events a^0 and b^0, labelled by a and b, respectively, are in conflict. Event c^0 causes the events a^i and it is concurrent with b^i for all $i \in \mathbb{N}$. It is an infinite PES associated with the Petri net $\mathcal{N}$ in Figure 1(b) in a way that will be discussed in Section 6.1, hence, it is strongly regular by Corollary 6.5. For instance, it has six (equivalence classes of) residuals extended with a single event from the past $\mathcal{E}_{\mathcal{N}}[\{\mathrm{c}^0\}] \cup \{\mathrm{c}^0\}$, $\mathcal{E}_{\mathcal{N}}[\{\mathrm{b}^0\}] \cup \{\mathrm{b}^0\}$, $\mathcal{E}_{\mathcal{N}}[\{\mathrm{c}^0, \mathrm{b}^0\}] \cup \{\mathrm{b}^0\}$, $\mathcal{E}_{\mathcal{N}}[\{\mathrm{c}^0, \mathrm{a}^0\}] \cup \{\mathrm{c}^0\}$, $\mathcal{E}_{\mathcal{N}}[\{\mathrm{c}^0, \mathrm{a}^0\}] \cup \{\mathrm{a}^0\}$, and $\mathcal{E}_{\mathcal{N}}[\{\mathrm{c}^0, \mathrm{b}^0, \mathrm{a}^1\}] \cup \{\mathrm{b}^0\}$.

3 A LOGIC FOR TRUE CONCURRENCY

In this section, we introduce the syntax and the semantics of the logic for concurrency of interest in the article. Originally defined in Reference [4], the logic has formulae that predicate over executability of events in computations and their dependency relations (causality and concurrency).

3.1 Syntax

Logic formulae include event variables, from a fixed denumerable set *Var*, denoted by $x, y, \ldots$. To keep the notation simple, tuples of variables such as $x_1, \ldots, x_n$ will be denoted by a corresponding boldface letter $\mathbf{x}$ and, abusing the notation, tuples will be often used as sets; e.g., we will write $x \in \mathbf{x}$ instead of $x = x_i$ for some $i \in \{1, \ldots, n\}$. The logic, in positive form, besides the standard propositional connectives $\wedge$ and $\vee$, includes diamond and box modalities. The formula $\langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z \rangle\!\rangle \varphi$ holds when in the current configuration an a-labelled event e is enabled that causally depends on the events bound to the variables in $\mathbf{x}$ and is concurrent with those in $\mathbf{y}$. Event e is executed and bound to variable z, and then the formula φ must hold in the resulting configuration. Dually, $[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!] \varphi$ is satisfied when all a-labelled events causally dependent on $\mathbf{x}$ and concurrent with $\mathbf{y}$ bring to a configuration where φ holds.

Fixpoint operators refer to propositional variables. To let them interact correctly with event variables, whose values can be passed from an iteration to the next one in the recursion, we use abstract propositions.

We fix a denumerable set of *abstract propositions* $\mathcal{X}^a$ (where the superscript "a" stands for "abstract"), ranged over by $X, Y, \ldots$, that are intended to represent formulae possibly containing (unnamed) free event variables. Each abstract proposition has an arity $ar(X)$, which indicates the number of free event variables in X. An abstract proposition X can be turn into a formula by specifying a name for its free variables. For $\mathbf{x}$ such that $|\mathbf{x}| = ar(X)$, we write $X(\mathbf{x})$ to indicate the abstract proposition X whose free event variables are named $\mathbf{x}$. When $ar(X) = 0$, we will write X instead of $X(\epsilon)$ omitting the trailing empty tuple of variables. We call $X(\mathbf{x})$ a *proposition* and denote by $\mathcal{X}$ the set of all propositions.

Definition 3.1 (Syntax). The syntax of $\mathcal{L}_{hp}$ over the sets of event variables *Var*, abstract propositions $\mathcal{X}^a$, and labels Λ is defined as follows:

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

The free event variables of a formula φ are denoted $fv(\varphi)$ and defined in the obvious way. Just note that the modalities act as binders for the variable representing the event executed, hence $fv(\langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z \rangle\!\rangle \varphi) = fv([\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!] \varphi) = (fv(\varphi) \setminus \{z\}) \cup \mathbf{x} \cup \mathbf{y}$. The free propositions in φ, i.e., the propositions not bound by μ or ν, are denoted by $fp(\varphi)$. Formulae $(\alpha X(\mathbf{x}).\varphi)(\mathbf{y})$ for $\alpha \in \{\mu, \nu\}$, are referred to as *α-formulae* and hereafter α ranges over $\{\nu, \mu\}$. In such formulae, we require that the tuple $\mathbf{x}$ does not include multiple occurrences of the same variable and, thus, it can be seen as a set, which must correspond exactly to the free event variables of the inner formula φ, i.e., $fv(\varphi) = \mathbf{x}$. Intuitively, the fixpoint part $\alpha X(\mathbf{x}).\varphi$ defines a recursive formula $X(\mathbf{x})$ whose free variables are then instantiated with $\mathbf{y}$. The formula $(\alpha X(\mathbf{x}).\varphi)(\mathbf{x})$ will be abbreviated as $\alpha X(\mathbf{x}).\varphi$. In References [4–6] only the simplified form of the fixpoint syntax was used. This slight extension allows for a simpler treatment of substitutions. When both $fv(\varphi)$ and $fp(\varphi)$ are empty, we say that φ is *closed*. When $\mathbf{x}$ or $\mathbf{y}$ are empty, they are often omitted, e.g., we write $\langle\!\langle \mathrm{a}\, z \rangle\!\rangle \varphi$ for $\langle\!\langle \emptyset, \overline{\emptyset} < \mathrm{a}\, z \rangle\!\rangle \varphi$.

Given a formula φ and variables $x, y \in Var$, we denote by $\varphi[y/x]$ the formula obtained from φ via a (capture avoiding) substitution of the free occurrences of x in φ by y. A function σ mapping

free variables of φ to variables will be called a substitution, and we will denote by $\varphi\sigma$ the formula $\varphi[\sigma(fv(\varphi))/fv(\varphi)]$.

When defining the substitution of formulae for propositions some care is needed in the treatment of variables. In fact, when replacing the proposition $Z(\mathbf{x})$ in φ with some formula ψ, actually each occurrence $Z(\mathbf{y})$ of Z must be replaced by a formula obtained from ψ by renaming its free event variables to $\mathbf{y}$. Given a proposition $Z(\mathbf{x}) \in \mathcal{X}$ and a formula ψ such that $fv(\psi) \subseteq \mathbf{x}$, we denote by $\varphi[Z(\mathbf{x}) := \psi]$ the formula obtained from φ by replacing free occurrences of $Z(\mathbf{y})$ by $\psi[\mathbf{y}/\mathbf{x}]$.

Detailed definitions of free event variables and propositions and routine results about substitutions can be found in Appendix A.1.

3.2 Abbreviations and Examples

In the logic, we can easily represent the possibility of performing concurrent events, each with its own dependencies. Borrowing the notation from Reference [4], we will write

$$(\langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z \rangle\!\rangle \otimes \langle\!\langle \mathbf{x}', \overline{\mathbf{y}'} < \mathrm{b}\, z' \rangle\!\rangle)\, \varphi$$

for the formula $\langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z \rangle\!\rangle \langle\!\langle \mathbf{x}', \overline{\mathbf{y}', z} < \mathrm{b}\, z' \rangle\!\rangle \varphi$, which declares the existence of two concurrent events, labelled by a and b, respectively, such that if we execute such events and bind them to z and z', respectively, then φ holds. In particular, the ability to perform a step consisting of two concurrent events labelled by a and b is simply expressed by the formula $(\langle\!\langle \mathrm{a}\, x \rangle\!\rangle \otimes \langle\!\langle \mathrm{b}\, y \rangle\!\rangle)\mathsf{T}$. This notation can be easily generalised to the execution of any number of concurrent events. We can dually define the formula $([\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!] \otimes [\![\mathbf{x}', \overline{\mathbf{y}'} < \mathrm{b}\, z']\!])\varphi$ stating that after the execution of all pairs of concurrent events, having the specified dependencies and labelled a and b, respectively, the formula φ holds.

We will also use a wildcard operator to refer to an event with an arbitrary label. When the set of labels Λ is finite, we write

$$\langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \_\, z \rangle\!\rangle \varphi$$

to denote the formula $\bigvee_{\mathrm{a}\in\Lambda} \langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z \rangle\!\rangle \varphi$, and dually $[\![\mathbf{x}, \overline{\mathbf{y}} < \_\, z]\!]\varphi$ for $\bigwedge_{\mathrm{a}\in\Lambda} [\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\varphi$. For instance, the formula $([\![\_\, y_1]\!] \otimes [\![\_\, y_2]\!] \otimes [\![\_\, y_3]\!])\mathsf{F}$ says that in the current state there are at most two concurrently enabled events. A formula $\nu X.(([\![\_\, y_1]\!] \otimes \ldots \otimes [\![\_\, y_k]\!])\mathsf{F} \ \wedge \ [\![\_\, z]\!]X)$ states that the level of parallelism will never exceed $k-1$.

The formula $\langle\!\langle \mathrm{c}\, x \rangle\!\rangle(\langle\!\langle x < \mathrm{a}\, y \rangle\!\rangle \mathsf{T} \wedge \langle\!\langle \overline{x} < \mathrm{b}\, z \rangle\!\rangle \mathsf{T})$ requires that, after the execution of a c-labelled event, one can choose between a causally dependent a-labelled event and a concurrent b-labelled event. This is satisfied by $\mathcal{E}_{\mathcal{N}}$ in Figure 1(a). Instead $\langle\!\langle \mathrm{c}\, x \rangle\!\rangle(\langle\!\langle \overline{x} < \mathrm{a}\, y \rangle\!\rangle \mathsf{T} \wedge \langle\!\langle \overline{x} < \mathrm{b}\, z \rangle\!\rangle \mathsf{T})$ requiring both events to be concurrent would be false.

As an example of property of infinite computations, consider the formula

$$[\![\mathrm{b}\, x]\!]\nu Z(x).((\langle\!\langle \mathrm{c}\, w \rangle\!\rangle \otimes \langle\!\langle \mathrm{b}\, z \rangle\!\rangle)\mathsf{T} \wedge [\![x < \mathrm{b}\, y]\!]Z(y)),$$

expressing that all non-empty causal chain of b-labelled events reach a state where it is possible to execute two concurrent events labelled c and b, respectively. Then the formula holds in $\mathcal{E}_{\mathcal{N}}$. Another formula satisfied by $\mathcal{E}_{\mathcal{N}}$ is $(\langle\!\langle \mathrm{c}\, x \rangle\!\rangle \otimes \langle\!\langle \mathrm{b}\, y \rangle\!\rangle)\nu X(x,y).(\langle\!\langle y, \overline{x} < \mathrm{b}\, z \rangle\!\rangle X(x,z))$ requiring the existence of an infinite causal chain of b-labelled events, concurrent with a c-labelled event.

Now consider the formula $\mu X.(\langle\!\langle \_\, z \rangle\!\rangle X \vee \langle\!\langle \mathrm{b}\, x \rangle\!\rangle \langle\!\langle x < \mathrm{a}\, y \rangle\!\rangle \nu Y.\langle\!\langle \_\, z \rangle\!\rangle Y)$. It requires the existence of an infinite run containing a b-labelled event immediately followed by a causally dependent a-labelled event, and it turns out to be false in the same PES. Intuitively, this is because any a-labelled event causally dependent on a b-labelled event is in conflict with the rest of the infinite chain of events, and then, after its execution, the computation is guaranteed to terminate. The formula $\langle\!\langle \mathrm{b}\, x \rangle\!\rangle \nu X(x).\, \mu Y(x).(\langle\!\langle x < \mathrm{b}\, y \rangle\!\rangle X(y) \vee \langle\!\langle \_\, z \rangle\!\rangle Y(x))$ states that there exists an infinite chain of causally related, possibly non-consecutive, b-labelled events and it is satisfied by $\mathcal{E}_{\mathcal{N}}$ in Figure 1(a).

As a final example, consider the notion of atomicity based on causality presented in Reference [18]. The control flow of a program with threads is modelled as a Petri net that captures the independence and interaction between threads. The causality between events in the partially ordered executions of the Petri net is used to define the notion of causal atomicity for program blocks. Roughly, a program block A is causally atomic if there are no events e_1, e_2 from A and e outside A such that $e_1 < e < e_2$. If each event is labelled by the block it is in, the causal atomicity property can be expressed in $\mathcal{L}_{hp}$ as

$$\nu X.\left([\![\_\, w]\!]X \ \wedge\ [\![\mathsf{A}\, x]\!]\nu Y(x).\left(\bigwedge_{B\in\Lambda\setminus\{A\}} [\![x < \mathsf{B}\, y]\!][\![y < \mathsf{A}\, z]\!]\mathsf{F} \ \wedge\ [\![\_\, w]\!]Y(x)\right)\right).$$

3.3 Alternation

An order induced on propositions by the nesting of fixpoints and the notion of fixpoint alternation in the formulae of $\mathcal{L}_{hp}$ will play a role in the article (in particular for defining the acceptance condition for the automaton in Section 5.2). We adapt some definition from Reference [16]. Hereafter, we will assume that in every formula different bound propositions have different names, so we can refer to *the* fixpoint subformula quantifying an abstract proposition. This requirement can always be fulfilled by alpha-renaming. This will help us to keep the notation simpler.

Definition 3.2 (Active Subformula). Given an α-formula $\varphi = (\alpha X(\mathbf{x}).\varphi')(\mathbf{y})$, we say that a subformula ψ of φ is a *direct active subformula*, written $\psi \sqsubseteq_d \varphi$, if $X \in fp(\psi)$. When $\psi \sqsubseteq_d^* \varphi$, we say that ψ is an *active subformula* of φ. We denote by $\mathsf{sf}(\varphi)$ the set of subformulae of a formula φ and by $\mathsf{asf}_\alpha(\varphi)$ the set of active α-subformulae.

Note that $\sqsubseteq_d$ is acyclic, since it refines the subformula relation, and thus $\sqsubseteq_d^*$ is a partial order.

Definition 3.3 (Alternation Depth). The *alternation depth* of a formula φ in $\mathcal{L}_{hp}$, written $\mathsf{ad}(\varphi)$, is defined, for a ν-formula φ, as $\mathsf{ad}(\varphi) = \max\{1 + \mathsf{ad}(\psi) \mid \psi \in \mathsf{asf}_\mu(\varphi)\}$ and dually, for a μ-formula φ, as $\mathsf{ad}(\varphi) = \max\{1 + \mathsf{ad}(\psi) \mid \psi \in \mathsf{asf}_\nu(\varphi)\}$. For any other formula φ, $\mathsf{ad}(\varphi) = max\{\mathsf{ad}(\psi) \mid \psi \in \mathsf{sf}(\varphi) \setminus \{\varphi\}\}$.

It is intended that $\max \emptyset = 0$; e.g., by the first clause above, the alternation depth of $\nu X(\mathbf{x}).\varphi$ is 0 in absence of active μ-subformulae.

As an example, consider the formula $\langle\!|\mathsf{b}\, x|\!\rangle \nu X(x).\, \mu Y(x).(\langle\!|x < \mathsf{b}\, y|\!\rangle X(y) \vee \langle\!|\_\, z|\!\rangle Y(x))$ from Section 3.2 and write it as $\langle\!|\mathsf{b}\, x|\!\rangle\varphi$, where $\varphi = \nu X(x).\psi$ and $\psi = \mu Y(x).(\langle\!|x < \mathsf{b}\, y|\!\rangle X(y) \vee \langle\!|\_\, z|\!\rangle Y(x))$. It has alternation depth 1, since ψ is a (direct) active subformula of φ, given the fact that $X \in fp(\psi)$. It is not difficult to see that all other formulae in Section 3.2 have instead alternation depth 0; e.g., $\mu X.(\langle\!|\_\, z|\!\rangle X \vee \langle\!|\mathsf{b}\, x|\!\rangle\langle\!|x < \mathsf{a}\, y|\!\rangle\nu Y.\langle\!|\_\, z|\!\rangle Y)$ has alternation depth 0 despite the nesting of a ν-subformula into a μ-subformula, since X does not appear free in the ν-subformula.

Hereafter, if X and X' are abstract propositions quantified in α-subformulae $(\alpha X(\mathbf{x}).\varphi)(\mathbf{y})$ and $(\alpha' X'(\mathbf{x}').\varphi')(\mathbf{y}')$, we will write $\mathsf{ad}(X)$ for $\mathsf{ad}((\alpha X(\mathbf{x}).\varphi)(\mathbf{y}))$ and $X \sqsubseteq_d X'$ for $(\alpha X(\mathbf{x}).\varphi)(\mathbf{y}) \sqsubseteq_d (\alpha' X'(\mathbf{x}').\varphi')(\mathbf{y}')$.

3.4 Semantics

Since the logic $\mathcal{L}_{hp}$ is interpreted over PESs, the satisfaction of a formula is defined with respect to a configuration C, representing the state of the computation and a (total) function $\eta : Var \to E$, called an *environment*, that binds free variables in the formula to events in C. Namely, if $Env_{\mathcal{E}}$ denotes the set of environments, the semantics of a formula will be a set of pairs in $\mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}}$. Given a set of pairs $S \subseteq \mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}}$ and two tuples of variables $\mathbf{x}$ and $\mathbf{y}$, with $|\mathbf{x}| = |\mathbf{y}|$, we define $S[\mathbf{y}/\mathbf{x}] = \{(C, \eta') \mid \exists (C, \eta) \in S \ \wedge \eta(\mathbf{x}) = \eta'(\mathbf{y})\}$. The semantics of $\mathcal{L}_{hp}$ also depends on a proposition environment providing a semantic interpretation for propositions.

Definition 3.4 (Proposition Environment). Let $\mathcal{E}$ be a PES. A *proposition environment* is a function $\pi : \mathcal{X} \to 2^{C(\mathcal{E})\times Env_{\mathcal{E}}}$ such that for all tuples of variables $\mathbf{x}$, $\mathbf{y}$ with $|\mathbf{x}| = |\mathbf{y}| = ar(X)$ it holds $\pi(X(\mathbf{y})) = \pi(X(\mathbf{x}))[\mathbf{y}/\mathbf{x}]$. The set of proposition environments, ranged by π, is denoted $PEnv_{\mathcal{E}}$.

The condition posed on proposition environments ensures that the semantics of a formula only depends on the events that the environment associates with its free variables and that it does not depend on the naming of the variables.

We can now give the semantics of the logic $\mathcal{L}_{hp}$. Given an event environment η and an event e, we write $\eta[x \mapsto e]$ to indicate the updated environment that maps x to e. Similarly, for a proposition environment π and $S \subseteq C(\mathcal{E}) \times Env_{\mathcal{E}}$, we write $\pi[Z(\mathbf{x}) \mapsto S]$ for the corresponding update (see Appendix A.1 for the detailed definition). For a pair $(C, \eta) \in C(\mathcal{E}) \times Env_{\mathcal{E}}$ and variables $\mathbf{x}$, $\mathbf{y}$, z, we define the $(\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}z)$-successors of (C, η), as

$$\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C,\eta) = \{(C', \eta[z \mapsto e]) \mid C \xrightarrow{\eta(\mathbf{x}),\overline{\eta(\mathbf{y})} < e}_{\mathrm{a}} C'\}.$$

In words, $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C, \eta)$ consists of the pairs (C', η') where C' is a configuration reachable from C, by executing an event e satisfying the requirement expressed by $\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}z$, namely, events in $\eta(\mathbf{x})$ are causes for e and events in $\eta(\mathbf{y})$ are concurrent with e. The environment η' is the update of η where event e has been bound to variable z.

Definition 3.5 (Semantics). Let $\mathcal{E}$ be a PES. The denotation of a formula φ in $\mathcal{L}_{hp}$ is given by the function $\{\!|\cdot|\!\}^{\mathcal{E}} : \mathcal{L}_{hp} \to PEnv_{\mathcal{E}} \to 2^{C(\mathcal{E})\times Env_{\mathcal{E}}}$ defined inductively as follows, where we write $\{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}$ instead of $\{\!|\varphi|\!\}^{\mathcal{E}}(\pi)$:

$$\begin{array}{rclcrcl}
\{\!|\mathsf{T}|\!\}^{\mathcal{E}}_{\pi} & = & C(\mathcal{E}) \times Env_{\mathcal{E}} & \qquad & \{\!|\mathsf{F}|\!\}^{\mathcal{E}}_{\pi} = \emptyset & \qquad & \{\!|Z(\mathbf{y})|\!\}^{\mathcal{E}}_{\pi} = \pi(Z(\mathbf{y})) \\
\{\!|\varphi_1 \wedge \varphi_2|\!\}^{\mathcal{E}}_{\pi} & = & \{\!|\varphi_1|\!\}^{\mathcal{E}}_{\pi} \cap \{\!|\varphi_2|\!\}^{\mathcal{E}}_{\pi} & & & & \{\!|\varphi_1 \vee \varphi_2|\!\}^{\mathcal{E}}_{\pi} = \{\!|\varphi_1|\!\}^{\mathcal{E}}_{\pi} \cup \{\!|\varphi_2|\!\}^{\mathcal{E}}_{\pi}
\end{array}$$

$$\{\!|\langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle\!\, \varphi|\!\}^{\mathcal{E}}_{\pi} = \{(C,\eta) \mid \mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C,\eta) \cap \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi} \neq \emptyset\} \quad ,$$

$$\{\!|[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\, \varphi|\!\}^{\mathcal{E}}_{\pi} = \{(C,\eta) \mid \mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C,\eta) \subseteq \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}\}$$

$$\{\!|(\alpha Z(\mathbf{x}).\varphi)(\mathbf{y})|\!\}^{\mathcal{E}}_{\pi} = \alpha(f_{\varphi,Z(\mathbf{x}),\pi})[\mathbf{y}/\mathbf{x}]$$

where $f_{\varphi,Z(\mathbf{x}),\pi} : 2^{C(\mathcal{E})\times Env_{\mathcal{E}}} \to 2^{C(\mathcal{E})\times Env_{\mathcal{E}}}$ is the function defined by $f_{\varphi,Z(\mathbf{x}),\pi}(S) = \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi[Z(\mathbf{x})\mapsto S]}$, which we refer to as the *semantic function* of φ, $Z(\mathbf{x})$, π. Moreover, $\nu(f_{\varphi,Z(\mathbf{x}),\pi})$ (respectively, $\mu(f_{\varphi,Z(\mathbf{x}),\pi})$) denotes the corresponding greatest (respectively, least) fixpoint. When $(C, \eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}$, we say that the PES $\mathcal{E}$ satisfies the formula φ in the configuration C and environments η, π. When φ is closed, hence the environments η, π are irrelevant, and $\mathcal{E}$ satisfies the formula φ in the empty configuration, we simply say that $\mathcal{E}$ satisfies φ.

The semantics of Boolean operators is as usual. The formula $\langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle\!\, \varphi$ holds in (C, η) when from configuration C there is an enabled a-labelled event e that is causally dependent on (at least) the events bound to the variables in $\mathbf{x}$ and concurrent with (at least) those bound to the variables in $\mathbf{y}$ and can be executed producing a new configuration $C' = C \cup \{e\}$ that, paired with the environment $\eta' = \eta[z \mapsto e]$, satisfies φ. Dually, $[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\, \varphi$ holds when all a-labelled events executable from C, caused by $\mathbf{x}$, and concurrent with $\mathbf{y}$ bring to a configuration where φ is satisfied.

The fixpoints corresponding to the formulae $(\alpha Z(\mathbf{y}).\varphi)(\mathbf{y})$ are guaranteed to exist by Knaster-Tarski theorem, since the set $2^{C(\mathcal{E})\times Env_{\mathcal{E}}}$ ordered by subset inclusion is a complete lattice and the functions, of which the fixpoints are calculated, are monotonic.

3.5 Pointed Configurations and Equisatisfaction

The satisfaction of a formula of $\mathcal{L}_{hp}$ in a given configuration surely depends on the future of the configuration, i.e., on the so-called residual of the event structure after the configuration.

Additionally, since the formula can contain free event variables that refer to past events, satisfaction also depends on how such past events are related (via concurrency and causality) with the future.

This motivates the notion of pointed configuration introduced below.

Definition 3.6 (Pointed Configuration). Let $\mathcal{E}$ be a PES and let V be a set. A *V-pointed configuration* is a pair $\langle C, \zeta \rangle$ where $C \in \mathcal{C}(\mathcal{E})$ and $\zeta : V \to C$ is a function.

The notion will be used in situations where V is a set of variables free in a formula, and ζ is the (restriction to such variables of the) environment.

We say that two V-pointed configurations have isomorphic pointed residuals when their residuals are related by a bijection ensuring that events corresponding to the same $x \in V$ have the same causal relations with the future. The formal definition follows:

Definition 3.7 (Isomorphism of Pointed Residual). Let $\mathcal{E}$ be a PES, V a set, and let $\langle C, \zeta \rangle$ and $\langle C', \zeta' \rangle$ be two V-pointed configurations of $\mathcal{E}$. We say that $\langle C, \zeta \rangle$, $\langle C', \zeta' \rangle$ have isomorphic residuals, written $\langle C, \zeta \rangle \approx_r \langle C', \zeta' \rangle$, if there is an isomorphism of the residuals $\iota : \mathcal{E}[C] \to \mathcal{E}[C']$ such that for all $x \in V$, $e \in \mathcal{E}[C]$, we have $\zeta(x) \leq e$ iff $\zeta'(x) \leq \iota(e)$.

We will show that pointed configurations with isomorphic residuals satisfy exactly the same formulae when free event variables correspond to the pointed events. For properly stating such property, since we will work with formulae containing free propositional variables, we need to restrict to proposition environments that are well-behaved in the sense formalised below with respect to the free propositions of the formulae of interest.

Definition 3.8 (Saturated Proposition Environment). Let $\mathcal{E}$ be a PES. Given a formula φ and a proposition environment $\pi \in PEnv_{\mathcal{E}}$, we say that π is *saturated for* φ when for all $X \in fp(\varphi)$ and $(C, \eta), (C', \eta') \in \mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}}$, with $\langle C, \eta|_{fv(\varphi)} \rangle \approx_r \langle C', \eta'|_{fv(\varphi)} \rangle$, if $(C, \eta) \in \pi(X(\mathbf{y}))$, for some $\mathbf{y}$, then $(C', \eta') \in \pi(X(\mathbf{y}))$.

In words, to be saturated for a formula φ, a proposition environment must assign to each free proposition in φ a semantics that respects the equivalence $\approx_r$ over $fv(\varphi)$-pointed configurations. Then, we have the desired result.

Lemma 3.9 (Equisatisfaction in Pointed Configurations with Isomorphic Residuals). *Let $\mathcal{E}$ be a PES, let φ be a formula of $\mathcal{L}_{hp}$, let $\pi \in PEnv_{\mathcal{E}}$ be a proposition environment saturated for φ, and let $(C_1, \eta_1), (C_2, \eta_2) \in \mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}}$. If $\langle C_1, \eta_1|_{fv(\varphi)} \rangle \approx_r \langle C_2, \eta_2|_{fv(\varphi)} \rangle$, then $(C_1, \eta_1) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}$ iff $(C_2, \eta_2) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}$.*

We finally observe that, for strongly regular PESs, the number of equivalence classes of pointed configurations with respect to $\approx_r$ is finite. The converse holds when the PES is boundedly branching, providing an alternative characterisation of strong regularity. This will play a basic role in the proofs of finiteness of tableaux and automata.

Lemma 3.10 (Strong Regularity and Residuals of Pointed Configurations). *A PES $\mathcal{E}$ is strongly regular iff it is boundedly branching and for any fixed finite set V, the equivalence $\approx_r$ is of finite index over V-pointed configurations of $\mathcal{E}$.*

4 A TABLEAU SYSTEM FOR MODEL CHECKING

In this section, we present a tableau system for testing whether a formula of the logic $\mathcal{L}_{hp}$ is satisfied by a semantic model given in the form of an event structure. We prove that the tableau rules are sound and complete over strongly regular event structures.

4.1 Tableau Rules

The tableau system works on a sort of sequents $C, \eta, \Delta \models^{\mathcal{E}} \varphi$, where φ is a formula, $C \in \mathcal{C}(\mathcal{E})$ is a configuration, η is an environment, and Δ is a finite set of definitions of the form $X(\mathbf{x}) = \psi$, with $\mathbf{x}$ a tuple of (distinct) event variables and $fv(\psi) \subseteq \mathbf{x}$. The triple $\Gamma = \langle C, \eta, \Delta \rangle$ is called the *context* and φ the *consequent*. In a tableaux built starting from a closed formula, in Δ each proposition $X(\mathbf{x})$ will be defined as the fixpoint subformula $\alpha X(\mathbf{x}).\, \psi$ where X is quantified. For technical reasons, we also allow Δ to bind propositions to general formulae. This will be used in the proofs of soundness and completeness, where we will need to bind propositions to suitably defined fixpoint approximants.

When $X(\mathbf{x}) = \psi$ is in Δ, we write $X(\mathbf{x}) \in dom(\Delta)$ and we denote the formula ψ as $\Delta(X(\mathbf{x}))$. We assume that for each abstract proposition there is at most one definition, i.e., if $X(\mathbf{x}) = \psi$ and $X(\mathbf{x}') = \psi'$ are in Δ, then $\mathbf{x} = \mathbf{x}'$ and $\psi = \psi'$.

We denote by $\Delta[X(\mathbf{x}) \mapsto \psi]$ the updated definition set obtained from Δ by removing the previous definition of X, if any, and adding $X(\mathbf{x}) = \psi$.

We will work with a subclass of sequents where iterating the substitution of free propositions in the formula φ with their definitions in Δ, we eventually obtain a formula without free propositions. More precisely, let $C, \eta, \Delta \models^{\mathcal{E}} \varphi$ be a sequent. For formulae ψ, ψ' write $\psi \rightarrow_\Delta \psi'$ when $\psi' = \psi[X(\mathbf{x}) := \Delta(X(\mathbf{x}))]$ for some $X(\mathbf{x}) \in dom(\Delta)$. Then the class of sequents we will work with can be defined as follows:

Definition 4.1 (Well-Formed Sequents). A sequent $C, \eta, \Delta \models^{\mathcal{E}} \varphi$ is *well-formed* if $\varphi \rightarrow_\Delta^* \psi$ for some ψ such that $fp(\psi) = \emptyset$. In this case, we denote the formula ψ by $(\varphi)_\Delta$.

It is easy to realise that the formula $(\varphi)_\Delta$ is well-defined, i.e., when it exists it is unique (up to alpha-renaming of event variables).

The truth of a well-formed sequent can be now defined in the obvious way:

Definition 4.2 (Truth). A well-formed sequent $C, \eta, \Delta \models^{\mathcal{E}} \varphi$ is *true* if $(C, \eta) \in \{\!|(\varphi)_\Delta|\!\}^{\mathcal{E}}_\pi$, where π is any proposition environment.

Observe that in the definition above, the proposition environment π is irrelevant, since $C, \eta, \Delta \models^{\mathcal{E}} \varphi$ is well-formed and thus the formula $(\varphi)_\Delta$ does not include any free proposition.

The tableau rules will be of the form

$$\frac{C, \eta, \Delta \;\models^{\mathcal{E}}\; \varphi}{C_1, \eta_1, \Delta_1 \;\models^{\mathcal{E}}\; \varphi_1 \;\ldots\; C_k, \eta_k, \Delta_k \;\models^{\mathcal{E}}\; \varphi_k}\;\delta,$$

where $k > 0$ and δ is a possible side condition required to hold. The intuition is that the truth of the sequent in the premise reduces to the truth of those in the conclusion. In the following, the index $\mathcal{E}$, when clear from the context, will be dropped. Moreover, all sequents will be, sometimes tacitly, assumed to be well-formed.

The tableau rules for the logic $\mathcal{L}_{hp}$, are reported in Table . The rules for propositional connectives are straightforward. For instance, the truth of $\varphi \vee \psi$ is reduced to the truth of either φ or ψ. The context is not altered.

Similarly, the truth of a modal formula is reduced to the truth of the subformula after the modal operator in suitable contexts chosen according to the semantics. For the formula $\langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle\, \varphi$ the rule ($\Diamond$) prescribes the existence of at least one transition leading to a context where φ holds. The rule ($\Box$) for $[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\, \varphi$ requires that all transitions lead to contexts where φ holds. Observe that, working with strongly regular PESs, which are boundedly branching, rule ($\Box$) always has a finite number of sequents in the conclusion.

Rule (Int) reduces the truth of a fixpoint formula $(\alpha Z(\mathbf{x}).\varphi)(\mathbf{y})$ to that of the proposition $Z(\mathbf{y})$ in a context where the definition list Δ is extended by defining $Z(\mathbf{x})$ as the corresponding fixpoint subformula. We will say that the node $C, \eta, \Delta \models (\alpha Z(\mathbf{x}).\varphi)(\mathbf{y})$ *introduces the abstract proposition* Z.

Table 1. The Tableau Rules for Logic $\mathcal{L}_{hp}$

$$(\wedge)\ \frac{C,\eta,\Delta \models \varphi \wedge \psi}{C,\eta,\Delta \models \varphi \quad C,\eta,\Delta \models \psi}$$

$$(\vee_{\mathrm{L}})\ \frac{C,\eta,\Delta \models \varphi \vee \psi}{C,\eta,\Delta \models \varphi} \qquad (\vee_{\mathrm{R}})\ \frac{C,\eta,\Delta \models \varphi \vee \psi}{C,\eta,\Delta \models \psi}$$

$$(\Diamond)\ \frac{C,\eta,\Delta \models \langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z \rangle\!\rangle \varphi}{C',\eta[z \mapsto e],\Delta \models \varphi} \quad \exists\, e.\ C \xrightarrow{\eta(\mathbf{x}),\overline{\eta(\mathbf{y})} < e}_{\mathrm{a}} C'$$

$$(\Box)\ \frac{C,\eta,\Delta \models [\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!] \varphi}{C_1,\eta_1,\Delta \models \varphi \ \ldots\ C_n,\eta_n,\Delta \models \varphi} \quad \{(C_1,\eta_1),\ldots,(C_n,\eta_n)\} = \mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C,\eta)$$

$$(\mathrm{Int})\ \frac{C,\eta,\Delta \models (\alpha Z(\mathbf{x}).\varphi)(\mathbf{y})}{C,\eta,\Delta' \models Z(\mathbf{y})} \quad \Delta' = \Delta[Z(\mathbf{x}) \mapsto \alpha Z(\mathbf{x}).\varphi]$$

$$(\mathrm{Unf}_\alpha)\ \frac{C,\eta,\Delta \models Z(\mathbf{z})}{C,\eta[\mathbf{x} \mapsto \eta(\mathbf{z})],\Delta \models \varphi} \quad \neg\gamma \text{ and } \Delta(Z(\mathbf{x})) = \alpha Z(\mathbf{x}).\varphi$$

Rule (Unf_α) is applied when the consequent is a proposition $Z(\mathbf{z})$: It just unfolds the proposition according to its definition in Δ. When starting from a closed formula, the proposition will be always bound to a fixpoint formula. The component γ in the side condition will be described in the next section. It is called *stop* condition and, as suggested by its name, it is intended to prevent the reduction to continue unboundedly.

The tableau rules satisfy backwards soundness, i.e., we can show that the premise is true when all the sequents in the conclusion are. This property will play a basic role in Section 4.3 for proving the soundness of the tableau system.

Lemma 4.3 (Backwards Soundness). *Every rule of the tableau system is backwards sound.*

4.2 The Stop Condition

The unfolding rule (Unf_α), in absence of any countermeasure, makes the tableau construction procedure non-terminating. To solve this problem, a side condition, called the *stop condition*, is added that prevents the application of the rule when a context is reached that is equivalent, in a suitable sense to be defined, to a context occurring in an ancestor of the current node, for the same fixpoint formula.

The notion of equivalence between contexts has to be chosen carefully not to break the soundness of the technique. Surely two contexts C,η,Δ and C',η',Δ' for a formula φ, to be considered equivalent, must have isomorphic futures, i.e., the residuals $\mathcal{E}[C]$ and $\mathcal{E}[C']$ must be isomorphic as PESs. This is not sufficient, though, since φ can express history-dependent properties that relate the future with the past events. Hence, we additionally ask that event variables of φ are mapped to events in C and C', respectively, which have the same relations (causality and concurrency) with the corresponding futures. More formally, we ask that C and C', seen as configurations pointed by $fv(\varphi)$, have isomorphic residuals (see Definition 3.7).

Given a tableau for a closed formula and a node labelled by $C,\eta,\Delta \models X(\mathbf{y})$, with $X(\mathbf{x}) = \psi$ in Δ, necessarily $\psi = \alpha X(\mathbf{x}).\varphi$ and the node has some ancestor introducing X. We will denote by $\Delta^{\uparrow}(X)$

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\emptyset, \eta, \emptyset \models \langle\!|\mathrm{b}\, x|\!\rangle \varphi}{\{\mathrm{b}^0\}, \eta[x \mapsto \mathrm{b}^0], \emptyset \models \varphi}(\Diamond)}{\{\mathrm{b}^0\}, \eta[x \mapsto \mathrm{b}^0], \{X(x) = \varphi\} \models X(x)}(\mathsf{Int})}{\{\mathrm{b}^0\}, \eta[x \mapsto \mathrm{b}^0], \{X(x) = \varphi\} \models \psi}(\mathsf{Unf}_\nu)}{\{\mathrm{b}^0\}, \eta[x \mapsto \mathrm{b}^0], \{X(x) = \varphi, Y(x) = \psi\} \models Y(x)}(\mathsf{Int})}{\{\mathrm{b}^0\}, \eta[x \mapsto \mathrm{b}^0], \{X(x) = \varphi, Y(x) = \psi\} \models \langle\!|x < \mathrm{b}\, y|\!\rangle X(y) \vee \langle\!|\_\, z|\!\rangle Y(x)}(\mathsf{Unf}_\mu)}{\{\mathrm{b}^0\}, \eta[x \mapsto \mathrm{b}^0], \{X(x) = \varphi, Y(x) = \psi\} \models \langle\!|x < \mathrm{b}\, y|\!\rangle X(y)}(\vee_{\mathsf{L}})}{\{\mathrm{b}^0, \mathrm{b}^1\}, \eta[x \mapsto \mathrm{b}^0, y \mapsto \mathrm{b}^1], \{X(x) = \varphi, Y(x) = \psi\} \models X(y)}(\Diamond)
$$

Fig. 2. A successful tableau for $\langle\!|\mathrm{b}\, x|\!\rangle \nu X(x).\mu Y(x).(\langle\!|x < \mathrm{b}\, y|\!\rangle X(y) \vee \langle\!|\_\, z|\!\rangle Y(x))$ in $\mathcal{E}_{\mathcal{N}}$.

the closest of such ancestors. The notation is slightly abused, since $\Delta^{\uparrow}(X)$ is not determined by Δ, still it is suggestive, since $\Delta^{\uparrow}(X)$ is the node where the definition of X in Δ has been updated more recently. More precisely, one can think that $\Delta^{\uparrow}(.)$ is an additional component of the sequent.

Observe that, once it has been set, the definition $X(\mathbf{x}) = \psi$ of a proposition remains unchanged in the tableau and it will be referred as the definition of X in the tableau. Instead, the component $\Delta^{\uparrow}(X)$ changes at each application of rule (Int).

Definition 4.4 (Stop Condition). The *stop condition* γ for rule (Unf_α) in Table is as follows:

there is an ancestor of the premise $C, \eta, \Delta \models Z(\mathbf{z})$ labelled $C', \eta', \Delta' \models Z(\mathbf{y})$, such that $\Delta^{\uparrow}(Z) = \Delta'^{\uparrow}(Z)$ and $\langle C, \eta[\mathbf{x} \mapsto \eta(\mathbf{z})]|_{\mathbf{x}}\rangle \approx_r \langle C', \eta'[\mathbf{x} \mapsto \eta'(\mathbf{y})]|_{\mathbf{x}}\rangle$.

Informally, the stop condition holds when in an ancestor node in the tableau an instance of the abstract proposition X has been unfolded in an equivalent context, and between such ancestor and the current node, X has not been reintroduced. This intuitively means that the two X's refer to the same fixpoint instance. In this case, we can safely avoid to continue along this path. Instead, when the stop condition fails, it makes sense to further unfold the fixpoint, since the current context is still "different enough" from those previously encountered. Note that the equivalence of contexts is checked after renaming the variables to those associated with X in the fixpoint formula quantifying the proposition.

In References [11, 12, 45], the finiteness of the model is an essential requirement for the finiteness of the tableaux. In our case, as already mentioned, the PES model is commonly infinite, even for finite-state systems. However, working with strongly regular PESs, thanks to the fact that they are boundedly branching, only a finite part of the model is used in every step of the tableau construction. Moreover after going sufficiently in depth, thanks to the strong regularity property, the PES starts "repeating" cyclically the same structure, something that will allow us to show that the stop condition eventually holds. These facts will allow to conclude, later in Section 4.3, that the satisfaction of a formula can be established by checking only a finite part of the PES.

As an example, consider the formula $\langle\!|\mathrm{b}\, x|\!\rangle \nu X(x).\, \mu Y(x).(\langle\!|x < \mathrm{b}\, y|\!\rangle X(y) \vee \langle\!|\_\, z|\!\rangle Y(x))$ from Section 3.2, which was claimed to be satisfied by $\mathcal{E}_{\mathcal{N}}$ in Figure 1(a). Write it as $\langle\!|\mathrm{b}\, x|\!\rangle \varphi$, where $\varphi = \nu X(x).\psi$ and $\psi = \mu Y(x).(\langle\!|x < \mathrm{b}\, y|\!\rangle X(y) \vee \langle\!|\_\, z|\!\rangle Y(x))$. A tableau for the sequent $\emptyset, \eta, \emptyset \models \langle\!|\mathrm{b}\, x|\!\rangle \varphi$ is given in Figure 2. The unfolding rule cannot be applied to the bottom sequent $\{\mathrm{b}^0, \mathrm{b}^1\}, \eta[x \mapsto \mathrm{b}^0, y \mapsto \mathrm{b}^1], \{X(x) = \varphi, Y(x) = \psi\} \models X(y)$ because the stop condition holds. In fact, there is an ancestor, namely, $\{\mathrm{b}^0\}, \eta[x \mapsto \mathrm{b}^0], \{X(x) = \varphi\} \models X(x)$, having the same introduction node for the abstract proposition X and such that $\langle\{\mathrm{b}^0, \mathrm{b}^1\}, \eta[x \mapsto \mathrm{b}^1]|_x\rangle \approx_r \langle\{\mathrm{b}^0\}, \eta[x \mapsto \mathrm{b}^0]|_x\rangle$.

4.3 Soundness and Completeness of the Tableau System

In this section, we show that the truth of a formula φ over a strongly regular event structure can be reduced to the existence of a suitably defined successful tableau.

Definition 4.5 (Tableau for a Closed Formula). Let $\mathcal{E}$ be a PES. Given a closed formula φ in $\mathcal{L}_{hp}$, a tableau for φ is a tableau for a sequent $\emptyset, \eta, \emptyset \models^{\mathcal{E}} \varphi$, where $\eta \in Env_{\mathcal{E}}$.

Note that the environment η can be chosen arbitrarily: It is irrelevant, since the formula has no free event variables.

A maximal tableau is a tableau where all leaves are labelled by sequents to which no rule applies. We first clarify when a maximal tableau is considered successful.

Definition 4.6 (Successful Tableau). A *successful* tableau is a finite maximal tableau where every leaf is labelled by a sequent $C, \eta, \Delta \models^{\mathcal{E}} \varphi$ such that one of the following holds:

(1) $\varphi = \mathsf{T}$
(2) $\varphi = [\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\, \psi$
(3) $\varphi = Z(\mathbf{y})$ and $\Delta(Z(\mathbf{x})) = \nu Z(\mathbf{x}).\psi$.

An example of successful tableau can be found in Figure 2.

Observe that the choice of the rule to be applied at a step of the construction of a tableau is non-deterministic in the case of $\langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z \rangle\!\rangle \varphi$ and $\varphi \vee \psi$. This means that there can be various maximal tableaux for the same sequent. However, when we work on strongly regular PESs, the fact that they are boundedly branching ensures that at each step the number of possible non-deterministic choices is finite and bounded. This later plays a role for deducing that there can be only a finite number of maximal tableaux for each given sequent.

We first focus on the finiteness issue and then move on to the soundness and completeness of the technique.

4.3.1 Finiteness. We aim at proving that all tableaux for a sequent $C, \eta, \Delta \models^{\mathcal{E}} \varphi$ are finite. A first basic observation is that an infinite tableau for a sequent $C, \eta, \Delta \models^{\mathcal{E}} \varphi$ necessarily includes a path where the same proposition is unfolded infinitely many times without being reintroduced. The proof relies on some properties of the order $\sqsubseteq_d^*$ over propositions (see Definition 3.2).

Lemma 4.7 (Fixpoint Introduction). *Let $\mathcal{E}$ be a PES and let τ be a tableau for a closed formula φ. Let n be any node in the tableau labelled by $C, \eta, \Delta \models X(\mathbf{x})$ for some $X(\mathbf{x}) \in \mathcal{X}$.*

(1) If n has a descendant n' labelled by $C', \eta', \Delta' \models Y(\mathbf{y})$, for some $Y(\mathbf{y}) \in \mathcal{X}$, and Y is not introduced between n and n', then $X \sqsubseteq_d^* Y$.
(2) If n has a descendant n' that introduces X, then there is a node n'' between n and n' with consequent $Y(\mathbf{y})$ such that $X \sqsubseteq_d Y$ (hence, $X \sqsubseteq_d^* Y$ and $X \neq Y$).

We already observed that the fact that strongly regular PESs are boundedly branching implies that also the constructed tableaux are boundedly branching. Then, by König's lemma, an infinite tableau necessarily includes an infinite path. Using Lemma 4.7, it is not difficult to show that such path includes infinitely many repetitions of the same abstract proposition without introductions, i.e., we get the following property:

Lemma 4.8 (Infinite Occurrences of Propositions in Tableaux). *Let $\mathcal{E}$ be a finitely branching PES. An infinite tableau for a closed formula φ contains an infinite path where some abstract proposition Z occurs (and thus is unfolded) infinitely often without being introduced.*

We can finally deduce the finiteness of the tableaux for a sequent that in turn implies that the number of tableaux is finite. This fact is essential for termination of the model-checking procedure.

Theorem 4.9 (tableaux finiteness). *For a strongly regular PES $\mathcal{E}$ and a closed formula φ, every tableau for a sequent $C, \eta, \Delta \models^{\mathcal{E}} \varphi$ is finite. Hence, the number of tableaux for $C, \eta, \Delta \models^{\mathcal{E}} \varphi$ is finite.*

Roughly the proof, detailed in the Appendix, proceeds as follows: An infinite tableau, by Lemma 4.8, would contain an abstract proposition occurring infinitely often in a path, without being reintroduced. By using Lemma 3.10, we would deduce that the proposition would occur infinitely often within contexts with isomorphic pointed residuals. This would lead to a contradiction, since at the first repetition the stop condition (see Definition 4.4) would have prescribed of terminating the branch. From the fact that all tableaux are finite and boundedly branching, we conclude that there are finitely many of them.

4.3.2 Soundness and Completeness. We can now prove the soundness and completeness of the tableau system. For this, we use the possibility of reducing the satisfaction of a formula to the satisfaction of a finite approximant. While on finite models this is immediate, over event structures where the space of configurations is infinite, this does not work, in general. In fact, due to alternation, the semantic function associated with a formula might be non-continuous, hence it is not guaranteed that fixpoints will be reached in at most ω steps. However, here, the result can be obtained by exploiting the finiteness of pointed configurations up to $\approx_r$ in strongly regular PESs.

Definition 4.10 (Finite Approximant). Let $\varphi = \nu Z(\mathbf{x}).\psi$ be a fixpoint formula. The *ith approximant* φ^i, for $i \in \mathbb{N}$, is inductively defined by $\varphi^0 = \mathsf{T}$ and $\varphi^{i+1} = \psi[Z(\mathbf{x}) := \varphi^i]$. Dually, if $\varphi = \mu Z(\mathbf{x}).\psi$, then we define $\varphi^0 = \mathsf{F}$ and $\varphi^{i+1} = \psi[Z(\mathbf{x}) := \varphi^i]$.

We next observe that, as anticipated, despite the fact that the state space of configurations is infinite, for strongly regular PESs, all formulae, even those with alternation, reach the fixpoint after a finite number of steps.

Lemma 4.11 (Finite Approximants Properties). *Let $\mathcal{E}$ be a strongly regular PES, let $\pi \in PEnv_{\mathcal{E}}$ be a saturated proposition environment, and let $\varphi = \alpha Z(\mathbf{x}).\,\psi$ be a fixpoint formula. Then there exists $i \in \mathbb{N}$ such that $\{\!|\varphi|\!\}^{\mathcal{E}}_{\pi} = \{\!|\varphi^i|\!\}^{\mathcal{E}}_{\pi}$. Hence, for any configuration $C \in \mathcal{C}(\mathcal{E})$ and environment $\eta \in Env_{\mathcal{E}}$:*

(1) if $\varphi = \nu Z(\mathbf{x}).\psi$ and $(C, \eta) \notin \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}$, then $(C, \eta) \in \{\!|\varphi^n|\!\}^{\mathcal{E}}_{\pi} \setminus \{\!|\varphi^{n+1}|\!\}^{\mathcal{E}}_{\pi}$ for some $n \leq i$;
(2) if $\varphi = \mu Z(\mathbf{x}).\psi$ and $(C, \eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}$, then $(C, \eta) \in \{\!|\varphi^{n+1}|\!\}^{\mathcal{E}}_{\pi} \setminus \{\!|\varphi^n|\!\}^{\mathcal{E}}_{\pi}$ for some $n \leq i$.

We can now show that the model-checking problem reduces to the construction of a successful tableau. We prove separately soundness and completeness by resorting to some variations of the tableau system in Table that we call ν-pseudo-tableau and μ-pseudo-tableau systems, respectively.

Definition 4.12 (ν-pseudo-tableaux). The ν-pseudo-tableau system is obtained from that in Table , by working with sequents $C, \eta, \Delta \models \varphi$ where the definition list Δ contains definitions $Z(\mathbf{x}) = (\nu Z(\mathbf{x}).\varphi)^n$ and inserting the new unfolding rule below:

$$(\mathsf{Unf}^{\mathrm{a}}_{\nu}) \quad \frac{C, \eta, \Delta \;\models\; Z(\mathbf{z})}{C, \eta[\mathbf{x} \mapsto \eta(\mathbf{z})], \Delta \;\models\; \varphi} \quad \neg\gamma \text{ and } \Delta(Z(\mathbf{x})) = (\nu Z(\mathbf{x}).\varphi)^n \,.$$

The notion of successful ν-pseudo-tableau is as in Definition 4.6, but we additionally allow a leaf to be labelled by $C, \eta, \Delta \models^{\mathcal{E}} Z(\mathbf{z})$ with $\Delta(Z(\mathbf{x})) = (\nu Z(\mathbf{x}).\psi)^n$ and $(C, \eta) \in \{\!|(Z(\mathbf{z}))_{\Delta}|\!\}_{\pi}$ for $\pi \in PEnv$.

It can be easily seen that also rule $(\mathsf{Unf}^{\mathrm{a}}_{\nu})$ is backwards sound. In fact, assume that the sequent in the conclusion is true, i.e., $(C, \eta[\mathbf{x} \mapsto \eta(\mathbf{z})]) \in \{\!|(\varphi)_{\Delta}|\!\}^{\mathcal{E}}_{\pi} = \{\!|(\varphi[Z(X) := (\nu Z(\mathbf{x}).\varphi)^n])_{\Delta}|\!\}^{\mathcal{E}}_{\pi} = \{\!|(\nu Z(\mathbf{x}).\varphi)^{n+1}_{\Delta}|\!\}^{\mathcal{E}}_{\pi}$, for some $\pi \in PEnv$. Now observe that $(Z(\mathbf{x}))_{\Delta} = (\nu Z(\mathbf{x}).\varphi)^n_{\Delta}$. Hence, $(C, \eta[\mathbf{x} \mapsto \eta(\mathbf{z})]) \in \{\!|(\nu Z(\mathbf{x}).\varphi)^{n+1}_{\Delta}|\!\}^{\mathcal{E}}_{\pi} \subseteq \{\!|(\nu Z(\mathbf{x}).\varphi)^n_{\Delta}|\!\}^{\mathcal{E}}_{\pi} = \{\!|(Z(\mathbf{x}))_{\Delta}|\!\}^{\mathcal{E}}_{\pi} = \{\!|Z(\mathbf{z})|\!\}^{\mathcal{E}}_{\pi}[\mathbf{x}/\mathbf{z}]$. Thus, $(C, \eta) \in \{\!|Z(\mathbf{z})|\!\}^{\mathcal{E}}_{\pi}$, which means that the sequent in the premise is true.

Soundness will be an immediate consequence of the following technical result that implies that there cannot be successful ν-pseudo-tableaux with false leaves, that is, leaves labelled by sequents that are not true.

Lemma 4.13 (Shortening ν-Pseudo-Tableaux). *Let $\mathcal{E}$ be a strongly regular PES and let τ be a successful ν-pseudo-tableau. If τ has a false leaf and for all false leaves $C, \eta, \Delta \models X(\mathbf{z})$, the node $\Delta^{\uparrow}(X)$ is in τ, then there exists a successful ν-pseudo-tableaux τ', strictly smaller than τ, with a false leaf and where for all false leaves $C, \eta, \Delta \models X(\mathbf{z})$, the node $\Delta^{\uparrow}(X)$ is in τ'.*

The above lemma immediately implies that a successful ν-pseudo-tableau τ where all false leaves $C, \eta, \Delta \models X(\mathbf{z})$ are such that $\Delta^{\uparrow}(X)$ is in τ actually cannot include false leaves. Since a successful tableau for a closed formula is a successful ν-pseudo-tableau with the above property, we conclude that it cannot have false leaves and thus, by backwards soundness, all nodes including the root must be true, i.e., we have the following:

Lemma 4.14 (Soundness). *Let $\mathcal{E}$ be a strongly regular PES and φ be a closed formula of $\mathcal{L}_{hp}$. If φ has a successful ν-pseudo-tableau (hence, in particular, if it has a successful tableau), then $\mathcal{E}$ satisfies φ.*

For completeness, we resort to a dual variation of basic tableaux, referred to as μ-pseudo-tableaux.

Definition 4.15 (μ-Pseudo-Tableaux). The μ-pseudo-tableau system is obtained from that in Table , by working with sequents $C, \eta, \Delta \models \varphi$ where Δ contains definitions $Z(\mathbf{x}) = (\mu Z(\mathbf{x}).\varphi)^n$ and replacing (Unf_μ) by the new unfolding rule below:

$$(\mathsf{Unf}^{\mathsf{a}}_{\mu})\ \frac{C, \eta, \Delta \models Z(\mathbf{z})}{C, \eta', \Delta' \models \psi} \quad \neg\gamma \text{ and } \Delta(Z(\mathbf{x})) = \mu Z(\mathbf{x}).\psi \text{ or } (\mu Z(\mathbf{x}).\psi)^n,$$

where $\eta' = \eta[\mathbf{x} \mapsto \eta(\mathbf{z})]$ and $\Delta' = \Delta[Z(\mathbf{x}) \mapsto (\mu Z(\mathbf{x}).\psi)^k]$ with $k \in \mathbb{N}$ such that

$$(C, \eta') \in \{\!|(\mu Z(\mathbf{x}).\psi)^{k+1}_{\Delta}|\!\}_{\pi} \setminus \{\!|(\mu Z(\mathbf{x}).\psi)^{k}_{\Delta}|\!\}_{\pi}, \tag{1}$$

for $\pi \in PE n \nu$. Moreover, if $\Delta(Z(\mathbf{x})) = (\mu Z(\mathbf{x}).\psi)^n$, then it is required that $k < n$. The stop condition γ is the usual one (see Definition 4.4).

The notion of successful μ-pseudo-tableau is exactly as in Definition 4.6.

Notice that in rule $(\mathsf{Unf}^{\mathsf{a}}_{\mu})$ the requirement (1) posed on k is in fact a semantic condition. This inclusion may seem strange in a tableau rule, however, it should not be worrisome, since μ-pseudo-tableaux are solely devoted to prove the correctness of the method.

An easy but crucial observation is that rule $(\mathsf{Unf}^{\mathsf{a}}_{\mu})$ is applicable exactly in the situations in which the premise is true (as detailed in the Appendix). Additionally, rule $(\mathsf{Unf}^{\mathsf{a}}_{\mu})$ preserves the well-formedness of the sequents when applied to a true premise, i.e., if the premise is well-formed and true, then the conclusion is well-formed. This immediately follows from the observation that for $k < n$ it holds $fp((\mu Z(\mathbf{x}).\psi)^k) \subseteq fp((\mu Z(\mathbf{x}).\psi)^n) \subseteq fp(\mu Z(\mathbf{x}).\psi)$.

Theorem 4.9 can be easily extended to μ-pseudo-tableaux, i.e., we can prove that every μ-pseudo-tableau is finite. Moreover, notice that a successful μ-pseudo-tableau can be transformed into a successful tableau simply by replacing, in all sequents, the definition list Δ with Δ' where each approximant is substituted by the corresponding least fixpoint. More precisely, for all $X(\mathbf{x}) \in dom(\Delta)$, we let $\Delta'(X) = \mu X(\mathbf{x}).\psi$ if $\Delta(X(\mathbf{x})) = (\mu X(\mathbf{x}).\psi)^n$, and $\Delta'(X(\mathbf{x})) = \Delta(X(\mathbf{x}))$, otherwise. Using this fact one can prove completeness.

Lemma 4.16 (Completeness). *Let $\mathcal{E}$ be a strongly regular PES and let φ be a closed formula of $\mathcal{L}_{hp}$. If $\mathcal{E}$ satisfies φ, then φ has a successful μ-pseudo tableau and thus a successful tableau.*

Combining the previous lemmata, we get to the desired result.

Theorem 4.17 (Soundness and Completeness of the Tableau System). *Given a strongly regular PES $\mathcal{E}$ and a closed formula φ of $\mathcal{L}_{hp}$, the formula φ has successful tableau if and only if $\mathcal{E}$ satisfies φ.*

5 AUTOMATA-BASED MODEL CHECKING

On the way to a practical implementation of a model-checker for $\mathcal{L}_{hp}$, in this section, we develop an automata-theoretic approach to model checking $\mathcal{L}_{hp}$. We show how the model-checking problem for $\mathcal{L}_{hp}$ on strongly regular PESs can be reduced to the non-emptiness of the language of suitably generated nondeterministic parity tree automata. The automaton that naturally arises from a PES and a formula has an infinite number of states. We discuss how it can be quotiented to a finite automaton accepting the same language, which can thus be used for model-checking purposes.

5.1 Infinite Parity Tree Automata and Their Quotient

Automata on infinite trees are a powerful tool for various problems in the setting of branching temporal logics. Here, we focus on nondeterministic parity tree automata [30], with some non-standard features. We work on k-trees (rather than on binary trees), a choice that will simplify the presentation, and we allow for possibly infinite state automata.

When automata are used for model-checking it is standard to restrict to unlabelled trees. A *k-bounded branching tree* or *k-tree*, for short, is a subset $\mathcal{T} \subseteq [1,k]^\star$, such that

(1) $\mathcal{T}$ is prefix closed, i.e., if $uv \in \mathcal{T}$ then $u \in \mathcal{T}$
(2) $u1 \in \mathcal{T}$ for all $u \in \mathcal{T}$
(3) for all $i \in [2,k]$ if $ui \in \mathcal{T}$ then $u(i-1) \in \mathcal{T}$.

Elements of $\mathcal{T}$ are the nodes of the tree. The empty string ϵ corresponds to the root. A string of the form ui corresponds to the ith child of node u. Hence, by (2) each branch is infinite and by (3) the presence of the ith child implies the presence of the jth children for $j \leq i$.

Definition 5.1 (Nondeterministic Parity Automaton). A *k-bounded* nondeterministic parity tree automaton (NPA) is a tuple $\mathcal{A} = \langle Q, \rightarrow, q_0, \mathcal{F} \rangle$ where Q is a (possibly infinite) set of states, $\rightarrow \subseteq Q \times \bigcup_{i=1}^{k} Q^i$ is the *transition relation*, $q_0 \in Q$ is the *initial state*, and $\mathcal{F} = (F_0, \ldots, F_h)$ is the *acceptance condition*, where $F_0, \ldots, F_h \subseteq Q$ are mutually disjoint subsets of states.

Transitions are written as $q \rightarrow (q_1, \ldots, q_m)$ instead of $(q, (q_1, \ldots, q_m)) \in \rightarrow$.

The acceptance condition $\mathcal{F} = (F_0, \ldots, F_h)$ defines the priority of states of the automaton, where a state $q \in F_i$ is assigned priority i. Sets $F_0, \ldots, F_h$ are required to be mutually disjoint, since every state can have at most one priority.

Definition 5.2 ((Accepting) Runs). Given a k-tree $\mathcal{T}$, a *run* of $\mathcal{A}$ on $\mathcal{T}$ is a labelling of $\mathcal{T}$ over the states $r : \mathcal{T} \rightarrow Q$ consistent with the transition relation, i.e., such that $r(\epsilon) = q_0$ and for all $u \in \mathcal{T}$, with m children, there is a transition $r(u) \rightarrow (r(u1), \ldots, r(um))$ in $\mathcal{A}$. A *path* in the run r is an infinite sequence of states $p = (q_0, q_1, \ldots)$ labelling a complete path from the root in the tree. It is called *accepting* if there exists an even number $l \in [0,h]$ such that the set $\{j \mid q_j \in F_l\}$ is infinite and the set $\{j \mid q_j \in \bigcup_{l<i\leq h} F_i\}$ is finite. In this case, we denote such l by $\mathcal{F}(p)$. The run r is *accepting* if all paths in r are accepting.

In words, given a run r, a path p in r is accepting if there are indices i such that states in F_i occur infinitely often in the path p and the largest of such indices is even. The run is accepting if all paths in it are.

In the sequel, we will also refer to a *path in an NPA* $\mathcal{A}$, meaning a possibly infinite sequence $p = (q_1, q_2, \ldots)$ such that for all $i < |p|$ there is a transition $q_i \to (q'_1, \ldots, q_{i+1}, \ldots, q'_m)$. Note that, given a run r of an automaton $\mathcal{A}$, each path in r is also a path in $\mathcal{A}$.

Definition 5.3 (Language of an NPA). Let $\mathcal{A}$ be an NPA. The language of $\mathcal{A}$, denoted by $L(\mathcal{A})$, consists of the trees $\mathcal{T}$ that admit an accepting run.

Observe that for a k-bounded NPA, the language $L(\mathcal{A})$ consists of a set of k-trees.

The possibility of having an infinite number of states in an NPA and the associated acceptance condition are non-standard. However, it is easy to see that whenever an NPA is finite, the acceptance condition coincides with the standard one requiring a single state with maximal even priority to occur infinitely often in each path. In fact, for finite NPAs, all sets in the acceptance condition $\mathcal{F}$ are finite and thus asking that for a path $p = (q_0, q_1, \ldots)$ the set $\{j \mid q_j \in F_i\}$ is infinite amounts to asking that some state in F_i repeats infinitely often in the path p.

The NPA naturally associated with a formula and event structure will be infinite. To have an effective procedure for checking the satisfaction of a formula, we will build a suitable quotient of the NPA, with respect to an equivalence that preserves emptiness.

For this reason, we next introduce a notion of bisimulation over NPAs and observe that bisimulation equivalences preserve the language of NPAs (and thus in particular language emptiness). An analogous notion is studied in Reference [1] in the setting of nondeterministic tree automata over finite trees.

Definition 5.4 (Bisimulation). Given an NPA $\mathcal{A}$, a symmetric relation $R \subseteq Q \times Q$ over the set of states is a bisimulation whenever for all $(q, q') \in R$

(1) for all $i \in [0, h]$, $q \in F_i$ iff $q' \in F_i$;
(2) if $q \to (q_1, \ldots, q_m)$, then $q' \to (q'_1, \ldots, q'_m)$ with $(q_i, q'_i) \in R$ for $i \in [1, m]$.

Given an NPA $\mathcal{A}$ and an equivalence $\equiv$ on the set of states that is a bisimulation, we define the quotient as $\mathcal{A}_{/\equiv} = \langle Q_{/\equiv}, \to_{/\equiv}, [q_0]_{\equiv}, \mathcal{F}_{/\equiv} \rangle$ where $[q]_{\equiv} \to_{/\equiv} ([q_1]_{\equiv}, \ldots, [q_m]_{\equiv})$ if $q \to (q_1, \ldots, q_m)$ and $\mathcal{F}_{/\equiv} = (F_{0/\equiv}, \ldots, F_{h/\equiv})$. Note that by condition (1) in Definition 5.4 above, the acceptance condition is well-defined, i.e., all $F_{i/\equiv}$ are pairwise-disjoint. We can show that an NPA and its quotient accept exactly the same language.

Theorem 5.5 (Language Preservation). *Let $\mathcal{A}$ be an NPA and let $\equiv$ be an equivalence on the set of states, which is a bisimulation. Then $L(\mathcal{A}_{/\equiv}) = L(\mathcal{A})$.*

Since NPAs are nondeterministic, different runs (possibly infinitely many) can exist for the same input tree. Still, the non-emptiness problem, also for our k-ary variant, is decidable when the number of states is finite (and solvable by a corresponding parity game [26]).

5.2 NPAs for Model Checking

We show how, given a PES and a closed formula of $\mathcal{L}_{hp}$, we can build an NPA in a way that, for strongly regular PESs, the satisfaction of φ in $\mathcal{E}$ reduces to the non-emptiness of the language of the NPA. The construction is inspired by that in Reference [16] for the mu-calculus. The automaton is typically infinite, but we discuss how an effective model-checking procedure can be obtained by quotienting the infinite NPA to a finite one.

Definition 5.6 (NPA for a Formula). Let $\mathcal{E}$ be a (boundedly branching) PES and let φ be a closed formula of $\mathcal{L}_{hp}$. The NPA for $\mathcal{E}$ and φ is $\mathcal{A}_{\mathcal{E},\varphi} = \langle Q, \to, q_0, \mathcal{F} \rangle$ defined as follows: The set of states $Q \subseteq \mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}} \times \mathrm{sf}(\varphi)$ is defined as $Q = \{(C, \eta, \psi) \mid \eta(fv(\psi)) \subseteq C\}$. The initial state $q_0 = (\emptyset, \eta, \varphi)$, for some chosen environment $\eta \in Env_{\mathcal{E}}$. The transition relation is defined, for any state $q = (C, \eta, \psi) \in Q$, by:

- if $\psi = \mathsf{T}$ or $\psi = \mathsf{F}$, then $q \to (q)$
- if $\psi = \psi_1 \wedge \psi_2$, then $q \to (q_1, q_2)$ where $q_i = (C, \eta, \psi_i)$, $i \in \{1, 2\}$
- if $\psi = \psi_1 \vee \psi_2$, then $q \to (q_1)$ and $q \to (q_2)$ where $q_i = (C, \eta, \psi_i)$, $i \in \{1, 2\}$
- if $\psi = [\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\psi'$ and $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C, \eta) = \{(C_1, \eta_1), \ldots, (C_n, \eta_n)\} \neq \emptyset$, then $q \to (q_1, \ldots, q_n)$ where $q_i = (C_i, \eta_i, \psi')$ for $i \in [1, n]$, otherwise $q \to (q)$
- if $\psi = \langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle\psi'$ and $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C, \eta) = \{(C_1, \eta_1), \ldots, (C_n, \eta_n)\} \neq \emptyset$, then $q \to (q_i)$ where $q_i = (C_i, \eta_i, \psi')$ for $i \in [1, n]$, otherwise $q \to (q)$
- if $\psi = (\alpha X(\mathbf{x}).\psi')(\mathbf{y})$, then $q \to (q')$ where $q' = (C, \eta, X(\mathbf{y}))$
- if $\psi = X(\mathbf{z})$ and $\psi' \in \mathrm{sf}(\varphi)$ is the unique subformula such that $\psi' = (\alpha X(\mathbf{x}).\psi'')(\mathbf{y})$, then $q \to (q')$ where $q' = (C, \eta[\mathbf{x} \mapsto \eta(\mathbf{z})], \psi'')$.

The acceptance condition is $\mathcal{F} = (F_0, \ldots, F_h)$ where $h = \mathrm{ad}(\varphi) + 1$ and the sets F_i are as follows: Consider $A_0, \ldots, A_h \subseteq \mathrm{sf}(\varphi)$ such that for $i \in [0, h]$, if i is even (odd), then A_i contains exactly all propositions quantified in ν-subformulae (μ-subformulae) with alternation depth i or $i - 1$. Then $F_0 = (\mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}} \times (A_0 \cup \{\mathsf{T}\})) \cup B$ where $B = \{(C, \eta, [\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\psi) \mid \mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C, \eta) = \emptyset\}$ is the set of all subformulae of φ in a context where they are trivially true, and $F_i = \mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}} \times A_i$, for $i \in [1, h]$.

States of $\mathcal{A}_{\mathcal{E},\varphi}$ are triples (C, η, φ) consisting of a configuration C, an environment η, and a subformula ψ of the original formula φ. (The environment η fixed for the initial state is irrelevant, since formula φ is closed.) The intuition is that a transition reduces the truth of a formula in a state to that of subformulae in possibly updated states. It can just decompose the formula, as it happens for $\wedge$ or $\vee$, check the satisfaction of a modal operator, thus changing the state consequently, or unfold a fixpoint. Whenever $q = (C, \eta, \psi)$ with $\psi = (\alpha X(\mathbf{x}).\psi')(\mathbf{y})$ and thus $q \to (q')$ with $q' = (C, \eta, X(\mathbf{y}))$, we say that q *introduces* X. If $\psi = X(\mathbf{z})$ and thus $q \to (q')$ where $q' = (C, \eta[\mathbf{x} \mapsto \eta(\mathbf{z})], \psi'')$, with ψ' the body of the fixpoint subformula quantifying X in φ, we say that q *unfolds* X. The automaton $\mathcal{A}_{\mathcal{E},\varphi}$ is bounded (by the branching bound of the PES) but normally infinite (whenever the PES $\mathcal{E}$ is infinite and the formula φ includes some non-trivial fixpoint).

We next show that for a strongly regular PES the truth of the formula φ on the PES $\mathcal{E}$ reduces to the non-emptiness of the language of $\mathcal{A}_{\mathcal{E},\varphi}$. A basic ingredient is an equivalence that can be defined on the NPA relying on the notion of residual of pointed configuration (Definition 3.7).

Definition 5.7 (Future Equivalence). Let $\mathcal{E}$ be a PES, φ be a formula, and let $q_i = (C_i, \eta_i, \psi_i)$, $i \in \{1, 2\}$ be two states of the NPA $\mathcal{A}_{\mathcal{E},\varphi}$. We say that q_1 and q_2 are *future-equivalent*, written $q_1 \approx_f q_2$, if there exists a formula ψ and substitutions $\sigma_i : fv(\psi) \to fv(\psi_i)$ such that $\psi\sigma_i = \psi_i$, for $i \in \{1, 2\}$, and the $fv(\psi)$-pointed configurations $\langle C_i, \eta_i \circ \sigma_i \rangle$ have isomorphic pointed residuals.

Intuitively, two states are equivalent if they involve the same subformula (up to renaming of the event variables) and the configurations, pointed by the free variables in the formulae, have isomorphic residuals. Future equivalence can be shown to be a bisimulation on the NPA $\mathcal{A}_{\mathcal{E},\varphi}$ in the sense of Definition 5.4 and, whenever $\mathcal{E}$ is strongly regular, it is of finite index.

Lemma 5.8 ($\approx_f$ is a Bisimulation). *Let $\mathcal{E}$ be a strongly regular PES and let φ be a closed formula of $\mathcal{L}_{hp}$. Then the future equivalence $\approx_f$ on $\mathcal{A}_{\mathcal{E},\varphi}$ is a bisimulation and it is of finite index.*

As an example, consider the formula $\langle\!|\mathrm{b}\, x|\!\rangle\varphi$ from Section 3.2, where $\varphi = \nu X(x).\psi$ and $\psi = \mu Y(x).(\langle\!|x < \mathrm{b}\, y|\!\rangle X(y) \vee \langle\!|\_ z|\!\rangle Y(x))$. The automaton $\mathcal{A}_{\mathcal{E}_\mathcal{N},\langle\!|\mathrm{b}\, x|\!\rangle\varphi}$ built for model-checking such formula in the PES $\mathcal{E}_\mathcal{N}$ of Figure 1(a) would be infinite. The automaton resulting as the quotient of $\mathcal{A}_{\mathcal{E}_\mathcal{N},\langle\!|\mathrm{b}\, x|\!\rangle\varphi}$ with respect to the future equivalence $\approx_f$ is finite. A fragment of such automaton is reported in Figure 3. The two curly lines represent transitions that "appear" as an effect of the quotient operation. For instance, in the infinite NPA, state q_6 would have had a single transition to

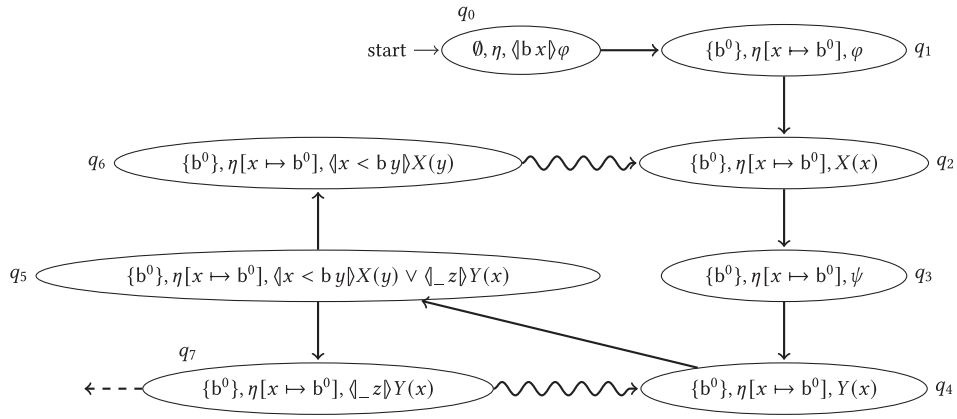


Fig. 3. Quotient automaton obtained from the infinite automaton $\mathcal{A}_{\mathcal{E}_\mathcal{N}, \langle\!|\mathrm{b}\, x|\!\rangle\varphi}$ via future equivalence.

state $(\{\mathrm{b}^0, \mathrm{b}^1\}, \eta[x \mapsto \mathrm{b}^0, y \mapsto \mathrm{b}^1], X(y))$. Since the latter is future-equivalent to q_2, in the quotient it is merged to q_2 and the transition from q_6 instead loops back to the corresponding equivalence class (represented by q_2). From q_7 there is also a transition, represented by a dashed line, to state $(\{\mathrm{b}^0, \mathrm{c}^0\}, \eta[x \mapsto \mathrm{b}^0, z \mapsto \mathrm{c}^0], Y(x))$. This state is not future-equivalent to any of the previous ones, and it would lead to the rest of the reachable states of the automaton, not shown in the figure. However, the states displayed are already sufficient to prove that the language of the NPA is non-empty. Indeed, the sequence of states $(q_0, q_1, (q_2, q_3, q_4, q_5, q_6)^*)$, where the loop $q_2, q_3, q_4, q_5, q_6, q_2$ repeats indefinitely, represents an accepting run. In fact, the state with maximal priority repeating infinitely often is q_2 and its priority is even.

We already hinted at the similarity between tableau rules and transitions in the automaton associated with a PES and a formula. We next formalise this relation by showing that future equivalence can be used to prune runs of the automaton $\mathcal{A}_{\mathcal{E},\varphi}$ in a way that a suitably chosen accepting run, after pruning, will correspond to a successful tableau.

Definition 5.9 (Pruned k-tree). Let $\mathcal{E}$ be a PES, let φ be a closed formula of $\mathcal{L}_{hp}$, and let r be a run of the NPA $\mathcal{A}_{\mathcal{E},\varphi}$ on a k-tree $\mathcal{T}$. Given a path $p = (u_0, u_1, \ldots)$ in $\mathcal{T}$, we call a node u_j a *repetition* if one of the following conditions holds:

(1) the formula in $r(u_j)$ is of the kind T, F, or $[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\psi'$, $\langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle\psi'$ with $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}z}(C, \eta) = \emptyset$ (hence, starting from u_j the path consists of the repetition of the state $r(u_j)$);
(2) the formula in $r(u_j)$ is a proposition $X(\mathbf{x})$ and there is $i < j$ such that $r(u_i) \approx_f r(u_j)$ and X is not introduced between u_i and u_j.

In case (1), we let $\Omega(u_j) = u_j$, while in case (2), we let $\Omega(u_j) = u_i$ where $i < j$ is the minimal index such that u_i satisfies the condition and we call $\Omega(u_j)$ the *repetition witness* for u_j. If $r(u_j) \in F_l$ for some l, we say that u_j is a repetition of priority l. The *pruned run* $\mathcal{T}^{(r)}$ is the largest subtree of r where repetitions have no successor.

The pruned run is obtained by cutting each path at the first repetition, hence in the pruned run each leaf has a repetition witness.

An adaptation of the results developed for tableaux in Section 4.3.1 allows us to prove that the pruning of a run is always finite. The details can be found in Appendix C.2.

Lemma 5.10 (Pruned Runs are Finite). *Let $\mathcal{E}$ be a strongly regular PES, let φ be a closed formula of $\mathcal{L}_{hp}$, and let $\mathcal{A}_{\mathcal{E},\varphi}$ be the corresponding NPA. For any run r of $\mathcal{A}_{\mathcal{E},\varphi}$ on a k-tree $\mathcal{T}$, the corresponding pruned run $\mathcal{T}^{(r)}$ is finite.*

Observe that by pruning an accepting run, we could still get something that does not correspond to a successful tableau. The crux is that, even though a path is accepting, and thus it includes repetitions over states of maximum priority that is even, it could also include, at the beginning, some repetitions over least fixpoints. In this case, the pruned run will have the corresponding states, with odd priority, at some leaves. For instance, consider the NPA in Figure 3. The run corresponding to the sequence of states $(q_0, q_1, q_2, q_3, q_4, q_5, q_7, q_4, q_5, q_6, (q_2, q_3, q_4, q_5, q_6)^*)$ is accepting, since the state q_2 occurs infinitely often. However, the first repetition along this run is q_4. Thus, the pruned run would be the subsequence $(q_0, q_1, q_2, q_3, q_4, q_5, q_7, q_4)$, and so it would terminate in q_4, which has odd priority.

To prove that whenever there is an accepting run, there is one without "useless" repetitions, we formalise the corresponding notion:

Definition 5.11 (Noisy Repetition). Let $\mathcal{E}$ be a PES, let φ be a closed formula of $\mathcal{L}_{hp}$, and let $\mathcal{A}_{\mathcal{E},\varphi}$ be the corresponding NPA. Let r be an accepting run on a k-tree $\mathcal{T}$. A repetition u in r is called *noisy* if it has odd or no priority and no ancestor u' of u is a repetition of even priority.

We show that noisy repetitions can be removed still getting a valid run. We first observe that in an accepting run, each infinite path p includes a repetition over a state whose priority is $\mathcal{F}(p)$ (hence, even; see Definition 5.2).

Lemma 5.12 (Maximal Priority Repetitions). *Let $\mathcal{E}$ be a PES, let φ be a closed formula of $\mathcal{L}_{hp}$, and let r be an accepting run of the NPA $\mathcal{A}_{\mathcal{E},\varphi}$ on a k-tree $\mathcal{T}$. For each infinite accepting path $p = (u_0, u_1, \ldots)$ in r there exists a repetition u_i of priority $\mathcal{F}(p)$.*

The above lemma implies that an accepting run r over a k-tree $\mathcal{T}$ has a finite number of noisy repetitions. In fact, it is immediate to see that each path in the run has a finite number of noisy repetition, since they must precede the first repetition of priority $\mathcal{F}(p)$. We conclude by the fact that $\mathcal{T}$ has branching bounded by k.

We can now show that if an automaton has an accepting run, then it has an accepting run without noisy repetitions.

Lemma 5.13 (Avoiding Noisy Repetitions). *Let $\mathcal{E}$ be a PES and let φ be a closed formula of $\mathcal{L}_{hp}$. If $L(\mathcal{A}_{\mathcal{E},\varphi}) \neq \emptyset$, then $\mathcal{A}_{\mathcal{E},\varphi}$ has an accepting run r without noisy repetitions.*

We thus reach the desired conclusion, i.e., when an automaton has a non-empty language, it has an accepting run that, once pruned, has all leaves with even priority.

Lemma 5.14 (Pruned Run with Even Leaves). *Let $\mathcal{E}$ be a PES and let φ be a closed formula of $\mathcal{L}_{hp}$. If $L(\mathcal{A}_{\mathcal{E},\varphi}) \neq \emptyset$, then $\mathcal{A}_{\mathcal{E},\varphi}$ has an accepting run r on a k-tree $\mathcal{T}$ such that in $\mathcal{T}^{(r)}$ all leaves have even priority.*

Using the above lemma, it is easy to prove that if $\mathcal{A}_{\mathcal{E},\varphi}$ has a non-empty language, then $\mathcal{E}$ satisfies φ. The proof relies on the fact that an accepting run for $\mathcal{A}_{\mathcal{E},\varphi}$ whose pruning has all leaves with even priority, can be easily transformed into a successful tableau for φ, so Lemma 4.3 allows us to conclude.

Lemma 5.15 (Non-Emptiness Implies Satisfaction). *Let $\mathcal{E}$ be a strongly regular PES and let $\breve{\varphi}$ be a closed formula. If $L(\mathcal{A}_{\mathcal{E},\breve{\varphi}}) \neq \emptyset$, then $\mathcal{E} \models \breve{\varphi}$.*

Conversely, given a run r whose pruning has all leaves with even priority, it could still be the case that r is not accepting, because it does not take advantage of the possibility of cycling over the leaves. For instance, consider again the NPA in Figure 3. The run corresponding to the sequence of states $(q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_2, q_3, (q_4, q_5, q_7)^*)$ is not accepting, since the only state with a priority that occurs infinitely often is q_4 and, as already observed, q_4 has odd priority. However, the first repetition along this run is q_2 and thus the pruned run would be the subsequence $(q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_2)$, terminating in the state q_2, which has even priority. Indeed, recognising the presence of the loop $q_2, q_3, q_4, q_5, q_6, q_2$, one can construct the run $(q_0, q_1, (q_2, q_3, q_4, q_5, q_6)^*)$, which is accepting.

In general, we can prove that if there exists a run whose pruning has all leaves with even priority, then there exists an accepting run.

Lemma 5.16 (Accepting Run for Pruned Runs). *Let $\mathcal{E}$ be a PES and let φ be a closed formula of $\mathcal{L}_{hp}$. If there exists a run r of the NPA $\mathcal{A}_{\mathcal{E},\varphi}$ on a k-tree $\mathcal{T}$ such that in $\mathcal{T}^{(r)}$ all leaves have even priority, then there exists also an accepting run of $\mathcal{A}_{\mathcal{E},\varphi}$.*

The above result allows us to conclude that if $\mathcal{E}$ satisfies φ, then $\mathcal{A}_{\mathcal{E},\varphi}$ has a non-empty language. Again, we rely on the results proven for tableaux. By Lemma 4.16, whenever $\mathcal{E}$ satisfies φ there is a successful tableau for φ. The proof then shows that a successful tableau for φ can be viewed as the pruning of a run where all leaves have even priority. By Lemma 5.16 this can be transformed into an accepting run for $\mathcal{A}_{\mathcal{E},\varphi}$.

Lemma 5.17 (Satisfaction Implies Non-Emptiness). *Let $\mathcal{E}$ be a strongly regular PES and let $\breve{\varphi}$ be a closed formula. If $\mathcal{E} \models \breve{\varphi}$, then $L(\mathcal{A}_{\mathcal{E},\breve{\varphi}}) \neq \emptyset$.*

Joining Lemmata 5.15 and 5.17, we have that the model-checking problem of a formula φ over a strongly regular PES $\mathcal{E}$ reduces to non-emptiness of the language of the automaton $\mathcal{A}_{\mathcal{E},\varphi}$.

Theorem 5.18 (Model Checking Via Non-emptiness). *Let $\mathcal{E}$ be a strongly regular PES and let φ be a closed formula of $\mathcal{L}_{hp}$. Then $L(\mathcal{A}_{\mathcal{E},\varphi}) \neq \emptyset$ iff $\mathcal{E}$ satisfies φ.*

The above result, combined with a suitable bisimulation equivalence $\equiv$ of finite index, can be exploited to obtain an effective procedure for checking the satisfaction of a formula. In fact, given a bisimulation $\equiv$ over $\mathcal{A}_{\mathcal{E},\varphi}$ of finite index, the quotient automaton $\mathcal{A}_{\mathcal{E},\varphi/\equiv}$ is finite and, exploiting Theorems 5.18 and 5.5, we can verify whether $\mathcal{E} \models \varphi$ by checking the emptiness of the language accepted by $\mathcal{A}_{\mathcal{E},\varphi/\equiv}$. Clearly a concrete algorithm will not first generate the infinite state NPA and then take the quotient, but it rather performs the quotient on the fly: Whenever a new state would be equivalent to one already generated, the transition loops back to the already existing state.

When $\mathcal{E}$ is strongly regular, a reference bisimulation equivalence of finite index on $\mathcal{A}_{\mathcal{E},\varphi}$ is future equivalence. An obstacle towards the use of the quotiented NPA for model checking purposes is the fact that the future equivalence could be hard to compute (or even undecidable). To make the construction effective, we need a decidable bisimulation equivalence on the NPA and the effectiveness of the set of successors of a state. This is further discussed in the next section.

6 MODEL CHECKING PETRI NETS

We show how the abstract automata-based model-checking approach outlined in the previous section can be instantiated on finite safe Petri nets, a classical model of concurrency and distribution [38], by identifying a suitable effective bisimulation equivalence on the NPA.

6.1 Petri Nets and Their Event Structure Semantics

A *Petri net* is a tuple $\mathcal{N} = (P, T, F, M_0)$ where P, T are disjoint sets of *places* and *transitions*, respectively, $F : (P \times T) \cup (T \times P) \to \{0, 1\}$ is the *flow function*, and $M_0 : P \to \mathbb{N}$ is the initial marking,

i.e., the initial state of the net. We assume that the set of transitions is a subset of a fixed set $\mathbb{T}$ with a labelling $\lambda_N : \mathbb{T} \to \Lambda$.

Places can be seen as abstract resources. The state of a net $\mathcal{N}$ is thus a so-called *marking*, i.e., a function $M : P \to \mathbb{N}$, indicating for each place the number of instances, called *tokens*, available in the place. Transitions are actions that, to be executed, require some instances of resources (one for each place $p \in P$ such that $F(p,t) = 1$). The execution consumes such resources and produces some new instances (one for each $p \in P$ such that $F(t,p) = 1$). An example of Petri net can be found in Figure 1(b). Graphically, places and transitions are drawn as circles and rectangles, respectively, while the flow function is rendered by means of directed arcs connecting places and transitions. Markings are represented by inserting tokens (black dots) in the corresponding places.

Formally, a transition $t \in T$ is *enabled* at a marking M if $M(p) \geq F(p,t)$ for all $p \in P$. In this case it can be *fired* leading to a new marking M' defined by $M'(p) = M(p) + F(t,p) - F(p,t)$ for all places $p \in P$. This is written $M[t\rangle M'$. We denote by $\mathcal{R}(\mathcal{N})$ the set of markings reachable in $\mathcal{N}$ via a sequence of firings starting from the initial marking. We say that a marking M is *coverable* if there exists $M' \in \mathcal{R}(\mathcal{N})$ such that $M \leq M'$, pointwise. A net $\mathcal{N}$ is *safe* if for every reachable marking $M \in \mathcal{R}(\mathcal{N})$ and all $p \in P$, we have $M(p) \leq 1$, i.e., each place contains at most one token. Hereafter, we will consider only safe nets. Hence, a marking M will be often confused with the corresponding subset of places $\{p \mid M(p) = 1\} \subseteq P$. For $x \in P \cup T$ the *pre-set* and *post-set* are defined ${}^\bullet x = \{y \in P \cup T \mid F(y,x) = 1\}$ and $x^\bullet = \{y \in P \cup T \mid F(x,y) = 1\}$, respectively.

The concurrent behaviour of a Petri net can be represented by its unfolding $\mathcal{U}(N)$, defined below as an acyclic net constructed inductively starting from the initial marking of N and then adding, at each step, an occurrence of each enabled transition of N. In what follows, we indicate by π_1 the projection over the first component of a pair, i.e., $\pi_1(a,b) = a$.

Definition 6.1 (Net Unfolding). Let $\mathcal{N} = (P, T, F, M_0)$ be a safe net. The unfolding is the least net $\mathcal{U}(\mathcal{N}) = (P^U, T^U, F^U, M_0^U)$ such that

- $M_0^U = \{(p, \bot) \mid p \in M_0\} \subseteq P^U$, where $\bot$ is a new element, not in P, T or F;
- if $t \in T$ and $X \subseteq P^U$ is coverable with $\pi_1(X) = {}^\bullet t$, then $(t, X) \in T^U$;
- for any $e = (t, X) \in T^U$, the set $Z = \{(p, e) \mid p \in t^\bullet\} \subseteq P^U$; moreover ${}^\bullet e = X$ and $e^\bullet = Z$.

Places and transitions in the unfolding represent instances of tokens and firing of transitions, respectively, of the original net. The projection π_1 over the first component maps places and transitions of the unfolding to the corresponding items of the original net $\mathcal{N}$. The second component records the "causal history," i.e., for places the transition that generated the token (or $\bot$ for tokens in the initial marking that have not been generated by any transition) and for transitions the set of tokens used for the firing. The initial marking M_0^U consists of the set of minimal places. For historical reasons transitions and places in the unfolding are also called *events* and *conditions*, respectively.

One can define *causality* $\leq_N$ over the unfolding as the reflexive and transitive closure of the flow relation. Explicitly, $\leq_N$ is the smallest reflexive and transitive relation such that $x \leq_N y$ whenever $x \in {}^\bullet y$. *Conflict* is the smallest relation such that $e \#_N e'$ if ${}^\bullet e \cap {}^\bullet e' \neq \emptyset$ (i.e., when e and e' compete for a common resource), and inherited along causality, namely, if $x \#_N y$ and $y \leq_N z$, then $x \#_N z$. The events T^U of the unfolding of a finite safe net, endowed with causality and conflict, form a PES.

Definition 6.2 (PES for a net). Let $\mathcal{N} = (P, T, F, M_0)$ be a safe net and let $\mathcal{U}(\mathcal{N}) = (P^U, T^U, F^U, M_0^U)$ be its unfolding. The PES associated with $\mathcal{N}$ is $\mathcal{E}(\mathcal{N}) = \langle T^U, \leq_N, \#_N \rangle$.

As an example, the unfolding $\mathcal{U}(\mathcal{N})$ of the running example net $\mathcal{N}$ and the corresponding PES can be found in Figures 1(c) and 1(a), respectively.

The transitions of a configuration $C \in \mathcal{C}(\mathcal{E}(\mathcal{N}))$ can be fired in any order compatible with causality, producing a marking $C^\circ = (M_0^U \cup \bigcup_{t\in C} t^\bullet) \setminus (\bigcup_{t\in C} {}^\bullet t)$ in $\mathcal{U}(\mathcal{N})$; in turn, this corresponds to a reachable marking of $\mathcal{N}$ given by $\mathsf{M}(C) = \pi_1(C^\circ)$.

6.2 Automata Model Checking for Petri Nets

The PES associated with a finite safe Petri net is known to be regular [46]. We next prove that it is also strongly regular and, thus, we can apply the theory developed so far for model checking $\mathcal{L}_{hp}$ over finite safe Petri nets.

Let $\mathcal{N} = (P, T, F, M_0)$ be a safe Petri net. A basic observation is that the residual of the PES $\mathcal{E}(\mathcal{N})$ with respect to a configuration $C \in \mathcal{C}(\mathcal{E}(\mathcal{N}))$ is uniquely determined by the marking produced by C. This correspondence can be extended to pointed configurations by considering markings that additionally record, for the events of interest in the past, the places in the marking that are caused by such events. This motivates the definition below.

Definition 6.3 (Pointed Marking). Let $\mathcal{N} = (P, T, F, M_0)$ be a safe Petri net. Given a set V, a *V-pointed marking* is a pair $\langle M, r\rangle$ where $M \subseteq P$ is a safe marking and $r : V \to 2^M$.

A V-pointed configuration $\langle C, \zeta\rangle$ of $\mathcal{E}(\mathcal{N})$ naturally induces a V-pointed marking $\mathsf{M}(\langle C, \zeta\rangle) = \langle \mathsf{M}(C), r\rangle$ where $r(x) = \{\pi_1(b) \mid b \in C^\circ \ \wedge\ \zeta(x) < b\}$. We next observe that pointed configurations producing the same pointed marking have isomorphic pointed residuals.

Proposition 6.4 (Pointed Markings vs Residuals). *Let $\mathcal{N} = (P, T, F, M_0)$ be a safe Petri net. Given a set V and two V-pointed configurations $\langle C_1, \zeta_1\rangle$, $\langle C_2, \zeta_2\rangle$ in $\mathcal{E}(\mathcal{N})$, if $\mathsf{M}(\langle C_1, \zeta_1\rangle) = \mathsf{M}(\langle C_2, \zeta_2\rangle)$, then $\langle C_1, \zeta_1\rangle \approx_r \langle C_2, \zeta_2\rangle$.*

By the above result the PES associated with a finite safe Petri net is strongly regular. In fact, the number of residuals of V-pointed configurations, up to isomorphism, by Proposition 6.4, is smaller than the number of V-pointed markings, which is clearly finite, since the net is finite and safe. Furthermore, we already know that the PES associated with a finite safe Petri net is regular, thus it is boundedly branching. Hence, one can conclude by using Lemma 3.10.

Corollary 6.5 (Strong Regularity). *Let $\mathcal{N}$ be a finite safe Petri net. Then the corresponding PES $\mathcal{E}(\mathcal{N})$ is strongly regular.*

To instantiate the model-checking framework to finite safe Petri nets, the idea is to take an equivalence over the infinite NPA that equates states whose (pointed) configurations induce the same pointed marking.

Definition 6.6 (Pointed-Marking Equivalence on NPA). Let $\mathcal{N}$ be a finite safe Petri net and let φ be a closed formula of $\mathcal{L}_{hp}$. Two states q_1, q_2 in the NPA $\mathcal{A}_{\mathcal{E}(\mathcal{N}),\varphi}$ are *pointed-marking equivalent*, written $q_1 \approx_m q_2$, if $q_i = (C_i, \eta_i, \psi)$, $i \in \{1, 2\}$, for some $\psi \in \mathsf{sf}(\varphi)$ and $\mathsf{M}(\langle C_1, \eta_{1\,|fv(\psi)}\rangle) = \mathsf{M}(\langle C_2, \eta_{2\,|fv(\psi)}\rangle)$.

Using Proposition 6.4, we can immediately prove that $\approx_m$ refines $\approx_f$. Moreover, we can show that $\approx_m$ is a bisimulation in the sense of Definition 5.4.

Proposition 6.7 (Pointed Marking Equivalence is a Bisimulation). *Let $\mathcal{N}$ be a finite safe Petri net and let φ be a closed formula of $\mathcal{L}_{hp}$. The equivalence $\approx_m$ on the automaton $\mathcal{A}_{\mathcal{E}(\mathcal{N}),\varphi}$ is a bisimulation and it is of finite index.*

Relying on Propositions 6.4 and 6.7, we can provide an explicit construction of the quotient automaton $\mathcal{A}_{\mathcal{E}(\mathcal{N}),\varphi\,/\approx_m}$. We introduce a convenient notation for transitions between pointed markings. Given the tuples of variables $\mathbf{x}$, $\mathbf{y}$, a set V such that $\mathbf{x} \cup \mathbf{y} \subseteq V$, and a V-pointed marking

$\langle M, r\rangle$, we write $\langle M, r\rangle \xrightarrow{\mathbf{x},\overline{\mathbf{y}} < t}_{\mathrm{a}, z} \langle M', r'\rangle$ if $M[t\rangle M'$, $\lambda_N(t) = \mathrm{a}$, for all $x \in \mathbf{x}$, we have $r(x) \cap {}^\bullet t \neq \emptyset$, for all $y \in \mathbf{y}$ it holds $r(y) \cap {}^\bullet t = \emptyset$, and r' is defined by $r'(z) = t^\bullet$ and $r'(w) = (r(w) \cap M') \cup \{s \mid r(w) \cap {}^\bullet t \neq \emptyset \ \wedge \ s \in t^\bullet\}$, for $w \neq z$. In words, from the pointed marking $\langle M, r\rangle$, the transition t is fired and "pointed" by variable z. Transition t is required to consume tokens caused by $\mathbf{x}$ and not to consume tokens caused by $\mathbf{y}$ to be itself caused by $\mathbf{x}$ and independent from $\mathbf{y}$. After the firing, clearly, z causes $t^\bullet$ and variables that were causes of some $p \in {}^\bullet t$ become causes of the places in $t^\bullet$.

Construction 1 (Quotient NPA). *Let $\mathcal{N}$ be a finite safe Petri net and let φ be a closed formula of $\mathcal{L}_{hp}$. The quotient NPA $\mathcal{A}_{\mathcal{E}(\mathcal{N}),\varphi\,/_{\approx m}}$ is defined as follows: The set of states $Q = \{(M, r, \psi) \mid M \in \mathcal{R}(\mathcal{N}) \ \wedge \ r : fv(\psi) \to 2^M \ \wedge \ \psi \in \mathrm{sf}(\varphi)\}$. The initial state $q_0 = (M_0, \emptyset, \varphi)$. The transition relation is defined, for any state $q = (M, r, \psi) \in Q$, by:*

- *if $\psi = \mathsf{T}$ or $\psi = \mathsf{F}$, then $q \to (q)$*
- *if $\psi = \psi_1 \wedge \psi_2$, then $q \to (q_1, q_2)$ where $q_i = (M, r, \psi_i)$, $i \in \{1, 2\}$*
- *if $\psi = \psi_1 \vee \psi_2$, then $q \to (q_1)$ and $q \to (q_2)$ where $q_i = (M, r, \psi_i)$, $i \in \{1, 2\}$*
- *if $\psi = [\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\psi'$, let $S = \{(M', r'_{\lfloor fv(\psi')}) \mid \langle M, r\rangle \xrightarrow{\mathbf{x},\overline{\mathbf{y}} < t}_{a, z} \langle M', r'\rangle\}$;*
  — if $S = \{(M_1, r_1), \ldots, (M_n, r_n)\} \neq \emptyset$, then $q \to (q_1, \ldots, q_n)$ where $q_i = (M_i, r_i, \psi')$ for $i \in [1, n]$,
  — otherwise, $q \to (q)$
- *if $\psi = \langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle\psi'$, let $S = \{(M', r'_{\lfloor fv(\psi')}) \mid \langle M, r\rangle \xrightarrow{\mathbf{x},\overline{\mathbf{y}} < t}_{a, z} \langle M', r'\rangle\}$;*
  — if $S = \{(M_1, r_1), \ldots, (M_n, r_n)\} \neq \emptyset$, then $q \to (q_i)$ where $q_i = (M_i, r_i, \psi')$ for $i \in [1, n]$,
  — otherwise, $q \to (q)$
- *if $\psi = (\alpha X(\mathbf{x}).\psi')(\mathbf{y})$, then $q \to (q')$ where $q' = (M, r, X(\mathbf{y}))$*
- *if $\psi = X(\mathbf{z})$ and $\psi' \in \mathrm{sf}(\varphi)$ is the unique subformula such that $\psi' = (\alpha X(\mathbf{x}).\psi'')(\mathbf{y})$, then $q \to (q')$ where $q' = (M, r[\mathbf{x} \mapsto r(\mathbf{z})], \psi'')$.*

The acceptance condition is analogous to that in Definition 5.6.

The automaton $\mathcal{A}_{\mathcal{E}(\mathcal{N}),\varphi\,/_{\approx m}}$ is finite for all finite safe Petri nets $\mathcal{N}$. Thus, it can be used for model-checking a formula φ of $\mathcal{L}_{hp}$ over a finite safe Petri net by means of a language emptiness check. This is detailed in the next section.

6.3 A Prototype Tool

The algorithm for model-checking Petri nets outlined before is implemented in a prototype tool called TCWB (*True Concurrency Workbench*) [34], written in Haskell. The tool inputs a safe Petri net $\mathcal{N}$ and a closed formula φ of $\mathcal{L}_{hp}$ and outputs the truth value of the formula on the initial marking of $\mathcal{N}$. The truth of the formula is reduced to the non-emptiness of the language of the automaton $\mathcal{A}_{\mathcal{E}(\mathcal{N}),\varphi\,/_{\approx m}}$ (Theorem 5.18). The algorithm builds the quotient NPA $\mathcal{A}_{\mathcal{E}(\mathcal{N}),\varphi\,/_{\approx m}}$ "on demand," i.e., the states of the automaton are generated when they are explored in the search for an accepting run. A path is recognised as accepting when it includes a loop where a $\sqsubseteq_d^*$-maximal subformula (see Definition 3.2) is T, a $[\![\ ]\!]$-subformula or a proposition quantified in a ν-subformula. In this way only a fragment of $\mathcal{A}_{\mathcal{E}(\mathcal{N}),\varphi\,/_{\approx m}}$ relevant to decide the satisfaction of φ is actually built.

A loop is identified when a state is encountered that is pointed-marking equivalent to an ancestor state. According to Definition 6.3, to be pointed-marking equivalent two states must contain the very same subformula. It is easy to see that this requirement could be relaxed, e.g., by allowing different names for the free event variables (as it happens in Definition 5.7). In principle, a coarser pointed-marking equivalence could reduce the size of the quotiented automaton and thus increase

Table 2. Results of Tests Performed on TCWB, Compared with Those of CWB

Process name	Process states	Formula	Subformulae	TCWB (CPU s)	CWB (CPU s)
5 cyclers	1,024	*Live*	6	<1	<1
6 cyclers	4,096	*Live*	6	<1	1
7 cyclers	16,384	*Live*	6	8	5
8 cyclers	65,536	*Live*	6	247	28
9 cyclers	262,144	*Live*	6	-	570

efficiency. However, even with the advantages deriving from the reduction of size, in practice, the cost of checking whether formulae are the same up-to variable renaming would lead to an overall less efficient procedure. The implementation adopts an intermediate approach, which tries to rename event variables only in the case of propositions. Moreover, states are kept in an ordered structure that allows for a binary search. Again, maintaining such structure has some cost, but since the queries vastly outnumber the updates, this turns out to be convenient.

As a side remark, we tried a direct implementation of the tableau-based procedure. As anticipated, it resulted to have very poor performances mainly because of the repeated exploration of "equivalent" paths during the construction of a proof tree, which is avoided in the automata-theoretic procedure.

Given a net $\mathcal{N} = (P, T, F, M_0)$ and a formula φ, the number of states in the quotient automaton $\mathcal{A}_{\mathcal{E}(\mathcal{N}),\varphi_{/\approx_m}}$ can be bounded as follows: Recall that a state consists of a triple (M, r, ψ) where $\psi \in \mathsf{sf}(\varphi)$, M is a reachable marking and $r : fv(\psi) \to 2^M$ is a function. This leads to an upper bound $O(|\mathsf{sf}(\varphi)| \cdot |\mathcal{R}(\mathcal{N})| \cdot 2^{|P|\cdot v})$, where $v = max\{|fv(\psi)| : \psi \in \mathsf{sf}(\varphi)\}$ is the largest number of event variables appearing free in a subformula of φ. In turn, since $|\mathcal{R}(\mathcal{N})| \leq 2^{|P|}$, this is bounded by $O(|\mathsf{sf}(\varphi)| \cdot 2^{|P|\cdot(v+1)})$. The size of the automaton is thus exponential in the size of the net and linear in the size of the formula. Moving from the interleaving fragment of the logic (where $v = 0$) to formulae capable of expressing true concurrent properties thus causes an exponential blow-up. However, note that the worst-case scenario requires all transitions to be related by causality and concurrency to all places in all possible ways, something that should be quite unlikely in practice. Indeed, despite the fact that the tool is very preliminary and more tweaks and optimisations could improve its efficiency, for the practical tests we performed, the execution time seems to be typically well below than the theoretical worst-case upper bound.

We performed some simple tests. Despite the absence of another tool with the same purpose to compare with, they were useful to grasp some information. All tests were performed using Petri nets representing processes made of a number, indicated in the process name, of parallel copies of a cycler, which is a sequential loop always repeating the same four states.

- *Absence of deadlock.* This property can be expressed in the interleaving fragment of $\mathcal{L}_{hp}$ via the formula $Live = \nu X.(\langle\!\_\, x\rangle\!\rangle \mathsf{T} \wedge [\![\_\, y]\!]X)$. Hence, we can compare with a classical tool for model-checking formulae of the mu-calculus, i.e., Edinburgh Concurrency Workbench (CWB). The CWB exploits a game-theoretical formalisation of the model-checking problem and search for a winning strategy for one of two players. The results are reported in Table 2. The symbol "-" in the CPU time column indicates an execution time exceeding 1,800 seconds. The CWB is faster, as expected, since it is a well-optimised tool and the greater expressiveness of $\mathcal{L}_{hp}$ requires the maintenance of more complex data structures. Still, the fact that the efficiency is comparable suggests that the overhead deriving by the need of setting up the data structures needed to deal with pointed markings is acceptably small.

Table 3. Results of Tests Performed on TCWB Using True Concurrent Properties

Process name	Process states	Formula	Subformulae	TCWB (CPU s)
5 cyclers	1,024	*Atom*	12	<1
6 cyclers	4,096	*Atom*	12	5
7 cyclers	16,384	*Atom*	12	97
8 cyclers	65,536	*Atom*	12	-

- *Causal atomicity* [18] for a block labelled by a. We assume that every action of a process is either labelled by a or b. Hence, the process has only one block, required to be atomic, labelled by a; everything else is outside the block and is labelled by b. Then, as mentioned in Section 3.2, causal atomicity can be expressed by the formula $Atom = \nu X.([\![\_\, w]\!]X \wedge [\![\mathrm{a}\, x]\!]\nu Y(x).([\![x < \mathrm{b}\, y]\!][\![y < \mathrm{a}\, z]\!]\mathsf{F} \wedge [\![\_\, w]\!]Y(x)))$. Note that all transitions in the cyclers are labelled by a. While this might seem strange for the task of interest, it has been chosen since it produces the worst-case scenario. In fact, every pair of transitions must be checked for the absence of an atomicity violation, since all transitions are in the same atomic block. In this case the "true concurrent" operators of $\mathcal{L}_{hp}$ are needed to express the property. Thus, it cannot be tested on CWB. The results of the tests on TCWB are displayed in Table 3.

Interestingly, in the tests above, moving from the interleaving fragment of the logic to true concurrent properties, the size of the automata does not grow exponentially, as the theoretical bound would suggest. Indeed, as mentioned before, the worst case would require all transitions to be related to all places in all possible ways, which is very unlikely in practice and surely not happening in the processes used for the tests. In this particular case, the size of the automaton grows by a factor n equal to the number of cyclers in the system. This is because every reachable marking consists of n places and every transition causes exactly one of such places. Then, since the largest number of event variables occurring free in a subformula of *Atom* is 1, the number of states of the automaton up to pointed-marking equivalence grows by the factor n, with respect to the interleaving case. So the upper bound to the size of the automaton for the true concurrent property is just $O(|\mathrm{sf}(Atom)| \cdot |\mathcal{R}(\mathcal{N})| \cdot n)$, where $\mathcal{N}$ is the Petri net representing the n cyclers process, while in the interleaving case, the size coincides with the theoretical bound, i.e., $O(|\mathrm{sf}(Live)| \cdot |\mathcal{R}(\mathcal{N})|)$.

7 CONCLUSIONS

We studied the model-checking problem for the logic for true concurrency $\mathcal{L}_{hp}$, representing the logical counterpart of a classical true concurrent behavioural equivalence, i.e., history-preserving bisimilarity. Resorting to a tableau-based technique, we showed that the problem is decidable for the class of strongly regular PESs, which include regular trace PESs. We then devised an automata-theoretic model-checking approach relying on parity tree automata, amenable to a more efficient implementation. As an example of instantiation on a concrete formalism, we showed how the technique can be implemented on finite safe Petri nets, also producing a proof-of-concept tool.

We proved that the class of regular trace PESs is included in that of strongly regular PESs, which in turn is included in the class of regular PESs. The precise relation of strongly regular PESs with the other two classes is still unclear and interesting in view of Reference [10], which recently showed that regular trace PESs are strictly included in regular PESs, disproving Thiagarajan's conjecture.

Several other papers deal with model-checking for logics on event structures. In Reference [37] a technique is proposed for model-checking a CTL-style logic with modalities for immediate causality and conflict on a subclass of PESs. The logic is quite different from ours, as formulae are satisfied

by single events; the idea being that an event, with its causes, represents the local state of a component. The procedure involves the construction of a finite representation of the PES associated with a program that has some conceptual relation with our quotienting phase. In Reference [28] the author shows that first order logic and Monadic Trace Logic (MTL), a restricted form of Monadic Second Order (MSO) logic are decidable on regular trace event structures. The possibility of directly observing conflicts in MTL and thus of distinguishing behaviourally equivalent PESs (e.g., the PESs consisting of a single or two conflicting copies of an event), and the presence in $\mathcal{L}_{hp}$ of propositions that are non-monadic with respect to event variables, make these logics not immediate to compare. Still, a deeper investigation is worth to pursue, especially in view of the fact that, in the propositional case, the mu-calculus corresponds to the bisimulation invariant fragment of MSO logic [22]. Understanding which are the bisimulation invariant fragments of MSO over event structures, with respect to the various concurrent bisimulations, is an interesting program in itself.

The work summarised in Reference [20] develops a game-theoretic approach for model-checking a concurrent logic over partial order models. It has been observed in Reference [4] that such logic is incomparable to $\mathcal{L}_{hp}$. Preliminary investigations show that our model-checking framework could be adapted to such a logic and, more generally, to a logic joining the expressive power of the two. Moreover, further exploring the potentialities of a game-theoretic approach in our setting represents an interesting venue of further research.

Another open issue concerns the possibility of generalising the results in this article to the full logic $\mathcal{L}$ in Reference [4]. This is quite challenging: The full logic $\mathcal{L}$ induces a behavioural equivalence—hhp-bisimilarity—which is undecidable already for finite state Petri nets [25]. Note that this does not imply undecidability of the corresponding model-checking problem. On the semantic side, relaxing the restriction to strongly regular PESs appears to be quite problematic unless one is willing to deal with transfinite runs that, however, would be of very limited practical interest.

The tool is still preliminary. As suggested by its name (inspired to the Edinburgh Concurrency Workbench [44]), we would like to bring the TCWB to a more mature stage, working on optimisations and adding an interface that gives access to a richer set of commands. In particular, the tool is missing two main optimisations: a strategy for an efficient exploration of the automaton, and efficient data structures and ways to maintain and update them. Both have been carefully considered in Reference [44] for CWB and shown to bring to great improvements in performances. For instance, CWB uses so-called *assumptions* and *decisions* for choice-point during the exploration to efficiently build the winning strategy for a player. Despite the fact that our procedure is based on automata instead of games, we believe that similar techniques could be exploited also in the TCWB.

APPENDICES

A TECHNICALITIES AND PROOFS FOR SECTION 3

A.1 Free Variables and Substitutions

This section presents in detail some routine definitions and results about the logic, including the notions of free event and propositional variable, the notions of substitutions and some technical lemmata showing the independence of the semantics from the naming of the variables.

Definition A.1 (Free Variables). The *free variables* of a formula φ in $\mathcal{L}_{hp}$ are defined as:

$$
\begin{aligned}
fv(\mathsf{T}) &= \emptyset, & fv(\langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle\, \varphi) &= (fv(\varphi) \setminus \{z\}) \cup \mathbf{x} \cup \mathbf{y},\\
fv(\mathsf{F}) &= \emptyset, & fv([\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\, \varphi) &= (fv(\varphi) \setminus \{z\}) \cup \mathbf{x} \cup \mathbf{y},\\
fv(\varphi \wedge \psi) &= fv(\varphi) \cup fv(\psi), & fv(X(\mathbf{x})) &= \mathbf{x},\\
fv(\varphi \vee \psi) &= fv(\varphi) \cup fv(\psi), & fv((\alpha X(\mathbf{x}).\varphi)(\mathbf{y})) &= \mathbf{y} \qquad \alpha \in \{\mu, \nu\}.
\end{aligned}
$$

The fact that variables $\mathbf{x}$ are free in $X(\mathbf{x})$, $\nu X(\mathbf{x}).\varphi$ and $\mu X(\mathbf{x}).\varphi$ is reflected in the definition of free variable substitution. For instance, $X(\mathbf{x})[\mathbf{y}/\mathbf{x}] = X(\mathbf{y})$ and $(\nu X(x).\varphi)[y/x] = \nu X(y).(\varphi[y/x])$. Formulae are considered up to α-conversion of bound variables and substitution is assumed to be capture-free. We next define the free abstract propositions of a formula.

Definition A.2 (Free Propositions). The *free abstract propositions* in a formula φ, denoted by $fp(\varphi)$, are defined as:

$$\begin{array}{lll} fp(\mathsf{T}) = \emptyset, & & fp(\langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z \rangle\!\rangle\, \varphi) = fp(\varphi), \\ fp(\mathsf{F}) = \emptyset, & & fp([\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\, \varphi) = fp(\varphi), \\ fp(\varphi_1 \wedge \varphi_2) = fp(\varphi_1) \cup fp(\varphi_2), & & fp((\nu Z(\mathbf{x}).\varphi)(\mathbf{y})) = fp(\varphi) \setminus \{Z\}, \\ fp(\varphi_1 \vee \varphi_2) = fp(\varphi_1) \cup fp(\varphi_2), & & fp((\alpha Z(\mathbf{x}).\varphi)(\mathbf{y})) = fp(\varphi) \setminus \{Z\} \qquad \alpha \in \{\mu, \nu\}. \end{array}$$

Given an environment $\eta : Var \to E$, a variable $x \in Var$, and an event $e \in E$, we denote by $\eta[x \mapsto e]$ the environment defined, for $y \in Var$, by

$$\eta[x \mapsto e](y) = \begin{cases} e & \text{if } y = x \\ \eta(y) & \text{otherwise.} \end{cases}$$

Updates of a proposition environment must be properly defined to ensure that the semantics of a formula does not depend on the naming of its free variables. For $\pi \in PEnv_{\mathcal{E}}$ and $S \subseteq \mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}}$, we write $\pi[X(\mathbf{x}) \mapsto S]$ for the proposition environment defined by

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

Lemma A.3 (Renaming Variables and Propositions). *Let $\mathcal{E}$ be a PES, let π a proposition environments, and let φ be a formula of $\mathcal{L}_{hp}$.*

1. *Given two tuples of variables $\mathbf{x}, \mathbf{y}$ with $|\mathbf{x}| = |\mathbf{y}|$ and $fv(\phi) \subseteq \mathbf{x}$, then $\{\!|\varphi[\mathbf{y}/\mathbf{x}]|\!\}^{\mathcal{E}}_{\pi} = \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}[\mathbf{y}/\mathbf{x}]$, where $\_[\mathbf{y}/\mathbf{x}]$ is the operation defined in Section 3.4.*
2. *For all formula ψ and abstract proposition $Z \in \mathcal{X}^a$ such that $fv(\psi) = \mathbf{z}$ and $ar(Z) = |\mathbf{z}|$, it holds $\{\!|\varphi[Z(\mathbf{z}) := \psi]|\!\}^{\mathcal{E}}_{\pi} = \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi[Z(\mathbf{z}) \mapsto \{|\psi|\}^{\mathcal{E}}_{\pi}]}$.*

Proof. Both items can be proved by routine inductions on φ. □

From (1) above, it follows that the semantics of a formula φ in $\mathcal{L}_{hp}$ only depends on the events that the environment associates with the free variables $\mathbf{x}$ of the formula, i.e., if $C \in \mathcal{C}(\mathcal{E})$ and η, η' are environments such that $\eta(\mathbf{x}) = \eta'(\mathbf{x})$ pointwise, then $(C, \eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}$ iff $(C, \eta') \in \{\!|\varphi|\!\}^{\mathcal{E}}$. Analogously, from (2), we have that the semantics of φ only depends on the value of the proposition environment π on the free propositions of φ, while it is independent from the interpretation of those that do not occur free in it. Point (2) also shows how substitutions of propositions in formulae correspond to updates to proposition environments.

A.2 Proofs for Section 3.5

Lemma 3.9 (Equisatisfaction in Pointed Configurations with Isomorphic Residuals). *Let $\mathcal{E}$ be a PES, let φ be a formula of $\mathcal{L}_{hp}$, let $\pi \in PEnv_{\mathcal{E}}$ be a proposition environment saturated for φ, and let $(C_1, \eta_1), (C_2, \eta_2) \in \mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}}$. If $\langle C_1, \eta_1|_{fv(\varphi)}\rangle \approx_r \langle C_2, \eta_2|_{fv(\varphi)}\rangle$, then $(C_1, \eta_1) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}$ iff $(C_2, \eta_2) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}$.*

Proof. Assume $\langle C_1, \eta_1|_{fv(\varphi)}\rangle \approx_r \langle C_2, \eta_2|_{fv(\varphi)}\rangle$, via an isomorphism $\iota : \mathcal{E}[C_1] \to \mathcal{E}[C_2]$. We prove that if $(C_1, \eta_1) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}$, then $(C_2, \eta_2) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi}$. Since the isomorphism ι is bijective, the other

implication follows by symmetry. The proof proceeds by induction on the formula φ. We discuss only some representative cases:

- $\varphi = X(\mathbf{x})$
  Let $(C_1, \eta_1) \in \{\!|X(\mathbf{x})|\!\}^{\mathcal{E}}_{\pi} = \pi(X(\mathbf{x}))$. Since π is saturated for $X(\mathbf{x})$, by definition, we get that also $(C_2, \eta_2) \in \pi(X(\mathbf{x}))$, as desired.
- $\varphi = \psi_1 \wedge \psi_2$
  If $(C_1, \eta_1) \in \{\!|\psi_1 \wedge \psi_2|\!\}^{\mathcal{E}}_{\pi}$, then by definition of the semantics $(C_1, \eta_1) \in \{\!|\psi_1|\!\}_{\pi}$ and $(C_1, \eta_1) \in \{\!|\psi_2|\!\}_{\pi}$. By inductive hypothesis, we obtain $(C_2, \eta_2) \in \{\!|\psi_1|\!\}_{\pi}$ and $(C_2, \eta_2) \in \{\!|\psi_2|\!\}_{\pi}$ and thus $(C_2, \eta_2) \in \{\!|\psi_1 \wedge \psi_2|\!\}_{\pi}$.
- $\varphi = \langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z \rangle\!\rangle \psi$
  Assume that $(C_1, \eta_1) \in \{\!|\langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z \rangle\!\rangle \psi|\!\}_{\pi}$. By definition of the semantics there exists an event $e \in \mathcal{E}[C_1]$ such that $C_1 \xrightarrow{\eta_1(\mathbf{x}), \overline{\eta_1(\mathbf{y}) < e}}_{\mathrm{a}} C_1'$ and $(C_1', \eta_1') \in \{\!|\psi|\!\}_{\pi}$ with $\eta_1' = \eta_1[z \mapsto e]$. Since ι is an isomorphism of residuals of pointed configurations $\langle C_1, \eta_1|_{fv(\varphi)}\rangle \approx \langle C_2, \eta_2|_{fv(\varphi)}\rangle$, we readily deduce that $\iota(e)$ is enabled at C_2 and
  $$C_2 \xrightarrow{\eta_2(\mathbf{x}), \overline{\eta_2(\mathbf{y}) < \iota(e)}}_{\mathrm{a}} C_2'. \tag{2}$$
  Consider the restriction of ι to $\mathcal{E}[C_1']$ and call it $\iota' : \mathcal{E}[C_1'] \to \mathcal{E}[C_2']$. Clearly ι' is an isomorphism of PESs. Additionally, if we let $\eta_2' = \eta_2[z \mapsto \iota(e)]$, then it is easy to see that for all $x \in fv(\psi)$, $e_1 \in \mathcal{E}[C_1']$, it holds $\eta_1'(x) \leq e_1$ iff $\eta_2'(x) \leq \iota'(e_1)$. Hence, $\langle C_1', \eta_1'|_{fv(\psi)}\rangle \approx_r \langle C_2', \eta_2'|_{fv(\psi)}\rangle$. In fact, for all $x \in fv(\psi) \subseteq fv(\varphi) \cup \{z\}$:
  (1) if $x \in fv(\varphi)$, then we can observe that $e_1 \in \mathcal{E}[C_1'] \subseteq \mathcal{E}[C_1]$ and $\eta_i(x) = \eta_i'(x)$ for $i \in \{1, 2\}$. Then the desired property follows from the fact that ι is an isomorphism of residuals of the pointed configurations $\langle C_1, \eta_1|_{fv(\varphi)}\rangle$ and $\langle C_2, \eta_2|_{fv(\varphi)}\rangle$.
  (2) if $x = z$, then $\eta_1'(z) = e \in \mathcal{E}[C_1]$, $\eta_2'(z) = \iota(e) \in \mathcal{E}[C_2]$. Since $\iota : \mathcal{E}[C_1] \to \mathcal{E}[C_2]$ is an isomorphism of PESs, $\eta_1(x) \leq e_1'$ iff $\eta_2(x) = \iota(e) \leq \iota(e_1')$.
  Since $\langle C_1', \eta_1'|_{fv(\psi)}\rangle \approx_r \langle C_2', \eta_2'|_{fv(\psi)}\rangle$, by inductive hypothesis, $(C_2', \eta_2') \in \{\!|\psi|\!\}_{\pi}$. Recalling (2), we conclude $(C_2, \eta_2) \in \{\!|\langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z \rangle\!\rangle \psi|\!\}_{\pi}$, as desired.
- $\varphi = (\alpha X(\mathbf{x}).\psi)(\mathbf{y})$, for $\alpha \in \{\nu, \mu\}$
  We know that $\{\!|(\alpha X(\mathbf{x}).\psi)(\mathbf{y})|\!\}_{\pi} = \{\!|\alpha X(\mathbf{x}).\psi|\!\}_{\pi}[\mathbf{y}/\mathbf{x}]$. Thus, it suffices to show that if $(C_1, \eta_1) \in \{\!|\alpha X(\mathbf{x}).\psi|\!\}_{\pi}$, then $(C_2, \eta_2) \in \{\!|\alpha X(\mathbf{x}).\psi|\!\}_{\pi}$.
  We focus on the case $\alpha = \mu$ (the case $\alpha = \nu$ is perfectly dual). Observe that, by definition of the semantics, we have
  $$\{\!|\alpha X(\mathbf{x}).\psi|\!\}_{\pi} = \mu(f)$$
  where $f = f_{\psi, X(\mathbf{x}), \pi} : 2^{C(\mathcal{E}) \times Env_{\mathcal{E}}} \to 2^{C(\mathcal{E}) \times Env_{\mathcal{E}}}$ is defined by $f(S) = \{\!|\psi|\!\}^{\mathcal{E}}_{\pi[Z(\mathbf{x}) \mapsto S]}$.
  We know that $\mu(f) = f^{\gamma}(\emptyset)$ for some ordinal γ. The thesis follows by showing that for any ordinal β, if $(C_1, \eta_1) \in f^{\beta}(\emptyset)$, then $(C_2, \eta_2) \in f^{\beta}(\emptyset)$. The proof is by induction on β. The base case $\beta = 0$ is trivial, since $f^0(\emptyset) = \emptyset$. For a successor ordinal, if $(C_1, \eta_1) \in f^{\beta+1}(\emptyset) = f(f^{\beta}(\emptyset)) = \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi[Z(\mathbf{x}) \mapsto f^{\beta}(\emptyset)]}$. Observe that $\pi' = \pi[X(\mathbf{x}) \mapsto f^{\beta}(\emptyset)]$ is saturated for ψ, as it easily follows from the fact that π is saturated for φ and free variables of ψ are $fp(\psi) \subseteq fp(\varphi) \cup \{X\}$ and $fv(\varphi) = fv(\psi) = \mathbf{x}$. Thus, the only doubt could concern the proposition X, but the condition is satisfied also for X, by the inner inductive hypothesis applied to $f^{\beta}(\emptyset)$. Therefore, we can apply the outer inductive hypothesis to ψ to deduce that $(C_2, \eta_2) \in \{\!|\varphi|\!\}^{\mathcal{E}}_{\pi[Z(\mathbf{x}) \mapsto f^{\beta}(\emptyset)]} = f^{\beta+1}(\emptyset)$. The case of a limit ordinal is straightforward. □

Lemma 3.10 (Strong Regularity and Residuals of Pointed Configurations). *A PES $\mathcal{E}$ is strongly regular iff it is boundedly branching and for any fixed finite set V, the equivalence $\approx_r$ is of finite index over V-pointed configurations of $\mathcal{E}$.*

Proof. ($\Rightarrow$) Assume that $\mathcal{E}$ is strongly regular. Then by definition it is boundedly branching. Let V be a finite set and let $\mathcal{C}_V = \{\langle C, \zeta\rangle \mid C \in \mathcal{C}(\mathcal{E}) \ \wedge \ \zeta : V \to C\}$ be the set of V-pointed configurations.

By strong regularity, the set $\{\mathcal{E}[C] \cup \zeta(V) \mid C \in \mathcal{C}(\mathcal{E}) \ \wedge \ \zeta : V \to C\}$ is finite up to isomorphism of PESs. This does not immediately imply that $\mathcal{C}_V$ is finite up to $\approx_r$. In fact, given two V-pointed configurations $\langle C, \zeta\rangle$, $\langle C', \zeta'\rangle$, an isomorphism of PES $\iota : \mathcal{E}[C] \cup \zeta(V) \to \mathcal{E}[C'] \cup \zeta'(V)$ is not necessarily an isomorphism of the corresponding pointed residuals. This is certainly the case if additionally it holds $\iota(\zeta(x)) = \zeta'(x)$ for all $x \in V$.

Assume, by contradiction, that $\mathcal{C}_V$ is not finite up to $\approx_r$. Then, we can find a sequence of V-pointed configurations $\langle C_i, \zeta^i\rangle$, $i \in \mathbb{N}$ such that the PESs $\mathcal{E}[C_i] \cup \zeta^i(V)$ are all isomorphic, while $\langle C_i, \zeta^i\rangle$ are pairwise non-equivalent with respect to $\approx_r$. Let $\iota_i : \mathcal{E}[C_i] \cup \zeta^i(V) \to \mathcal{E}[C_{i+1}] \cup \zeta^{i+1}(V)$ be PES isomorphisms for all $i \in \mathbb{N}$ and denote by $\iota_{i,j} : \mathcal{E}[C_i] \cup \zeta^i(V) \to \mathcal{E}[C_j] \cup \zeta^j(V)$ the isomorphism resulting as the composition $\iota_{j-1} \circ \cdots \circ \iota_{i+1} \circ \iota_i$.

The key observations are the following: Define the causal depth of an event as $depth(e) = \max\{depth(e') \mid e' < e\}$. Then

(1) the events in $\zeta^i(V)$ have causal depth bounded by $|V|$ in the PES $\mathcal{E}[C_i] \cup \zeta^i(V)$;
(2) the PESs $\mathcal{E}[C_i] \cup \zeta^i(V)$ have a finite number of events of causal depth bounded by $|V|$, since the PES $\mathcal{E}$ is strongly regular (hence, b-bounded for some $b \in \mathbb{N}$);
(3) each isomorphism of PESs ι_i preserves the causal depth of events (i.e., $depth(\iota_i(e)) = depth(e)$).

By (1)–(3) above, we deduce that there are $i, j \in \mathbb{N}$, $i \leq j$ such that for all $x \in V$, we have $\iota_{1,i}(\zeta^1(x)) = \zeta^i(x)$ and $\iota_{1,j}(\zeta^1(x)) = \zeta^j(x)$. This implies that $\iota_{i,j}(\zeta^i(x)) = \zeta^j(x)$ for all $x \in V$.

Therefore, it is immediate to see that $\iota_{i,j}|_{\mathcal{E}[C_i]} : \mathcal{E}[C_i] \to \mathcal{E}[C_j]$ is an isomorphism of the residuals of the pointed configurations $\langle C_i, \zeta^i\rangle$ and $\langle C_j, \zeta^j\rangle$, contradicting the fact that the pointed configurations in the sequence had all non-isomorphic residuals.

($\Leftarrow$) Assume that $\mathcal{E}$ is boundedly branching and for all fixed finite set V, the equivalence $\approx_r$ is of finite index over V-pointed configurations of $\mathcal{E}$. We have to prove that $\mathcal{E}$ is strongly regular. Let k be a fixed integer and let $V = \{v_1, \ldots, v_k\}$ be a set of cardinality k. Each "extended" residual $\mathcal{E}[C] \cup \{e_1, \ldots, e_k\}$, with $e_1, \ldots, e_k \in C$ can be trivially seen as the residual of a V-pointed configuration $\langle C, \zeta\rangle$ with $\zeta(v_i) = e_i$ for $i \in \{1, \ldots, k\}$.

Again, from the fact that $\approx_r$ is of finite index over V-pointed configurations, we cannot immediately conclude. In fact, consider two extended residuals $\mathcal{E}[C] \cup \{e_1, \ldots, e_k\}$ and $\mathcal{E}[C'] \cup \{e'_1, \ldots, e'_k\}$ such that the corresponding V-pointed configurations $\langle C, \zeta\rangle$ and $\langle C', \zeta'\rangle$ are in the same class, i.e., $\langle C, \zeta\rangle \approx_r \langle C', \zeta'\rangle$. Let $\iota : \mathcal{E}[C] \to \mathcal{E}[C']$ be the corresponding isomorphism such that for all $x \in V$, $e \in \mathcal{E}[C]$, we have $\zeta(x) \leq e$ iff $\zeta'(x) \leq \iota(e)$. Observe that it is not necessarily the case that

$$\iota[\zeta(v_1) \mapsto \zeta'(v_1), \ldots, \zeta(v_k) \mapsto \zeta'(v_k)] : \mathcal{E}[C] \cup \{e_1, \ldots, e_k\} \to \mathcal{E}[C'] \cup \{e'_1, \ldots, e'_k\}$$

is an isomorphism of PESs, since the mapping $\zeta(v_1) \mapsto \zeta'(v_1), \ldots, \zeta(v_k) \mapsto \zeta'(v_k)$ does not necessarily respect causal dependencies. However, since, up to isomorphism, there are only finitely many partial orders with k elements, and similarly only finitely many possible labelling, we conclude that each $\approx_r$-class of V-pointed configurations splits in a finite number of classes of isomorphic extended residuals and thus we conclude. □

B TECHNICALITIES AND PROOFS FOR SECTION 4

B.1 Proofs for Section 4.1

Lemma B.1 (Fixed Points and Substitutions). *Let $C, \eta, \Delta \models^{\mathcal{E}} \varphi$ be a well-formed sequent and assume that $\Delta(Z(\mathbf{x})) = \alpha Z(\mathbf{x}).\varphi$. Then $\{\!|(\varphi)_\Delta|\!\}^{\mathcal{E}}_\pi = \{\!|(Z(\mathbf{x}))_\Delta|\!\}^{\mathcal{E}}_\pi$.*

Proof. Let $\{Y_1, \ldots, Y_n\} = fp(\varphi) \setminus \{Z\}$ and let

$$\varphi' = \varphi[Y_1(\mathbf{y}_1) := (Y_1(\mathbf{y}_1))_\Delta] \ldots [Y_n(\mathbf{y}_n) := (Y_n(\mathbf{y}_n))_\Delta].$$

Observe that

$$((Z(\mathbf{x}))_\Delta = (\Delta(Z(\mathbf{x})))_\Delta = (\alpha Z(\mathbf{x}).\varphi)_\Delta = \alpha Z(\mathbf{x}).\varphi' \tag{3}$$

and thus

$$(\varphi)_\Delta = (\varphi')_\Delta = \varphi'[Z(\mathbf{x}) := (Z(\mathbf{x}))_\Delta] = \varphi'[Z(\mathbf{x}) := \alpha Z(\mathbf{x}).\varphi']. \tag{4}$$

Putting things together, we have

$$\begin{aligned}
\{\!|(Z(\mathbf{x}))_\Delta|\!\}^{\mathcal{E}}_\pi &= \{\!|\alpha Z(\mathbf{x}).\varphi'|\!\}^{\mathcal{E}}_\pi && \text{[by (3)]}\\
&= \{\!|\varphi'|\!\}^{\mathcal{E}}_{\pi[Z(\mathbf{x}) \mapsto \alpha Z(\mathbf{x}).\varphi']} && \text{[by the semantics of fixpoints]}\\
&= \{\!|\varphi'[Z(\mathbf{x}) := \alpha Z(\mathbf{x}).\varphi']|\!\}^{\mathcal{E}}_\pi && \text{[by Lemma A.3 (2)]}\\
&= \{\!|(\varphi)_\Delta]|\!\}^{\mathcal{E}}_\pi && \text{[by (4)].}
\end{aligned}$$

□

Lemma 4.3 (Backwards Soundness). *Every rule of the tableau system is backwards sound.*

Proof. For each rule, we have to show that if the sequents in the conclusion are true, then also the sequent in the premise is true. The proof follows almost directly from the definition of the semantics of the logic. We only inspect some cases:

- Consider the rule
$$(\wedge)\ \frac{C, \eta, \Delta \models^{\mathcal{E}} \varphi \wedge \psi}{C, \eta, \Delta \models^{\mathcal{E}} \varphi \quad C, \eta, \Delta \models^{\mathcal{E}} \psi}.$$
Assume that the sequents in the conclusion are true, i.e., that $(C, \eta) \in \{\!|(\varphi)_\Delta|\!\}^{\mathcal{E}}_\pi$ and $(C, \eta) \in \{\!|(\psi)_\Delta|\!\}^{\mathcal{E}}_\pi$, for $\pi \in PEnv$. Just observe that $(\varphi \wedge \psi)_\Delta = (\varphi)_\Delta \wedge (\psi)_\Delta$. Then, we immediately conclude that $(C, \eta) \in \{\!|(\varphi \wedge \psi)_\Delta|\!\}^{\mathcal{E}}_\pi$, i.e., that the sequent $C, \eta, \Delta \models^{\mathcal{E}} \varphi \wedge \psi$ is true, as desired.
- Consider the rule
$$(\Diamond)\ \frac{C, \eta, \Delta \models^{\mathcal{E}} \langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle \varphi}{C', \eta[z \mapsto e], \Delta \models^{\mathcal{E}} \varphi},$$
where $e \in \mathcal{E}[C]$ and $C \xrightarrow{\eta(\mathbf{x}), \overline{\eta(\mathbf{y}) < e}}_{\mathrm{a}} C'$.
Assume that the sequent in the conclusion is true, i.e., that $(C', \eta[z \mapsto e]) \in \{\!|(\varphi)_\Delta|\!\}^{\mathcal{E}}_\pi$, for $\pi \in PEnv$. Then, by definition of the semantics, we immediately deduce that $(C, \eta) \in \{\!|\langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle (\varphi)_\Delta|\!\}^{\mathcal{E}}_\pi$. Since $(\langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle \varphi)_\Delta = \langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle (\varphi)_\Delta$, this proves that the sequent $C, \eta, \Delta \models^{\mathcal{E}} \langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle \varphi$ in the premise is true.
- Consider the rule:
$$(\mathrm{Int})\ \frac{C, \eta, \Delta \models^{\mathcal{E}} (\alpha Z(\mathbf{x}).\varphi)(\mathbf{y})}{C, \eta, \Delta' \models^{\mathcal{E}} Z(\mathbf{y})},$$
where $\Delta' = \Delta[Z(\mathbf{x})x \mapsto \alpha Z(\mathbf{x}).\varphi]$ and $\alpha \in \{\nu, \mu\}$.
Assume that the sequent in the conclusion is true, i.e., $(C, \eta) \in \{\!|(Z(\mathbf{y}))_{\Delta'}|\!\}^{\mathcal{E}}_\pi$, for $\pi \in PEnv$. We just need to observe that $(Z(\mathbf{y}))_{\Delta'} = ((\alpha Z(\mathbf{x}).\varphi)(\mathbf{y}))_{\Delta'} = ((\alpha Z(\mathbf{x}).\varphi)(\mathbf{y}))_\Delta$ where the last equality is motivated by the fact that Δ and Δ' differ only on Z, which is not free in $(\alpha Z(\mathbf{x}).\varphi)(\mathbf{y})$ (and neither is it in any formula ψ such that $(\alpha Z(\mathbf{x}).\varphi)(\mathbf{y}) \to^*_\Delta \psi$). Hence,

$$\{\!|(Z(\mathbf{y}))_{\Delta'}|\!\}^{\mathcal{E}}_{\pi} = \{\!|((\alpha Z(\mathbf{x}).\varphi)(\mathbf{y}))_{\Delta}|\!\}^{\mathcal{E}}_{\pi}.$$

This immediately implies that the sequent $C, \eta, \Delta \models^{\mathcal{E}} (\alpha Z(\mathbf{x}).\varphi)(\mathbf{y})$ in the premise is true.

- Consider the rule:

$$(\mathsf{Unf}_\alpha)\ \frac{C, \eta, \Delta \models^{\mathcal{E}} Z(\mathbf{z})}{C, \eta', \Delta \models^{\mathcal{E}} \varphi},$$

where $\Delta(Z(\mathbf{x})) = \alpha Z(\mathbf{x}).\varphi$, for $\alpha \in \{\nu, \mu\}$ and $\eta' = \eta[\mathbf{x} \mapsto \eta(\mathbf{z})]$.

Assume that the sequent in the conclusion is true, i.e., $(C, \eta') \in \{\!|(\varphi)_\Delta|\!\}^{\mathcal{E}}_{\pi}$, for some $\pi \in PEnv$. By Lemma B.1, $\{\!|(\varphi)_\Delta|\!\}^{\mathcal{E}}_{\pi} = \{\!|(Z(\mathbf{x}))_\Delta|\!\}^{\mathcal{E}}_{\pi}$. Thus, we have

$$\begin{aligned}
&(C, \eta') = \\
&\quad (C, \eta[\mathbf{x} \mapsto \eta(\mathbf{z})]) \in \{\!|(Z(\mathbf{x}))_\Delta|\!\}^{\mathcal{E}}_{\pi} \\
&\quad = \{\!|(Z(\mathbf{z})[\mathbf{x}/\mathbf{z}])_\Delta|\!\}^{\mathcal{E}}_{\pi} \\
&\quad = \{\!|(Z(\mathbf{z}))_\Delta[\mathbf{x}/\mathbf{z}]|\!\}^{\mathcal{E}}_{\pi} \\
&\quad = \{\!|(Z(\mathbf{z}))_\Delta|\!\}^{\mathcal{E}}_{\pi}[\mathbf{x}/\mathbf{z}].
\end{aligned}$$

Thus, $(C, \eta) \in \{\!|(Z(\mathbf{z}))_\Delta|\!\}^{\mathcal{E}}_{\pi}$, which means that the sequent in the premise is true. □

B.2 Proofs for Section 4.3

B.2.1 Finiteness.

Lemma 4.7 (Fixpoint Introduction). *Let $\mathcal{E}$ be a PES and let τ be a tableau for a closed formula φ. Let n be any node in the tableau labelled by $C, \eta, \Delta \models X(\mathbf{x})$ for some $X(\mathbf{x}) \in \mathcal{X}$.*

(1) If n has a descendant n' labelled by $C', \eta', \Delta' \models Y(\mathbf{y})$, for some $Y(\mathbf{y}) \in \mathcal{X}$, and Y is not introduced between n and n', then $X \sqsubseteq^*_d Y$.
(2) If n has a descendant n' that introduces X, then there is a node n'' between n and n' with consequent $Y(\mathbf{y})$ such that $X \sqsubseteq_d Y$ (hence, $X \sqsubseteq^*_d Y$ and $X \neq Y$).

Proof.

(1) We prove the property by induction on the number of sequents between n and n' labelled by a proposition. Let n_i be such nodes, labelled by $C_i, \eta_i, \Delta_i \models Z_i(\mathbf{z}_i)$ with $i \in \{1, \ldots, k\}$. Moreover, let $\alpha X(\mathbf{x}').\psi$ be the definition of X in the tableau.

($k = 0$) The only rule applicable to node n is (Unf_α); hence, the successor of n will have ψ as consequent. Since between n and n' there are no propositions and thus no other applications of (Unf_α) and Y is not introduced, necessarily $Y \in fp(\psi)$. Hence, either $Y = X$ or $X \sqsubseteq_d Y$. In any case, $X \sqsubseteq^*_d Y$.

($k > 0$) Consider node n_1 labelled by the sequent $C_1, \eta_1, \Delta_1 \models Z_1(\mathbf{z}_1)$. By inductive hypothesis

$$Z_1 \sqsubseteq^*_d Y. \tag{5}$$

Since between n and n_1 there are no propositions and thus no applications of the (Unf_α) rule, there are two possibilities: either $Z_1 \in fp(\psi)$ or Z_1 is introduced between n and n_1. In the first case, by inductive hypothesis $X \sqsubseteq^*_d Z_1$ and thus, recalling (5), we conclude $X \sqsubseteq^*_d Y$. In the second case, if φ_1 is the definition of Z_1 in the tableau, then φ_1 is a subformula of ψ, a fact that together with (5), again, implies $X \sqsubseteq^*_d Y$.

(2) Let $\alpha X(\mathbf{x}').\psi$ be the definition of X in the tableau. The only rule applicable to node n is (Unf_α), hence the successor of n will have ψ as consequent. Since all rules except (Unf_α)

reduce the size of the formula, there must be between n and n' an occurrence of the unfolding rule applied to some $Y \in fp(\alpha X(\mathbf{x}').\psi)$. By definition $X \sqsubseteq_d Y$ as desired (hence, $X \sqsubseteq_d^* Y$ and, clearly, $X \neq Y$, since $\sqsubseteq_d$ is irreflexive). □

Lemma 4.8 (Infinite Occurrences of Propositions in Tableaux). *Let $\mathcal{E}$ be a finitely branching PES. An infinite tableau for a closed formula φ contains an infinite path where some abstract proposition Z occurs (and thus is unfolded) infinitely often without being introduced.*

Proof. Let $\mathcal{E}$ be a finitely branching PES and let τ be a an infinite tableau for φ. Since the transition system of the PES is finitely branching, also the tableau τ is finitely branching. Hence, by Köenig's lemma, it includes an infinite path p. Since all tableau rules (see Figure 1) apart from (Unf_α) reduce the size of the consequent, path p must include infinitely many applications of (Unf_α) and thus infinitely many sequents having a proposition as consequent.

Since the only abstract propositions that can appear are those in φ, which are finitely many, there are abstract propositions that occur infinitely often (possibly with different event variables).

Denote by P the set of such abstract propositions and consider the suffix p' of p, where only abstract propositions in P occur.

Let $X \in P$ be maximal with respect to $\sqsubseteq_d^*$ and let p'' be a suffix of p' starting with an occurrence of X. Then X is not introduced in p'', otherwise, by Lemma 4.7(2), there would be a node n' in p'' with consequent $Y(\mathbf{y})$ such that $X \sqsubseteq_d^* Y$ and $X \neq Y$, contradicting the maximality of X. □

Theorem 4.9 (Tableaux Finiteness). *For a strongly regular PES $\mathcal{E}$ and a closed formula φ, every tableau for a sequent $C, \eta, \Delta \models^{\mathcal{E}} \varphi$ is finite. Hence, the number of tableaux for $C, \eta, \Delta \models^{\mathcal{E}} \varphi$ is finite.*

Proof. The proof is by contradiction. Suppose that there is an infinite tableau τ for the sequent $C, \eta, \Delta \models^{\mathcal{E}} \varphi$. By Lemma 4.8, in τ there is an infinite path π where a proposition X occurs infinitely many times without being introduced. Let $X(\mathbf{x}) = \alpha X(\mathbf{x}).\, \psi$ be the definition of X in the tableau.

By Lemma 3.10 the set of $\mathbf{x}$-pointed configurations of $\mathcal{E}$ is finite up to $\approx_r$. Since the proposition X is unfolded infinitely many times along π without being introduced, there are infinitely many sequents $C', \eta', \Delta' \models^{\mathcal{E}} X(\mathbf{x}')$ for which $X^{\uparrow}(\Delta')$ is the same node. Hence, the stop condition γ is necessarily satisfied at some point of the path, contradicting its infiniteness.

We next prove that also the number of tableaux is finite. Consider a tree where nodes are tableaux rooted $C, \eta, \Delta \models^{\mathcal{E}} \varphi$ and where the successors of each tableau τ are the tableaux obtained by extending τ with the application of a rule. Since $\mathcal{E}$ is strongly regular, the tree is finitely branching. If it were infinite, there would be an infinite path and thus there would be an infinite sequence of tableaux $(\tau_i)_{i \in \omega}$, such that τ_{i+1} extends τ_i. This in turn implies the existence of an infinite tableau, which contradicts the first part. □

B.2.2 Soundness and Completeness. The main result about soundness and completeness of the tableau system requires a technical lemma about fixpoints interpreted as approximants. First, we need to observe how the instantiation of a formula interact with the approximants.

Lemma B.2 (Grounding, Substitutions, and Approximants). *Let $C, \eta, \Delta \models \varphi$ be a well-formed sequent where $\varphi = \alpha Z(\mathbf{x}).\, \psi$ is a fixpoint formula. Then (a) $(\varphi)_\Delta$ is a fixpoint formula, (b) $(\varphi)_\Delta[\mathbf{y}/\mathbf{x}] = (\varphi[\mathbf{y}/\mathbf{x}])_\Delta$, and (c) for any $n \in \mathbb{N}$, it holds $((\varphi)_\Delta)^n = (\varphi^n)_\Delta$.*

Proof. Concerning point (a), reasoning as in the proof of Lemma B.1, it is easy to see that $(\varphi)_\Delta$ is a fixpoint formula. In fact, let $\{Y_1, \ldots, Y_n\} = fp(\psi) \setminus \{Z\}$ and let

$$\psi' = \psi[Y_1(\mathbf{y}_1) := (Y_1(\mathbf{y}_1))_\Delta] \ldots [Y_n(\mathbf{y}_n) := (Y_n(\mathbf{y}_n))_\Delta].$$

Then, observe that

$$(\varphi)_\Delta = (\alpha Z(\mathbf{x}).\psi)_\Delta = \alpha Z(\mathbf{x}).\psi'.$$

We now prove point (b) $(\varphi[\mathbf{y}/\mathbf{x}])_\Delta = (\varphi)_\Delta[\mathbf{y}/\mathbf{x}]$. Using the fact that $\alpha Z(\mathbf{x}).\psi = (\alpha Z(\mathbf{x}).\psi)(\mathbf{x})$, this is also almost immediate. In fact:

$$\begin{aligned}(\varphi[\mathbf{y}/\mathbf{x}])_\Delta &= ((\alpha Z(\mathbf{x}).\psi)(\mathbf{x})[\mathbf{y}/\mathbf{x}])_\Delta = ((\alpha Z(\mathbf{x}).\psi)(\mathbf{y}))_\Delta \\ &= (\alpha Z(\mathbf{x}).\psi')(\mathbf{y}) = (\alpha Z(\mathbf{x}).\psi')(\mathbf{x})[\mathbf{y}/\mathbf{x}] = (\varphi)_\Delta[\mathbf{y}/\mathbf{x}].\end{aligned}$$

Finally, we prove point (c), i.e., for any $n \in \mathbb{N}$, $(\varphi^n)_\Delta = ((\varphi)_\Delta)^n$, by induction on n. We focus on the case $\alpha = \nu$ (the proof is completely analogous when $\alpha = \mu$).

($n = 0$) Immediate, since $(\varphi^0)_\Delta = (\mathsf{T})_\Delta = \mathsf{T} = ((\varphi)_\Delta)^0$.

($n > 0$) Observe that

$$(\varphi^n)_\Delta = (\psi[Z(\mathbf{x}) := \varphi^{n-1}])_\Delta = \psi'[Z(\mathbf{x}) := (\varphi^{n-1})_\Delta],$$

where the last equality holds, since $fp(\varphi^{n-1}) \subseteq fp(\psi[Z(\mathbf{x}) := \varphi^{n-1}]) \subseteq fp(\psi) \setminus \{Z\}$, and

$$((\varphi)_\Delta)^n = (\alpha Z(\mathbf{x}).\psi')^n = \psi'[Z(\mathbf{x}) := ((\varphi)_\Delta)^{n-1}].$$

Thus, we can conclude, since, by inductive hypothesis, we know that $(\varphi^{n-1})_\Delta = ((\varphi)_\Delta)^{n-1}$. □

In words, the result above shows that the instantiation of the approximants of a fixed point formula φ coincides with the approximants of the instantiation of φ. For this reason, in the sequel, we will abuse the notation and write $(\varphi)^n_\Delta$ for $((\varphi)_\Delta)^n = (\varphi^n)_\Delta$.

Lemma 4.11 (Finite Approximants Properties). *Let $\mathcal{E}$ be a strongly regular PES, let $\pi \in PEnv_{\mathcal{E}}$ be a saturated proposition environment, and let $\varphi = \alpha Z(\mathbf{x}).\psi$ be a fixpoint formula. Then there exists $i \in \mathbb{N}$ such that $\{\!|\varphi|\!\}^{\mathcal{E}}_\pi = \{\!|\varphi^i|\!\}^{\mathcal{E}}_\pi$. Hence, for any configuration $C \in \mathcal{C}(\mathcal{E})$ and environment $\eta \in Env_{\mathcal{E}}$:*

(1) if $\varphi = \nu Z(\mathbf{x}).\psi$ and $(C, \eta) \notin \{\!|\varphi|\!\}^{\mathcal{E}}_\pi$, then $(C, \eta) \in \{\!|\varphi^n|\!\}^{\mathcal{E}}_\pi \setminus \{\!|\varphi^{n+1}|\!\}^{\mathcal{E}}_\pi$ for some $n \leq i$;
(2) if $\varphi = \mu Z(\mathbf{x}).\psi$ and $(C, \eta) \in \{\!|\varphi|\!\}^{\mathcal{E}}_\pi$, then $(C, \eta) \in \{\!|\varphi^{n+1}|\!\}^{\mathcal{E}}_\pi \setminus \{\!|\varphi^n|\!\}^{\mathcal{E}}_\pi$ for some $n \leq i$.

Proof. Let us first focus on the case $\varphi = \nu Z(\mathbf{x}).\psi$. Clearly, for all $i \in \mathbb{N}$, $\{\!|\varphi|\!\}^{\mathcal{E}}_\pi \subseteq \{\!|\varphi^i|\!\}^{\mathcal{E}}_\pi$. Assume by contradiction that the inclusion is always strict, i.e., $\{\!|\varphi|\!\}^{\mathcal{E}}_\pi \subsetneq \{\!|\varphi^i|\!\}^{\mathcal{E}}_\pi$ for all $i \in \mathbb{N}$. This implies that $\{\!|\varphi^{i+1}|\!\}^{\mathcal{E}}_\pi \subsetneq \{\!|\varphi^i|\!\}^{\mathcal{E}}_\pi$ for all $i \in \mathbb{N}$.

Therefore, there is an infinite sequence of pairs $(C_i, \eta_i) \in \mathcal{C}(\mathcal{E}) \times Env_{\mathcal{E}}$, for $i \in \mathbb{N}$, such that $(C_i, \eta_i) \in \{\!|\varphi^i|\!\}^{\mathcal{E}}_\pi \setminus \{\!|\varphi^{i+1}|\!\}^{\mathcal{E}}_\pi$. We deduce that for each i, we have $(C_i, \eta_i) \in \{\!|\varphi^i|\!\}^{\mathcal{E}}_\pi$ and $(C_j, \eta_j) \notin \{\!|\varphi^i|\!\}^{\mathcal{E}}_\pi$ for all $j < i$. Then, by Lemma 3.9, we would have that $\langle C_j, \eta_j|_{fv(\varphi^i)}\rangle \not\approx_r \langle C_i, \eta_i|_{fv(\varphi^i)}\rangle$ for all $i \in \mathbb{N}$ and $j < i$. This would mean that there are infinitely many different equivalence classes of $fv(\varphi^i)$-pointed configurations with respect to $\approx_r$. Since $|fv(\varphi^i)| \leq |fv(\varphi)| = |\mathbf{x}|$ and $\mathcal{E}$ is strongly regular, this fact contradicts Lemma 3.10. Therefore, there must exist $i \in \mathbb{N}$ such that $\{\!|\varphi|\!\}^{\mathcal{E}}_\pi = \{\!|\varphi^i|\!\}^{\mathcal{E}}_\pi$. Point (1) then immediately follows.

If $\varphi = \mu Z(\mathbf{x}).\psi$, then a dual reasoning proves that there is $i \in \mathbb{N}$ such that $\{\!|\varphi^i|\!\}^{\mathcal{E}}_\pi = \{\!|\varphi^{i+1}|\!\}^{\mathcal{E}}_\pi$ and thus $\{\!|\varphi|\!\}^{\mathcal{E}}_\pi = \{\!|\varphi^i|\!\}^{\mathcal{E}}_\pi$ and, again, point (2) immediately follows. □

Lemma 4.13 (Shortening ν-Pseudo-Tableaux). *Let $\mathcal{E}$ be a strongly regular PES and let τ be a successful ν-pseudo-tableau. If τ has a false leaf and for all false leaves $C, \eta, \Delta \models X(\mathbf{z})$, the node $\Delta^\uparrow(X)$ is in τ, then there exists a successful ν-pseudo-tableaux τ', strictly smaller than τ, with a false leaf and where for all false leaves $C, \eta, \Delta \models X(\mathbf{z})$, the node $\Delta^\uparrow(X)$ is in τ'.*

Proof. Since τ is successful its leaves are labelled by sequents $C, \eta, \Delta \models \varphi$, where φ is T, $[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\,\psi$ or a proposition $X(\mathbf{y})$ defined in τ as a largest fixpoint or as some ν-approximant. If $\varphi = \mathsf{T}$, then clearly the sequent is true. If $\varphi = [\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\,\psi$, then the definition requires that the set $\mathrm{Succ}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}_{\mathcal{E}}(C, \eta)$ is empty, and thus, observing that $(\varphi)_\Delta = [\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\,(\psi)_\Delta$, we deduce that the

leaf is vacuously true. Also leaves labelled by a proposition, defined as a ν-approximant are true by definition of successful tableau.

Therefore, the only possibly false leaves must have as consequent some proposition $X(\mathbf{y})$ defined as a largest fixpoint $\nu X(\mathbf{x}).\psi$. Amongst such false leaves choose one, labelled $C, \eta, \Sigma \models X(\mathbf{y})$, say, such that there is no other false leaf labelled $C', \eta', \Sigma' \models X'(\mathbf{y}')$ with $\Sigma'^{\uparrow}(X')$ above $\Sigma^{\uparrow}(X)$ in τ.

Call w the node $\Sigma^{\uparrow}(X)$. By definition of tableau the node w is labelled by a sequent with consequent $(\nu X(\mathbf{x}).\psi)(\mathbf{y}')$. Let τ_w be the subtableau of τ rooted in the successor of w, labelled $\tilde{C}, \tilde{\eta}, \tilde{\Delta} \models X(\tilde{\mathbf{y}})$, say.

Consider the other leaves in τ_w labelled by false sequents involving the same introduction of the abstract proposition X, i.e., whose consequent is of the kind $C, \eta, \Sigma \models X(\mathbf{y}'')$ for suitable $C, \eta, \Sigma, \mathbf{y}''$, with $\Sigma^{\uparrow}(X) = w$. For each such leaf, the fact that the sequent is false means $(C, \eta) \notin \{\!|(X(\mathbf{y}''))_{\Delta}|\!\}^{\mathcal{E}}$ for $\pi \in PEnv$. Observe that by Lemma B.2(b)

$$(X(\mathbf{y}''))_{\Delta} = (\Delta(X(\mathbf{x}))[{}^{\mathbf{y}''}/_{\mathbf{x}}])_{\Delta} = (\nu X(\mathbf{x}).\psi[{}^{\mathbf{y}''}/_{\mathbf{x}}])_{\Delta} = (\nu X(\mathbf{x}).\psi)_{\Delta}[{}^{\mathbf{y}''}/_{\mathbf{x}}].$$

Hence $(C, \eta) \notin \{\!|(\nu X(\mathbf{x}).\psi)_{\Delta}[{}^{\mathbf{y}''}/_{\mathbf{x}}]|\!\}^{\mathcal{E}} = \{\!|(\nu X(\mathbf{x}).\psi)_{\Delta}|\!\}^{\mathcal{E}}[{}^{\mathbf{y}''}/_{\mathbf{x}}]$, by Lemma A.3(1), and thus, we have $(C, \eta[\mathbf{x} \mapsto \eta(\mathbf{y}'')]) \notin \{\!|(\nu X(\mathbf{x}).\psi)_{\Delta}|\!\}^{\mathcal{E}}$. In turn, by applying first Lemma B.2(a) and then Lemma 4.11(1), this implies that there is $n \in \mathbb{N}$ such that $(C, \eta[\mathbf{x} \mapsto \eta(\mathbf{y}'')]) \in \{\!|(\nu X(\mathbf{x}).\psi)^{n}_{\Delta}|\!\}_{\pi} \setminus \{\!|(\nu X(\mathbf{x}).\psi)^{n+1}_{\Delta}|\!\}_{\pi}$.

Take one of such leaves l such that the corresponding n is as small as possible. Let l be labelled with the sequent $C, \eta, \Sigma \models X(\mathbf{z})$. Since l is a leaf, the stop condition must be satisfied, i.e., there is an ancestor k of l labelled $C', \eta', \Sigma' \models X(\mathbf{z}')$ such that $\Sigma^{\uparrow}(X) = \Sigma'^{\uparrow}(X) = w$ (hence, k in τ_w) and $\langle C, \eta[\mathbf{x} \mapsto \eta(\mathbf{z})]|_{\mathbf{x}}\rangle \approx_r \langle C', \eta'[\mathbf{x} \mapsto \eta'(\mathbf{z}')]|_{\mathbf{x}}\rangle$. Furthermore, since $(C, \eta[\mathbf{x} \mapsto \eta(\mathbf{z})]) \notin \{\!|(\nu X(\mathbf{x}).\psi)^{n+1}_{\Delta}|\!\}_{\pi}$ and $fv((\nu X(\mathbf{x}).\psi)^{n+1}_{\Delta}) \subseteq \mathbf{x}$, by Lemma 3.9, we have

$$(C', \eta'[\mathbf{x} \mapsto \eta'(\mathbf{z}')]) \notin \{\!|\nu X(\mathbf{x}).(\psi)^{n+1}_{\Delta}|\!\}_{\pi}. \tag{6}$$

Now transform the tableau τ_w into a new tableau τ'_w by replacing each definition list Δ in a sequent of τ_w with Δ' defined as follows:

- if $\Delta^{\uparrow}(X) = w$, then $\Delta' = \Delta[X \mapsto (\nu X(\mathbf{x}).\psi)^{n}]$,
- otherwise $\Delta' = \Delta$.

Clearly τ'_w is a well-defined ν-pseudo-tableau. Moreover, τ'_w is successful. The only doubt could concern leaves that in τ_w were labelled $C', \eta', \Delta' \models X(\mathbf{y}')$ with $\Delta'^{\uparrow}(X) = w$ and thus in τ'_w become $C', \eta', \Delta'' \models X(\mathbf{y}')$ with $\Delta'' = \Delta'[X(\mathbf{x}) \mapsto (\nu X(\mathbf{x}).\psi)^{n}]$. However, these leaves are true in τ'_w. In fact, otherwise, we would have that $(C', \eta') \notin \{\!|(X(\mathbf{y}'))_{\Delta''}|\!\}_{\pi} = \{\!|(\Delta''(X(\mathbf{y}')))_{\Delta'}|\!\}_{\pi} = \{\!|(\nu X(\mathbf{x}).\psi)^{n}_{\Delta'}|\!\}_{\pi} = \{\!|(\nu X(\mathbf{x}).\psi)^{n}_{\Delta}|\!\}_{\pi}$, where the last equality follows from the fact that $X \notin fp((\nu X(\mathbf{x}).\psi)^{n})$. Therefore, $(C'', \eta''[\mathbf{x} \mapsto \eta''(\mathbf{y}')]) \notin \{\!|(\nu X(\mathbf{x})).\psi)^{n}|\!\}_{\pi}$, which means that there would be $m < n$ such that $(C'', \eta''[\mathbf{x} \mapsto \eta''(\mathbf{y}')]) \in \{\!|(\nu X(\mathbf{x}).\psi)^{m}_{\Delta}|\!\}_{\pi} \setminus \{\!|(\nu X(\mathbf{x}).\psi)^{m+1}_{\Delta}|\!\}_{\pi}$, contradicting the choice of n.

Additionally, after the transformation, the successor of the node k defined above is labelled $C', \eta'[\mathbf{x} \mapsto \eta'(\mathbf{z}')], \Sigma'' \models \psi$ where $\Sigma'' = \Sigma'[X \mapsto (\nu X(\mathbf{x}).\psi)^{n}]$. It is easy to see that $(\psi)_{\Delta''} = (\nu X(\mathbf{x}).\psi)^{n+1}_{\Delta''} = (\nu X(\mathbf{x}).\psi)^{n+1}_{\Delta}$ where the last equality is motivated by the fact that X does not occur free in $(\nu X(\mathbf{x}).\psi)^{n+1}$. Jointly with (6) this gives $(C', \eta'[\mathbf{x} \mapsto \eta'(\mathbf{z}')]) \notin \{\!|\nu X(\mathbf{x}).(\psi)^{n+1}_{\Delta}|\!\}_{\pi} = \{\!|(\psi)_{\Delta''}|\!\}_{\pi}$, namely, the sequent $C', \eta'[\mathbf{x} \mapsto \eta'(\mathbf{z}')], \Sigma'' \models \psi$ is false, since $(C', \eta'[\mathbf{x} \mapsto \eta'(\mathbf{z}')]) \notin \{\!|\psi|\!\}_{\pi'}$. Therefore, by backwards soundness, some leaf of τ'_w must be labelled by a false sequent.

Consider any such false leaf and the corresponding sequent $C'', \eta'', \Delta' \models Y(\mathbf{y}')$. If the corresponding leaf of τ_w is labelled $C'', \eta'', \Delta \models Y(\mathbf{y}')$, we can have $\Delta' = \Delta[X \mapsto (\nu X(\mathbf{x}).\psi)^{n}]$ if $\Delta^{\uparrow}(X) = w$, $\Delta' = \Delta$ otherwise. We analyse the two cases separately:

- if $\Delta' = \Delta[X \mapsto (\nu X(\mathbf{x}).\psi)^n]$, then $Y \neq X$, because otherwise, by construction, the leaf would be true, as observed above. Moreover, $\Delta'^{\uparrow}(Y) = \Delta'^{\uparrow}(Y) = w'$ must be in τ'_w, i.e., a descendant of w, otherwise, it would contradict the choice of w.
- if $\Delta' = \Delta$, and thus $\Delta'^{\uparrow}(Y) = \Delta'^{\uparrow}(Y) = w' \neq w$, then again w' must be after w, hence in τ'_w, otherwise, it would contradict the choice of w.

Therefore, we reduced to a strictly smaller subtableau τ'_w of τ, which is successful, with a false leaf and where all false leaves involve propositions introduced in τ'_w, as desired. □

Lemma 4.14 (Soundness). *Let $\mathcal{E}$ be a strongly regular PES and φ be a closed formula of $\mathcal{L}_{hp}$. If φ has a successful ν-pseudo-tableau (hence, in particular, if it has a successful tableau), then $\mathcal{E}$ satisfies φ.*

Proof. Assume that the sequent $C, \eta, \emptyset \models^{\mathcal{E}} \varphi$ has a successful tableau τ. Then τ is a successful ν-pseudo-tableau. For all leaves (and thus for all false leaves) $C, \eta, \Delta \models X(\mathbf{z}), \Delta^{\uparrow}(X)$ is in τ. Therefore, if it had a false leaf, by Lemma 4.13, we could continue building strictly smaller ν-pseudo-tableau with the same properties, contradicting the finiteness of τ. Hence, all the sequents labelling the leaves of τ must be true, a fact that, by Lemma 4.3, implies that all the nodes of τ are true and thus, in particular, the sequent $C, \eta, \emptyset \models^{\mathcal{E}} \varphi$ labelling the root is true, as desired. □

Lemma B.3 (Unfolding μ-fixpoints). Let $C, \eta, \Delta \models Z(\mathbf{z})$ be a sequent such that $\Delta(Z(\mathbf{x})) = \mu Z(\mathbf{x}).\psi$ or $\Delta(Z(\mathbf{x})) = (\mu Z(\mathbf{x}).\psi)^n$. If $C, \eta, \Delta \models Z(\mathbf{z})$ is true and the stop condition γ does not hold, then rule $(\mathrm{Unf}^{\mathrm{a}}_{\mu})$ is applicable and the conclusion is true.

Proof. Assume that the stop condition γ does not hold. If the sequent $C, \eta, \Delta \models Z(\mathbf{z})$ is true, i.e., $(C, \eta) \in \{\!|(Z(\mathbf{z}))_\Delta|\!\}_\pi$ for $\pi \in PEnv$, then, by Lemma A.3(1), $(C, \eta') \in \{\!|(Z(\mathbf{x}))_\Delta|\!\}_\pi$ and, since $(Z(\mathbf{x}))_\Delta = (\Delta(Z(\mathbf{x})))_\Delta$, then $(C, \eta') \in \{\!|(\Delta(Z(\mathbf{x})))_\Delta|\!\}_\pi$. When $\Delta(Z(\mathbf{x})) = \mu Z(\mathbf{x}).\psi$ some $k \in \mathbb{N}$ such that condition (1) holds is guaranteed to exist by applying first Lemma B.2(a) and then Lemma 4.11(2). When instead, $\Delta(Z(\mathbf{x})) = (\mu Z(\mathbf{x}).\psi)^n$, we need only Lemma B.2(c). Moreover, in this case, we have $k < n$, since $(C, \eta') \in \{\!|(\mu Z(\mathbf{x}).\psi)^n_\Delta|\!\}_\pi$ and $(C, \eta') \in \{\!|(\mu Z(\mathbf{x}).\psi)^{k+1}_\Delta|\!\}_\pi \setminus \{\!|(\mu Z(\mathbf{x}).\psi)^k_\Delta|\!\}_\pi$.

Since $Z \notin fp((\mu Z(\mathbf{x}).\psi)^k)$ and Δ and the definition lists Δ' differ only on Z, we have that $(\psi)_{\Delta'} = (\psi[Z(\mathbf{x}) := (\mu Z(\mathbf{x}).\psi)^k_\Delta])_\Delta = (\mu Z(\mathbf{x}).\psi)^{k+1}_\Delta$ and thus $(C, \eta') \in \{\!|(\psi)_{\Delta'}|\!\}_\pi$, i.e., the successor sequent $C, \eta', \Delta' \models \psi$ is true. □

Lemma 4.16 (Completeness). *Let $\mathcal{E}$ be a strongly regular PES and let φ be a closed formula of $\mathcal{L}_{hp}$. If $\mathcal{E}$ satisfies φ, then φ has a successful μ-pseudo tableau and thus a successful tableau.*

Proof. We prove that each true sequent $C, \eta, \Delta \models \varphi$ admits a successful μ-pseudo-tableau. We first construct the tableau inductively, showing which rule to apply and arguing that the conclusion of the rule is again a true sequent. Then, we will show that it is successful. We distinguish various cases according to the shape of φ:

- $\varphi = \mathsf{T}$
  The node has no successors.
- $\varphi = \psi_1 \wedge \psi_2$
  We apply rule $(\wedge)$, which produces two successors labelled $C, \eta, \Delta \models \psi_1$ and $C, \eta, \Delta \models \psi_2$. Clearly $(\varphi)_\Delta = (\psi_1)_\Delta \wedge (\psi_2)_\Delta$, and, since $(C, \eta) \in \{\!|(\varphi)_\Delta|\!\}_\pi$, by definition of the semantics both sequents are true.
- $\varphi = \psi_1 \vee \psi_2$

Since $(C,\eta) \in \{\!|(\varphi)_\Delta|\!\}_\pi$ and clearly $(\varphi)_\Delta = (\psi_1)_\Delta \vee (\psi_2)_\Delta$, then by definition of the semantics $(C,\eta) \in \{\!|(\psi_i)_\Delta|\!\}_\pi$, for some $i \in \{1,2\}$. We apply rule $(\vee_\mathsf{L})$ or $(\vee_\mathsf{R})$, accordingly, producing a single true successor labelled $C, \eta, \Delta \models \psi_i$.

- $\varphi = \langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z \rangle\!\rangle\, \psi$

  Since $(C,\eta) \in \{\!|(\varphi)_\Delta|\!\}_\pi$ and $(\varphi)_\Delta = \langle\!\langle \mathbf{x}, \overline{\mathbf{y}} \leq \mathrm{a}\, z \rangle\!\rangle\, (\psi)_\Delta$, by definition of the semantics there exists an event $e \in \mathcal{E}[C]$ such that $C \xrightarrow{\eta(\mathbf{x}),\, \eta(\mathbf{y}) < e}_{\mathrm{a}} C'$, $\eta' = \eta[z \mapsto e]$ and $(C',\eta') \in \{\!|(\psi)_\Delta|\!\}_\pi$. Then, we can apply rule $(\Diamond)$ producing a single true successor labelled $C', \eta', \Delta \models \psi$.
- $\varphi = [\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\, \psi$

  Let the set of successor be $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C,\eta) = \{(C_1,\eta_1),\ldots,(C_n,\eta_n)\}$. Note that successors $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C,\eta)$ are finite, since the PES $\mathcal{E}$ is strongly regular and thus finitely branching.

  If $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C,\eta)$ is not empty, since $(C,\eta) \in \{\!|(\varphi)_\Delta|\!\}_\pi$ and $(\varphi)_\Delta = [\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\, (\psi)_\Delta$, by definition of the semantics, then we have that $(C_i,\eta_i) \in \{\!|(\psi)_\Delta|\!\}_\pi$ for all $i \in [1,n]$. Then rule $(\Box)$ can be applied producing n true successors labelled $C_i, \eta_i, \Delta \models \psi$, for $i \in [1,n]$.

  Otherwise, if $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C,\eta)$ is empty, the node has no successor.
- $\varphi = (\alpha Z(\mathbf{x}).\psi)(\mathbf{z})$

  We apply rule (Fix) that produces a single successor labelled $C, \eta, \Delta' \models Z(\mathbf{z})$ where $\Delta' = \Delta[Z(\mathbf{x}) \mapsto \alpha Z(\mathbf{x}).\psi]$.

  We show that the sequent $C, \eta, \Delta' \models Z(\mathbf{z})$ is true, i.e., $(C,\eta) \in \{\!|(Z(\mathbf{z}))_{\Delta'}|\!\}_\pi$ for $\pi \in \mathit{PEnv}$. First note that $(\varphi)_\Delta = (\varphi)_{\Delta'}$, since $Z \notin fp(\varphi)$ and Δ and Δ' coincide on all propositions except Z. Therefore $(Z(\mathbf{z}))_{\Delta'} = (\Delta'(Z(\mathbf{z})))_{\Delta'} = (\varphi)_{\Delta'} = (\varphi)_\Delta$. Since $(C,\eta) \in \{\!|(\varphi)_\Delta|\!\}_\pi$, we have that $(C,\eta) \in \{\!|(Z(\mathbf{z}))_{\Delta'}|\!\}_\pi$, as desired.
- $\varphi = Z(\mathbf{z})$

  We know that the sequent $C, \eta, \Delta \models \varphi$ is true, i.e., $(C,\eta) \in \{\!|(Z(\mathbf{z}))_\Delta|\!\}_\pi$ for $\pi \in \mathit{PEnv}$. If γ holds, then the node has no successors. Otherwise, if γ is false, we distinguish two cases depending on the definition of Z in Δ:

  (1) $\Delta(Z(\mathbf{x})) = \nu Z(\mathbf{x}).\psi$

  Since $(C,\eta) \in \{\!|(Z(\mathbf{z}))_\Delta|\!\}_\pi$, we have that $(C,\eta') \in \{\!|(Z(\mathbf{x}))_\Delta|\!\}_\pi$ where $\eta' = \eta[\mathbf{x} \mapsto \eta(\mathbf{z})]$. By Lemma B.1 $\{\!|(Z(\mathbf{x}))_\Delta|\!\}_\pi = \{\!|(\psi)_\Delta|\!\}_\pi$ and thus $(C,\eta') \in \{\!|(\psi)_\Delta|\!\}_\pi$. Therefore, by using rule (Unf_ν), we produce a single true successor labelled $C, \eta', \Delta \models \psi$.

  (2) $\Delta(Z(\mathbf{x})) = \mu Z(\mathbf{x}).\psi$ or $(\mu Z(\mathbf{x}).\psi)^n$

  By Lemma B.3, since the sequent $C, \eta, \Delta \models \varphi$ is true, we can apply rule $(\mathsf{Unf}^{\mathrm{a}}_\mu)$, thus producing a true successor labelled

$$C, \eta', \Delta \models \psi,$$

  where $\eta' = \eta[\mathbf{x} \mapsto \eta(\mathbf{z})]$ and $\Delta' = \Delta[Z(\mathbf{x}) \mapsto (\mu Z(\mathbf{x}).\psi)^k]$, for some $k \in \mathbb{N}$ such that $(C,\eta') \in \{\!|(\mu Z(\mathbf{x}).\psi)^{k+1}_\Delta|\!\}_\pi \setminus \{\!|(\mu Z(\mathbf{x}).\psi)^k_\Delta|\!\}_\pi$, which is ensured to exist.

Note that we do not need to consider the case $\varphi = \mathsf{F}$, since the sequent $C, \eta, \Delta \models \varphi$ would be false.

The inductive construction produces a μ-pseudo-tableau where all sequents are true. The only reason why it might not be successful could be the presence of a leaf labelled $C, \eta, \Delta \models X(\mathbf{z})$ where $\Delta(X(\mathbf{x})) = (\mu X(\mathbf{x}).\psi)^k$. We next show that this would imply that $C, \eta, \Delta \models X(\mathbf{z})$ is false, hence it is not possible.

Assume that there is a leaf l labelled $C, \eta, \Delta \models X(\mathbf{z})$ where $\Delta(X(\mathbf{x})) = (\mu X(\mathbf{x}).\psi)^k$. We prove that $C, \eta, \Delta \models X(\mathbf{z})$ is false, namely, that $(C,\eta) \notin \{\!|(X(\mathbf{z}))_\Delta|\!\}_\pi$ for $\pi \in \mathit{PEnv}$. Observe that

$(X(\mathbf{z}))_\Delta = ((\mu X(\mathbf{x}).\psi)^k[\mathbf{z}/\mathbf{x}])_\Delta = (\mu X(\mathbf{x}).\psi)^k_\Delta[\mathbf{z}/\mathbf{x}]$, hence, by Lemma A.3(1), we can conclude by proving that

$$(C, \eta[\mathbf{x} \mapsto \eta(\mathbf{z})]) \notin (\mu X(\mathbf{x}).\psi)^k_\Delta. \tag{7}$$

Since l is a leaf, it must satisfy the stop condition, i.e., it must have an ancestor v labelled $C', \eta', \Delta' \models X(\mathbf{z}')$ such that $\Delta^\uparrow(X) = \Delta'^\uparrow(X)$ and $\langle C, \eta[\mathbf{x} \mapsto \eta(\mathbf{z})]|_\mathbf{x}\rangle \approx_r \langle C', \eta'[\mathbf{x} \mapsto \eta'(\mathbf{z}')]|_\mathbf{x}\rangle$. By construction, the successor of v must be labelled $C', \eta'[\mathbf{x} \mapsto \eta'(\mathbf{z}')], \Delta'' \models \psi$, where $\Delta'' = \Delta'[X(\mathbf{x}) \mapsto (\mu X(\mathbf{x}).\psi)^n]$ for some $n \in \mathbb{N}$ such that $(C', \eta'[\mathbf{x} \mapsto \eta'(\mathbf{z}')]) \in \{\!|(\mu X(\mathbf{x}).\psi)^{n+1}_{\Delta'}|\!\}_\pi \setminus \{\!|((\mu X(\mathbf{x}).\psi)^{n+1}_{\Delta'}|\!\}_\pi$, for $\pi \in PEnv$. Recall that each time a proposition is unfolded using rule $(\mathrm{Unf}^{\mathrm{a}}_\mu)$, the index of the approximant strictly decreases, hence $k \leq n$. Then, we have that $(C', \eta'[\mathbf{x} \mapsto \eta'(\mathbf{z}')]) \notin \{\!|(\mu X(\mathbf{x}).\psi)^k_{\Delta'}|\!\}_\pi$, since $\{\!|(\mu X(\mathbf{x}).\psi)^k_{\Delta'}|\!\}_\pi \subseteq \{\!|(\mu X(\mathbf{x}).\psi)^n_{\Delta'}|\!\}_\pi$. Therefore, by Lemma 3.9, we deduce

$$(C, \eta[\mathbf{x} \mapsto \eta(\mathbf{z})]) \notin \{\!|(\mu X(\mathbf{x}).\psi)^k_{\Delta'}|\!\}_\pi,$$

since π is vacuously saturated for $(\mu X(\mathbf{x}).\psi)^k_{\Delta'}$ that has no free propositions.

We then reach the desired conclusion (7) by showing that $(\mu X(\mathbf{x}).\psi)^k_{\Delta'} = (\mu X(\mathbf{x}).\psi)^k_\Delta$. Note that, since $\Delta^\uparrow(X) = \Delta'^\uparrow(X)$, no proposition $Y \in fp(\psi) \setminus \{X\}$ can be unfolded or introduced along the path from v to l. Otherwise, if φ_X and φ_Y are the fixpoint formula binding X and Y, respectively, then we would have $\varphi_Y \sqsubseteq^*_d \varphi_Y \sqsubseteq^*_d \varphi_X$, while $\sqsubseteq^*_d$ is a partial order. Thus, since the definition list is updated only when a proposition is unfolded or introduced, Δ and Δ' coincide on all propositions in $fp((\mu X(\mathbf{x}).\psi)^k) \subseteq fp(\psi) \setminus \{X\}$. Clearly this implies that $(\mu X(\mathbf{x}).\psi)^k_{\Delta'} = ((\mu X(\mathbf{x}).\psi)^k)_\Delta$, as desired. All in all, we can conclude that there is a successful μ-pseudo-tableau (hence, a successful tableau) for each true sequent. □

Theorem 4.17 (Soundness and Completeness of the Tableau System). *Given a strongly regular PES $\mathcal{E}$ and a closed formula φ of $\mathcal{L}_{hp}$, the formula φ has successful tableau if and only if $\mathcal{E}$ satisfies φ.*

Proof. Corollary of Lemmata 4.14 and 4.16. □

C TECHNICALITIES AND PROOFS FOR SECTION 5

C.1 Proofs for Section 5.1

Theorem 5.5 (language preservation). *Let $\mathcal{A}$ be an NPA and let $\equiv$ be an equivalence on the set of states, which is a bisimulation. Then $L(\mathcal{A}_{/\equiv}) = L(\mathcal{A})$.*

Proof. Let $\mathcal{A} = (Q, q_0, \to, F)$ be an NPA and let $\equiv$ be an equivalence on Q, which is a bisimulation. We prove the two inclusions separately.

Let us first show that $L(\mathcal{A}) \subseteq L(\mathcal{A}_{/\equiv})$. Let $T \in L(\mathcal{A})$ be a tree. Then there exists an accepting run $r : T \to Q$. It is easy to show that $r' : T \to Q_{/\equiv}$ defined by $r'(u) = [r(u)]_\equiv$ for all $u \in T$ is an accepting run for T in $\mathcal{A}_{/\equiv}$.

In fact, it is a run on T, since $r'(\epsilon) = [q_0]_\equiv$, which is the initial state of $\mathcal{A}_{/\equiv}$. Moreover, for all $u \in T$, if $u1, \ldots, un$ are its children, then $r(u) \to (r(u1), \ldots, r(un))$ in $\mathcal{A}$, hence $r'(u) = [r(u)]_\equiv \to_{/\equiv} ([r(u1)]_\equiv, \ldots, [r(un)]_\equiv) = (r'(u1), \ldots, r'(un))$, as desired.

Finally, r' is accepting, since any path $([q_0]_\equiv, [q_1]_\equiv, [q_2]_\equiv, \ldots)$ in r' arises from a path $(q_0, q_1, q_2, \ldots)$ in r, which is accepting, and by construction $[q]_\equiv \in \mathcal{F}_{/\equiv}$ iff $q \in F$.

For the converse inclusion $L(\mathcal{A}_{/\equiv}) \subseteq L(\mathcal{A})$, let $T \in L(\mathcal{A}_{/\equiv})$ be a tree. Then there exists an accepting run $r' : T \to Q_{/\equiv}$. We can define a corresponding run $r : T \to Q$ in $\mathcal{A}$ as follows: For any node $u \in T$, we inductively define $r(w) \in Q$ in a way that $r'(u) = [r(u)]_\equiv$. For the root $r(\epsilon) = q_0$. For any each node $u \in T$ with children $u1, \ldots, un$, we know that there is a transition in the quotient $r'(u) \to_{/\equiv} (r'(u1), \ldots, r'(un))$. By construction this means that there is a transition $q \to (q_1, \ldots, q_n)$ in $\mathcal{A}$ such that $r'(u) = [q]_\equiv$ and $r'(ui) = [q_i]_\equiv$ for $i \in [1, n]$. By induction, we have $r(u) \in Q$ such

that $r'(u) = [r(u)]_{\equiv}$, hence $r(u) \equiv q$. Since $\equiv$ is a bisimulation, there are $q'_1, \ldots, q'_n \in Q$ such that $r'(u) \to (q'_1, \ldots, q'_n)$ and $q_i \equiv q'_i$ for $i \in [1, n]$. Therefore, for $i \in [1, n]$, we can define $r(ui) = q'_i$ and obtain $[r(ui)]_{\equiv} = [q_i]_{\equiv} = r'(ui)$, as desired.

Run r is accepting, as it follows from the fact that r' is so and the way $\mathcal{F}_{/\equiv}$ is defined from $\mathcal{F}$. □

C.2 Proofs for Section 5.2

Lemma 5.8 ($\approx_f$ is a Bisimulation). *Let $\mathcal{E}$ be a strongly regular PES and let φ be a closed formula of $\mathcal{L}_{hp}$. Then the future equivalence $\approx_f$ on $\mathcal{A}_{\mathcal{E},\varphi}$ is a bisimulation and it is of finite index.*

Proof. Let $q_1 = (C_1, \eta_1, \psi_1), q_2 = (C_2, \eta_2, \psi_2) \in Q$ be states of $\mathcal{A}_{\mathcal{E}(\mathcal{N}),\varphi}$ such that $q_1 \approx_m q_2$. Thus,

(i) there are a formula ψ and substitutions $\sigma_i : fv(\psi) \to fv(\psi_i)$, $i \in \{1, 2\}$, such that $\psi\sigma = \psi_i$ and
(ii) the $fv(\psi)$-pointed configurations $\langle C_1, \eta_1 \circ \sigma_1\rangle, \langle C_2, \eta_2 \circ \sigma_2\rangle$ have isomorphic pointed residuals via some isomorphism $\iota : \mathcal{E}[C_1] \to \mathcal{E}[C_2]$.

The first observation is that (i) and (ii) above imply that $q_1 \in F_i$ iff $q_2 \in F_i$, for $i \in [0, h]$, i.e., condition (1) of the definition of bisimulation (Definition 5.4).

In fact, by item (i), ψ_1 and ψ_2 differ at most for the name of the event variables, hence they have the same outer most operator and the same alternation depth. Moreover, if ψ_1 and hence ψ_2 are modal formulae, e.g., $\psi_1 = \langle\!|\mathbf{x}_1, \overline{\mathbf{y}_1} < \mathrm{a}\, z_1|\!\rangle \psi'_1$ and $\psi_2 = \langle\!|\mathbf{x}_2, \overline{\mathbf{y}_2} < \mathrm{a}\, z_2|\!\rangle \psi'_2$, then $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x}_1, \overline{\mathbf{y}_1} < \mathrm{a}z_1}(C_1, \eta_1) = \emptyset$ iff $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x}_2, \overline{\mathbf{y}_2} < \mathrm{a}z_2}(C_2, \eta_2) = \emptyset$. This follows from the more general observation that

$$C_1, \xrightarrow{\eta(\mathbf{x}_1), \overline{\eta(\mathbf{y}_1)} < e_1}_{\mathrm{a}} C'_1 \quad \text{iff} \quad C_2, \xrightarrow{\eta(\mathbf{x}_2), \overline{\eta(\mathbf{y}_2)} < \iota(e_1)}_{\mathrm{a}} C'_2. \tag{8}$$

In fact, let $C_1, \xrightarrow{\eta(\mathbf{x}_1), \overline{\eta(\mathbf{y}_1)} < e_1}_{\mathrm{a}} C'_1$ for some event e_1. Since e_1 is enabled in $\mathcal{E}[C_1]$ and ι is an isomorphism of $\mathcal{E}[C_1]$ and $\mathcal{E}[C_2]$, the image $\iota(e_1)$ is enabled in $\mathcal{E}[C_2]$. Moreover, for all $x_1 \in \mathbf{x}_1$, $\eta_1(x_1) < e_1$ and for all $y_1 \in \mathbf{y}_1$, $\neg(\eta_1(y_1) < e_1)$. Since $\psi\sigma_1 = \psi_1$, we get that for all $x \in \mathbf{x}$, $\eta_1 \circ \sigma_1(x) < e_1$ and for all $y \in \mathbf{y}$, $\neg(\eta_1 \circ \sigma_1(y) < e_1)$. Using the fact that ι is an isomorphism of residuals of pointed configurations, we deduce that for all $x \in \mathbf{x}$, $\eta_2 \circ \sigma_2(x) < \iota(e_1)$ and for all $y \in \mathbf{y}$, $\neg(\eta_2 \circ \sigma_2(y) < \iota(e_1))$. Thus, we get that for all $x_2 \in \mathbf{x}_2$, $\eta_2(x_2) < \iota(e_1)$ and for all $y_2 \in \mathbf{y}_2$, $\neg(\eta_2(y_2) < \iota(e_1))$, which means $C_2, \xrightarrow{\eta(\mathbf{x}_2), \overline{\eta(\mathbf{y}_2)} < \iota(e_1)}_{\mathrm{a}} C'_2$, as desired. The converse implication is immediate by symmetry.

From the above, recalling that the priority of a formula depends on the shape, the alternation depth, and for modal formulae, on the successor configurations, we conclude that condition (1) holds.

Now, let us focus on the condition (2). We proceed by cases, on the form of the formula ψ_1 (which, as observed, is of the same shape as ψ_2) and we show only that all transitions of q_1 are simulated by q_2, since the other direction follows by symmetry.

- $\psi_1 = \mathsf{T}$ or $\psi_1 = \mathsf{F}$
  Trivial, since the only transitions of q_1 and $q_2$are $q_1 \to (q_1)$ and $q_2 \to (q_2)$, and $q_1 \approx_f q_2$.
- $\psi_1 = \psi'_1 \wedge \psi''_1$
  The only transition of q_1 is $q_1 \to (q'_1, q''_1)$ where $q'_1 = (C, \eta, \psi'_1)$ and $q''_1 = (C, \eta, \psi''_1)$. By item (i), we immediately get that $\psi = \psi' \wedge \psi''$ and $\psi_2 = \psi'_2 \wedge \psi''_2$, with $\psi'\sigma_i = \psi'_i$ and $\psi''\sigma_i = \psi''_i$ for $i \in \{1, 2\}$.
  We can thus consider the only transition of q_2, i.e., $q_2 \to (q'_2, q''_2)$ where $q'_2 = (C_2, \eta_2, \psi'_2)$ and $q''_2 = (C_2, \eta_2, \psi''_2)$. Since we already knew that $\langle C_1, (\eta_1 \circ \sigma_1)|_{fv(\psi)}\rangle \approx_r \langle C_2, (\eta_2 \circ \sigma_2)|_{fv(\psi)}\rangle$, just observing that $fv(\psi'), fv(\psi'') \subseteq fv(\psi)$, we deduce

$\langle C_1, (\eta_1 \circ \sigma_1)|_{fv(\psi')}\rangle \approx_r \langle C_2, (\eta_2 \circ \sigma_2)|_{fv(\psi')}\rangle$ and $\langle C_1, (\eta_1 \circ \sigma_1)|_{fv(\psi'')}\rangle \approx_r \langle C_2, (\eta_2 \circ \sigma_2)|_{fv(\psi'')}\rangle$. Thus, we conclude that $q_1' \approx_f q_2'$ and $q_1'' \approx_f q_2''$, as desired.

- $\psi_1 = \psi_1' \vee \psi_1''$, $\psi_1 = (\alpha X_1(\mathbf{x}_1).\psi_1')(\mathbf{y})$, and $\psi_1 = X_1(\mathbf{x}_1)$.
  Analogous to the previous case.
- $\psi_1 = \langle\!|\mathbf{x}_1, \overline{\mathbf{y}_1} < \mathrm{a}\, z_1|\!\rangle\psi_1'$
  There are two possibilities. If $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x}_1, \overline{\mathbf{y}_1} < \mathrm{a} z_1}(C_1, \eta_1) = \emptyset$, then the only transition of q_1 is $q_1 \to (q_1)$. We already observed that in this case also $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x}_2, \overline{\mathbf{y}_2} < \mathrm{a} z_2}(C_2, \eta_2) = \emptyset$. and thus, we can take the only transition of q_2 i.e., $q_2 \to (q_2)$ and conclude, since $q_1 \approx_f q_2$.
  If instead $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x}_1, \overline{\mathbf{y}_1} < \mathrm{a} z_1}(C_1, \eta_1) = \emptyset$, then the transitions of q_1 are $q_1 \to (q_1')$ with $q_1' = (C_1', \eta[z_1 \mapsto e_1], \psi_1')$ for $C_1, \xrightarrow{\eta(\mathbf{x}_1), \overline{\eta(\mathbf{y}_1)} < e_1}_{\mathrm{a}} C_1'$. We observed that, in this case $C_2 \xrightarrow{\eta_2(\mathbf{x}), \eta_2(\mathbf{y}) < \iota(e)}_{\mathrm{a}} C_2'$ and thus for q_2, we can take the transition $q_2 \to (q_2')$ with $q_2' = (C_2', \eta_2[z_2 \mapsto \iota(e_1)], \psi_2')$. Since $e_i \in \mathcal{E}[C_i]$, $i \in \{1, 2\}$, and ι is an isomorphism, for all $e_1' \in \mathcal{E}[C_1]$, $e_1 < e_1'$ iff $\iota(e_1) < \iota(e_1')$. Moreover, by (i), we get that $\psi = \langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle\psi'$ and observing that $fv(\psi_i') \subseteq fv(\psi_i) \cup \{z_i\}$ and $fv(\psi') \subseteq fv(\psi) \cup \{z\}$, we have that $\psi'\sigma_i[z \mapsto z_i] = \psi_i'$ for $i \in \{1, 2\}$. Putting together the previous facts, we obtain also that the pointed configurations $\langle C_1', (\eta[z_1 \mapsto e_1] \circ \sigma_1[z \mapsto z_1])|_{fv(\psi')}\rangle$ and $\langle C_2', (\eta_2[z_2 \mapsto \iota(e_1')] \circ \sigma_2[z \mapsto z_2])|_{fv(\psi')}\rangle$ have isomorphic pointed residuals, with an isomorphism, which is the restriction of ι to $\mathcal{E}[C_1']$. Hence, we conclude that $q_1' \approx_f q_2'$.
- $\psi = [\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\psi_1$
  Analogous to the previous case.

To show that $\approx_f$ is of finite index, observe that for a fixed subformula $\psi \in \mathrm{sf}(\varphi)$, there is a finite number of $\approx_r$-equivalence classes of $fv(\psi)$-pointed configurations. Since the number of subformulae in $\mathrm{sf}(\varphi)$ is itself finite, we conclude. □

We next observe some properties of paths in automata in relation to fixpoint formulae. These are easy adaptations of similar results already proved for tableaux (Section 4.3).

Lemma C.1 (Fixpoint Introduction). *Let $\mathcal{E}$ be a PES, let φ be a closed formula in $\mathcal{L}_{hp}$, and let $\mathcal{A}_{\mathcal{E},\varphi}$ be the corresponding automaton. For any path p in $\mathcal{A}_{\mathcal{E},\varphi}$ and state q with formula $X(\vec{x})$*

(1) if there exists a state q' after q, with formula $Y(\mathbf{y})$ and Y is not introduced between q and q', then $X \sqsubseteq_d^* Y$;
(2) if there exists a state q' after q that introduces X, then there is a state between q and q' with formula $Y(\mathbf{x})$ with $X \sqsubseteq_d^* Y$.

Proof. Straightforward adaptation of the proof of Lemma 4.7. □

Lemma C.2 (Properties of Infinite Paths). *Let $\mathcal{E}$ be a PES, let φ be a closed formula in $\mathcal{L}_{hp}$, and let $\mathcal{A}_{\mathcal{E},\varphi}$ be the corresponding NPA. Each infinite path in $\mathcal{A}_{\mathcal{E},\varphi}$ has a suffix $p = (q_0, q_1, \ldots)$ such that one the following properties holds:*

(1) for all j, $q_j = (C, \eta, \psi)$, with ψ of the kind T, F, or $[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\psi'$, $\langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle\psi'$ with $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a} z}(C, \eta) = \emptyset$;
(2) the set of abstract propositions that occur infinitely often has a $\sqsubseteq_d^*$-largest element X (which thus has the largest alternation depth) and X is never introduced in p.

Proof. Consider an infinite path in the automaton $\mathcal{A}_{\mathcal{E},\varphi}$. If from some point on $q_j = (C, \eta, \psi)$, where ψ of the kind T, $\mathsf{F}[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\psi'$ or $\langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle\psi'$ with $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a} z}(C, \eta) = \emptyset$, we are done.

Otherwise, inspecting Definition 5.6, we see that the remaining transitions reduce the size of the formula associated with the state, the only exceptions being the unfolding transition. Therefore, infinitely many transitions must be unfoldings, meaning that there are infinitely many states with a proposition as formula.

Consider a suffix $p = (q_0, q_1, \ldots)$ of the path such that all propositions occur infinitely often and let X one of these proposition that is $\sqsubseteq_d^*$-maximal. Consider a suffix $p' = (q_\ell, q_{\ell+1}, \ldots)$ starting with a state q_ℓ having an occurrence of X as formula. Reasoning exactly as in Lemma 4.7, we can deduce that X is never introduced in p'.

Moreover, X is a $\sqsubseteq_d^*$-maximum. In fact, let Y be another maximal abstract proposition occurring infinitely often in p'. Then, we can consider states $q_i = (C_i, \eta_i, X(\mathbf{x}_i))$, $q_j = (C_j, \eta_j, Y(\mathbf{y}_j))$, and $q_i = (C_k, \eta_k, X(\mathbf{x}_k))$. If Y is not introduced between q_i and q_j then, by Lemma C.1(1), $X \sqsubseteq_d^* Y$, hence, by maximality, $X = Y$. Similarly, if X is not introduced between q_j and q_k then $Y \sqsubseteq_d^* X$, hence, by maximality, $X = Y$. Otherwise, it is easy to see that if ψ_X and ψ_Y are the fixpoint formulae binding X and Y, respectively, ψ_Y should be a strict subformula of ψ_X that, in turn, should be a strict subformula of ψ_Y, leading to a contradiction. □

Lemma 5.10 (Pruned Runs are Finite). *Let $\mathcal{E}$ be a strongly regular PES, let φ be a closed formula of $\mathcal{L}_{hp}$, and let $\mathcal{A}_{\mathcal{E},\varphi}$ be the corresponding NPA. For any run r of $\mathcal{A}_{\mathcal{E},\varphi}$ on a k-tree $\mathcal{T}$, the corresponding pruned run $\mathcal{T}^{(r)}$ is finite.*

Proof. We proceed by contradiction. We assume that $\mathcal{T}^{(r)}$ is infinite and we prove that it includes a repetition, in contrast with its definition. Since $\mathcal{T}^{(r)}$ is boundedly branching, by König's lemma, there is an infinite path $p = (u_0, u_1, \ldots)$ in $\mathcal{T}^{(r)}$. Hence, by Lemma C.2, there is a suffix $p' = (u_h, u_{h+1}, \ldots)$ such that one of the following properties hold and in both cases we conclude.

(1) $r(u_i) = q = (C, \eta, \psi)$ for all $i \geq h$, with ψ is of the kind T, F, or $\langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle\psi'$, $[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\psi'$ with $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C, \eta) = \emptyset$. Hence, u_i is a repetition.
(2) There is an abstract proposition X that occurs infinitely often and X is never introduced in p'. Since X occurs infinitely often in p' and, by Lemma 5.8, the equivalence $\approx_f$ is of finite index, there are two nodes u_i, u_j in p' with formulae $X(\mathbf{x}_i)$ and $X(\mathbf{x}_j)$, respectively, such that $r(u_i) \approx_f r(u_j)$. Therefore, u_j is a repetition. □

Lemma 5.12 (Maximal Priority Repetitions). *Let $\mathcal{E}$ be a PES, let φ be a closed formula of $\mathcal{L}_{hp}$, and let r be an accepting run of the NPA $\mathcal{A}_{\mathcal{E},\varphi}$ on a k-tree $\mathcal{T}$. For each infinite accepting path $p = (u_0, u_1, \ldots)$ in r there exists a repetition u_i of priority $\mathcal{F}(p)$.*

Proof. Let $p = (u_0, u_1, \ldots)$ be an infinite accepting path in r. By Lemma C.2, there is a suffix $p' = (u_h, u_{h+1}, \ldots)$ of p such that one of the following properties hold.

(1) $r(u_i) = q = (C, \eta, \psi)$ for all $i \geq h$, with ψ the kind T, F, or $\langle\!|\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z|\!\rangle\psi'$, $[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\psi'$ with $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C, \eta) = \emptyset$. The fact that the run and thus the path are accepting reduces the possible shapes of ψ to T and $[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\psi'$ with $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C, \eta) = \emptyset$. Hence, u_i is a repetition. Moreover, $\mathcal{F}(q) = 0$ and, since, $r(u_h) = r(u_i) = q \in F_0$ for $h \geq i$, this is the only priority repeating infinitely often, hence the largest.
(2) The set of abstract proposition that occurs infinitely often in p' has a $\sqsubseteq_d^*$-largest element X (which thus has the largest alternation depth) and X is never introduced in p'. Since X occurs infinitely often in p' and, by Lemma 5.8, the equivalence $\approx_f$ is of finite index, there are two nodes u_i, u_j in p' with formulae $X(\mathbf{x}_i)$ and $X(\mathbf{x}_j)$, respectively, such that $r(u_i) \approx_f r(u_j)$. We next observe that $r(u_i), r(u_j) \in F_l$ with $l = \mathcal{F}(p)$. In fact, recall that

$\mathrm{ad}(X)$ is maximal among the propositions occurring infinitely often. Thus, if there were $l' > l$ such that $r(u_i) \in F_{l'}$ infinitely often, there should be some Y such that

- Y is quantified in a ν-subformula and $\mathrm{ad}(Y) > \mathrm{ad}(X)$, contradicting the maximality of $\mathrm{ad}(X)$;
- Y is quantified in a μ-subformula and $\mathrm{ad}(Y) \geq \mathrm{ad}(X)$, contradicting the fact that the run r is accepting.

Summing up, $r(u_i) \approx_f r(u_j)$, state $r(u_i)$ has a proposition $X(\mathbf{x}_i)$ as a formula, and X is not introduced between u_i and u_j, hence u_j is the desired repetition. □

Lemma 5.13 (Avoiding Noisy Repetitions). *Let $\mathcal{E}$ be a PES and let φ be a closed formula of $\mathcal{L}_{hp}$. If $L(\mathcal{A}_{\mathcal{E},\varphi}) \neq \emptyset$, then $\mathcal{A}_{\mathcal{E},\varphi}$ has an accepting run r without noisy repetitions.*

Proof. Assume by contradiction that $L(\mathcal{A}_{\mathcal{E},\varphi}) \neq \emptyset$ but all accepting runs contain noisy repetitions. Let r be an accepting run on a k-tree $\mathcal{T}$ that has a minimal number of repetitions (this is finite, as already observed).

Let u be a noisy repetition in r. Since the run is accepting, the only possibility is that the formula of $r(u)$ is a proposition, say, $X(\mathbf{x})$ quantified in a least fixpoint. Let $u' = \Omega(u)$, i.e., u' is the closest ancestor of u such that $r(u) \approx_f r(u')$ and X is not introduced between u' and u, and let $\mathcal{T}_u$ and $\mathcal{T}_{u'}$ be the subtrees of $\mathcal{T}$ rooted in u and u', respectively.

Since $r(u) \approx_f r(u')$ and $\approx_f$ is an NPA bisimulation (see Lemma 5.8), we can replace $\mathcal{T}_{u'}$ by $\mathcal{T}_u$ and get a new valid run r'. It is accepting, since all paths in r' are obtained from paths in r by removing finitely many nodes.

We argue r' has strictly less repetitions than r, contradicting its minimality. In fact, u is no longer a repetition. Moreover, no new noisy repetitions are created. In fact, nodes that were not repetitions cannot become repetitions. Still, we could doubt that repetitions that were not noisy become so. For repetitions that were not in $\mathcal{T}_{u'}$, the ancestors are unchanged, hence they are noisy after the transformation if and only if they were before. Repetitions in $\mathcal{T}_{u'}$ but not in $\mathcal{T}_u$ are removed. Let us thus focus on repetitions in $\mathcal{T}_u$. Let v be one of such repetitions. Assume that u has odd priority and it is not noisy due to the presence of an ancestor w that is a repetition of even priority. Clearly, w cannot be an ancestor of u, otherwise u would not be noisy. Hence, it must be between u and v. Still, we could think that after the transformation u could become noisy, because $w' = \Omega(w)$ is between u' and u, hence it is removed and w ceases to be a repetition. However, if this were the case let $Y(\mathbf{y})$ and $Y(\mathbf{y}')$ be the formulae associated with w and w', respectively. Since X is not introduced between u' and u, hence between w' and u, by Lemma 4.7(1), $Y \sqsubseteq_d^* X$. Since Y is not introduced between $w' = \Omega(w)$ and w, hence between u and w, again by Lemma 4.7(1), $X \sqsubseteq_d^* Y$. Thus, $X = Y$, but this is absurd, since X and Y are quantified in a least and greatest fixpoint, respectively. □

Lemma C.3 (Pruned Run Without Noise). *Let $\mathcal{E}$ be a PES, φ be a closed formula of $\mathcal{L}_{hp}$, and let r be a run of the NPA $\mathcal{A}_{\mathcal{E},\varphi}$ on a k-tree $\mathcal{T}$. If r is without noisy repetitions, then all leaves in $\mathcal{T}^{(r)}$ have even priority.*

Proof. By Definition 5.9, the leaves in $\mathcal{T}^{(r)}$ are all repetitions. We conclude by observing that, in an accepting run, non-noisy repetitions have necessarily even priority. In fact, let u be a repetition in a certain path p and assume that u has no or odd priority. Since by Lemma 5.12 there is a repetition in p of largest priority $\mathcal{F}(p)$, which is even, then u would be noisy. □

Lemma 5.14 (Pruned Run with Even Leaves). *Let $\mathcal{E}$ be a PES and let φ be a closed formula of $\mathcal{L}_{hp}$. If $L(\mathcal{A}_{\mathcal{E},\varphi}) \neq \emptyset$, then $\mathcal{A}_{\mathcal{E},\varphi}$ has an accepting run r on a k-tree $\mathcal{T}$ such that in $\mathcal{T}^{(r)}$ all leaves have even priority.*

Proof. Immediate corollary of Lemmata 5.13 and C.3. □

Lemma 5.16 (Accepting Run for Pruned Runs). *Let $\mathcal{E}$ be a PES and let φ be a closed formula of $\mathcal{L}_{hp}$. If there exists a run r of the NPA $\mathcal{A}_{\mathcal{E},\varphi}$ on a k-tree $\mathcal{T}$ such that in $\mathcal{T}^{(r)}$ all leaves have even priority, then there exists also an accepting run of $\mathcal{A}_{\mathcal{E},\varphi}$.*

Proof. Let r be a run such that all leaves in $\mathcal{T}^{(r)}$ have even priority. Consider the directed graph obtained by adding to every leaf w of $\mathcal{T}^{(r)}$ a "back arc" in the following way: If $r(w) = (C, \eta, \psi)$ where ψ is T or $[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\, \psi'$, then add a self-loop. Otherwise, ψ must be a proposition $Z(\mathbf{z})$ quantified in a ν-subformula, since all leaves have even priority. In this case the arc goes from l to the successor of its repetition witness $\Omega(l)$.

Unfold such a directed graph, starting from the root ϵ of $\mathcal{T}^{(r)}$, thus obtaining a k-tree $\mathcal{T}'$ (different from $\mathcal{T}$, in general). More formally, $\mathcal{T}'$ is the tree whose nodes are finite paths $v = (u_0, u_1, \ldots, u_n)$ in the graph and where each node $v = (u_0, u_1, \ldots, u_n)$ has $v' = (u_0, u_1, \ldots, u_{n-1})$ as parent. Clearly, all complete paths in $\mathcal{T}'$ are infinite, since every node of the graph has a successor. Moreover, paths in $\mathcal{T}'$ will be sometimes confused with the corresponding paths in the graph.

We show that r can be used to build an accepting run r' of the quotient $\mathcal{A}_{\mathcal{E},\varphi/\approx_f}$ over $\mathcal{T}'$. For each node $v = (u_0, u_1, \ldots, u_n)$, define $r'(v) = [r(u_n)]_{\approx_f}$. Since for each node u of the original run, $r(u) \approx_f r(\Omega(u))$, it is not difficult to realise that r' is a well-defined run of $\mathcal{A}_{\mathcal{E},\varphi/\approx_f}$ on $\mathcal{T}'$.

We argue that the run r' is accepting, that is, for any complete path $p = (v_0, v_1, \ldots)$ in r', if we consider the set of priorities that repeat infinitely often

$$I(p) = \{l \in [0, h] \mid \{i \mid r'(v_i) \in F_l\} \text{ is infinite}\},$$

then $I(P)$ is non-empty and its maximum is even.

Since p is infinite, $\mathcal{T}^{(r)}$ is finite (by Lemma 5.10), and the back arcs have been added only to the leaves, by construction some leaf w of $\mathcal{T}^{(r)}$ must repeat infinitely many times in p (or, more precisely, $v_i = v_{i-1}w$ infinitely often). Since w has even priority, there are two possibilities:

- $r(w)$ is T or $[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\, \psi'$ with $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C, \eta) = \emptyset$, or
- $r(w)$ is a proposition quantified in a ν-subformula.

In the first case, from some point on, $v_i = v_{i-1}w$ and $r(w)$, hence $r'(v_i)$, have priority 0 and this is the only priority repeating infinitely often. Hence, we are done.

In the second case, we know that $I(p) \neq \emptyset$. To conclude that the maximum is even, we proceed as follows: We show that for any proposition X quantified in a μ-subformula such that X appears infinitely many times along p, there is a proposition Y quantified in a ν-subformula such that $\mathrm{ad}(Y) > \mathrm{ad}(X)$ and Y appears infinitely many times along p. Recalling how the priority is defined based on the alternation depth, it is easy to see that this implies that for any node of odd priority there is a descendant node of larger even priority, whence the desired property.

Thus, assume that the formula $X(\mathbf{x})$, with X quantified in a μ-subformula, occurs infinitely often in p. Since $\mathcal{T}^{(r)}$ is finite, this implies that there is a node u of $\mathcal{T}^{(r)}$ such that the formula of $r(u)$ is $X(\mathbf{x})$ and u is traversed infinitely often in p. Moreover, since all leaves of $\mathcal{T}^{(r)}$ have even priority, u must be an inner node.

Now, let v_i, v_j be two consecutive occurrences of u in p, i.e., $v_i = v_{i-1}u$ and $v_j = v_{j-1}u$ with $i < j$. Note that, since u is an inner node of $\mathcal{T}^{(r)}$ and there is a path from v_i to v_j, i.e., from u to u in the graph, there must be an index k with $i < k < j$ such that u_k corresponds to a leaf w of $\mathcal{T}^{(r)}$, and its repetition witness $\Omega(w)$ is above u in $\mathcal{T}^{(r)}$. Assume that k is the smallest such index. Then w is a descendant of u in $\mathcal{T}^{(r)}$ (otherwise there should be an "earlier" leaf of index $u_{k'}$, with $i < k' < k$ whose witness is a common ancestor of u and w, contradicting the minimality of k). Therefore,

u is between $\Omega(w)$ and w in $\mathcal{T}^{(r)}$. Let $Y(\mathbf{y})$ be the formula in $r(w)$. Note that Y is quantified in a ν-subformula, since v is a leaf. By definition of repetition witness, Y is not introduced $\Omega(w)$ and w. Thus, by Lemma C.1,(1) we have that $X \sqsubseteq_d^* Y$. Moreover $X \neq Y$, since X is quantified in a μ-subformula and Y in a ν-subformula. Hence, by definition of alternation depth, $\mathrm{ad}(X) < \mathrm{ad}(Y)$.

We proved that between any two consecutive occurrences of u there is a node with a formula $Y(\mathbf{y})$, quantified in a ν-subformula, such that $\mathrm{ad}(X) < \mathrm{ad}(Y)$. Since u occurs infinitely often in p, the variable Y will be the same in infinitely many cases. This is what we aimed at.

Summing up, every complete path of r' is accepting and thus the run r' on the k-tree $\mathcal{T}'$ is an accepting run for $\mathcal{A}_{\mathcal{E},\varphi_{/\approx_f}}$ on $\mathcal{T}'$, thus $L(\mathcal{A}_{\mathcal{E},\varphi_{/\approx_f}}) \neq \emptyset$. Since $\approx_f$ is a bisimulation on $\mathcal{A}_{\mathcal{E},\varphi}$, by Theorem 5.5, $L(\mathcal{A}_{\mathcal{E},\varphi_{/\approx_f}}) = L(\mathcal{A}_{\mathcal{E},\varphi})$, hence, we conclude. □

Lemma 5.15 (Non-Emptiness Implies Satisfaction). *Let $\mathcal{E}$ be a strongly regular PES and let $\breve{\varphi}$ be a closed formula. If $L(\mathcal{A}_{\mathcal{E},\breve{\varphi}}) \neq \emptyset$, then $\mathcal{E} \models \breve{\varphi}$.*

Proof. Let $\mathcal{E}$ be a strongly regular PES, let $\breve{\varphi}$ in $\mathcal{L}_{hp}$ be a closed formula, and let $\mathcal{A}_{\mathcal{E},\breve{\varphi}}$ be the corresponding automaton. Assume that $L(\mathcal{A}_{\mathcal{E},\breve{\varphi}}) \neq \emptyset$. By Lemma 5.14, for $\mathcal{A}_{\mathcal{E},\breve{\varphi}}$ there exists an accepting run r on a k-tree $\mathcal{T}$, such that all leaves of $\mathcal{T}^{(r)}$ have even priority. We show that $\mathcal{T}^{(r)}$ can be easily transformed into a successful tableau for the sequent $\emptyset, \breve{\eta}, \emptyset \models \breve{\varphi}$.

First, all nodes of $\mathcal{T}^{(r)}$ can be labelled with sequents as follows: Recall that r maps each node u to a state $r(u) = (C, \eta, \psi)$. This is transformed into a sequent $C, \eta, \Delta_u \models \psi$, with the definition list Δ_u defined inductively as follows: For the root ϵ, we let $\Delta_u = \emptyset$. For non-root nodes ui, if $r(u) = (C, \eta, \psi)$ with $\psi = (\alpha X(\mathbf{x}).\psi')(\mathbf{y})$, i.e., $r(u)$ introduces X, then $\Delta_{ui} = \Delta_u[X(\mathbf{x}) \mapsto \alpha X(\mathbf{x}).\psi']$. In all other cases, $\Delta_{ui} = \Delta_u$.

Since the leaves l of $\mathcal{T}^{(r)}$ have even priority, the corresponding formulae are either:

(1) T or $[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\psi'$ with $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C, \eta) = \emptyset$ or
(2) $X(\mathbf{x})$ with X bound in a ν-subformula.

In the first case, by definition of the transition relation and of pruned k-tree, the predecessor u of the leaf l is such that $r(u) = r(l)$ and $l = u1$ is its only successor. Call τ the tree obtained by removing such leaves.

Inspecting the automaton transitions and the tableau rules, it is immediate to realise that the sequents labelling each internal node of τ and its successors are the premise and the conclusions, respectively, of a tableau rule. The stop condition γ (see Definition 4.4) is not satisfied by internal nodes, since they are not repetitions. Moreover, no tableau rule applies to the sequents labelling the leaves of τ. This is clearly the case for leaves in item (1) above. For the leaves in item (2), the rule (Unf_ν) cannot be applied, since the stop condition γ is guaranteed to hold by Definition 5.9.

Therefore, τ is a tableau, and it is clearly successful, since, as already observed, every leaf is as required in Definition 4.6. □

Lemma 5.17 (Satisfaction Implies Non-Emptiness). *Let $\mathcal{E}$ be a strongly regular PES and let $\breve{\varphi}$ be a closed formula. If $\mathcal{E} \models \breve{\varphi}$, then $L(\mathcal{A}_{\mathcal{E},\breve{\varphi}}) \neq \emptyset$.*

Proof. Let $q_0 = (\emptyset, \breve{\eta}, \breve{\varphi})$ be the initial state of $\mathcal{A}_{\mathcal{E},\breve{\varphi}}$. Let τ be a successful tableau for the formula $\breve{\varphi}$, which is guaranteed to exist by Lemma 4.16.

The tableau τ can be transformed into a run r of $\mathcal{A}_{\mathcal{E},\breve{\varphi}}$ in the following way: A prefix of r corresponds exactly to the tableau τ: each sequent $C, \eta, \Delta \models \varphi$ is transformed into a state (C, η, φ). In particular, the root of τ, labelled by the sequent $\emptyset, \breve{\eta}, \emptyset \models \breve{\varphi}$, corresponds to the initial state

$q_0 = (\emptyset, \check{\eta}, \check{\varphi})$. By an inspection of the tableau rules and the automaton transitions it is immediate to realise that this is indeed an incomplete run of $\mathcal{A}_{\mathcal{E},\check{\varphi}}$ starting from the initial state q_0.

Now, since by definition of $\mathcal{A}_{\mathcal{E},\check{\varphi}}$, every state has a successor, clearly r can be extended to a full correct run r of $\mathcal{A}_{\mathcal{E},\check{\varphi}}$ on some k-tree $\mathcal{T}$.

Note that r might not be accepting. Still, the prefix of r, corresponding to τ, does not contain repetitions except for those associated with the leaves of τ (in particular, the stop condition γ never holds in the inner nodes of τ). Since τ is successful (Definition 4.6) leaves w are labelled by proposition quantified in a ν-subformula, T, or $[\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\,\psi$ with $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C,\eta) = \emptyset$. In the last two cases, a path in r reaching w will cycle on indefinitely on the same formula. In the first case, a path in r through w could possibly include repetitions after w, but these will not be noisy, thanks to the presence of w. Hence, r has no noisy repetitions and thus, by Lemmata C.3 and 5.16, there exists a run r' of $\mathcal{A}_{\mathcal{E},\check{\varphi}}$, which is accepting. Therefore, $\mathcal{L}(\mathcal{A}_{\mathcal{E},\check{\varphi}}) \neq \emptyset$. □

Theorem 5.18 (Model Checking via Non-emptiness). *Let $\mathcal{E}$ be a strongly regular PES and let φ be a closed formula of $\mathcal{L}_{hp}$. Then $L(\mathcal{A}_{\mathcal{E},\varphi}) \neq \emptyset$ iff $\mathcal{E}$ satisfies φ.*

Proof. Corollary of Lemmata 5.15 and 5.17. □

C.3 Proofs for Section 6

Definition C.4 (Residual of the Unfolding). Let $\mathcal{N} = (P, T, F, M_0)$ be a safe Petri net and let $\mathcal{U}(\mathcal{N}) = (P^U, T^U, F^U, M_0^U)$ be its unfolding. Given a configuration $C \in \mathcal{C}(\mathcal{U}(\mathcal{N}))$, we define the *residual* of $\mathcal{U}(\mathcal{N})$ after C as $\mathcal{U}(\mathcal{N})[C] = (P_c^U, T_c^U, F_c^U, C^\circ)$, where $P_c^U = \{b' \in P^U \mid \exists\, b \in C^\circ.\ b \leq b'\}$, $T_c^U = \{e \in T^U \mid \exists\, b \in C^\circ.\ b \leq e\}$, and F_c^U is the restriction of F^U to $(P_c^U \times T_c^U) \cup (T_c^U \times P_c^U)$.

The residual $\mathcal{U}(\mathcal{N})[C]$ is isomorphic to the unfolding of $\mathcal{N}$ with $\mathsf{M}(C)$ as initial marking. Moreover, the PES underlying $\mathcal{U}(\mathcal{N})[C]$ is isomorphic to the residual $\mathcal{E}(\mathcal{N})[C]$.

Proposition 6.4 (Pointed Markings vs Residuals). *Let $\mathcal{N} = (P, T, F, M_0)$ be a safe Petri net. Given a set V and two V-pointed configurations $\langle C_1, \zeta_1\rangle$, $\langle C_2, \zeta_2\rangle$ in $\mathcal{E}(\mathcal{N})$, if $\mathsf{M}(\langle C_1, \zeta_1\rangle) = \mathsf{M}(\langle C_2, \zeta_2\rangle)$, then $\langle C_1, \zeta_1\rangle \approx_r \langle C_2, \zeta_2\rangle$.*

Proof. Let $\mathcal{N} = (P, T, F, M_0)$ be a safe Petri net, consider its unfolding $\mathcal{U}(\mathcal{N}) = (P^U, T^U, F^U, M_0^U)$ and two V-pointed configurations $\langle C_1, \zeta_1\rangle$, $\langle C_2, \zeta_2\rangle$ such that $\mathsf{M}(\langle C_1, \zeta_1\rangle) = \mathsf{M}(\langle C_2, \zeta_2\rangle)$.

First note that, since, in particular, $\mathsf{M}(C_1) = \mathsf{M}(C_2)$, the residuals $\mathcal{U}(\mathcal{N})[C_1]$ and $\mathcal{U}(\mathcal{N})[C_2]$ are isomorphic. Let $\iota : \mathcal{U}(\mathcal{N})[C_1] \to \mathcal{U}(\mathcal{N})[C_2]$ be the corresponding net isomorphism. Its restriction to the underlying PESs, abusing the notation, is still denoted by ι and it establishes an isomorphism of the residuals $\mathcal{E}(\mathcal{N})[C_1]$ and $\mathcal{E}(\mathcal{N})[C_2]$. We next prove that ι is also an isomorphism of residuals of the pointed configurations $\langle C_1, g_1\rangle$ and $\langle C_2, g_2\rangle$, i.e., that for all $x \in V$ and let $e_1 \in \mathcal{E}(\mathcal{N})[C_1]$, we have $g_1(x) \leq e_1$ iff $g_2(x) \leq \iota(e_1)$.

Let $x \in V$ and let $e_1 \in \mathcal{E}(\mathcal{N})[C_1]$. If $g_1(x) \leq e_1$, then there is $b_1 \in {C_1}^\circ$ such that $g_1(x) \leq b_1 \leq e_1$. Thus, $\pi_1(b_1) \in\in r_1(x) = r_2(x)$. Therefore, there is $b_2 \in {C_2}^\circ$ such that $\pi_1(b_2) = \pi_1(b_1)$ and $g_2(x) \leq b_2$.

Now, since the net is safe the isomorphism ι must map b_1 to b_2. Therefore, $g_2(x) \leq b_2 = \iota(b_1) \leq \iota(e_1)$, where the last step is motivated by the fact that the isomorphism preserves causality. Hence, $g_2(x) \leq \iota(e_1)$, as desired.

Conversely, if $g_2(x) \leq \iota(e_1)$, we can deduce that $g_1(x) \leq e_1$ in an analogous way. □

Proposition 6.7 (Pointed Marking Equivalence is a Bisimulation). *Let $\mathcal{N}$ be a finite safe Petri net and let φ be a closed formula of $\mathcal{L}_{hp}$. The equivalence $\approx_m$ on the automaton $\mathcal{A}_{\mathcal{E}(\mathcal{N}),\varphi}$ is a bisimulation and it is of finite index.*

Proof. Let $q = (C, \eta, \psi), q' = (C', \eta', \psi') \in Q$ be states of $\mathcal{A}_{\mathcal{E}(\mathcal{N}),\varphi}$ such that $q \approx_m q'$. This means that $\mathsf{M}(\langle C_1, \eta_1|_{fv(\psi)}\rangle) = \mathsf{M}(\langle C_2, \eta_2|_{fv(\psi)}\rangle) = \langle M, r\rangle$.

Concerning condition (1) of Definition 5.4, the fact that for all $i \in [0, h]$, $q \in F_i$ iff $q' \in F_i$ immediately follows from Lemma 5.8, recalling that $\approx_m$ refines future equivalence $\approx_f$.

Let us focus on condition (2). Assume $q \to (q_1, \ldots, q_n)$. To prove that $q' \to (q'_1, \ldots, q'_n)$ with $q_i \approx_m q'_i$ for $i \in [1, n]$, we distinguish various cases according to the shape of ψ.

- $\psi = \mathsf{T}$ or $\psi = \mathsf{F}$
  Trivial, since the only transitions of q and q' are $q \to (q)$ and $q' \to (q')$.
- $\psi = \psi_1 \wedge \psi_2$
  The only transition of q is $q \to (q_1, q_2)$ where $q_i = (C, \eta, \psi_i)$, $i \in \{1, 2\}$. By construction also $q' \to (q'_1, q'_2)$ where $q'_i = (C', \eta', \psi_i)$, $i \in \{1, 2\}$. To conclude that $q_i \approx_m q_1 i$, for $i \in \{1, 2\}$, just observe that $fv(\psi_i) \subseteq fv(\psi)$ and thus such states are associated with the same pointed marking $\langle M, r|_{fv(\psi_i)}\rangle$.
- $\psi = \psi_1 \vee \psi_2$, $\psi = (\alpha X(\mathbf{x}).\psi')(\mathbf{y})$, or $\psi = X(\mathbf{y})$
  These cases are analogous to the previous one.
- $\psi = \langle\!\langle \mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z\rangle\!\rangle \psi_1$
  in this case, if $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C, \eta) \neq \emptyset$, then $q \to (q_1)$ where $q_1 = (C_1, \eta_1, \psi_1)$ with $(C_1, \eta_1) \in \mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C, \eta)$, which means that $C \xrightarrow{\eta(\mathbf{x}),\overline{\eta(\mathbf{y})}<e}_a C_1$ and $\eta_1 = \eta[z \mapsto e]$.
  Let $t = \pi_1(e)$ be the transition in $\mathcal{N}$ corresponding to the event e of $\mathcal{U}(\mathcal{N})$. For each x in $\mathbf{x}$, since $\eta(x) < e$, there must be a condition $b_x \in {}^\bullet e$ such that $\eta(x) < b_x$ and thus, in the associated pointed marking $\langle M, r\rangle$, we have ${}^\bullet t \cap r(x) \neq \emptyset$. Dually, for each y in $\mathbf{y}$, since it is not the case that $\eta(y) < e$, for all conditions $b \in {}^\bullet e$, we have $\neg(\eta(y) < b)$ and thus ${}^\bullet t \cap r(y) = \emptyset$.
  Since, by hypothesis, q' is associated with the same pointed marking $\langle M, r\rangle$ as q, there is an event e' such that $\pi_1(e') = t$ and $e' \in en(C')$. Moreover, for all variables $w \in fv(\psi)$, we have $\eta'(w) < e'$ iff $\eta'(w) \cap {}^\bullet e' \neq \emptyset$ iff $r(w) \cap {}^\bullet t \neq \emptyset$. Therefore, by the considerations above, for each x in $\mathbf{x}$, $\eta'(x) < e'$ and for no y in $\mathbf{y}$, $\eta'(y) < e'$. This means that if we let $\eta'_1 = \eta'[z \mapsto e']$, then there exists $(C'_1, \eta'_1) \in \mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C', \eta')$ and thus $q' \to (q'_1)$ where $q'_1 = (C'_1, \eta'_1, \psi_1)$. Finally, $q_1 \approx_m q'_1$, i.e., they are associated with the same $fv(\psi_1)$-pointed marking $\langle M', r'\rangle$ where $M' = (M \setminus {}^\bullet t) \cup t^\bullet$. Moreover, $fv(\psi_1) \subseteq fv(\psi) \cup \{z\}$ and for each variable $w \in fv(\psi_1) \setminus \{z\}$, we have $r'(w) = (r(w) \cap M') \cup \{s \mid s \in t^\bullet \wedge\ r(w) \cap {}^\bullet t \neq \emptyset\}$ and $r'(z) = t^\bullet$.
  If instead $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C, \eta) = \emptyset$, then $q \to (q)$. From the fact that q and q' are associated with the same pointed marking, reasoning as in the previous case, we deduce that also $\mathrm{Succ}_{\mathcal{E}}^{\mathbf{x},\overline{\mathbf{y}}<\mathrm{a}z}(C', \eta') = \emptyset$ and thus $q' \to (q')$. Hence, we conclude
- $\psi = [\![\mathbf{x}, \overline{\mathbf{y}} < \mathrm{a}\, z]\!]\psi_1$
  Analogous to the previous case.

The fact that $\approx_m$ is of finite index immediately follows from the observation that the number of $fv(\psi)$-pointed markings of $\mathcal{N}$, where $\psi \in \mathrm{sf}(\varphi)$, is finite. In fact both $\mathrm{sf}(\varphi)$ and $\mathcal{R}(\mathcal{N})$ are finite. Moreover, for each fixed subformula $\psi \in \mathrm{sf}(\varphi)$ and marking $M \in \mathcal{R}(\mathcal{N})$, the number of $fv(\psi)$-pointed markings $\langle M, r\rangle$, with $r : fv(\psi) \to 2^M$ is clearly finite. □

ACKNOWLEDGMENTS

We are grateful to Perdita Stevens for insightful hints and pointers to the literature and to the anonymous reviewers for their comments on the conference papers.

REFERENCES

[1] P. A. Abdulla, L. Kaati, and J. Högberg. 2006. Bisimulation minimization of tree automata. In *Proceedings of the CIAA'06*, O. H. Ibarra and H.-C. Yen (Eds.), Vol. 4094. Springer, 173–185.

[2] R. Alur, D. A. Peled, and W. Penczek. 1995. Model-checking of causality properties. In *Proceedings of the LICS'95*. IEEE Computer Society, 90–100.

[3] P. Baldan and A. Carraro. 2015. A causal view on non-intereference. *Fundam. Inform.* 140, 1 (2015), 1–38.

[4] P. Baldan and S. Crafa. 2014. A logic for true concurrency. *J. ACM* 61, 4 (2014), 24:1–24:36.

[5] Paolo Baldan and Tommaso Padoan. 2017. Local model checking in a logic for true concurrency. In *Proceedings of the FoSSaCS'17* (LNCS, Vol. 10203), Javier Esparza and Andrzej S. Murawski (Eds.). Springer, 407–423.

[6] P. Baldan and T. Padoan. 2018. Automata for true concurrency properties. In *Proceedings of the FoSSaCS'18*, C. Baier and U. Del Lago (Eds.), Vol. 10803. Springer, 165–182.

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

[8] E. Best, R. Devillers, A. Kiehn, and L. Pomello. 1991. Fully concurrent bisimulation. *Acta Inform.* 28 (1991), 231–261.

[9] J. Bradfield and S. Fröschle. 2002. Independence-friendly modal logic and true concurrency. *Nord. J. Comput.* 9, 1 (2002), 102–117.

[10] J. Chalopin and V. Chepoi. 2017. A counterexample to Thiagarajan's conjecture on regular event structures. In *Proceedings of the ICALP'17* (LIPIcs, Vol. 80), I. Chatzigiannakis, P. Indyk, F. Kuhn, and A. Muscholl (Eds.). Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 101:1–101:14.

[11] E. M. Clarke and B.-H. Schlingloff. 2001. Model checking. In *Handbook of Automated Reasoning*, A. Robinson and A. Voronkov (Eds.). Elsevier.

[12] R. Cleaveland. 1990. Tableau-based model checking in the propositional mu-calculus. *Acta Inform.* 27, 8 (1990), 725–747.

[13] R. De Nicola and G. Ferrari. 1990. Observational logics and concurrency models. In *Proceedings of the FST-TCS'90* (LNCS, Vol. 472), K. V. Nori and C. E. V. Madhavan (Eds.). Springer, 301–315.

[14] P. Degano, R. De Nicola, and U. Montanari. 1988. Partial orderings descriptions and observations of nondeterministic concurrent processes. In *Proceedings of the REX Workshop (LNCS, Vol. 354)*, Jaco W. de Bakker, Willem P. de Roever, and Grzegorz Rozenberg (Eds.). Springer, 438–466.

[15] M. Dumas and L. García-Bañuelos. 2015. Process mining reloaded: Event structures as a unified representation of process models and event logs. In *Proceedings of the Petri Nets'15 (LNCS, Vol. 9115)*, R. R. Devillers and A. Valmari (Eds.). Springer, 33–48.

[16] E. A. Emerson, C. S. Jutla, and A. P. Sistla. 2001. On model checking for the μ-calculus and its fragments. *Theoret. Comput. Sci.* 258, 1–2 (2001), 491–522.

[17] J. Esparza and K. Heljanko. 2008. *Unfoldings—A Partial Order Approach to Model Checking*. Springer.

[18] A. Farzan and P. Madhusudan. 2006. Causal atomicity. In *Proceedings of the CAV'06 (LNCS, Vol. 4144)*, T. Ball and R. B. Jones (Eds.). 315–328.

[19] J. Gutierrez. 2009. Logics and bisimulation games for concurrency, causality and conflict. In *Proceedings of the FoSSaCS'09 (LNCS, Vol. 5504)*, L. de Alfaro (Ed.). Springer, 48–62.

[20] J. Gutierrez. 2011. *On Bisimulation and Model-checking for Concurrent Systems with Partial Order Semantics*. Ph.D. Dissertation. University of Edinburgh.

[21] J. Gutierrez and J. C. Bradfield. 2009. Model-checking games for fixpoint logics with partial order models. In *Proceedings of the CONCUR'09 (LNCS, Vol. 5710)*, M. Bravetti and G. Zavattaro (Eds.). Springer, 354–368.

[22] D. Janin and I. Walukiewicz. 1996. On the expressive completeness of the propositional mu-calculus with respect to monadic second order logic. In *Proceedings of the CONCUR'96*, U. Montanari and V. Sassone (Eds.). Springer, 263–277.

[23] L. Jategaonkar and A. R. Meyer. 1996. Deciding true concurrency equivalences on safe, finite nets. *Theoret. Comput. Sci.* 154, 1 (1996), 107–143.

[24] A. Jeffrey and J. Riely. 2016. On thin air reads towards an event structures model of relaxed memory. In *Proceedings of the LICS'16*, M. Grohe, E. Koskinen, and N. Shankar (Eds.). ACM, 759–767.

[25] M. Jurdzinski, M. Nielsen, and J. Srba. 2003. Undecidability of domino games and hhp-bisimilarity. *Inform. Comput.* 184, 2 (2003), 343–368.

[26] H. Klauck. 2002. Algorithms for parity games. In *Automata, Logics, and Infinite Games: A Guide to Current Research*, Erich Grädel, Wolfgang Thomas, and Thomas Wilke (Eds.). Vol. 2500. Springer, 107–129.

[27] K. Lodaya and P. S. Thiagarajan. 1987. A modal logic for a subclass of event structures. In *Proceedings of the ICALP'87*. Springer-Verlag, 290–303.

[28] P. Madhusudan. 2003. Model-checking trace event structures. In *Proceedings of the LICS'13*. IEEE Computer Society, 371–380.

[29] U. Montanari and M. Pistore. 1997. Minimal transition systems for history-preserving bisimulation. In *Proceedings of the STACS'97 (LNCS, Vol. 1200)*, R. Reischuk and M. Morvan (Eds.). Springer, 413–425.
[30] A. W. Mostowski. 1985. Regular expressions for infinite trees and a standard form of automata. In *Proceedings of Computation Theory: Fifth Symposium 1984 (LNCS, Vol. 208)*, A. Skowron (Ed.). Springer, 157–168.
[31] M. Mukund and P. S. Thiagarajan. 1989. An axiomatization of event structures. In *Proceedings of the FST-TCS'89*, C. E. Veni Madhavan (Ed.). Springer Berlin, 143–160.
[32] M. Mukund and P. S. Thiagarajan. 1992. A logical characterization of well branching event structures. *Theor. Comput. Sci.* 96, 1 (1992), 35–72.
[33] M. Nielsen and C. Clausen. 1995. Games and logics for a noninterleaving bisimulation. *Nord. J. Comput.* 2, 2 (1995), 221–249.
[34] T. Padoan. 2018. True Concurrency Workbench. Retrieved from http://github.com/tpadoan/TCWB.
[35] W. Penczek. 1990. A temporal logic for event structures. In *Mathematical Logic*, P. P. Petkov (Ed.). Springer, 327–338. DOI: 10.1007/978-1-4613-0609-2_23
[36] W. Penczek. 1995. Branching time and partial order in temporal logics. In *Time and Logic: A Computational Approach*. UCL Press, 179–228.
[37] W. Penczek. 1997. Model-checking for a subclass of event structures. In *Proceedings of the TACAS'97 (LNCS, Vol. 1217)*, E. Brinksma (Ed.). Springer, 145–164.
[38] C. A. Petri. 1962. *Kommunikation mit Automaten*. Technischen Hoschule Darmstadt.
[39] I. Phillips and I. Ulidowski. 2014. Event identifier logic. *Math. Struct. Comput. Sci.* 24, 2 (2014), 1–51. DOI: https://doi.org/10.1017/S0960129513000510
[40] J. Pichon-Pharabod and P. Sewell. 2016. A concurrency semantics for relaxed atomics that permits optimisation and avoids thin-air executions. In *Proceedings of the POPL'16*, R. Bodík and R. Majumdar (Eds.). ACM, 622–633.
[41] S. Pinchinat, F. Laroussinie, and Ph. Schnoebelen. 1994. *Logical Characterization of Truly Concurrent Bisimulation*. Technical Report 114. LIFIA-IMAG, Grenoble.
[42] C. Prisacariu. 2014. Higher dimensional modal logic. *CoRR* abs/1405.4100 (2014), 43.
[43] Alexander M. Rabinovich and Boris A. Trakhtenbrot. 1988. Behaviour structures and nets. *Fundam. Inform.* 11 (1988), 357–404.
[44] P. Stevens and C. Stirling. 1998. Practical model-checking using games. In *Proceedings of the TACAS'98*, B. Steffen (Ed.). Springer, 85–101.
[45] C. Stirling and D. Walker. 1991. Local model checking in the modal mu-calculus. *Theor. Comput. Sci.* 89, 1 (1991), 161–177.
[46] P. S. Thiagarajan. 2002. Regular event structures and finite Petri nets: A conjecture. In *Formal and Natural Computing—Essays Dedicated to Grzegorz Rozenberg (on occasion of his 60th birthday) (LNCS, Vol. 2300)*, W. Brauer, H. Ehrig, J. Karhumäki, and A. Salomaa (Eds.). Springer, 244–256.
[47] R. J. van Glabbeek and U. Goltz. 2001. Refinement of actions and equivalence notions for concurrent systems. *Acta Inform.* 37, 4/5 (2001), 229–327.
[48] W. Vogler. 1991. Deciding history preserving bisimilarity. In *Proceedings of the ICALP'91 (LNCS, Vol. 510)*, J. Albert, B. Monien, and M. Rodríguez-Artalejo (Eds.). Springer, 495–505.
[49] G. Winskel. 1987. Event structures. In *Petri Nets: Applications and Relationships to Other Models of Concurrency (LNCS, Vol. 255)*, W. Brauer, W. Reisig, and G. Rozenberg (Eds.). Springer, 325–392.
[50] G. Winskel. 2011. Events, causality and symmetry. *Comput. J.* 54, 1 (2011), 42–57.

Received September 2018; revised November 2019; accepted July 2020