Strategy Logic with Imperfect Information

RAPHAËL BERTHON, École Normale Supérieure de Rennes, France
BASTIEN MAUBERT, Università degli Studi di Napoli "Federico II", Italy
ANIELLO MURANO, Università degli Studi di Napoli "Federico II", Italy
SASHA RUBIN, Università degli Studi di Napoli "Federico II", Italy
MOSHE Y. VARDI, Rice University, USA

We introduce an extension of Strategy Logic for the imperfect-information setting, called SL_{ii}, and study its model-checking problem. As this logic naturally captures multi-player games with imperfect information, this problem is undecidable; but 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, and we prove that model-checking SL_{ii} restricted to hierarchical instances is decidable. This result, because it allows for complex patterns of existential and universal quantification on strategies, greatly generalises the decidability of distributed synthesis for systems with hierarchical information. It allows us to easily derive new decidability results concerning strategic problems under imperfect information such as the existence of Nash equilibria, or rational synthesis.

To establish this result we go through an intermediary, "low-level" logic much more adapted to automata techniques. $QCTL^*$ is an extension of CTL^* with second-order quantification over atomic propositions that has been used to study strategic logics with perfect information. We extend it to the imperfect information setting by parameterising second-order quantifiers with observations. The simple syntax of the resulting logic, $QCTL_{ii}^*$, allows us to provide a conceptually neat reduction of SL_{ii} to $QCTL_{ii}^*$ that separates concerns, allowing one to forget about strategies and players and focus solely on second-order quantification. While the model-checking problem of $QCTL_{ii}^*$ is, in general, undecidable, we identify a syntactic fragment of hierarchical formulas and prove, using an automata-theoretic approach, that it is decidable.

We apply our result to solve complex strategic problems in the imperfect-information setting. We first show that the existence of Nash equilibria for deterministic strategies is decidable in games with hierarchical information. We also introduce distributed rational synthesis, a generalisation of rational synthesis to the imperfect-information setting. Because it can easily be expressed in our logic, our main result provides a solution to this problem in the case of hierarchical information.

CCS Concepts: • **Theory of computation** → **Logic and verification**; **Modal and temporal logics**; *Automata over infinite objects*;

Additional Key Words and Phrases: strategic reasoning, imperfect information, perfect recall, distributed synthesis, hierarchical information, Nash equilibria, rational synthesis

1 INTRODUCTION

Temporal logics such as LTL [67] or CTL* [28] are extremely successful logics that have been studied in great detail and extended in many directions along the past decades, notably in relation with the development of the model-checking approach to program verification [22]. When considering systems with multiple components such as multi-agent systems or distributed programs, popular extensions of temporal logics are the family of so-called *logics for strategic reasoning*, or *strategic logics*, which introduce operators that can express the existence of strategies for components to ensure that the system's executions satisfy certain temporal properties.

A fundational logic in this family is Alternating-time Temporal Logic (ATL) [1]. It extends CTL* with a coalition operator $\langle A \rangle \varphi$, where A is a subset of components/agents of the system, which reads as "coalition A has a strategy to enforce property φ no matter what the other components/agents do". This logic is thus quite expressive, as it allows for instance to express the existence of winning strategies in games played on graphs. However it is not well suited to reason about other important solution concepts in game theory, such as Nash equilibria. To address this problem Strategy Logic (SL) was introduced [20, 60]. In SL strategies are treated as first-order objects, thanks to strategy variables x that can be quantified upon and bound to players: $\langle\!\langle x \rangle\!\rangle$ reads as "there exists a strategy x", and (a, x) reads as "strategy x is assigned to player a". This leads to a very expressive logic that can express many solution concepts from game-theory such as best response, existence of Nash equilibria or subgame-perfect equilibria.

Imperfect information. An essential property of realistic multi-player games is that players often have a limited view of the system. Such imperfect information, or partial observation, is usually captured by equipping the models with equivalence relations o (called *observations*) over the state space, that specify indistinguishable states. Strategies are then required to be *uniform*, i.e., they cannot assign different moves to indistinguishable situations. Imperfect information is known to make games computationally harder to solve. For two-player reachability games, Reif showed in [73] that deciding the existence of winning strategies is ExpTime-complete for imperfect information, while it is in Ptime for perfect information. This result has later been generalised to omega-regular objectives [7, 26], and adapted to the setting of program synthesis from temporal specifications [49, 68]. In the case of multiple players/components/agents, which interests us here, the situation is even worse: the existence of distributed winning strategies is undecidable already for two players with incomparable observation trying to enforce some reachability objective in the presence of an adversarial third player [65], and a similar result was also proved in the framework of distributed synthesis [69]. Since then, the formal-methods community has spent much effort finding restrictions and variations that ensure decidability [8, 31, 35, 50, 64, 66, 69, 74]. The common thread in these approaches is hierarchical information: players can be totally ordered according to how well they observe the game. Another line of works establishes that decidability can be retained by forbidding private communication, i.e., by considering variants around the idea that all new information should be public [4, 5, 11, 72, 79, 80].

Strategy Logic with imperfect information. We propose an extension of Strategy Logic to the imperfect-information setting, which we call SL_{ii}. The first step is to choose how to introduce imperfect information in the logic. In the formal-methods literature it is typical to associate observations to players. In SL_{ii}, instead, we associate observations to strategies: the strategy quantifier $\langle\!\langle x \rangle\!\rangle$ from SL is now parameterised by observation o, written $\langle\!\langle x \rangle\!\rangle^o$. This novelty allows one to

express, in the logic, that a player's observation changes over time, to capture for instance the loss of a sensor resulting in a diminished observation power. We also add to our logic SL_{ii} the outcome quantifier $\mathbf{A}$ from Branching-time Strategy Logic (BSL) [45], which quantifies on outcomes of strategies currently used by the agents, and the unbinding operator $(a, ?)$, which frees an agent from her current strategy. This does not increase the expressivity of the logic but presents advantages that we discuss in Section 2.2. For instance it allows us to naturally consider nondeterministic strategies (Strategy Logic only considers deterministic ones), which in turn allows us to capture module checking, the extension of model checking to open systems [42, 43, 52].

The logic SL_{ii} is very powerful: it is an extension of SL (which considers perfect information), and of the imperfect-information strategic logics $ATL^*_{i,R}$ [15] and $ATL^*_{sc,i}$ [55]. As already mentioned, SL_{ii} can express the distributed synthesis problem [69]. This problem asks whether there are strategies for components $a_1, \ldots, a_n$ of a distributed system to enforce some property given as an LTL formula ψ against all behaviours of the environment. This can be expressed by the SL_{ii} formula $\Phi_{\text{SYNTH}} := \langle\!\langle x_1 \rangle\!\rangle^{o_1} \ldots \langle\!\langle x_n \rangle\!\rangle^{o_n}(a_1, x_1) \ldots (a_n, x_n)\mathbf{A}\psi$, where o_i represents the local view of component a_i. Also, SL_{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. For instance, it can express the existence of Nash equilibria.

Main result. Of course, the high expressivity of SL_{ii} comes at a cost from a computational complexity point of view. Its satisfiability problem is undecidable (this is already true of SL), and so is its model-checking problem (this is already true of $ATL^*_{i,R}$ even for the single formula $\langle\!\langle a, b \rangle\!\rangle \mathbf{F}p$ [25], which means that agents a and b have a strategy profile to reach a situation where p holds). We mentioned that the two main settings in which decidability is retrieved for distributed synthesis are hierarchical information and public actions. We extend the first approach to the setting of strategic logics by introducing a syntactic class of "hierarchical instances" of SL_{ii}, i.e., formula/model pairs, and proving that the model-checking problem on this class of instances is decidable. Intuitively, an instance of SL_{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 straightforward to check (in linear time) whether an instance is hierarchical or not.

Applications. Because the syntax of SL_{ii} allows for arbitrary alternations of quantifiers in the formulas, our decidability result for hierarchical instances allows one to decide strategic problems more involved than module checking and distributed synthesis. For instance, we show in Section 7 how one can apply our result to establish that the existence of Nash equilibria is decidable in games with imperfect information, in the case of hierarchical observations and deterministic strategies. This problem is relevant as Nash equilibria do not always exist in games with imperfect information [30]. We then consider the problem of rational synthesis [23, 30, 33, 48], both in its cooperative and non-cooperative variants. We introduce the generalisations of these problems to the case of imperfect information, and call them cooperative and non-cooperative *rational distributed synthesis*. We then apply again our main result to establish that they are decidable in hierarchical systems for deterministic strategies. For the non-cooperative variant, we need the additional assumption that the environment is at least as informed as the system. This is the case for example when one ignores the actual observation power of the environment, and considers that it plays with perfect information. Doing so yields systems that are robust to any observation power the environment may have. As Reif puts it, this amounts to synthesising strategies that are winning even if the opponent "cheats" and uses information it is not supposed to have access to [73].

Approach. In order to solve the model-checking problem for SL_{ii} we introduce an intermediate logic $QCTL_{ii}^*$, an extension to the imperfect-information setting of $QCTL^*$ [53], 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 $QCTL_{ii}^*$ have internal structure, much like the multi-player game structures from [63] and distributed systems [39]. Model-checking $QCTL_{ii}^*$ 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 [56]). We introduce the syntactical class $QCTL_{i,\subseteq}^*$ 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. We provide a reduction from model checking SL_{ii} to model checking $QCTL_{ii}^*$ that preserves being hierarchical, thus establishing our main contribution, i.e., that model checking the hierarchical instances of SL_{ii} is decidable.

Complexity. To establish the precise complexity of the problems we solve, we introduce a new measure on formulas called *simulation depth*. This measure resembles the notion of alternation depth (see, e.g., [60]), which counts alternations between existential and universal strategy (or second-order) quantifications. But instead of merely counting alternations between such operators, simulation depth reflects the underlying automata operations required to treat formulas, while remaining a purely syntactical notion. We prove that the model-checking problem for the hierarchical fragment of $QCTL_{ii}^*$ and SL_{ii} are both $(k + 1)$-ExpTime-complete for formulas of simulation depth at most k. Already for the perfect-information fragment, this result is more precise than what was previously known. Indeed, precise upper bounds based on alternation depth were known for syntactic fragments of SL but not for the full logic [60].

Related work. The literature on imperfect information in formal methods and artificial intelligence is very vast. Imperfect information has been considered in two-player games [7, 26, 73], module checking [43, 52], distributed synthesis of reactive systems [31, 50, 69] and strategies in multiplayer games [8, 64, 65], Nash equilibria [11, 13, 72], rational synthesis [30, 38], doomsday equilibria [19], admissible strategies [14], quantitative objectives [24, 62], and more, some of which we detail below.

Limited alternation of strategy quantification was studied in [17], in which several decidability results are proved for two and three alternations of existential and universal quantifiers. Except for one where the first player has perfect information, all the problems solved in this work are hierarchical instances, and are thus particular cases of our main result.

Quantified μ-Calculus with partial observation is studied in [66], where the model-checking problem is solved by considering a syntactic constraint based on hierarchical information, as we do for $QCTL_{ii}^*$. However they consider asynchronous perfect recall, and the automata techniques they use to deal with imperfect information cannot be used in the synchronous perfect-recall setting that we consider in this work. Similarly the narrowing operation on tree automata (see Section 4.1), which is crucial in our model-checking procedure, considers synchronous perfect recall and does not seem easy to adapt to the asynchronous setting.

A number of works have considered strategic logics with imperfect information. Various semantics for ATL with imperfect information have been studied in, e.g., [41, 44]. The model-checking problem for these logics, which is undecidable for agents with perfect recall [25], has been studied for agents with bounded memory, for which decidability is recovered [58, 75]. An epistemic strategic logic with original operators different from those of ATL and SL is proposed in [40]. It considers imperfect information strategies, but only for agents without memory. Concerning perfect recall,

which interest us in this work, decidability results have also been obtained for ATL [37] and ATL with strategy context [55] when agents have the same information.

In [45], a branching-time variant of SL is extended with epistemic operators and agents with perfect recall. Strategies are not required to be uniform in the semantics, but this requirement can be expressed in the language. However no decidability result is provided. Another variant of SL extended with epistemic operators and imperfect-information, perfect-recall strategies is presented in [3], but model checking is not studied. The latter logic is extended in [4], in which its model-checking problem is solved on the class of systems where all agents' actions are public, which is an assumption orthogonal to hierarchical information.

The work closest to ours is [32] which introduces a logic CL in which one can encode many distributed synthesis problems. In this logic, hierarchical information is a necessary consequence of the syntax and semantics, and as a result its model-checking problem is decidable. However, CL is close in spirit to our $\text{QCTL}^*_{i,\subseteq}$, and its semantics is less intuitive than that of SL_{ii}. Furthermore, by means of a natural translation we derive that CL is strictly included in the hierarchical instances of SL_{ii} (Section 6.2). In particular, hierarchical instances of SL_{ii} can express non-observable goals, while CL cannot. When considering players that choose their own goals it may be natural to assume that they can observe the facts that define whether their objectives are satisfied or not. But when synthesising programs for instance, it may be enough that their behaviours enforce the desired properties, without them having the knowledge that it is enforced. Such non-observable winning conditions have been studied in, e.g., [8, 16, 24].

Outline. In Section 2 we define SL_{ii} and hierarchical instances, and present some examples. In Section 3 we define QCTL^*_{ii} and its hierarchical fragment $\text{QCTL}^*_{i,\subseteq}$. The proof that model checking $\text{QCTL}^*_{i,\subseteq}$ is decidable, including the required automata preliminaries, is in Section 4. The hierarchy-preserving translation of SL_{ii} into QCTL^*_{ii} is in Section 5. In Section 6 we compare SL_{ii} with related logics, and in Section 7 we apply our main result to obtain decidability results for various strategic problems under imperfect information. Finally we conclude and discuss future work in Section 8.

2 SL WITH IMPERFECT INFORMATION

In this section we introduce SL_{ii}, an extension of SL to the imperfect-information setting with synchronous perfect-recall. Our logic presents several original features compared to SL, which we discuss in detail in Section 2.3: we introduce an *outcome quantifier* akin to the path quantifier in branching-time temporal logics, we allow for nondeterministic strategies and unbinding agents from their strategies, and we annotate strategy quantifiers with observation symbols which denote the information available to strategies. We first fix some basic notations.

2.1 Notations

Let Σ be an alphabet. A *finite* (resp. *infinite*) *word* over Σ is an element of Σ^* (resp. Σ^ω). Words are written $w = w_0 w_1 w_2 \ldots$, i.e., indexing begins with 0. The *length* of a finite word $w = w_0 w_1 \ldots w_n$ is $|w| := n + 1$, and $\text{last}(w) := w_n$ is its last letter. Given a finite (resp. infinite) word w and $0 \le i < |w|$ (resp. $i \in \mathbb{N}$), we let w_i be the letter at position i in w, $w_{\le i}$ is the prefix of w that ends at position i and $w_{\ge i}$ is the suffix of w that starts at position i. We write $w \preccurlyeq w'$ if w is a prefix of w', and $pref(w)$ is the set of finite prefixes of word w. Finally, the domain of a mapping f is written $dom(f)$, its codomain $codom(f)$, and for $n \in \mathbb{N}$ we let $[n] := \{i \in \mathbb{N} : 1 \le i \le n\}$.

2.2 Syntax

For the rest of the paper, for convenience we fix a number of parameters for our logics and models: AP is a finite non-empty set of *atomic propositions*, Ag is a finite non-empty set of *agents* or *players*,

and Var is a finite non-empty set of *variables*. The main novelty of our logic is that we specify which information is available to a strategy, by annotating strategy quantifiers $\langle\langle x \rangle\rangle$ with *observation symbols* o from a finite set Obs, that we also fix for the rest of the paper. When we consider model-checking problems, these data are implicitly part of the input.

Definition 2.1 (SL_{ii} *Syntax).* The syntax of SL_{ii} is defined by the following grammar:

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

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

Formulas of type φ are called *state formulas*, those of type ψ are called *path formulas*, and SL_{ii} consists of all the state formulas defined by the grammar.

Boolean operators and temporal operators, $\mathbf{X}$ (read "next") and $\mathbf{U}$ (read "until"), have the usual meaning. The *strategy quantifier* $\langle\langle x \rangle\rangle^o$ is a first-order-like quantification on strategies: $\langle\langle x \rangle\rangle^o \varphi$ reads as "there exists a strategy x that takes decisions based on observation o such that φ holds", where x is a strategy variable. The *binding operator* (a, x) assigns a strategy to an agent, and $(a, x)\varphi$ reads as "when agent a plays strategy x, φ holds". The *unbinding operator* $(a, ?)$ instead releases agent a from her current strategy, if she has one, and $(a, ?)\varphi$ reads as "when agent a is not assigned any strategy, φ holds". Finally, the *outcome quantifier* $\mathbf{E}$ quantifies on outcomes of strategies currently in use: $\mathbf{E}\psi$ reads as "ψ holds in some outcome of the strategies currently used by the players".

We use abbreviations $\top := p \vee \neg p$, $\bot := \neg\top$, $\varphi \rightarrow \varphi' := \neg\varphi \vee \varphi'$, $\varphi \leftrightarrow \varphi' := \varphi \rightarrow \varphi' \wedge \varphi' \rightarrow \varphi$ for boolean connectives, $\mathbf{F}\varphi := \top\mathbf{U}\varphi$ (read "eventually φ"), $\mathbf{G}\varphi := \neg\mathbf{F}\neg\varphi$ (read "globally φ") for temporal operators, $[\![x]\!]^o \varphi := \neg\langle\langle x \rangle\rangle^o \neg\varphi$ (read "for all strategies x based on observation o, φ holds") and $\mathbf{A}\psi := \neg\mathbf{E}\neg\psi$ (read "all outcomes of the current strategies satisfy ψ").

For every formula $\varphi \in \text{SL}_{ii}$, we let *free* (φ) be the set of variables that appear free in φ, i.e., that appear out of the scope of a strategy quantifier. A formula φ is a *sentence* if *free* (φ) is empty. Finally, we let the *size* $|\varphi|$ of a formula φ be the number of symbols in φ.

2.3 Discussion on the syntax

We discuss the syntactic differences between our logic and usual Strategy Logic.

Outcome quantifier. This quantifier was introduced in Branching-time Strategy Logic (BSL) [45], which corresponds to the perfect-information fragment of the logic we define here. It removes a quirk of previous definitions, in which temporal operators could only be evaluated in contexts where all agents were assigned a strategy. The outcome quantifier, instead, allows for evaluation of temporal properties on partial assignments. As a result, the notions of free agents and agent-complete assignments from previous definitions of Strategy Logic are no longer needed (see, e.g., [60]). In addition, the outcome quantifier highlights the inherent branching-time nature of Strategy Logic: indeed, in SL, branching-time properties can be expressed by resorting to artificial strategy quantifications for all agents. It will also make the correspondence with QCTL^*_{ii} tighter, which will allow us to establish the precise complexity of the problem we solve, while the exact complexity of model checking classic SL with perfect information is still not known. Finally, since the usual definition of SL requires that the current strategies define a unique outcome on which linear-time temporal operators are evaluated, only deterministic strategies were considered. The introduction of the outcome quantifier allows us to consider nondeterministic strategies.

Unbinding. With the possibility to evaluate temporal operators even when some agents are not bound to any strategy, it becomes interesting to include the unbinding operator $(a, ?)$, introduced

in [54] for ATL with strategy context and also present in BSL. Note that the outcome quantifier and unbinding operator do not increase the expressivity of SL, at the level of sentences [45].

Observations. In games with imperfect information and ATL-like logics with imperfect information, a strategy is always bound to some player, and thus it is clear with regards to what observations it should be defined. In SL on the other hand, strategy quantification and binding are separate. This adds expressive power with regards to ATL by allowing, for instance, to assign the same strategy to two different players, but it also entails that when a quantification is made on a strategy, one does not know with regards to which observation this strategy should be defined. We know of three ways to solve this. One is the approach followed here, which consists in associating with strategy quantifiers an observation power. The second solution is to abandon the separation between quantification and binding and to use instead quantifiers of the form $\exists_a$, meaning "there exists a strategy for player a", like in [2, 21]: with this operator, the strategy is immediately bound to player a, which indicates with regards to which observation the strategy should be compatible. The third one, adopted in [4], consists in requiring that a strategy be uniform for all agents to whom it will be bound in the formula. We chose to adopt the first solution for its simplicity and expressiveness. Indeed the second solution limits expressiveness by disallowing, for instance, binding the same strategy to different agents. The third solution leads to a logic that is more expressive than the second one, but less than the first one. Indeed, the logic that we study here can capture the logic from [4] (assuming that models contain observations corresponding to unions of individual observations), and in addition SL$_{ii}$ can express changes of agents' observation power.

2.4 Semantics

The models of SL$_{ii}$ are classic concurrent game structures extended by an interpretation for observation symbols in Obs.

Definition 2.2 (CGS$_{ii}$). A *concurrent game structure with imperfect information* (or CGS$_{ii}$ for short) is a tuple $\mathcal{G} = (\mathrm{Ac}, V, E, \ell, v_\iota, O)$ where

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

For $o \in \mathrm{Obs}$, $O(o)$ is an equivalence relation on positions, that we may write $\sim_o$. It represents what a strategy with observation o can see: $O(o)$-equivalent positions are indistinguishable to such a strategy. Also, $\ell(v)$ is the set of atomic propositions that hold in position v.

We define the size $|\mathcal{G}|$ of a CGS$_{ii}$ $\mathcal{G} = (\mathrm{Ac}, V, E, \ell, v_\iota, O)$ as the size of an explicit encoding of the transition function: $|\mathcal{G}| := |V| \times |\mathrm{Ac}|^{|\mathrm{Ag}|} \times \lceil \log(|V|) \rceil$. We may write $v \in \mathcal{G}$ for $v \in V$.

We now introduce a number of notions involved in the semantics of SL$_{ii}$. Consider a CGS$_{ii}$ $\mathcal{G} = (\mathrm{Ac}, V, E, \ell, v_\iota, O)$.

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

Plays. A *finite* (resp. *infinite*) *play* is a finite (resp. infinite) word $\rho = v_0 \ldots v_n$ (resp. $\pi = v_0 v_1 \ldots$) such that $v_0 = v_\iota$ and for every i such that $0 \leq i < |\rho| - 1$ (resp. $i \geq 0$), there exists a joint action $\boldsymbol{c}$ such that $E(v_i, \boldsymbol{c}) = v_{i+1}$.

Strategies. A (nondeterministic) *strategy* is a function $\sigma : V^+ \rightarrow 2^{Ac} \setminus \emptyset$ that maps each finite play to a nonempty finite set of actions that the player may play. A strategy σ is *deterministic* if for all ρ, $\sigma(\rho)$ is a singleton. We let *Str* denote the set of all strategies.

Assignments. An *assignment* is a partial function $\chi : \text{Ag} \cup \text{Var} \rightarrow \textit{Str}$, 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 $dom(\chi) \cup \{a\}$ 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; also, $\chi[a \mapsto ?]$ is the restriction of χ to domain $dom(\chi) \setminus \{a\}$. In addition, given a formula $\varphi \in \text{SL}_{ii}$, an assignment is *variable-complete for* φ if its domain contains all free variables of φ.

Outcomes. For an assignment χ and a finite play ρ, we let $\text{Out}(\chi, \rho)$ be the set of infinite plays that start with ρ and are then extended by letting players follow the strategies assigned by χ. Formally, $\text{Out}(\chi, \rho)$ is the set of plays of the form $\rho \cdot v_1 v_2 \ldots$ such that for all $i \geq 0$, there exists c such that for all $a \in dom(\chi) \cap \text{Ag}$, $c_a \in \chi(a)(\rho \cdot v_1 \ldots v_i)$ and $v_{i+1} = E(v_i, c)$, with $v_0 = \text{last}(\rho)$.

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 $i \in \{0, \ldots, |\rho| - 1\}$.

Imperfect-information strategies. For $o \in \text{Obs}$, a strategy σ is an *o-strategy* if $\sigma(\rho) = \sigma(\rho')$ 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 $\textit{Str}_o$ be the set of all o-strategies.

Definition 2.3 (SL$_{ii}$ semantics). The semantics of a state formula is defined on a CGS$_{ii}$ $\mathcal{G}$, an assignment χ that is variable-complete for φ, and a finite play ρ. For a path formula ψ, the finite play is replaced with an infinite play π and an index $i \in \mathbb{N}$. The definition by mutual induction is as follows:

$$
\begin{aligned}
&\mathcal{G}, \chi, \rho \models p && \text{if} && p \in \ell(\text{last}(\rho)) \\
&\mathcal{G}, \chi, \rho \models \neg\varphi && \text{if} && \mathcal{G}, \chi, \rho \not\models \varphi \\
&\mathcal{G}, \chi, \rho \models \varphi \vee \varphi' && \text{if} && \mathcal{G}, \chi, \rho \models \varphi \text{ or } \mathcal{G}, \chi, \rho \models \varphi' \\
&\mathcal{G}, \chi, \rho \models \langle\!\langle x \rangle\!\rangle^o \varphi && \text{if} && \exists \sigma \in \textit{Str}_o \text{ s.t. } \mathcal{G}, \chi[x \mapsto \sigma], \rho \models \varphi \\
&\mathcal{G}, \chi, \rho \models (a, x)\varphi && \text{if} && \mathcal{G}, \chi[a \mapsto \chi(x)], \rho \models \varphi \\
&\mathcal{G}, \chi, \rho \models (a, ?)\varphi && \text{if} && \mathcal{G}, \chi[a \mapsto ?], \rho \models \varphi \\
&\mathcal{G}, \chi, \rho \models \mathbf{E}\psi && \text{if} && \text{there exists } \pi \in \text{Out}(\chi, \rho) \text{ such that } \mathcal{G}, \chi, \pi, |\rho| - 1 \models \psi \\[6pt]
&\mathcal{G}, \chi, \pi, i \models \varphi && \text{if} && \mathcal{G}, \chi, \pi_{\leq i} \models \varphi \\
&\mathcal{G}, \chi, \pi, i \models \neg\psi && \text{if} && \mathcal{G}, \chi, \pi, i \not\models \psi \\
&\mathcal{G}, \chi, \pi, i \models \psi \vee \psi' && \text{if} && \mathcal{G}, \chi, \pi, i \models \psi \text{ or } \mathcal{G}, \chi, \pi, i \models \psi' \\
&\mathcal{G}, \chi, \pi, i \models \mathbf{X}\psi && \text{if} && \mathcal{G}, \chi, \pi, i + 1 \models \psi \\
&\mathcal{G}, \chi, \pi, i \models \psi \mathbf{U}\psi' && \text{if} && \exists j \geq i \text{ s.t. } \mathcal{G}, \chi, \pi, j \models \psi' \\
& && && \text{and } \forall k \text{ s.t. } i \leq k < j, \ \mathcal{G}, \chi, \pi, k \models \psi
\end{aligned}
$$

Remark 1. Observe that because of the semantics of the outcome quantifier, and unlike usual definitions of SL, the meaning of an SL$_{ii}$ sentence depends on the assignment in which it is evaluated. For instance the SL$_{ii}$ formula $\mathbf{AF}p$ is clearly a sentence, but whether $\mathcal{G}, \chi, \rho \models \mathbf{AF}p$ holds or not depends on which agents are bound to a strategy in χ and what these strategies are. However, as usual, a sentence does not require an assignment to be evaluated, and for an SL$_{ii}$ sentence φ we let $\mathcal{G}, \rho \models \varphi$ if $\mathcal{G}, \emptyset, \rho \models \varphi$ for the empty assignment $\emptyset$, and we write $\mathcal{G} \models \varphi$ if $\mathcal{G}, v_\iota \models \varphi$.

SL is the fragment of SL_{ii} obtained by interpreting all observation symbols as the identity relation (which models perfect information), restricting to deterministic strategies, and considering only assignments in which each agent has a strategy (in this case the outcome of an assignment consists of a single play; one can thus get rid of the outcome quantifier and evaluate temporal operators in the unique outcome of the current assignment, as usually done in SL). Also, CTL^* is the fragment of SL_{ii} which uses no binding, unbinding or strategy quantification.

2.5   Discussion on the semantics

We now discuss some aspects of the semantics.

Evaluation on finite plays. Unlike previous definitions of Strategy Logic, we evaluate formulas on finite plays (instead of positions), where the finite play represents the whole history starting from the initial position of the CGS_{ii} in which the formula is evaluated. There are several reasons to do so. First, it allows us to define the semantics more simply without having to resort to the notion of assignment translations. Second, it makes it easier to see the correctness of the reduction to $QCTL^*_{ii}$, that we present in Section 5. In SL, a strategy only has access to the history of the game starting from the point where the strategy quantifier from which it arises has been evaluated. In contrast, in SL_{ii} strategies have access to the whole history, starting from the initial position. However this does not affect the semantics, in the sense that the perfect-information fragment of SL_{ii} with deterministic strategies corresponds to SL. Indeed, when agents have perfect information, having access to the past or not does not affect the existence of strategies to enforce temporal properties that only concern the future.

Players not remembering their actions. Our definition of synchronous perfect recall only considers the sequence of positions in finite plays, and forgets about actions taken by players. In particular, it is possible in this definition that a player cannot distinguish between two finite plays in which she plays different actions. This definition is standard in games with imperfect information [7, 8, 26, 80], since remembering one's actions or not is indifferent for the existence of distributed winning strategies or Nash equilibria. However it makes a difference for some more involved solution concepts that are expressible in strategic logics such as SL_{ii}. For instance it is observed in [10, Appendix A] that some games admit subgame-perfect equilibria only if agents remember their own past actions. Nonetheless we consider the setting where agents do not remember their actions, as it is the most general. Indeed, as noted in [18, Remark 2.1, p.8], one can simulate agents that remember their own actions by storing in positions of the game the information of the last joint move played (this may create $|Ac|^{|Ag|}$ copies of each position, but the branching degree is unchanged). One can then adapt indistinguishability relations to take actions into account. For instance, for an observation symbol o and an agent a, one could consider a new observation symbol o_a that would be interpreted in the enriched game structure as the refinement of $\sim_o$ that considers two positions indistinguishable if they are indistinguishable for $\sim_o$ and contain the same last action for agent a. Binding agent a only to strategies that use observation of the form o_a for some o captures the fact that agent a remembers her actions.

Agents changing observation. In SL_{ii} observations are not bound to agents but to strategies. And because agents can change their strategy thanks to the binding operator, it follows that they can change observation, or more precisely they can successively play with strategies that have different observations. For instance consider a controller that observes a system through a set of n sensors $S = \{s_1, \ldots, s_n\}$ as in, e.g., [9]. Let o_i be the observation power provided by the set of sensors $S \setminus \{s_i\}$ (one can think of a system where states are tuples of local states, each sensor observing one component). Also let o be the observation power provided by the full set S of sensors,

and let atom fault$_i$ represent the fact that a fault occurs on sensor s_i. The formula

$$\varphi := \langle\!\langle x \rangle\!\rangle^o(a, x) \mathbf{AG} \left(\text{safe} \wedge \bigwedge_{i=1}^{n} \text{fault}_i \rightarrow \langle\!\langle x \rangle\!\rangle^{o_i}(a, x) \mathbf{AG} \, \text{safe}_i \right)$$

expresses that the controller a has a strategy (which uses all sensors in S) to maintain the system safe, and if a sensor is lost, it can respond by switching to a strategy using the remaining sensors to maintain some alternative, possibly weaker, security requirement safe$_i$.

2.6 Model checking and hierarchical instances

We now introduce the main decision problem of this paper, which is the model-checking problem for SL$_{ii}$. An SL$_{ii}$-*instance* is a model together with a formula, i.e., it is a pair $(\mathcal{G}, \Phi)$ where $\mathcal{G}$ is a CGS$_{ii}$ and $\Phi \in \text{SL}_{ii}$.

Definition 2.4 (Model checking SL$_{ii}$). The *model-checking problem* for SL$_{ii}$ is the decision problem that, given an SL$_{ii}$-instance $(\mathcal{G}, \Phi)$, 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 [63]. Since this problem is easily reduced to the model-checking problem for SL$_{ii}$, we get the following result.

THEOREM 2.5. *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 $(\mathcal{G}, \Phi)$ is hierarchical if, as one goes down a path in the syntactic tree of Φ, the observations tied to quantifications become finer.

Definition 2.6 (Hierarchical instances). An SL$_{ii}$-instance $(\mathcal{G}, \Phi)$ is *hierarchical* if for every subformula $\varphi_1 = \langle\!\langle y \rangle\!\rangle^{o_1} \varphi_1'$ of Φ and subformula $\varphi_2 = \langle\!\langle x \rangle\!\rangle^{o_2} \varphi_2'$ of φ_1', it holds that $O(o_2) \subseteq O(o_1)$.

If $O(o_2) \subseteq O(o_1)$ we say that o_2 is *finer* than o_1 in $\mathcal{G}$, and that o_1 is *coarser* than o_2 in $\mathcal{G}$. Intuitively, this means that a player with observation o_2 observes game $\mathcal{G}$ no worse than, i.e., knows at least as much as a player with observation o_1.

Remark 2. If one uses the trick described in Section 2.5 to model agents that remember their own actions, then for an agent a to know at least as much as another agent b it needs to be the case that, in particular, agent a observes all actions played by agent b.

Example 2.7 (Fault-tolerant diagnosibility). Consider the following formula from Section 2.5:

$$\varphi := \langle\!\langle x \rangle\!\rangle^o(a, x) \mathbf{AG} \left(\text{safe} \wedge \bigwedge_{i=1}^{n} \text{fault}_i \rightarrow \langle\!\langle x \rangle\!\rangle^{o_i}(a, x) \mathbf{AG} \, \text{safe}_i \right)$$

As already discussed, it expresses that the controller can react to the loss of a sensor to keep ensuring some property of the system. Clearly, the controller's observation o_i after the loss of sensor i is coarser than its original observation o, and thus formula φ in such a system does not form a hierarchical instance.

We now give an example of scenario where hierarchical instances occur naturally.

Example 2.8 (Security levels). Consider a system with different "security levels", where higher levels have access to more data (i.e., can observe more). Assume that the CGS$_{ii}$ $\mathcal{G}$ is such that $O(o_n) \subseteq O(o_{n-1}) \subseteq \ldots \subseteq O(o_1)$: in other words, level n has the highest security clearance, while level 1 has the lowest. Consider that agent a wants to reach some objective marked by atom "goal", that it starts with the lowest observation clearance o_1, and that atomic formula "promote$_i$" means

that the agent is granted access to level i (observe that whenever we have promote$_i$, we should also have promote$_j$ for all $j < i$). For every i we let

$$\varphi_i(\varphi') := \text{goal} \vee (\text{promote}_i \wedge \langle\!\langle x \rangle\!\rangle^{o_i}(a, x)\mathbf{AF}\varphi')$$

Now the formula

$$\varphi := \varphi_1(\varphi_2(\ldots \varphi_{n-1}(\varphi_n(\text{goal}))\ldots))$$

means that agent a can enforce her goal, possibly by first getting access to higher security levels and using this additional observation power to reach the goal. Because the strategy quantifications that are deeper in the formula have access to more information, this formula forms a hierarchical instance in $\mathcal{G}$.

Here is the main contribution of this work:

THEOREM 2.9. *The model-checking problem for* SL$_{ii}$ *restricted to the class of hierarchical instances is decidable.*

We prove this result in Section 5 by reducing it to the model-checking problem for the hierarchical fragment of a logic called QCTL* with imperfect information, which we now introduce and study in order to use it as an intermediate, "low-level" logic between tree automata and SL$_{ii}$. We then discuss some applications of this theorem in Section 7.

3 QCTL* WITH IMPERFECT INFORMATION

In this section we introduce an imperfect-information extension of QCTL* [34, 46, 47, 53, 77], which is an extension of CTL* with second-order quantification on atomic propositions. 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 [63] and distributed systems [39], and we find it very suitable to application of automata techniques, as discussed in Section 3.3.

The syntax of QCTL$_{ii}^*$ 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 QCTL$_{ii}^*$: this means that we interpret formulas on trees that are the unfoldings of Kripke structures (this will capture the fact that players in SL$_{ii}$ 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.

For the rest of the section we fix some natural number $n \in \mathbb{N}$ which parameterises the logic QCTL$_{ii}^*$, and which is the number of components in states of the models.

3.1 QCTL$_{ii}^*$ Syntax

The syntax of QCTL$_{ii}^*$ 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.

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

Definition 3.1 (QCTL$_{ii}^*$ Syntax). The syntax of QCTL$_{ii}^*$ 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 $p \in \text{AP}$ and $\mathbf{o} \subseteq [n]$.

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

Given a QCTL^*_{ii} formula φ, we define the set of *quantified propositions* $\text{AP}_{\exists}(\varphi) \subseteq \text{AP}$ 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* $\text{AP}_f(\varphi) \subseteq \text{AP}$ as the set of atomic propositions that have an occurrence which is not under the scope of any quantifier of the form $\exists^{\mathbf{o}}p$. Observe that $\text{AP}_{\exists}(\varphi) \cap \text{AP}_f(\varphi)$ may not be empty, i.e., a proposition may appear both free and quantified in (different places of) a formula.

3.2 QCTL^*_{ii} semantics

Several semantics have been considered for QCTL^*, the two most studied being the *structure semantics* and the *tree semantics* (see [53] for more details). For the semantics of QCTL^*_{ii} we adapt the tree semantics, and we explain the reasons for doing so in Section 3.3.

As already mentioned, for QCTL^*_{ii} we consider structures whose states are tuples of local states. We now define these structures and related notions.

Definition 3.2 (Compound Kripke structures). A *compound Kripke structure*, or CKS, over AP is a tuple $\mathcal{S} = (S, R, \ell, s_\iota)$ where

- $S \subseteq \prod_{i\in[n]} L_i$ is a set of *states*, with $\{L_i\}_{i\in[n]}$ a family of n disjoint finite sets of *local states*,
- $R \subseteq S \times S$ is a left-total[1] *transition relation*,
- $\ell : S \to 2^{\text{AP}}$ is a *labelling function* and
- $s_\iota \in S$ is an *initial state*.

A *path* in $\mathcal{S}$ is an infinite sequence of states $\lambda = s_0 s_1 \ldots$ such that for all $i \in \mathbb{N}$, $(s_i, s_{i+1}) \in R$. A *finite path* is a finite non-empty prefix of a path. We may write $s \in \mathcal{S}$ for $s \in S$, and we define the *size* $|\mathcal{S}|$ of a CKS $\mathcal{S} = (S, R, s_\iota, \ell)$ as its number of states: $|\mathcal{S}| := |S|$.

Since we will interpret QCTL^*_{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 of *directions* (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 $u \cdot x \neq r$, then $u \in \tau$,
- 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 for every $u \in \tau$ and $x \in X$, $u \cdot x \in \tau$. 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.

Labellings. 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 $\ell : \tau \to 2^{\text{AP}}$ is a *labelling*, which maps each node to the set of propositions that hold there. For $p \in \text{AP}$, a p-*labelling* for a tree is a mapping $\ell_p : \tau \to \{0, 1\}$ that indicates in which nodes p holds, and for a labelled tree $t = (\tau, \ell)$, the p-labelling of t is the p-labelling $u \mapsto 1$ if $p \in \ell(u)$, 0 otherwise. The composition of a labelled tree $t = (\tau, \ell)$ with a p-labelling ℓ_p for τ is defined as $t \otimes \ell_p := (\tau, \ell')$, where $\ell'(u) = \ell(u) \cup \{p\}$ if $\ell_p(u) = 1$, and $\ell(u) \setminus \{p\}$ otherwise. A p-labelling for a labelled tree $t = (\tau, \ell)$ is a p-labelling for its domain τ. A *pointed labelled tree* is a pair (t, u) where u is a node of t.

---

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

If $u = w \cdot x$, the *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 $\ell_u(x \cdot w') = \ell(w \cdot x \cdot w')$. A labelled tree is *regular* if it has finitely many disctinct subtrees.

In the tree semantics of QCTL$^*_{\text{ii}}$ that we consider here, formulas are evaluated on tree unfoldings of CKS, which we now define.

Tree unfoldings. Let $\mathcal{S} = (S, R, \ell, s_\iota)$ be a compound Kripke structure over AP. The *tree-unfolding of $\mathcal{S}$* is the (AP, S)-tree $t_{\mathcal{S}} := (\tau, \ell')$, where τ is the set of all finite paths that start in s_ι, and for every $u \in \tau$, $\ell'(u) := \ell(\text{last}(u))$.

Note that a labelled tree is regular if and only if it is the unfolding of some finite Kripke structure.

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 and trees over $X \times Y$ (point-wise).

Given a family of (disjoint) sets of local states $\{L_i\}_{i \in [n]}$ and a subset $I \subseteq [n]$, 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. For $I, J \subseteq [n]$ and $z \in L_I$, we also define $z{\downarrow}_J := z{\downarrow}_{L_{I \cap J}}$, where z is seen as a pair $z = (x, y) \in L_{I \cap J} \times L_{I \setminus J}$, i.e., we apply the above definition with $X = L_{I \cap J}$ and $Y = L_{I \setminus J}$. 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. In particular, for every L_I-tree τ, $\tau{\downarrow}_\emptyset$ is the only $L_\emptyset$-tree, $\mathbf{0}^\omega$.

Quantification and uniformity. In QCTL$^*_{\text{ii}}$ $\exists^{\mathbf{o}} p. \varphi$ holds in a tree t if there is some $\mathbf{o}$-uniform p-labelling of t such that t with this p-labelling satisfies φ. Intuitively, a p-labelling of a tree is $\mathbf{o}$-uniform if every two nodes that are indistinguishable for observation $\mathbf{o}$ agree on p.

Definition 3.3 ($\mathbf{o}$-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 $x{\downarrow}_{\mathbf{o}} = x'{\downarrow}_{\mathbf{o}}$.
- Two words $u = u_0 \ldots u_i$ and $u' = u'_0 \ldots u'_j$ over alphabet L_I are *$\mathbf{o}$-indistinguishable*, written $u \approx_{\mathbf{o}} u'$, if $i = j$ and for all $k \in \{0, \ldots, i\}$ we have $u_k \approx_{\mathbf{o}} u'_k$.
- A p-labelling for a tree τ is *$\mathbf{o}$-uniform* if for all $u, u' \in \tau$, $u \approx_{\mathbf{o}} u'$ implies $\ell_p(u) = \ell_p(u')$.

Definition 3.4 (QCTL$^*_{\text{ii}}$ semantics). We define by induction the satisfaction relation $\models$ of QCTL$^*_{\text{ii}}$. Let $t = (\tau, \ell)$ be an AP-labelled L_I-tree, u a node and λ a path in τ:

$$
\begin{aligned}
t, u &\models p && \text{if} && p \in \ell(u) \\
t, u &\models \neg\varphi && \text{if} && t, u \not\models \varphi \\
t, u &\models \varphi \vee \varphi' && \text{if} && t, u \models \varphi \text{ or } t, u \models \varphi' \\
t, u &\models \mathbf{E}\psi && \text{if} && \exists \lambda \in \mathit{Paths}(u) \text{ s.t. } t, \lambda \models \psi \\
t, u &\models \exists^{\mathbf{o}} p. \varphi && \text{if} && \exists \ell_p \text{ a } \mathbf{o}\text{-uniform } p\text{-labelling for } t \text{ such that } t \otimes \ell_p, u \models \varphi \\
t, \lambda &\models \varphi && \text{if} && t, \lambda_0 \models \varphi \\
t, \lambda &\models \neg\psi && \text{if} && t, \lambda \not\models \psi \\
t, \lambda &\models \psi \vee \psi' && \text{if} && t, \lambda \models \psi \text{ or } t, \lambda \models \psi' \\
t, \lambda &\models \mathbf{X}\psi && \text{if} && t, \lambda_{\geq 1} \models \psi \\
t, \lambda &\models \psi \mathbf{U}\psi' && \text{if} && \exists i \geq 0 \text{ s.t. } t, \lambda_{\geq i} \models \psi' \text{ and } \forall j \text{ s.t. } 0 \leq j < i, \ t, \lambda_{\geq j} \models \psi
\end{aligned}
$$

We write $t \models \varphi$ for $t, r \models \varphi$, where r is the root of t. Given a CKS $\mathcal{S}$ and a QCTL$^*_{\text{ii}}$ formula φ, we also write $\mathcal{S} \models \varphi$ if $\mathcal{S}, s_\iota \models \varphi$.

Example 3.5. 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 QCTL$^*_{\text{ii}}$ 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 QCTL$_{ii}^*$ with tree semantics subsumes MSO with equal level [27, 56, 78]. In Theorem 3.7 we make use of a similar observation to prove that model-checking QCTL$_{ii}^*$ is undecidable.

3.3 Discussion on the definition of QCTL$_{ii}^*$

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

Modelling of imperfect information. We model imperfect information by means of local states (rather than equivalence relations) because this greatly facilitates the use of automata techniques. More precisely, in our decision procedure of Section 4 we use an operation on tree automata called *narrowing*, which was introduced in [49] 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 $X \times Y$ (intuitively, this widening is the $X \times Y$-tree in which each node is labelled as its projection on the original X-tree; see Section 4 for details).

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. The two most studied semantics for QCTL* 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 *memoryful* strategies. Since in this work we are interested in perfect-recall strategies, we only consider the tree semantics.

3.4 Model checking QCTL$_{ii}^*$

We now define the model-checking problem studied in the rest of this section.

Definition 3.6 (Model checking QCTL$_{ii}^*$). The *model-checking problem for* QCTL$_{ii}^*$ is the following decision problem: given an instance $(\mathcal{S}, \Phi)$ where $\mathcal{S}$ is a CKS, and Φ is a QCTL$_{ii}^*$ formula, return 'Yes' if $\mathcal{S} \models \Phi$ and 'No' otherwise.

We now prove that the model-checking problem for QCTL$_{ii}^*$ is undecidable. This comes as no surprise since, as we will show in Section 5, QCTL$_{ii}^*$ can express the existence of distributed winning strategies in imperfect-information games. However we propose a proof that shows the connection between QCTL$_{ii}^*$ and MSO with equal-level predicate [27, 56, 78]. This proof also has the benefit of showing that QCTL$_{ii}^*$ is undecidable already for formulas that involve only propositional quantifiers that observe either everything or nothing.

THEOREM 3.7. *The model-checking problem for* QCTL$_{ii}^*$ *is undecidable.*

PROOF. Let $\mathrm{MSO_{eq}}$ denote the extension of the logic MSO (without unary predicates) by a binary predicate symbol eq. $\mathrm{MSO_{eq}}$ 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 $\mathrm{MSO_{eq}}$ into $\mathrm{QCTL^*_{ii}}$, and our result follows since the $\mathrm{MSO_{eq}}$-theory of the binary tree is undecidable [56]. The translation from $\mathrm{MSO_{eq}}$ to $\mathrm{QCTL^*_{ii}}$ is obtained by extending that from MSO to QCTL [53], using the formula $\mathbf{level}(\cdot)$ from Example 3.5 to help capture the equal-length predicate.

We define a translation $\widehat{}$ from $\mathrm{MSO_{eq}}$ to $\mathrm{QCTL^*_{ii}}$ such that for every tree t with root r, nodes $u_1, \ldots, u_i \in t$ and sets of nodes $U_1, \ldots, U_j \subseteq t$, and every $\mathrm{MSO_{eq}}$ formula $\varphi(x, x_1, \ldots, x_i, X_1, \ldots, X_j)$, we have that

$$t, r, u_1, \ldots, u_i, U_1, \ldots, U_j \models \varphi(x, x_1, \ldots, x_i, X_1, \ldots, X_j) \quad \text{if and only if} \quad \widehat{t}, r \models \widehat{\varphi} \qquad (1)$$

where $\widehat{t}$ is obtained from t by defining the labelling for fresh atomic propositions p_{x_k} and p_{X_k}, with $k \in [i]$, as follows: $p_{x_k} \in \widehat{\ell}(u)$ if $u = u_k$ and $p_{X_k} \in \widehat{\ell}(u)$ if $u \in U_k$.

The translation of MSO to $\mathrm{QCTL^*}$ from [53] can be extended to one from $\mathrm{MSO_{eq}}$ to $\mathrm{QCTL^*_{ii}}$ by adding rules for the equal level predicate. Indeed, for $\varphi(x, x_1, \ldots, x_i, X_1, \ldots, X_j) \in \mathrm{MSO_{eq}}$, we inductively define the $\mathrm{QCTL^*_{ii}}$ formula $\widehat{\varphi}$ as follows, where $k \in [i]$:

$$
\begin{aligned}
\widehat{x = x_k} &:= p_{x_k} & \widehat{x_k = x_l} &:= \mathbf{EF}(p_{x_k} \wedge p_{x_l}) \\
\widehat{x \in X_k} &:= p_{X_k} & \widehat{x_k \in X_l} &:= \mathbf{EF}(p_{x_k} \wedge p_{X_l}) \\
\widehat{\neg \varphi'} &:= \neg \widehat{\varphi'} & \widehat{\varphi_1 \vee \varphi_2} &:= \widehat{\varphi_1} \vee \widehat{\varphi_2} \\
\widehat{\exists x_k . \varphi'} &:= \exists p_{x_k}. \, \mathrm{uniq}(p_{x_k}) \wedge \widehat{\varphi'} \\
\widehat{\exists X_k . \varphi'} &:= \exists p_{X_k}. \, \widehat{\varphi'} \\
\widehat{S(x, x_k)} &:= \mathbf{EX} p_{x_k} & \widehat{S(x_k, x)} &:= \bot \\
\widehat{S(x_k, x_l)} &:= \mathbf{EF}(p_{x_k} \wedge \mathbf{EX} p_{x_l})
\end{aligned}
$$

where $\mathrm{uniq}(p) := \mathbf{EF} p \wedge \forall q. \ (\mathbf{EF}(p \wedge q) \rightarrow \mathbf{AG}(p \rightarrow q))$ holds in a tree iff it has exactly one node labelled with p. To understand the $x = x_k$ and $x \in X_k$ cases, consider that x will be interpreted as the root. For the $S(x_k, x)$ case, observe that x has no incoming edge since it is interpreted as the root. Second-order quantification $\exists X_k$ is translated into quantification on atomic proposition p_{X_k}, and first-order quantification $\exists x_k$ is treated similarly, with the additional constraint that quantification is limited to p_{x_k}-labellings that set p_{x_k} to true in one and only one node of the tree.

The rules for eq are as follows:

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

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. For the second case, recall from Example 3.5 that the $\mathrm{QCTL^*_{ii}}$ 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. The correctness of the translation follows from (1), which is proven by induction.

Now take an instance $(t, \varphi(x))$ of the model-checking problem for $\mathrm{MSO_{eq}}$ on the full binary tree t. 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, $t_{\mathcal{S}} = t$, and applying (1) we get:

$$t, s_0 \models \varphi(x) \quad \text{iff} \quad \widehat{t}, s_0 \models \widehat{\varphi}.$$

Observe that in the previous line, because there are no free variables besides x, which stands for the root, we have that $\widehat{t} = t = t_S$, hence we have indeed produced an instance of the model-checking problem for $\text{QCTL}^*_{\text{ii}}$. □

4 A DECIDABLE FRAGMENT OF $\text{QCTL}^*_{\text{ii}}$: HIERARCHY ON OBSERVATIONS

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

Definition 4.1 (Hierarchical formulas). A $\text{QCTL}^*_{\text{ii}}$ formula φ is *hierarchical* if for all subformulas $\varphi_1 = \exists^{\mathbf{o}_1} p_1. \varphi_1'$ and $\varphi_2 = \exists^{\mathbf{o}_2} p_2. \varphi_2'$ of φ 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 innermore propositional quantifiers observe at least as much as outermore ones.

Example 4.2. Formula $\exists^{\{1,2\}} p. \exists^{\{1,2,4\}} q. \mathbf{AG}(p \vee q)$ is hierarchical because $\{1,2\} \subseteq \{1,2,4\}$. On the other hand, formula $\exists^{\{1,2\}} p. (\exists^{\{1,2,4\}} q. \mathbf{AG}(p \vee q) \wedge \exists^{\{3\}} q'. \mathbf{EF}(p \wedge q'))$ is not, because $\{1,2\} \nsubseteq \{3\}$. Note that neither is it the case that $\{3\} \subseteq \{1,2\}$: the observation power of quantifiers $\exists^{\{1,2\}} p.$ and $\exists^{\{3\}} q'.$ are incomparable. Finally, formula $\forall^{\{1,2,3\}} p. \exists^{\{1,2\}} q. .\mathbf{AG}(p \vee q)$ is not hierarchical even though $\{1,2\} \subseteq \{1,2,3\}$, as the quantifier that observes best is *higher* in the syntactic tree.

We let $\text{QCTL}^*_{\text{i},\subseteq}$ be the set of hierarchical $\text{QCTL}^*_{\text{ii}}$ formulas.

THEOREM 4.3. *Model checking $\text{QCTL}^*_{\text{i},\subseteq}$ is non-elementary decidable.*

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

4.1 Alternating parity tree automata

We recall alternating parity tree automata. Because their semantics is defined via acceptance games, we start with basic definitions for two-player turn-based parity games, or simply parity games.

Parity games. A *parity game* is a structure $\mathcal{G} = (V, E, v_\iota, C)$, where $V = V_E \uplus V_A$ is a set of *positions* partitioned between positions of Eve (V_E) and those of Adam (V_A), $E \subseteq V \times V$ is a set of *moves*, v_ι is an initial position and $C : V \to \mathbb{N}$ is a colouring function of finite codomain. In positions V_E, Eve chooses the next position, while Adam chooses in positions V_A. A play is an infinite sequence of positions $v_0 v_1 v_2 \ldots$ such that $v_0 = v_\iota$ and for all $i \geq 0$, $(v_i, v_{i+1}) \in E$ (written $v_i \to v_{i+1}$). We assume that for every $v \in V$ there exists $v' \in V$ such that $v \to v'$. A strategy for Eve is a partial function $V^* \to V$ that maps each finite prefix of a play ending in a position $v \in V_E$ to a next position v' such that $v \to v'$. A play $v_0 v_1 v_2 \ldots$ *follows* a strategy σ of Eve if for every $i \geq 0$ such that $v_i \in V_E$, $v_{i+1} = \sigma(v_0 \ldots v_i)$. A strategy σ is winning if every play that follows it satisfies the parity condition, i.e., the least colour seen infinitely often along the play is even.

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, $\delta : Q \times 2^{\text{AP}} \to \mathbb{B}^+(X \times Q)$ 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 $a \in 2^{\text{AP}}$, $\delta(q, a)$ is written in disjunctive normal form and for every direction $x \in X$ each disjunct contains exactly one element of $\{x\} \times Q$. An NTA is *deterministic* if for each $q \in Q$ and $a \in 2^{\text{AP}}$, $\delta(q, a)$ consists of a single disjunct.

Acceptance of a pointed labelled tree (t, u_ι), where $t = (\tau, \ell)$, by an ATA $\mathcal{A} = (Q, \delta, q_\iota, C)$ is defined via the parity game $\mathcal{G}(\mathcal{A}, t, u_\iota) = (V, E, v_\iota, C')$ where $V = \tau \times Q \times \mathbb{B}^+(X \times Q)$, position (u, q, α) belongs to Eve if α is of the form $\alpha_1 \vee \alpha_2$ or $[x, q']$, and to Adam otherwise, $v_\iota = (u_\iota, q_\iota, \delta(q_\iota, u_\iota))$, and $C'(u, q, \alpha) = C(q)$. Moves in $\mathcal{G}(\mathcal{A}, t, u_\iota)$ are defined by the following rules:

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

Positions of the form $(u, q, \top)$ and $(u, q, \bot)$ are sinks, winning for Eve and Adam respectively.

A pointed labelled tree (t, u) is *accepted* by $\mathcal{A}$ if Eve has a winning strategy in $\mathcal{G}(\mathcal{A}, t, u)$, and the *language* of $\mathcal{A}$ is the set of pointed labelled trees accepted by $\mathcal{A}$, written $\mathcal{L}(\mathcal{A})$. We write $t \in \mathcal{L}(\mathcal{A})$ if $(t, r) \in \mathcal{L}(\mathcal{A})$, where r is the root of t. Finally, the *size* $|\mathcal{A}|$ of an ATA $\mathcal{A}$ is its number of states plus the sum of the sizes of all formulas appearing in the transition function.

Word automata. When the set of directions X is a singleton, directions can be forgotten and infinite trees can be identified with infinite words. We thus call *parity word automaton* a parity tree automaton on (AP, X)-trees where X is a singleton. In the case of a nondeterministic parity word automaton, transitions can be represented as usual as a mapping $\Delta : Q \times 2^{AP} \rightarrow 2^Q$ which, in a state $q \in Q$, reading the label $a \in 2^{AP}$ of the current position in the word, indicates a set of states $\Delta(q, a)$ from which Eve can choose to send in the next position of the word.

We recall four classic operations on tree automata.

Complementation. Given an ATA $\mathcal{A} = (Q, \delta, q_\iota, C)$, we define its *dual* $\overline{\mathcal{A}} = (Q, \overline{\delta}, q_\iota, \overline{C})$ where, for each $q \in Q$ and $a \in 2^{AP}$, $\overline{\delta}(q, a)$ is the dual of $\delta(q, a)$, i.e., conjunctions become disjunctions and vice versa, and $C(q) := C(q) + 1$.

THEOREM 4.4 (COMPLEMENTATION [61]). *For every labelled tree t and node u in t,*

$$(t, u) \in \mathcal{L}(\overline{\mathcal{A}}) \text{ if, and only if, } (t, u) \notin \mathcal{L}(\mathcal{A}).$$

Projection. The second construction is a projection operation, used by Rabin to deal with second-order monadic quantification:

THEOREM 4.5 (PROJECTION [71]). *Given an NTA $\mathcal{N}$ on (AP, X)-trees and an atomic proposition $p \in$ AP, one can build in linear time an NTA $\mathcal{N} \Downarrow_p$ on (AP $\setminus \{p\}, X$)-trees such that*

$$(t, u) \in \mathcal{L}(\mathcal{N} \Downarrow_p) \quad \text{iff} \quad \text{there exists a } p\text{-labelling } \ell_p \text{ for } t \text{ s.t. } (t \otimes \ell_p, u) \in \mathcal{L}(\mathcal{N}).$$

Intuitively, $\mathcal{N} \Downarrow_p$ is automaton $\mathcal{N}$ with the only difference that when it reads the label of a node, it can choose to run as if p was either true or false: 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 $a \in 2^{AP}$. Another way of seeing it is that $\mathcal{N} \Downarrow_p$ guesses a p-labelling for the input tree, and simulates $\mathcal{N}$ on this modified input.

Simulation. 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 following result:

THEOREM 4.6 (SIMULATION [61]). *Given an ATA $\mathcal{A}$, one can build in exponential time an NTA $\mathcal{N}$ such that $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{A})$.*

The last construction was introduced by Kupferman and Vardi to deal with imperfect information aspects in distributed synthesis. To describe it we need to define a widening operation on trees which expands the directions in a tree.

Tree widening. We generalise the widening operation defined in [49]. In the following definitions we fix a CKS $\mathcal{S} = (S, R, s_\iota, \ell)$, and for $I \subseteq [n]$ we let $S_I := \{s \downarrow_I | s \in S\} \subseteq L_I$ (recall that $L_I = \prod_{i \in I} L_i$). Let $J \subseteq I \subseteq [n]$. For every S_J-tree τ rooted in s_J and $s_I \in S_I$ such that $s_I \downarrow_J = s_J$, we define the I-widening of τ as the S_I-tree

$$\tau \uparrow^I_{s_I} := \{u \in s_I \cdot S_I^* \mid u \downarrow_J \in \tau\}.$$

For an (AP, S_J)-tree $t = (\tau, \ell)$ rooted in s_J and $s_I \in S_I$ such that $s_I \downarrow_J = s_J$, we let

$$t \uparrow^I_{s_I} := (\tau \uparrow^I_{s_I}, \ell'), \text{ where } \ell'(u) := \ell(u \downarrow_J).$$

When clear from the context we may omit the subscript s_I. It is the case in particular when referring to *pointed* widenings of trees: $(t \uparrow^I, u)$ stands for $(t \uparrow^I_{u_0}, u)$.

Narrowing. We now state a result from [49] in our slightly more general setting (the proof can be adapted straightforwardly). The rough idea of this narrowing operation on ATA is that, if one just observes S_J, uniform p-labellings on S_I-trees can be obtained by choosing the labellings directly on S_J-trees, and then lifting them to S_I.

THEOREM 4.7 (NARROWING [49]). *Given an ATA $\mathcal{A}$ on S_I-trees one can build in linear time an ATA $\mathcal{A} \downarrow_J$ on S_J-trees such that for every pointed (AP, S_J)-tree (t, u) and every $u' \in S_I^+$ such that $u' \downarrow_J = u$,*

$$(t, u) \in \mathcal{L}(\mathcal{A} \downarrow_J) \text{ iff } (t \uparrow^I, u') \in \mathcal{L}(\mathcal{A}).$$

4.2 Translating $\text{QCTL}^*_{i,\subseteq}$ to ATA

In order to prove Theorem 4.3 we need some more notations and a technical lemma that contains the automata construction.

Definition 4.8. For every $\varphi \in \text{QCTL}^*_{ii}$, we let

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

where $\text{Obs}(\varphi)$ is the set of concrete observations that occur in φ, with the intersection over the empty set defined as $[n]$. For a CKS $\mathcal{S}$ with state set $S \subseteq \prod_{i \in [n]} L_i$ we also let $S_\varphi := \{s \downarrow_{I_\varphi} | s \in S\}$.

Elements of S_φ will be the possible directions used by the automaton we build for φ. In other words, the automaton for φ will work on S_φ-trees. The intuition is that the observations in φ determine which components of the model's states can be observed by the automaton.

Our construction, that transforms a $\text{QCTL}^*_{i,\subseteq}$ formula φ and a CKS $\mathcal{S}$ into an ATA, builds upon the classic construction from [51], which builds ATA for CTL* formulas. In addition, we use projection of automata to treat second-order quantification, and to deal with imperfect information we resort to automata narrowing.

Moreover, we use tree automata in an original way that allows us to deal with non-observable atomic propositions, which in turn makes it possible to consider non-observable winning conditions in our decidable fragment of SL_{ii}. 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 [51]. We instead encode the model in the automata, using the input tree only to guess labellings for quantified propositions.

Encoding the model in the automaton. Quantification on atomic propositions is classically performed by means of automata projection (see Theorem 4.5). 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 narrowing operation (see Theorem 4.7). 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 is that unless $t = (\tau, \ell)$ is $\mathbf{o}$-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, unless we restrict to models where all atomic propositions are observable for all observations $\mathbf{o}$, we cannot pass the model as input to our automata, which will work on narrowings of trees.

Therefore, to model check a QCTL$_{\text{ii}}^*$ formula φ on a CKS $\mathcal{S}$, each state of the automaton that we build for φ will contain a state of $\mathcal{S}$. The automaton 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 no longer represents the model, but we use it to carry labellings for quantified atomic propositions in 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 Coordination Logic [32]: by separating between quantified atomic propositions (that need to be uniform and are carried by the input tree) and free atomic propositions (that state facts about the model and are coded in the automaton), we manage to remove the restriction present in CL, that requires all facts about the model to be known to every strategy (see Proposition 6.3 in Section 6.2). To do this we assume without loss of generality that propositions that are quantified in φ do not appear free in φ, i.e., AP$_\exists(\varphi) \cap$ AP$_f(\varphi) = \emptyset$.

Finally, given a formula φ, 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 atoms in AP$_\exists(\varphi)$, which can thus be forgotten. Thus, from now on we will assume that an instance $(\mathcal{S}, \Phi)$ of the model-checking problem for QCTL$_{\text{ii}}^*$ is such that AP$_\exists(\Phi) \cap$ AP$_f(\Phi) = \emptyset$ and $\mathcal{S}$ is a CKS over AP$_f(\Phi)$.

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 4.9 (Merge). Let $t = (\tau, \ell)$ be a complete (AP, X)-tree and $t' = (\tau', \ell')$ an (AP$'$, X)-tree with same root as t, where AP $\cap$ AP$' = \emptyset$. We define the *merge* of t and t' as the (AP $\cup$ AP$'$, X)-tree

$$t \barwedge t' := (\tau \cap \tau' = \tau', \ell''),$$

where $\ell''(u) = \ell(u) \cup \ell'(u)$.

We now describe our automata construction. Let $(\mathcal{S}, \Phi)$ be an instance of the model-checking problem for QCTL$_{\text{i},\subseteq}^*$, where $\mathcal{S} = (S, R, \ell_{\mathcal{S}}, s_\iota)$.

LEMMA 4.10 (TRANSLATION). *For every subformula φ of Φ and state s of $\mathcal{S}$, one can build an ATA $\mathcal{A}_s^\varphi$ on (AP$_\exists(\Phi)$, S_φ)-trees such that for every (AP$_\exists(\Phi)$, S_φ)-tree t rooted in $s_\iota \downarrow_{I_\varphi}$, every $u \in t_{\mathcal{S}}$ ending in s, it holds that*

$$(t, u \downarrow_{I_\varphi}) \in \mathcal{L}(\mathcal{A}_s^\varphi) \quad \text{iff} \quad t \uparrow^{[n]} \barwedge t_{\mathcal{S}}, u \models \varphi.$$

PROOF. Let AP$_\exists =$ AP$_\exists(\Phi)$ and AP$_f =$ 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 (AP$_\exists$, S_φ)-trees.

$\varphi = p$: First, by Definition 4.8, $S_\varphi = S_{[n]} = S$. We let $\mathcal{A}_s^p$ be the ATA over S-trees with one unique state q_ι, with transition function defined as follows:

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

$\varphi = \neg\varphi'$: We let $\mathcal{A}_s^\varphi := \overline{\mathcal{A}_s^{\varphi'}}$.

$\varphi = \varphi_1 \vee \varphi_2$: Because $I_\varphi = I_{\varphi_1} \cap I_{\varphi_2}$, and each $\mathcal{A}_s^{\varphi_i}$ for $i \in \{1, 2\}$ 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 $\mathcal{A}_i := \mathcal{A}_s^{\varphi_i} \!\downarrow_{I_\varphi} = (Q^i, \delta^i, q_\iota^i, C^i)$. Letting q_ι be a fresh initial state we define $\mathcal{A}_s^\varphi := (\{q_\iota\} \cup Q^1 \cup Q^2, \delta, q_\iota, C)$, where δ and C agree with δ^i and C^i, respectively, on states from Q^i, and $\delta(q_\iota, a) = \delta^1(q_\iota^1, a) \vee \delta^2(q_\iota^2, a)$. The colour of q_ι does not matter.

$\varphi = \mathrm{E}\psi$: Let $\max(\psi) = \{\varphi_1, \ldots, \varphi_k\}$ be the set of maximal state subformulas of ψ. In a first step we see these maximal state subformulas as atomic propositions, we see ψ as an LTL formula over $\max(\psi)$, and we 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 (and uses two colours) [81]. We define the ATA $\mathcal{A}$ that, given as input a $(\max(\psi), S_\varphi)$-tree t, nondeterministically guesses a path λ in $t \!\uparrow^{[n]} \!\!\wedge t_S$, or equivalently a path in $\mathcal{S}$ starting from s, and simulates $\mathcal{W}^\psi$ on it, assuming that the labels it reads while following $\lambda \!\downarrow_{I_\varphi}$ in its input t correctly represent the truth value of formulas in $\max(\psi)$ along λ. Recall that $\mathcal{S} = (S, R, s_\iota, \ell_S)$; we define $\mathcal{A} := (Q, \delta, q_\iota, C)$, where

- $Q = Q^\psi \times S$,
- $q_\iota = (q_\iota^\psi, s)$,
- for each $(q^\psi, s') \in Q$, $C(q^\psi, s') = C^\psi(q^\psi)$, 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 in $2^{\max(\psi)}$, chooses nondeterministically a transition in $\mathcal{W}^\psi$, chooses a next state s'' in S and proceeds in the corresponding direction $s'' \!\downarrow_{I_\varphi} \in S_\varphi$.

Now from $\mathcal{A}$ we build the automaton $\mathcal{A}_s^\varphi$ over S_φ-trees labelled with "real" atomic propositions in $\mathrm{AP}_\exists$. Intuitively, 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, after having guessed a finite path $u \in t_S$ ending in state s', $\mathcal{A}_s^\varphi$ guesses that φ_i holds, it checks this guess by starting a copy of automaton $\mathcal{A}_{s'}^{\varphi_i}$ from node $v = u \!\downarrow_{I_\varphi}$ in its input t.

Formally, for each $s' \in \mathcal{S}$ and each $\varphi_i \in \max(\psi)$ we first build $\mathcal{A}_{s'}^{\varphi_i}$, which works on S_{φ_i}-trees. Observe that $I_\varphi = \cap_{i=1}^k I_{\varphi_i}$, so that we need to narrow down these automata[2]: We let $\mathcal{A}_{s'}^i := \mathcal{A}_{s'}^{\varphi_i} \!\downarrow_{I_\varphi} = (Q_{s'}^i, \delta_{s'}^i, q_{s'}^i, C_{s'}^i)$. 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 the dualisation of $\mathcal{A}_{s'}^i$, and we

---

[2]In the conference version of this work [6] we made a mistake here: we wrote that $I_\varphi = I_{\varphi_i}$, which is not the case in general. As a consequence we do need to narrow down automata, unlike what was written in the conference version.

assume without loss of generality all the state sets are pairwise disjoint. 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

$$\left(\delta\left((q^\psi, s'), a'\right) \wedge \bigwedge_{\varphi_i \in a'} \delta_{s'}^i(q_{s'}^i, a) \wedge \bigwedge_{\varphi_i \notin a'} \overline{\delta_{s'}^i}(\overline{q_{s'}^i}, a) \right). \tag{2}$$

Note that in general it is not possible to define a $\max(\psi)$-labelling of t that faithfully represents the truth values of formulas in $\max(\psi)$ for all nodes in $t_{\mathcal{S}}$, because a node in t may correspond to different nodes in $t_{\mathcal{S}}$ that have same projection on S_φ but satisfy different formulas of $\max(\psi)$. However this is not a problem because different copies of $\mathcal{A}_s^\varphi$ that visit the same node can guess different labellings, depending on the actual state of $\mathcal{S}$ (which is part of the state of $\mathcal{A}_s^\varphi$).

$\varphi = \exists^{\mathbf{o}} p. \varphi'$: We build automaton $\mathcal{A}_s^{\varphi'}$ that works on $S_{\varphi'}$-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 $S_{\mathbf{o}}$-trees and obtain $\mathcal{A}_1 := \mathcal{A}_s^{\varphi'} \downarrow_{\mathbf{o}}$. By Theorem 4.6 we can nondeterminise it to get $\mathcal{A}_2$, which by Theorem 4.5 we can project with respect to p, finally obtaining $\mathcal{A}_s^\varphi := \mathcal{A}_2 \Downarrow_p$.

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

$\varphi = p$: First, note that $I_p = [n]$, so that t is rooted in $s_\iota \downarrow_{I_\varphi} = s_\iota$, and $u \downarrow_{I_\varphi} = u$. Also recall that u ends in s. Let us consider first the case where $p \in \mathrm{AP}_f$. By definition of $\mathcal{A}_s^p$, we have that $(t, u) \in \mathcal{L}(\mathcal{A}_s^p)$ if and only if $p \in \ell_{\mathcal{S}}(s)$. We also have $t \uparrow^{[n]} \wedge\hspace{-0.35em}\wedge\hspace{-0.35em}\wedge t_{\mathcal{S}}, u \models p$ if and only if $p \in \ell'(u)$, where ℓ' is the labelling of tree $t \uparrow^{[n]} \wedge\hspace{-0.35em}\wedge\hspace{-0.35em}\wedge t_{\mathcal{S}}$. By definition of unfolding and merge, we have that $\ell'(u) = \ell_{\mathcal{S}}(s)$, which concludes this direction. Now if $p \in \mathrm{AP}_\exists$: by definition of $\mathcal{A}_s^p$, we have $(t, u) \in \mathcal{L}(\mathcal{A}_s^p)$ if and only if $p \in \ell(u)$; also, by definition of the merge and unfolding, we have that $t \uparrow^{[n]} \wedge\hspace{-0.35em}\wedge\hspace{-0.35em}\wedge t_{\mathcal{S}}, u \models p$ if and only if $p \in \ell(u)$, and we are done.

$\varphi = \neg\varphi'$: Correctness follows from the induction hypothesis and Theorem 4.4.

$\varphi_1 \vee \varphi_2$: We have $\mathcal{A}_i = \mathcal{A}_s^{\varphi_i} \downarrow_{I_\varphi}$, so by Theorem 4.7 we have $(t, u \downarrow_{I_\varphi}) \in \mathcal{L}(\mathcal{A}_i)$ if and only if $(t \uparrow^{I_{\varphi_i}}, u \downarrow_{I_{\varphi_i}}) \in \mathcal{L}(\mathcal{A}_s^{\varphi_i})$, which by induction hypothesis holds if and only if $(t \uparrow^{I_{\varphi_i}}) \uparrow^{[n]} \wedge\hspace{-0.35em}\wedge\hspace{-0.35em}\wedge t_{\mathcal{S}}, u \models \varphi_i$, i.e., $t \uparrow^{[n]} \wedge\hspace{-0.35em}\wedge\hspace{-0.35em}\wedge t_{\mathcal{S}}, u \models \varphi_i$. We conclude by observing that $\mathcal{L}(\mathcal{A}_s^\varphi) = \mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)$.

$\varphi = \mathbf{E}\psi$: Suppose that $t \uparrow^{[n]} \wedge\hspace{-0.35em}\wedge\hspace{-0.35em}\wedge t_{\mathcal{S}}, u \models \mathbf{E}\psi$. There exists an infinite path λ in $t \uparrow^{[n]} \wedge\hspace{-0.35em}\wedge\hspace{-0.35em}\wedge t_{\mathcal{S}}$ starting at u such that $t \uparrow^{[n]} \wedge\hspace{-0.35em}\wedge\hspace{-0.35em}\wedge t_{\mathcal{S}}, \lambda \models \psi$. 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 $\max(\psi)$, i.e., for each node λ_k of λ and formula $\varphi_i \in \max(\psi)$, it holds that $\varphi_i \in w_k$ if and only if $t \uparrow^{[n]} \wedge\hspace{-0.35em}\wedge\hspace{-0.35em}\wedge t_{\mathcal{S}}, \lambda_k \models \varphi_i$. To show that $(t, u \downarrow_{I_\varphi}) \in \mathcal{L}(\mathcal{A}_s^\varphi)$ we show that Eve can win the acceptance game $\mathcal{G}(\mathcal{A}_s^\varphi, t, u \downarrow_{I_\varphi})$. In this game, Eve can guess the path λ while the automaton follows $\lambda \downarrow_{I_\varphi}$ in its input t, and she can also guess the corresponding word w on $2^{\max(\psi)}$. By construction of $\mathcal{W}^\psi$, Eve has a winning strategy σ_ψ in the acceptance game of $\mathcal{W}^\psi$ on w. From λ, w and σ_ψ we can easily define a strategy for Eve in $\mathcal{G}(\mathcal{A}_s^\varphi, t, u \downarrow_{I_\varphi})$ on all positions that can be reached while Adam does not choose to challenge her on a guess she made for the truth value of some maximal state subformula, and on such plays this strategy is winning because σ_ψ is winning.

Now if Adam challenges her on one of these guesses: Let $\lambda_k \in t\uparrow^{[n]} \ \text{⫴}\ t_S$ be a node along λ, let s' be its last direction and let $\lambda'_k = \lambda_k \downarrow_{I_\varphi} \in t$. Assume that in node λ'_k 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 λ'_k, i.e., such that $\varphi_i \in w_k$. Formally, in the evaluation game this means that Adam chooses the conjunct $\delta^i_{s'}(q^i_{s'}, a)$ in transition formula 2, where $a = \ell(\lambda'_k)$, thus moving to position $(\lambda'_k, (q^\psi, s'), \delta^i_{s'}(q^i_{s'}, a))$. We want to show that Eve wins from this position. To do so we first show that $(t, \lambda'_k) \in \mathcal{L}(\mathcal{A}^i_{s'})$.

First, since $\mathcal{A}^i_{s'} = \mathcal{A}^{\varphi_i}_{s'} \downarrow_{I_\varphi}$, by Theorem 4.7, $(t, \lambda'_k) \in \mathcal{L}(\mathcal{A}^i_{s'})$ if and only if $(t\uparrow^{I_{\varphi_i}}, \lambda_k \downarrow_{I_{\varphi_i}}) \in \mathcal{L}(\mathcal{A}^{\varphi_i}_{s'})$. Next, by applying the induction hypothesis we get that $(t\uparrow^{I_{\varphi_i}}, \lambda_k \downarrow_{I_{\varphi_i}}) \in \mathcal{L}(\mathcal{A}^{\varphi_i}_{s'})$ if and only if $t\uparrow^{I_{\varphi_i}}\uparrow^{[n]} \ \text{⫴}\ t_S, \lambda_k \models \varphi_i$, i.e., $t\uparrow^{[n]} \ \text{⫴}\ t_S, \lambda_k \models \varphi_i$. The latter holds because $\varphi_i \in w_k$, and by assumption w_k agrees with λ_k on φ_i. Thus $(t, \lambda'_k) \in \mathcal{L}(\mathcal{A}^i_{s'})$.

This means that Eve has a winning strategy from the initial position $(\lambda'_k, q^i_{s'}, \delta^i_{s'}(q^i_{s'}, a))$ of the acceptance game of $\mathcal{A}^i_{s'}$ on (t, λ'_k). Since $(\lambda'_k, q^i_{s'}, \delta^i_{s'}(q^i_{s'}, a))$ and $(\lambda'_k, (q^\psi, s'), \delta^i_{s'}(q^i_{s'}, a))$ contain the same node λ'_k and transition formula $\delta^i_{s'}(q^i_{s'}, a)$, the subgames that start in these positions are isomorphic and a winning strategy in one of these positions induces a winning strategy in the other, and therefore Eve wins Adam's challenge (recall that positional strategies are sufficient in parity games [82]). With a similar argument, we get that also when Adam challenges Eve on some $\varphi_i \in \max(\psi)$ assumed not to be true in node λ_k, Eve wins the challenge. Finally, Eve wins the acceptance game of $\mathcal{A}^\varphi_s$ on $(t, u \downarrow_{I_\varphi})$, and thus $(t, u \downarrow_{I_\varphi}) \in \mathcal{L}(\mathcal{A}^\varphi_s)$.

For the other direction, assume that $(t, u \downarrow_{I_\varphi}) \in \mathcal{L}(\mathcal{A}^\varphi_s)$, i.e., Eve wins the evaluation game of $\mathcal{A}^\varphi_s$ on $(t, u \downarrow_{I_\varphi})$. A winning strategy for Eve describes a path λ in t_S from s, which is also a path in $t\uparrow^{[n]} \ \text{⫴}\ t_S$ from u. 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 a winning strategy for Eve in the acceptance game of $\mathcal{W}^\psi$ on w. Hence $t\uparrow^{[n]} \ \text{⫴}\ t_S, \lambda \models \psi$, and $t\uparrow^{[n]} \ \text{⫴}\ t_S, u \models \varphi$.

$\boldsymbol{\varphi = \exists^{\mathbf{o}} p. \varphi'}$: First, by definition we have $I_\varphi = \mathbf{o} \cap I_{\varphi'}$. Because φ is hierarchical, $\mathbf{o} \subseteq \mathbf{o'}$ for every $\mathbf{o'}$ that occurs in φ', and thus $\mathbf{o} \subseteq I_{\varphi'}$. It follows that $I_\varphi = \mathbf{o}$. Next, by Theorem 4.5 we have that

$$(t, u \downarrow_{I_\varphi}) \in \mathcal{L}(\mathcal{A}^\varphi_s) \quad \text{iff} \quad \exists \ell_p \text{ a } p\text{-labelling for } t \text{ such that } (t \otimes \ell_p, u) \in \mathcal{L}(\mathcal{A}_2). \tag{3}$$

By Theorem 4.6, $\mathcal{L}(\mathcal{A}_2) = \mathcal{L}(\mathcal{A}_1)$, and since $\mathcal{A}_1 = \mathcal{A}^{\varphi'}_s \downarrow_{\mathbf{o}} = \mathcal{A}^{\varphi'}_s \downarrow_{I_\varphi}$ we get by Theorem 4.7 that

$$(t \otimes \ell_p, u \downarrow_{I_\varphi}) \in \mathcal{L}(\mathcal{A}_2) \quad \text{iff} \quad ((t \otimes \ell_p)\uparrow^{L_{\varphi'}}, u \downarrow_{I_{\varphi'}}) \in \mathcal{L}(\mathcal{A}^{\varphi'}_s). \tag{4}$$

By induction hypothesis,

$$((t \otimes \ell_p)\uparrow^{L_{\varphi'}}, u \downarrow_{I_{\varphi'}}) \in \mathcal{L}(\mathcal{A}^{\varphi'}_s) \quad \text{iff} \quad (t \otimes \ell_p)\uparrow^{L_{\varphi'}}\uparrow^{[n]} \ \text{⫴}\ t_S, u \models \varphi'. \tag{5}$$

Now, by points (3), (4) and (5) and the fact that $(t \otimes \ell_p)\uparrow^{I_{\varphi'}}\uparrow^{[n]} = (t \otimes \ell_p)\uparrow^{[n]}$, we get that

$$(t, u \downarrow_{I_\varphi}) \in \mathcal{L}(\mathcal{A}^\varphi_s) \quad \text{iff} \quad \exists \ell_p \text{ a } p\text{-labelling for } t \text{ such that } (t \otimes \ell_p)\uparrow^{[n]} \ \text{⫴}\ t_S, u \models \varphi'. \tag{6}$$

We now prove the following equation which, together with point (6), concludes the proof:

$$\begin{array}{c} \exists \ell_p \text{ a } p\text{-labelling for } t \text{ such that } (t \otimes \ell_p)\uparrow^{[n]} \ \text{⫴}\ t_S, u \models \varphi' \\ \text{iff} \\ t\uparrow^{[n]} \ \text{⫴}\ t_S, u \models \exists^{\mathbf{o}} p. \varphi' \end{array} \tag{7}$$

Assume that there exists a p-labelling ℓ_p for t such that $(t \otimes \ell_p)\uparrow^{[n]} \ \text{⫴}\ t_S, u \models \varphi'$. Let ℓ'_p be the p-labelling of $(t \otimes \ell_p)\uparrow^{[n]} \ \text{⫴}\ t_S$. By definition of the merge, ℓ'_p is equal to the p-labelling of $(t \otimes \ell_p)\uparrow^{[n]}$, which by definition of the widening is I_φ-uniform, i.e., it is $\mathbf{o}$-uniform. In addition, it is clear that $(t \otimes \ell_p)\uparrow^{[n]} \ \text{⫴}\ t_S = (t\uparrow^{[n]} \ \text{⫴}\ t_S) \otimes \ell'_p$, which concludes this direction.

For the other direction, assume that $t\uparrow^{[n]} \,\backslash\!\!\backslash\, t_S, u \models \exists^{\mathbf{o}}p.\,\varphi'$: there exists a $\mathbf{o}$-uniform p-labelling ℓ'_p for $t\uparrow^{[n]} \,\backslash\!\!\backslash\, t_S$ such that $(t\uparrow^{[n]} \,\backslash\!\!\backslash\, t_S) \otimes \ell'_p, u \models \varphi'$. We define a p-labelling ℓ_p for t such that $(t \otimes \ell_p)\uparrow^{[n]} \,\backslash\!\!\backslash\, t_S, u \models \varphi'$. First, let us write $t' = t\uparrow^{[n]} \,\backslash\!\!\backslash\, t_S = (\tau', \ell')$. For each node u of t, let

$$\ell_p(u) = \begin{cases} \ell'_p(u') & \text{if there exists } u' \in \tau' \text{ such that } u'{\downarrow_{\mathbf{o}}} = u, \\ 0 & \text{otherwise.} \end{cases}$$

This is well defined because ℓ'_p is $\mathbf{o}$-uniform in p, so that if two nodes u', v' project on u, we have $u' \approx_{\mathbf{o}} v'$ and thus $\ell'_p(u') = \ell'_p(v')$. In case there is no $u' \in \tau'$ such that $u'{\downarrow_{I_\varphi}} = u$, the value of $\ell_p(u)$ has no impact on $(t \otimes \ell_p)\uparrow^{[n]} \,\backslash\!\!\backslash\, t_S$. Finally, $(t \otimes \ell_p)\uparrow^{[n]} \,\backslash\!\!\backslash\, t_S = (t\uparrow^{[n]} \,\backslash\!\!\backslash\, t_S) \otimes \ell'_p$, hence the result. □

4.3 Proof of Theorem 4.3

We now prove Theorem 4.3. Let S be a CKS with initial state s_ι, and let $\Phi \in \text{QCTL}^*_{\text{i},\subseteq}$. By Lemma 4.10 one can build an ATA $\mathcal{A}^\Phi_{s_\iota}$ such that for every labelled S_φ-tree t rooted in $s_\iota{\downarrow_{I_\varphi}}$, and every node $u \in t_S$, $(t, u{\downarrow_{I_\varphi}}) \in \mathcal{L}(\mathcal{A}^\varphi_{s_\iota})$ if, and only if, $t\uparrow^{[n]} \,\backslash\!\!\backslash\, t_S, u \models \Phi$.

Let τ be the full S_φ-tree rooted in $s_\iota{\downarrow_{I_\varphi}}$, and let $t = (\tau, \ell_\emptyset)$, where $\ell_\emptyset$ is the empty labelling. Clearly, $t\uparrow^{[n]} \,\backslash\!\!\backslash\, t_S = t_S$, and because t is rooted in $s_\iota{\downarrow_{I_\varphi}}$, we get that $t \in \mathcal{L}(\mathcal{A}^\varphi_{s_\iota})$ if, and only if $t_S \models \Phi$, i.e., $S \models \Phi$. It remains to check whether tree t, which is regular, is accepted by $\mathcal{A}^\Phi_{s_\iota}$. This can be done by solving a parity game built from the product of $\mathcal{A}^\Phi_{s_\iota}$ with a finite Kripke structure representing t [57].

4.4 Complexity

To state a precise upper bound on the complexity of our procedure, we first introduce a syntactic notion of *simulation depth* for formulas of SL_{ii}. While alternation depth (see, e.g., [60]) simply counts the number of alternations between existential and universal strategy quantifications, simulation depth reflects automata operations required to treat a formula, and counts the maximum number of nested simulations of alternating tree automata that need to be performed when applying our automata construction. However, like alternation depth, it is a purely syntactic notion. Formally we define a function $\text{sd} : \text{QCTL}^*_{\text{ii}} \to \mathbb{N} \times \{\text{nd}, \text{alt}\}$ which returns, for each formula φ, a pair $\text{sd}(\varphi) = (k, x)$ where k is the simulation depth of φ, and $x \in \{\text{nd}, \text{alt}\}$ indicates whether the automaton $\mathcal{A}^\varphi_s$ built from φ and a state s of a CKS S is nondeterministic (nd) or alternating (alt). If $\text{sd}(\varphi) = (k, x)$ we shall denote k by $\text{sd}_k(\varphi)$ and x by $\text{sd}_x(\varphi)$. The inductive definition for state formulas is as follows:

$$\text{sd}(p) := (0, \text{nd})$$

$$\text{sd}(\neg\varphi) := (\text{sd}_k(\varphi), \text{alt})$$

$$\text{sd}(\varphi_1 \vee \varphi_2) := \left(\max_{i \in \{1,2\}} \text{sd}_k(\varphi_i), x\right),$$

$$\text{where } x = \begin{cases} \text{nd} & \text{if } \text{sd}_x(\varphi_1) = \text{sd}_x(\varphi_2) = \text{nd} \\ \text{alt} & \text{otherwise} \end{cases}$$

$$\text{sd}(\mathbf{E}\psi) := \begin{cases} (0, \text{nd}) & \text{if } \psi \in \text{LTL} \\ (\max_{\varphi \in \max(\psi)} \text{sd}_k(\varphi), \text{alt}) & \text{otherwise} \end{cases}$$

$$\text{sd}(\exists^{\mathbf{o}}p.\,\varphi) := (k, \text{nd}),$$

$$\text{where } k = \begin{cases} \text{sd}_k(\varphi) & \text{if } \text{sd}_x(\varphi) = \text{nd and } \mathbf{o} = I_\varphi \quad (\text{recall Definition 4.8}) \\ \text{sd}_k(\varphi) + 1 & \text{otherwise} \end{cases}$$

We explain each case. For an atomic proposition p, the automaton $\mathcal{A}_s^p$ is clearly nondeterministic and no simulation is involved in its construction. For a formula $\neg\varphi$, the automaton $\mathcal{A}_s^{\neg\varphi}$ is obtained by dualising $\mathcal{A}_s^\varphi$, an operation that in general does not return a nondeterministic automaton but an alternating one; also this dualisation does not involve any simulation, hence the definition of the first component. Now for the disjunction, the first component should be clear; for the second one, observe that by construction of $\mathcal{A}_s^{\varphi_1 \vee \varphi_2}$, if both $\mathcal{A}_s^{\varphi_1}$ and $\mathcal{A}_s^{\varphi_2}$ are nondeterministic, then so is $\mathcal{A}_s^{\varphi_1 \vee \varphi_2}$; otherwise, it is alternating. For the path quantifier, by construction $\mathcal{A}_s^{\mathbf{E}\psi}$ is alternating in the general case as it starts copies of automata for each maximal state subformula in ψ; for the first component, we recall that $\max(\psi)$ denotes the set of these maximal state subformulas and we observe that no additional simulation is performed to build $\mathcal{A}_s^{\mathbf{E}\psi}$ besides those needed to construct the automata for the maximal state subformulas. If ψ is an LTL formula, then one can build the nondeterministic word automaton $\mathcal{W}^\psi$ directly working on "real" atomic propositions in $\mathrm{AP}_\exists \cup \mathrm{AP}_f$. The automaton $\mathcal{A}$ can then be built working directly on $\mathrm{AP}_\exists$, with $\mathcal{W}^\psi$ reading valuations for $\mathrm{AP}_\exists$ in the input tree and those for atoms in AP_f in the current state of $\mathcal{S}$. Because we do not need to guess valuations of maximal state subformulas and launch additional automata to check that these guesses are correct, we obtain a nondeterministic automaton. Finally, for a formula of the form $\exists^{\mathbf{o}}p.\,\varphi$, to build automaton $\mathcal{A}_s^{\exists^{\mathbf{o}}p.\,\varphi}$ we first build $\mathcal{A}_s^\varphi$, which we then narrow down to work on $L_{\mathbf{o}}$-trees. Since the narrowing operation introduces alternation, we need to nondeterminise the resulting automaton before projecting it with respect to p. Now observe that if $I_\varphi = \mathbf{o}$ we do not need to perform this narrowing, and thus if $\mathcal{A}_s^\varphi$ is a nondeterministic automaton we can directly perform the projection. This justifies the definition of the first component; for the second one, observe that the projection of a nondeterministic automaton is also nondeterministic.

Example 4.11. Assume that $n = 3$, i.e., states of CKS have three components (recall that $[3] = \{1, 2, 3\}$). Let us consider formula $\varphi = \forall^{\{1,3\}}p.\,\forall^{[3]}q.\,\exists^{[3]}r.\,\mathbf{EG}(p \wedge q \vee r)$. We describe how its simulation depth is computed. First, let us rewrite $\varphi = \neg\exists^{\{1,3\}}p.\,\exists^{[3]}q.\,\neg\exists^{[3]}r.\,\mathbf{EG}(p \wedge q \vee r)$.

Since $\mathbf{G}(p \wedge q \vee r)$ is an LTL formula, $sd(\mathbf{EG}(p \wedge q \vee r)) = (0, \mathrm{nd})$. Next, because $I_{\mathbf{EG}(p \wedge q \vee r)} = [3]$, it follows that $sd(\exists^{[3]}r.\,\mathbf{EG}(p \wedge q \vee r)) = (0, \mathrm{nd})$, and $sd(\neg\exists^{[3]}r.\,\mathbf{EG}(p \wedge q \vee r)) = (0, \mathrm{alt})$. Next we have that $sd(\exists^{[3]}q.\,\neg\exists^{[3]}r.\,\mathbf{EG}(p \wedge q \vee r)) = (1, \mathrm{nd})$. This reflects the fact that the automaton obtained for formula $\neg\exists^{[3]}r.\,\mathbf{EG}(p \wedge q \vee r)$, which is alternating because of complementation, needs to be simulated before projecting it over q. Then, because $\{1, 3\} \neq [3]$, it holds that $sd(\exists^{\{1,3\}}p.\,\exists^{[3]}q.\,\neg\exists^{[3]}r.\,\mathbf{EG}(p \wedge q \vee r)) = (2, \mathrm{nd})$: to project over p we first need to narrow down the previous automaton to make it see only components 1 and 3, and because the narrowing operation introduces alternation, the resulting automaton needs to be simulated before projecting it. Finally, we get that $sd(\varphi) = (2, \mathrm{alt})$

We now introduce two additional depth measures on $\mathrm{QCTL}_{\mathrm{ii}}^*$ formulas, which help us establish more precise upper bounds on the sizes of the automata we build. For every $\mathrm{QCTL}_{\mathrm{ii}}^*$ formula φ, we let $\mathbf{Ed}(\varphi)$ be the maximum number of nested path quantifiers $\mathbf{E}$ in φ, and $\exists d(\varphi)$ is the maximum number of nested second-order quantifiers $\exists$ in φ. We also inductively define the function $\exp(k \mid n)$, for $k, n \in \mathbb{N}$, as follows: $\exp(0 \mid n) := n$ and $\exp(k + 1 \mid n) := 2^{\exp(k \mid n)}$.

PROPOSITION 4.12. *Let Φ be a $\mathrm{QCTL}_{\mathrm{i},\subseteq}^*$ formula, $\mathcal{S}$ a CKS and $s \in \mathcal{S}$ a state.*

- *If $sd_k(\Phi) = 0$, $\mathcal{A}_s^\Phi$ has at most $f_{\mathcal{S}}^\Phi$ states and 2 colours, and*
- *if $sd_k(\Phi) \geq 1$, $\mathcal{A}_s^\Phi$ has at most $\exp(sd_k(\Phi) \mid f_{\mathcal{S}}^\Phi \log f_{\mathcal{S}}^\Phi)$ states and its number of colours is at most $\exp(sd_k(\Phi) - 1 \mid f_{\mathcal{S}}^\Phi \log f_{\mathcal{S}}^\Phi)$,*

where $f_{\mathcal{S}}^\Phi = m_1^{\exists d(\Phi)}|\Phi||\mathcal{S}|^{\mathbf{Ed}(\Phi)} 2^{m_2|\Phi|\mathbf{Ed}(\Phi)}$, with $m_1, m_2 \in \mathbb{N}$ constants.

Also, if $\mathcal{A}_s^\varphi$ has state set Q then for each $q \in Q$ and $a \in 2^{\mathrm{AP}_\exists(\Phi)}$ we have $|\delta(q, a)| \leq |\mathcal{S}||Q|^{|\mathcal{S}|}2^{H|\varphi|}$, where $H = 1 + \mathbf{Ed}(\varphi)$.

Constants m_1 and m_2 are derived from constants in the complexity of, respectively, the simulation procedure, and the procedure that builds a nondeterministic word automaton for an LTL formula. For more detail, see the proof of Proposition 4.12 in Appendix A.

From this we get the following complexity result.

PROPOSITION 4.13. *The model-checking problem for* $\mathrm{QCTL}^*_{i,\subseteq}$ *formulas of simulation depth at most* k *is* $(k + 1)$-EXPTIME-*complete*.

PROOF. We start with the upper bounds. For an instance $(\Phi, \mathcal{S})$, our decision procedure in Section 4.3 first builds automaton $\mathcal{A}_{s_\iota}^\Phi$, and concludes by testing whether the full S_Φ-tree with empty labelling t is accepted by $\mathcal{A}_{s_\iota}^\Phi$. This can be done in time $O((|\mathcal{A}_{s_\iota}^\Phi| \cdot |t|)^l)$, where $|t|$ is the size of a smallest Kripke structure representing the regular tree t, $|\mathcal{A}_{s_\iota}^\Phi|$ is the sum of the number of states and sizes of formulas in the transition function of $\mathcal{A}_{s_\iota}^\Phi$, and l the number of colours it uses [57]. Clearly t can be represented by a Kripke structure of size $|S_\Phi|$, so that $|t| \leq |S_\Phi| \leq |\mathcal{S}|$.

By Proposition 4.12, each formula in the transition function of $\mathcal{A}_{s_\iota}^\Phi$ is of size at most $|\mathcal{S}||Q|^{|\mathcal{S}|}2^{H|\Phi|}$, where Q is the set of states in $\mathcal{A}_{s_\iota}^\Phi$ and $H = 1 + \mathbf{Ed}(\Phi)$. There are at most $|Q|2^{|\mathrm{AP}_\exists(\Phi)|}$ such formulas[3] and $|\mathrm{AP}_\exists(\Phi)| \leq |\Phi|$, so that $|\mathcal{A}_{s_\iota}^\Phi| \leq |Q| + |Q|2^{|\mathrm{AP}_\exists(\Phi)|}|\mathcal{S}||Q|^{|\mathcal{S}|}2^{H|\Phi|} \leq 2|\mathcal{S}||Q|^{|\mathcal{S}|+1}2^{(H+1)|\Phi|}$. Also $H + 1 \leq |\Phi|$, so we finally have $|\mathcal{A}_{s_\iota}^\Phi| \leq 2|\mathcal{S}||Q|^{|\mathcal{S}|+1}2^{|\Phi|^2}$.

If $k = 0$, by Proposition 4.12 $\mathcal{A}_{s_\iota}^\Phi$ has at most $f_{\mathcal{S}}^\Phi$ states and 2 colours, and $f_{\mathcal{S}}^\Phi$ is polynomial in $|\mathcal{S}|$ but exponential in $|\Phi|$. Therefore $|\mathcal{A}_{s_\iota}^\Phi|$ is exponential in $|\Phi|$ and in $|\mathcal{S}|$, and so is the complexity of checking that t is accepted by $\mathcal{A}_{s_\iota}^\Phi$.

If $k \geq 1$, by Proposition 4.12, $|Q|$ is k-exponential in $f_{\mathcal{S}}^\Phi \log f_{\mathcal{S}}^\Phi$, and $f_{\mathcal{S}}^\Phi \log f_{\mathcal{S}}^\Phi$ itself is polynomial in $|\mathcal{S}|$ but exponential in $|\Phi|$. As a result, $|\mathcal{A}_{s_\iota}^\Phi|$ is $(k+1)$-exponential in $|\Phi|$ and k-exponential in $|\mathcal{S}|$. Finally, still by Proposition 4.12, the number of colours l is $(k-1)$-exponential in $f_{\mathcal{S}}^\Phi \log f_{\mathcal{S}}^\Phi$, hence k-exponential in $|\Phi|$. Checking that t is accepted by $\mathcal{A}_{s_\iota}^\Phi$ can thus be done in time $(k+1)$-exponential in $|\Phi|$, and k-exponential in $|\mathcal{S}|$, which finishes to establish the upper bounds.

For the lower bounds, consider the fragment $\mathrm{EQ}^k\mathrm{CTL}^*$ of QCTL^* (with perfect information) which consists in formulas in prenex normal form, i.e., with all second-order quantifications at the beginning, with at most k alternations between existential and universal quantifiers, counting the first quantifier as one alternation (see [53, p.8] for a formal definition). Clearly, $\mathrm{EQ}^k\mathrm{CTL}^*$ is a fragment of $\mathrm{QCTL}^*_{\mathrm{ii}}$ (with $n = 1$), and formulas of $\mathrm{EQ}^k\mathrm{CTL}^*$ have simulation depth at most k. It is proved in [53] that model checking $\mathrm{EQ}^k\mathrm{CTL}^*$ is $(k + 1)$-EXPTIME-hard. □

Remark 3. One may wonder why we do not get our lower bounds from the distributed synthesis problem in systems with hierarchical information. The reason is that this problem is k-EXPTIME-complete for LTL or CTL^* specifications [50, 69] and can be expressed with formulas of simulation depth k, and thus would only provide k-EXPTIME lower-bounds for simulation depth k, while our problem is $k + 1$-EXPTIME-complete. This may seem surprising, but we point out that thanks to alternation of existential and universal quantifiers, $\mathrm{QCTL}^*_{\mathrm{ii}}$ formulas with simulation depth k can express more complex problems than classic distributed synthesis, such as existence of Nash equilibria (see Section 7.1).

Improved upper bound. We now refine the previous result by observing that some subformulas can be model-checked independently in a bottom-up labelling algorithm which uses the above

---

[3]In fact the final automaton $\mathcal{A}_{s_\iota}^\Phi$ does not read anything in its input, hence the alphabet could be considered to be a singleton. We thus have only $|Q|$ different formulas in the transition function, at most.

model-checking procedure as a subroutine. The height of exponential of the overall procedure for a formula Φ is thus determined by the maximal simulation-depth of the successive independent subformulas φ treated by the labelling algorithm, instead of the simulation depth of the full formula Φ. To make this precise we define the *simulation number* of a sentence, akin to the alternation number introduced in [60].

Let $\Phi \in \mathrm{QCTL}_{\mathrm{ii}}^*$, and assume without loss of generality that $\mathrm{AP}_\exists(\Phi) \cap \mathrm{AP}_f(\Phi) = \emptyset$. A state subformula φ of Φ is a *subsentence* if no atom quantified in Φ appears free in φ, i.e., φ is a subsentence of Φ if $\mathrm{AP}_\exists(\Phi) \cap \mathrm{AP}_f(\varphi) = \emptyset$.[4] The *simulation number* $\mathrm{sn}(\Phi)$ of a $\mathrm{QCTL}_{\mathrm{ii}}^*$ formula Φ is the maximal simulation depth of Φ's subsentences, where the simulation depth is computed by considering strict subsentences as atoms.

Note that because temporal operators of $\mathrm{SL}_{\mathrm{ii}}$ can only talk about the future, the truth value of a subsentence in a node u of an unfolding t_S only depends on the current state $\mathrm{last}(u)$. The bottom-up labelling algorithm for an instance (Φ, S) thus consists in iteratively model checking innermore subsentences of Φ in all states of S, marking the states where they hold with fresh atomic propositions with which the corresponding subsentences are replaced in Φ.

PROPOSITION 4.14. *The model-checking problem for* $\mathrm{QCTL}_{\mathrm{i},\subseteq}^*$ *formulas of simulation number at most k is $(k+1)$-ExpTime-complete.*

5 MODEL-CHECKING HIERARCHICAL INSTANCES OF $\mathrm{SL}_{\mathrm{ii}}$

In this section we establish that the model-checking problem for $\mathrm{SL}_{\mathrm{ii}}$ restricted to the class of hierarchical instances is decidable (Theorem 2.9).

5.1 Reduction to $\mathrm{QCTL}_{\mathrm{ii}}^*$

We build upon the proof in [54] that establishes the decidability of the model-checking problem for $\mathrm{ATL}_{\mathrm{sc}}^*$ by reduction to the model-checking problem for QCTL^*. The main difference is that we reduce to the model-checking problem for $\mathrm{QCTL}_{\mathrm{ii}}^*$ instead, using quantifiers on atomic propositions parameterised with observations that reflect the ones used in the $\mathrm{SL}_{\mathrm{ii}}$ model-checking instance.

Let $(\mathcal{G}, \Phi)$ be a hierarchical instance of the $\mathrm{SL}_{\mathrm{ii}}$ model-checking problem, and assume without loss of generality that each strategy variable is quantified at most once in Φ. We define an equivalent instance of the model-checking problem for $\mathrm{QCTL}_{\mathrm{i},\subseteq}^*$.

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

Suppose $\mathrm{Obs} = \{o_1, \ldots, o_n\}$, and let $\mathcal{G} = (\mathrm{Ac}, V, E, \ell, v_\iota, O)$. 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 $\mathrm{CGS}_{\mathrm{ii}}$ to define the dynamics, we also let $L_{n+1} := V$.

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

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

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

---

[4]Observe that since we always assume that $\mathrm{AP}_\exists(\Phi) \cap \mathrm{AP}_f(\Phi) = \emptyset$, Φ is a subsentence of itself.

For every finite play $\rho = v_0 \ldots v_k$, define the node $u_\rho := s_{v_0} \ldots s_{v_k}$ in $t_{\mathcal{S}_\mathcal{G}}$ (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 $t_{\mathcal{S}_\mathcal{G}}$.

Constructing the $\text{QCTL}^*_{i,\subseteq}$ **formulas** $(\varphi)_s^f$**.** We now describe how to transform an SL_{ii} formula φ and a partial function $f : \text{Ag} \rightharpoonup \text{Var}$ into a QCTL^*_{ii} formula $(\varphi)_s^f$ (that will also depend on $\mathcal{G}$). Suppose that $\text{Ac} = \{c_1, \ldots, c_l\}$, and define $(\varphi)_s^f$ and $(\psi)_p^f$ by mutual induction on state and path formulas. The base cases are as follows: $(p)_s^f := p$ and $(\varphi)_p^f := (\varphi)_s^f$. Boolean and temporal operators are simply obtained by distributing the translation: $(\neg\varphi)_s^f := \neg(\varphi)_s^f$, $(\neg\psi)_p^f := \neg(\psi)_p^f$, $(\varphi_1 \vee \varphi_2)_s^f := (\varphi_1)_s^f \vee (\varphi_2)_s^f$, $(\psi_1 \vee \psi_2)_p^f := (\psi_1)_p^f \vee (\psi_2)_p^f$, $(\mathbf{X}\psi)_p^f := \mathbf{X}(\psi)_p^f$ and $(\psi_1 \mathbf{U} \psi_2)_p^f := (\psi_1)_p^f \mathbf{U}(\psi_2)_p^f$.

We continue with the case of the strategy quantifier:

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

$$\text{where} \qquad \varphi_{\text{str}}(x) := \mathbf{AG} \bigvee\nolimits_{c \in \text{Ac}} p_c^x$$

$$\text{and} \qquad \widetilde{o}_i := \{j \mid O(o_i) \subseteq O(o_j)\}.$$

The intuition is that for each possible action $c \in \text{Ac}$, an existential quantification on the atomic proposition p_c^x "chooses" for each node u_ρ of the tree $t_{\mathcal{S}_\mathcal{G}}$ whether strategy x allows action c in ρ or not, and it does so uniformly with regards to observation $\widetilde{o}$. $\varphi_{\text{str}}(x)$ checks that at least one action is allowed in each node, and thus that atomic propositions p_c^x indeed define a strategy.

We define $\widetilde{o}_i$ as $\{j \mid O(o_i) \subseteq O(o_j)\}$ instead of $\{i\}$ in order to obtain a hierarchical instance. Note that including all coarser observations does not increase the information accessible to the quantifier: indeed, two nodes are $\{i\}$-indistinguishable if and only if they are $\widetilde{o}_i$-indistinguishable.

Here are the remaining cases:

$$((a, x)\varphi)_s^f := (\varphi)_s^{f[a \mapsto x]} \qquad \text{for } x \in \text{Var} \cup \{?\}$$

$$\text{and} \qquad (\mathbf{E}\psi)_s^f := \mathbf{E}(\psi_{\text{out}}^f \wedge (\psi)_p^f)$$

$$\text{where} \qquad \psi_{\text{out}}^f := \mathbf{G} \bigvee\nolimits_{v \in V} \left(p_v \wedge \bigvee\nolimits_{\mathbf{c} \in \text{Ac}^{\text{Ag}}} \bigwedge\nolimits_{a \in dom(f)} p_{\mathbf{c}_a}^{f(a)} \wedge \mathbf{X} p_{E(v, \mathbf{c})} \right).$$

ψ_{out}^f checks that each player a in the domain of f follows the strategy coded by the $p_c^{f(a)}$.

Remark 4. If we consider the fragment of SL_{ii} that only allows for deterministic strategies, the translation can be adapted by simply replacing formula $\varphi_{\text{str}}(x)$ above with its deterministic variant

$$\varphi_{\text{str}}^{\text{det}}(x) := \mathbf{AG} \bigvee_{c \in \text{Ac}} (p_c^x \wedge \bigwedge_{c' \neq c} \neg p_{c'}^x),$$

which ensures that *exactly one* action is chosen for strategy x in each finite play, and thus that atomic propositions p_c^x characterise a deterministic strategy.

To prove correctness of the translation, given a strategy σ and a strategy variable x we let $\ell_\sigma^x := \{\ell_{p_c^x} \mid c \in \text{Ac}\}$ be the family of p_c^x-labellings for tree $t_{\mathcal{S}_\mathcal{G}}$ defined as follows: for each finite play ρ in $\mathcal{G}$ and $c \in \text{Ac}$, we let $\ell_{p_c^x}(u_\rho) := 1$ if $c \in \sigma(\rho)$, 0 otherwise. For a labelled tree t with same domain as $t_{\mathcal{S}_\mathcal{G}}$ we write $t \otimes \ell_\sigma^x$ for $t \otimes \ell_{p_{c_1}^x} \otimes \ldots \otimes \ell_{p_{c_l}^x}$.

Given an infinite play π and a point $i \in \mathbb{N}$, we also let $\lambda_{\pi, i}$ be the infinite path in $t_{\mathcal{S}_\mathcal{G}}$ that starts in node $u_{\pi_{\leq i}}$ and is defined as $\lambda_{\pi, i} := u_{\pi_{\leq i}} u_{\pi_{\leq i+1}} u_{\pi_{\leq i+2}} \ldots$

Finally, for an assignment χ and a partial function $f : \text{Ag} \rightharpoonup \text{Var}$, we say that f is *compatible* with χ if $dom(\chi) \cap \text{Ag} = dom(f)$ and for all $a \in dom(f)$, $\chi(a) = \chi(f(a))$.

PROPOSITION 5.1. *For every state subformula φ and path subformula ψ of Φ, finite play ρ, infinite play π, point $i \in \mathbb{N}$, for every assignment χ variable-complete for φ (resp. ψ) and partial function $f : \text{Ag} \rightharpoonup \text{Var}$ compatible with χ, assuming also that no x_i in $dom(\chi) \cap \text{Var} = \{x_1, \ldots, x_k\}$ is quantified in φ or ψ, we have*

$$\mathcal{G}, \chi, \rho \models \varphi \qquad \text{if and only if} \qquad t_{\mathcal{S}_\mathcal{G}} \otimes \ell^{x_1}_{\chi(x_1)} \otimes \ldots \otimes \ell^{x_k}_{\chi(x_k)}, u_\rho \models (\varphi)^f_s$$

$$\mathcal{G}, \chi, \pi, i \models \psi \qquad \text{if and only if} \qquad t_{\mathcal{S}_\mathcal{G}} \otimes \ell^{x_1}_{\chi(x_1)} \otimes \ldots \otimes \ell^{x_k}_{\chi(x_k)}, \lambda_{\pi,i} \models (\psi)^f_p$$

In addition, $\mathcal{S}_\mathcal{G}$ is of size linear in $|\mathcal{G}|$, and $(\varphi)^f_s$ and $(\psi)^f_p$ are of size linear in $|\mathcal{G}|^2 + |\varphi|$.

PROOF. The proof is by induction on φ. We detail the cases for binding, strategy quantification and outcome quantification, the others follow simply by definition of $\mathcal{S}_\mathcal{G}$ for atomic propositions and induction hypothesis for remaining cases.

For $\varphi = (a, x)\varphi'$, we have $\mathcal{G}, \chi, \rho \models (a, x)\varphi'$ if and only if $\mathcal{G}, \chi[a \mapsto \chi(x)], \rho \models \varphi'$. The result follows by using the induction hypothesis with assignment $\chi[a \mapsto x]$ and function $f[a \mapsto x]$. This is possible because $f[a \mapsto x]$ is compatible with $\chi[a \mapsto x]$: indeed $dom(\chi[a \mapsto x]) \cap \text{Ag}$ is equal to $dom(\chi) \cap \text{Ag} \cup \{a\}$ which, by assumption, is equal to $dom(f) \cup \{a\} = dom(f[a \mapsto x])$. Also by assumption, for all $a' \in dom(f)$, $\chi(a') = \chi(f(a'))$, and by definition $\chi[a \mapsto \chi(x)](a) = \chi(x) = \chi(f[a \mapsto x](a))$.

For $\varphi = \langle\!\langle x \rangle\!\rangle^o \varphi'$, assume first that $\mathcal{G}, \chi, \rho \models \langle\!\langle x \rangle\!\rangle^o \varphi'$. There exists an o-uniform strategy σ such that

$$\mathcal{G}, \chi[x \mapsto \sigma], \rho \models \varphi'.$$

Since f is compatible with χ, it is also compatible with assignment $\chi' = \chi[x \mapsto \sigma]$. By assumption, no variable in $\{x_1, \ldots, x_k\}$ is quantified in φ, so that $x \neq x_i$ for all i, and thus $\chi'(x_i) = \chi(x_i)$ for all i; and because no strategy variable is quantified twice in a same formula, x is not quantified in φ', so that no variable in $\{x_1, \ldots, x_k, x\}$ is quantified in φ'. By induction hypothesis

$$t_{\mathcal{S}_\mathcal{G}} \otimes \ell^{x_1}_{\chi'(x_1)} \otimes \ldots \otimes \ell^{x_k}_{\chi'(x_k)} \otimes \ell^x_{\chi'(x)}, u_\rho \models (\varphi')^f_s.$$

Because σ is o-uniform, each $\ell_{p^x_c} \in \ell^x_\sigma = \ell^x_{\chi'(x)}$ is $\widetilde{o}$-uniform, and it follows that

$$t_{\mathcal{S}_\mathcal{G}} \otimes \ell^{x_1}_{\chi'(x_1)} \otimes \ldots \otimes \ell^{x_k}_{\chi'(x_k)}, u_\rho \models \exists^{\widetilde{o}} p^x_{c_1} \ldots \exists^{\widetilde{o}} p^x_{c_l}. \varphi_{\text{str}}(x) \wedge (\varphi')^f_s.$$

Finally, since $\chi'(x_i) = \chi(x_i)$ for all i, we conclude that

$$t_{\mathcal{S}_\mathcal{G}} \otimes \ell^{x_1}_{\chi(x_1)} \otimes \ldots \otimes \ell^{x_k}_{\chi(x_k)}, u_\rho \models (\langle\!\langle x \rangle\!\rangle^o \varphi')^f_s.$$

For the other direction, assume that

$$t_{\mathcal{S}_\mathcal{G}} \otimes \ell^{x_1}_{\chi(x_1)} \otimes \ldots \otimes \ell^{x_k}_{\chi(x_k)}, u_\rho \models (\varphi)^f_s,$$

and recall that $(\varphi)^f_s = \exists^{\widetilde{o}} p^x_{c_1} \ldots \exists^{\widetilde{o}} p^x_{c_l}. \varphi_{\text{str}}(x) \wedge (\varphi')^f_s$. Write $t = t_{\mathcal{S}_\mathcal{G}} \otimes \ell^{x_1}_{\chi(x_1)} \otimes \ldots \otimes \ell^{x_k}_{\chi(x_k)}$. There exist $\widetilde{o}$-uniform $\ell_{p^x_c}$-labellings such that

$$t \otimes \ell_{p^x_{c_1}} \otimes \ldots \otimes \ell_{p^x_{c_l}} \models \varphi_{\text{str}}(x) \wedge (\varphi')^f_s.$$

By $\varphi_{\text{str}}(x)$, these labellings code for a strategy σ, and because they are $\widetilde{o}$-uniform, σ is o-uniform. Let $\chi' = \chi[x \mapsto \sigma]$. For all $1 \leq i \leq k$, by assumption $x \neq x_i$, and thus $\chi'(x_i) = \chi(x_i)$. The above can thus be rewritten

$$t_{\mathcal{S}_\mathcal{G}} \otimes \ell^{x_1}_{\chi'(x_1)} \otimes \ldots \otimes \ell^{x_k}_{\chi'(x_k)} \otimes \ell^x_{\chi'(x)} \models \varphi_{\text{str}}(x) \wedge (\varphi')^f_s.$$

By induction hypothesis we have $\mathcal{G}, \chi[x \mapsto \sigma], \rho \models \varphi'$, hence $\mathcal{G}, \chi, \rho \models \langle\langle x \rangle\rangle^o \varphi'$.

For $\varphi = \mathbf{E}\psi$, assume first that $\mathcal{G}, \chi, \rho \models \mathbf{E}\psi$. There exists a play $\pi \in \text{Out}(\chi, \rho)$ such that $\mathcal{G}, \chi, \pi, |\rho| - 1 \models \psi$. By induction hypothesis, $t_{\mathcal{S}_{\mathcal{G}}} \otimes \ell^{x_1}_{\chi(x_1)} \otimes \ldots \otimes \ell^{x_k}_{\chi(x_k)}, \lambda_{\pi, |\rho|-1} \models (\psi)^f_p$. Since π is an outcome of χ, each agent $a \in dom(\chi) \cap \text{Ag}$ follows strategy $\chi(a)$ in π. Because $dom(\chi) \cap \text{Ag} = dom(f)$ and for all $a \in dom(f)$, $\chi(a) = \chi(f(a))$, each agent $a \in dom(f)$ follows the strategy $\chi(f(a))$, which is coded by atoms $p^{f(a)}_c$ in the translation of Φ. Therefore $\lambda_{\pi, |\rho|-1}$ also satisfies ψ^χ_{out}, hence $t_{\mathcal{S}_{\mathcal{G}}} \otimes \ell^{x_1}_{\chi(x_1)} \otimes \ldots \otimes \ell^{x_k}_{\chi(x_k)}, \lambda_{\pi, |\rho|-1} \models \psi^\chi_{\text{out}} \wedge (\psi)^f_p$, and we are done.

For the other direction, assume that $t_{\mathcal{S}_{\mathcal{G}}} \otimes \ell^{x_1}_{\chi(x_1)} \otimes \ldots \otimes \ell^{x_k}_{\chi(x_k)}, u_\rho \models \mathbf{E}(\psi^f_{\text{out}} \wedge (\psi)^f_p)$. There exists a path λ in $t_{\mathcal{S}_{\mathcal{G}}} \otimes \ell^{x_1}_{\chi(x_1)} \otimes \ldots \otimes \ell^{x_k}_{\chi(x_k)}$ starting in node u_ρ that satisfies both ψ^f_{out} and $(\psi)^f_p$. By construction of $\mathcal{S}_{\mathcal{G}}$ there exists an infinite play π such that $\pi_{\leq |\rho|-1} = \rho$ and $\lambda = \lambda_{\pi, |\rho|-1}$. By induction hypothesis, $\mathcal{G}, \chi, \pi, |\rho| - 1 \models \psi$. Because $\lambda_{\pi, |\rho|-1}$ satisfies ψ^f_{out}, $dom(\chi) \cap \text{Ag} = dom(f)$, and for all $a \in dom(f)$, $\chi(a) = \chi(f(a))$, it is also the case that $\pi \in \text{Out}(\chi, \rho)$, hence $\mathcal{G}, \chi, \rho \models \mathbf{E}\psi$.

The size of $\mathcal{S}_{\mathcal{G}}$, $(\varphi)^f_s$ and $(\psi)^f_p$ are easily verified. □

Applying Proposition 5.1 to the sentence Φ, $\rho = v_\iota$, any assignment χ, and the empty function $\emptyset$, we get:

$$\mathcal{G} \models \Phi \quad \text{if and only if} \quad t_{\mathcal{S}_{\mathcal{G}}} \models (\Phi)^\emptyset_s.$$

Preserving hierarchy. To complete the proof of Theorem 2.9 it remains to check that $(\Phi)^\emptyset_s$ is a hierarchical $\text{QCTL}^*_{\text{ii}}$ formula, which is the case because Φ is hierarchical in $\mathcal{G}$ and for every two observations o_i and o_j in Obs such that $O(o_i) \subseteq O(o_j)$, by definition of $\widetilde{o_k}$ we have that $\widetilde{o_i} \subseteq \widetilde{o_j}$.

5.2 Complexity

We now establish the complexity of model checking hierarchical instances of SL_{ii}. As we did for $\text{QCTL}^*_{\text{ii}}$, we first define the simulation depth of SL_{ii} state formulas. In the following inductive definition, O_φ denotes the intersection of all indistinguishability relations used in φ: $O_\varphi := \cap_{o \in \varphi} O(o)$, with the empty intersection being defined as the identity relation (perfect information). Also, for a path formula ψ, $\max(\psi)$ is the set of maximal state subformulas in ψ.

$$\text{sd}(p) := (0, \text{nd}) \qquad\qquad\qquad \text{sd}(\neg\varphi) := (\text{sd}_k(\varphi), \text{alt})$$

$$\text{sd}(\varphi_1 \vee \varphi_2) := \left(\max_{i \in \{1,2\}} \text{sd}_k(\varphi_i), x \right),$$

$$\text{where } x = \begin{cases} \text{nd} & \text{if } \text{sd}_x(\varphi_1) = \text{sd}_x(\varphi_2) = \text{nd} \\ \text{alt} & \text{otherwise} \end{cases}$$

$$\text{sd}(\langle\langle x \rangle\rangle^o \varphi) := (k, \text{nd}),$$

$$\text{where } k = \begin{cases} \text{sd}_k(\varphi) & \text{if } \text{sd}_x(\varphi) = \text{nd} \text{ and } O(o) = O_\varphi \\ \text{sd}_k(\varphi) + 1 & \text{otherwise} \end{cases}$$

$$\text{sd}((a, x)\varphi) := \text{sd}(\varphi)$$

$$\text{sd}(\mathbf{E}\psi) := \begin{cases} (0, \text{nd}) & \text{if } \psi \in \text{LTL} \\ (\max_{\varphi \in \max(\psi)} \text{sd}_k(\varphi), \text{alt}) & \text{otherwise} \end{cases}$$

PROPOSITION 5.2. *The model-checking problem for hierarchical instances of* SL_{ii} *of simulation depth at most* k *is* $(k + 1)$-EXPTIME-*complete.*

PROOF. The upper bounds follow from the fact that the translated formulas in our reduction have essentially the same simulation depth as the original ones. However this is not quite right, because in the case where $\mathrm{sd}_x(\varphi) = \mathrm{nd}$ and $O(o) = O_\varphi$ we have $\mathrm{sd}(\langle\!\langle x \rangle\!\rangle^o \varphi) = (\mathrm{sd}_k(\varphi), \mathrm{nd})$, while $\mathrm{sd}((\langle\!\langle x \rangle\!\rangle^o \varphi)^f_s) = (\mathrm{sd}_k((\varphi)^f_s) + 1, \mathrm{nd})$: indeed, while it is the case that $O(o) = O_\varphi$ implies that $\widetilde{o} = I_{(\varphi)^f_s}$, the translation introduces a conjunction with $\varphi_{\mathrm{str}}(x)$, and even when $\mathrm{sd}_x((\varphi)^f_s) = \mathrm{nd}$, we have $\mathrm{sd}_x(\varphi_{\mathrm{str}}(x) \wedge (\varphi)^f_s) = \mathrm{alt}$. According to Proposition 4.13, this should thus induce an additional exponential to check the translated formula. However, this can be avoided by noticing that the fixed formula $\varphi_{\mathrm{str}}(x) = \mathbf{AG} \bigvee_{c \in \mathrm{Ac}} p^x_c$ can be checked by a simple *deterministic* tree automaton with two states q_{check} and q_{rej}: the automaton starts in state q_{check}, which is accepting (it has parity zero); when it visits a node u in state q_{check}, if $\ell(u)$ satisfies $\bigvee_{c \in \mathrm{Ac}} p^x_c$, then the automaton sends state q_{check} to all children of u, otherwise it sends the state q_{rej} to all children. State q_{rej} is rejecting (it has parity one) and is a sink: it sends itself to all children, independently on the label of the visited node. If we restrict $\mathrm{SL}_{\mathrm{ii}}$ to deterministic strategies, the same observation can be made: the automaton that checks formula $\varphi^{\mathrm{det}}_{\mathrm{str}}(x) = \mathbf{AG} \bigvee_{c \in \mathrm{Ac}}(p^x_c \wedge \bigwedge_{c' \neq c} \neg p^x_{c'})$ is the same as the one described above, except that it checks whether $\bigvee_{c \in \mathrm{Ac}}(p^x_c \wedge \bigwedge_{c' \neq c} \neg p^x_{c'})$ is satisfied by the label of the current node.

Given two tree automata $\mathcal{A}_1$ and $\mathcal{A}_2$, one deterministic and one nondeterministic, one can easily build a nondeterministic automaton $\mathcal{A}_1 \cap \mathcal{A}_2$ of size $|\mathcal{A}_1| \times |\mathcal{A}_2|$ that accepts the intersection of their languages, so that in this case the conjunction does not introduce alternation, and thus we do not need an additional simulation before projecting to guess the strategy. We could refine the notion of simulation depth to reflect this, but we find that it would become very cumbersome for little added benefit, so we keep this observation in this proof.

The lower bounds are inherited from $\mathrm{QCTL}^*_{\mathrm{ii}}$ thanks to the polynomial reduction presented in Section 6.2.2, which preserves simulation depth. □

We point out that all instances of the model-checking problem for the perfect-information fragment are hierarchical, and thus this result provides improved upper-bounds for SL, which was only known to be in k-EXPTIME for formulas of length at most k [60]. Also the lower bounds for $\mathrm{QCTL}^*_{\mathrm{ii}}$ are inherited directly from the perfect-information fragment QCTL^*, which reduces to the perfect-information fragment of $\mathrm{SL}_{\mathrm{ii}}$ following the construction from Section 6.2.2. Therefore the lower bounds hold already for the perfect-information fragment of $\mathrm{SL}_{\mathrm{ii}}$. Note however that this does not provide lower bounds for the usual, linear-time variant of Strategy Logic, where path quantifiers in QCTL^* formulas must be simulated with strategy quantifications which increase the simulation depth of the resulting Strategy Logic formulas. The exact complexity of the linear-time variant is not known, even in the perfect-information case.

Simulation number. The intuition behind the alternation number as considered in [60] is to refine the classic alternation depth between existential and universal quantifiers by observing that subsentences of a sentence Φ to model-check can be treated independently thanks to a bottom-up labelling algorithm: innermost sentences are evaluated in all states of the model and replaced in Φ by atomic propositions that label the states where they hold. The alternation number of Φ is the maximum alternation depth of the successive subsentences that are treated by this bottom-up procedure, and it determines the complexity of the overall model-checking procedure.

However, as discussed in Remark 1, the semantics of the outcome quantifier makes sentences sensitive to the assignment in which they are evaluated. As a result, to define the notion of alternation number in our setting, we introduce a notion of *independent subsentence*. Intuitively, a subsentence φ of a sentence Φ is *independent* if it redefines or unbinds the strategies of all players who are bound to a strategy when φ is reached in the evaluation of Φ. More precisely, we say that an agent a is *bound* in a syntactic subformula φ of Φ if the path that leads to φ in Φ's syntactic

tree contains a binding operator (a, x) for a which is not followed by an unbinding $(a, ?)$ for her. A subsentence φ of Φ is *independent* if all agents that are bound in φ are either rebound by an operator (a, x) or unbound by an operator $(a, ?)$ before any outcome quantifier is met in φ. In an independent subsentence φ, the semantics of the outcome quantifier does not depend on strategies that are quantified outside φ, and in fact a subsentence φ of Φ is independent if and only if the formula that corresponds to φ in $(\Phi)_s^{\emptyset}$ is a subsentence of $(\Phi)_s^{\emptyset}$.

Similarly to what we did for QCTL_{ii}^* we now define the *simulation number* $\text{sn}(\Phi)$ of an SL_{ii} sentence Φ as the maximum of the simulation depths for independent subsentences, where strict independent subsentences are counted as atoms.

LEMMA 5.3. *For every hierarchical instance* $(\mathcal{G}, \Phi)$ *of* SL_{ii}, $sn(\Phi) = sn((\Phi)_s^{\emptyset})$.

The following then follows from Proposition 5.1, Lemma 5.3 and Proposition 4.14.

PROPOSITION 5.4. *The model-checking problem for hierarchical instances of* SL_{ii} *of simulation number at most* k *is* $(k + 1)$-EXPTIME-*complete*.

We now compare the latter result with the complexity of model checking SL[NG], the nested goal fragment of Strategy Logic with perfect information (we refer the interested reader to [60] for a definition of this fragment). It is established in [21, 60] that this problem is in $(k + 1)$-EXPTIME for formulas of *alternation number* k. We remark that the simulation number of an SL[NG] formula translated in our branching-time version of SL (this is done by adding outcome quantifiers between bindings and temporal operators) is equal to its alternation number plus one, and thus Proposition 5.4 gives a $(k + 2)$-EXPTIME upper bound for SL[NG] formulas of alternation number k. In [21, 60] the extra exponential is avoided by resorting to universal and nondeterministic tree automata, depending on whether the innermost strategy quantification is existential or universal, to deal with temporal formulas. Thus, the innermost strategy quantification can be dealt with without incurring an exponential blowup.

The same thing cannot be done for SL_{ii}, for two reasons. The first one is that in general the innermost strategy quantification may have imperfect information and thus require a narrowing of the automaton; this operation introduces alternation, which needs to be removed at the cost of one exponential before dealing with strategy quantification. The second reason is that even when the innermost strategy has perfect information, the outcome quantifier that we introduce in Strategy Logic allows the expression of CTL* formulas which cannot be dealt with by nondeterministic and universal automata as is done in [21, 60].

6   COMPARISON WITH RELATED LOGICS

In this section we first show that SL_{ii} subsumes SL and the main imperfect-information extensions of ATL. Then we show that model checking Coordination Logic (CL) reduces to model checking hierarchical instances of SL_{ii} where the truth of all atomic propositions in the model is known by all agents (or more precisely, all observations in the concurrent game structures are fine enough to observe the truth value of all atomic propositions).

6.1   Comparison with ATL

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 [15]. In SL_{ii} instead each strategy x is assigned an indistinguishability relation o when it is quantified. Associating observations to strategies rather than players allows

us to obtain a logic SL_{ii} that is a clean generalisation of (perfect-information) SL, and subsumes imperfect-information extensions of ATL^* that associate observations to players. Concerning SL, it is rather easy to see that every sentence in SL has an equivalent in the fragment of SL_{ii} with deterministic strategies where all observation symbols are interpreted as perfect information. We now prove that SL_{ii} also subsumes ATL^* with imperfect information.

PROPOSITION 6.1. *For every* $ATL^*_{i,R}$ *formula[5]* φ *there is an* SL_{ii} *formula* φ' *such that for every* CGS_{ii} $\mathcal{G}$ *there is a* CGS_{ii} $\mathcal{G}'$ *such that* $\mathcal{G} \models \varphi$ *if, and only if,* $\mathcal{G}' \models \varphi'$.

We recall that an $ATL^*_{i,R}$ formula $\langle A \rangle \psi$ reads as "there are strategies for players in A such that ψ holds whatever players in $Ag \setminus A$ do". Formula φ' is built from φ by replacing each subformula of the form $\langle A \rangle \psi$, where $A = \{a_1, \ldots, a_k\} \subset Ag$ is a coalition of players and $Ag \setminus A = \{a_{k+1}, \ldots, a_n\}$ with formula $\langle\!\langle x_1 \rangle\!\rangle^{o_1} \ldots \langle\!\langle x_k \rangle\!\rangle^{o_k} (a_1, x_1) \ldots (a_k, x_k)(a_{k+1}, ?) \ldots (a_n, ?) \mathbf{A} \, \psi'$, 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, SL_{ii} also subsumes the imperfect-information extension of ATL^* with strategy context (see [55] for the definition of ATL^*_{sc} with partial observation, which we refer to as $ATL^*_{sc,i}$).

PROPOSITION 6.2. *For every* $ATL^*_{sc,i}$ *formula* φ *there is an* SL_{ii} *formula* φ' *such that for every* CGS_{ii} $\mathcal{G}$ *there is a* CGS_{ii} $\mathcal{G}'$ *such that* $\mathcal{G} \models \varphi$ *if, and only if,* $\mathcal{G}' \models \varphi'$.

The only difference between $ATL^*_{sc,i}$ and $ATL^*_{i,R}$ is the following: in $ATL^*_{i,R}$, when a subformula of the form $\langle A \rangle \psi$ is met, we quantify existentially on strategies for players in A and quantify universally on possible outcomes obtained by letting other players behave however they want. Therefore, if any player in $Ag \setminus A$ had previously been assigned a strategy, it is forgotten. In $ATL^*_{sc,i}$ 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 6.1: it suffices not to unbind agents outside the coalition from their strategies. $\mathcal{G}'$ is defined as for Proposition 6.1.

6.2 Comparison with Coordination Logic

There is a natural and simple translation of instances of the model-checking problem of CL [32] into the hierarchical instances of SL_{ii}. Moreover, the image of this translation consists of instances of SL_{ii} with a very restricted form: atoms mentioned in the SL_{ii}-formula are observable by all observations of the CGS_{ii}, i.e., for all $o \in Obs$ and $p \in AP$, $v \sim_o v'$ implies that $p \in \ell(v)$ iff $p \in \ell(v')$.

PROPOSITION 6.3. *There is an effective translation that, given a* CL-*instance* $(\mathcal{S}, \varphi)$ *produces a hierarchical* SL_{ii}-*instance* $(\mathcal{G}, \Phi)$ *such that*

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

To do this, one first translates CL into (hierarchical) $QCTL^*_{ii}$, the latter is defined in the next section. This step is a simple reflection of the semantics of CL in that of $QCTL^*_{ii}$. Then one translates $QCTL^*_{ii}$ into SL_{ii} by a simple adaptation of the translation of $QCTL^*$ into ATL^*_{sc} [54].

We briefly recall the syntax and semantics of CL, and refer to [32] for further details.

---

[5]See [15] for the definition of $ATL^*_{i,R}$, where subscript i refers to "imperfect information" and subscript R to "perfect recall". Also, we consider the so-called *objective semantics* for $ATL^*_{i,R}$.

Notation for trees. Note that our definition for trees (see Section 3.2) differs slightly from the one in [32], where the root is the empty word. Here we adopt this convention to stay closer to notations in [32]. Thus, (Y, X)-trees in CL are of the form (τ, l) where $\tau \subseteq X^*$ and $l : \tau \rightarrow 2^Y$.

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 $t \oplus t' := t{\uparrow}^{2^{Z \setminus X}} \uplus t'{\uparrow}^{2^{Z \setminus Y}}$.

CL Syntax. Let C be a set of *coordination variables*, and let $\mathcal{S}$ be a set of *strategy variables* disjoint from 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 $x \in C \cup \mathcal{S}$, $C \subseteq \mathcal{C}$ and $s \in \mathcal{S}$, and with the restriction that each coordination variable appears in at most one *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 $Scope_\varphi(s)$ 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 $f_s : (2^{Scope_\varphi(s)})^* \rightarrow 2^{\{s\}}$. 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 $(\mathrm{AP}_\varphi, 2^{\mathcal{F}_\varphi})$-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:

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

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, $t_{\lambda_i} \oplus ((2^{Scope_\varphi(s)})^*, f)$ is a $2^{\mathcal{F}_\varphi \cup Scope_\varphi(s)}$-tree. By definition, $Scope_\varphi(s) = \mathcal{F}_\varphi \cup C = \mathcal{F}_{\varphi'}$. It follows that $\mathcal{F}_\varphi \cup Scope_\varphi(s) = Scope_\varphi(s) = \mathcal{F}_{\varphi'}$, hence φ' is indeed evaluated on a $\mathcal{F}_{\varphi'}$-tree.

Remark 5. 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 $p \in \mathrm{AP}_\varphi$, 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 where atomic propositions are not observable to all players.

The input to the model-checking problem for CL consists of a CL formula φ and a finite representation of a $(\mathrm{AP}_\varphi, 2^{\mathcal{F}_\varphi})$-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 $(\mathrm{AP}_\varphi, 2^{\mathcal{F}_\varphi})$-tree $t = (\tau, \ell')$ is a structure $\mathcal{S} = (S, R, \ell, s_\iota)$ such that

- $S = 2^{\mathcal{F}_\varphi}$,

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

and $t = t_S$ is the unfolding of S.

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

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

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

We will present the translation in two steps: first from CL-instances into $\mathrm{QCTL}^*_{\mathrm{i},\subseteq}$-instances, and then from $\mathrm{QCTL}^*_{\mathrm{ii}}$-instances to $\mathrm{SL}_{\mathrm{ii}}$-instances such that $\mathrm{QCTL}^*_{\mathrm{i},\subseteq}$-instances translate to hierarchical $\mathrm{SL}_{\mathrm{ii}}$-instances.

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

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

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

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

Second, we define concrete observations corresponding to strategy variables in Φ. As explained in [32], and as reflected in the semantics of CL, the intuition is that a strategy variable s in formula Φ observes coordination variables $Scope_\varphi(s)$. Therefore, we simply define, for each strategy variable s in Φ, the concrete observation $\mathbf{o}_s := Scope_\varphi(s)$.

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

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

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

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

LEMMA 6.4. $t_S \models \Phi$ *iff* $t_{\widetilde{S}} \models A\widetilde{\Phi}$.

Importantly, we notice that $A\widetilde{\Phi} \in \text{QCTL}^*_{i,\subseteq}$, and that:

LEMMA 6.5. *For every* $x \in \text{AP}_\varphi$ *and every s quantified in* Φ, $t_{\widetilde{S}}$ *is* $\mathbf{o}_s$-*uniform in* x.

6.2.2 Translation from QCTL^*_ii *to* SL_ii. We now present a translation of QCTL^*_ii-instances to SL_ii-instances. It is a simple adaptation of the reduction from the model-checking problem for QCTL^* to the model-checking problem for ATL^*_sc presented in [54].

Let $(\mathcal{S}, \Phi)$ be an instance of the model-checking problem for QCTL^*_ii, where $\mathcal{S} = (S, R, \ell, s_\iota)$ and $S \subseteq \prod_{i \in [n]} L_i$. 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 $\text{AP}_\exists(\Phi) = \{p_1, \ldots, p_k\}$ 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 CGS_ii $\mathcal{G}^{\mathcal{S}} := (\text{Ac}, V, E, \ell', v_\iota, O)$ over agents $\text{Ag} := \{a_0, a_1, \ldots, a_k\}$, observations $\text{Obs} := \{o_0, o_1, \ldots, o_k\}$ and atomic propositions $\text{AP} := \text{AP}_\exists(\Phi) \cup \{p_S\}$, 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 $p_i \in \text{AP}_\exists(\Phi)$.

To this aim, we let

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

and

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

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

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

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 $\mathbf{c} \in \text{Ac}^\text{Ag}$ such that $\mathbf{c}_0 = c^i$,

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

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 $\mathbf{c} \in \text{Ac}^\text{Ag}$,

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

Positions of the form v_{p_i} and $v_\perp$ are sink positions.

The labelling function ℓ' is defined as follows:

$$\ell'(v) := \begin{cases} \ell(s) \cup \{p_S\} & \text{if } v = v_s \text{ for some } s \in S \\ \emptyset & \text{if } v \in \{v_{s,i} \mid s \in S, i \in [k]\} \cup \{v_{p_0}, v_\perp\} \\ \{p_i\} & \text{if } v = v_{p_i} \text{ with } i \in [k] \end{cases}$$

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

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

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

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

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 $O(o_i)$, it determines an $\mathbf{o}_i$-uniform labelling for p_i in t_S. Reciprocally, an $\mathbf{o}_i$-uniform labelling for p_i in t_S induces an $O(o_i)$-strategy for agent a_i. It remains to transform Φ into an SL_{ii}-formula.

We define the SL_{ii} formula $\widetilde{\Phi}$ by induction on Φ as follows:

$$\widetilde{p} := \begin{cases} \mathbf{EXX}p & \text{if } p = p_i \\ p & \text{otherwise} \end{cases}$$

$$\widetilde{\neg\varphi} := \neg\widetilde{\varphi}$$

$$\widetilde{\varphi_1 \vee \varphi_2} := \widetilde{\varphi_1} \vee \widetilde{\varphi_2}$$

$$\widetilde{\mathbf{E}\psi} := \mathbf{E}(\mathbf{G}p_S \wedge \widetilde{\psi})$$

$$\widetilde{\exists^{\mathbf{o}_i}p_i.\,\varphi} := \langle\!\langle x_i \rangle\!\rangle^{o_i}(a_i, x_i)\widetilde{\varphi}.$$

The cases for path formulas are obtained by distributing over the operators.

Observe that player 0 is never bound to a strategy. In the case for atomic propositions, the existential quantification on outcomes thus lets player 0 choose to move to a position where agent i fixes the value for p_i according to his strategy, fixed by the strategy quantifier in the translation for formulas of the form $\exists^{\mathbf{o}_i}p_i.\,\varphi$. In the translation of formulas of the form $\mathbf{E}\psi$, the existential quantification on outcomes lets player 0 choose a path in the original CKS $\mathcal{S}$.

We have the following:

LEMMA 6.6. $\mathcal{S} \models \Phi$ *if and only if* $\mathcal{G}^{\mathcal{S}} \models \widetilde{\Phi}$.

We observe that if Φ is hierarchical, then $(\widetilde{\Phi}, \mathcal{G}^{\mathcal{S}})$ is a hierarchical instance, and:

LEMMA 6.7. *For every* $p \in \mathrm{AP}_f(\Phi)$ *and for every* $i \in [k]$, *if* t_S *is* $\mathbf{o}_i$*-uniform in* p *then* $v \sim_{o_i} v'$ *implies that* $p \in \ell(v)$ *iff* $p \in \ell(v')$.

Combining Lemma 6.4 with Lemma 6.6 we get a reduction from the model-checking problem for CL to that for the hierarchical fragment of SL_{ii}, and Lemma 6.5 together with Lemma 6.7 show that in the models produced by this reduction, all atomic propositions are observable to all players. This implies that in CL one cannot reason about strategic problems with unobservable objectives. As a result it does not fully capture classic distributed synthesis [50, 69], where the specification can talk about all variables, hidden and visible. It also shows that CL does not capture in a natural way ATL with imperfect information as defined in [1, Section 7.1], where imperfect information of agents is modelled by defining which atomic propositions they can observe. This, as well as unobservable objectives, can be naturally modelled in SL_{ii}.

7 APPLICATIONS

In this section we apply Theorem 2.9 to decide the existence of Nash Equilibria in hierarchical games of imperfect information. We then use a similar approach to obtain decidability results for the rational synthesis problem. In this section, for a tuple of agents $\boldsymbol{a} = (a_i)_{i \in [m]}$ and tuple of strategy variables $\boldsymbol{x} = (x_i)_{i \in [m]}$, we let $(\boldsymbol{a}, \boldsymbol{x})$ be a macro for $(a_1, x_1) \ldots (a_m, x_m)$, and similarly for the unbinding operator $(\boldsymbol{a}, ?)$ which stands for $(a_1, ?) \ldots (a_m, ?)$.

7.1 Existence of Nash Equilibria in games with hierarchical observations

A Nash equilibrium in a game is a tuple of strategies such that no player has an incentive to deviate. Let $\text{Ag} = \{a_i : i \in [n]\}$. Assuming that agent a_i has observation o_i and LTL goal ψ_i, the following SL_{ii} formula expresses the existence of a Nash equilibrium:

$$\Phi_{\text{NE}} := \langle\!\langle x_1 \rangle\!\rangle^{o_1} \ldots \langle\!\langle x_n \rangle\!\rangle^{o_n} (\boldsymbol{a}, \boldsymbol{x}) \bigwedge_{i \in [n]} \left[\left(\langle\!\langle y_i \rangle\!\rangle^{o_i} (a_i, y_i) \, \mathbf{A} \psi_i \right) \to \mathbf{A} \psi_i \right]$$

where $\boldsymbol{a} = (a_i)_{i \in [n]}$ and $\boldsymbol{x} = (x_i)_{i \in [n]}$.

Nash equilibria do not always exist when one restricts attention to pure strategies, as we do in this work. This is the case already in finite games, and by extension also in the infinite concurrent games played on graphs that we consider. This motivates the study of the Nash equilibria existence problem in such games. In the perfect information case, the problem has been solved for ω-regular objectives, as well as more complex semi-quantitative objectives [12]. When moving to imperfect information, for two players the problem is decidable for LTL objectives [38] and parity objectives [30]. However, as for distributed synthesis, existence of Nash equilibria becomes undecidable for more than two players. This result is proved in [11] for constrained Nash equilibria (when one specifies for each player whether her objective is satisfied or not), and in [38] for unconstrained equilibria. In both cases the proof proceeds by reduction from the distributed synthesis problem [63, 69].

The only known decidable cases for more than two players assume that all players receive the same information. In [11] the problem is solved on games where players observe the evolution of the game via *public signals* and objectives are given by visible parity conditions or mean-payoff functions. In [4], an epistemic extension of strategy logic is used to solve the existence of Nash equilibria on games with *broadcast actions* for objectives given as formulas from epistemic temporal logic. A stronger notion of Nash equilibria, called *locally consistent equilibria*, is studied in [72]. In a locally consistent equilibrium, each player's strategy has to be a best response not only to other players' strategies in the equilibrium, but also to all strategies that are indistinguishable from those in the equilibrium. It is proved in [72] that the existence of such equilibria is decidable on a model of games close in spirit to those with public signals studied in [11].

Here we show that the existence of Nash equilibria is decidable for n players when observations are hierarchical and objectives are given as LTL formulas. Note that this result is orthogonal to those described above, which all allow in a way or another some non-hierarchical information: in [11] players know their own actions in addition to the public signals, in [72] they know their private local state, and in [4] they can have incomparable initial knowledge of the situation.

Definition 7.1. A CGS_{ii} $\mathcal{G}$ presents *hierarchical observation* [8] if the "finer-than" relation is a total ordering, i.e., if for all $o, o' \in \text{Obs}$, either $O(o) \subseteq O(o')$ or $O(o') \subseteq O(o)$.

Let $\mathcal{G}$ be a CGS_{ii} with hierarchical observation, and since all agents have symmetric roles in the problem considered, assume without loss of generality that $O(o_n) \subseteq \ldots \subseteq O(o_1)$.

Because of the nested strategy quantifiers $\langle\!\langle y_i \rangle\!\rangle^{o_i}$, the instance $(\mathcal{G}, \Phi_{\text{NE}})$ is *not* hierarchical even if $\mathcal{G}$ yields hierarchical observation (unless $O(o_i) = O(o_j)$ for all $i, j \in [n]$). However, considering

the special observation symbol o_p that is always interpreted as the identity relation (and thus represents perfect observation), and letting

$$\Phi' := \langle\!\langle x_1 \rangle\!\rangle^{o_1} \ldots \langle\!\langle x_n \rangle\!\rangle^{o_n} (\boldsymbol{a}, \boldsymbol{x}) \bigwedge_{i \in [n]} \left[\left(\langle\!\langle y_i \rangle\!\rangle^{o_p} (a_i, y_i) \, \mathbf{E} \psi_i \right) \to \mathbf{E} \psi_i \right],$$

we have that Φ' forms a hierarchical instance with any CGS$_{ii}$ that presents hierarchical observation. Besides, we can prove that for deterministic strategies, Φ' is equivalent to Φ_{NE}:

LEMMA 7.2. *If we consider deterministic strategies, then $\Phi_{NE} \equiv \Phi'$.*

PROOF. Concerning the universal versus existential quantification on outcomes, it is enough to observe that assigning a deterministic strategy to each agent determines a unique outcome. Next, to change each inner o_i for o_p, we exploit the fact that in a one-player game of partial observation (such a game occurs when all but one player have fixed their strategies), the player has a strategy enforcing some goal iff she has a uniform strategy enforcing that goal.

To see this, it is enough to establish that for every CGS$_{ii}$ $\mathcal{G}$ and position v,

$$\mathcal{G}, \chi, v \models \langle\!\langle y_i \rangle\!\rangle^{o_p} (a_i, y_i) \, \mathbf{E} \psi_i \leftrightarrow \langle\!\langle y_i \rangle\!\rangle^{o_i} (a_i, y_i) \, \mathbf{E} \psi_i,$$

for every $i \in [n]$ and every assignment χ such that $\chi(a_j)$ is defined for all j.

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 an o_p-strategy (i.e., a perfect-information strategy) such that $\mathcal{G}, \chi', v_\iota \models \psi_i$, where $\chi' = \chi[y_i \mapsto \sigma, a_i \mapsto \sigma]$. Because we consider deterministic strategies and χ' assigns a strategy to each agent, it defines a unique outcome π from the initial position, i.e., $\mathrm{Out}(\chi', v_\iota) = \{\pi\}$. We construct an o_i-strategy σ' such that if a_i uses it instead of σ, we obtain the same outcome π, i.e., $\mathrm{Out}(\chi'', v_\iota) = \{\pi\}$, where $\chi'' = \chi[y_i \mapsto \sigma', a_i \mapsto \sigma']$. This can be done as follows: if $\rho \sim_{o_i} \pi_{\leq |\rho|-1}$ then define $\sigma'(\rho) := \sigma(\pi_{\leq |\rho|-1})$, and otherwise let $\sigma'(\rho) := c$ for some fixed action $c \in \mathrm{Ac}$. It is easy to see that σ' is an o_i-strategy and that χ'' produces the same outcome as χ from v_ι. □

COROLLARY 7.3. *If we consider deterministic strategies, then the existence of Nash Equilibria in games with hierarchical observation and k different observations is in $(k + 1)$-ExpTime.*

PROOF. Deciding the existence of a Nash Equilibrium in a CGS$_{ii}$ $\mathcal{G}$ amounts to model-checking formula Φ_{NE} in $\mathcal{G}$, which by Lemma 7.2 is equivalent to model-checking Φ' in $\mathcal{G}$ if we restrict to deterministic strategies. Because Φ' forms hierarchical instances with games that yield hierarchical observation, by Theorem 2.9 we can model check it on such games. Now because each ψ_i is an LTL formula, we have that

$$\mathrm{sd} \left(\langle\!\langle y_i \rangle\!\rangle^{o_p} (a_i, y_i) \, \mathbf{E} \psi_i \right) = (0, \mathrm{nd}),$$

$$\mathrm{sd} \left(\bigwedge_{i \in [n]} \left[\left(\langle\!\langle y_i \rangle\!\rangle^{o_p} (a_i, y_i) \, \mathbf{E} \psi_i \right) \to \mathbf{E} \psi_i \right] \right) = (0, \mathrm{alt}),$$

and finally we obtain that $\mathrm{sd}(\Phi') = (k, \mathrm{nd})$, where k is the number of different observations in $\mathcal{G}$, i.e., $k = |\{O(o_1), \ldots, O(o_n)\}|$. By Proposition 5.2, we can model check Φ' on $\mathcal{G}$ in time $(k + 1)$-exponential, which concludes. □

We now show that, using the same trick, our main result can be applied to solve a more general problem called *rational synthesis*.

7.2 Rational distributed synthesis in games with hierarchical observations

In classic synthesis, the environment is considered monolithic and "hostile", in the sense that the system to be synthesised should be able to deal with all possible behaviours of the environment, even the most undesirable ones. This is a very strong requirement that can not always be met. When the environment can be considered rational, and its objective is known, it is reasonable to relax this requirement by asking that the system to synthesise behave well against the *rational* behaviours of the environment. This problem is known as the *rational synthesis* problem [23, 30, 33, 48]. In the setting considered in the works above-mentioned, the system is seen as an agent a and the environment is composed of several components, say $\{e_1, \ldots, e_m\}$, that are assumed to be rational and follow individual objectives. While [23] and [30] consider various types of objectives such as reachability, safety or parity, here we consider LTL objectives as is done in [33, 48]: the specification for the system is an LTL formula ψ_g, and the objective of each component e_i of the environment is an LTL formula ψ_i. However note that the decidability results we establish would also hold for arbitrary omega-regular objectives.

7.2.1 Rational synthesis: state of the art. Two variants of the rational synthesis problem have been considered: the *cooperative* one, in which it is possible to tell the environment how to behave, as long as the suggested behaviour for each component forms an equilibrium, and the *non-cooperative* one, in which the components of the environment may have any behaviour that forms an equilibrium. The existence of a solution to these problems can be expressed by the formulas $\Phi_{\text{c-RS}}$ and $\Phi_{\text{nc-RS}}$, respectively, defined as follows:

$$\Phi_{\text{c-RS}} := \langle\!\langle x \rangle\!\rangle^{o_p} \langle\!\langle y_1 \rangle\!\rangle^{o_p} \ldots \langle\!\langle y_m \rangle\!\rangle^{o_p} (a, x)(\boldsymbol{e}, \boldsymbol{y}) \, \varphi_\gamma \wedge \mathbf{A}\psi_g$$

$$\Phi_{\text{nc-RS}} := \langle\!\langle x \rangle\!\rangle^{o_p} [\![y_1]\!]^{o_p} \ldots [\![y_m]\!]^{o_p} (a, x)(\boldsymbol{e}, \boldsymbol{y}) \, \varphi_\gamma \to \mathbf{A}\psi_g$$

where $\boldsymbol{e} = (e_i)_{i \in [m]}$, $\boldsymbol{y} = (y_i)_{i \in [m]}$, and φ_γ expresses that $\boldsymbol{y}$ forms an equilibrium for the environment. Also, as in the previous section, o_p represents the perfect-information observation. Three different kinds of equilibria are considered in [48]: profiles of dominant strategies, Nash equilibria, and subgame-perfect equilibria. Here we only consider Nash equilibria, because subgames of games with imperfect information should start in situations where all players have perfect information of the state, which we do not know how to express in SL$_{\text{ii}}$; and for dominant strategies, the natural formula to express them does not give rise to non-trivial decidable cases in the imperfect-information setting that we introduce later. The rational synthesis problem for Nash equilibria is obtained by replacing φ_γ in the above formula with:

$$\varphi_{\text{NE}} := \bigwedge_{i \in [m]} \left[\left(\langle\!\langle y_i' \rangle\!\rangle^{o_p} (e_i, y_i') \, \mathbf{A}\psi_i \right) \to \mathbf{A}\psi_i \right]$$

It is proved in [48] that these problems are decidable for perfect information. Concerning imperfect information, because the existence of Nash equilibria is undecidable for three players, the problem is undecidable when the environment consists of at least three components [30]. Three decidable cases are known: when the environment consists of a single component [30], when actions of all components are public [4], and when only the system has imperfect information while the (finitely many) components of the environment are perfectly informed [30].

We now extend the latter result by defining a generalisation of the rational synthesis problem that we call *rational distributed synthesis*, and solving it in the case of hierarchical information. The case where the environment is perfectly informed and the system consists of a single component, solved in [30], is a particular case of our Corollary 7.5 below[6]. However the other decidability

---

[6]We only consider LTL objectives, but our automata construction can be adapted to handle all ω-regular objectives.

result established in [30] does not assume hierarchical information, and thus cannot be derived from the results we now present.

7.2.2 Rational distributed synthesis. While for perfect information, distributed synthesis amounts to synthesis for a single meta-component which tells each component what to do, in the context of imperfect information it makes sense to consider that the system to be synthesised is composed of various components $\{a_1, \ldots, a_n\}$ with different observation power, say o_i for component a_i. We also let o_i^e be the observation of the environment's component e_i, for $i \in [m]$.

We consider the imperfect-information variants of cooperative and non-cooperative rational synthesis defined by the following SL_{ii} formulas:

$$\Phi_{\text{c-RS}}^{\text{ii}} := \langle\!\langle x_1 \rangle\!\rangle^{o_1} \ldots \langle\!\langle x_n \rangle\!\rangle^{o_n} \langle\!\langle y_1 \rangle\!\rangle^{o_1^e} \ldots \langle\!\langle y_m \rangle\!\rangle^{o_m^e} (a, x)(e, y) \, \varphi_\gamma \wedge \mathbf{A}\psi_g$$

$$\Phi_{\text{nc-RS}}^{\text{ii}} := \langle\!\langle x \rangle\!\rangle^{o_1} \ldots \langle\!\langle x_n \rangle\!\rangle^{o_n} [\![y_1]\!]^{o_1^e} \ldots [\![y_m]\!]^{o_m^e} (a, x)(e, y) \, \varphi_\gamma \rightarrow \mathbf{A}\psi_g$$

The formula for Nash equilibrium is adapted as follows:

$$\varphi_{\text{NE}}^{\text{ii}} := \bigwedge_{i \in [m]} \left[\left(\langle\!\langle y_i' \rangle\!\rangle^{o_i^e} (e_i, y_i') \, \mathbf{A}\psi_i \right) \rightarrow \mathbf{A}\psi_i \right]$$

The only difference with the perfect-information case is that we use the observation of the different components of the environment instead of the perfect-information observation.

We call the problems expressed by formulas $\Phi_{\text{c-RS}}^{\text{ii}}$ and $\Phi_{\text{nc-RS}}^{\text{ii}}$ *cooperative rational distributed synthesis* and *non-cooperative rational distributed synthesis*, respectively. As in the previous section on the existence of Nash equilibria, one can see that even if there is a total hierarchy on all observations, these formula do not yield hierarchical instances unless all observations are the same. However, the trick applied in the proof of Corollary 7.3 also applies here, both for Nash equilibria and subgame-perfect equilibria, i.e., we can replace each o_i^e with o_p in $\varphi_{\text{NE}}^{\text{ii}}$ without affecting the semantics of formulas $\Phi_{\text{c-RS}}^{\text{ii}}$ and $\Phi_{\text{nc-RS}}^{\text{ii}}$. As a result, when there is a hierarchy on observations $o_1, \ldots, o_n, o_1^e, \ldots, o_m^e$, the cooperative rational distributed synthesis is decidable.

COROLLARY 7.4. *If we consider deterministic strategies and hierarchical observations, then cooperative rational distributed synthesis is decidable.*

For the non-cooperative variant, one cannot switch universal quantifications on strategies for the environments with existential quantifications for the system in order to obtain hierarchical instances, as the resulting formula would then capture a different problem. As a consequence, in addition to a hierarchy on observations $o_1, \ldots, o_n, o_1^e, \ldots, o_m^e$, we need to have that the components of the environment observe better than the components of the system or, in other words, that the least informed component of the environment observes better than the best informed component of the system. When it is the case, we say that the environment is *more informed* than the system.

COROLLARY 7.5. *Non-cooperative rational distributed synthesis is decidable for deterministic strategies and hierarchical observations where the environment is more informed than the system.*

This result applies for instance when there is hierarchical information amongst the components of the system, and the environment has perfect information. Note that when the system consists of a single component, this corresponds to the second decidability result in [30]. As we mentioned in the introduction, considering that the opponent has perfect information is something classic in two-player games with imperfect information, as doing so ensures that the strategy one synthesises is winning no matter how much the opponent observes. In Reif's words, this amounts to considering the possibility that the opponent may "cheat" and use information that it normally does not have access to [73]. The non-cooperative rational synthesis problem is not precisely a

two-player game, but it resembles one in the sense that the system as a whole (composed of its various components $a_1, \ldots, a_n$) should win against any "rational" behaviour of the environment as a whole. In this view, considering that the components of the environment have perfect information thus yields a distributed system that is robust to possible leaks of hidden information to the environment.

Remark 6. When all components of the environment have perfect information, $\Phi^{ii}_{c\text{-RS}}$ and $\Phi^{ii}_{nc\text{-RS}}$ already form hierarchical instances with games where there is hierarchical observation amongst the system's components, and one does not need to resort to the trick used in the proof of Corollary 7.3. A consequence is that in that case, corollaries 7.4 and 7.5 also hold for nondeterministic strategies.

8 CONCLUSION

We introduced SL_{ii}, a logic for reasoning about strategic behaviour in multi-player games with imperfect information. The syntax specifies the observations with which strategies have to work, and thus allows one to reason about strategic problems in settings where agents can change observation power, for instance by being eventually granted access to previously hidden information. Moreover our logic contains an outcome quantifier and an unbinding operator which simplify the semantics, make it easier to express branching-time properties, allow us to naturally consider nondeterministic strategies, and make the correspondence with $QCTL^*_{ii}$ tighter, enabling us to derive precise complexity results for the model-checking of SL_{ii}.

We isolated the class of hierarchical formula/model pairs $(\Phi, \mathcal{G})$ and proved that for such instances one can decide whether $\mathcal{G} \models \Phi$. The proof reduces (hierarchical) instances of SL_{ii} to (hierarchical) formulas of $QCTL^*_{ii}$, a low-level logic that we introduced, and that serves as a natural bridge between SL_{ii} and automata constructions. We also studied in detail the complexity of the model-checking problems solved in this work. To do so we introduced a new measure on formulas called *simulation depth*. This measure, though being a purely syntactic notion, reflects the complexity of automata constructions required to treat a given formula.

Since one can alternate quantifiers in SL_{ii}, our decidability result goes beyond synthesis and can be used to easily obtain the decidability of many strategic problems. In this work we applied it to the problem of existence of Nash equilibria in games with hierarchical observation, and to the imperfect-information generalisations of rational synthesis that we called (cooperative and non-cooperative) *rational distributed synthesis*. Our result has also been used to prove that the existence of admissible strategies in games with hierarchical information is decidable [14].

An interesting direction for future work would be to try and adapt the notion of hierarchical instances to allow for situations in which hierarchies can change along a play, as done in [8]. We would also like to consider alternatives to the synchronous perfect recall setting considered here, such as the classic asynchronous perfect recall setting [29, 70], or the more recent notion of causal knowledge [36]. Finally, it is often interesting in presence of imperfect information to introduce epistemic operators to reason explicitly about what agents know. We already generalised the main result of this work to an extension of SL_{ii} with such operators [59]; we would like to see if this can be used to reason about subgame-perfect equilibria in games with imperfect information, which do not seem to be easy to characterise in SL_{ii}, as mentioned in Section 7.2.1. Indeed, in games with imperfect information, the notion of subgame specifies that the initial situation should be known to all players [76], a property that epistemic logics are meant to be able to express.

ACKNOWLEDGMENTS

REFERENCES

[1] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. 2002. Alternating-time temporal logic. *J. ACM* 49, 5 (2002), 672–713. https://doi.org/10.1145/585265.585270

[2] Francesco Belardinelli. 2014. Reasoning about Knowledge and Strategies: Epistemic Strategy Logic. In *SR'14*. 27–33. https://doi.org/10.4204/EPTCS.146.4

[3] Francesco Belardinelli. 2015. A Logic of Knowledge and Strategies with Imperfect Information. In *LAMAS'15*. 1–15.

[4] Francesco Belardinelli, Alessio Lomuscio, Aniello Murano, and Sasha Rubin. 2017. Verification of Broadcasting Multi-Agent Systems against an Epistemic Strategy Logic. In *IJCAI'17*. 91–97. https://doi.org/10.24963/ijcai.2017/14

[5] Francesco Belardinelli, Alessio Lomuscio, Aniello Murano, and Sasha Rubin. 2017. Verification of Multi-agent Systems with Imperfect Information and Public Actions. In *AAMAS'17*. 1268–1276.

[6] Raphael Berthon, Bastien Maubert, Aniello Murano, Sasha Rubin, and Moshe Y. Vardi. 2017. Strategy Logic with imperfect information. In *LICS'17*. IEEE, 1–12. https://doi.org/10.1109/LICS.2017.8005136

[7] Dietmar Berwanger, Krishnendu Chatterjee, Martin De Wulf, Laurent Doyen, and Thomas A Henzinger. 2010. Strategy construction for parity games with imperfect information. *Information and computation* 208, 10 (2010), 1206–1220.

[8] Dietmar Berwanger, Anup Basil Mathew, and Marie van den Bogaard. 2018. Hierarchical information and the synthesis of distributed strategies. *Acta Inf.* 55, 8 (2018), 669–701. https://doi.org/10.1007/s00236-017-0306-5

[9] Benjamin Bittner, Marco Bozzano, Alessandro Cimatti, and Xavier Olive. 2012. Symbolic Synthesis of Observability Requirements for Diagnosability. In *AAAI'12*.

[10] Patricia Bouyer. 2017. Games on graphs with a public signal monitoring. *arXiv preprint arXiv:1710.07163* (2017).

[11] Patricia Bouyer. 2018. Games on Graphs with a Public Signal Monitoring. In *FOSSACS'18*. Springer, 530–547. https://doi.org/10.1007/978-3-319-89366-2_29

[12] Patricia Bouyer, Romain Brenguier, Nicolas Markey, and Michael Ummels. 2015. Pure Nash Equilibria in Concurrent Deterministic Games. *Logical Methods in Computer Science* 11, 2 (2015). https://doi.org/10.2168/LMCS-11(2:9)2015

[13] Patricia Bouyer, Nicolas Markey, and Steen Vester. 2017. Nash equilibria in symmetric graph games with partial observation. *Information and Computation* 254 (2017), 238–258.

[14] Romain Brenguier, Arno Pauly, Jean-François Raskin, and Ocan Sankur. 2017. Admissibility in Games with Imperfect Information. In *CONCUR'17*, Vol. 85.

[15] Nils Bulling and Wojciech Jamroga. 2014. Comparing variants of strategic ability: how uncertainty and memory influence general properties of games. *AAMAS'14* 28, 3 (2014), 474–518.

[16] Krishnendu Chatterjee and Laurent Doyen. 2010. The complexity of partial-observation parity games. In *International Conference on Logic for Programming Artificial Intelligence and Reasoning*. Springer, 1–14.

[17] Krishnendu Chatterjee and Laurent Doyen. 2014. Games with a Weak Adversary. In *ICALP'14*. 110–121. https://doi.org/10.1007/978-3-662-43951-7_10

[18] Krishnendu Chatterjee and Laurent Doyen. 2014. Partial-observation stochastic games: How to win when belief fails. *ACM Transactions on Computational Logic (TOCL)* 15, 2 (2014), 16. https://doi.org/10.1145/2579821

[19] Krishnendu Chatterjee, Laurent Doyen, Emmanuel Filiot, and Jean-François Raskin. 2017. Doomsday equilibria for omega-regular games. *Inf. Comput.* 254 (2017), 296–315. https://doi.org/10.1016/j.ic.2016.10.012

[20] Krishnendu Chatterjee, Thomas A Henzinger, and Nir Piterman. 2010. Strategy logic. *Information and Computation* 208 (2010).

[21] Krishnendu Chatterjee, Thomas A. Henzinger, and Nir Piterman. 2010. Strategy Logic. *Inf. Comput.* 208, 6 (2010), 677–693. https://doi.org/10.1016/j.ic.2009.07.004

[22] Edmund M Clarke, Orna Grumberg, and Doron Peled. 1999. *Model checking*. MIT press.

[23] Rodica Condurache, Emmanuel Filiot, Raffaella Gentilini, and Jean-François Raskin. 2016. The Complexity of Rational Synthesis. In *ICALP'16*. 121:1–121:15. https://doi.org/10.4230/LIPIcs.ICALP.2016.121

[24] Aldric Degorre, Laurent Doyen, Raffaella Gentilini, Jean-François Raskin, and Szymon Toruńczyk. 2010. Energy and mean-payoff games with imperfect information. In *CSL'10*. Springer, 260–274.

[25] Catalin Dima and Ferucio Laurentiu Tiplea. 2011. Model-checking ATL under Imperfect Information and Perfect Recall Semantics is Undecidable. *CoRR* (2011). arXiv:1102.4225

[26] Laurent Doyen and Jean-François Raskin. 2011. Games with imperfect information: Theory and algorithms. *Lectures in Game Theory for Computer Scientists* (2011), 185–212.

[27] Calvin C. Elgot and Michael O. Rabin. 1966. Decidability and Undecidability of Extensions of Second (First) Order Theory of (Generalized) Successor. *JSL* 31, 2 (1966), 169–181. https://doi.org/10.2307/2269808

[28] E Allen Emerson and Joseph Y Halpern. 1986. "Sometimes" and "not never" revisited: on branching versus linear time temporal logic. *Journal of the ACM (JACM)* 33, 1 (1986), 151–178.

[29] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. 1995. *Reasoning about knowledge.* Vol. 4. MIT press Cambridge.

[30] Emmanuel Filiot, Raffaella Gentilini, and Jean-François Raskin. 2018. Rational Synthesis Under Imperfect Information. In *LICS'18.* ACM, 422–431.

[31] Bernd Finkbeiner and Sven Schewe. 2005. Uniform Distributed Synthesis. In *LICS'05.* 321–330. https://doi.org/10.1109/LICS.2005.53

[32] Bernd Finkbeiner and Sven Schewe. 2010. Coordination Logic. In *CSL'10.* 305–319. https://doi.org/10.1007/978-3-642-15205-4_25

[33] Dana Fisman, Orna Kupferman, and Yoad Lustig. 2010. Rational synthesis. In *TACAS'10.* Springer, 190–204. https://doi.org/10.1007/978-3-642-12002-2_16

[34] Tim French. 2001. Decidability of quantifed propositional branching time logics. In *AJCAI'01.* 165–176. https://doi.org/10.1007/3-540-45656-2_15

[35] Paul Gastin, Nathalie Sznajder, and Marc Zeitoun. 2009. Distributed synthesis for well-connected architectures. *FMSD* 34, 3 (2009), 215–237. https://doi.org/10.1007/s10703-008-0064-7

[36] Blaise Genest, Doron Peled, and Sven Schewe. 2015. Knowledge= observation+ memory+ computation. In *International Conference on Foundations of Software Science and Computation Structures.* Springer, 215–229.

[37] Dimitar P. Guelev, Catalin Dima, and Constantin Enea. 2011. An alternating-time temporal logic with knowledge, perfect recall and past: axiomatisation and model-checking. *Journal of Applied Non-Classical Logics* 21, 1 (2011), 93–131. https://doi.org/10.3166/jancl.21.93-131

[38] Julian Gutierrez, Giuseppe Perelli, and Michael Wooldridge. 2018. Imperfect information in Reactive Modules games. *Inf. Comput.* 261, Part (2018), 650–675. https://doi.org/10.1016/j.ic.2018.02.023

[39] Joseph Y. Halpern and Moshe Y. Vardi. 1989. The complexity of reasoning about knowledge and time. I. Lower bounds. *JCSS* 38, 1 (1989), 195–237.

[40] Xiaowei Huang and Ron Van Der Meyden. 2014. A Temporal Logic of Strategic Knowledge. In *KR'14.*

[41] W. Jamroga and N. Bulling. 2011. Comparing variants of strategic ability. In *IJCAI'11.* AAAI Press, 252–257. https://doi.org/10.1023/A:1026171312755

[42] Wojciech Jamroga and Aniello Murano. 2014. On module checking and strategies. In *AAMAS'14.* International Foundation for Autonomous Agents and Multiagent Systems, 701–708.

[43] Wojciech Jamroga and Aniello Murano. 2015. Module checking of strategic ability. In *AAMAS'15.* International Foundation for Autonomous Agents and Multiagent Systems, 227–235.

[44] Wojciech Jamroga and Wiebe van der Hoek. 2004. Agents that Know How to Play. *Fundam. Inform.* 63, 2-3 (2004), 185–219.

[45] Sophia Knight and Bastien Maubert. 2019. Dealing with imperfect information in Strategy Logic. arXiv:arXiv:1908.02488 Presented at SR'15.

[46] Orna Kupferman. 1999. Augmenting branching temporal logics with existential quantification over atomic propositions. *JLC* 9, 2 (1999), 135–147. https://doi.org/10.1093/logcom/9.2.135

[47] Orna Kupferman, Parthasarathy Madhusudan, Pazhamaneri Subramaniam Thiagarajan, and Moshe Y. Vardi. 2000. Open Systems in Reactive Environments: Control and Synthesis. In *CONCUR'00.* 92–107.

[48] Orna Kupferman, Giuseppe Perelli, and Moshe Y. Vardi. 2016. Synthesis with rational environments. *Ann. Math. Artif. Intell.* 78, 1 (2016), 3–20. https://doi.org/10.1007/s10472-016-9508-8

[49] Orna Kupferman and Moshe Y. Vardi. 1999. Church's problem revisited. *BSL* (1999), 245–263.

[50] Orna Kupferman and Moshe Y. Vardi. 2001. Synthesizing distributed systems. In *LICS'01.* 389–398. https://doi.org/10.1109/LICS.2001.932514

[51] Orna Kupferman, Moshe Y. Vardi, and Pierre Wolper. 2000. An automata-theoretic approach to branching-time model checking. *JACM* 47, 2 (2000), 312–360. https://doi.org/10.1145/333979.333987

[52] Orna Kupferman, Moshe Y Vardi, and Pierre Wolper. 2001. Module checking. *Information and Computation* 164, 2 (2001), 322–344.

[53] François Laroussinie and Nicolas Markey. 2014. Quantified CTL: Expressiveness and Complexity. *LMCS* 10, 4 (2014). https://doi.org/10.2168/LMCS-10(4:17)2014

[54] François Laroussinie and Nicolas Markey. 2015. Augmenting ATL with strategy contexts. *Inf. Comput.* 245 (2015), 98–123. https://doi.org/10.1016/j.ic.2014.12.020

[55] François Laroussinie, Nicolas Markey, and Arnaud Sangnier. 2015. ATLsc with partial observation. In *GandALF'15.* 43–57. https://doi.org/10.4204/EPTCS.193.4

[56] Hans Läuchli and Christian Savioz. 1987. Monadic second order definable relations on the binary tree. *JSL* 52, 01 (1987), 219–226. https://doi.org/10.2307/2273878

[57] Christof Löding. 2011. Automata on Infinite Trees. In *preliminary version for the handbook Automata: from Mathematics to Applications*, Jean-Eric Pin (Ed.).

[58] Alessio Lomuscio and Franco Raimondi. 2006. MCMAS : A Model Checker for Multi-agent Systems. In *TACAS'06 (LNCS 4314)*. 450–454.

[59] Bastien Maubert and Aniello Murano. 2018. Reasoning about knowledge and strategies under hierarchical information. In *KR'18*.

[60] Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y. Vardi. 2014. Reasoning About Strategies: On the Model-Checking Problem. *ACM Trans. Comput. Log.* 15, 4 (2014), 34:1–34:47. https://doi.org/10.1145/2631917

[61] David E. Muller and Paul E. Schupp. 1995. Simulating Alternating Tree Automata by Nondeterministic Automata: New Results and New Proofs of the Theorems of Rabin, McNaughton and Safra. *TCS* 141, 1&2 (1995), 69–107. https://doi.org/10.1016/0304-3975(94)00214-4

[62] Guillermo A Pérez. 2017. The fixed initial credit problem for partial-observation energy games is Ack-complete. *Inform. Process. Lett.* 118 (2017), 91–99.

[63] Gary Peterson, John Reif, and Salman Azhar. 2001. Lower bounds for multiplayer noncooperative games of incomplete information. *CAMWA* 41, 7 (2001), 957–992. https://doi.org/10.1016/S0898-1221(00)00333-3

[64] Gary Peterson, John Reif, and Salman Azhar. 2002. Decision algorithms for multiplayer noncooperative games of incomplete information. *CAMWA* 43, 1 (2002), 179–206. https://doi.org/10.1016/S0898-1221(01)00282-6

[65] Gary L. Peterson and John H. Reif. 1979. Multiple-Person Alternation. In *SFCS'79*. 348–363. https://doi.org/10.1109/SFCS.1979.25

[66] Sophie Pinchinat and Stéphane Riedweg. 2005. A decidable class of problems for control under partial observation. *IPL* 95, 4 (2005), 454–460. https://doi.org/10.1016/j.ipl.2005.04.011

[67] Amir Pnueli. 1977. The Temporal Logic of Programs. In *FOCS*. 46–57.

[68] Amir Pnueli and Roni Rosner. 1989. On the synthesis of a reactive module. In *POPL*. 179–190.

[69] Amir Pnueli and Roni Rosner. 1990. Distributed reactive systems are hard to synthesize. In *FOCS'90*. 746–757. https://doi.org/10.1109/FSCS.1990.89597

[70] Bernd Puchala. 2010. Asynchronous Omega-Regular Games with Partial Information. In *MFCS*. 592–603.

[71] Michael O Rabin. 1969. Decidability of second-order theories and automata on infinite trees. *TAMS* 141 (1969), 1–35. https://doi.org/10.1090/S0002-9947-1969-0246760-1

[72] Ramaswamy Ramanujam and Sunil Simon. 2010. A communication based model for games of imperfect information. In *International Conference on Concurrency Theory*. Springer, 509–523.

[73] John H Reif. 1984. The complexity of two-player games of incomplete information. *Journal of computer and system sciences* 29, 2 (1984), 274–301. https://doi.org/10.1016/0022-0000(84)90034-5

[74] Sven Schewe and Bernd Finkbeiner. 2007. Distributed Synthesis for Alternating-Time Logics. In *ATVA'07*. 268–283. https://doi.org/10.1007/978-3-540-75596-8_20

[75] Pierre-Yves Schobbens. 2004. Alternating-time logic with imperfect recall. *Electr. Notes Theor. Comput. Sci.* 85, 2 (2004), 82–93. https://doi.org/10.1016/S1571-0661(05)82604-0

[76] Reinhard Selten. 1965. Spieltheoretische behandlung eines oligopolmodells mit nachfrageträgheit: Teil i: Bestimmung des dynamischen preisgleichgewichts. *Zeitschrift für die gesamte Staatswissenschaft/Journal of Institutional and Theoretical Economics* H. 2 (1965), 301–324.

[77] A Prasad Sistla. 1983. *Theoretical Issues in the Design and Certification of Distributed Systems.* Ph.D. Dissertation. Harvard University, Cambridge, MA, USA.

[78] Wolfgang Thomas. 1992. Infinite Trees and Automaton-Definable Relations over omega-Words. *TCS* 103, 1 (1992), 143–159. https://doi.org/10.1016/0304-3975(92)90090-3

[79] Ron van der Meyden and Moshe Y. Vardi. 1998. Synthesis from knowledge-based specifications. In *CONCUR'98*. Springer, 34–49.

[80] Ron van der Meyden and Thomas Wilke. 2005. Synthesis of Distributed Systems from Knowledge-Based Specifications. In *CONCUR'05*. 562–576.

[81] Moshe Y. Vardi and Pierre Wolper. 1994. Reasoning about infinite computations. *IC* 115, 1 (1994), 1–37.

[82] Wieslaw Zielonka. 1998. Infinite Games on Finitely Coloured Graphs with Applications to Automata on Infinite Trees. *TCS* 200, 1-2 (1998), 135–183.

A PROOF OF PROPOSITION 4.12

First, for every LTL formula ψ one can build a parity word automaton $\mathcal{W}^\psi$ with two colours and $2^{O(|\psi|)}$ states [81]. Let $K_\psi \in \mathbb{N}$ be such that the number of states of $\mathcal{W}^\psi$ is bounded by $2^{K_\psi|\psi|}$.

We also state a more precise version of Theorem 4.6: for every ATA $\mathcal{A}$ with n states and l colours, one can build an NTA $\mathcal{N}$ with at most $2^{O(nl\log(nl))}$ states and $O(nl)$ colours such that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{N})$ [57, 61]. We let $K_1, K_2 \in \mathbb{N}$ be such that the number of states of $\mathcal{N}$ is bounded by $2^{K_1 nl\log(nl)}$ and the number of colours by $K_2 nl$.

Proposition 4.12 follows directly from the following.

PROPOSITION A.1. *Let* Φ *be a* QCTL$^*_{i,\subseteq}$ *formula,* $\mathcal{S}$ *a CKS, and let* AP$_\exists$ = AP$_\exists(\Phi)$. *For every sub-formula* φ *of* Φ *and state* $s \in \mathcal{S}$, *it holds that:*

- *if* $sd_k(\varphi) = 0$, $\mathcal{A}^\varphi_s$ *has at most* $f^\varphi_\mathcal{S}$ *states and 2 colours,*
- *if* $sd_k(\varphi) \geq 1$, $\mathcal{A}^\varphi_s$ *has at most* $\exp(sd_k(\varphi) \mid f^\varphi_\mathcal{S} \log f^\varphi_\mathcal{S})$ *states, and its number of colours is at most* $\exp(sd_k(\varphi) - 1 \mid f^\varphi_\mathcal{S} \log f^\varphi_\mathcal{S})$,

with $f^\varphi_\mathcal{S} = (4K_1 + 2K_2)^{\exists d(\varphi)}|\varphi||\mathcal{S}|^{\mathbf{Ed}(\varphi)}2^{K_\psi|\varphi|\mathbf{Ed}(\varphi)}$.

In addition, if $\mathcal{A}^\varphi_s$ *has state set* Q, *for each* $q \in Q$ *and* $a \in 2^{AP_\exists}$, *we have* $|\delta(q, a)| \leq |\mathcal{S}||Q|^{|\mathcal{S}|}2^{H|\varphi|}$, *where* $H = 1 + \mathbf{Ed}(\varphi)$.

PROOF. We prove the result by induction on φ.

$\boldsymbol{\varphi = p}$: in this case $sd_k(\varphi) = \exists d(\varphi) = \mathbf{Ed}(\varphi) = 0$. By construction, $\mathcal{A}^\varphi_s$ has one state q_ι and two colours, so that the first part of the claim holds. In addition, each formula of its transition function is of size one, so that the second part of the claim also holds.

$\boldsymbol{\varphi = \neg\varphi'}$: Complementing an ATA does not change the number of states, number of colours or size of formulas in the transition function, so that the result follows by induction hypothesis and the fact that $|\varphi'| \leq |\varphi|$ and $\mathbf{Ed}(\varphi) = \mathbf{Ed}(\varphi')$.

$\boldsymbol{\varphi = \varphi_1 \vee \varphi_2}$: To establish the claim about number of states and colours we split cases. First we consider the case where $sd_k(\varphi) = 0$. In that case we also have $sd_k(\varphi_1) = sd_k(\varphi_2) = 0$. By induction hypothesis, for $i \in \{1, 2\}$, $\mathcal{A}^{\varphi_i}_s$ has at most $f^{\varphi_i}_\mathcal{S}$ states and 2 colours. These automata are then narrowed down, but the narrowing operation leaves the size of formulas in the transition function unchanged (in fact they may become smaller, but not bigger, see [49]). Therefore, by construction $\mathcal{A}^\varphi_s$ has at most $1 + f^{\varphi_1}_\mathcal{S} + f^{\varphi_2}_\mathcal{S}$ states and two colours.

Now we have that

$$1 + f^{\varphi_1}_\mathcal{S} + f^{\varphi_2}_\mathcal{S} = 1 + \sum_{i \in \{1,2\}} (4K_1 + 2K_2)^{\exists d(\varphi_i)}|\varphi_i||\mathcal{S}|^{\mathbf{Ed}(\varphi_i)}2^{K_\psi|\varphi_i|\mathbf{Ed}(\varphi_i)}$$

$$= 1 + (4K_1 + 2K_2)^{\exists d(\varphi)}|\varphi||\mathcal{S}|^{\mathbf{Ed}(\varphi)} \sum_{i \in \{1,2\}} 2^{K_\psi|\varphi_i|\mathbf{Ed}(\varphi)}$$

$$\leq 1 + (4K_1 + 2K_2)^{\exists d(\varphi)}|\varphi||\mathcal{S}|^{\mathbf{Ed}(\varphi)}2^{K_\psi(|\varphi_1|+|\varphi_2|)\mathbf{Ed}(\varphi)}$$

$$1 + f^{\varphi_1}_\mathcal{S} + f^{\varphi_2}_\mathcal{S} \leq (4K_1 + 2K_2)^{\exists d(\varphi)}|\varphi||\mathcal{S}|^{\mathbf{Ed}(\varphi)}2^{K_\psi(|\varphi_1|+|\varphi_2|+1)\mathbf{Ed}(\varphi)}$$

We get that

$$1 + f^{\varphi_1}_\mathcal{S} + f^{\varphi_2}_\mathcal{S} \leq f^\varphi_\mathcal{S} \tag{8}$$

which concludes the claim about the number of states.

Now for the case where $sd_k(\varphi) \geq 1$. By definition of nondeterminisation depth, for at least one $i \in \{1, 2\}$ we have $sd_k(\varphi_i) \geq 1$. Also, the number of colours used in $\mathcal{A}^\varphi_s$ is the maximum between

the number of colours used in $\mathcal{A}_s^{\varphi_1}$ and those used in $\mathcal{A}_s^{\varphi_2}$. By induction hypothesis it is the case that $\mathcal{A}_s^{\varphi_i}$ has at most $\exp(\mathrm{sd}_k(\varphi_i) - 1 \mid f_\mathcal{S}^{\varphi_i} \log f_\mathcal{S}^{\varphi_i})$ colours if $\mathrm{sd}_k(\varphi_i) \geq 1$, or 2 if $\mathrm{sd}_k(\varphi_i) = 0$. Therefore, the number of colours in $\mathcal{A}_s^{\varphi}$ is at most $\exp(\mathrm{sd}_k(\varphi_i) - 1 \mid f_\mathcal{S}^{\varphi_i} \log f_\mathcal{S}^{\varphi_i})$ for some i, which is less than $\exp(\mathrm{sd}_k(\varphi) - 1 \mid f_\mathcal{S}^{\varphi} \log f_\mathcal{S}^{\varphi})$.

For the number of states $|Q|$ in $\mathcal{A}_s^{\varphi}$, we have that $|Q| = 1 + |Q_1| + |Q_2|$, where Q_i is the set of states of $\mathcal{A}_s^{\varphi_i}$. By induction hypothesis we get

$$
\begin{aligned}
|Q| &\leq 1 + \sum_{i \in \{1,2\}} \exp(\mathrm{sd}_k(\varphi_i) \mid f_\mathcal{S}^{\varphi_i} \log f_\mathcal{S}^{\varphi_i}) \\
&\leq 1 + \exp(\mathrm{sd}_k(\varphi) \mid \sum_{i \in \{1,2\}} f_\mathcal{S}^{\varphi_i} \log f_\mathcal{S}^{\varphi_i}) \\
&\leq \exp(\mathrm{sd}_k(\varphi) \mid (\sum_{i \in \{1,2\}} f_\mathcal{S}^{\varphi_i} + 1) \log f_\mathcal{S}^{\varphi}) \\
|Q| &\leq \exp(\mathrm{sd}_k(\varphi) \mid f_\mathcal{S}^{\varphi} \log f_\mathcal{S}^{\varphi}) \qquad\qquad \text{(using Equation (8))}
\end{aligned}
$$

which concludes the claim about the number of states.

Concerning the size of formulas in the transition function, for all states from $\mathcal{A}_s^{\varphi_1}$ and $\mathcal{A}_s^{\varphi_2}$ the transition function is unchanged and the result thus holds by induction hypothesis. For the remaining state q_ι, we have by definition $\delta(q_\iota, a) = \delta^1(q_\iota^1, a) \vee \delta^2(q_\iota^2, a)$ and thus $|\delta(q_\iota, a)| = |\delta^1(q_\iota^1, a)| + |\delta^2(q_\iota^2, a)| + 1$. By induction hypothesis we get that

$$
\begin{aligned}
|\delta(q_\iota, a)| &\leq |\mathcal{S}||Q_1|^{|\mathcal{S}|} 2^{H(\varphi_1)|\varphi_1|} + |\mathcal{S}||Q_2|^{|\mathcal{S}|} 2^{H(\varphi_2)|\varphi_2|} + 1 \\
&\leq |\mathcal{S}| 2^{H(\varphi)(|\varphi_1| + |\varphi_2|)} (|Q_1|^{|\mathcal{S}|} + |Q_2|^{|\mathcal{S}|}) \\
&\leq |\mathcal{S}| 2^{H(\varphi)|\varphi|} (|Q_1| + |Q_2|)^{|\mathcal{S}|}
\end{aligned}
$$

And thus $|\delta(q_\iota, a)| \leq |\mathcal{S}| 2^{H(\varphi)|\varphi|} |Q|^{|\mathcal{S}|}$ as required.

$\varphi = \mathbf{E}\psi$: The word automaton built for the LTL skeleton of ψ is in fact a Büchi automaton, and thus uses only two colours. The number of colours used by $\mathcal{A}_s^{\varphi}$ is therefore the maximum number of colours used by the automata $\mathcal{A}_s^{\varphi_i}$ built for the maximal state subformulas φ_i in ψ, and the result follows by induction hypothesis.

Concerning the number of states, let $|Q_\varphi|$ (resp. $|Q_i|$, $|Q_\psi|$) be the number of states in $\mathcal{A}_s^{\varphi}$ (resp. $\mathcal{A}_s^{\varphi_i}$, $\mathcal{W}^{\psi}$). Note that the number of states in $\mathcal{A}_{s'}^{\varphi_i}$ does not depend on s'. Recall that $\max(\psi) = \{\varphi_1, \ldots, \varphi_n\}$ is the set of maximal state subformulas of ψ, and let ψ' be the LTL skeleton of ψ, i.e., the LTL formula obtained from ψ by replacing maximal state subformulas φ_i with propositions p_{φ_i}. We thus have

$$
\begin{aligned}
|Q| &= |Q_\psi||\mathcal{S}| + 2|\mathcal{S}| \sum_{i \in [n]} |Q_i| \\
&\leq 2^{K_\psi |\psi'|} |\mathcal{S}| + 2|\mathcal{S}| \sum_{i \in [n]} \exp(\mathrm{sd}_k(\varphi_i) \mid f_\mathcal{S}^{\varphi_i} \log f_\mathcal{S}^{\varphi_i}) \\
|Q| &\leq 2^{K_\psi |\psi'|} |\mathcal{S}| \left(1 + \exp(\mathrm{sd}_k(\varphi) \mid \sum_{i \in [n]} f_\mathcal{S}^{\varphi_i} \log f_\mathcal{S}^{\varphi_i}) \right)
\end{aligned}
$$

And thus

$$|Q| \leq 2^{K_\psi |\psi'|} |\mathcal{S}| \left(1 + \exp\left(\mathrm{sd}_k(\varphi) \mid \log f_\mathcal{S}^\varphi \sum_{i \in [n]} f_\mathcal{S}^{\varphi_i}\right) \right) \tag{9}$$

Now observe that for each $i \in [n]$ we have that $\mathrm{Ed}(\varphi_i) \leq \mathrm{Ed}(\varphi) - 1$, and $\exists \mathrm{d}(\varphi_i) = \exists \mathrm{d}(\varphi)$. Therefore,

$$\sum_{i \in [n]} f_\mathcal{S}^{\varphi_i} = (4K_1 + 2K_2)^{\exists \mathrm{d}(\varphi)} \sum_{i \in [n]} |\varphi_i| |\mathcal{S}|^{\mathrm{Ed}(\varphi_i)} 2^{K_\psi |\varphi_i| \mathrm{Ed}(\varphi_i)}$$

$$\leq (4K_1 + 2K_2)^{\exists \mathrm{d}(\varphi)} |\mathcal{S}|^{\mathrm{Ed}(\varphi)-1} \left(\sum_{i \in [n]} |\varphi_i| \right) 2^{K_\psi (\mathrm{Ed}(\varphi)-1) \sum_{i \in [n]} |\varphi_i|}$$

Using this in Equation ([9](#)) we get

$$|Q| \leq 2^{K_\psi |\psi'|} |\mathcal{S}| \left(1 + \exp\left(\mathrm{sd}_k(\varphi) \mid (4K_1 + 2K_2)^{\exists \mathrm{d}(\varphi)} |\mathcal{S}|^{\mathrm{Ed}(\varphi)-1} \left(\sum_{i \in [n]} |\varphi_i| \right) 2^{K_\psi (\mathrm{Ed}(\varphi)-1) \sum_{i \in [n]} |\varphi_i|} \log f_\mathcal{S}^\varphi \right) \right)$$

$$\leq 2^{K_\psi |\psi'|} \left(1 + \exp\left(\mathrm{sd}_k(\varphi) \mid (4K_1 + 2K_2)^{\exists \mathrm{d}(\varphi)} |\mathcal{S}|^{\mathrm{Ed}(\varphi)} \left(\sum_{i \in [n]} |\varphi_i| \right) 2^{K_\psi (\mathrm{Ed}(\varphi)-1) \sum_{i \in [n]} |\varphi_i|} \log f_\mathcal{S}^\varphi \right) \right)$$

$$\leq 2^{K_\psi |\psi'|} \exp\left(\mathrm{sd}_k(\varphi) \mid (4K_1 + 2K_2)^{\exists \mathrm{d}(\varphi)} |\mathcal{S}|^{\mathrm{Ed}(\varphi)} \left(1 + \sum_{i \in [n]} |\varphi_i| \right) 2^{K_\psi (\mathrm{Ed}(\varphi)-1) \sum_{i \in [n]} |\varphi_i|} \log f_\mathcal{S}^\varphi \right)$$

$$\leq \exp\left(\mathrm{sd}_k(\varphi) \mid (4K_1 + 2K_2)^{\exists \mathrm{d}(\varphi)} |\mathcal{S}|^{\mathrm{Ed}(\varphi)} |\varphi| 2^{K_\psi B} \log f_\mathcal{S}^\varphi \right),$$

where $B = (\mathrm{Ed}(\varphi) - 1) \sum_{i \in [n]} |\varphi_i| + |\psi'|$. To conclude it only remains to show that $B \leq |\varphi| \mathrm{Ed}(\varphi)$. Because $\varphi = \mathbf{E}\psi$, it holds that $\mathrm{Ed}(\varphi) \geq 1$. If $\mathrm{Ed}(\varphi) = 1$, we have $B = |\psi'| \leq |\varphi| \mathrm{Ed}(\varphi)$. Now if $\mathrm{Ed}(\varphi) \geq 2$, we have

$$B = (\mathrm{Ed}(\varphi) - 2) \sum_{i \in [n]} |\varphi_i| + |\psi'| + \sum_{i \in [n]} |\varphi_i|$$

Clearly, $\sum_{i \in [n]} |\varphi_i| \leq |\varphi|$, and $|\psi'| + \sum_{i \in [n]} |\varphi_i| \leq 2|\varphi|$, and the result follows. Note that it could seem that $|\psi'| + \sum_{i \in [n]} |\varphi_i| \leq |\varphi|$. It is true if one defines the size of a formula as the number of connectors, but not if one also counts atomic propositions, as we do here. However it is true that $|\psi'| + \sum_{i \in [n]} |\varphi_i| \leq 2|\varphi|$, independently of the definition of formulas' size.

It remains to establish the claim about the size of transition formulas. By definition, for every state q of $\mathcal{A}_s^\varphi$ that comes from some $\mathcal{A}_{s'}^i$ or $\overline{\mathcal{A}_{s'}^i}$, the transition function is unchanged and thus the result follows by induction hypothesis and the fact that narrowing and complementation do not increase the size of formulas in transition functions. Now for the remaining states, for each $(q^\psi, s') \in Q$ and every $a \in 2^{\mathrm{AP}_\exists(\Phi)}$, we have

$$|\delta((q^\psi, s'), a)| \leq \sum_{a' \in 2^{\max(\psi)}} \left(|\delta_\psi((q^\psi, s'), a')| + 1 + \sum_{\varphi_i \in a'} (|\delta_{s'}^i(q_{s'}^i, a)| + 1) + \sum_{\varphi_i \notin a'} (|\overline{\delta_{s'}^i}(\overline{q_{s'}^i}, a)| + 1) \right)$$

Now by induction hypothesis, and because complementation does not increase the size of formulas, we get:

$$|\delta((q^\psi, s'), a)| \leq \sum_{a' \in 2^{\max(\psi)}} \left(|\delta_\psi((q^\psi, s'), a')| + 2 \sum_{i \in [n]} |\mathcal{S}| 2^{H(\varphi_i)|\varphi_i|} |Q_i|^{|\mathcal{S}|} \right) + 2^{|\max(\psi)|} + 2|\max(\psi)| 2^{|\max(\psi)|}, \tag{10}$$

where $|Q_i|$ is the number of states in automaton $\mathcal{A}_{s'}^{\varphi_i}$. Now by definition,

$$|\delta_\psi((q^\psi, s'), a')| = \left(\sum_{q' \in \Delta^\psi(q^\psi, a')} \sum_{s'' \in R(s')} 1\right) + |\Delta^\psi(q^\psi, a')||R(s')| - 1$$

$$|\delta_\psi((q^\psi, s'), a')| \le 2|\Delta^\psi(q^\psi, a')||R(s')| - 1$$

We thus have

$$|\delta_\psi((q^\psi, s'), a')| \le 2|Q_{\psi'}||\mathcal{S}| - 1 \tag{11}$$

where $Q_{\psi'}$ is the set of states of the word automaton $\mathcal{W}^\psi$. Using this in Equation 10 we get:

$$|\delta((q^\psi, s'), a)| \le 2^{|\max(\psi)|}\left(2|Q_{\psi'}||\mathcal{S}| - 1 + 2\sum_{i \in [n]} |\mathcal{S}|2^{H(\varphi_i)|\varphi_i|}|Q_i|^{|\mathcal{S}|}\right) + 2^{|\max(\psi)|} + 2|\max(\psi)|2^{|\max(\psi)|}$$

$$|\delta((q^\psi, s'), a)| \le 2^{|\max(\psi)|+1}|\mathcal{S}|\left(|Q_{\psi'}| + \sum_{i \in [n]} 2^{H(\varphi_i)|\varphi_i|}|Q_i|^{|\mathcal{S}|}\right) + 2|\max(\psi)|2^{|\max(\psi)|}$$

But for natural numbers $\{a_i, b_i\}_{i \in [n]}$, it holds that

$$\sum_{i \in [n]} 2^{a_i} b_i = 2^{\sum_{i \in [n]} a_i} \sum_{i \in [n]} b_i - \sum_{i \in [n]} 2^{a_i}(2^{\sum_{j \ne i} a_j} - 1)b_i$$

Applying this to $a_i = H(\varphi_i)|\varphi_i|$ and $b_i = |Q_i|^{|\mathcal{S}|}$ we obtain

$$\sum_{i \in [n]} 2^{H(\varphi_i)|\varphi_i|}|Q_i|^{|\mathcal{S}|} = 2^{\sum_{i \in [n]} H(\varphi_i)|\varphi_i|} \sum_{i \in [n]} |Q_i|^{|\mathcal{S}|} - \sum_{i \in [n]} 2^{H(\varphi_i)|\varphi_i|}(2^{\sum_{j \ne i} H(\varphi_j)|\varphi_j|} - 1)|Q_i|^{|\mathcal{S}|}$$

We thus get that

$$|\delta((q^\psi, s'), a)| \le 2^{|\max(\psi)|+1}|\mathcal{S}|\left(|Q_{\psi'}| + 2^{\sum_{i \in [n]} H(\varphi_i)|\varphi_i|} \sum_{i \in [n]} |Q_i|^{|\mathcal{S}|}\right) + C,$$

with

$$C = 2|\max(\psi)|2^{|\max(\psi)|} - 2^{|\max(\psi)|+1}|\mathcal{S}| \sum_{i \in [n]} 2^{H(\varphi_i)|\varphi_i|}(2^{\sum_{j \ne i} H(\varphi_j)|\varphi_j|} - 1)|Q_i|^{|\mathcal{S}|}$$

$$= 2^{|\max(\psi)|}\left(2|\max(\psi)| - 2|\mathcal{S}| \sum_{i \in [n]} 2^{H(\varphi_i)|\varphi_i|}(2^{\sum_{j \ne i} H(\varphi_j)|\varphi_j|} - 1)|Q_i|^{|\mathcal{S}|}\right)$$

If $n = |\max(\psi)| > 1$, i.e., there are at least two maximal state subformulas, then $\sum_{j \ne i} H(\varphi_j)|\varphi_j| > 0$, hence $2|\mathcal{S}| \sum_{i \in [n]} 2^{H(\varphi_i)|\varphi_i|}(2^{\sum_{j \ne i} H(\varphi_j)|\varphi_j|} - 1)|Q_i|^{|\mathcal{S}|} \ge 4n = 4|\max(\psi)|$, which implies that $C \le 0$, and thus

$$|\delta((q^\psi, s'), a)| \le 2^{|\max(\psi)|+1}|\mathcal{S}|\left(|Q_{\psi'}| + 2^{\sum_{i \in [n]} H(\varphi_i)|\varphi_i|} \sum_{i \in [n]} |Q_i|^{|\mathcal{S}|}\right)$$

$$\le 2^{|\max(\psi)|+1}|\mathcal{S}|2^{\sum_{i \in [n]} H(\varphi_i)|\varphi_i|}\left(|Q_{\psi'}|^{|\mathcal{S}|} + \sum_{i \in [n]} |Q_i|^{|\mathcal{S}|}\right)$$

$$\leq |\mathcal{S}|2^{|\max(\psi)|+1+(H(\varphi)-1)\sum_{i\in[n]}|\varphi_i|}\left(|Q_{\psi'}| + \sum_{i\in[n]}|Q_i|\right)^{|\mathcal{S}|}$$

$$\leq |\mathcal{S}|2^{|\varphi|+(H(\varphi)-1)|\varphi|}|Q|^{|\mathcal{S}|}$$

$$|\delta((q^\psi, s'), a)| \leq |\mathcal{S}|2^{H(\varphi)|\varphi|}|Q|^{|\mathcal{S}|}$$

It remains to consider the case where $\max(\psi) = \{\varphi_1\}$. In that case there are only two letters in the alphabet $2^{\max(\psi)}$, which are $\emptyset$ and $\{\varphi_1\}$. The transition formulas then simplify and one gets that

$$|\delta((q^\psi, s'), a)| \leq |\delta_\psi((q^\psi, s'), \emptyset)| + 1 + |\overline{\delta_{s'}^1(\overline{q_{s'}^1}, a)}| + 1 + |\delta_\psi((q^\psi, s'), \{\varphi_1\})| + 1 + |\delta_{s'}^1(q_{s'}^1, a)|$$

Using Equation (11) and the induction hypothesis we get

$$|\delta((q^\psi, s'), a)| \leq 4|Q_{\psi'}||\mathcal{S}| - 2 + 2|\mathcal{S}|2^{H(\varphi_1)|\varphi_1|}|Q_1|^{|\mathcal{S}|} + 3$$

$$\leq 1 + 2|\mathcal{S}|(2|Q_{\psi'}| + 2^{H(\varphi_1)|\varphi_1|}|Q_1|^{|\mathcal{S}|})$$

$$\leq 1 + 2|\mathcal{S}|2^{(H(\varphi)-1)|\varphi_1|}(|Q_{\psi'}|^{|\mathcal{S}|} + |Q_1|^{|\mathcal{S}|})$$

$$\leq 1 + |\mathcal{S}|2^{H(\varphi)|\varphi|}(|Q_{\psi'}|^{|\mathcal{S}|} + |Q_1|^{|\mathcal{S}|})$$

$$|\delta((q^\psi, s'), a)| \leq |\mathcal{S}|2^{H(\varphi)|\varphi|}|Q|^{|\mathcal{S}|}$$

$\boldsymbol{\varphi = \exists^o p. \varphi'}$: We first establish the claim for states and colours, and we start with the case $sd_k(\varphi) = sd_k(\varphi')$. By definition we necessarily have that $sd_x(\varphi') = \mathsf{nd}$, i.e., $\mathcal{A}_s^{\varphi'}$ is nondeterministic, and $\mathbf{o} = I_{\varphi'}$, therefore there is no need to use narrowing or nondeterminisation here. $\mathcal{A}_s^\varphi$ is obtained by directly projecting $\mathcal{A}_s^{\varphi'}$, an operation that does not change the number of states or colours, so that the claim for states and colours follows directly by induction hypothesis.

Now we consider the case where $sd_k(\varphi) \neq sd_k(\varphi')$, which implies that $sd_k(\varphi) \geq 1$. Let n be the number of states and l the number of colours in $\mathcal{A}_s^{\varphi'}$. In this case $\mathcal{A}_s^{\varphi'}$ is first narrowed down, which does not change number of states or colours. The resulting automaton is then nondeterminised, yielding an automaton with at most $2^{K_1 nl \log nl}$ states and $K_2 nl$ colours.

Again, we split cases: if $sd_k(\varphi') = 0$, by induction hypothesis, $n \leq f_\mathcal{S}^{\varphi'}$ and $l = 2$. For the number of colours, observing that $\exists d(\varphi) = \exists d(\varphi') + 1$, we have

$$K_2 nl \leq 2K_2 f_\mathcal{S}^{\varphi'} = 2K_2(4K_1 + 2K_2)^{\exists d(\varphi')}|\varphi'||\mathcal{S}|^{\mathsf{Ed}(\varphi')}2^{K_\psi|\varphi'|\mathsf{Ed}(\varphi')}$$

$$\leq (4K_1 + 2K_2)^{\exists d(\varphi)}|\varphi||\mathcal{S}|^{\mathsf{Ed}(\varphi)}2^{K_\psi|\varphi|\mathsf{Ed}(\varphi)}$$

$$K_2 nl \leq \exp(sd_k(\varphi) - 1 \mid f_\mathcal{S}^\varphi \log f_\mathcal{S}^\varphi)$$

For the number of states, we have that

$$2^{K_1 nl \log nl} \leq 2^{2K_1 f_\mathcal{S}^{\varphi'} \log(2f_\mathcal{S}^{\varphi'})} \leq \exp(sd_k\varphi \mid f_\mathcal{S}^\varphi \log(f_\mathcal{S}^\varphi))$$

Now for the final case, if $sd_k(\varphi) = sd_k(\varphi') + 1$ and $sd_k(\varphi') \geq 1$, by induction hypothesis $n \leq \exp(sd_k(\varphi') \mid f_\mathcal{S}^{\varphi'} \log f_\mathcal{S}^{\varphi'})$ and $l \leq \exp(sd_k(\varphi') - 1 \mid f_\mathcal{S}^{\varphi'} \log f_\mathcal{S}^{\varphi'})$. For the number of colours in $\mathcal{A}_s^\varphi$ we thus get

$$K_2 nl \leq K_2 \exp(sd_k(\varphi') - 1 \mid f_\mathcal{S}^{\varphi'} \log f_\mathcal{S}^{\varphi'} 2^{f_\mathcal{S}^{\varphi'} \log f_\mathcal{S}^{\varphi'}})$$

$$\leq \exp(sd_k(\varphi') \mid 2K_2 f_\mathcal{S}^{\varphi'} \log f_\mathcal{S}^{\varphi'})$$

$$K_2 nl \leq \exp(sd_k(\varphi) - 1 \mid f_\mathcal{S}^\varphi \log f_\mathcal{S}^\varphi)$$

Concerning the number of states, we observe that

$$nl \leq \exp\left(\mathrm{sd}_k(\varphi') - 1 \mid f_S^{\varphi'} \log f_S^{\varphi'} 2^{f_S^{\varphi'} \log f_S^{\varphi'}}\right)$$

$$nl \leq \exp\left(\mathrm{sd}_k(\varphi') \mid 2 f_S^{\varphi'} \log f_S^{\varphi'}\right)$$

$$K_1 nl \log nl \leq \exp\left(\mathrm{sd}_k(\varphi') - 1 \mid 2 K_1 f_S^{\varphi'} \log f_S^{\varphi'} 2^{2 f_S^{\varphi'} \log f_S^{\varphi'}}\right)$$

$$K_1 nl \log nl \leq \exp\left(\mathrm{sd}_k(\varphi') \mid 4 K_1 f_S^{\varphi'} \log f_S^{\varphi'}\right)$$

$$K_1 nl \log nl \leq \exp\left(\mathrm{sd}_k(\varphi') \mid f_S^{\varphi} \log f_S^{\varphi}\right)$$

$$2^{K_1 nl \log nl} \leq \exp\left(\mathrm{sd}_k(\varphi) \mid f_S^{\varphi} \log f_S^{\varphi}\right)$$

It only remains to establish the claim for the size of transition formulas. Since $\mathcal{A}_s^{\varphi}$ is nondeterministic, formulas $\delta(q, a)$ are written in disjunctive normal form and for every direction $x \in S_\varphi$ each disjunct contains exactly one element of $\{x\} \times Q$, where Q is the set of states in $\mathcal{A}_s^{\varphi}$. As a result, each formula $\delta(q, a)$ is of size

$$|\delta(q, a)| \leq |Q|^{|S_\varphi|}(2|S_\varphi| - 1) + |Q|^{|S_\varphi|} - 1$$

$$\leq 2|S_\varphi||Q|^{|S_\varphi|}$$

$$|\delta(q, a)| \leq 2^{H(\varphi)|\varphi|}|\mathcal{S}||Q|^{|\mathcal{S}|}$$

$\square$