Strategy Logic with Imperfect Information

Raphaël Berthon*, Bastien Maubert†, Aniello Murano†, Sasha Rubin† and Moshe Y. Vardi‡
*École Normale Supérieure de Rennes, Rennes, France
†Università degli Studi di Napoli Federico II, Naples, Italy
‡Rice University, Houston, Texas, USA

***Abstract*—We introduce an extension of Strategy logic for the imperfect-information setting, called $\mathsf{SL}_{\mathrm{ii}}$, and study its model-checking problem. As this logic naturally captures multi-player games with imperfect information, the problem turns out to be undecidable. We introduce a syntactical class of "hierarchical instances" for which, intuitively, as one goes down the syntactic tree of the formula, strategy quantifications are concerned with finer observations of the model. We prove that model-checking $\mathsf{SL}_{\mathrm{ii}}$ restricted to hierarchical instances is decidable. This result, because it allows for complex patterns of existential and universal quantification on strategies, greatly generalises previous ones, such as decidability of multi-player games with imperfect information and hierarchical observations, and decidability of distributed synthesis for hierarchical systems.**

To establish the decidability result, we introduce and study $\mathsf{QCTL}^*_{\mathrm{ii}}$, an extension of QCTL (itself an extension of CTL with second-order quantification over atomic propositions) by parameterising its quantifiers with observations. The simple syntax of $\mathsf{QCTL}^*_{\mathrm{ii}}$ allows us to provide a conceptually neat reduction of $\mathsf{SL}_{\mathrm{ii}}$ to $\mathsf{QCTL}^*_{\mathrm{ii}}$ that separates concerns, allowing one to forget about strategies and players and focus solely on second-order quantification. While the model-checking problem of $\mathsf{QCTL}^*_{\mathrm{ii}}$ is, in general, undecidable, we identify a syntactic fragment of hierarchical formulas and prove, using an automata-theoretic approach, that it is decidable. The decidability result for $\mathsf{SL}_{\mathrm{ii}}$ follows since the reduction maps hierarchical instances of $\mathsf{SL}_{\mathrm{ii}}$ to hierarchical formulas of $\mathsf{QCTL}^*_{\mathrm{ii}}$.

I. Introduction

Logics for strategic reasoning are a powerful tool for expressing correctness properties of multi-player graph-games, which in turn are natural models for reactive systems and discrete event systems. In particular, ATL^* and its related logics were introduced to capture the realisability/synthesis problem for open systems with multiple components. These logics were designed as extensions of branching-time logics such as CTL^* that allow one to write alternating properties directly in the syntax, i.e., statements of the form "there exist strategies, one for each player in A, such for all strategies of the remaining players, the resulting play satisfies φ". Strategy logic (SL) [1] generalises these by treating strategies as first-order objects x that can be quantified $\langle\!\langle x \rangle\!\rangle$ (read "there exists a strategy x") and bound to players (a, x) (read "player a uses strategy x"). This syntax has flexibility very similar to first-order logic, and thus allows one to directly express many solution concepts from game-theory, e.g., the SL formula $\langle\!\langle x_1 \rangle\!\rangle \langle\!\langle x_2 \rangle\!\rangle (a_1, x_1)(a_2, x_2) \wedge_{i=1,2} [\langle\!\langle y_i \rangle\!\rangle (a_i, y_i) goal_i] \rightarrow goal_i$ expresses the existence of a Nash equilibrium in a two-player game (with individual Boolean objectives).

An essential property of realistic multi-player games is that players only have a limited view of the state of the system. This is captured by introducing partial-observability into the models, i.e., equivalence-relations o (called *observations*) over the state space that specify indistinguishable states. In the formal-methods literature it is typical to associate observations to players. In this paper, instead, we associate observations to strategies. Concretely, we introduce an extension $\mathsf{SL}_{\mathrm{ii}}$ of SL that annotates the strategy quantifier $\langle\!\langle x \rangle\!\rangle$ by an observation o, written $\langle\!\langle x \rangle\!\rangle^o$. Thus, both the model and the formulas mention observations o. This novelty allows one to express, in the logic, that a player's observation changes over time.

Our logic $\mathsf{SL}_{\mathrm{ii}}$ is extremely powerful: it is an extension of SL (which is in the perfect-information setting), and of the imperfect-information strategic logics $\mathsf{ATL}^*_{\mathrm{i,R}}$ [2] and $\mathsf{ATL}^*_{\mathrm{sc,i}}$ [3]. A canonical specification in multi-player game of partial observation is that the players, say $a_1, \ldots, a_n$, can form a coalition and beat the environment player, say b. This can be expressed in $\mathsf{SL}_{\mathrm{ii}}$ as $\Phi_{\mathrm{SYNTH}} := \langle\!\langle x_1 \rangle\!\rangle^{o_1} \ldots \langle\!\langle x_n \rangle\!\rangle^{o_n} [\![y]\!]^o (a_1, x_1) \ldots (a_n, x_n)(b, y)\varphi$, where φ is quantifier- and binding-free. Also, $\mathsf{SL}_{\mathrm{ii}}$ can express more complicated specifications by alternating quantifiers, binding the same strategy to different agents and rebinding (these are inherited from SL), as well as changing observations.

The complexity of $\mathsf{SL}_{\mathrm{ii}}$ is also visible from an algorithmic point of view. Its satisfiability problem is undecidable (this is already true of SL), and its model-checking problem is undecidable (this is already true of $\mathsf{ATL}^*_{\mathrm{i,R}}$, in fact, even for the single formula $\langle\{a, b\}\rangle \mathbf{F} p$ in systems with three players [4]). In fact, similar undecidability occurs in foundational work in multi-player games of partial observation, and in distributed synthesis [5], [6]. Since then, the formal-methods community has spent much effort finding restrictions and variations that ensure decidability [7]–[13]. The common thread in these approaches is that the players' observations (or what they can deduce from their observations) are hierarchical.

Motivated by the problem of finding decidable extensions of strategy logic in the imperfect-information setting, we introduce a syntactic class of "hierarchical instances" of $\mathsf{SL}_{\mathrm{ii}}$, i.e., formula/model pairs, and prove that the model-checking problem on this class of instances is decidable. Intuitively, an instance of $\mathsf{SL}_{\mathrm{ii}}$ is hierarchical if, as one goes down the syntactic tree of the formula, the observations annotating strategy quantifications can only become finer. Although the class of hierarchical instances refers not only to the syntax of the logic but also to the model, the class is syntactical in the

sense that it depends only on the structure of the formula and the observations in the model. Moreover, it is easy to check (i.e., in linear time) if an instance is hierarchical or not.

The class of hierarchical instances generalises some existing approaches and supplies new classes of systems and properties that can be model-checked. For instance, suppose that there is a total order $\preceq$ among the players such that $a \preceq b$ implies player b's observation is finer than player a's observation — such games are said to yield "hierarchical observation" in [13]. In such games it is known that synthesis against CTL^* specifications is decidable (in fact, this holds for ω-regular specifications [8], [13]). This corresponds to hierarchical instances of SL_{ii} in which the observations form a total order in the model and the formula is of the form Φ_{SYNTH} (mentioned above). On the other hand, in hierarchical instances of SL_{ii}, the ordering on observations can be a pre partial-order (i.e., not just total), and one can arbitrarily alternate quantifiers in the formulas. For instance, hierarchical instances allow one to decide if a game that yields hierarchical information has a Nash equilibrium. For example, assuming observations p_1, p_2, o_1, o_2 with p_i finer than o_2 (for $i = 1, 2$), and o_2 finer than o_1, the formula $\langle\langle x_1 \rangle\rangle^{o_1} \langle\langle x_2 \rangle\rangle^{o_2} (a_1, x_1)(a_2, x_2) \wedge_{i=1,2} [\langle\langle y_i \rangle\rangle^{p_i} (a_i, y_i) goal_i] \rightarrow goal_i$ expresses that there exists a strategy profile (x_1, x_2) of uniform strategies, such that neither player has an incentive to deviate using a strategy that observes at least as much as the observations that both players started with. Observe that this formula is in fact *equivalent* to the existence of a Nash equilibrium, i.e., to the same formula in which $p_i = o_i$. This shows we can decide the existence of Nash equilibria in games that yield hierarchical observation.

In order to reason about SL_{ii} we introduce an intermediate logic $\mathsf{QCTL}^*_{\text{ii}}$, an extension to the imperfect-information setting of QCTL^* [14], itself an extension of CTL^* by second-order quantifiers over atoms. This is a low-level logic that does not mention strategies and into which one can effectively compile instances of SL_{ii}. States of the models of the logic $\mathsf{QCTL}^*_{\text{ii}}$ have internal structure, much like the multi-player game structures from [15] and distributed systems [16]. Model-checking $\mathsf{QCTL}^*_{\text{ii}}$ is also undecidable (indeed, we show how to reduce from the MSO-theory of the binary tree extended with the equal-length predicate, known to be undecidable [17]). We introduce the syntactical class $\mathsf{QCTL}^*_{\text{i},\subseteq}$ of hierarchical formulas as those in which innermost quantifiers observe more than outermost quantifiers, and prove that model-checking is decidable using an extension of the automata-theoretic approach for branching-time logics (our decision to base models of $\mathsf{QCTL}^*_{\text{ii}}$ on local states greatly eases the use of automata). Moreover, the compilation of SL_{ii} into $\mathsf{QCTL}^*_{\text{ii}}$ preserves being hierarchical, thus establishing our main contribution, i.e., that model checking the hierarchical instances of SL_{ii} is decidable.

Related work. Formal methods for reasoning about reactive systems with multiple components have been studied mainly in two theoretical frameworks: a) multi-player graph-games of partial-observation [6], [8], [13] and b) synthesis in distributed architectures [5], [7], [9], [11], [18] (the relationship between these two frameworks is discussed in [13]). All of these works consider the problem of synthesis, which (for objectives in temporal logics) can be expressed in SL_{ii} using the formula Φ_{SYNTH} mentioned above. Limited alternation was studied in [12] that, in the language of SL_{ii}, considers the model-checking problem of formulas of the form $\langle\langle x_1 \rangle\rangle^{o_1} [\![x_2]\!]^{o_2} \langle\langle x_3 \rangle\rangle^{o_3} (a_1, x_1)(a_2, x_2)(a_3, x_3)\varphi$, where φ is an ω-regular objective. They prove that this is decidable in case player player 3 has perfect observation and player 2 observes at least as much as player 1.

In contrast to all these works, formulas of SL_{ii} can express much more complex specifications by alternating quantifiers, sharing strategies, rebinding, and changing observations.

Recently, [13] generalised the classic result of [8]: it weakens the assumption of hierarchical observation to hierarchical information (which are both static notions), and then, further to dynamic hierarchical information which allows for the hierarchy amongst players' information to change along a play. However, they only consider the synthesis problem.

We are aware of two papers that (like we do) give simultaneous structural constraints on both the formula and the model that result in decidability: in the context of synthesis in distributed architecture with process delays, [18] considers CTL^* specifications that constrain external variables by the input variables that may effect them in the architecture; and in the context of asynchronous perfect-recall, [10] considers a syntactical restriction on instances for Quantified μ-Calculus with partial observation (in contrast, we consider the case of synchronous perfect recall).

The work closest to ours is [19] which introduces a decidable logic CL in which one can encode many distributed synthesis problems. However, CL is close in spirit to our $\mathsf{QCTL}^*_{\text{i},\subseteq}$, and is more appropriate as a tool than as a high-level specification logic like SL_{ii}. Furthermore, by means of a natural translation we derive that CL is strictly included in the hierarchical instances of SL_{ii} (Section II-E). In particular, we find that hierarchical instances of SL_{ii} can express non-observable goals, while CL does not. Non-observable goals arise naturally in problems in distributed synthesis [5].

Finally, our logic SL_{ii} is the first generalisation of (the syntax and semantics of) SL to include strategies with partial observation, and, unlike CL, generalises previous logics with partial-observation strategies, i.e., $\mathsf{ATL}^*_{\text{i,R}}$ [2] and $\mathsf{ATL}^*_{\text{sc,i}}$ [3]. A comparison of SL_{ii} to SL, CL, and $\mathsf{ATL}^*_{\text{sc,i}}$ is given in Section II-E.

Plan. The definition of SL_{ii} and of hierarchical instances, and the discussion about Nash equilibria, are in Section II. The definition of $\mathsf{QCTL}^*_{\text{ii}}$, and its hierarchical fragment $\mathsf{QCTL}^*_{\text{i},\subseteq}$, are in Section III. The proof that model checking $\mathsf{QCTL}^*_{\text{i},\subseteq}$ is decidable, including the required automata preliminaries, are in Section IV. The translation of SL_{ii} into $\mathsf{QCTL}^*_{\text{ii}}$, and the fact that this preserves hierarchy, are in Section V. Missing details are in the Appendix.

II. SL WITH IMPERFECT INFORMATION

In this section we introduce SL_{ii}, an extension of SL [1] to the imperfect-information setting with synchronous perfect-recall. Our logic presents two original features: first, observations are not bound to players (as is done in extensions of ATL by imperfect information [20] or logics for reasoning about knowledge [21]), and second, we have syntactic observations in the language, which need to be interpreted.

We follow the presentation of SL in [1], except that we make some changes that simplify their presentation but do not change their semantics, and some that allow us to capture imperfect information. We introduce the syntax and semantics of SL_{ii}, carefully detailing these changes. We first fix some notation used throughout the paper.

A. Notation

Let Σ be an alphabet. A *finite* (resp. *infinite*) *word* over Σ is an element of Σ^* (resp. Σ^ω). Words are written $w = w_0w_1w_2\ldots$, i.e., indexing begins with 0. The *length* of a finite word $w = w_0w_1\ldots w_n$ is $|w| := n+1$, and $\text{last}(w) := w_n$ is its last letter. Given a finite (resp. infinite) word w and $0 \leq i \leq |w|$ (resp. $i \in \mathbb{N}$), we let w_i be the letter at position i in w, $w_{\leq i}$ is the prefix of w that ends at position i and $w_{\geq i}$ is the suffix of w that starts at position i. We write $w \preccurlyeq w'$ if w is a prefix of w', and $w^{\preccurlyeq}$ is the set of finite prefixes of word w. Finally, the domain of a mapping f is written $dom(f)$, and for $n \in \mathbb{N}$ we let $[n] := \{i \in \mathbb{N} : 1 \leq i \leq n\}$. The literature sometimes refers to "imperfect information" and sometimes to "partial observation"; we will use the terms interchangeably.

B. Syntax

The syntax of SL_{ii} is similar to that of strategy logic SL in [1]: the only difference is that we annotate strategy quantifiers $\langle\!\langle x \rangle\!\rangle$ by *observation symbols* o. For the rest of the paper, for convenience we fix a number of parameters for our logics and models: AP is a finite set of *atomic propositions*, Ag is a finite set of *players*, Var is a finite set of *variables* and Obs is a finite set of *observation symbols*. When we consider model-checking problems, these data are implicitly part of the input.

Definition 1 (SL_{ii} Syntax)**.** *The syntax of* SL_{ii} *is defined by the following grammar:*

$$\varphi := p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathbf{X}\varphi \mid \varphi\mathbf{U}\varphi \mid \langle\!\langle x \rangle\!\rangle^o \varphi \mid (a,x)\varphi$$

where $p \in$ AP*,* $x \in$ Var*,* $o \in$ Obs *and* $a \in$ Ag*.*

We use abbreviations $\top := p \vee \neg p$, $\bot := \neg\top$, $\varphi \to \varphi' := \neg\varphi \vee \varphi'$, $\varphi \leftrightarrow \varphi' := \varphi \to \varphi' \wedge \varphi' \to \varphi$ for boolean connectives, $\mathbf{F}\varphi := \top\mathbf{U}\varphi$, $\mathbf{G}\varphi := \neg\mathbf{F}\neg\varphi$ for temporal operators, and finally $[\![x]\!]^o\varphi := \neg\langle\!\langle x \rangle\!\rangle^o\neg\varphi$.

The notion of free variables and sentences are defined as for SL. We recall these for completeness (formal definitions are in the appendix). A variable x appears *free* in a formula φ if it appears out of the scope of a strategy quantifier, and an player a appears free in φ if a temporal operator (either $\mathbf{X}$ or $\mathbf{U}$) appears in φ out of the scope of any binding for player a. We let *free*(φ) be the set of variables and players that appear free in φ. If *free*(φ) is empty, φ is a *sentence*.

C. Semantics

The models of SL_{ii} are like those of SL, i.e., concurrent game structures, but extended by a finite set of observations Obs and, for each $o \in$ Obs, by an equivalence-relation $\mathcal{O}(o)$ over positions that represents what a player using a strategy with that observation can see. That is, $\mathcal{O}(o)$-equivalent positions are indistinguishable to a player using a strategy associated with observation o.

Definition 2 (CGS_{ii})**.** *A* concurrent game structure with imperfect information *(or CGS*$_{\text{ii}}$ *for short)* $\mathcal{G} = (\text{Ac}, V, E, \ell, v_\iota, \mathcal{O})$ *where*

- Ac *is a finite non-empty set of* actions*,*
- V *is a finite non-empty set of* positions*,*
- $E : V \times \text{Ac}^{\text{Ag}} \to V$ *is a* transition function*,*
- $\ell : V \to 2^{\text{AP}}$ *is a* labelling function*,*
- $v_\iota \in V$ *is an* initial position*,*
- $\mathcal{O} : \text{Obs} \to V \times V$ *is an* observation interpretation *mapping observations to equivalence relations on positions.*

We may write $\sim_o$ for $\mathcal{O}(o)$, and $v \in \mathcal{G}$ for $v \in V$.

The notions of joint actions, plays, strategies and assignments are similar to those for SL. We will recall these for completeness. Since the main difference between the semantics of SL and SL_{ii} is in the strategy-quantification case where we require strategies to be consistent with observations, we define synchronous perfect recall and o-strategies before defining the semantics of SL_{ii}. Finally, we mention that we make some (inconsequential) simplifications to the semantics of SL as proposed in [1]: i) we dispense with partial assignments and only consider complete assignments, ii) our semantics are w.r.t. a finite play instead of a position (in the Appendix we prove that these simplifications do not change the expressive power of SL).

Joint actions. In a position $v \in V$, each player a chooses an action $c_a \in$ Ac, and the game proceeds to position $E(v, \boldsymbol{c})$, where $\boldsymbol{c} \in \text{Ac}^{\text{Ag}}$ stands for the *joint action* $(c_a)_{a\in\text{Ag}}$. Given a joint action $\boldsymbol{c} = (c_a)_{a\in\text{Ag}}$ and $a \in$ Ag, we let $\boldsymbol{c}_a$ denote c_a. For each position $v \in V$, $\ell(v)$ is the set of atomic propositions that hold in v.

Plays and strategies. A *finite* (resp. *infinite*) *play* is a finite (resp. infinite) word $\rho = v_0\ldots v_n$ (resp. $\pi = v_0v_1\ldots$) such that $v_0 = v_\iota$ and for all i with $0 \leq i < |\rho| - 1$ (resp. $i \geq 0$), there exists a joint action $\boldsymbol{c}$ such that $E(v_i, \boldsymbol{c}) = v_{i+1}$. We let Plays be the set of finite plays. A *strategy* is a function $\sigma :$ Plays $\to$ Ac, and the set of all strategies is denoted *Str*.

Assignments. An *assignment* is a function $\chi :$ Ag$\cup$Var $\to$ *Str*, assigning a strategy to each player and variable. For an assignment χ, player a and strategy σ, $\chi[a \mapsto \sigma]$ is the assignment that maps a to σ and is equal to χ on the rest of its domain, and $\chi[x \mapsto \sigma]$ is defined similarly, where x is a variable.

Outcomes. For an assignment χ and a finite play ρ, we let out(χ, ρ) be the only infinite play that starts with ρ and is then extended by letting players follow the strategies assigned by χ.

Formally, $\text{out}(\chi,\rho) := \rho\cdot v_1 v_2 \dots$ where, for all $i \geq 0$, $v_{i+1} = E(v_i, \boldsymbol{c})$, where $v_0 = \text{last}(\rho)$ and $\boldsymbol{c} = (\chi(a)(\rho\cdot v_1 \dots v_i))_{a\in\text{Ag}}$.

Synchronous perfect recall. In this work we consider players with *synchronous perfect recall*, meaning that each player remembers the whole history of a play, a classic assumption in games with imperfect information and logics of knowledge and time. Each observation relation is thus extended to finite plays as follows: $\rho \sim_o \rho'$ if $|\rho| = |\rho|'$ and $\rho_i \sim_o \rho'_i$ for every $i \in \{0,\dots,|\rho|-1\}$. For $o \in \text{Obs}$, an *o-strategy* is a strategy $\sigma : V^+ \to \text{Ac}$ such that $\sigma(\rho) = \sigma(\rho')$ whenever $\rho \sim_o \rho'$. The latter constraint captures the essence of imperfect information, which is that players can base their strategic choices only on the information available to them. For $o \in \text{Obs}$ we let Str_o be the set of all o-strategies.

Definition 3 (SL_{ii} Semantics)**.** *The semantics $\mathcal{G},\chi,\rho \models \varphi$ is defined inductively, where φ is an SL_{ii}-formula, $\mathcal{G} = (\text{Ac}, V, E, \ell, v_\iota, \mathcal{O})$ is a CGS$_{\text{ii}}$, ρ is a finite play, and χ is an assignment:*

$$
\begin{array}{lll}
\mathcal{G},\chi,\rho \models p & \textit{if} & p \in \ell(\textit{last}(\rho)) \\
\mathcal{G},\chi,\rho \models \neg\varphi & \textit{if} & \mathcal{G},\chi,\rho \not\models \varphi \\
\mathcal{G},\chi,\rho \models \varphi \vee \varphi' & \textit{if} & \mathcal{G},\chi,\rho \models \varphi \textit{ or } \mathcal{G},\chi,\rho \models \varphi' \\
\mathcal{G},\chi,\rho \models \langle\!\langle x\rangle\!\rangle^o \varphi & \textit{if} & \exists\, \sigma \in \text{Str}_o \textit{ s.t. } \mathcal{G},\chi[x\mapsto\sigma],\rho \models \varphi \\
\mathcal{G},\chi,\rho \models (a,x)\varphi & \textit{if} & \mathcal{G},\chi[a\mapsto\chi(x)],\rho \models \varphi
\end{array}
$$

and, writing $\pi = \text{out}(\chi,\rho)$:

$$
\begin{array}{lll}
\mathcal{G},\chi,\rho \models \mathbf{X}\varphi & \textit{if} & \mathcal{G},\chi,\pi^{\leq|\rho|} \models \varphi \\
\mathcal{G},\chi,\rho \models \varphi\mathbf{U}\varphi' & \textit{if} & \exists\, i \geq 0 \textit{ s.t. } \mathcal{G},\chi,\pi^{\leq|\rho|+i-1} \models \varphi' \\
 & & \quad \textit{and } \forall\, j \textit{ s.t. } 0 \leq j < i, \\
 & & \mathcal{G},\chi,\pi^{\leq|\rho|+j-1} \models \varphi
\end{array}
$$

To explain the temporal operators, we remind the reader that positions begin at 0 and thus π_n is the $(n+1)$-st position of π. The satisfaction of a sentence is independent of the assignment (the easy proof is in the Appendix). For an SL_{ii} sentence φ we thus let $\mathcal{G},\rho \models \varphi$ if $\mathcal{G},\chi,\rho \models \varphi$ for some assignment χ, and we write $\mathcal{G} \models \varphi$ if $\mathcal{G},v_\iota \models \varphi$.

D. Model checking and hierarchical instances

Model Checking. We now introduce the main decision problem of this paper, i.e., the model-checking problem for SL_{ii}. An SL_{ii}*-instance* is a formula/model pair $(\Phi,\mathcal{G})$ where $\Phi \in \mathsf{SL}_{\text{ii}}$ and $\mathcal{G}$ is a CGS$_{\text{ii}}$. The *model-checking problem* for SL_{ii} is the decision problem that, given an SL_{ii}-instance $(\Phi,\mathcal{G})$, returns 'yes' if $\mathcal{G} \models \Phi$, and 'no' otherwise.

It is well known that deciding the existence of winning strategies in multi-player games with imperfect information is undecidable for reachability objectives [15]. Since this problem is easily reduced to the model-checking problem for SL_{ii}, we get the following result.

Theorem 1. *The model-checking problem for SL_{ii} is undecidable.*

Hierarchical instances. We now isolate a sub-problem obtained by restricting attention to *hierarchical instances*. Intuitively, an SL_{ii}-instance $(\Phi,\mathcal{G})$ is hierarchical if, as one goes down a path in the syntactic tree of Φ, the observations tied to quantifications become finer. We now make these notions precise.

Definition 4 (Hierarchical instances)**.** *An SL_{ii}-instance $(\Phi,\mathcal{G})$ is* hierarchical *if for all subformulas φ_1,φ_2 of Φ of the form $\varphi_2 = \langle\!\langle x\rangle\!\rangle^{o_2}\varphi'_2$ and $\varphi_1 = \langle\!\langle y\rangle\!\rangle^{o_1}\varphi'_1$ where φ_1 is a subformula of φ'_2, it holds that $\mathcal{O}(o_1) \subseteq \mathcal{O}(o_2)$.*

If $\mathcal{O}(o_1) \subseteq \mathcal{O}(o_2)$ we say that o_1 is *finer* than o_2 in $\mathcal{G}$, and that o_2 is *coarser* than o_1 in $\mathcal{G}$. Intuitively, this means that a player with observation o_1 observes game $\mathcal{G}$ no worse than, i.e., is not less informed, a player with observation o_2.

Example 1 (Security levels)**.** *We illustrate hierarchical instances in a "security levels" scenario, e.g., higher levels have access to more data (i.e., can observe more). Assume that the CGS$_{\text{ii}}$ $\mathcal{G}$ has $\mathcal{O}(o_3) \subseteq \mathcal{O}(o_2) \subseteq \mathcal{O}(o_1)$ (i.e., level 3 has the highest security clearance, while level 1 has the lowest). Let $\varphi = (a_1,x_1)(a_2,x_2)(a_3,x_3)\mathbf{G}p$. The SL_{ii} formula $\Phi := \langle\!\langle x_1\rangle\!\rangle^{o_1}[\![x_2]\!]^{o_2}\langle\!\langle x_3\rangle\!\rangle^{o_3}\varphi$ forms a hierarchical instance when paired with $\mathcal{G}$. It expresses that the player a_1 (with lowest clearance) can collude with player a_3 (having the highest clearance) to ensure a safety property p, even in the presence of an adversary a_2 (with intermediate clearance), as long as the strategy used by a_3 can also depend on the strategy used by a_2.*

On the other hand, the similar formula $\langle\!\langle x_1\rangle\!\rangle^{o_1}\langle\!\langle x_3\rangle\!\rangle^{o_3}[\![x_2]\!]^{o_2}\varphi$, which implies that the strategy used by a_3 cannot depend on the adversarial strategy used by a_2, does not form a hierarchical instance when paired with $\mathcal{G}$.

Here is the main contribution of this paper:

Theorem 2. *The model-checking problem for SL_{ii} restricted to the class of hierarchical instances is decidable.*

This is proved in Section V by reducing it to the model-checking problem of the hierarchical fragment of a logic called $\mathsf{QCTL}^*_{\text{ii}}$, which we introduce, and prove decidable, in Section III. We now give an important corollary of the main theorem.

A Nash equilibrium in a game is a tuple of strategies such that no player has the incentive to deviate. Assuming that goals are written in SL_{ii}, say $goal_i$ for $i \in \text{Ag}$, and $\text{Ag} = \{a_i : i \in [n]\}$, the following formula of SL_{ii} expresses the existence of a Nash equilibrium:

$$\Phi_{\text{NE}} := \langle\!\langle x_1\rangle\!\rangle^{o_1} \dots \langle\!\langle x_n\rangle\!\rangle^{o_n}(a_1,x_1)\dots(a_n,x_n)\Psi_{\text{NE}}$$

where $\Psi_{\text{NE}} := \bigwedge_{i\in[n]}[(\langle\!\langle y_i\rangle\!\rangle^{o_i}(a_i,y_i)goal_i) \to goal_i]$.

A CGS$_{\text{ii}}$ $\mathcal{G}$ is said to *yield hierarchical observation* [13] if the "finer-than" relation is a total ordering, i.e., if for all $o,o' \in \text{Obs}$, either $\mathcal{O}(o) \subseteq \mathcal{O}(o')$ or $\mathcal{O}(o') \subseteq \mathcal{O}(o)$.

Note that the instance $(\Phi_{\text{NE}},\mathcal{G})$ is *not* hierarchical (unless $\mathcal{O}(o_i) = \mathcal{O}(o_j)$ for all $i,j \in \text{Ag}$). *Nonetheless*, we can decide if a game that yields hierarchical observation has a Nash equilibrium:

Corollary 1. *The following problem is decidable: given a CGS$_{\text{ii}}$ that yields hierarchical observation, whether $\mathcal{G} \models \Phi_{\text{NE}}$.*

Proof. The main idea is to use the fact that in a one-player game of partial-observation (such a game occurs when all but one player have fixed their strategies, as in the definition of Nash equilibrium), the player has a strategy enforcing some goal iff the player has a uniform-strategy enforcing that goal. Here are the details. Let $\mathcal{G} = (\mathrm{Ac}, V, E, \ell, v_\iota, \mathcal{O})$ be a $\mathrm{CGS}_{\mathrm{ii}}$ that yields hierarchical observation. Suppose the observation set is Obs. To decide if $\mathcal{G} \models \Phi_{\mathrm{NE}}$ first build a new $\mathrm{CGS}_{\mathrm{ii}}$ $\mathcal{G}' = (\mathrm{Ac}, V, E, \ell, v_\iota, \mathcal{O}')$ over observations $\mathrm{Obs}' := \mathrm{Obs} \cup \{o_p\}$ such that $\mathcal{O}'(o) = \mathcal{O}(o)$ and $\mathcal{O}'(o_p) = \{(v, v) : v \in V\}$, and consider the sentence

$$\Phi' := \langle\!\langle x_1 \rangle\!\rangle^{o_1} \dots \langle\!\langle x_n \rangle\!\rangle^{o_n} (a_1, x_1) \dots (a_n, x_n) \Psi'$$

where $\Psi' := \bigwedge_{i \in [n]} [(\langle\!\langle y_i \rangle\!\rangle^{o_p} (a_i, y_i) goal_i) \to goal_i]$.

Then $(\Phi', \mathcal{G}')$ is a hierarchical instance, and by Theorem 2 we can decide $\mathcal{G}' \models \Phi'$. We claim that $\mathcal{G}' \models \Phi'$ iff $\mathcal{G} \models \Phi_{\mathrm{NE}}$. To see this, it is enough to establish that:

$$\mathcal{G}', \chi, v_\iota \models \langle\!\langle y_i \rangle\!\rangle^{o_p} (a_i, y_i) goal_i \leftrightarrow \langle\!\langle y_i \rangle\!\rangle^{o_i} (a_i, y_i) goal_i,$$

for every $i \in [n]$ and every assignment χ such that $\chi(x_i) = \chi(a_i)$ is an o_i-uniform strategy.

To this end, fix i and χ. The right-to-left implication is immediate (since o_p is finer than o_i). For the converse, let σ be a p-uniform strategy (i.e., perfect-information) such that $\mathcal{G}', \chi[y_i \mapsto \sigma, a_i \mapsto \sigma], v_\iota \models goal_i$. Let $\pi := \mathrm{out}(\chi[y_i \mapsto \sigma, a_i \mapsto \sigma], v_\iota)$. Construct an o_i-uniform strategy σ' that agrees with σ on prefixes of π. This can be done as follows: if $\rho \sim_{o_i} \pi_{\leq j}$ for some j then define $\sigma'(\rho) = \sigma(\pi_{\leq j})$ (note that this is well-defined since if there is some such j then it is unique), and otherwise define $\sigma'(\rho) = a$ for some fixed action $a \in \mathrm{Ac}$. □

E. Comparison with other logics

The main difference between SL and ATL-like strategic logics is that in the latter a strategy is always bound to some player, while in the former bindings and quantifications are separated. This separation adds expressive power, e.g., one can bind the same strategy to different players. Extending ATL with imperfect-information is done by giving each player an indistinguishability relation that its strategies must respect [2]. Our extension of SL by imperfect information, instead, assigns each strategy x an indistinguishability relation o when it is quantified $\langle\!\langle x \rangle\!\rangle^o$. Thus $\langle\!\langle x \rangle\!\rangle^o \varphi$ means "there exists a strategy with observation o such that φ holds". Associating observations in this way, i.e., to strategies rather than players has two consequences. First, it is a clean generalisation of SL in the perfect information setting [1]. Define the *perfect-information fragment of* $\mathsf{SL}_{\mathrm{ii}}$ to be the logic $\mathsf{SL}_{\mathrm{ii}}$ assuming that $\mathrm{Obs} = \{o\}$ and $\mathcal{O}(o) = \{(v, v) : v \in \mathcal{G}\}$. The next proposition says that the perfect-information fragment of $\mathsf{SL}_{\mathrm{ii}}$ is a notational variant of SL [1] (the proof is in the Appendix).

Proposition 1. *For every* SL *sentence* φ *there is an* $\mathsf{SL}_{\mathrm{ii}}$ *sentence* φ' *with* $\mathrm{Obs} = \{o\}$*, such that for every CGS* $\mathcal{G}$ *there is a CGS*$_{\mathrm{ii}}$ $\mathcal{G}'$ *with* $\mathcal{O}(o) = \{(v, v) : v \in \mathcal{G}\}$ *such that* $\mathcal{G} \models \varphi$ *iff* $\mathcal{G}' \models \varphi'$*.*

Second, $\mathsf{SL}_{\mathrm{ii}}$ subsumes imperfect-information extensions of ATL^* that associate observations to players.

Proposition 2. *For every* $\mathsf{ATL}^*_{\mathrm{i,R}}$ *formula*[1] φ *there is an* $\mathsf{SL}_{\mathrm{ii}}$ *formula* φ' *such that for every CGS*$_{\mathrm{ii}}$ $\mathcal{G}$ *there is a CGS*$_{\mathrm{ii}}$ $\mathcal{G}'$ *such that* $\mathcal{G} \models \varphi$ *iff* $\mathcal{G}' \models \varphi'$*.*

We recall that an $\mathsf{ATL}^*_{\mathrm{i,R}}$ formula $\langle A \rangle \psi$ reads as "there are strategies for players in A such that ψ holds whatever players in $\mathrm{Ag} \setminus A$ do". Formula φ' is built from φ by replacing each subformula of the form $\langle A \rangle \psi$, where $A = \{a_1, \dots, a_k\} \subset \mathrm{Ag}$ is a coalition of players and $\mathrm{Ag} \setminus A = \{a_{k+1}, \dots, a_n\}$, with formula $\langle\!\langle x_1 \rangle\!\rangle^{o_1} \dots \langle\!\langle x_k \rangle\!\rangle^{o_k} [\![x_{k+1}]\!]^{o_p} \dots [\![x_n]\!]^{o_p} (a_1, x_1) \dots (a_n, x_n) \psi'$, where ψ' is the translation of ψ. Then $\mathcal{G}'$ is obtained from $\mathcal{G}$ by interpreting each o_i as the equivalence relation for player i in $\mathcal{G}$, and interpreting o_p as the identity relation.

Third, $\mathsf{SL}_{\mathrm{ii}}$ also subsumes the imperfect-information extension of ATL^* with strategy context (see [3] for the definition of $\mathsf{ATL}^*_{\mathrm{sc}}$ with partial observation, which we refer to as $\mathsf{ATL}^*_{\mathrm{sc,i}}$).

Proposition 3. *For every* $\mathsf{ATL}^*_{\mathrm{sc,i}}$ *formula* φ *there is an* $\mathsf{SL}_{\mathrm{ii}}$ *formula* φ' *such that for every CGS*$_{\mathrm{ii}}$ $\mathcal{G}$ *there is a CGS*$_{\mathrm{ii}}$ $\mathcal{G}'$ *such that* $\mathcal{G} \models \varphi$ *iff* $\mathcal{G}' \models \varphi'$*.*

The only difference between $\mathsf{ATL}^*_{\mathrm{sc,i}}$ and $\mathsf{ATL}^*_{\mathrm{i,R}}$ is the following: in $\mathsf{ATL}^*_{\mathrm{i,R}}$, when a subformula of the form $\langle A \rangle \psi$ is met, we quantify existentially on strategies for players in A, and then we consider all possible outcomes obtained by letting other players behave however they want. Therefore, if any player in $\mathrm{Ag} \setminus A$ had previously been assigned a strategy, it is forgotten. In $\mathsf{ATL}^*_{\mathrm{sc,i}}$ on the other hand, these strategies are stored in a *strategy context*, which is a *partial* assignment χ, defined for the subset of players currently bound to a strategy. A strategy context allows one to quantify universally only on strategies of players who are not in A and who are not already bound to a strategy. It is then easy to adapt the translation presented for Proposition 2 by parameterising it with a strategy context. $\mathcal{G}'$ is defined as for Proposition 2.

Fourth, there is a natural and simple translation of instances of the model-checking problem of CL [19] into the hierarchical instances of $\mathsf{SL}_{\mathrm{ii}}$. Moreover, the image of this translation consists of instances of $\mathsf{SL}_{\mathrm{ii}}$ with a very restricted form, i.e., atoms mentioned in the $\mathsf{SL}_{\mathrm{ii}}$-formula are observable (by all observations of the $\mathrm{CGS}_{\mathrm{ii}}$), i.e., $v \sim_o v'$ implies $p \in \ell(v) \leftrightarrow p \in \ell(v')$.

Proposition 4. *There is an effective translation that, given a* CL*-instance* $(\mathcal{S}, \varphi)$ *produces a hierarchical* $\mathsf{SL}_{\mathrm{ii}}$*-instance* $(\mathcal{G}, \Phi)$ *such that*

1) $\mathcal{S} \models \varphi$ *iff* $\mathcal{G} \models \Phi$*,*
2) *For all atoms* p *in* Φ*, and all observations* $o \in \mathrm{Obs}$*, we have that* $v \sim_o v'$ *implies* $p \in \ell(v) \leftrightarrow p \in \ell(v')$*.*

To do this, one first translates CL into (hierarchical) $\mathsf{QCTL}^*_{\mathrm{ii}}$, the latter is defined in the next section. This step is a simple

[1] See [2] for the definition of $\mathsf{ATL}^*_{\mathrm{i,R}}$, where subscript i refers to "imperfect information" and subscript R to "perfect recall". Also, we consider the so-called *objective semantics* for $\mathsf{ATL}^*_{\mathrm{i,R}}$.

reflection of the semantics of CL in that of $\mathsf{QCTL}^*_{\mathsf{ii}}$. Then one translates $\mathsf{QCTL}^*_{\mathsf{ii}}$ into $\mathsf{SL}_{\mathsf{ii}}$ by a simple adaptation of the translation of QCTL^* into $\mathsf{ATL}^*_{\mathsf{sc}}$ [22]. Details are in the Appendix.

III. QCTL^* WITH IMPERFECT INFORMATION

In this section we introduce an imperfect-information extension of QCTL^* [14], [23]–[27]. In order to introduce imperfect information, instead of considering equivalence relations between states as in concurrent game structures, we will enrich Kripke structures by giving internal structure to their states, i.e., we see states as n-tuples of local states. This way of modelling imperfect information is inspired from Reif's multi-player game structures [15] and distributed systems [16], and we find it very suitable to application of automata techniques, as discussed in Section III-C.

The syntax of $\mathsf{QCTL}^*_{\mathsf{ii}}$ is similar to that of QCTL^*, except that we annotate second-order quantifiers by subsets $\mathbf{o} \subseteq [n]$. The idea is that quantifiers annotated by $\mathbf{o}$ can only "observe" the local states indexed by $i \in \mathbf{o}$. We define the tree-semantics of $\mathsf{QCTL}^*_{\mathsf{ii}}$: this means that we interpret formulas on trees that are the unfoldings of Kripke structures (this will capture the fact that players in $\mathsf{SL}_{\mathsf{ii}}$ have synchronous perfect recall).

We then define the syntactic class of *hierarchical formulas* and prove, using an automata-theoretic approach, that model checking this class of formulas is decidable.

A. $\mathsf{QCTL}^*_{\mathsf{ii}}$ *Syntax*

The syntax of $\mathsf{QCTL}^*_{\mathsf{ii}}$ is very similar to that of QCTL^*: the only difference is that we annotate quantifiers by a set of indices that defines the "observation" of that quantifier.

Definition 5 ($\mathsf{QCTL}^*_{\mathsf{ii}}$ Syntax)**.** *Fix $n \in \mathbb{N}$. The syntax of $\mathsf{QCTL}^*_{\mathsf{ii}}$ is defined by the following grammar:*

$$\varphi := p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathbf{E}\psi \mid \exists^{\mathbf{o}} p.\, \varphi$$
$$\psi := \varphi \mid \neg\psi \mid \psi \vee \psi \mid \mathbf{X}\psi \mid \psi \mathbf{U} \psi$$

where $p \in \mathrm{AP}$ and $\mathbf{o} \subseteq [n]$.

Formulas of type φ are called *state formulas*, those of type ψ are called *path formulas*, and $\mathsf{QCTL}^*_{\mathsf{ii}}$ consists of all the state formulas defined by the grammar. We use standard abbreviation $\top := p \vee \neg p$, $\bot := \neg\top$, $\mathbf{F}\psi := \top\mathbf{U}\psi$, $\mathbf{G}\psi := \neg\mathbf{F}\neg\psi$ and $\mathbf{A}\psi := \neg\mathbf{E}\neg\psi$. We use $\exists p.\, \varphi$ as a shorthand for $\exists^{[n]} p.\, \varphi$. Finally we let $\forall p.\, \varphi := \neg\exists p.\, \neg\varphi$.

Given a $\mathsf{QCTL}^*_{\mathsf{ii}}$ formula φ, we define the set of *quantified propositions* $\mathrm{AP}_\exists(\varphi) \subseteq \mathrm{AP}$ as the set of atomic propositions p such that φ has a subformula of the form $\exists^{\mathbf{o}} p.\, \varphi$. We also define the set of *free propositions* $\mathrm{AP}_f(\varphi) \subseteq \mathrm{AP}$ as the set of atomic propositions that appear out of the scope of any quantifier of the form $\exists^{\mathbf{o}} p$. Observe that $\mathrm{AP}_\exists(\varphi) \cap \mathrm{AP}_f(\varphi)$ may not be empty in general, *i.e.*, a proposition may appear both free and quantified in (different places of) a formula.

B. $\mathsf{QCTL}^*_{\mathsf{ii}}$ *tree-semantics*

We define the semantics on structures whose states are tuples of local states.

Local states. Let $\{L_i\}_{i\in[n]}$ denote $n \in \mathbb{N}$ disjoint finite sets of *local states*. For $I \subseteq [n]$, we let $L_I := \prod_{i\in I} L_i$ if $I \neq \emptyset$, and $L_\emptyset := \{\mathbf{0}\}$ where $\mathbf{0}$ is a special symbol.

Concrete observations. A set $\mathbf{o} \subseteq [n]$ is called a *concrete observation* (to distinguish it from observations o in the definitions of $\mathsf{SL}_{\mathsf{ii}}$).

Compound Kripke structures. These are like Kripke structures except that the states are elements of $L_{[n]}$. A *compound Kripke structure*, or CKS, over AP, is a tuple $\mathcal{S} = (S, R, s_\iota, \ell)$ where $S \subseteq L_{[n]}$ is a set of *states*, $R \subseteq S \times S$ is a left-total[2] *transition relation*, $s_\iota \in S$ is an *initial state*, and $\ell : S \to 2^{\mathrm{AP}}$ is a *labelling function*.

A *path* in $\mathcal{S}$ is an infinite sequence of states $\lambda = s_0 s_1 \ldots$ such that for all $i \in \mathbb{N}$, $(s_i, s_{i+1}) \in R$. For $s \in S$, we let $\mathrm{Paths}(s)$ be the set of all paths that start in s. A *finite path* is a finite non-empty prefix of a path. We may write $s \in \mathcal{S}$ for $s \in S$. Since we will interpret $\mathsf{QCTL}^*_{\mathsf{ii}}$ on unfoldings of CKS, we now define infinite trees.

Trees. In many works, trees are defined as prefix-closed sets of words with the empty word ϵ as root. Here trees represent unfoldings of Kripke structures, and we find it more convenient to see a node u as a sequence of states and the root as the initial state. Let X be a finite set (typically a set of states). An *X-tree* τ is a nonempty set of words $\tau \subseteq X^+$ such that:

- there exists $r \in X$, called the *root* of τ, such that each $u \in \tau$ starts with r ($r \preccurlyeq u$);
- if $u \cdot x \in \tau$ and $u \neq \epsilon$, then $u \in \tau$, and
- if $u \in \tau$ then there exists $x \in X$ such that $u \cdot x \in \tau$.

The elements of a tree τ are called *nodes*. If $u \cdot x \in \tau$, we say that $u \cdot x$ is a *child* of u. The *depth* of a node u is $|u|$. An X-tree τ is *complete* if $u \in \tau, x \in X$ implies $u \cdot x \in \tau$. A *path* in τ is an infinite sequence of nodes $\lambda = u_0 u_1 \ldots$ such that for all $i \in \mathbb{N}$, u_{i+1} is a child of u_i, and $Paths(u)$ is the set of paths that start in node u. An AP-*labelled X-tree*, or (AP, X)-*tree* for short, is a pair $t = (\tau, \ell)$, where τ is an X-tree called the *domain* of t and $\ell : \tau \to 2^{\mathrm{AP}}$ is a *labelling*. For a labelled tree $t = (\tau, \ell)$ and an atomic proposition $p \in \mathrm{AP}$, we define the *p-projection of t* as the labelled tree $t \Downarrow_p := (\tau, \ell \Downarrow_p)$, where for each $u \in \tau$, $\ell \Downarrow_p (u) := \ell(u) \setminus \{p\}$. For a set of trees $\mathcal{L}$, we let $\mathcal{L} \Downarrow_p := \{t \Downarrow_p \mid t \in \mathcal{L}\}$. Finally, two labelled trees $t = (\tau, \ell)$ and $t' = (\tau', \ell')$ are *equivalent modulo p*, written $t \equiv_p t'$, if $t \Downarrow_p = t' \Downarrow_p$ (in particular, $\tau = \tau'$).

Narrowing. Let X and Y be two finite sets, and let $(x, y) \in X \times Y$. The *X-narrowing* of (x, y) is $(x, y)\downarrow_X := x$. This definition extends naturally to words over $X \times Y$ (pointwise), and thus to $X \times Y$-trees.

For $J \subseteq I \subseteq [n]$ and $z = (l_i)_{i\in I} \in L_I$, we also define $z\downarrow_J := z \downarrow_{L_J}$, where z is seen as a pair $z = (z_1, z_2) \in L_J \times L_{I\setminus J}$. This is well defined because having taken sets L_i to be disjoint, the ordering of local states in z is indifferent. We also extend this definition to words and trees.

[2] *i.e.*, for all $s \in S$, there exists s' such that $(s, s') \in R$.

Observe that when narrowing a tree, nodes with same narrowing are merged. In particular, for every L_I-tree τ, $\tau{\downarrow_\emptyset}$ is the only $L_\emptyset$-tree, $\mathbf{0}^\omega$.

Quantification and uniformity. In $\mathsf{QCTL}^*_{\mathrm{ii}}$ the intuitive meaning of $\exists^{\mathbf{o}} p.\,\varphi$ in a tree t is that there is some equivalent tree t' modulo p such that t' is $\mathbf{o}$-uniform in p and satisfies φ. Intuitively, a tree is $\mathbf{o}$-uniform in p if it is uniformly labelled by p, i.e., if every two nodes that are indistinguishable when projected onto the local states indexed by $\mathbf{o} \subseteq [n]$ agree on their labelling of p.

Definition 6 ($\mathbf{o}$-indistinguishability and $\mathbf{o}$-uniformity in p)**.** *Fix $\mathbf{o} \subseteq [n]$ and $I \subseteq [n]$.*

- *Two tuples $x, x' \in L_I$ are* $\mathbf{o}$-indistinguishable*, written $x \approx_{\mathbf{o}} x'$, if $x{\downarrow_{I\cap\mathbf{o}}} = x'{\downarrow_{I\cap\mathbf{o}}}$.*
- *Two words $u = u_0 \dots u_i$ and $u' = u'_0 \dots u'_j$ over alphabet L_I are* $\mathbf{o}$-indistinguishable*, written $u \approx_{\mathbf{o}} u'$, if $i = j$ and for all $k \in \{0, \dots, i\}$ we have $u_k \approx_{\mathbf{o}} u'_k$.*
- *A tree t is* $\mathbf{o}$-uniform in p *if for every pair of nodes $u, u' \in \tau$ such that $u \approx_{\mathbf{o}} u'$, we have $p \in \ell(u)$ iff $p \in \ell(u')$.*

Finally, we inductively define the satisfaction relation $\models$ for the semantics on trees, where $t = (\tau, \ell)$ is a 2^{AP}-labelled L_I-tree, u is a node and λ is a path in τ:

$t, u \models p$	if	$p \in \ell(u)$
$t, u \models \neg\varphi$	if	$t, u \not\models \varphi$
$t, u \models \varphi \vee \varphi'$	if	$t, u \models \varphi$ or $t, u \models \varphi'$
$t, u \models \mathbf{E}\psi$	if	$\exists\, \lambda \in Paths(u)$ s.t. $t, \lambda \models \psi$
$t, u \models \exists^{\mathbf{o}} p.\,\varphi$	if	$\exists\, t' \equiv_p t$ s.t. t' is $\mathbf{o}$-uniform in p and $t', u \models \varphi$.
$t, \lambda \models \varphi$	if	$t, \lambda_0 \models \varphi$
$t, \lambda \models \neg\psi$	if	$t, \lambda \not\models \psi$
$t, \lambda \models \psi \vee \psi'$	if	$t, \lambda \models \psi$ or $t, \lambda \models \psi'$
$t, \lambda \models \mathbf{X}\psi$	if	$t, \lambda_{\geq 1} \models \psi$
$t, \lambda \models \psi \mathbf{U} \psi'$	if	$\exists\, i \geq 0$ s.t. $t, \lambda_{\geq i} \models \psi'$ and $\forall j$ s.t. $0 \leq j < i,\ t, \lambda_{\geq j} \models \psi$

We write $t \models \varphi$ for $t, r \models \varphi$, where r is the root of t.

Example 2. *Consider the following* CTL *formula:*

$$\mathbf{border}(p) := \mathbf{AF}p \wedge \mathbf{AG}(p \rightarrow \mathbf{AXAG}\neg p).$$

This formula holds in a labelled tree if and only if each path contains exactly one node labelled with p. Now, consider the following $\mathsf{QCTL}^*_{\mathrm{ii}}$ *formula:*

$$\mathbf{level}(p) := \exists^{\emptyset} p.\,\mathbf{border}(p).$$

For a blind quantifier, two nodes of a tree are indistinguishable if and only if they have same depth. Therefore, this formula holds on a tree iff the p's label all and only the nodes at some fixed depth. This formula can thus be used to capture the equal level predicate on trees. Actually, just as QCTL^* *captures* MSO*, one can prove that* $\mathsf{QCTL}^*_{\mathrm{ii}}$ *with tree semantics subsumes* MSO *with equal level [17], [28], [29]. In Theorem 3 we make use of a similar observation to prove that model-checking* $\mathsf{QCTL}^*_{\mathrm{ii}}$ *is undecidable.*

Model-checking problem for $\mathsf{QCTL}^*_{\mathrm{ii}}$ under tree semantics. For the model-checking problem, we interpret $\mathsf{QCTL}^*_{\mathrm{ii}}$ on unfoldings of CKSs.

Tree unfoldings $t_{\mathcal{S}}(s)$. Let $\mathcal{S} = (S, R, s_\iota, \ell)$ be a compound Kripke structure over AP, and let $s \in S$. The *tree-unfolding of $\mathcal{S}$ from s* is the (AP, S)-tree $t_{\mathcal{S}}(s) := (\tau, \ell')$, where τ is the set of all finite paths that start in s, and for every $u \in \tau$, $\ell'(u) := \ell(\mathrm{last}(u))$. Given a CKS $\mathcal{S}$, a state $s \in \mathcal{S}$ and a $\mathsf{QCTL}^*_{\mathrm{ii}}$ formula φ, we write $\mathcal{S}, s \models \varphi$ if $t_{\mathcal{S}}(s) \models \varphi$. Write $\mathcal{S} \models \varphi$ if $t_{\mathcal{S}}(s_\iota) \models \varphi$.

The *model-checking problem for* $\mathsf{QCTL}^*_{\mathrm{ii}}$ *under tree-semantics* is the following decision problem: given an instance $(\varphi, \mathcal{S})$ where $\mathcal{S}$ is a CKS, and φ is a $\mathsf{QCTL}^*_{\mathrm{ii}}$ formula, return 'Yes' if $\mathcal{S} \models \varphi$ and 'No' otherwise.

C. Discussion of the definition of $\mathsf{QCTL}^*_{\mathrm{ii}}$

We now motivate in detail some aspects of $\mathsf{QCTL}^*_{\mathrm{ii}}$.

Modelling of imperfect information. We model imperfect information by means of local states (rather than equivalence relations) since this greatly facilitates the use of automata techniques. More precisely, in our decision procedure of Section IV, we make extensive use of an operation on tree automata called *narrowing*, which was introduced in [30] to deal with imperfect-information in the context of distributed synthesis for temporal specifications. Given an automaton $\mathcal{A}$ that works on $X \times Y$-trees, where X and Y are two finite sets, and assuming that we want to model an operation performed on trees while observing only the X component of each node, this narrowing operation allows one to build from $\mathcal{A}$ an automaton $\mathcal{A}'$ that works on X-trees, such that $\mathcal{A}'$ accepts an X-tree if and only if $\mathcal{A}$ accepts its widening to $X \times Y$ (see Section IV for details). One can then make this automaton $\mathcal{A}'$ perform the desired operation, which will by necessity be performed uniformly with regards to the partial observation, since the Y component is absent from the input trees.

With our definition of compound Kripke structures, their unfoldings are trees over the Cartesian product $L_{[n]}$. To model a quantification $\exists^{\mathbf{o}} p$ with observation $\mathbf{o} \subseteq [n]$, we can thus use the narrowing operation to forget about components L_i, for $i \in [n] \setminus \mathbf{o}$. We then use the classic projection of nondeterministic tree automata to perform existential quantification on atomic proposition p. Since the choice of the p-labelling is made directly on $L_{\mathbf{o}}$-trees, it is necessarily $\mathbf{o}$-uniform.

Choice of the tree semantics. QCTL^* is obtained by adding to CTL^* second-order quantification on atomic propositions. Several semantics have been considered. The two most studied ones are the *structure semantics*, in which formulas are evaluated directly on Kripke structures, and the *tree semantics*, in which Kripke structures are first unfolded into infinite trees. Tree semantics thus allows quantifiers to choose the value of a quantified atomic proposition in each *finite path* of the model, while in structure semantics the choice is only made in each state. When QCTL^* is used to express existence

of strategies, existential quantification on atomic propositions labels the structure with strategic choices; in this kind of application, structure semantics reflects so-called *positional* or *memoryless* strategies, while tree semantics captures *perfect-recall* or *memoryfull* strategies. Since in this work we are interested in perfect-recall strategies, we only consider the tree semantics.

D. Model checking $\mathsf{QCTL}^*_{\mathrm{ii}}$

We now prove that the model-checking problem for $\mathsf{QCTL}^*_{\mathrm{ii}}$ under tree semantics is undecidable. This comes as no surprise since, as we will show, $\mathsf{QCTL}^*_{\mathrm{ii}}$ can express the existence of winning strategies in imperfect-information games.

Theorem 3. *The model-checking problem for $\mathsf{QCTL}^*_{\mathrm{ii}}$ under tree-semantics is undecidable.*

Proof. Let $\mathsf{MSO}_{\mathrm{eq}}$ denote the extension of the logic MSO by a binary predicate symbol eq. Formulas of $\mathsf{MSO}_{\mathrm{eq}}$ are interpreted on trees, and the semantics of $\mathrm{eq}(x, y)$ is that x and y have the same depth in the tree. There is a translation of MSO-formulas to QCTL^*-formulas that preserves satisfaction [14]. This translation can be extended to map $\mathsf{MSO}_{\mathrm{eq}}$-formulas to $\mathsf{QCTL}^*_{\mathrm{ii}}$-formulas using the formula $\mathbf{level}(\cdot)$ from Example 2 to help capture the equal-length predicate. Our result follows since the $\mathsf{MSO}_{\mathrm{eq}}$-theory of the binary tree is undecidable [17]. □

IV. A decidable fragment of $\mathsf{QCTL}^*_{\mathrm{ii}}$: hierarchy on observations

The main result of this section is the identification of an important decidable fragment of $\mathsf{QCTL}^*_{\mathrm{ii}}$.

Definition 7 (Hierarchical formulas)**.** *A $\mathsf{QCTL}^*_{\mathrm{ii}}$ formula φ is* hierarchical *if for all subformulas φ_1, φ_2 of the form $\varphi_1 = \exists^{\mathbf{o}_1} p_1.\, \varphi'_1$ and $\varphi_2 = \exists^{\mathbf{o}_2} p_2.\, \varphi'_2$ where φ_2 is a subformula of φ'_1, we have $\mathbf{o}_1 \subseteq \mathbf{o}_2$.*

In other words, a formula is hierarchical if innermost propositional quantifiers observe at least as much as outermost ones. We let $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$ be the set of hierarchical $\mathsf{QCTL}^*_{\mathrm{ii}}$ formulas.

Theorem 4. *Model checking $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$ under tree semantics is non-elementary decidable.*

Since our decision procedure for the hierarchical fragment of $\mathsf{QCTL}^*_{\mathrm{ii}}$ is based on an automata-theoretic approach, we recall some definitions and results for alternating tree automata.

A. Alternating parity tree automata

We briefly recall the notion of alternating (parity) tree automata. Because it is sufficient for our needs and simplifies definitions, we assume that all input trees are complete trees. For a set Z, $\mathbb{B}^+(Z)$ is the set of formulas built from the elements of Z as atomic propositions using the connectives $\vee$ and $\wedge$, and with $\top, \bot \in \mathbb{B}^+(Z)$. An *alternating tree automaton (ATA) on (AP, X)-trees* is a structure $\mathcal{A} = (Q, \delta, q_\iota, C)$ where Q is a finite set of states, $q_\iota \in Q$ is an initial state, $\delta : Q \times 2^{\mathrm{AP}} \to \mathbb{B}^+(X \times Q)$ is a transition function, and $C : Q \to \mathbb{N}$ is a colouring function. To ease reading we shall write atoms in $\mathbb{B}^+(X \times Q)$ between brackets, such as $[x, q]$. A *nondeterministic tree automaton (NTA) on (AP, X)-trees* is an ATA $\mathcal{A} = (Q, \delta, q_\iota, C)$ such that for every $q \in Q$ and $a \in 2^{\mathrm{AP}}$, $\delta(q, a)$ is written in disjunctive normal form and for every direction $x \in X$ each disjunct contains exactly one element of $\{x\} \times Q$. Acceptance is defined as usual (see, *e.g.*, [31]), and we let $\mathcal{L}(\mathcal{A})$ be the set of trees accepted by $\mathcal{A}$.

We recall three classic results on tree automata. The first one is that nondeterministic tree automata are closed under projection, and was established by Rabin to deal with second-order monadic quantification:

Theorem 5 (Projection [32])**.** *Given an NTA $\mathcal{N}$ and an atomic proposition $p \in \mathcal{AP}$, one can build an NTA $\mathcal{N} \Downarrow_p$ such that $\mathcal{L}(\mathcal{N} \Downarrow_p) = \mathcal{L}(\mathcal{N}) \Downarrow_p$.*

Because it will be important to understand the automata construction for our decision procedure in Section IV, we briefly recall that the projected automaton $\mathcal{N} \Downarrow_p$ is simply automaton $\mathcal{N}$ with the only difference that when it reads the label of a node, it can choose whether p is there or not: if δ is the transition function of $\mathcal{N}$, that of $\mathcal{N} \Downarrow_p$ is $\delta'(q, a) = \delta(q, a \cup \{p\}) \vee \delta(q, a \setminus \{p\})$, for any state q and label $a \in 2^{\mathrm{AP}}$. Another way of seeing it is that $\mathcal{N} \Downarrow_p$ first guesses a p-labelling for the input tree, and then simulates $\mathcal{N}$ on this modified input. To prevent $\mathcal{N} \Downarrow_p$ from guessing different labels for a same node in different executions, it is crucial that $\mathcal{N}$ be nondeterministic, which is the reason why we need the next classic result: the crucial simulation theorem, due to Muller and Schupp.

Theorem 6 (Simulation [33])**.** *Given an ATA $\mathcal{A}$, one can build an NTA $\mathcal{N}$ such that $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{A})$.*

The third result was established by Kupferman and Vardi to deal with imperfect information aspects in distributed synthesis. To state it we need to define a widening operation on trees which simply expands the directions in a tree.

Widening [30]. Let X and Y be two finite sets, let t be an X-tree with root x, and let $y \in Y$. The *Y-widening* of t with root (x, y) is the $X \times Y$-tree

$$\tau \uparrow^Y_y := \{u \in (x, y) \cdot (X \times Y)^* \mid u \downarrow_X \in \tau\}.$$

For an (AP, X)-tree $t = (\tau, \ell)$, we let $t \uparrow^Y_y := (\tau \uparrow^Y_y, \ell')$ where $\ell'(u) := \ell(u \downarrow_X)$.

Similarly to the narrowing operation, we extend this definition to tuples of local states by letting, for $J \subseteq I \subseteq [n]$, τ an L_J-tree and $z' \in L_{I \setminus J}$,

$$\tau \uparrow^I_{z'} := \tau \uparrow^{L_{I \setminus J}}_{z'},$$

and similarly for a labelled L_J-tree t,

$$t \uparrow^I_{z'} := t \uparrow^{L_{I \setminus J}}_{z'}.$$

Recall that because the sets of local states L_i are disjoint, the order of local states in a tuple does not matter and we can identify L_I with $L_J \times L_{I \setminus J}$.

The rough idea of the narrowing operation on ATA is that, if one just observes X, uniform p-labellings on $X \times Y$-trees can be obtained by choosing the labellings directly on X-trees, and then lifting them to $X \times Y$.

Theorem 7 (Narrowing [30])**.** *Given an ATA $\mathcal{A}$ on $X \times Y$-trees one can build an ATA $\mathcal{A}{\downarrow_X}$ on X-trees such that for all $y \in Y$, $t \in \mathcal{L}(\mathcal{A}{\downarrow_X})$ iff $t{\uparrow_y^{X\times Y}} \in \mathcal{L}(\mathcal{A})$.*

B. Translating $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$ to ATA

In order to prove Theorem 4 we need some more notations and a technical lemma that contains the automata construction.

For every $\varphi \in \mathsf{QCTL}^*_{\mathrm{ii}}$, we let

$$I_\varphi := \bigcap_{\mathbf{o}\in \mathrm{Obs}(\varphi)} \mathbf{o} \subseteq [n],$$

where $\mathrm{Obs}(\varphi)$ is the set of concrete observations that occur in φ, with the intersection over the empty set defined as $[n]$. We also let $L_\varphi := L_{I_\varphi}$ (recall that for $I \subseteq [n]$, $L_I = \prod_{i\in I} L_i$).

Our construction, that transforms a $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$ formula φ and a CKS $\mathcal{S}$ into an ATA, builds upon the classic construction from [34], which builds hesitant ATA for CTL^* formulas. Since our aim here is to establish decidability and that the hesitant condition is only used to improve complexity, we do not consider it. However we need to develop an original technique to implement quantifiers with imperfect information thanks to automata narrowing and projection.

The classical approach to model checking via tree automata is to build an automaton that accepts all tree models of the input formula, and check whether it accepts the unfolding of the model [34]. We now explain how we adapt this approach.

Narrowing of non-uniform trees. Quantification on atomic propositions is classically performed by means of automata projection (see Theorem 5). But in order to obtain a labelling that is uniform with regards to the observation of the quantifier, we need to make use of the narrow operation (see Theorem 7). Intuitively, to check that a formula $\exists^{\mathbf{o}} p.\, \varphi$ holds in a tree t, we would like to work on its narrowing $t' := t{\downarrow_{\mathbf{o}}}$, guess a labelling for p on this tree thanks to automata projection, thus obtaining a tree t'_p, take its widening $t''_p := t'_p{\uparrow^{[n]}}$, obtaining a tree with an $\mathbf{o}$-uniform labelling for p, and then check that φ holds on t''_p. The problem here is that, while the narrowing $\tau{\downarrow_{\mathbf{o}}}$ of an unlabelled tree τ is well defined (see Section III-B), that of a labelled tree $t = (\tau, \ell)$ is undefined: indeed, unless t is o-uniform in every atomic proposition in AP, there is no way to define the labelling of $\tau{\downarrow_{\mathbf{o}}}$ without losing information. This implies that we cannot feed (a narrowing of) the unfolding of the model to our automata. Still, we need an input tree to be successively labelled and widened to guess uniform labellings.

Splitting quantified from free propositions. To address this problem, we remark that since we are interested in model checking a $\mathsf{QCTL}^*_{\mathrm{ii}}$ formula φ on a CKS $\mathcal{S}$, the automaton that we build for φ can depend on $\mathcal{S}$. It can thus guess paths in $\mathcal{S}$, and evaluate free occurrences of atomic propositions in $\mathcal{S}$ without reading the input tree. The input tree is thus no longer used to represent the model. However we use it to carry labellings for quantified propositions $\mathrm{AP}_\exists(\varphi)$: we provide the automaton with an input tree whose labelling is initially empty, and the automaton, through successive narrowing and projection operations, decorates it with uniform labellings for quantified atomic propositions.

We remark that this technique allows one to go beyond CL [19]: by separating between quantified atomic propositions (that need to be uniform) and free atomic propositions (that state facts about the model), we manage to remove the restriction present in CL, that requires that all facts about the model are known to every strategy/agent (see the Appendix).

To do this we assume without loss of generality that propositions that are quantified in φ do not appear free in φ, *i.e.*, $\mathrm{AP}_\exists(\varphi) \cap \mathrm{AP}_f(\varphi) = \emptyset$. If necessary, for every $p \in \mathrm{AP}_\exists(\varphi) \cap \mathrm{AP}_f(\varphi)$, we take a fresh atomic proposition p' and replace all quantified occurrences of p in φ with p'. We obtain an equivalent formula φ' on $\mathrm{AP}' := \mathrm{AP} \cup \{p' \mid p \in \mathrm{AP}_\exists(\varphi) \cap \mathrm{AP}_f(\varphi)\}$ such that $\mathrm{AP}_\exists(\varphi') \cap \mathrm{AP}_f(\varphi') = \emptyset$. Observe also that given a formula φ such that $\mathrm{AP}_\exists(\varphi) \cap \mathrm{AP}_f(\varphi) = \emptyset$, a CKS $\mathcal{S}$ and a state $s \in \mathcal{S}$, the truth value of φ in $\mathcal{S}, s$ does not depend on the labelling of $\mathcal{S}$ for atomic propositions in $\mathrm{AP}_\exists(\varphi)$, which can thus be forgotten.

As a consequence, henceforth we assume that an instance $(\varphi, \mathcal{S})$ of the model-checking problem for $\mathsf{QCTL}^*_{\mathrm{ii}}$ is such that $\mathrm{AP}_\exists(\varphi) \cap \mathrm{AP}_f(\varphi) = \emptyset$, and $\mathcal{S}$ is a CKS over $\mathrm{AP}(\varphi)$.

Merging the decorated input tree and the model. To state the correctness of our construction, we will need to merge the labels for quantified propositions, carried by the input tree, with those for free propositions, carried by CKS $\mathcal{S}$. Because, through successive widenings, the input tree (represented by t in the definition below) will necessarily be a complete tree, its domain will always contain the domain of the unfolding of $\mathcal{S}$ (represented by t' below), hence the following definition.

Definition 8 (Merge)**.** *Let $t = (\tau, \ell)$ be a complete (AP, X)-tree and $t' = (\tau', \ell')$ an (AP', X)-tree, where $\mathrm{AP} \cap \mathrm{AP}' = \emptyset$. We define the* merge *of t and t' as the $(\mathrm{AP} \cup \mathrm{AP}', X)$-tree $t \between t' := (\tau \cap \tau' = \tau', \ell'')$, where $\ell''(u) = \ell(u) \cup \ell'(u)$.*

We now describe our automata construction and establish the following lemma, which is our main technical contribution.

Lemma 1 (Translation)**.** *Let $(\Phi, \mathcal{S})$ be an instance of the model-checking problem for $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$. For every subformula φ of Φ and state s of $\mathcal{S}$, one can build an ATA $\mathcal{A}^\varphi_s$ on $(\mathrm{AP}_\exists(\Phi), L_\varphi)$-trees such that for every $(\mathrm{AP}_\exists(\Phi), L_\varphi)$-tree t rooted in $s{\downarrow_{I_\varphi}}$,*

$$t \in \mathcal{L}(\mathcal{A}^\varphi_s) \quad \textit{iff} \quad t{\uparrow_y^{[n]}} \between t_{\mathcal{S}}(s) \models \varphi, \quad \textit{where } y = s{\downarrow_{[n]\setminus I_\varphi}}.$$

For an L_I-tree t, from now on $t{\uparrow^{[n]}} \between t_{\mathcal{S}}(s)$ stands for $t{\uparrow_y^{[n]}} \between t_{\mathcal{S}}(s)$, where $y = s{\downarrow_{[n]\setminus I}}$: the missing local states in the root of t are filled with those from s.

Proof sketch of Lemma 1. Let $(\Phi, \mathcal{S})$ be an instance of the model-checking problem for $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$. Let $\Phi \in \mathsf{QCTL}^*_{\mathrm{i},\subseteq}$, and let $\mathrm{AP}_\exists = \mathrm{AP}_\exists(\Phi)$ and $\mathrm{AP}_f = \mathrm{AP}_f(\Phi)$, and recall that $\mathcal{S}$ is labelled over AP_f. For each state $s \in S$ and each subformula

φ of Φ (note that all subformulas of Φ are also hierarchical), we define by induction on φ the ATA $\mathcal{A}_s^\varphi$ on $(\mathrm{AP}_\exists, L_\varphi)$-trees.

$\varphi = p$:

We let $\mathcal{A}_s^p$ be the ATA over $L_{[n]}$-trees with one unique state q_ι, with transition function defined as follows:

$$\delta(q_\iota, a) = \begin{cases} & p \in \mathrm{AP}_f \text{ and } p \in \ell_\mathcal{S}(s) \\ \top \quad \text{if} & \text{or} \\ & p \in \mathrm{AP}_\exists \text{ and } p \in a \\ & p \in \mathrm{AP}_f \text{ and } p \notin \ell_\mathcal{S}(s) \\ \bot \quad \text{if} & \text{or} \\ & p \in \mathrm{AP}_\exists \text{ and } p \notin a \end{cases}$$

$\varphi = \neg\varphi'$:

We obtain $\mathcal{A}_s^\varphi$ by dualising $\mathcal{A}_s^{\varphi'}$, a classic operation.

$\varphi = \varphi_1 \vee \varphi_2$:

Because $I_\varphi = I_{\varphi_1} \cap I_{\varphi_2}$, and each $\mathcal{A}_s^{\varphi_i}$ works on L_{φ_i}-trees, we first narrow them so that they work on L_φ-trees: for $i \in \{1, 2\}$, we let $\mathcal{A}_i := \mathcal{A}_s^{\varphi_i}{\downarrow_{I_\varphi}}$. We then build $\mathcal{A}_s^\varphi$ by taking the disjoint union of $\mathcal{A}_1$ and $\mathcal{A}_2$ and adding a new initial state that nondeterministically chooses which of $\mathcal{A}_1$ or $\mathcal{A}_2$ to execute on the input tree, so that $\mathcal{L}(\mathcal{A}_s^\varphi) = \mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)$.

$\varphi = \mathbf{E}\psi$:

The aim is to build an automaton $\mathcal{A}_s^\varphi$ that works on L_φ-trees and that on input t, checks for the existence of a path in $t{\uparrow^{[n]}} \between t_\mathcal{S}(s)$ that satisfies ψ. Observe that a path in $t{\uparrow^{[n]}} \between t_\mathcal{S}(s)$ is a path in $t_\mathcal{S}(s)$, augmented with the labelling for atomic propositions in $\mathrm{AP}_\exists$ carried by t.

To do so, $\mathcal{A}_s^\varphi$ guesses a path λ in $(\mathcal{S}, s)$. It remembers the current state in $\mathcal{S}$, which provides the labelling for atomic propositions in AP_f, and while it guesses λ it follows its L_φ-narrowing in its input tree t (which always exists since input to tree automata are complete trees), reading the labels to evaluate propositions in $\mathrm{AP}_\exists$.

Let $\max(\psi) = \{\varphi_1, \ldots, \varphi_n\}$ be the set of maximal state sub-formulas of ψ. In a first step we see these maximal state sub-formulas as atomic propositions. The formula ψ can thus be seen as an LTL formula, and we can build a nondeterministic parity word automaton $\mathcal{W}^\psi = (Q^\psi, \Delta^\psi, q_\iota^\psi, C^\psi)$ over alphabet $2^{\max(\psi)}$ that accepts exactly the models of ψ [35]. [3] We define the ATA $\mathcal{A}$ that, given as input a $(\max(\psi), L_\varphi)$-tree t, nondeterministically guesses a path λ in $t{\uparrow^{[n]}} \between t_\mathcal{S}(s)$ and simulates $\mathcal{W}^\psi$ on it, assuming that the labels it reads while following $\lambda{\downarrow_{I_\varphi}}$ in its input correctly represent the truth value of formulas in $\max(\psi)$ along λ. Recall that $\mathcal{S} = (S, R, s_\iota, \ell_\mathcal{S})$; we define $\mathcal{A} := (Q, \delta, q_\iota, C)$, where

- $Q = Q^\psi \times S$,
- $q_\iota = (q_\iota^\psi, s)$,
- $C(q^\psi, s') = C^\psi(q^\psi)$, and
- for each $(q^\psi, s') \in Q$ and $a \in 2^{\max(\psi)}$,

$$\delta((q^\psi, s'), a) = \bigvee_{q' \in \Delta^\psi(q^\psi, a)} \bigvee_{s'' \in R(s')} [s''{\downarrow_{I_\varphi}}, (q', s'')].$$

The intuition is that $\mathcal{A}$ reads the current label, chooses nondeterministically which transition to take in $\mathcal{W}^\psi$, chooses a next state in $\mathcal{S}$ and proceeds in the corresponding direction in X_φ. Thus, $\mathcal{A}$ accepts a $\max(\varphi)$-labelled L_φ-tree t iff there is a path in t that is the L_φ-narrowing of some path in $t_\mathcal{S}(s)$, and that satisfies ψ, where maximal state formulas are considered as atomic propositions.

Now from $\mathcal{A}$ we build the automaton $\mathcal{A}_s^\varphi$ over L_φ-trees labelled with "real" atomic propositions in $\mathrm{AP}_\exists$. In each node it visits, $\mathcal{A}_s^\varphi$ guesses what should be its labelling over $\max(\psi)$, it simulates $\mathcal{A}$ accordingly, and checks that the guess it made is correct. If the path being guessed in $t_\mathcal{S}(s)$ is currently in node u ending with state s', and $\mathcal{A}_s^\varphi$ guesses that φ_i holds in u, it checks this guess by starting a simulation of automaton $\mathcal{A}_{s'}^{\varphi_i}$ from node $v = u{\downarrow_{I_\varphi}}$ in its input t.

For each $s' \in \mathcal{S}$ and each $\varphi_i \in \max(\psi)$ we first build $\mathcal{A}_{s'}^{\varphi_i}$, and we let $\mathcal{A}_{s'}^i := \mathcal{A}_{s'}^{\varphi_i} = (Q_{s'}^i, \delta_{s'}^i, q_{s'}^i, C_{s'}^i)$. We also let $\overline{\mathcal{A}_{s'}^i} = (\overline{Q_{s'}^i}, \overline{\delta_{s'}^i}, \overline{q_{s'}^i}, \overline{C_{s'}^i})$ be its dualisation, and we assume w.l.o.g. that all the state sets are pairwise disjoint. Observe that because each φ_i is a maximal state sub-formula, we have $I_{\varphi_i} = I_\varphi$, so that we do not need to narrow down automata $\mathcal{A}_{s'}^i$ and $\overline{\mathcal{A}_{s'}^i}$. We define the ATA

$$\mathcal{A}_s^\varphi = (Q \cup \bigcup_{i,s'} Q_{s'}^i \cup \overline{Q_{s'}^i}, \delta', q_\iota, C'),$$

where the colours of states are left as they were in their original automaton, and δ is defined as follows. For states in $Q_{s'}^i$ (resp. $\overline{Q_{s'}^i}$), δ agrees with $\delta_{s'}^i$ (resp. $\overline{\delta_{s'}^i}$), and for $(q^\psi, s') \in Q$ and $a \in 2^{\mathrm{AP}_\exists}$ we let $\delta'((q^\psi, s'), a)$ be the disjunction over $a' \in 2^{\max(\psi)}$ of

$$\left(\delta\left((q^\psi, s'), a'\right) \wedge \bigwedge_{\varphi_i \in a'} \delta_{s'}^i(q_{s'}^i, a) \wedge \bigwedge_{\varphi_i \notin a'} \overline{\delta_{s'}^i}(\overline{q_{s'}^i}, a)\right).$$

$\varphi = \exists^{\mathbf{o}} p.\, \varphi'$:

We build automaton $\mathcal{A}_s^{\varphi'}$ that works on $L_{\varphi'}$-trees; because φ is hierarchical, we have that $\mathbf{o} \subseteq I_{\varphi'}$ and we can narrow down $\mathcal{A}_s^{\varphi'}$ to work on $L_\mathbf{o}$-trees and obtain $\mathcal{A}_1 := \mathcal{A}_s^{\varphi'}{\downarrow_\mathbf{o}}$. By Theorem 6 we can nondeterminise it to get $\mathcal{A}_2$, which by Theorem 5 we can project with respect to p, finally obtaining $\mathcal{A}_s^\varphi := \mathcal{A}_2{\Downarrow_p}$.

The proof that the construction is correct can be found in the Appendix. □

C. Proof of Theorem 4

We can now prove Theorem 4. Let $\mathcal{S}$ be a CKS, $s \in \mathcal{S}$, and $\varphi \in \mathsf{QCTL}^*_{\mathsf{i},\subseteq}$. By Lemma 1 one can build an ATA $\mathcal{A}_s^\varphi$ such that for every labelled L_φ-tree t rooted in $s{\downarrow_{I_\varphi}}$, it holds that $t \in \mathcal{L}(\mathcal{A}_s^\varphi)$ iff $t \uparrow^{[n]} \between t_\mathcal{S}(s) \models \varphi$. Let τ be the full L_φ-tree rooted in $s{\downarrow_\varphi}$, and let $t = (\tau, \ell_\emptyset)$, where $\ell_\emptyset$ is the empty labelling. Clearly, $t{\uparrow^{[n]}} \between t_\mathcal{S}(s) = t_\mathcal{S}(s)$, and because t is rooted in $s{\downarrow_\varphi}$, we have $t \in \mathcal{L}(\mathcal{A}_s^\varphi)$ iff $t_\mathcal{S}(s) \models \varphi$. It

[3] Note that, as usual for nondeterministic word automata, we take the transition function of type $\Delta^\psi : Q^\psi \times 2^{\max(\psi)} \to 2^{Q^\psi}$.

only remains to build a simple deterministic tree automaton $\mathcal{A}$ over L_φ-trees such that $\mathcal{L}(\mathcal{A}) = \{t\}$, and check for emptiness of the alternating tree automaton $\mathcal{L}(\mathcal{A} \cap \mathcal{A}^\varphi_s)$. Because nondeterminisation makes the size of the automaton gain one exponential for each nested quantifier on propositions, the procedure is nonelementary, and hardness is inherited from the model-checking problem for QCTL [14].

V. Model-checking hierarchical instances of SL_{ii}

In this section we establish that the model-checking problem for SL_{ii} restricted to the class of hierarchical instances is decidable (Theorem 2).

We build upon the proof in [22] that establishes the decidability of the model-checking problem for $\mathsf{ATL}^*_{\text{sc}}$ by reduction to the model-checking problem for QCTL^*. The main difference is that we reduce to the model-checking problem for $\mathsf{QCTL}^*_{\text{ii}}$ instead, using quantifiers on atomic propositions parameterised with observations that reflect the ones used in the SL_{ii} model-checking instance.

Let $(\Phi, \mathcal{G})$ be a hierarchical instance of the SL_{ii} model-checking problem, where $\mathcal{G} = (\text{Ac}, V, E, \ell, v_\iota, \mathcal{O})$. We will first show how to define a CKS $\mathcal{S}_\mathcal{G}$ and a bijection $\rho \mapsto u_\rho$ between the set of finite plays ρ starting in a given position v and the set of nodes in $t_{\mathcal{S}_\mathcal{G}}(s_v)$.

Then, for every subformula φ of Φ and partial function $f : \text{Ag} \rightharpoonup \text{Var}$, we will define a $\mathsf{QCTL}^*_{\text{ii}}$ formula $(\varphi)^f$ (that will also depend on $\mathcal{G}$) such that the following holds:

Proposition 5. *Suppose that* $\mathit{free}(\varphi) \cap \text{Ag} \subseteq \mathit{dom}(f)$*, and* $f(a) = x$ *implies* $\chi(a) = \chi(x)$ *for all* $a \in \mathit{dom}(f)$*. Then*

$$\mathcal{G}, \chi, \rho \models \varphi \quad \textit{if and only if} \quad t_{\mathcal{S}_\mathcal{G}}(s_\rho) \models (\varphi)^f.$$

Applying this to the sentence Φ, any assignment χ, and the empty function $\emptyset$, we get that

$$\mathcal{G}, \chi, v_\iota \models \Phi \quad \text{if and only if} \quad t_{\mathcal{S}_\mathcal{G}}(s_{v_\iota}) \models (\Phi)^\emptyset.$$

Constructing the CKS $\mathcal{S}_\mathcal{G}$. We will define $\mathcal{S}_\mathcal{G}$ so that (indistinguishable) nodes in its tree-unfolding correspond to (indistinguishable) finite plays in $\mathcal{G}$. The CKS will make use of atomic propositions $\text{AP}_v := \{p_v \mid v \in V\}$ (that we assume to be disjoint from AP). The idea is that p_v allows the $\mathsf{QCTL}^*_{\text{ii}}$ formula $(\Phi)^\emptyset$ to refer to the current position v in $\mathcal{G}$. Later we will see that $(\Phi)^\emptyset$ will also make use of atomic propositions $\text{AP}_c := \{p^x_c \mid c \in \text{Ac} \text{ and } x \in \text{Var}\}$ that we assume, again, are disjoint from $\text{AP} \cup \text{AP}_v$. This allows the formula to use p^x_c to refer to the actions c advised by strategies x.

Suppose $\text{Obs} = \{o_1, \dots, o_n\}$. For $i \in [n]$, define the local states $L_i := \{[v]_{o_i} \mid v \in V\}$ where $[v]_o$ is the equivalence class of v for relation $\sim_o$. Since we need to know the actual position of the CGS_{ii} to define the dynamics, we also let $L_{n+1} := V$.

Define the CKS $\mathcal{S}_\mathcal{G} := (S, R, s_\iota, \ell')$ where

- $S := \{s_v \mid v \in V\}$,
- $R := \{(s_v, s_{v'}) \mid \exists \boldsymbol{c} \in \text{Ac}^{\text{Ag}} \text{ s.t. } E(v, \boldsymbol{c}) = v'\} \subseteq S^2$,
- $s_\iota := s_{v_\iota}$,
- $\ell'(s_v) := \ell(v) \cup \{p_v\} \subseteq \text{AP} \cup \text{AP}_v$,

and $s_v := ([v]_{o_1}, \dots, [v]_{o_n}, v) \in \prod_{i \in [n+1]} L_i$.

We now show how to connect finite plays in $\mathcal{G}$ with nodes in the tree unfolding of $\mathcal{S}_\mathcal{G}$. For every finite play $\rho = v_0 \dots v_k$, define the node $u_\rho := s_{v_0} \dots s_{v_k}$ in $t_{\mathcal{S}_\mathcal{G}}(s_{v_0})$ (which exists, by definition of $\mathcal{S}_\mathcal{G}$ and of tree unfoldings). Note that the mapping $\rho \mapsto u_\rho$ defines a bijection between the set of finite plays and the set of nodes in $t_{\mathcal{S}_\mathcal{G}}(s_\iota)$.

Constructing the $\mathsf{QCTL}^*_{\text{i},\subseteq}$ formulas $(\varphi)^f$. We now describe how to transform an SL_{ii} formula φ and a partial function $f : \text{Ag} \rightharpoonup \text{Var}$ into a $\mathsf{QCTL}^*_{\text{ii}}$ formula $(\varphi)^f$ (that will also depend on $\mathcal{G}$). Suppose that $\text{Ac} = \{c_1, \dots, c_l\}$, and define $(\varphi)^f$ by induction on φ. We begin with the simple cases: $(p)^f := p$; $(\neg\varphi)^f := \neg(\varphi)^f$; and $(\varphi_1 \vee \varphi_2)^f := (\varphi_1)^f \vee (\varphi_2)^f$.

We continue with the second-order quantifier case:

$$(\langle\!\langle x \rangle\!\rangle^o \varphi)^f := \exists^{\widetilde{o}} p^x_{c_1} \dots \exists^{\widetilde{o}} p^x_{c_l} . \varphi_{\text{str}}(x) \wedge (\varphi)^f$$

where $\widetilde{o_i} := \{j \mid \mathcal{O}(o_i) \subseteq \mathcal{O}(o_j)\}$, and

$$\varphi_{\text{str}}(x) := \mathbf{AG} \bigvee_{c \in \text{Ac}} (p^x_c \wedge \bigwedge_{c' \neq c} \neg p^x_{c'}).$$

We describe this formula in words. For each possible action $c \in \text{Ac}$, an existential quantification on the atomic proposition p^x_c "chooses" for each finite play $\rho = v_0 \dots v_k$ of $\mathcal{G}$ (or, equivalently, for each node u_ρ of the tree $t_{\mathcal{S}_\mathcal{G}}(s_{v_0})$) whether strategy x plays action c in ρ or not. Formula $\varphi_{\text{str}}(x)$ ensures that in each finite play, exactly one action is chosen for strategy x, and thus that atomic propositions p^x_c indeed characterise a strategy, call it σ_x.[4]

Moreover, a quantifier with concrete observation $\widetilde{o_i}$ receives information corresponding to observation o_i (observe that for all $i \in [n]$, $i \in \widetilde{o_i}$) as well as information corresponding to coarser observations. Note that including all coarser observations does not increase the information accessible to the quantifier: indeed, one can show that two nodes are $\{i\}$-indistinguishable if and only if they are $\widetilde{o_i}$-indistinguishable. However, this definition of $\widetilde{o_i}$ allows us to obtain hierarchical formulas. Since quantification on propositions p^x_c is done uniformly with regards to concrete observation $\widetilde{o_i}$, it follows that σ_x is an o_i-strategy.

Here are the remaining cases:

$$\begin{aligned}
((a,x)\varphi)^f &:= (\varphi)^{f[a \mapsto x]} \\
(\mathbf{X}\varphi_1)^f &:= \mathbf{A}\big(\psi_{\text{out}}(f) \to \mathbf{X}(\varphi_1)^f\big) \\
(\varphi_1 \mathbf{U} \varphi_2)^f &:= \mathbf{A}\big(\psi_{\text{out}}(f) \to (\varphi_1)^f \mathbf{U} (\varphi_2)^f\big)
\end{aligned}$$

where

$$\psi_{\text{out}}(f) := \mathbf{G}\left(\bigwedge_{v \in V} \bigwedge_{\boldsymbol{c} \in \text{Ac}^{\text{Ag}}} \left(p_v \wedge \bigwedge_{a \in \text{Ag}} p^{f(a)}_{\boldsymbol{c}_a} \to \mathbf{X} p_{E(v,\boldsymbol{c})}\right)\right).$$

The formula $\psi_{\text{out}}(f)$ is used to select the unique path assuming that every player, say a, follows the strategy $\sigma_{f(a)}$.

This completes the justification of Proposition 5.

[4] More precisely, if $\varphi_{\text{str}}(x)$ holds in node u_ρ, it ensures that propositions from AP_c define a partial strategy, defined on all nodes of the subtree rooted in u_ρ. This is enough because SL_{ii} can only talk about the future: when evaluating a formula in a finite play ρ, the definition of strategies on plays that do not start with ρ is irrelevant.

Preserving hierarchy. To complete the proof we show that $(\Phi)^{\emptyset}$ is a hierarchical $\mathsf{QCTL}^{*}_{\mathsf{ii}}$ formula. This simply follows from the fact that Φ is hierarchical in $\mathcal{G}$ and that for every two observations o_i and o_j in Obs such that $\mathcal{O}(o_i) \subseteq \mathcal{O}(o_j)$, by definition of $\widetilde{o_k}$ we have that $\widetilde{o_i} \subseteq \widetilde{o_j}$.

This completes the proof of Theorem 2.

VI. Outlook

We introduced $\mathsf{SL}_{\mathsf{ii}}$, a logic for reasoning about strategic behaviour in multi-player games with imperfect information. The syntax mentions observations, and thus allows one to write formulas that talk about dynamic observations. We isolated the class of hierarchical formula/model pairs $\Phi, \mathcal{G}$ and proved that one can decide if $\mathcal{G} \models \Phi$. The proof reduces (hierarchical) instances to (hierarchical) formulas of $\mathsf{QCTL}^{*}_{\mathsf{ii}}$, a low-level logic that we introduced, and that serves as a natural bridge between $\mathsf{SL}_{\mathsf{ii}}$ (that talks about players and strategies) and automata constructions. We believe that $\mathsf{QCTL}^{*}_{\mathsf{ii}}$ is of independent interest and deserves study in its own right.

Since one can alternate quantifiers in $\mathsf{SL}_{\mathsf{ii}}$, our decidability result goes beyond synthesis. As we showed, we can use it to decide if a game that yields hierarchical observation has a Nash equilibrium. A crude but easy analysis of our main decision procedure shows it is non-elementary.

This naturally leads to a number of avenues for future work: define and study the expressive power and computational complexity of fragments of $\mathsf{SL}_{\mathsf{ii}}$ [36]; adapt the notion of hierarchical instances to allow for situations in which hierarchies can change infinitely often along a play [13]; and extend the logic to include epistemic operators for individual and common knowledge, as is done in [37], which are important for reasoning about distributed systems [21].

References

[1] F. Mogavero, A. Murano, G. Perelli, and M. Y. Vardi, "Reasoning about strategies: On the model-checking problem," *TOCL*, vol. 15, no. 4, pp. 34:1–34:47, 2014.

[2] N. Bulling and W. Jamroga, "Comparing variants of strategic ability: how uncertainty and memory influence general properties of games," *AAMAS'14*, vol. 28, no. 3, pp. 474–518, 2014.

[3] F. Laroussinie, N. Markey, and A. Sangnier, "ATLsc with partial observation," in *GandALF'15*, 2015, pp. 43–57.

[4] C. Dima and F. L. Tiplea, "Model-checking ATL under imperfect information and perfect recall semantics is undecidable," *CoRR*, vol. abs/1102.4225, 2011.

[5] A. Pnueli and R. Rosner, "Distributed reactive systems are hard to synthesize," in *FOCS'90*, 1990, pp. 746–757.

[6] G. L. Peterson and J. H. Reif, "Multiple-person alternation," in *SFCS'79*, 1979, pp. 348–363.

[7] O. Kupermann and M. Vardi, "Synthesizing distributed systems," in *LICS'01*, 2001, pp. 389–398.

[8] G. Peterson, J. Reif, and S. Azhar, "Decision algorithms for multiplayer noncooperative games of incomplete information," *CAMWA*, vol. 43, no. 1, pp. 179–206, 2002.

[9] B. Finkbeiner and S. Schewe, "Uniform distributed synthesis," in *LICS'05*, 2005, pp. 321–330.

[10] S. Pinchinat and S. Riedweg, "A decidable class of problems for control under partial observation," *IPL*, vol. 95, no. 4, pp. 454–460, 2005.

[11] S. Schewe and B. Finkbeiner, "Distributed synthesis for alternating-time logics," in *ATVA'07*, 2007, pp. 268–283.

[12] K. Chatterjee and L. Doyen, "Games with a weak adversary," in *ICALP'14*, 2014, pp. 110–121.

[13] D. Berwanger, A. B. Mathew, and M. Van den Bogaard, "Hierarchical information patterns and distributed strategy synthesis," in *ATVA'15*, 2015, pp. 378–393.

[14] F. Laroussinie and N. Markey, "Quantified CTL: expressiveness and complexity," *LMCS*, vol. 10, no. 4, 2014.

[15] G. Peterson, J. Reif, and S. Azhar, "Lower bounds for multiplayer noncooperative games of incomplete information," *CAMWA*, vol. 41, no. 7, pp. 957–992, 2001.

[16] J. Y. Halpern and M. Y. Vardi, "The complexity of reasoning about knowledge and time. i. lower bounds," *JCSS*, vol. 38, no. 1, pp. 195–237, 1989.

[17] H. Läuchli and C. Savioz, "Monadic second order definable relations on the binary tree," *JSL*, vol. 52, no. 01, pp. 219–226, 1987.

[18] P. Gastin, N. Sznajder, and M. Zeitoun, "Distributed synthesis for well-connected architectures," *FMSD*, vol. 34, no. 3, pp. 215–237, 2009.

[19] B. Finkbeiner and S. Schewe, "Coordination logic," in *CSL'10*, 2010, pp. 305–319.

[20] R. Alur, T. A. Henzinger, and O. Kupferman, "Alternating-time temporal logic," *JACM*, vol. 49, no. 5, pp. 672–713, 2002.

[21] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi, *Reasoning about knowledge*. MIT press Cambridge, 1995, vol. 4.

[22] F. Laroussinie and N. Markey, "Augmenting ATL with strategy contexts," *IC*, vol. 245, pp. 98–123, 2015.

[23] A. Sistla, "Theoretical Issues in the Design and Cerification of Distributed Systems." Ph.D. dissertation, Harvard University, Cambridge, MA, USA, 1983.

[24] E. A. Emerson and A. P. Sistla, "Deciding branching time logic," in *STOC'84*, 1984, pp. 14–24.

[25] O. Kupferman, "Augmenting branching temporal logics with existential quantification over atomic propositions," in *CAV'95*, 1995, pp. 325–338.

[26] O. Kupferman, P. Madhusudan, P. S. Thiagarajan, and M. Y. Vardi, "Open systems in reactive environments: Control and synthesis," in *CONCUR'00*, 2000, pp. 92–107.

[27] T. French, "Decidability of quantifed propositional branching time logics," in *AJCAI'01*, 2001, pp. 165–176.

[28] C. C. Elgot and M. O. Rabin, "Decidability and undecidability of extensions of second (first) order theory of (generalized) successor," *JSL*, vol. 31, no. 2, pp. 169–181, 1966.

[29] W. Thomas, "Infinite trees and automaton-definable relations over omega-words," *TCS*, vol. 103, no. 1, pp. 143–159, 1992.

[30] O. Kupferman and M. Y. Vardi, "Church's problem revisited," *BSL*, pp. 245–263, 1999.

[31] D. E. Muller and P. E. Schupp, "Alternating automata on infinite trees," *TCS*, vol. 54, pp. 267–276, 1987.

[32] M. O. Rabin, "Decidability of second-order theories and automata on infinite trees," *TAMS*, vol. 141, pp. 1–35, 1969.

[33] D. E. Muller and P. E. Schupp, "Simulating alternating tree automata by nondeterministic automata: New results and new proofs of the theorems of Rabin, McNaughton and Safra," *TCS*, vol. 141, no. 1&2, pp. 69–107, 1995.

[34] O. Kupferman, M. Y. Vardi, and P. Wolper, "An automata-theoretic approach to branching-time model checking," *JACM*, vol. 47, no. 2, pp. 312–360, 2000.

[35] M. Y. Vardi and P. Wolper, "Reasoning about infinite computations," *IC*, vol. 115, no. 1, pp. 1–37, 1994.

[36] F. Mogavero, A. Murano, G. Perelli, and M. Y. Vardi, "What makes ATL* decidable? a decidable fragment of Strategy Logic," in *CONCUR'12*, 2012, pp. 193–208.

[37] P. Cermák, A. Lomuscio, F. Mogavero, and A. Murano, "MCMAS-SLK: A model checker for the verification of strategy logic specifications," in *CAV'14*, 2014, pp. 525–532.

[38] W. Zielonka, "Infinite games on finitely coloured graphs with applications to automata on infinite trees," *TCS*, vol. 200, no. 1-2, pp. 135–183, 1998.

APPENDIX

A. Strategy logic under imperfect information

We establish two facts about SL_{ii} mentioned in the body: 1) the statement $\mathcal{G}, \chi, \rho \models \varphi$ is independent of χ if φ is a sentence; 2) Proposition 1 which says that the perfect-information fragment of SL_{ii} is a notational variant of SL given in [1].

Truth of sentences are independent of the assignment.

We begin with a formal definition of free players and variables in SL_{ii}.

Definition 9 (Free players and variables)**.** *The set* $\mathit{free}(\varphi)$ *of* free players *and* free variables *of an* SL_{ii} *formula* φ *is defined as follows:*

- $\mathit{free}(p) := \emptyset$*, where* $p \in \text{AP}$.
- $\mathit{free}(\neg\varphi) := \mathit{free}(\varphi)$.
- $\mathit{free}(\varphi_1 \vee \varphi_2) := \mathit{free}(\varphi_1) \cup \mathit{free}(\varphi_2)$.
- $\mathit{free}(\mathbf{X}\varphi) := \text{Ag} \cup \mathit{free}(\varphi)$.
- $\mathit{free}(\varphi_1 \mathbf{U} \varphi_2) := \text{Ag} \cup \mathit{free}(\varphi_1) \cup \mathit{free}(\varphi_2)$.
- $\mathit{free}(\langle\!\langle x \rangle\!\rangle^o \varphi) := \mathit{free}(\varphi) \setminus \{x\}$.
- $\mathit{free}((a,x)\varphi) := \mathit{free}(\varphi)$*, if* $a \notin \mathit{free}(\varphi)$.
- $\mathit{free}((a,x)\varphi) := (\mathit{free}(\varphi) \setminus \{a\}) \cup \{x\}$*, if* $a \in \mathit{free}(\varphi)$.

Lemma 2. *For all CGS*$_{\text{ii}}$ $\mathcal{G}$*, finite plays* ρ*, assignments* χ, χ'*, and* SL_{ii} *sentences* Φ*,*

$$\mathcal{G}, \chi, \rho \models \Phi \quad \textit{iff} \quad \mathcal{G}, \chi', \rho \models \Phi.$$

Proof. Since a sentence has no free variables or players, it is enough to prove the following for formulas φ: if χ and χ' agree on the free variables and players of φ, then $\mathcal{G}, \chi, \rho \models \varphi$ iff $\mathcal{G}, \chi', \rho \models \varphi$. The proof is by induction on φ.

The case of an atom is immediate since the semantics do not depend on the assignment.

The case of negation follows immediately from the inductive hypothesis using the fact that $\mathit{free}(\neg\varphi) = \mathit{free}(\varphi)$. The case of disjunction is similar.

For the quantifier case $\langle\!\langle x \rangle\!\rangle^o \varphi$ use the fact that $\chi[x \mapsto \sigma]$ and $\chi'[x \mapsto \sigma]$ agree on the free placeholders of φ (since we assumed that χ, χ' agree on the free placeholders of $\langle\!\langle x \rangle\!\rangle^o \varphi$).

For the binding case $(a,x)\varphi$ use the fact that $\chi[a \mapsto \chi(x)]$ and $\chi'[a \mapsto \chi'(x)]$ agree on the free placeholders of φ.

For the next-operator case $\mathbf{X}\varphi$ use the fact that $\text{out}(\chi, \rho) = \text{out}(\chi', \rho)$ (since out only depends on the strategies assigned to players), and χ, χ' agree on $\mathit{free}(\varphi)$ since $\mathit{free}(\varphi) \subseteq \mathit{free}(\mathbf{X}\varphi)$. The case of $\mathbf{U}$ is similar. □

1) Comparison of the perfecti-information fragment of SL_{ii} *with the definition of* SL *from [1]:* We recall the definitions of the syntax and semantics of SL.

Definition 10 (SL syntax [1])**.** *The syntax of* SL *is defined by the following grammar:*

$$\varphi := p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U} \varphi \mid \langle\!\langle x \rangle\!\rangle \varphi \mid (a,x)\varphi$$

where $p \in \text{AP}$*,* $x \in \text{Var}$*, and* $a \in \text{Ag}$.

An *assignment* is a partial function $\chi : \text{Ag} \cup \text{Var} \rightharpoonup \mathit{Str}$, assigning to each player and variable in its domain a strategy. For an assignment χ, a player a and a strategy σ, $\chi[a \mapsto \sigma]$ is the assignment of domain $\mathit{dom}(\chi) \cup \{a\}$ that maps a to σ and is equal to χ on the rest of its domain, and $\chi[x \mapsto \sigma]$ is defined similarly, where x is a variable. An assignment χ is *player-complete for* $\mathcal{G}$, or simply *player-complete* when $\mathcal{G}$ is clear from the context, if it assigns a strategy to each player in $\mathcal{G}$, *i.e.*, $\text{Ag} \subseteq \mathit{dom}(\chi)$. For a player-complete assignment χ and a position v, let $\text{out}(\chi, v)$ be the infinite play obtained when the game starts in v and all players follow the strategies assigned by χ: $\text{out}(\chi, v) := v_0 v_1 \dots$ with $v_0 = v$ and for each $i \geq 0$, $v_{i+1} = E(v_i, \boldsymbol{c})$, where $\boldsymbol{c} = (\chi(a)(v_0 \dots v_i))_{a \in \text{Ag}}$.

In addition, given a formula $\varphi \in \mathsf{SL}$, an assignment is *variable-complete for* φ, or simply *variable-complete* when φ is clear from the context, if $\mathit{free}(\varphi) \cap \text{Var} \subseteq \mathit{dom}(\chi)$.

Given an assignment χ and an initial play ρ, define the ρ-translation of χ as the assignment χ^ρ such that $\mathit{dom}(\chi^\rho) = \mathit{dom}(\chi)$, and for every $l \in \mathit{dom}(\chi^\rho)$, $\chi^\rho(l) = \chi(l)^\rho$. We also define the *global translation* of a complete assignment χ together with a position v as follows: for every $i \in \mathbb{N}$, the i-*global translation* of (χ, v) is defined as $(\chi, v)^i := (\chi^{\pi \leq i}, \pi_i)$, where $\pi = \text{out}(\chi, v)$.

Models of SL formulas are concurrent games structures (CGS), i.e., tuples $\mathcal{G} = (\text{Ac}, V, E, \ell, v_\iota)$ where the entries are as for CGS_{ii} (but without $\mathcal{O}\mathit{bs}$ and $\mathcal{O}$).

Definition 11 (SL Semantics [1])**.** *Let* φ *be an* SL*-formula. The semantics of* φ *in a CGS* $\mathcal{G}$ *at position* $v \in \mathcal{G}$ *with a variable-complete assignment* χ *is defined inductively as follows:*

$$\begin{array}{lll}
\mathcal{G}, \chi, v \models p & \textit{if} & p \in \ell(v) \\
\mathcal{G}, \chi, v \models \neg\varphi & \textit{if} & \mathcal{G}, \chi, v \not\models \varphi \\
\mathcal{G}, \chi, v \models \varphi \vee \varphi' & \textit{if} & \mathcal{G}, \chi, v \models \varphi \textit{ or } \mathcal{G}, \chi, v \models \varphi' \\
\mathcal{G}, \chi, v \models \langle\!\langle x \rangle\!\rangle \varphi & \textit{if} & \exists\, \sigma \in \text{Str} \textit{ s.t. } \mathcal{G}, \chi[x \mapsto \sigma], v \models \varphi \\
\mathcal{G}, \chi, v \models (a,x)\varphi & \textit{if} & \mathcal{G}, \chi[a \mapsto \chi(x)], v \models \varphi
\end{array}$$

If, in addition, χ *is player-complete, then*

$$\begin{array}{lll}
\mathcal{G}, \chi, v \models \mathbf{X}\varphi & \textit{if} & \mathcal{G}, (\chi, v)^1 \models \varphi \\
\mathcal{G}, \chi, v \models \varphi \mathbf{U} \varphi' & \textit{if} & \exists\, i \geq 0 \textit{ s.t. } \mathcal{G}, (\chi, v)^i \models \varphi' \textit{ and} \\
 & & \forall\, j \textit{ s.t. } 0 \leq j < i,\ \mathcal{G}, (\chi, v)^j \models \varphi
\end{array}$$

For a sentence $\varphi \in \mathsf{SL}$ define $\mathcal{G}, v \models \varphi$ if $\mathcal{G}, \emptyset, v \models \varphi$ (where $\emptyset$ is the empty assignment), and write $\mathcal{G} \models \varphi$ if $\mathcal{G}, v_\iota \models \varphi$. This completes the recap of the syntax and semantics of SL.

Proof of proposition 1. We prove this in two steps. First, note that in the definition of SL, one can restrict to complete assignments when defining $\models$. Indeed, an easy induction shows that for all partial assignments χ, χ' such that χ' extends χ (i.e., $\mathit{dom}(\chi) \subseteq \mathit{dom}(\chi)$ and $\chi(l) = \chi'(l)$ for all $l \in \mathit{dom}(\chi)$), we have that $\mathcal{G}, \chi, v \models \varphi$ iff $\mathcal{G}, \chi', v \models \varphi$. Thus, from now on we assume that all assignments of SL are complete. In addition, in [1] the authors define $\mathcal{G}, v \models \varphi$, for φ a sentence, as $\mathcal{G}, \emptyset, v \models \varphi$ where $\emptyset$ is the empty assignment. This is equivalent to stating that $\mathcal{G}, \chi, v \models \varphi$ for all complete assignments χ.

To prove proposition 1, let φ be an SL formula. Fix an observation symbol o and let $\text{Obs} := \{o\}$. Define the SL_{ii}

formula φ' by annotating every strategy quantifier in φ by o, i.e., by replacing $\langle\langle x \rangle\rangle$ by $\langle\langle x \rangle\rangle^o$. Let $\mathcal{G}$ be a CGS. Define the CGS$_{\text{ii}}$ $\mathcal{G}'$ to be $(\mathcal{G}, \mathcal{O}bs, \mathcal{O})$ where $\mathcal{O}(o) = \{(v, v) : v \in V\}$ is the identity observation. We now prove that for every complete assignment χ, and $v \in \mathcal{G}$, $\mathcal{G}, \chi, v \models \varphi$ iff $\mathcal{G}', \chi, v \models \varphi'$, which establishes the proposition when φ is a sentence. This follows by induction on φ using the following inductive hypothesis: For all $n \in \mathbb{N}$,

$$\mathcal{G}, (\chi, v)^n \models \varphi \text{ iff } \mathcal{G}', \chi, \pi^{\leq n} \models \varphi,$$

where $\pi = \text{out}(\chi, v)$. The inductive step is immediate from the definitions of the semantics of SL and SL$_{\text{ii}}$.

B. Translation of CL *into* SL$_{\text{ii}}$

In this section we prove Proposition 4 by giving two canonical translations: first from CL-instances into QCTL$^*_{\text{i},\subseteq}$-instances, and then translating QCTL$^*_{\text{ii}}$-instances into hierarchical SL$_{\text{ii}}$-instances.

We briefly recall the syntax and semantics of CL, and refer to [19] for further details. For definitions about trees, see Section III-B.

Notation for trees. Note that our definition for trees slightly differs from the one in [19], where the root is the empty word. Here we adopt this convention to keep closer to notations in [19]. Thus, (Y, X)-trees in CL are of the form (τ, l) where $\tau \subseteq X^*$ and $l : \tau \to 2^Y$.

For two disjoint sets X and Y, we identify $2^X \times 2^Y$ with $2^{X \cup Y}$. Let X and Y be two sets with $Z = X \cup Y$, and let M and N be two disjoint sets. Given an M-labelled 2^Z-tree $t = (\tau, \ell_M)$ and an N-labelled 2^Z-tree $t' = (\tau', \ell_N)$ with same domain $\tau = \tau'$, we define $t \uplus t' := (\tau, \ell')$, where for every $u \in \tau$, $\ell'(u) = \ell_M(u) \cup \ell_N(u)$. Now, given a complete M-labelled 2^X-tree $t = ((2^X)^*, \ell_M)$ and a complete N-labelled 2^Y-tree $t' = ((2^Y)^*, \ell_N)$, we define $t \oplus t' := t{\uparrow^{2^Z}_{2^X}} \uplus t'{\uparrow^{2^Z}_{2^Y}}$.

CL Syntax. Let $\mathcal{C}$ be a set of *coordination variables*, and let $\mathcal{S}$ be a set of *strategy variables* disjoint from $\mathcal{C}$. The syntax of CL is given by the following grammar:

$$\varphi ::= x \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U} \varphi \mid \exists C \exists s.\, \varphi$$

where $x \in \mathcal{C} \cup \mathcal{S}$, $C \subseteq \mathcal{C}$ and $s \in \mathcal{S}$, and with the restriction that each coordination variable appears in at most once *subtree quantifier* $\exists C \exists s.$, and similarly for strategy variables.

The notion of free and bound (coordination or strategy) variables is as usual. The set of free coordination variables in φ is noted $\mathcal{F}_\varphi$. A bound coordination variable c is *visible* to a strategy variable s if s is in the scope of the quantifier that introduces c, and $\mathit{Scope}_\varphi(s)$ is the union of the set of bound coordination variables visible to s and the set of free coordination variables (note that this union is disjoint). We will see, in the semantics, that the meaning of a bound strategy variable s is a strategy $f_s : (2^{\mathit{Scope}_\varphi(s)})^* \to 2^{\{s\}}$. Free strategy variables are called *atomic propositions*, and we denote the set of atomic propositions in φ by AP_φ.

CL Semantics. A CL formula φ is evaluated on a complete AP_φ-labelled $2^{\mathcal{F}_\varphi}$-tree t. An $(\text{AP}_\varphi, 2^{\mathcal{F}_\varphi})$-tree $t = (\tau, \ell)$ satisfies a CL formula φ if for every path λ that starts in the root we have $t, \lambda, 0 \models \varphi$, where the satisfaction of a formula at position $i \geq 0$ of a path λ is defined inductively as follows:

$$\begin{array}{lll}
t, \lambda, i \models p & \text{if} & p \in \ell(\lambda_i) \\
t, \lambda, i \models \neg\varphi' & \text{if} & t, \lambda, i \not\models \varphi' \\
t, \lambda, i \models \varphi_1 \vee \varphi_2 & \text{if} & t, \lambda, i \models \varphi_1 \text{ or } t, \lambda, i \models \varphi_2 \\
t, \lambda, i \models \mathbf{X}\varphi' & \text{if} & t, \lambda, i+1 \models \varphi' \\
t, \lambda, i \models \varphi_1 \mathbf{U} \varphi_2 & \text{if} & \exists\, j \geq i \text{ s.t. } t, \lambda, j \models \varphi_2 \text{ and} \\
 & & \forall k \text{ s.t. } i \leq k < j,\ t, \lambda, k \models \varphi_1 \\
t, \lambda, i \models \exists C \exists s.\, \varphi' & \text{if} & \exists\, f : (2^{\mathit{Scope}_\varphi(s)})^* \to 2^{\{s\}} \text{ s.t.} \\
 & & t_{\lambda_i} \oplus ((2^{\mathit{Scope}_\varphi(s)})^*, f) \models \varphi',
\end{array}$$

where t_{λ_i} is the subtree of t rooted in λ_i.

First, observe that in the last inductive case, t_{λ_i} being a $2^{\mathcal{F}_\varphi}$-tree, $t_{\lambda_i} \oplus ((2^{\mathit{Scope}_\varphi(s)})^*, f)$ is a $2^{\mathcal{F}_\varphi \cup \mathit{Scope}_\varphi(s)}$-tree. By definition, $\mathit{Scope}_\varphi(s) = \mathcal{F}_\varphi \cup C = \mathcal{F}_{\varphi'}$. It follows that $\mathcal{F}_\varphi \cup \mathit{Scope}_\varphi(s) = \mathit{Scope}_\varphi(s) = \mathcal{F}_{\varphi'}$, hence φ' is indeed evaluated on a $\mathcal{F}_{\varphi'}$-tree.

Remark 1. *Note that all strategies observe the value of all atomic propositions. Formally, for every* CL*-formula φ of the form $\varphi = \exists C_1 \exists s_1. \ldots, \exists C_i \exists s_i.\, \varphi'$ evaluated on a $2^{\mathcal{F}_\varphi}$-tree $t = (\tau, \ell)$, φ' is evaluated on a $2^{\mathcal{F}_\varphi \cup C_1 \cup \ldots \cup C_i}$-tree $t' = (\tau', \ell')$ such that for every $p \in \text{AP}_\varphi$, for every pair of nodes $u, u' \in t'$ such that $u \downarrow_{2^{\mathcal{F}_\varphi}} = u' \downarrow_{2^{\mathcal{F}_\varphi}}$, it holds that $p \in \ell'(u)$ iff $p \in \ell'(u')$.*

Thus, in CL *one cannot directly capture strategic problems on concurrent game structures where atomic propositions are not observable to all players. This contrasts with the definition of* ATL *with imperfect information in [20, Section 7.1] where imperfect information of agents is modelled by defining which atomic propositions they can observe.*

We now describe a natural translation of CL-instances to SL$_{\text{ii}}$-instances. This translation has two consequences:

1) It reduces the model-checking problem of CL to that of the hierarchical fragment of SL$_{\text{ii}}$.
2) It shows that CL only produces instances in which all atoms are uniform with regard to all observations, i.e., instances $(\Phi, \mathcal{G})$ such that for every $p \in \text{AP}$ and $o \in \text{Obs}$, $v \sim_o v'$ implies $p \in \ell(v) \leftrightarrow p \in \ell(v')$.

The input to the model-checking problem for CL consists of a CL formula φ and a finite representation of a $(\text{AP}_\varphi, 2^{\mathcal{F}_\varphi})$-tree t. The standard assumption is to assume t is a regular tree, i.e., is the unfolding of a finite structure. Precisely, a *finite representation* of a $(\text{AP}_\varphi, 2^{\mathcal{F}_\varphi})$-tree $t = (\tau, \ell')$ is a structure $\mathcal{S} = (S, R, s_\iota, \ell)$ such that

- $S = 2^{\mathcal{F}_\varphi}$,
- $s_\iota \in S$,
- $R = S \times S$,
- $\ell : S \to 2^{\text{AP}_\varphi}$,

and $t = t_{\mathcal{S}}(s_\iota)$ is the unfolding of $\mathcal{S}$.

Thus, an *instance* of the model-checking problem for CL is a pair $(\varphi, \mathcal{S})$ where φ is a CL formula (over variables

$\mathcal{S} \cup \mathcal{C}$), and $\mathcal{S} = (S, R, s_\iota, \ell)$ is a finite representation of a $(\mathrm{AP}_\varphi, 2^{\mathcal{F}_\varphi})$-tree. The *model-checking problem for* CL is the following decision problem: given an instance $(\varphi, \mathcal{S})$, return 'Yes' if $t_{\mathcal{S}}(s_\iota) \models \varphi$ and 'No' otherwise.

We will present the translation in two steps: first from CL-instances into $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$-instances, and then from $\mathsf{QCTL}^*_{\mathrm{ii}}$-instances to $\mathsf{SL}_{\mathrm{ii}}$-instances such that $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$-instances translate to hierarchical $\mathsf{SL}_{\mathrm{ii}}$-instances. We remind the reader that we use tree-semantics for $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$ (indeed, we only defined tree-semantics, not structure-semantics, for $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$).

1) Translating CL *to* $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$*:* Let $(\varphi, \mathcal{S})$ be an instance of the model-checking problem for CL, where $\mathcal{S} = (S, R, s_\iota, \ell)$. We will construct a $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$-instance $(\overline{\varphi}, \overline{\mathcal{S}})$ such that $\mathcal{S} \models \varphi$ iff $\overline{\mathcal{S}} \models \overline{\varphi}$. Let $\overline{\mathrm{AP}}$ be the set of all strategy variables occuring in φ, let $\mathcal{C}(\varphi)$ be the set of coordination variables that appear in φ, and assume, w.l.o.g., that $\mathcal{C}(\varphi) = [n]$ for some $n \in \mathbb{N}$. Let *hidden*$(\varphi) := \mathcal{C}(\varphi) \setminus \mathcal{F}_\varphi$.

First, we define the CKS $\overline{\mathcal{S}}$ over $\overline{\mathrm{AP}}$: the idea is to add in the structure $\mathcal{S}$ the local states corresponding to coordination variables that are not seen by all the strategies.

Formally, $\overline{\mathcal{S}} := (\overline{S}, \overline{R}, \overline{s_\iota}, \overline{\ell})$ where

- $\overline{S} = \prod_{c \in \mathcal{C}(\varphi)} L_c$ where $L_c = \{c_0, c_1\}$ (for all $c \in \mathcal{F}_\varphi$),
- $\overline{R} = \overline{S} \times \overline{S}$,
- $\overline{s_\iota} \in \overline{S}$ is any state s such that $s{\downarrow_{\mathcal{F}_\varphi}} = s_\iota$, and
- for every $s \in \overline{S}$, $\overline{\ell}(s) = \ell(s{\downarrow_{\mathcal{F}_\varphi}})$.

Second, we define concrete observations corresponding to strategy variables in φ. As explained in [19], and as reflected in the semantics of CL, the intuition is that a strategy variable s in formula φ observes coordination variables $\mathit{Scope}_\varphi(s)$. Therefore, we simply define, for each strategy variable s in φ, the concrete observation $\mathbf{o}_s := \mathit{Scope}_\varphi(s)$.

Finally, we define the $\mathsf{QCTL}^*_{\mathrm{ii}}$ formula $\overline{\varphi}$. This is done by induction on φ as follows (recall that we take for atomic propositions in $\mathsf{QCTL}^*_{\mathrm{ii}}$ the set of all strategy variables in φ):

$$\begin{aligned}
\overline{x} &:= x \\
\overline{\neg\varphi} &:= \neg\overline{\varphi} \\
\overline{\varphi_1 \vee \varphi_2} &:= \overline{\varphi_1} \vee \overline{\varphi_2} \\
\overline{\mathbf{X}\varphi} &:= \mathbf{X}\overline{\varphi} \\
\overline{\varphi_1 \mathbf{U} \varphi_2} &:= \overline{\varphi_1}\mathbf{U}\overline{\varphi_2} \\
\overline{\exists C \exists s.\, \varphi} &:= \exists^{\mathbf{o}_s} s.\, \mathbf{A}\overline{\varphi}
\end{aligned}$$

In the last case, note that $C \subseteq \mathbf{o}_s = \mathit{Scope}_\varphi(s)$.

Note that $\overline{\varphi}$ is a hierarchical $\mathsf{QCTL}^*_{\mathrm{ii}}$-formula. Also, one can easily check that the following holds:

Lemma 3. $t_{\mathcal{S}}(s_\iota) \models \varphi$ *iff* $t_{\overline{\mathcal{S}}}(\overline{s_\iota}) \models \mathbf{A}\overline{\varphi}$.

Importantly, we notice that $\mathbf{A}\overline{\varphi} \in \mathsf{QCTL}^*_{\mathrm{i},\subseteq}$, and that:

Lemma 4. *For every* $x \in \mathrm{AP}_\varphi$ *and every* s *quantified in* φ, $t_{\overline{\mathcal{S}}}(\overline{s_\iota})$ *is* $\mathbf{o}_s$*-uniform in* x.

2) Translation from $\mathsf{QCTL}^*_{\mathrm{ii}}$ *to* $\mathsf{SL}_{\mathrm{ii}}$*:* We now present a translation of $\mathsf{QCTL}^*_{\mathrm{ii}}$-instances to $\mathsf{SL}_{\mathrm{ii}}$-instances. It is a simple adaption of the the reduction from the model-checking problem for QCTL^* to the model-checking problem for $\mathsf{ATL}^*_{\mathrm{sc}}$ presented in [22].

Let $(\mathcal{S} = (S, R, s_\iota, \ell), \varphi)$ be an instance of the model-checking problem for $\mathsf{QCTL}^*_{\mathrm{ii}}$, where the set of states is $S \subseteq \prod_{i \in [n]} L_i$. We assume, without loss of generality, that every atomic proposition is quantified at most once, and that if it appears quantified it does not appear free. Also, let $\mathrm{AP}_\exists(\varphi) = \{p_1, \ldots, p_k\}$ be the set of atomic propositions quantified in φ, and for $i \in [k]$, let $\mathbf{o}_i$ be the concrete observation associated to the quantifier on p_i.

We build the $\mathrm{CGS}_{\mathrm{ii}}$ $\mathcal{G}_\mathcal{S} := (\mathrm{Ac}, V, E, \ell', v_\iota, \mathcal{O})$ over agents $\mathrm{Ag} := \{a_0, a_1, \ldots, a_k\}$, observations $\mathrm{Obs} := \{o_0, o_1, \ldots, o_k\}$ and atomic propositions $\mathrm{AP} := \mathrm{AP}(\varphi) \cup \{p_S\}$, where p_S is a fresh atomic proposition. Intuitively, agent a_0 is in charge of choosing transitions in $\mathcal{S}$, while agent a_i for $i \geq 1$ is in charge of choosing the valuation for $p_i \in \mathrm{AP}_\exists(\varphi)$.

To this aim, we let

$$V := \begin{array}{l} \{v_s \mid s \in S\} \cup \\ \{v_{s,i} \mid s \in S \text{ and } i \in [k]\} \cup \\ \{v_{p_i} \mid 0 \leq i \leq k\} \cup \\ \{v_\bot\} \end{array}$$

and

$$\mathrm{Ac} := \{c^s \mid s \in S\} \cup \{c^i \mid 0 \leq i \leq k\}.$$

In positions of the form v_s with $s \in S$, transitions are determined by the action chosen by agent a_0. First, she can choose to simulate a transition in $\mathcal{S}$: for every joint action $\boldsymbol{c} \in \mathrm{Ac}^{\mathrm{Ag}}$ such that $\boldsymbol{c}_0 = c^{s'}$,

$$E(v_s, \boldsymbol{c}) := \begin{cases} v_{s'} & \text{if } R(s, s') \\ v_\bot & \text{otherwise.} \end{cases}$$

She can also choose to move to a position in which agent a_i will choose the valuation for p_i in the current node: for every joint action $\boldsymbol{c} \in \mathrm{Ac}^{\mathrm{Ag}}$ such that $\boldsymbol{c}_0 = c^i$,

$$E(v_s, \boldsymbol{c}) := \begin{cases} v_{s,i} & \text{if } i \neq 0 \\ v_\bot & \text{otherwise.} \end{cases}$$

Next, in a position of the form $v_{s,i}$, agent a_i determines the transition, which codes the labelling of p_i in the current node: choosing c^i means that p_i holds in the current node, choosing any other action codes that p_i does not hold. Formally, for every joint action $\boldsymbol{c} \in \mathrm{Ac}^{\mathrm{Ag}}$,

$$E(v_{s,i}, \boldsymbol{c}) := \begin{cases} v_{p_i} & \text{if } \boldsymbol{c}_i = c^i \\ v_\bot & \text{otherwise.} \end{cases}$$

Positions of the form $v_{s,i}$ and $v_\bot$ are sink positions.

The labelling function ℓ' is defined as follows:

$$\ell'(v) := \begin{cases} \ell(s) \cup \{p_S\} & \text{if } v \in \{v_s \mid s \in S\} \\ \emptyset & \text{if } v \in \{v_{s,i} \mid s \in S, i \in [k]\} \\ & \qquad \cup \{v_{p,0}, v_\bot\} \\ \{p_i\} & \text{if } v = v_{p_i} \text{ with } i \in [k] \end{cases}$$

Finally we let $v_\iota := v_{s_\iota}$ and we define the observation interpretation as follows:

$$\mathcal{O}(o_0) := \{(v, v) \mid v \in V\},$$

meaning that agent a_0 has perfect information, and for $i \in [k]$, $\mathcal{O}(o_i)$ is the smallest reflexive relation such that

$$\mathcal{O}(o_i) \supseteq \bigcup_{s,s' \in S} \{(v_s, v_{s'}), (v_{s,i}, v_{s',i}) \mid s \approx_{\mathbf{o}_i} s'\}.$$

We explain the latter definition. First, observe that for every finite play ρ in $\mathcal{G}_\mathcal{S}$ that stays in $V_S = \{v_s \mid s \in S\}$, writing $\rho = v_{s_0} \dots v_{s_n}$, one can associate a finite path $\lambda_\rho = s_0 \dots s_n$ in $\mathcal{S}$. This mapping actually defines a bijection between the set of finite paths in $\mathcal{S}$ that start in s_ι and the set of finite plays in $\mathcal{G}_\mathcal{S}$ that remain in V_S.

Now, according to the definition of the transition function, a strategy σ_i for agent i with $i \in [k]$ is only relevant on finite plays of the form $\rho = \rho' \cdot v_{s,i}$, where $\rho' \in V_S^*$, and $\sigma_i(\rho)$ is meant to determine whether p_i holds in $\lambda_{\rho'}$. If σ_i is o_i-uniform, by definition of $\mathcal{O}(o_i)$, it determines an $\mathbf{o}_i$-uniform labelling for p in $t_\mathcal{S}(s_\iota)$. Reciprocally, an $\mathbf{o}_i$-uniform labelling for p in $t_\mathcal{S}(s_\iota)$ induces an $\mathcal{O}(o_i)$-strategy for agent a_i.

It remains to transform φ into an SL_{ii}-formula. To simulate the path quantifier of $\mathsf{QCTL}^*_{\text{ii}}$, a strategy for agent a_0 is enough as she alone chooses the path in $\mathcal{S}$. However the semantics of SL_{ii} is only defined on complete assignments, and we therefore need to make sure that all agents are bound to a strategy before using a temporal operator, and we must do so without breaking the hierarchy. For this we record in the translation what was the observation of the last propositional quantifier translated.

We define the SL_{ii} formula $\overline{\varphi}^{\mathbf{o}_i}$ by induction on φ as follows:

$$\overline{p}^{\mathbf{o}_i} := \begin{cases} \mathbf{A}^{\mathbf{o}_i} \langle\!\langle x_0 \rangle\!\rangle^{o_0}(a_0, x_0)(a_i, x_i)\mathbf{XX}p & \text{if } p = p_i \\ p & \text{otherwise} \end{cases}$$

$$\overline{\neg\varphi}^{\mathbf{o}_i} := \neg\overline{\varphi}^{\mathbf{o}_i}$$

$$\overline{\varphi_1 \vee \varphi_2}^{\mathbf{o}_i} := \overline{\varphi_1}^{\mathbf{o}_i} \vee \overline{\varphi_2}^{\mathbf{o}_i}$$

$$\overline{\mathbf{X}\varphi}^{\mathbf{o}_i} := \mathbf{X}\overline{\varphi}^{\mathbf{o}_i}$$

$$\overline{\varphi_1 \mathbf{U} \varphi_2}^{\mathbf{o}_i} := \overline{\varphi_1}^{\mathbf{o}_i} \mathbf{U} \overline{\varphi_2}^{\mathbf{o}_i}$$

$$\overline{\mathbf{E}\psi}^{\mathbf{o}_i} := \mathbf{A}^{\mathbf{o}_i} \langle\!\langle x_0 \rangle\!\rangle^{o_0}(a_0, x_0)(\mathbf{G}p_S \wedge \overline{\psi}^{\mathbf{o}_i})$$

$$\overline{\exists^{\mathbf{o}_j} p_j.\, \varphi}^{\mathbf{o}_i} := \langle\!\langle x_j \rangle\!\rangle^{o_j} \overline{\psi}^{\mathbf{o}_j}$$

where $\mathbf{A}^{\mathbf{o}_i}$ is a macro for $[\![x_0]\!]^{o_i} \dots [\![x_k]\!]^{o_i}(a_0, x_0) \dots (a_k, x_k)$. The cases for path formulas are similar. The universal quantification on paths is just used to obtain a complete assignment, then we redefine the relevant strategies.

We have the following:

Lemma 5. $\mathcal{S} \models \varphi$ *iff* $\mathcal{G}_\mathcal{S} \models \overline{\varphi}^{[n]}$.

We observe that if φ is hierarchical, then $(\overline{\varphi}^{[n]}, \mathcal{G}_\mathcal{S})$ is a hierarchical instance, and:

Lemma 6. *For every $p \in \mathrm{AP}_f(\varphi)$ and for every $i \in [k]$, if $t_\mathcal{S}(s_\iota)$ is $\mathbf{o}_i$-uniform in p then $v \sim_{o_i} v'$ implies that $p \in \ell(v)$ iff $p \in \ell(v')$.*

C. Proof of Theorem 3

Theorem 3. *The model-checking problem for $\mathsf{QCTL}^*_{\text{ii}}$ under tree-semantics is undecidable.*

Proof. Let MSO_{eq} denote the extension of the logic MSO (without unary predicates) by a binary predicate symbol eq. MSO_{eq} is interpreted on the full binary tree, and the semantics of $\mathrm{eq}(x, y)$ is that x and y have the same depth in the tree. We show how to effectively translate MSO_{eq} into $\mathsf{QCTL}^*_{\text{ii}}$; then our result follows since the MSO_{eq}-theory of the binary tree is undecidable [17].

The translation of MSO to QCTL^* from [14] can be extended to one from MSO_{eq} to $\mathsf{QCTL}^*_{\text{ii}}$ by adding rules for the equal level predicate. Indeed, for $\varphi(x, x_1, \dots, x_i, X_1, \dots, X_j) \in \mathsf{MSO}$, we inductively define the $\mathsf{QCTL}^*_{\text{ii}}$ formula $\widehat{\varphi}$ as follows:

$$\widehat{x = x_k} := p_{x_k} \qquad \widehat{x_k = x_l} := \mathbf{EF}(p_{x_k} \wedge p_{x_l})$$

$$\widehat{x \in X_k} := p_{X_k} \qquad \widehat{x_k \in X_l} := \mathbf{EF}(p_{x_k} \wedge p_{X_l})$$

$$\widehat{\neg\varphi'} := \neg\widehat{\varphi'} \qquad \widehat{\varphi_1 \vee \varphi_2} := \widehat{\varphi_1} \vee \widehat{\varphi_2}$$

$$\widehat{\exists x_k.\varphi'} := \exists p_{x_k}.\, \mathrm{uniq}(p_{x_k}) \wedge \widehat{\varphi'}$$

$$\widehat{\exists X_k.\varphi'} := \exists p_{X_k}.\, \widehat{\varphi'}$$

$$\widehat{S(x, x_k)} := \mathbf{EX}p_{x_k} \qquad \widehat{S(x_k, x)} := \bot$$

$$\widehat{S(x_k, x_l)} := \mathbf{EF}(p_{x_k} \wedge \mathbf{EX}p_{x_l})$$

where $\mathrm{uniq}(p) := \mathbf{EF}p \wedge \forall q.\ (\mathbf{EF}(p \wedge q) \to \mathbf{AG}(p \to q))$ holds in a tree iff it has exactly one node labelled with p. To understand the $x = x_k$ case, we will interpret x as the root. To understand the $S(x_k, x)$ case, observe that x has no incoming edge since it is interpreted as the root.

The rules for eq are as follows:

$$\widehat{\mathrm{eq}(x, x_k)} := p_{x_k}$$

$$\widehat{\mathrm{eq}(x_k, x_l)} := \exists^\emptyset p.\, \mathbf{border}(p) \wedge \mathbf{AG}(p_{x_k} \to p \wedge p_{x_l} \to p)$$

To understand the first case, observe that since x is interpreted as the root, x_k is on the same level as x if and only if it is also assigned the root. To understand the second case, recall from Example 2 that the $\mathsf{QCTL}^*_{\text{ii}}$ formula $\exists^\emptyset p.\, \mathbf{border}(p)$ places one unique horizontal line of p's in the tree, and thus requiring that x_k and x_l be both on this line ensures that they are on the same level.

To finish, let $\mathcal{S}$ be a CKS with two states s_0 and s_1 (local states are irrelevant here), whose transition relation is the complete relation, and with empty labelling function. Clearly, $t_\mathcal{S}(s_0)$ is the full binary tree. It is easy to prove the following by induction:

For every $\varphi(x, x_1, \dots, x_i, X_1, \dots, X_j) \in \mathsf{MSO}_{\text{eq}}$, $t_\mathcal{S}(s) \models \varphi(s, u_1, \dots, u_i, U_1, \dots, U_j)$ if and only if $t_\mathcal{S}(s_0)', s \models \widehat{\varphi}$, where $t_\mathcal{S}(s_0)'$ is obtained from $t_\mathcal{S}(s_0)$ by changing the labelling for variables p_{x_k} and p_{X_k} as follows: $p_{x_k} \in \ell'(u)$ if $u = u_k$ and $p_{X_k} \in \ell'(u)$ if $u \in U_k$.

In particular, it follows that

$$t_\mathcal{S}(s_0), s_0 \models \varphi(x) \text{ iff } t_\mathcal{S}(s_0), s_0 \models \widehat{\varphi}.$$

This completes the proof. □

D. Proof of Lemma 1

Before presenting the proof of Lemma 1 we recall the definition of acceptance by ATA via games between Eve and Adam. Let $\mathcal{A} = (Q, \delta, q_\iota, C)$ be an ATA over (AP, X)-trees, let $t = (\tau, \ell)$ be such a tree and let $u_\iota \in \tau$. We define the parity game $\mathcal{G}(\mathcal{A}, t, u_\iota) = (V, E, v_\iota, C')$: the set of positions is $V = \tau \times Q \times \mathbb{B}^+(X \times Q)$, the initial position is $v_\iota = (u_\iota, q_\iota, \delta(q_\iota, u_\iota))$, and a position (u, q, α) belongs to Eve if α is of the form $\alpha_1 \vee \alpha_2$ or $[x, q']$; otherwise it belongs to Adam. Moves in $\mathcal{G}(\mathcal{A}, t, u_\iota)$ are defined by the following rules:

$$(u, q, \alpha_1 \dagger \alpha_2) \to (u, q, \alpha_i) \quad \text{where } \dagger \in \{\vee, \wedge\} \text{ and } i \in \{1, 2\},$$
$$(u, q, [x, q']) \to (u \cdot x, q', \delta(q', \ell(u \cdot x)))$$

Positions of the form $(u, q, \top)$ and $(u, q, \bot)$ are deadlocks, winning for Eve and Adam respectively. The colouring is inherited from the one of the automaton: $C'(u, q, \alpha) = C(q)$.

A tree t is *accepted* in node u by $\mathcal{A}$ if Eve has a winning strategy in $\mathcal{G}(\mathcal{A}, t, u)$.

We now recall the lemma and the automata construction, and we prove it to be correct.

Lemma 7 (Translation)**.** *Let $(\Phi, \mathcal{S})$ be an instance of the model-checking problem for $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$. For every subformula φ of Φ and state s of $\mathcal{S}$, one can build an ATA $\mathcal{A}^\varphi_s$ on $(\mathrm{AP}_\exists(\Phi), L_\varphi)$-trees such that for every $(\mathrm{AP}_\exists(\Phi), L_\varphi)$-tree t rooted in $s{\downarrow_{I_\varphi}}$,*

$$t \in \mathcal{L}(\mathcal{A}^\varphi_s) \quad \textit{iff} \quad t{\uparrow}^{[n]}_y \between t_{\mathcal{S}}(s) \models \varphi, \quad \textit{where } y = s{\downarrow_{[n]\setminus I_\varphi}}.$$

For an L_I-tree t, from now on $t{\uparrow}^{[n]} \between t_{\mathcal{S}}(s)$ stands for $t{\uparrow}^{[n]}_y \between t_{\mathcal{S}}(s)$, where $y = s{\downarrow_{[n]\setminus I}}$: while lifting tree t to $[n]$, the missing local states in the root of t are filled with those from s.

Proof sketch of Lemma 1. Let $(\Phi, \mathcal{S})$ be an instance of the model-checking problem for $\mathsf{QCTL}^*_{\mathrm{i},\subseteq}$. Let $\Phi \in \mathsf{QCTL}^*_{\mathrm{i},\subseteq}$, and let $\mathrm{AP}_\exists = \mathrm{AP}_\exists(\Phi)$ and $\mathrm{AP}_f = \mathrm{AP}_f(\Phi)$, and recall that $\mathcal{S}$ is labelled over AP_f. For each state $s \in S$ and each subformula φ of Φ (note that all subformulas of Φ are also hierarchical), we define by induction on φ the ATA $\mathcal{A}^\varphi_s$ on $(\mathrm{AP}_\exists, L_\varphi)$-trees.

$\varphi = p$:

We let $\mathcal{A}^p_s$ be the ATA over $L_{[n]}$-trees with one unique state q_ι, with transition function defined as follows:

$$\delta(q_\iota, a) = \begin{cases} \top & \text{if } \begin{array}{c} p \in \mathrm{AP}_f \text{ and } p \in \ell_{\mathcal{S}}(s) \\ \text{or} \\ p \in \mathrm{AP}_\exists \text{ and } p \in a \end{array} \\ \bot & \text{if } \begin{array}{c} p \in \mathrm{AP}_f \text{ and } p \notin \ell_{\mathcal{S}}(s) \\ \text{or} \\ p \in \mathrm{AP}_\exists \text{ and } p \notin a \end{array} \end{cases}$$

$\varphi = \neg\varphi'$:

We obtain $\mathcal{A}^\varphi_s$ by dualising $\mathcal{A}^{\varphi'}_s$, a classic operation.

$\varphi = \varphi_1 \vee \varphi_2$:

Because $I_\varphi = I_{\varphi_1} \cap I_{\varphi_2}$, and each $\mathcal{A}^{\varphi_i}_s$ works on L_{φ_i}-trees, we first narrow them so that they work on L_φ-trees: for $i \in \{1, 2\}$, we let $\mathcal{A}_i := \mathcal{A}^{\varphi_i}_s{\downarrow_{I_\varphi}}$. We then build $\mathcal{A}^\varphi_s$ by taking the disjoint union of $\mathcal{A}_1$ and $\mathcal{A}_2$ and adding a new initial state that nondeterministically chooses which of $\mathcal{A}_1$ or $\mathcal{A}_2$ to execute on the input tree, so that $\mathcal{L}(\mathcal{A}^\varphi_s) = \mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)$.

$\varphi = \mathbf{E}\psi$:

The aim is to build an automaton $\mathcal{A}^\varphi_s$ that works on L_φ-trees and that on input t, checks for the existence of a path in $t{\uparrow}^{[n]} \between t_{\mathcal{S}}(s)$ that satisfies ψ. Observe that a path in $t{\uparrow}^{[n]} \between t_{\mathcal{S}}(s)$ is a path in $t_{\mathcal{S}}(s)$, augmented with the labelling for atomic propositions in $\mathrm{AP}_\exists$ carried by t.

To do so, $\mathcal{A}^\varphi_s$ guesses a path λ in $(\mathcal{S}, s)$. It remembers the current state in $\mathcal{S}$, which provides the labelling for atomic propositions in AP_f, and while it guesses λ it follows its L_φ-narrowing in its input tree t (which always exists since input to tree automata are complete trees), reading the labels to evaluate propositions in $\mathrm{AP}_\exists$.

Let $\max(\psi) = \{\varphi_1, \ldots, \varphi_n\}$ be the set of maximal state sub-formulas of ψ. In a first step we see these maximal state sub-formulas as atomic propositions. The formula ψ can thus be seen as an LTL formula, and we can build a nondeterministic parity word automaton $\mathcal{W}^\psi = (Q^\psi, \Delta^\psi, q^\psi_\iota, C^\psi)$ over alphabet $2^{\max(\psi)}$ that accepts exactly the models of ψ [35]. [5] We define the ATA $\mathcal{A}$ that, given as input a $(\max(\psi), L_\varphi)$-tree t, nondeterministically guesses a path λ in $t{\uparrow}^{[n]} \between t_{\mathcal{S}}(s)$ and simulates $\mathcal{W}^\psi$ on it, assuming that the labels it reads while following $\lambda{\downarrow_{I_\varphi}}$ in its input correctly represent the truth value of formulas in $\max(\psi)$ along λ. Recall that $\mathcal{S} = (S, R, s_\iota, \ell_{\mathcal{S}})$; we define $\mathcal{A} := (Q, \delta, q_\iota, C)$, where

- $Q = Q^\psi \times S$,
- $q_\iota = (q^\psi_\iota, s)$,
- $C(q^\psi, s') = C^\psi(q^\psi)$, and
- for each $(q^\psi, s') \in Q$ and $a \in 2^{\max(\psi)}$,

$$\delta((q^\psi, s'), a) = \bigvee_{q' \in \Delta^\psi(q^\psi, a)} \bigvee_{s'' \in R(s')} [s''{\downarrow_{I_\varphi}}, (q', s'')].$$

The intuition is that $\mathcal{A}$ reads the current label, chooses nondeterministically which transition to take in $\mathcal{W}^\psi$, chooses a next state in $\mathcal{S}$ and proceeds in the corresponding direction in X_φ. Thus, $\mathcal{A}$ accepts a $\max(\varphi)$-labelled L_φ-tree t iff there is a path in t that is the L_φ-narrowing of some path in $t_{\mathcal{S}}(s)$, and that satisfies ψ, where maximal state formulas are considered as atomic propositions.

Now from $\mathcal{A}$ we build the automaton $\mathcal{A}^\varphi_s$ over L_φ-trees labelled with "real" atomic propositions in $\mathrm{AP}_\exists$. In each node it visits, $\mathcal{A}^\varphi_s$ guesses what should be its labelling over $\max(\psi)$, it simulates $\mathcal{A}$ accordingly, and checks that the guess it made is correct. If the path being guessed in $t_{\mathcal{S}}(s)$ is currently in node u ending with state s', and $\mathcal{A}^\varphi_s$ guesses that φ_i holds in

[5] Note that, as usual for nondeterministic word automata, we take the transition function of type $\Delta^\psi : Q^\psi \times 2^{\max(\psi)} \to 2^{Q^\psi}$.

u, it checks this guess by starting a simulation of automaton $\mathcal{A}^{\varphi_i}_{s'}$ from node $v = u{\downarrow_{I_\varphi}}$ in its input t.

For each $s' \in \mathcal{S}$ and each $\varphi_i \in \max(\psi)$ we first build $\mathcal{A}^{\varphi_i}_{s'}$, and we let $\mathcal{A}^i_{s'} := \mathcal{A}^{\varphi_i}_{s'} = (Q^i_{s'}, \delta^i_{s'}, q^i_{s'}, C^i_{s'})$. We also let $\overline{\mathcal{A}^i_{s'}} = (\overline{Q^i_{s'}}, \overline{\delta^i_{s'}}, \overline{q^i_{s'}}, \overline{C^i_{s'}})$ be its dualisation, and we assume w.l.o.g. that all the state sets are pairwise disjoint. Observe that because each φ_i is a maximal state sub-formula, we have $I_{\varphi_i} = I_\varphi$, so that we do not need to narrow down automata $\mathcal{A}^i_{s'}$ and $\overline{\mathcal{A}^i_{s'}}$. We define the ATA

$$\mathcal{A}^\varphi_s = (Q \cup \bigcup_{i,s'} Q^i_{s'} \cup \overline{Q^i_{s'}}, \delta', q_\iota, C'),$$

where the colours of states are left as they were in their original automaton, and δ is defined as follows. For states in $Q^i_{s'}$ (resp. $\overline{Q^i_{s'}}$), δ agrees with $\delta^i_{s'}$ (resp. $\overline{\delta^i_{s'}}$), and for $(q^\psi, s') \in Q$ and $a \in 2^{\mathrm{AP}_\exists}$ we let

$$\delta'((q^\psi, s'), a) := \bigvee_{a' \in 2^{\max(\psi)}} \Bigg(\delta\left((q^\psi, s'), a'\right) \wedge \bigwedge_{\varphi_i \in a'} \delta^i_{s'}(q^i_{s'}, a) \wedge \bigwedge_{\varphi_i \notin a'} \overline{\delta^i_{s'}}(\overline{q^i_{s'}}, a) \Bigg).$$

$\varphi = \exists^{\mathbf{o}} p.\, \varphi'$:

We build automaton $\mathcal{A}^{\varphi'}_s$ that works on $L_{\varphi'}$-trees; because φ is hierarchical, we have that $\mathbf{o} \subseteq I_{\varphi'}$ and we can narrow down $\mathcal{A}^{\varphi'}_s$ to work on $L_{\mathbf{o}}$-trees and obtain $\mathcal{A}_1 := \mathcal{A}^{\varphi'}_s{\downarrow_{\mathbf{o}}}$. By Theorem 6 we can nondeterminise it to get $\mathcal{A}_2$, which by Theorem 5 we can project with respect to p, finally obtaining $\mathcal{A}^\varphi_s := \mathcal{A}_2 \Downarrow_p$.

Correctness. We now prove by induction on φ that the construction is correct. In each case, we let $t = (\tau, \ell)$ be a complete $(\mathrm{AP}_\exists, L_\varphi)$-tree rooted in $s{\downarrow_{L_\varphi}}$.

$\varphi = p$:

First, note that $I_p = [n]$, so that t is rooted in $s{\downarrow_{L_p}} = s$. Let us consider first the case where $p \in \mathrm{AP}_f$. By definition of $\mathcal{A}^p_s$, we have that $t \in \mathcal{L}(\mathcal{A}^p_s)$ iff $p \in \ell_\mathcal{S}(s)$. On the other hand, by definition of the merge operation, of the unfolding, and because $\mathrm{AP}_\exists$ and AP_f are disjoint, we have $t{\uparrow^{[n]}} \between t_\mathcal{S}(s) \models p$ iff $p \in \ell_\mathcal{S}(s)$, and we are done. Now if $p \in \mathrm{AP}_\exists$: by definition of $\mathcal{A}^p_s$, we have $t \in \mathcal{L}(\mathcal{A}^p_s)$ iff $p \in \ell(s)$; also, by definition of the merge, of the unfolding, and because $\mathrm{AP}_\exists$ and AP_f are disjoint, we have that $t{\uparrow^{[n]}} \between t_\mathcal{S}(s) \models p$ iff $p \in \ell(s)$, which concludes.

$\varphi = \neg\varphi'$:

This case is trivial.

$\varphi_1 \vee \varphi_2$:

We have $\mathcal{A}_i = \mathcal{A}^{\varphi_i}_s{\downarrow_{L_\varphi}}$, so by Theorem 7, for all $y \in L_{I_{\varphi_i} \setminus I_\varphi}$, we have $t \in \mathcal{L}(\mathcal{A}_i)$ iff $t{\uparrow^{L_{\varphi_i}}_y} \in \mathcal{L}(\mathcal{A}^{\varphi_i}_s)$, which by induction hypothesis holds iff $(t{\uparrow^{L_{\varphi_i}}_y}){\uparrow^{[n]}} \between t_\mathcal{S}(s) \models \varphi_i$. Choosing $y = s{\downarrow_{I_{\varphi_i} \setminus I_\varphi}}$, we get that $t \in \mathcal{L}(\mathcal{A}_i)$ iff $t{\uparrow^{[n]}} \between t_\mathcal{S}(s) \models \varphi_i$. We conclude by reminding that $\mathcal{L}(\mathcal{A}^\varphi_s) = \mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)$.

$\varphi = \mathbf{E}\psi$:

Suppose that $t' = t{\uparrow^{[n]}} \between t_\mathcal{S}(s) \models \mathbf{E}\psi$. There exists a path λ starting at the root s of t' such that $t', \lambda \models \psi$. Again, let $\max(\psi)$ be the set of maximal state subformulas of φ, and let w be the infinite word over $2^{\max(\psi)}$ that agrees with λ on the state formulas in $\max(\psi)$. By definition, $\mathcal{W}^\psi$ has an accepting execution on w. Now in the acceptance game of $\mathcal{A}^\varphi_s$ on t, Eve can guess the path λ, following $\lambda{\downarrow_{X_\varphi}}$ in its input t, and she can also guess the corresponding word w on $2^{\max(\psi)}$ and an accepting execution of $\mathcal{W}^\psi$ on w. Let $u' \in t'$ be a node of λ, s' its last direction and let $u = u'{\downarrow_{L_\varphi}} \in t$. Assume that in node u of the input tree, in a state $(q^\psi, s') \in Q$, Adam challenges Eve on some $\varphi_i \in \max(\psi)$ that she assumes to be true in u', *i.e.*, Adam chooses the conjunct $\delta^i_{s'}(q^i_{s'}, a)$, where a is the label of u. Note that in the evaluation game this means that Adam moves to position $(u, (q^\psi, s'), \delta^i_{s'}(q^i_{s'}, a))$. We want to show that Eve wins from this position.

Let t_u (resp. $t'_{u'}$) be the subtree of t (resp. t') starting in u (resp. u')[6]. It is enough to show that t_u is accepted by $\mathcal{A}^i_{s'} = \mathcal{A}^{\varphi_i}_{s'}$. Observe that t_u is rooted in the last direction of $u = u'{\downarrow_{L_\varphi}}$, and since the last direction of u' is s' we have that t_u is rooted in $s'{\downarrow_{L_\varphi}}$. Since $I_\varphi = I_{\varphi_i}$ we have that t_u is rooted in $s'{\downarrow_{L_{\varphi_i}}}$. We can thus apply the induction hypothesis on t_u with φ_i, and we get that

$$t_u \in \mathcal{L}(\mathcal{A}^{\varphi_i}_{s'}) \quad \text{iff} \quad t_u{\uparrow^{[n]}} \between t_\mathcal{S}(s') \models \varphi_i. \tag{1}$$

Now, because u' ends in s' we also have that

$$t'_{u'} = t_u{\uparrow^{[n]}} \between t_\mathcal{S}(s'). \tag{2}$$

Putting (1) and (2) together, we obtain that

$$t_u \in \mathcal{L}(\mathcal{A}^i_{s'}) \quad \text{iff} \quad t'_{u'} \models \varphi_i. \tag{3}$$

Because we have assumed that Eve guesses w correctly, we also have that $t', u' \models \varphi_i$, *i.e.*, $t'_{u'} \models \varphi_i$. This, together with (3), gives us that t_u is accepted by $\mathcal{A}^i_{s'}$.

Eve thus has a winning strategy from the initial position of the acceptance game of $\mathcal{A}^i_{s'}$ on t_u. This initial position is $(u, q^i_{s'}, \delta^i_{s'}(q^i_{s'}, a))$. Since $(u, q^i_{s'}, \delta^i_{s'}(q^i_{s'}, a))$ and $(u, q, \delta^i_{s'}(q^i_{s'}, a))$ contain the same node u and transition formula $\delta^i_{s'}(q^i_{s'}, a)$, a winning strategy in one of these positions[7] is also a winning strategy in the other, and therefore Eve wins Adam's challenge. With a similar argument, we get that also when Adam challenges Eve on some φ_i assumed not to be true in node v, Eve wins the challenge. Finally, Eve wins the acceptance game of $\mathcal{A}^\varphi_s$ on t, and thus $t \in \mathcal{L}(\mathcal{A}^\varphi_s)$.

For the other direction, assume that $t \in \mathcal{L}(\mathcal{A}^\varphi_s)$, *i.e.*, Eve wins the evaluation game of $\mathcal{A}^\varphi_s$ on t. Again, let $t' = t{\uparrow^{[n]}}$

[6] If $u = w \cdot x$, a subtree t_u of $t = (\tau, \ell)$ is defined as $t_u := (\tau_u, \ell_u)$ with $\tau_u = \{x \cdot w' \mid w \cdot x \cdot w' \in \tau\}$, and $\ell_u(x \cdot w') = \ell(w \cdot x \cdot w')$: we remove from each node all directions before $\mathrm{last}(u)$.

[7] Recall that positional strategies are sufficient in parity games [38].

$\between t_{\mathcal{S}}(s)$. A winning strategy for Eve describes a path λ in $t_{\mathcal{S}}(s)$, which is also a path in t'. This winning strategy also defines an infinite word w over $2^{\max(\psi)}$ such that w agrees with λ on the formulas in $\max(\psi)$, and it also describes an accepting run of $\mathcal{W}^{\psi}$ on w. Hence $t', \lambda \models \psi$, and $t' \models \varphi$.

$\varphi = \exists^{\mathbf{o}} p.\, \varphi'$:

First, observe that because φ is hierarchical, we have that $I_{\varphi} = \mathbf{o}$. Next, by Theorem 5 we have that

$$t \in \mathcal{L}(\mathcal{A}_s^{\varphi}) \quad \text{iff} \quad \text{there exists } t_p \equiv_p t \text{ s.t. } t_p \in \mathcal{L}(\mathcal{A}_2). \tag{4}$$

By Theorem 6, $\mathcal{L}(\mathcal{A}_2) = \mathcal{L}(\mathcal{A}_1)$, and since $\mathcal{A}_1 = \mathcal{A}_s^{\varphi'} \downarrow_{\mathbf{o}} = \mathcal{A}_s^{\varphi'} \downarrow_{L_{\varphi}}$ we get by Theorem 7 that

$$t_p \in \mathcal{L}(\mathcal{A}_2) \quad \text{iff} \quad t_p \uparrow_y^{L_{\varphi'}} \in \mathcal{L}(\mathcal{A}_s^{\varphi'}), \text{ where } y = s \downarrow_{(I_{\varphi'} \setminus I_{\varphi})}. \tag{5}$$

Now t_p and t have the same root, $s \downarrow_{L_{\varphi}}$. The root of $t_p \uparrow_y^{L_{\varphi'}}$ is thus $(s \downarrow_{L_{\varphi}}, y) = s \downarrow_{L_{\varphi'}}$, and we can apply the induction hypothesis on $t_p \uparrow_y^{L_{\varphi'}}$ with φ':

$$t_p \uparrow_y^{L_{\varphi'}} \in \mathcal{L}(\mathcal{A}_s^{\varphi'}) \quad \text{iff} \quad (t_p \uparrow_y^{L_{\varphi'}}) \uparrow^{[n]} \between t_{\mathcal{S}}(s) \models \varphi'. \tag{6}$$

Now, combining points (4), (5) and (6), together with the fact that $(t_p \uparrow_y^{L_{\varphi'}}) \uparrow^{[n]} \between t_{\mathcal{S}}(s) = t_p \uparrow^{[n]} \between t_{\mathcal{S}}(s)$ gives us that

$$t \in \mathcal{L}(\mathcal{A}_s^{\varphi}) \quad \text{iff} \quad \exists\, t_p \equiv_p t \text{ s.t. } t_p \uparrow^{[n]} \between t_{\mathcal{S}}(s) \models \varphi'. \tag{7}$$

We now prove equation below which, together with point (7), concludes the proof:

$$\begin{gathered} \exists\, t_p \equiv_p t \text{ s.t. } t_p \uparrow^{[n]} \between t_{\mathcal{S}}(s) \models \varphi' \\ \text{iff} \\ t \uparrow^{[n]} \between t_{\mathcal{S}}(s) \models \exists^{\mathbf{o}} p.\, \varphi' \end{gathered} \tag{8}$$

For the first direction, assume that there exists $t_p \equiv_p t$ such that $t_p \uparrow^{[n]} \between t_{\mathcal{S}}(s) \models \varphi'$. First, by definition of the merge, because $p \in \mathrm{AP}_{\exists}$ and $\mathrm{AP}_{\exists}$ and AP_f are disjoint, the p-labelling of $t_p \uparrow^{[n]} \between t_{\mathcal{S}}(s)$ is determined by the p-labelling of $t_p \uparrow^{[n]}$, which by definition of the widening is $\mathbf{o}$-uniform. In addition it is clear that $t_p \uparrow^{[n]} \between t_{\mathcal{S}}(s) \equiv_p t \uparrow^{[n]} \between t_{\mathcal{S}}(s)$, which concludes this direction.

For the other direction, assume that $t \uparrow^{[n]} \between t_{\mathcal{S}}(s) \models \exists^{\mathbf{o}} p.\, \varphi'$: there exists $t'_p \equiv_p t \uparrow^{[n]} \between t_{\mathcal{S}}(s)$ such that t'_p is $\mathbf{o}$-uniform in p and $t'_p \models \varphi'$. Let us write $t'_p = (\tau', \ell'_p)$ and $t = (\tau, \ell)$. We define $t_p := (\tau, \ell_p)$ where for each $u \in \tau$, if there exists $u' \in \tau'$ such that $u' \downarrow_{\mathbf{o}} = u$, we let

$$\ell_p(u) = \begin{cases} \ell(u) \cup \{p\} & \text{if } p \in \ell'_p(u') \\ \ell(u) \setminus \{p\} & \text{otherwise.} \end{cases}$$

This is well defined because t'_p is $\mathbf{o}$-uniform in p: if two nodes u', v' project on u, we have $u' \approx_{\mathbf{o}} v'$ and thus they agree on p. In case there is no $u' \in \tau'$ such that $u' \downarrow_{L_{\varphi}} = u$, we can let $\ell_p(u) = \ell(u)$ as this node disappears in $t_p \uparrow^{[n]} \between t_{\mathcal{S}}(s)$. Clearly, $t_p \equiv_p t$. Now we write $t''_p = t_p \uparrow^{[n]} \between t_{\mathcal{S}}(s)$ and we prove that $t''_p = t'_p$ hence $t''_p \models \varphi'$, which concludes. It is clear that t''_p and t'_p have the same domain. Also, because $t'_p \equiv_p t \uparrow^{[n]} \between t_{\mathcal{S}}(s)$ and $t''_p = t_p \uparrow^{[n]} \between t_{\mathcal{S}}(s)$, by definition of the merge both agree with $t_{\mathcal{S}}(s)$ for all atomic propositions in AP_f. Because $t_p \equiv_p t$, and again by definition of the merge, t''_p and t'_p also agree on all atomic propositions in $\mathrm{AP}_{\exists} \setminus \{p\}$. Finally, by definition of t_p and because t'_p is $\mathbf{o}$-uniform in p, we get that t''_p and t'_p also agree on p, and therefore $t''_p = t'_p$. $\square$