Quantified CTL with imperfect information*

Raphaël Berthon[1], Bastien Maubert[2], and Aniello Murano[3]

1 École Normale Supérieure de Rennes, France
raphael.berthon@ens-rennes.fr
2 Università degli Studi di Napoli Federico II, Italy
bastien.maubert@gmail.com
3 Università degli Studi di Napoli Federico II, Italy
murano@na.infn.it

Abstract

Quantified CTL (QCTL) is a well-studied temporal logic that extends CTL with quantification over atomic propositions. It has recently come to the fore as a powerful intermediary framework to study logics for strategic reasoning. We extend it to include imperfect information by parameterising quantifiers with an observation that defines how well they observe the model, thus constraining their behaviour. We consider two different semantics, one related to the notion of *no memory*, the other to *perfect recall*. We study the expressiveness of our logic, and show that it coincides with MSO for the first semantics and with MSO with equal level for the second one. We establish that the model-checking problem is PSPACE-complete for the first semantics. While it is undecidable for the second one, we identify a syntactic fragment, defined by a notion of hierarchical formula, which we prove to be decidable thanks to an automata-theoretic approach.

1 Introduction

Temporal logic is a powerful framework widely used in formal system-design and verification [9, 41]. It allows reasoning over the temporal evolution of a system, without referring explicitly to the elapsing of time. One of the most significant contributions of the field is *model checking*, which allows to verify system correctness by checking whether a mathematical model of the system satisfies a temporal logic formula expressing its desired behaviour [8, 9, 26, 27].

Depending on the view of the nature of time, two types of temporal logics are mainly considered. In *linear-time temporal logics* such as LTL [41] time is treated as if each moment in time had a unique possible future. Conversely, in *branching-time temporal logics* such as CTL [8] and CTL* [16], each moment in time may split into various possible futures; existential and universal quantifiers then allow expressing properties of either one or all the possible futures. While LTL is suitable to express path properties, CTL is more appropriate for state-based ones, and CTL* for both. These logics are "easy-to-use", can express important system properties such as *liveness* or *safety*, enjoy good fundamental theoretical properties such as invariance under tree-unwinding of models, and come with reasonable complexities for the main related decision problems. For instance, the model-checking and satisfiability problems for CTL* are PSPACE-Complete [1] and 2-EXPTIME-Complete [48], respectively.

Along the years, CTL* has been extended in a number of ways in order to verify the behavior of a broad variety of systems. In multi-agent open-system verification, *Alternating-Time Temporal Logic* (ATL*), introduced by Alur, Henzinger, and Kupferman [3], is particularly successful. This generalization of CTL* replaces path quantifiers with *strategic modalities*, that is modalities over teams of agents that describe the ability to cooperate in order to

* This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 709188.

achieve a goal against adversaries. ATL^* model checking is a very active area of research and it has been studied in several domains, including communication protocols [47], fair exchange protocols [21, 19], and agent-oriented programs [11]. The complexity of the problem has been extensively studied in a multitude of papers, and algorithms have been implemented in tools [32]. Remarkably, ATL^* has inspired fresh and more powerful logics such as *Strategy Logic* (SL) [7, 35, 34], ATL^* *with strategy context* (ATL^*_{sc}) [5, 10], ATL^* *with Irrevocable strategies* (IATL^*) [2] and *Memoryful* ATL^* ($\mathrm{m}\mathsf{ATL}^*$) [36]. These logics are progressively overtaking ATL^*; in particular this is the case for SL as it can can express fundamental game-theoretic concepts such as Nash Equilibrium and Subgame Perfect Equilibrium [34].

In the landscape of temporal logics, another breakthrough contribution comes from *Quantified* CTL^* (QCTL^*), which extends CTL^* with the possibility to quantify over atomic propositions [45, 15, 23, 24, 17, 30]. QCTL^* turns out to be very expressive (indeed, it is equivalent to Monadic Second-Order Logic, MSO for short) and was usefully applied in a number of scenarios. Recently it has come to the fore as a convenient and uniform intermediary logic to easily obtain algorithms for ATL^*_{sc}, SL, as well as related formalisms [30, 33, 34]. Indeed, strategies can be represented by atomic propositions labelling the execution tree of the game structure under study, and strategy quantification can thus be expressed by means of propositional quantifications. As a remark, quantification in QCTL^* can be interpreted either on Kripke structures (*structure semantics*) or their execution tree (*tree semantics*), allowing for the encoding of memoryless or perfect-recall strategies, respectively. This difference impacts also the complexity of the related decision problems: for instance, moving from structure to tree semantics, model checking jumps from PSPACE to non-elementary.

In game theory and open-system verification an important body of work has been devoted to *imperfect information*, which refers to settings in which players have partial information about the moves taken by the others [6, 13, 20, 25, 42]. This is a common situation in real-life scenarios where players have to act without having all the relevant information at hand. In computer science this situation occurs for example when some system's variables are internal/private [44]. Imperfect information is usually modelled by indistinguishability relations over the states of the game. During a play, some players may not know precisely in which state they are, and therefore they cannot base their actions on the exact current situation: they must choose their actions uniformly over indistinguishable states [25].

This uniformity constraint deeply impacts the complexity of decision problems. It is well known that multi-player games with imperfect information are computationally hard, in general undecidable [38], and to retain positive complexity results one needs to restrict players' capabilities, by bounding their memory of past moves [12] or putting some hierarchical order over their observational power [42]. Unfortunately, most of the approaches exploited under full observation are not appropriate for imperfect information. In particular this is the case of QCTL^*, unless opportunely adapted. In this paper we work in this direction by incorporating in QCTL^* the essence of imperfect information, that is the uniformity constraint on choices. We believe it may provide a uniform framework to obtain new results on logics for strategic reasoning under imperfect information, as does QCTL^* in the perfect information setting.

Our contribution. We introduce $\mathsf{QCTL}^*_{\mathrm{i}}$, an opportune extension of QCTL^* that integrates the central feature of imperfect information, *i.e.*, uniformity constraint on choices. We add internal structure to the states of the models, much like in Reif's multiplayer game structures [38] or distributed systems [18], and we parameterise propositional quantifiers with observations that define what portions of the states a quantifier can "observe". The semantics is adapted to capture the idea of quantifications on atomic propositions being made with partial observation. Like in [30], we consider both structure and tree semantics.

We study the expressive power of $\mathsf{QCTL}^*_{\mathsf{i}}$. By using the same argument as for QCTL^* [30], we first show that $\mathsf{QCTL}^*_{\mathsf{i}}$ and $\mathsf{QCTL}_{\mathsf{i}}$ are equally expressive for both semantics. Then we prove that for the structure semantics, these logics are no more expressive than QCTL, and thus coincide with MSO. Finally we show that under tree semantics $\mathsf{QCTL}_{\mathsf{i}}$ is expressively equivalent to MSO extended with the equal level predicate ($\mathsf{MSO}_{\mathrm{eq}}$, see [14, 31, 46]).

Concerning the model-checking problem we first prove that under structure semantics it is PSPACE-complete for both $\mathsf{QCTL}^*_{\mathsf{i}}$ and $\mathsf{QCTL}_{\mathsf{i}}$, like QCTL. Under tree semantics, undecidability follows from the equivalence with $\mathsf{MSO}_{\mathrm{eq}}$. However we identify a decidable syntactic fragment, consisting of those formulas in which nested quantifiers have hierarchically ordered observations, innermost ones observing more than outermost ones. We call such formulas *hierarchical formulas*. Interestingly, a decidability result for Quantified μ-Calculus with partial observation [40] uses a similar syntactic restriction. This logic is very close to ours, but orthogonal: while our tree semantics relies on a synchronous perfect-recall notion of imperfect information, theirs is asynchronous. This hierarchical restriction is also related to decidability results for games with imperfect information [39, 4] and distributed synthesis [22]. Our decision procedure relies on automata constructions involving the *narrowing* operation introduced by Kupferman and Vardi in [28] for distributed synthesis. We believe that our choice of modelling imperfect information by means of local states eases greatly the use of automata techniques to tackle imperfect information. Finally, our result provides new decidability results for ATL^*_{sc} with imperfect information (not presented here), and we trust it will find applications in other logics, such as SL with imperfect information.

Plan. In Section 2 we recall Kripke structures and trees, and the syntax and semantics of QCTL^*. We then present $\mathsf{QCTL}^*_{\mathsf{i}}$ in Section 3, we study its expressiveness in Section 4 and its model-checking problem in Section 5. We conclude and discuss future work in Section 6.

2 Preliminaries

Let Σ be an alphabet. A *finite* (resp. *infinite*) *word* over Σ is an element of Σ^* (resp. Σ^ω). The empty word is classically noted ϵ, and $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$. The *length* of a word is $|w| := 0$ if w is the empty word ϵ, if $w = w_0 w_1 \dots w_n$ is a finite non-empty word then $|w| := n+1$, and for an infinite word w we let $|w| := \omega$. Given a word w and $0 \leq i, j \leq |w|-1$, we let w_i be the letter at position i in w and $w[i,j]$ be the subword of w that starts at position i and ends at position j. If w is infinite, we let $w^i := w[i,\omega]$. We write $w \preccurlyeq w'$ if w is a prefix of w', and $w^{\preccurlyeq}$ is the set of finite prefixes of word w. Finally, for $n \in \mathbb{N}$ we let $[n] := \{1, \dots, n\}$.

2.1 Kripke structures and trees

Let $\mathcal{AP}$ be a countably infinite set of *atomic propositions* and let $AP \subset \mathcal{AP}$ be a finite subset.

▶ **Definition 1.** A *Kripke structure* over AP is a tuple $\mathcal{S} = (S, R, \ell)$ where S is a set of *states*, $R \subseteq S \times S$ is a left-total[1] *transition relation* and $\ell : S \to 2^{AP}$ is a *labelling function*.

A *pointed Kripke structure* is a pair $(\mathcal{S}, s)$ where $s \in \mathcal{S}$, and the *size* $|\mathcal{S}|$ of a Kripke structure $\mathcal{S}$ is its number of states. A *path* in a structure $\mathcal{S} = (S, R, \ell)$ is an infinite word $\lambda \in S^\omega$ such that for all $i \in \mathbb{N}$, $(\lambda_i, \lambda_{i+1}) \in R$. For $s \in S$, we let $\mathrm{Paths}(s)$ be the set of all paths that start in s. A *finite path* is a finite non-empty prefix of a path.

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

We now define (infinite) trees. In many works, trees are defined as prefixed-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 as a sequence of states and the root as the initial state, hence the following definition, where X is a finite set:

▶ **Definition 2.** 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;
- if $u \cdot x \in \tau$ and $u \neq \epsilon$, then $u \in \tau$, and
- if $u \in \tau$ then there exists $x \in X$ such that $u \cdot x \in \tau$.

The elements of a tree τ are called *nodes*. If $u \cdot x \in \tau$, we say that $u \cdot x$ is a *child* of u. An X-tree is *full* if every node u has a child $u \cdot x$ for each $x \in X$. The *depth* of a node u is $|u|$. Similarly to Kripke structures, a *path* is an infinite sequence of nodes $\lambda = u_0 u_1 \dots$ such that for all $i \in \mathbb{N}$, u_{i+1} is a child of u_i, and $Paths(u)$ is the set of paths that start in node u. An AP*-labelled* X*-tree*, or (AP, X)*-tree* for short, is a pair $t = (\tau, \ell)$, where τ is an X-tree called the *domain* of t and $\ell : \tau \to 2^{AP}$ is a *labelling*. For a labelled tree $t = (\tau, \ell)$ and an atomic proposition $p \in \mathcal{AP}$, we define the *p-projection of t* as the labelled tree $t{\Downarrow_p} := (\tau, \ell{\Downarrow_p})$, where for each $u \in \tau$, $\ell{\Downarrow_p}(u) := \ell(u) \setminus \{p\}$. For a set of trees $\mathcal{L}$, we let $\mathcal{L}{\Downarrow_p} := \{t{\Downarrow_p} \mid t \in \mathcal{L}\}$.

▶ **Definition 3** (Tree unfoldings)**.** Let $\mathcal{S} = (S, R, \ell)$ be a Kripke structure over AP, and let $s \in S$. The *tree-unfolding of $\mathcal{S}$ from s* is the (AP, S)-tree $t_{\mathcal{S},s} = (\tau, \ell')$, where τ is the set of all finite paths that start in s, and for every $u \in \tau$, $\ell'(u) = \ell(\text{last}(u))$.

2.2 QCTL*, syntax and semantics

We recall the syntax of QCTL*, as well as both the structure and tree semantics.

▶ **Definition 4.** The syntax of QCTL* is defined by the following grammar:

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

where $p \in \mathcal{AP}$. Formulas of type φ are called *state formulas*, those of type ψ are called *path formulas*, and QCTL* consists of state formulas.

Like in [30] we consider two different semantics, the *structure semantics* and the *tree semantics*: in the former formulas are evaluated directly on the structure, while in the latter the structure is first unfolded into an infinite tree. In the first case, quantifying over p means choosing a truth value for p in each state of the structure, while in the second case it is possible to choose a different truth value for p in each finite path of the structure.

2.2.1 Structure semantics

A QCTL* state (resp. path) formula is evaluated in a state (resp. path) of a Kripke structure. To define the semantics of quantifications over propositions, the following definition is handy.

▶ **Definition 5.** For $p \in \mathcal{AP}$, two structures $\mathcal{S} = (S, R, \ell)$ and $\mathcal{S}' = (S', R', \ell')$ are *equivalent modulo p*, written $\mathcal{S} \equiv_p \mathcal{S}'$, if $S = S'$, $R = R'$ and for each state $s \in S$, $\ell(s) \setminus \{p\} = \ell'(s) \setminus \{p\}$. This definition also applies to labelled trees seen as infinite Kripke structures.

The satisfaction relation $\models_s$ for the structure semantics is defined inductively as follows, where $\mathcal{S} = (S, R, \ell)$ is a Kripke structure, s is a state and λ is a path in $\mathcal{S}$:

$$
\begin{array}{lll}
\mathcal{S}, s \models_s p & \text{if} & p \in \ell(s) \qquad\qquad\qquad \mathcal{S}, s \models_s \neg\varphi \quad \text{if} \quad \mathcal{S}, s \not\models_s \varphi \\
\mathcal{S}, s \models_s \varphi \vee \varphi' & \text{if} & \mathcal{S}, s \models_s \varphi \text{ or } \mathcal{S}, s \models_s \varphi' \\
\mathcal{S}, s \models_s \mathbf{E}\psi & \text{if} & \text{there exists } \lambda \in \text{Paths}(s) \text{ such that } \mathcal{S}, \lambda \models_s \psi \\
\mathcal{S}, s \models_s \exists p.\, \varphi & \text{if} & \text{there exists } \mathcal{S}' \equiv_p \mathcal{S} \text{ such that } \mathcal{S}', s \models \varphi \\
\mathcal{S}, \lambda \models_s \varphi & \text{if} & \mathcal{S}, \lambda_0 \models_s \varphi \qquad\qquad\qquad \mathcal{S}, \lambda \models_s \neg\psi \quad \text{if} \quad \mathcal{S}, \lambda \not\models_s \psi \\
\mathcal{S}, \lambda \models_s \psi \vee \psi' & \text{if} & \mathcal{S}, \lambda \models_s \psi \text{ or } \mathcal{S}, \lambda \models_s \psi' \qquad \mathcal{S}, \lambda \models_s \mathbf{X}\psi \quad \text{if} \quad \mathcal{S}, \lambda^1 \models_s \psi \\
\mathcal{S}, \lambda \models_s \psi \mathbf{U} \psi' & \text{if} & \text{there exists } i \geq 0 \text{ such that } \mathcal{S}, \lambda^i \models_s \psi' \text{ and for } 0 \leq j < i,\ \mathcal{S}, \lambda^j \models_s \psi
\end{array}
$$

2.2.2 Tree semantics

In the tree semantics, a formula holds in a state s of a structure $\mathcal{S}$ if it holds in the tree-unfolding of $\mathcal{S}$ from s. The semantics of QCTL^* on trees could be derived from the structure semantics, seeing 2^{AP}-labelled trees as infinite-state Kripke structures. We define it explicitly on trees though, as it will make the presentation of the semantics for QCTL_i clearer.

The satisfaction relation $\models_t$ for the tree semantics is thus defined inductively as follows, where $t = (\tau, \ell)$ is a 2^{AP}-labelled X-tree, u is a node and λ is a path in τ:

$$
\begin{array}{lll}
t, u \models_t p & \text{if} & p \in \ell(u) \qquad\qquad\qquad t, u \models_t \neg\varphi \quad \text{if} \quad t, u \not\models_t \varphi \\
t, u \models_t \varphi \vee \varphi' & \text{if} & t, u \models_t \varphi \text{ or } t, u \models_t \varphi' \\
t, u \models_t \mathbf{E}\psi & \text{if} & \text{there exists } \lambda \in \mathit{Paths}(u) \text{ such that } t, \lambda \models_t \psi \\
t, u \models_t \exists p.\, \varphi & \text{if} & \text{there exists } t' \equiv_p t \text{ such that } t', u \models \varphi \\
t, \lambda \models_t \varphi & \text{if} & t, \lambda_0 \models_t \varphi \qquad\qquad\qquad t, \lambda \models_t \neg\psi \quad \text{if} \quad t, \lambda \not\models_t \psi \\
t, \lambda \models_t \psi \vee \psi' & \text{if} & t, \lambda \models_t \psi \text{ or } t, \lambda \models_t \psi' \qquad t, \lambda \models_t \mathbf{X}\psi \quad \text{if} \quad t, \lambda^1 \models_t \psi \\
t, \lambda \models_t \psi \mathbf{U} \psi' & \text{if} & \text{there exists } i \geq 0 \text{ such that } t, \lambda^i \models_t \psi' \text{ and for } 0 \leq j < i,\ t, \lambda^j \models_t \psi
\end{array}
$$

We may write $t \models_t \varphi$ for $t, r \models_t \varphi$, where r is the root of t, and given a Kripke structure $\mathcal{S}$, a state s and a QCTL^* formula φ, we write $\mathcal{S}, s \models_t \varphi$ if $t_{\mathcal{S},s} \models_t \varphi$.

3 QCTL^* with imperfect information

We now enrich the models, syntax and semantics to capture the idea of quantifications on atomic propositions being made with a partial observation of the system.

3.1 Compound Kripke structures

First, we enrich Kripke structures by adding internal structure to states: we set them as tuples of local states. To ease presentation and obtain finite alphabets for our tree automata in Section 5.2.2, we fix a collection $\{L_i\}_{i\in[n]}$ of n disjoint finite sets of *local states*.

For $I \subseteq [n]$, we let $X_I := \bigtimes_{i\in I} L_i$. Let $J \subseteq I \subseteq [n]$. For $x = (l_i)_{i\in I} \in X_I$, we define the X_J*-projection* of x as $x{\downarrow_{X_J}} := (l_i)_{i\in J}$. If $J = \emptyset$, we let $x{\downarrow_\emptyset} := \mathbf{0}$, where $\mathbf{0}$ is a special symbol, and we let $X_\emptyset := \{\mathbf{0}\}$. This definition extends naturally to words and trees over X_I. Observe that when projecting a tree, nodes with same projection are merged. In particular, for every X_I-tree τ, $\tau{\downarrow_\emptyset}$ is the only $X_\emptyset$-tree, $\mathbf{0}^\omega$. We also define a *lift* operator $\uparrow_y^I$ that, given an X_J-tree rooted in x and a tuple $y \in X_{I\setminus J}$, produces the X_I-tree rooted in (x, y) defined as $\tau{\uparrow_y^{X_I}} := \{u \in (x, y) \cdot X_I^* \mid u{\downarrow_{X_J}} \in \tau\}$. Observe that because the sets $\{L_i\}_{i\in[n]}$ are disjoint, the ordering of elements in tuples of X_I does not matter. For an (AP, X_J)-tree $t = (\tau, \ell)$, we define $t{\uparrow_y^{X_I}} := (\tau{\uparrow_y^{X_I}}, \ell')$ where $\ell'(u) := \ell(u{\downarrow_{X_J}})$. In the following we may write $\uparrow_y^I$ for $\uparrow_y^{X_I}$, and $\downarrow_J$ instead of $\downarrow_{X_J}$.

▶ **Definition 6.** A *compound Kripke structure*, or CKS, is a Kripke structure $\mathcal{S} = (S, R, \ell)$ such that $S \subseteq X_{[n]}$. We call n the *dimension* of CKSs.

▶ Remark. Note that by fixing finite sets of local states, we also fix a finite set of possible states. If it were not so, our translation from $\mathsf{QCTL_i}$ to QCTL in Theorem 13, as well as the one from $\mathsf{QCTL_i}$ to $\mathsf{MSO_{eq}}$ in Theorem 15 would no longer be valid, making us also lose Corollary 14. We would no longer have equivalence in expressivity, but we would still have that $\mathsf{QCTL_i}$ is at least as expressive as MSO (resp. $\mathsf{MSO_{eq}}$) for structure semantics (resp. tree semantics). Also our results on model checking in Section 5, and in particular our main result, Theorem 23, would still be valid.

To model the fact that quantifiers may not observe some local states, we define a notion of observation and the associated notion of observational indistinguishability.

▶ **Definition 7.** An *observation* is a finite set of indices $o \subset \mathbb{N}$. For an observation o and $I \subseteq [n]$, two tuples $x, x' \in X_I$ are *o-indistinguishable*, written $x \sim_o x'$, if $x{\downarrow_{I\cap o}} = x'{\downarrow_{I\cap o}}$.

Intuitively, a quantifier with observation o must choose the valuation of atomic propositions *uniformly* with respect to o, and this notion of uniformity will vary between the structure semantics and the tree semantics. But first, let us introduce the syntax of $\mathsf{QCTL}^*_{\mathsf{i}}$.

3.2 $\mathsf{QCTL}^*_{\mathsf{i}}$, syntax and semantics

The syntax of $\mathsf{QCTL}^*_{\mathsf{i}}$ is that of QCTL^*, except that quantifiers over atomic propositions are parameterised by a set of indices that defines what local states the quantifier can "observe".

▶ **Definition 8.** The syntax of $\mathsf{QCTL}^*_{\mathsf{i}}$ is defined by the following grammar:

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

where $p \in \mathcal{AP}$ and $o \subset \mathbb{N}$ is an observation.

We use standard abbreviations: $\top := p \vee \neg p$, $\bot := \neg\top$, $\mathbf{F}\psi := \top\mathbf{U}\psi$, $\mathbf{G}\psi := \neg\mathbf{F}\neg\psi$ and $\mathbf{A}\psi := \neg\mathbf{E}\neg\psi$. The size $|\varphi|$ of a formula φ is defined inductively as usual, but the following case: $|\exists^o p.\ \varphi| := 1 + |o| + |\varphi|$. We also classically define the syntactic fragment $\mathsf{QCTL_i}$:

▶ **Definition 9.** The syntax of $\mathsf{QCTL_i}$ is defined by the following grammar:

$$\varphi := p \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists^o p.\ \psi \mid \mathbf{EX}\varphi \mid \mathbf{AX}\varphi \mid \mathbf{E}\varphi\mathbf{U}\varphi \mid \mathbf{A}\varphi\mathbf{U}\varphi$$

where $p \in \mathcal{AP}$ and $o \subset \mathbb{N}$ is an observation.

3.2.1 Structure semantics

In the case of structure semantics, uniformity is defined as follows:

▶ **Definition 10.** Let $\mathcal{S} = (S, R, \ell)$ be a CKS, $p \in \mathcal{AP}$ and $o \subset \mathbb{N}$. $\mathcal{S}$ is *o-uniform in p* if for every pair of states $s, s' \in S$ such that $s \sim_o s'$, it holds that $p \in \ell(s)$ if and only if $p \in \ell(s')$.

We enrich the satisfaction relation $\models_s$ with the following inductive case, where $(\mathcal{S}, s)$ is a pointed CKS:

$$\mathcal{S}, s \models_s \exists^o p.\ \varphi \quad \text{if} \quad \text{there exists } \mathcal{S}' \equiv_p \mathcal{S} \text{ such that } \mathcal{S}' \text{ is } o\text{-uniform in } p \text{ and } \mathcal{S}', s \models_s \varphi$$

Observe that $\exists^{\{1,\ldots,n\}} p.\ \varphi$ is equivalent to the QCTL^* formula $\exists p.\ \varphi$.

3.2.2 Tree semantics

As observed in the introduction, propositional quantifiers can be seen as having perfect recall in the tree semantics and no memory in the structure semantics. The following definition for indistinguishability on trees, which differs from that for CKS, reflects this difference.

▶ **Definition 11.** Let $t = (\tau, \ell)$ be a labelled X_I-tree, $p \in \mathcal{AP}$ an atomic proposition and $o \subset \mathbb{N}$ an observation. Two nodes $u = u_0 \dots u_i$ and $u' = u'_0 \dots u'_j$ of τ are *o-indistinguishable*, written $u \approx_o u'$, if $i = j$ and for all $k \in \{0, \dots, i\}$ we have $u_k \sim_o u'_k$. Tree t is *o-uniform in p* if for every pair of nodes $u, u' \in \tau$ such that $u \approx_o u'$, we have $p \in \ell(u)$ iff $p \in \ell(u')$.

The tree semantics for $\mathsf{QCTL}^*_\mathrm{i}$ is defined on labelled X_n-trees, and it is obtained by enriching $\models_t$ as follows:

$$t, u \models_t \exists^o p.\, \varphi \quad \text{if} \quad \text{there exists } t' \equiv_p t \text{ such that } t' \text{ is } o\text{-uniform in } p \text{ and } t', u \models_t \varphi.$$

Consider the following CTL formula: $\mathbf{border}(p) := \mathbf{AF}p \wedge \mathbf{AG}(p \to \mathbf{AXAG}\neg p)$.

This formula holds in a labelled tree if and only if each path contains exactly one node labelled with p. Therefore, evaluating the QCTL_i formula $\exists^\emptyset p.\, \mathbf{border}(p)$ amounts to choosing a level of the tree where to place one horizontal line of p's.

4 Expressiveness

In this section we study the expressive power of our logics. We first observe that for both semantics, $\mathsf{QCTL}^*_\mathrm{i}$ and QCTL_i are equally expressive. We then prove that with structure semantics QCTL_i is expressively equivalent to QCTL, and thus also to MSO. Finally we show that under tree semantics $\mathsf{QCTL}^*_\mathrm{i}$ is expressively equivalent to MSO with equal level predicate. Note that Theorem 13, Corollary 14 and Theorem 15 below only hold if the logics can talk about the local states. For this reason, in this section we assume a set of dedicated atomic propositions $AP_l = \bigcup_{i \in [n]} \bigcup_{l \in L_i} \{p_l\} \subset \mathcal{AP}$ such that for every CKS $\mathcal{S} = (S, R, \ell)$, for each $i \in [n]$ and $l \in L_i$, for each state $s = (l_1, \dots, l_n) \in S$, we have $p_l \in \ell(s)$ iff $l_i = l$.

4.1 $\mathsf{QCTL}^*_\mathrm{i}$, QCTL_i and QCTL

We first remark that for the same reason why QCTL^* is no more expressive than QCTL, also $\mathsf{QCTL}^*_\mathrm{i}$ and QCTL_i are equally expressive (the proof of [30, Proposition 3.8] readily applies):

▶ **Theorem 12.** *Under both semantics,* $\mathsf{QCTL}^*_\mathrm{i}$ *and* QCTL_i *are expressively equivalent.*

We now prove that for the structure semantics, QCTL_i is no more expressive than QCTL, and thus has the same expressivity as MSO.

▶ **Theorem 13.** *Under structure semantics,* QCTL_i *and* QCTL *are expressively equivalent.*

Proof. It is quite clear that QCTL_i subsumes QCTL. Observe however that the quantifier on propositions from QCTL can be translated using the quantifier $\exists^{[n]}$ only because we have fixed the dimension of our models. If we allowed for models with arbitrary dimension we would have to add the classic quantifier $\exists$ in the syntax of QCTL_i for it to capture QCTL.

For the other direction, we define a translation $\widetilde{\ }$ from QCTL_i to QCTL. We only provide the inductive case for the quantification on propositions, the others being trivial.

$$\widetilde{\exists^o p.\, \varphi} := \exists p.\, \left(\bigwedge_{(l_1,\dots,l_k) \in X_{o \cap [n]}} \mathbf{AG}(\bigwedge_{i=1}^{k} p_{l_i} \to p) \vee \mathbf{AG}(\bigwedge_{i=1}^{k} p_{l_i} \to \neg p) \right) \wedge \widetilde{\varphi}.$$

Observe that checking uniformity of p in the reachable part of the model is sufficient, as the labelling of unreachable states is indifferent. It can be proven easily that for every CKS $\mathcal{S}$, state $s \in \mathcal{S}$ and formula $\varphi \in \mathsf{QCTL_i}$, it holds that $\mathcal{S}, s \models_s \varphi$ iff $\mathcal{S}, s \models_s \widetilde{\varphi}$. ◀

▶ Remark. One can check that $|\widetilde{\varphi}| = O(nm^n|\varphi|)$, where $m = \max_{i\in[n]} |L_i|$.

4.2 $\mathsf{QCTL_i}$ and MSO with equal level

We briefly recall the definition $\mathsf{MSO_{eq}}$ (see, *e.g.*, [14, 46] for more detail). In the following, $\mathit{Var}_1 = \{x, y, \ldots\}$ (resp. $\mathit{Var}_2 = \{X, Y, \ldots\}$) is a countably-infinite set of first-order (resp. second-order) variables. We also use a predicate P_p for each atomic proposition $p \in \mathcal{AP}$.

The syntax of MSO with equal level relation, or $\mathsf{MSO_{eq}}$, is given by the following grammar:

$$\varphi ::= P_p(x) \mid x = y \mid S(x, y) \mid x \in X \mid \mathrm{eq}(x, y) \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x.\varphi \mid \exists X.\varphi$$

where $p \in \mathcal{AP}$, $x, y \in \mathit{Var}_1$ and $X \in \mathit{Var}_2$.

The syntax of MSO is obtained by removing the $\mathrm{eq}(x, y)$ production rule. We write $\varphi(x_1, \ldots, x_i, X_1, \ldots, X_j)$ to indicate that variables $x_1, \ldots, x_i$ and $X_1, \ldots, X_j$ may appear free in φ. Without loss of generality we assume that a variable cannot appear both free and quantified in a formula. We use the standard semantics of MSO, the successor relation symbol S being interpreted by the transition relation on Kripke structures, and by the child relation on trees. The semantics for $\mathsf{MSO_{eq}}$ is only defined on trees, and $\mathrm{eq}(x, y)$ holds if the nodes denoted by x and y are at the same depth in the tree. We write $\mathcal{M}, s_1, \ldots, s_i, S_1, \ldots, S_j \models \varphi(x_1, \ldots, x_i, X_1, \ldots, X_j)$ when φ holds in model $\mathcal{M}$ when x_k (resp. X_k) is interpreted as s_k (resp. S_k) for $k \in [i]$ (resp. $k \in [j]$).

Since we aim at comparing the expressivity of MSO (with equal level predicate in the case of tree semantics) with that of the modal logic $\mathsf{QCTL_i}$, we will consider MSO formulas of the form $\varphi(x)$, where x is a free variable representing the point of evaluation in the model.

First, we have seen that under the structure semantics $\mathsf{QCTL_i}$ has the same expressivity as QCTL. Since QCTL has the same expressivity as MSO (evaluated on reachable parts of the structures) [30], we obtain the following corollary of Theorem 13:

▶ **Corollary 14.** *Under structure semantics,* MSO *and* $\mathsf{QCTL_i}$ *are equally expressive.*

We now turn to the case of the tree semantics. The constraint put on the tree semantics for the proposition quantifier involves testing length equality for arbitrarily long paths or, in terms of trees, comparing the depths of arbitrarily deep nodes. It is thus not a surprise that $\mathsf{QCTL_i}$ with tree semantics is more expressive than MSO on trees. It also seems natural that extending MSO with the equal level predicate allows to capture this constraint on proposition quantification, and thus that $\mathsf{MSO_{eq}}$ is as expressive as $\mathsf{QCTL_i}$ with tree semantics. We establish with the following theorem that in fact the other direction also holds.

▶ **Theorem 15.** *Under tree semantics,* $\mathsf{MSO_{eq}}$ *and* $\mathsf{QCTL_i}$ *are equally expressive.*

Proof. We first show how to express in $\mathsf{MSO_{eq}}$ that two nodes in the unfolding of a CKS are o-indistinguishable (see Definition 11). Let o be an observation. We define the $\mathsf{MSO_{eq}}$ formula $\varphi^o(x, y)$ as follows:

$$\varphi^o(x,y) := \mathrm{eq}(x,y) \wedge \forall x'.\forall y'.\left(x' \preccurlyeq x \wedge y' \preccurlyeq y \wedge \mathrm{eq}(x',y') \to \bigwedge_{i\in o\cap[n]} \bigwedge_{l\in L_i} P_{p_l}(x') \leftrightarrow P_{p_l}(y')\right)$$

where $x' \preccurlyeq x$ is an MSO formula expressing that there is a path from x' to x.

Clearly, for every CKS $\mathcal{S}$ and nodes u, u' in its unfolding $t_{\mathcal{S},s}$ from some state s,

$$t_{\mathcal{S},s}, u, u' \models \varphi^o(x,y) \quad \text{iff} \quad u \approx_o u'.$$

It is then easy to see that $\mathsf{QCTL_i}$ with tree semantics can be translated into $\mathsf{MSO_{eq}}$: the translation for CTL is standard, and propositional quantification with imperfect information can be expressed using second order quantification and the above formula for o-indistinguishability.

For the other direction, we build upon the following translation from MSO to QCTL, presented in [30]. For $\varphi(x, x_1, \dots, x_i, X_1, \dots, X_j) \in \mathsf{MSO}$, we inductively define $\widehat{\varphi}$ as:

$$\begin{array}{rclrcl}
\widehat{P_p(x)} & = & p & \widehat{P_p(x_k)} & = & \mathbf{EF}(p_{x_k} \wedge p) \\
\widehat{x = x_k} & = & p_{x_k} & \widehat{x_k = x_l} & = & \mathbf{EF}(p_{x_k} \wedge p_{x_l}) \\
\widehat{x \in X_k} & = & p_{X_k} & \widehat{x_k \in X_k} & = & \mathbf{EF}(p_{x_k} \wedge p_{X_k}) \\
\widehat{\neg\varphi'} & = & \neg\widehat{\varphi'} & \widehat{\varphi_1 \vee \varphi_2} & = & \widehat{\varphi_1} \vee \widehat{\varphi_2} \\
\widehat{\exists x_k.\varphi'} & = & \exists p_{x_k}.\, \mathrm{uniq}(p_{x_k}) \wedge \widehat{\varphi'} & \widehat{\exists X_k.\varphi'} & = & \exists p_{X_k}.\, \widehat{\varphi'} \\
\widehat{S(x, x_k)} & = & \mathbf{EX} p_{x_k} & \widehat{S(x_k, x)} & = & \perp
\end{array}$$

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

where $\mathrm{uniq}(p) := \mathbf{EF}p \wedge \forall q.\ (\mathbf{EF}(p \wedge q) \to \mathbf{AG}(p \to q))$ holds in a tree iff it has exactly one node labelled with p. Observe that x being interpreted as the root of a tree it has no incoming edge, hence the translation of $S(x_k, x)$.

We extend this translation into one from $\mathsf{MSO_{eq}}$ to $\mathsf{QCTL_i}$ by adding the following rules:

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

Observe that x being interpreted as the root, x_k is on the same level as x if and only if it is also assigned the root. Concerning the second case, recall from Section 3.2.2 that the $\mathsf{QCTL_i}$ formula $\exists^{\emptyset} p.\ \mathbf{border}(p)$ places in the tree one unique horizontal line of p's. Requiring that x_k and x_l be both on this line thus ensures that they are on the same level. It is then easy to prove by induction the following lemma:

▶ **Lemma 16.** *For every $\varphi(x, x_1, \dots, x_i, X_1, \dots, X_j) \in \mathsf{MSO_{eq}}$ and every pointed CKS $(\mathcal{S}, s)$,*

$$t_{\mathcal{S},s}, s, u_1, \dots, u_i, U_1, \dots, U_j \models \varphi(x, x_1, \dots, x_i, X_1, \dots, X_j) \quad \textit{iff} \quad t'_{\mathcal{S}}(s), s \models_t \widehat{\varphi}$$

where $t'_{\mathcal{S},s}$ is obtained from $t_{\mathcal{S},s}$ by changing the labelling for variables p_{x_k} and p_{X_k} as follows: $p_{x_k} \in \ell'(u)$ if $u = u_k$ and $p_{X_k} \in \ell'(u)$ if $u \in U_k$.

In particular, it follows that $t_{\mathcal{S},s}, s \models \varphi(x)$ iff $t_{\mathcal{S},s}, s \models \widehat{\varphi}$. ◀

▶ Remark. The two-way translation between $\mathsf{QCTL_i}$ and $\mathsf{MSO_{eq}}$ shows that when local states are identified by atomic propositions, there is a normal form for $\mathsf{QCTL_i}$ formulas involving only blind and perfect-information quantifiers.

5 Model checking $\mathsf{QCTL_i}$

We now study the model-checking problem for $\mathsf{QCTL_i^*}$, both for structure and tree semantics. In other terms, we study the problem of deciding, given a finite CKS $\mathcal{S}$, a state $s \in \mathcal{S}$ and a $\mathsf{QCTL_i^*}$ formula φ, whether it holds that $\mathcal{S}, s \models_s \varphi$ (or $\mathcal{S}, s \models_t \varphi$ for the tree semantics).

5.1 Structure semantics

We first prove that under structure semantics, similarly to QCTL^* and QCTL, the model-checking problem is PSPACE-complete for both QCTL_i and $\mathsf{QCTL}^*_\mathrm{i}$. Observe that if n is fixed the translation from QCTL_i to QCTL from Theorem 13 suffices to obtain the upper bound. But this translation, being exponential in n (see Remark 4.1), is not enough if n is not fixed; we provide an algorithm to show that the result holds even if n is part of the input.

▶ **Theorem 17.** *Under structure semantics, model checking* $\mathsf{QCTL}^*_\mathrm{i}$ *is* PSPACE*-complete.*

Proof. Hardness follows from the PSPACE-hardness of model checking QCTL [30]. For the upper bound, we modify the algorithm described in [30, Theorem 4.2]. The main difference is that when we guess a labelling for p on a CKS $\mathcal{S}$, we need to check that this labelling is uniform. With structure semantics this can be done in deterministic time $O(|\mathcal{S}|^2 \cdot n)$: look at every pair of states, and check that if they are observationally equivalent (tested by comparing at most n pairs of local states) then they agree on p.

We prove that the model-checking problem for $\mathsf{QCTL}^*_\mathrm{i}$ is in PSPACE by induction on the nesting depth k of propositional quantification in input formulas. If $k = 0$, *i.e.*, the input formula is a CTL^* formula, call a CTL^* model-checking algorithm running in polynomial space. For nesting depth $k+1$, the input formula φ is of the form $\varphi = \Phi[q_i \mapsto \exists^{o_i} p_i.\ \varphi_i]$, where Φ is a CTL^* formula and for each i, q_i is a fresh atomic proposition, o_i is an observation and φ_i a $\mathsf{QCTL}^*_\mathrm{i}$ formula of nesting depth at most k. For each i, guess in linear time a labelling for p_i, check in quadratic time that it is uniform, evaluate formula φ_i in each state with this labelling, and mark states where it holds with q_i. By induction hypothesis, evaluating φ_i can be done in polynomial space. It just remains to evaluate the CTL^* formula Φ in polynomial space. The overall procedure thus runs in nondeterministic polynomial space, and because NPSPACE = PSPACE, the problem is in PSPACE. ◀

5.2 Tree semantics

We turn to the case of tree semantics. The first undecidability result comes at no surprise since QCTL_i can express the existence of winning strategies in imperfect-information games.

▶ **Theorem 18.** *Under tree semantics, the model-checking problem for* QCTL_i *is undecidable.*

Proof. The MSO_eq theory of the binary tree is undecidable [31], and with Lemma 16 we obtain a reduction to the model-checking problem for QCTL_i. ◀

5.2.1 Alternating tree automata

We briefly recall the notion of alternating (parity) tree automata. For a set Z, $\mathbb{B}^+(Z)$ is the set of formulas built with elements of Z as atomic propositions, using only 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, $\delta : Q \times 2^{AP} \to \mathbb{B}^+(X \times Q)$ is a transition function, and $C : Q \to \mathbb{N}$ is a colouring function. To ease reading we shall write atoms in transition formulas 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 $a \in 2^{AP}$, if $\delta(q, a)$ is written in disjunctive normal form, then for every direction $x \in X$, each disjunct contains exactly one element of $\{x\} \times Q$. The *size* of an ATA is its number of states and its *index* is its number of different colours.

Because we work with trees that are not necessarily complete as they represent unfoldings of Kripke structures, we find it convenient to assume that the state set is partitioned between

$Q^\top$ and $Q^\perp$: when sent in a direction where there is no node in the input tree, states in $Q^\top$ accept immediately while states in $Q^\perp$ reject immediately[2].

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

$$(u, q, \alpha_1 \dagger \alpha_2) \to (u, q, \alpha_i) \quad \text{where } \dagger \in \{\vee, \wedge\} \text{ and } i \in \{1, 2\},$$
$$(u, q, [x, q']) \to \begin{cases} (u \cdot x, q', \delta(q', \ell(u \cdot x))) & \text{if } u \cdot x \in t \\ (u, q, \top) & \text{if } u \cdot x \notin t \text{ and } q \in Q^\top \\ (u, q, \perp) & \text{if } u \cdot x \notin t \text{ and } q \in Q^\perp \end{cases}$$

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

A tree t is *accepted* from node u by $\mathcal{A}$ if Eve has a winning strategy in $\mathcal{G}(\mathcal{A}, t, u)$, and we let $\mathcal{L}(\mathcal{A})$ be the set of trees accepted by $\mathcal{A}$ from their root.

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

▶ **Theorem 19** (Projection [43]). *Given an NTA $\mathcal{N}$ and an atomic proposition $p \in \mathcal{AP}$, one can build an NTA $\mathcal{N}\Downarrow_p$ of same size and index such that $\mathcal{L}(\mathcal{N}\Downarrow_p) = \mathcal{L}(\mathcal{N})\Downarrow_p$.*

Because it will be important to understand the automata construction for our decision procedure in Section 5.2.2, we briefly recall that the projected automaton $\mathcal{N}\Downarrow_p$ is simply automaton $\mathcal{N}$ with the only difference that when it reads the label of a node, it can choose whether p is there or not: if δ is the transition function of $\mathcal{N}$, that of $\mathcal{N}\Downarrow_p$ is $\delta'(q, a) = \delta(q, a \cup \{p\}) \vee \delta(q, a \setminus \{p\})$, for any state q and label $a \in 2^{AP}$. Another way of seeing it is that $\mathcal{N}\Downarrow_p$ first guesses a p-labelling for the input tree, and then simulates $\mathcal{N}$ on this modified input. To prevent $\mathcal{N}\Downarrow_p$ from guessing different labels for a same node in different executions, it is crucial that $\mathcal{N}$ be nondeterministic, reason why we need the next classic result: the crucial simulation theorem, due to Muller and Schupp.

▶ **Theorem 20** (Simulation [37]). *Given an ATA $\mathcal{A}$, one can build an NTA $\mathcal{N}$ of exponential size and linear index such that $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{A})$.*

The last one was established by Kupferman and Vardi to deal with imperfect information aspects in distributed synthesis. The rough idea is that, if one just observes X, uniform p-labellings on $X \times Y$-trees can be obtained by choosing the labellings directly on X-trees, and then lifting them to $X \times Y$.

▶ **Theorem 21** (Narrowing [28]). *Given an ATA $\mathcal{A}$ on $X \times Y$-trees, one can build an ATA $\mathcal{A}{\downarrow_X}$ on X-trees of same size such that for all $y \in Y$, $t \in \mathcal{L}(\mathcal{A}{\downarrow_X})$ iff $t{\uparrow_y^{X \times Y}} \in \mathcal{L}(\mathcal{A})$.*

In fact the result in [28] is stated for t (and thus also $t{\uparrow^{X \times Y}}$) a complete tree, but the proof transfers straightforwardly to this slightly more general result.

[2] Note that we could also work only with complete trees, with a special symbol labelling missing nodes.

5.2.2 A decidable fragment: hierarchy on observations

We turn to our main result, which is the identification of an important decidable fragment.

▶ **Definition 22.** A $\mathsf{QCTL}^*_{\mathrm{i}}$ formula φ is *hierarchical* if for all subformulas φ_1, φ_2 of the form $\varphi_1 = \exists^{o_1} p_1.\, \varphi_1'$ and $\varphi_2 = \exists^{o_2} p_2.\, \varphi_2'$ where φ_2 is a subformula of φ_1', we have $o_1 \subseteq o_2$.

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

▶ **Theorem 23.** *Under tree semantics, model checking* $\mathsf{QCTL}^*_{\mathrm{i},\subset}$ *is non-elementary decidable.*

In order to prove this we establish Lemma 25 below, but we first introduce a few more notations. For every $\varphi \in \mathsf{QCTL}^*_{\mathrm{i}}$, we let $I_\varphi := \bigcap_{o \in O} o$, where O is the set of observations that occur in φ, with the intersection over the empty set defined as $[n]$. We also let $X_\varphi := X_{I_\varphi}$ (recall that for $I \subseteq [n]$, $X_I = \bigtimes_{i \in I} L_i$). We will need a final important definition.

▶ **Definition 24** (Merge)**.** Let $t = (\tau, \ell)$ be an (AP, X)-tree and $t' = (\tau', \ell')$ an (AP', X)-tree. We define the *merge* of t and t' as the $(AP \cup AP')$-labelled X-tree $t \merge t' := (\tau \cap \tau', \ell'')$, where $\ell''(u) = \ell(u) \cup \ell'(u)$.

We explain the idea behind this definition. In our decision procedure, quantification on atomic propositions is performed by means of automata projection (see Theorem 19). But in order to obtain uniform labellings for these propositions, we need to first narrow down our automata and our trees (see Theorem 21), and in this process we lose information on the labelling of atomic propositions in the CKS $\mathcal{S}$ on which we evaluate the formula. To address this problem, first we assume without loss of generality that propositions that are quantified upon in Φ do not appear free in Φ. We can then partition propositions in Φ between those that are quantified upon, $AP_\exists$, and those that appear free, AP_f. We use the input tree of the automaton we build to carry the labelling for $AP_\exists$, and in the end the input tree is merged with the unfolding of $\mathcal{S}$ that carries the labelling to evaluate propositions in AP_f.

▶ **Lemma 25.** *Let* $\Phi \in \mathsf{QCTL}^*_{\mathrm{i},\subset}$ *with* $AP_\exists$ *and* AP_f *defined as above, and let* $\mathcal{S}$ *be a finite CKS over* AP_f*. For every subformula* φ *of* Φ *and state* s *of* $\mathcal{S}$*, one can build an ATA* $\mathcal{A}_s^\varphi$ *on* $(AP_\exists, X_\varphi)$*-trees such that for every* $(AP_\exists, X_\varphi)$*-tree* t *rooted in* $s{\downarrow_{X_\varphi}}$*,*

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

For an X_I-tree t, from now on $t{\uparrow^{[n]}} \merge t_{\mathcal{S},s}$ stands for $t{\uparrow_y^{[n]}} \merge t_{\mathcal{S},s}$, where $y = s{\downarrow_{[n]\setminus I}}$.

Proof. Let $\Phi \in \mathsf{QCTL}^*_{\mathrm{i},\subset}$, and let $AP_\exists$ (resp. AP_f) be the set of atomic propositions that are quantified upon (resp. that appear free) in Φ. Modulo renaming of atomic propositions, we can assume without loss of generality that $AP_\exists$ and AP_f are disjoint. Let $\mathcal{S} = (S, R, \ell_{\mathcal{S}})$ be a finite CKS 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 $\mathcal{A}_s^\varphi$. The definition builds upon the classic construction for CTL^* from [29].

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

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

The idea is that since we know the state $s \in S$ in which we want to evaluate the formula, we can read the labelling for atomic propositions in AP_f (those that are not quantified upon) directly from s. However, for propositions in $AP_\exists$, we need to read them from the input tree. Indeed, if $p \in AP_\exists$ it means that p is quantified upon in Φ: there is a subformula $\exists^o p.\, \varphi$ of Φ such that p is a subformula of φ. The automaton $\mathcal{A}_s^{\exists^o p.\, \varphi}$ will be built by narrowing, nondeterminising and projecting $\mathcal{A}_s^{\varphi}$ on p. On a given input tree t, $\mathcal{A}_s^{\exists^o p.\, \varphi}$ will thus guess a labelling for p in each node of t and simulate (the nondeterminised narrowing of) $\mathcal{A}_s^{\varphi}$ on this modified input. $\mathcal{A}_s^{\varphi}$ must therefore read the labelling for p from its input tree.

$\varphi = \neg\varphi'$: We obtain $\mathcal{A}_s^{\varphi}$ by dualising $\mathcal{A}_s^{\varphi'}$, which is a classic operation on ATAs.

$\varphi = \varphi_1 \vee \varphi_2$: Because $I_\varphi = I_{\varphi_1} \cap I_{\varphi_2}$, and each $\mathcal{A}_s^{\varphi_i}$ works on X_{φ_i}-trees, we need to narrow them so that they work on X_φ-trees: for $i \in \{1, 2\}$, we let $\mathcal{A}_i := \mathcal{A}_s^{\varphi_i}\!\downarrow_{I_\varphi}$.

We then build $\mathcal{A}_s^{\varphi}$ by taking the disjoint union of $\mathcal{A}_1$ and $\mathcal{A}_2$ and adding a new initial state that nondeterministically chooses which of $\mathcal{A}_1$ or $\mathcal{A}_2$ to execute on the input tree, so that $\mathcal{L}(\mathcal{A}_s^{\varphi}) = \mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)$.

$\varphi = \mathbf{E}\psi$: The aim is to build an automaton $\mathcal{A}_s^{\varphi}$ that works on X_φ-trees and that on input t, checks for the existence of a path in $t\!\uparrow^{[n]} \curlywedge t_{\mathcal{S},s}$ that satisfies ψ. To do so, $\mathcal{A}_s^{\varphi}$ guesses a path λ in $(\mathcal{S}, s)$. It remembers the current state in $\mathcal{S}$, which provides the labelling for atomic propositions in AP_f, and while it guesses λ it follows its projection on X_φ in its input tree t, reading the labels to evaluate propositions in $AP_\exists$.

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

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

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

The intuition is that $\mathcal{A}$ reads the current label, chooses nondeterministically which transition to take in $\mathcal{W}^\psi$, chooses a next state in $\mathcal{S}$ and proceeds in the corresponding direction in X_φ. To ensure[3] that the path it guesses is not only in $t_{\mathcal{S},s}$ but also in $t\!\uparrow^{[n]}$, it is enough to make sure that it always tries to stay inside its input tree t, which is achieved by letting $Q^\top = \emptyset$ and $Q^\perp = Q$. Thus, $\mathcal{A}$ accepts exactly the $\max(\varphi)$-labelled X_φ-trees t in which there exists a path that corresponds to some path in $t\!\uparrow^{[n]} \curlywedge t_{\mathcal{S},s}$ that satisfies ψ, where maximal state formulas are considered as atomic propositions.

Now from $\mathcal{A}$ we build the automaton $\mathcal{A}_s^{\varphi}$ over X_φ-trees labelled with real atomic propositions in $AP_\exists$. In each node it visits, this automaton guesses what should be its labelling over $\max(\psi)$, it simulates $\mathcal{A}$ accordingly, and checks that the guesses it makes are correct.

---

[3] Actually this is not very important since the tree t on which our automata will work will always be such that the domain of $t\!\uparrow^{[n]}$ contains the domain of $t_{\mathcal{S},s}$.

If the path being guessed in $t{\uparrow}^{[n]} \mathbin{\wedge\!\!\!\wedge} t_{\mathcal{S},s}$ is currently in node u ending with state s', and $\mathcal{A}^{\varphi}_s$ guesses that φ_i holds in u, it checks this guess by starting a simulation of automaton $\mathcal{A}^{\varphi_i}_{s'}$ from node $v = u{\downarrow}_{X_\varphi}$ in its input t.
For each $s' \in \mathcal{S}$ and each $\varphi_i \in \max(\psi)$ we first build $\mathcal{A}^{\varphi_i}_{s'}$, which works on X_{φ_i}-trees. Observe that $I_\varphi = \cap_{i=1}^{n} I_{\varphi_i}$, so that we need to narrow down these automata: We let $\mathcal{A}^i_{s'} := \mathcal{A}^{\varphi_i}_{s'}{\downarrow}_{I_\varphi} = (Q^i_{s'}, \delta^i_{s'}, q^i_{s'}, C^i_{s'})$. We also let $\overline{\mathcal{A}^i_{s'}} = (\overline{Q^i_{s'}}, \overline{\delta^i_{s'}}, \overline{q^i_{s'}}, \overline{C^i_{s'}})$ be its dualisation, and we assume w.l.o.g. that all the state sets are pairwise disjoint. We define the ATA $\mathcal{A}^{\varphi}_s = (Q \cup \bigcup_{i,s'} Q^i_{s'} \cup \overline{Q^i_{s'}}, \delta', q_\iota, C')$, where the colours of states are left as they were in their original automaton, and δ is defined as follows. For states in $Q^i_{s'}$ (resp. $\overline{Q^i_{s'}}$), δ agrees with $\delta^i_{s'}$ (resp. $\overline{\delta^i_{s'}}$), and for $(q^\psi, s') \in Q$ and $a \in 2^{AP_\exists}$ we let

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

$\varphi = \exists^o p.\ \varphi'$: We build automaton $\mathcal{A}^{\varphi'}_s$ that works on $X_{\varphi'}$-trees; because φ is hierarchical, we have that $o \subseteq I_{\varphi'}$ and we can narrow down $\mathcal{A}^{\varphi'}_s$ to work on X_o-trees and obtain $\mathcal{A}_1 := \mathcal{A}^{\varphi'}_s{\downarrow}_{X_o}$. By Theorem 20 we can nondeterminise it to get $\mathcal{A}_2$, which by Theorem 19 we can project with respect to p, finally obtaining $\mathcal{A}^{\varphi}_s := \mathcal{A}_2{\Downarrow}_p$.

We now prove by induction on φ that the construction is correct. In each case, we let $t = (\tau, \ell)$ be an $(AP_\exists, X_\varphi)$-tree rooted in $s{\downarrow}_{X_\varphi}$.

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

$\varphi = \neg\varphi'$: trivial.

$\varphi = \varphi_1 \vee \varphi_2$: We have $\mathcal{A}_i = \mathcal{A}^{\varphi_i}_s{\downarrow}_{X_\varphi}$, so by Theorem 21 we get that $t \in \mathcal{L}(\mathcal{A}_i)$ iff $t{\uparrow}^{X_{\varphi_i}} \in \mathcal{L}(\mathcal{A}^{\varphi_i}_s)$, which by induction hypothesis holds iff $(t{\uparrow}^{X_{\varphi_i}}){\uparrow}^{[n]} \mathbin{\wedge\!\!\!\wedge} t_{\mathcal{S},s} \models_t \varphi_i$, *i.e.*, iff $t{\uparrow}^{[n]} \mathbin{\wedge\!\!\!\wedge} t_{\mathcal{S},s} \models_t \varphi_i$. We conclude by reminding that $\mathcal{L}(\mathcal{A}^{\varphi}_s) = \mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)$.

$\varphi = \mathbf{E}\psi$: Suppose that $t' = t{\uparrow}^{[n]} \mathbin{\wedge\!\!\!\wedge} t_{\mathcal{S},s} \models_t \mathbf{E}\psi$. There exists a path λ starting at the root s of t' such that $t', \lambda \models \psi$. Again, let $\max(\psi)$ be the set of maximal state subformulas of φ, and let w be the infinite word over $2^{\max(\psi)}$ that agrees with n λ on the state formulas in $\max(\psi)$. By definition, $\mathcal{W}^\psi$ has an accepting execution on w. Now in the acceptance game of $\mathcal{A}^{\varphi}_s$ on t, Eve can guess the path λ, following $\lambda{\downarrow}_{X_\varphi}$ in its input t, and she can also guess the corresponding word w on $2^{\max(\psi)}$ and an accepting execution of $\mathcal{W}^\psi$ on w. Let $u' \in t'$ be a node of λ, s' its last direction and let $u = u'{\downarrow}_{X_\varphi} \in t$. Assume that in node u of the input tree, in a state $(q^\psi, s') \in Q$, Adam challenges Eve on some $\varphi_i \in \max(\psi)$ that she assumes to be true in u', *i.e.*, Adam chooses the conjunct $\delta^i_{s'}(q^i_{s'}, a)$, where a is the label of u. Note that in the evaluation game this means that Adam moves to position $(u, (q^\psi, s'), \delta^i_{s'}(q^i_{s'}, a))$. We want to show that Eve wins from this position.
Let t_u (resp. $t'_{u'}$) be the subtree of t (resp. t') starting in u (resp. u')[4]. It is enough to show that t_u is accepted by $\mathcal{A}^i_{s'} = \mathcal{A}^{\varphi_i}_{s'}{\downarrow}_{I_\varphi}$. Observe that t_u is rooted in the last direction

[4] 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 $\ell_u(x \cdot w') = \ell(w \cdot x \cdot w')$: we remove from each node all directions before $\mathrm{last}(u)$.

of $u = u'\downarrow_{I_\varphi}$, and since the last direction of u' is s' we have that t_u is rooted in $s'\downarrow_{I_\varphi}$. Let us write $t'' = t_u\uparrow^{I_{\varphi_i}}_{s''}$, where $s'' = s'\downarrow_{I_{\varphi_i}}$. By Theorem 21, because $\mathcal{A}^i_{s'} = \mathcal{A}^{\varphi_i}_{s'}\downarrow_{I_\varphi}$ and $s'\downarrow_{I_\varphi} = (s'\downarrow_{I_{\varphi_i}})\downarrow_{I_\varphi}$, we have that

$$t_u \in \mathcal{L}(\mathcal{A}^i_{s'}) \quad \text{iff} \quad t'' \in \mathcal{L}(\mathcal{A}^{\varphi_i}_{s'}). \tag{1}$$

Since t'' is rooted in $s'\downarrow_{I_{\varphi_i}}$ we can apply the induction hypothesis on t'' with φ_i, and we get that

$$t'' \in \mathcal{L}(\mathcal{A}^{\varphi_i}_{s'}) \quad \text{iff} \quad t''\uparrow^{[n]} \curlywedge t_{\mathcal{S},s'} \models_t \varphi_i. \tag{2}$$

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

$$t'_{u'} = t''\uparrow^{[n]} \curlywedge t_{\mathcal{S},s'}. \tag{3}$$

Putting (1), (2) and (3) together, we obtain that

$$t_u \in \mathcal{L}(\mathcal{A}^i_{s'}) \quad \text{iff} \quad t'_{u'} \models_t \varphi_i. \tag{4}$$

Because we have assumed that Eve guesses w correctly, we also have that $t', u' \models_t \varphi_i$, *i.e.*, $t'_{u'} \models_t \varphi_i$. This, together with (4), gives us that t_u is accepted by $\mathcal{A}^i_{s'}$.
Eve thus has a winning strategy from the initial position of the acceptance game of $\mathcal{A}^i_{s'}$ on t_u. This initial position is $(u, q^i_{s'}, \delta^i_{s'}(q^i_{s'}, a))$. Since $(u, q^i_{s'}, \delta^i_{s'}(q^i_{s'}, a))$ and $(u, (q^\psi, s'), \delta^i_{s'}(q^i_{s'}, a))$ contain the same node u and transition formula $\delta^i_{s'}(q^i_{s'}, a)$, a winning strategy in one of these positions[5] is also a winning strategy in the other, and therefore Eve wins Adam's challenge. With a similar argument, we get that also when Adam challenges Eve on some φ_i assumed not to be true in node v, Eve wins the challenge. Finally, Eve wins the acceptance game of $\mathcal{A}^\varphi_s$ on t, and thus $t \in \mathcal{L}(\mathcal{A}^\varphi_s)$.
For the other direction, assume that $t \in \mathcal{L}(\mathcal{A}^\varphi_s)$, *i.e.*, Eve wins the evaluation game of $\mathcal{A}^\varphi_s$ on t. Again, let $t' = t\uparrow^{[n]} \curlywedge t_{\mathcal{S},s}$. A winning strategy for Eve describes a path λ in $t_{\mathcal{S},s}$, which is also a path in t'. This winning strategy also defines an infinite word w over $2^{\max(\psi)}$ such that w agrees with λ on the formulas in $\max(\psi)$, and it also describes an accepting run of $\mathcal{W}^\psi$ on w. Hence $t', \lambda \models_t \psi$, and $t' \models_t \varphi$.

$\varphi = \exists^o p.\, \varphi'$**:** First, observe that because φ is hierarchical, we have that $I_\varphi = o$. Next, by Theorem 19 we have that

$$t \in \mathcal{L}(\mathcal{A}^\varphi_s) \quad \text{iff} \quad \text{there exists } t_p \equiv_p t \text{ such that } t_p \in \mathcal{L}(\mathcal{A}_2). \tag{5}$$

By Theorem 20, $\mathcal{L}(\mathcal{A}_2) = \mathcal{L}(\mathcal{A}_1)$, and since $\mathcal{A}_1 = \mathcal{A}^{\varphi'}_s\downarrow_{X_o} = \mathcal{A}^{\varphi'}_s\downarrow_{X_\varphi}$ we get by Theorem 21 that

$$t_p \in \mathcal{L}(\mathcal{A}_2) \quad \text{iff} \quad t_p\uparrow^{X_{\varphi'}}_y \in \mathcal{L}(\mathcal{A}^{\varphi'}_s), \ \text{where } y = s\downarrow_{(I_{\varphi'} \setminus I_\varphi)}. \tag{6}$$

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

$$t_p\uparrow^{X_{\varphi'}}_y \in \mathcal{L}(\mathcal{A}^{\varphi'}_s) \quad \text{iff} \quad t_p\uparrow^{X_{\varphi'}}_y\uparrow^{[n]} \curlywedge t_{\mathcal{S},s} \models_t \varphi'. \tag{7}$$

Now, with (5), (6) and (7) together with the fact that $t_p\uparrow^{X_{\varphi'}}_y\uparrow^{[n]} = t_p\uparrow^{[n]}$, we get that

$$t \in \mathcal{L}(\mathcal{A}^\varphi_s) \quad \text{iff} \quad \text{there exists } t_p \equiv_p t \text{ such that } t_p\uparrow^{[n]} \curlywedge t_{\mathcal{S},s} \models_t \varphi'. \tag{8}$$

[5] Recall that positional strategies are sufficient in parity games [49].

Let us prove that the right-hand side of (8) holds if and only if $t\uparrow^{[n]} \mathbin{/\!\!\!\backslash} t_{\mathcal{S},s} \models_t \exists^o p.\ \varphi'$. For the first direction, assume that there exists $t_p \equiv_p t$ such that $t_p\uparrow^{[n]} \mathbin{/\!\!\!\backslash} t_{\mathcal{S},s} \models_t \varphi'$. First, by definition of the merge, because $p \in AP_\exists$ and $AP_\exists$ and AP_f are disjoint, the p-labelling of $t_p\uparrow^{[n]} \mathbin{/\!\!\!\backslash} t_{\mathcal{S},s}$ is determined by the p-labelling of $t_p\uparrow^{[n]}$, which by definition of the lift is o-uniform. In addition it is clear that $t_p\uparrow^{[n]} \mathbin{/\!\!\!\backslash} t_{\mathcal{S},s} \equiv_p t\uparrow^{[n]} \mathbin{/\!\!\!\backslash} t_{\mathcal{S},s}$, which concludes this direction.

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

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

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

We can now prove Theorem 23. Let $(\mathcal{S}, s)$ be a pointed CKS, and let $\varphi \in \mathsf{QCTL}^*_{\mathrm{i},\subset}$. By Lemma 25 one can build an ATA $\mathcal{A}^\varphi_s$ such that for every labelled X_φ-tree t rooted in $s\downarrow_{X_\varphi}$, it holds that $t \in \mathcal{L}(\mathcal{A}^\varphi_s)$ iff $t\uparrow^{[n]} \mathbin{/\!\!\!\backslash} t_{\mathcal{S},s} \models_t \varphi$. Let τ be the full X_φ-tree rooted in $s\downarrow_{X_\varphi}$, and let $t = (\tau, \ell_\emptyset)$, where $\ell_\emptyset$ is the empty labelling. Clearly, $t\uparrow^{[n]} \mathbin{/\!\!\!\backslash} t_{\mathcal{S},s} = t_{\mathcal{S},s}$, and because t is rooted in $s\downarrow_{X_\varphi}$, we have $t \in \mathcal{L}(\mathcal{A}^\varphi_s)$ iff $t_{\mathcal{S},s} \models_t \varphi$. It only remains to build a simple deterministic tree automaton $\mathcal{A}$ over X_φ-trees such that $\mathcal{L}(\mathcal{A}) = \{t\}$, and check for emptiness of the alternating tree automaton $\mathcal{L}(\mathcal{A} \cap \mathcal{A}^\varphi_s)$. Because nondeterminisation makes the size of the automaton gain one exponential for each nested quantifier on propositions, the procedure is nonelementary, and hardness is inherited from the model-checking problem for QCTL [30].

6 Conclusion and future work

We have introduced the essence of imperfect information in QCTL^*, by adding internal structure to states of the models and parameterising propositional quantifiers with observational power over this internal structure. We considered both the structure and tree semantics, intimately related to the notions of *no memory* and *perfect recall* in game strategies, respectively. For the structure semantics we showed that our logic coincides with QCTL in expressive power, and thus also with MSO, and that the model-checking problem is PSPACE-complete, as for QCTL. For the tree semantics however we showed that our logic is expressively equivalent to MSO with equal level, and that its model-checking problem is thus undecidable. But we established, thanks to automata techniques made possible by our modelling choices, that model checking hierarchical formulas is decidable.

Several future work directions await us. First it would be interesting to study QCTL_i under the amorphous semantics, studied by French for QCTL in [17]. We would also like to investigate fragments with better complexity, as well as the satisfiability problem for QCTL_i. Then we believe that there may be interesting connections with Chain Logic with equal level, a restriction of $\mathsf{MSO}_{\mathrm{eq}}$ that is decidable on trees. Does it correspond to another interesting decidable fragment of $\mathsf{QCTL}^*_{\mathrm{i}}$? Finally, we aim at exploiting our last result in various logics for strategic reasoning with imperfect information, such as ATL^*_{sc} and SL.

References

1 E. A.Emerson and C.-L. Lei. Modalities for model checking: Branching time strikes back. In *PL'85*, pages 84–96. ACM Press, 1985.

2 T. Ågotnes, V. Goranko, and W. Jamroga. Alternating-Time Temporal Logics with Irrevocable Strategies. In *TARK'07*, pages 15–24, 2007.

3 R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-Time Temporal Logic. *JACM*, 49(5):672–713, 2002.

4 Dietmar Berwanger, Anup Basil Mathew, and Marie van den Bogaard. Hierarchical information patterns and distributed strategy synthesis. In *Automated Technology for Verification and Analysis - 13th International Symposium, ATVA 2015, Shanghai, China, October 12-15, 2015, Proceedings*, pages 378–393, 2015. `doi:10.1007/978-3-319-24953-7\_28`.

5 T. Brihaye, A. Da Costa Lopes, F. Laroussinie, and N. Markey. ATL with Strategy Contexts and Bounded Memory. In *LFCS'09*, LNCS 5407, pages 92–106. Springer, 2009.

6 P. Čermák, A. Lomuscio, F. Mogavero, and A. Murano. MCMAS-SLK: A Model Checker for the Verification of Strategy Logic Specifications. In *CAV'14*, LNCS 8559, pages 524–531. Springer, 2014.

7 K. Chatterjee, T.A. Henzinger, and N. Piterman. Strategy Logic. *Information and Computation*, 208(6):677–693, 2010.

8 E.M. Clarke and E.A. Emerson. Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic. In *LP'81*, LNCS 131, pages 52–71. Springer, 1981.

9 E.M. Clarke, O. Grumberg, and D.A. Peled. *Model Checking.* MIT Press, 2002.

10 A. Da Costa, F. Laroussinie, and N. Markey. ATL with Strategy Contexts: Expressiveness and Model Checking. In *FSTTCS'10*, LIPIcs 8, pages 120–132, 2010.

11 M. Dastani and W. Jamroga. Reasoning about strategies of multi-agent programs. In *AAMAS'10*, pages 997–1004. IFAAMAS, 2010.

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

13 C. Dima and F.L. Tiplea. Model-checking ATL under Imperfect Information and Perfect Recall Semantics is Undecidable. Technical report, arXiv, 2011.

14 C. C. Elgot and M. O. Rabin. Decidability and undecidability of extensions of second (first) order theory of (generalized) successor. *J. Symb. Log.*, 31(2):169–181, 1966. URL: `http://dx.doi.org/10.2307/2269808`, `doi:10.2307/2269808`.

15 E Allen Emerson and A Prasad Sistla. Deciding branching time logic. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 14–24. ACM, 1984.

16 E.A. Emerson and J.Y. Halpern. "Sometimes" and "Not Never" Revisited: On Branching Versus Linear Time. *JACM*, 33(1):151–178, 1986.

17 Tim French. Decidability of quantifed propositional branching time logics. In *Australian Joint Conference on Artificial Intelligence*, pages 165–176. Springer, 2001.

18 Joseph Y Halpern and Moshe Y Vardi. The complexity of reasoning about knowledge and time. i. lower bounds. *Journal of Computer and System Sciences*, 38(1):195–237, 1989.

19 W. Jamroga, S. Mauw, and M. Melissen. Fairness in non-repudiation protocols. In *Proceedings of STM'11*, volume 7170 of *LNCS*, pages 122–139, 2012.

20 W. Jamroga and A. Murano. On Module Checking and Strategies. In *AAMAS'14*, pages 701–708. IFAAMAS, 2014.

21 S. Kremer and J.-F. Raskin. A game-based verification of non-repudiation and fair exchange protocols. *Journal of Computer Security*, 11(3), 2003.

22 O Kupermann and M.Y Vardi. Synthesizing distributed systems. In *Logic in Computer Science, 2001. Proceedings. 16th Annual IEEE Symposium on*, pages 389–398. IEEE, 2001.

23 O. Kupferman. Augmenting branching temporal logics with existential quantification over atomic propositions. In *CAV'95*, LNCS 939, pages 325–338. Springer, 1995.

24 O. Kupferman, P. Madhusudan, P. S. Thiagarajan, and M. Y. Vardi. Open systems in reactive environments: Control and synthesis. In *CONCUR'00*, LNCS 1877, pages 92–107. Springer, 2000.

25 O. Kupferman and M. Y. Vardi. Module checking revisited. In *CAV'97*, volume 1254 of *LNCS*, pages 36–47. Springer, 1997.

26 O. Kupferman, M.Y. Vardi, and P. Wolper. An Automata Theoretic Approach to Branching-Time Model Checking. *JACM*, 47(2):312–360, 2000.

27 O. Kupferman, M.Y. Vardi, and P. Wolper. Module Checking. *IC*, 164(2):322–344, 2001.

28 Orna Kupferman and Moshe Y Vardi. Church's problem revisited. *Bulletin of Symbolic Logic*, pages 245–263, 1999.

29 Orna Kupferman, Moshe Y. Vardi, and Pierre Wolper. An automata-theoretic approach to branching-time model checking. *J. ACM*, 47(2):312–360, 2000.

30 François Laroussinie and Nicolas Markey. Quantified CTL: expressiveness and complexity. *Logical Methods in Computer Science*, 10(4), 2014. URL: `http://dx.doi.org/10.2168/LMCS-10(4:17)2014`, `doi:10.2168/LMCS-10(4:17)2014`.

31 Hans Läuchli and Christian Savioz. Monadic second order definable relations on the binary tree. *The Journal of Symbolic Logic*, 52(01):219–226, 1987.

32 A. Lomuscio and F. Raimondi. MCMAS : A model checker for multi-agent systems. In *TACAS'06*, LNCS 4314, pages 450–454, 2006.

33 A.D.C. Lopes, F. Laroussinie, and N. Markey. ATL with Strategy Contexts: Expressiveness and Model Checking. In *FSTTCS'10*, LIPIcs 8, pages 120–132. Leibniz-Zentrum fuer Informatik, 2010.

34 F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. Reasoning About Strategies: On the Model-Checking Problem. *TOCL*, 15(4):34:1–42, 2014.

35 F. Mogavero, A. Murano, and M.Y. Vardi. Reasoning About Strategies. In *FSTTCS'10*, LIPIcs 8, pages 133–144. Leibniz-Zentrum fuer Informatik, 2010.

36 F. Mogavero, A. Murano, and M.Y. Vardi. Relentful Strategic Reasoning in Alternating-Time Temporal Logic. In *LPAR'10*, LNAI 6355, pages 371–387. Springer, 2010.

37 David E. Muller and Paul E. Schupp. Simulating alternating tree automata by nondeterministic automata: New results and new proofs of the theorems of rabin, mcnaughton and safra. *Theor. Comput. Sci.*, 141(1&2):69–107, 1995.

38 Gary Peterson, John Reif, and Salman Azhar. Lower bounds for multiplayer noncooperative games of incomplete information. *Computers & Mathematics with Applications*, 41(7):957–992, 2001.

39 Gary Peterson, John Reif, and Salman Azhar. Decision algorithms for multiplayer noncooperative games of incomplete information. *Computers & Mathematics with Applications*, 43(1):179–206, 2002.

40 Sophie Pinchinat and Stéphane Riedweg. A decidable class of problems for control under partial observation. *Information Processing Letters*, 95(4):454–460, 2005.

41 A. Pnueli. The Temporal Logic of Programs. In *FOCS'77*, pages 46–57. IEEE Computer Society, 1977.

42 A. Pnueli and R. Rosner. On the Synthesis of a Reactive Module. In *POPL'89*, pages 179–190. Association for Computing Machinery, 1989.

43 Michael O Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the american Mathematical Society*, 141:1–35, 1969.

44 John H. Reif. The complexity of two-player games of incomplete information. *J. Comput. Syst. Sci.*, 29(2):274–301, 1984.

45 A.P. Sistla. *Theoretical Issues in the Design and Cerification of Distributed Systems.* PhD thesis, Harvard University, Cambridge, MA, USA, 1983.

46 W. Thomas. Infinite trees and automaton-definable relations over omega-words. *Theor. Comput. Sci.*, 103(1):143–159, 1992.

47 W. van der Hoek and M. Wooldridge. Cooperation, knowledge and time: Alternating-time Temporal Epistemic Logic and its applications. *Studia Logica*, 75(1):125–157, 2003.

48 M.Y. Vardi and L.J. Stockmeyer. Improved upper and lower bounds for modal logics of programs: Preliminary report. In *STOC'85*, pages 240–251, 1985.

49 Wieslaw Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theor. Comput. Sci.*, 200(1-2):135–183, 1998.