

Why Propositional Quantification Makes Modal Logics on Trees Robustly Hard?

Bartosz Bednarczyk
Institute of Computer Science
University of Wrocław, Poland

Stéphane Demri
LSV, CNRS, ENS Paris-Saclay
Université Paris-Saclay, France

Abstract—Adding propositional quantification to the modal logics K , T or $S4$ is known to lead to undecidability but CTL with propositional quantification under the tree semantics ($QCTL^t$) admits a non-elementary Tower-complete satisfiability problem. We investigate the complexity of strict fragments of $QCTL^t$ as well as of the modal logic K with propositional quantification under the tree semantics. More specifically, we show that $QCTL^t$ restricted to the temporal operator EX is already Tower-hard, which is unexpected as EX can only enforce local properties. When $QCTL^t$ restricted to EX is interpreted on N -bounded trees for some $N \geq 2$, we prove that the satisfiability problem is $AExp_{Poi}$ -complete; $AExp_{Poi}$ -hardness is established by reduction from a recently introduced tiling problem, instrumental for studying the model-checking problem for interval temporal logics. As consequences of our proof method, we prove Tower-hardness of $QCTL^t$ restricted to EF or to $EXEF$ and of the well-known modal logics K , KD , GL , $S4$, $K4$ and $D4$, with propositional quantification under a semantics based on classes of trees.

I. INTRODUCTION

a) The flavours of propositional quantification in modal and temporal logics: An early work on modal logics with propositional quantification can be found in [1] and the undecidability of the propositional modal logic K (resp. T , $K4$ and $S4$) augmented with propositional quantification is first established in [2] (by reduction from the second-order arithmetic). A notable exception is the decidability of $S5$ augmented with propositional quantification [2], [3] but having strictly more than one $S5$ modality leads again to undecidability [4] (see also [5], [6]).

Many subsequent works have dealt with second-order modal logics, see e.g. [7], [8], [9], but in the realm of temporal logics, LTL with propositional quantification (written QLTL) is introduced in Sistla's PhD thesis [10] (see also [11]) and non-elementarity of the satisfiability problem is a consequence of [12]. By contrast, CTL with propositional quantification (written QCTL) admits already an undecidable satisfiability problem by [2] but its variant under the tree semantics (written $QCTL^t$) admits a Tower-complete satisfiability problem [13], [14] (Tower is introduced in [15]). Having a tree semantics means that the formulae of $QCTL^t$ are interpreted on computation trees obtained from the unwinding of finite (total) Kripke

structures, which allows us to regain decidability (see a similar approach in [16]). This is a major observation from [13], partly motivated by the design of decision procedures for ATL with strategy contexts [17]. The tree semantics, as far as satisfiability is concerned, amounts to consider Kripke structures that are finite-branching trees in which all the maximal branches are infinite.

The modal logic K with propositional quantification from [2] is interpreted under the structure semantics, as classified in [13], but many variants of propositional quantification exist in the literature (see e.g. [18], [19], [20] and [21], [22] in the context of epistemic reasoning). Sometimes, propositional quantification is syntactically restricted in the temporal language, but its inclusion is motivated by a gain of expressive power while preserving the decidability of the reasoning tasks. By way of example, in [23], an extension of the modal μ -calculus with partial propositional quantification is introduced to perform control synthesis, whereas an extension for model-checking computer systems is also presented in [24].

Interestingly enough, hybrid logics with the downarrow operator \downarrow_x , see e.g. [25], [26], [27], [28], can be understood as a form of propositional quantification since $\downarrow_x \phi$ enforces that the propositional variable x holds true only at the current world (before evaluating the formula ϕ). In such logics, the companion formula $@_x \psi$ allows to jump to the unique world satisfying x ; $@_x$ is a powerful operator to navigate in the structure [27] (obviously, this is related to the universal modality, see e.g. [29]).

b) Our motivations: As recalled above, the modal logics K augmented with propositional quantification are undecidable [2] and a fortiori, undecidability holds for fragments of CTL with propositional quantification. Actually, these results hold under the structure semantics but $QCTL^t$ (tree semantics) admits Tower-complete satisfiability and model-checking problems [13]. As $QCTL^t$ can express that every tree node has exactly one child, Tower-hardness for the satisfiability problem for $QCTL^t$, is a corollary of the Tower-hardness of the satisfiability problem for QLTL, see e.g. [12], [11], [30].

Given the central position of the modal logic K , surprisingly, the complexity of the satisfiability problem for K with propositional quantification under the tree semantics has never been investigated (closely related to $QCTL^t_X$ –

i.e. QCTL^t restricted to \mathbf{EX} – as \mathbf{EX} corresponds to \diamond in \mathbf{K} but with total models). $\text{QCTL}_{\mathbf{X}}^t$ is a natural and modest fragment of QCTL^t and we aim at characterising its complexity. Furthermore, the model-checking problem for QCTL^t is TOWER-hard even with input Kripke structures having at most two successor worlds per world and TOWER-hardness of QLTL holds with linear structures of length ω , see e.g. [11]. Thus, it is worth understanding what happens with the satisfiability problem for $\text{QCTL}_{\mathbf{X}}^t$ when the tree models are N -bounded (i.e., each node has at most N children) for some fixed $N \geq 1$.

c) *Our contributions:* Given QCTL^t , the extension of CTL with propositional quantification under the tree semantics (i.e., the models are finite-branching trees where all the maximal branches are infinite), let $\text{QCTL}_{\leq N}^t$ be its variant in which the models are N -bounded, for some $N \geq 1$. We write $\text{QCTL}_{\mathbf{X}}^t$ and $\text{QCTL}_{\mathbf{X}, \leq N}^t$, to denote respectively the restriction to operator \mathbf{EX} , and $\text{QCTL}_{\mathbf{F}}^t$ to denote the restriction of QCTL^t to \mathbf{EF} .

(\star) We prove that for all $N \geq 2$, the satisfiability problem for $\text{QCTL}_{\mathbf{X}, \leq N}^t$ is AEXP_{POL} -complete (Section III). AEXP_{POL} is the class of problems decidable with an exponential-time alternating TM with a polynomial number of alternations. For AEXP_{POL} -hardness, our reduction is from the alternating multi-tiling problem introduced in [31], [32] (used to prove that the model-checking problem for the interval temporal logic $B\bar{E}$ with regular expressions is AEXP_{POL} -hard).

($\star\star$) More generally and despite the modest scope of \mathbf{EX} , the satisfiability problem for $\text{QCTL}_{\mathbf{X}}^t$ is shown TOWER-hard (Theorem 20), by a uniform reduction from k - NEXP -complete tiling problems (uniformity is with respect to k). The corresponding upper bound is known from [13] and it is worth noting that all the TOWER upper bound results presented in this paper are based on translations into the satisfiability problem for QCTL^t , sometimes via intermediate decision problems, which eventually, uses Rabin’s Theorem [33] in some essential way. The hardness proof is one of the main results of the paper and amounts first to show that one can enforce that a node has a number of children equal to some tower of exponentials of height k with a formula of size exponential in k . By contrast, checking the satisfiability status of CTL^* formulae, requires only to consider tree models with branching degree bounded by the size of the formula, see e.g. [34], [30]. Once this complex construction enforcing a very high number of children is performed, the reduction from the tiling problems can be done with the help of other properties on the number of children. Hence, even though $\text{QCTL}_{\mathbf{X}}$ under the structure semantics is undecidable [2] and the variant of $\text{QCTL}_{\mathbf{X}}^t$ under the tree semantics (QCTL^t) is decidable by [13], the problem admits a high complexity despite the local range of \mathbf{EX} .

($\star\star\star$) By adapting our proof method, we show that $\text{QCTL}_{\mathbf{F}}^t$ and $\text{QCTL}_{\mathbf{XF}}^t$ (variant of $\text{QCTL}_{\mathbf{F}}^t$ with the unique operator \mathbf{EXEF}) are TOWER-hard too (see Section V-A).

($\star\star\star$) As \mathbf{EX} , \mathbf{EF} and \mathbf{EXEF} correspond to the modality \diamond in several modal logics (e.g., \mathbf{EXEF} corresponds to \diamond in transitive frames), we are able to establish TOWER-completeness for standard modal logics with propositional quantification when interpreted on tree-like structures (see Section V-B). For instance, as the provability logic \mathbf{GL} (after Gödel and Löb) is complete for the class of finite transitive trees, i.e., the accessibility relation is the transitive closure of the child-relation in the tree, see e.g. [35], we also investigate the satisfiability problem under the finite tree semantics. We show that $\text{QCTL}_{\mathbf{X}}^{ft}$ and $\text{QCTL}_{\mathbf{XF}}^{ft}$ (*ft* stands for ‘finite tree semantics’) are TOWER-complete too. The satisfiability problem for \mathbf{K} (resp. \mathbf{GL}) with propositional quantification under the finite (resp. transitive) tree semantics is shown TOWER-complete. TOWER-complete versions of $\mathbf{S4}$, $\mathbf{D4}$ and $\mathbf{K4}$ with propositional quantification under adequate classes of tree-like structures are also established in Section V-B.

II. PRELIMINARIES

A. Kripke structures and computation trees

Below, we recall standard definitions about Kripke structures. Let $AP = \{p, q, x, y, \dots\}$ be a countably infinite set of propositional variables. A *Kripke structure* \mathcal{K} is a triple $\langle W, R, l \rangle$, where W is a set of *worlds*, $R \subseteq W \times W$ is a total transition relation (for all $w \in W$, there is $w' \in W$ such that $(w, w') \in R$) and $l : W \rightarrow 2^{AP}$ is a labelling function. Totality is a standard property for defining classes of models for temporal logics such as CTL. In the sequel, by a ‘Kripke structure’ we mean a structure according to the above definition, otherwise when arbitrary (or finite, etc.) Kripke structures need to be considered, we explicitly specify which classes of structures we have in mind. For instance, a tree model (resp. finite tree model) $\mathfrak{T} = \langle V, E, l \rangle$ is an arbitrary Kripke structure where (V, E) is a (resp. finite) rooted tree. Standard definitions about trees are omitted herein. Such structures play an important role in the paper, as the tree semantics involves specific tree models (those obtained as computation trees of Kripke structures).

Given an arbitrary Kripke structure $\mathcal{K} = \langle W, R, l \rangle$ and $w \in W$, a *finite path* π from w is a finite sequence w_0, \dots, w_n such that $w_0 = w$ and for all $i \in [0, n-1]$, we have $(w_i, w_{i+1}) \in R$. An *infinite path* from w is an infinite sequence w_0, \dots, w_n, \dots such that $w_0 = w$ and for all $i \geq 0$, we have $(w_i, w_{i+1}) \in R$. With $\Pi_{\mathcal{K}, w}$ we denote the set of all finite paths starting from a world $w \in W$ on a Kripke structure \mathcal{K} . Given an arbitrary Kripke structure $\mathcal{K} = \langle W, R, l \rangle$ and $w \in W$, the *computation tree unfolding* \mathcal{K} from w is the tree model $\mathfrak{T}_{\mathcal{K}, w} = \langle V, E, l' \rangle$ such that (a) $V \stackrel{\text{def}}{=} \Pi_{\mathcal{K}, w}$, (b) $\pi E \pi' \stackrel{\text{def}}{\iff} \pi = w_0, \dots, w_n, \pi'$ is of the form π, w_{n+1} for some $w_{n+1} \in W$ and $(w_n, w_{n+1}) \in R$, and (c) for all $\pi = w_0, \dots, w_n \in V$, we have $l'(\pi) \stackrel{\text{def}}{=} l(w_n)$. So, when \mathcal{K} is finite and total, $\mathfrak{T}_{\mathcal{K}, w}$ is a finite-branching tree model in which all the maximal branches are infinite

(Thus it is also a –total– Kripke structure). In the sequel, unless otherwise stated, the tree semantics involves such finite-branching trees with infinite maximal branches.

B. The logics QCTL^s , QCTL^t and QCTL^{ft}

The formulae of QCTL-like logics are defined from the grammar below by extending the set of formulae from CTL [36] with propositional quantification:

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid \mathbf{EX}\phi \mid \mathbf{E}(\phi\mathbf{U}\phi) \mid \mathbf{A}(\phi\mathbf{U}\phi) \mid \exists p \phi,$$

where $p \in AP$. We use the standard abbreviations $\vee, \rightarrow, \leftrightarrow, \perp, \top$, as well as other temporal operators like \mathbf{AX} , \mathbf{EF} , \mathbf{AG} : $\mathbf{AX}(\phi) \stackrel{\text{def}}{=} \neg\mathbf{EX}(\neg\phi)$, $\mathbf{EF}(\phi) \stackrel{\text{def}}{=} \mathbf{E}(\top\mathbf{U}\phi)$ and $\mathbf{AG}(\phi) \stackrel{\text{def}}{=} \neg\mathbf{EF}(\neg\phi)$. Similarly, the universal quantifier $\forall p \phi$ is used as $\neg\exists p \neg\phi$. We denote by $|\phi|$ the length of the formula ϕ measured in a standard way, i.e., as the number of symbols used to write ϕ . The modal depth of a formula ϕ , written $md(\phi)$ is defined as usually, as the maximal number of nested temporal operators in ϕ .

The restriction of QCTL to the only temporal operator \mathbf{EX} (resp. \mathbf{EF}) is written $\text{QCTL}_{\mathbf{X}}$ (resp. $\text{QCTL}_{\mathbf{F}}$). \mathbf{AX} (resp. \mathbf{AG}) is allowed in $\text{QCTL}_{\mathbf{X}}$ (resp. in $\text{QCTL}_{\mathbf{F}}$) as it is the dual operator of \mathbf{EX} (resp. \mathbf{EF}). The main object of study in the paper is the logic $\text{QCTL}_{\mathbf{X}}$ under the tree semantics (below, written $\text{QCTL}_{\mathbf{X}}^t$). Moreover, we write $\text{QCTL}_{\mathbf{XF}}$ to denote the restriction of QCTL to the (combined) temporal operator \mathbf{EXEF} . In the rest of the paper, we refer to QCTL (or to some of its fragments) to denote a set of formulae and the notation QCTL^\bullet with a superscript ‘ \bullet ’ refers to a logic based on QCTL (or on some of its fragments) under a specific semantics and for which the symbol ‘ \bullet ’ is just a reminder of the semantics.

The propositional quantifiers can be interpreted in various ways, see e.g. [24], [37], [18], [23], [20]. Below, we provide a satisfaction relation defined on arbitrary Kripke structures, providing the so-called *structure semantics* for QCTL (and leading to the logic QCTL^s). However, the version of QCTL with formulae interpreted on computation trees obtained from the unfoldings of finite Kripke structures (i.e., in that case the satisfaction relation operates on tree-like structures), is known as QCTL under the *tree semantics* (written QCTL^t) and is extensively studied in [13]. To define formally the semantics for propositional quantifiers, we introduce the notion of X -equivalence where X is a set of propositional variables. We say that two Kripke structures $\mathcal{K} = \langle W, R, l \rangle$ and $\mathcal{K}' = \langle W', R', l' \rangle$ are X -equivalent (written $\mathcal{K} \approx_X \mathcal{K}'$), iff $W = W'$, $R = R'$ and $l(w) \cap X = l'(w) \cap X$ for all $w \in W$.

Given an arbitrary Kripke structure $\mathcal{K} = \langle W, R, l \rangle$ (possibly infinite, non-total, etc.), $w \in W$ and a formula ϕ in QCTL, the satisfaction relation \models is defined as follows (usual clauses for Boolean connectives are omitted):

$$\begin{aligned} \mathcal{K}, w \models p & \quad \text{iff} \quad p \in l(w) \\ \mathcal{K}, w \models \mathbf{EX}\phi & \quad \text{iff} \quad \exists w' \text{ s.t. } (w, w') \in R \text{ and } \mathcal{K}, w' \models \phi \\ \mathcal{K}, w \models \mathbf{E}(\phi\mathbf{U}\psi) & \quad \text{iff} \quad \exists \text{ path } w_0, \dots, w_n \text{ s.t. } w_0 = w, \\ & \quad \mathcal{K}, w_n \models \psi \text{ and } \forall i \in [0, n-1], \\ & \quad \text{we have } \mathcal{K}, w_i \models \phi \end{aligned}$$

$$\begin{aligned} \mathcal{K}, w \models \mathbf{A}(\phi\mathbf{U}\psi) & \quad \text{iff} \quad \forall \text{ infinite paths } w_0, \dots, w_n, \dots \\ & \quad \text{s.t. } w_0 = w, \exists j \geq 0 \text{ s.t. } \mathcal{K}, w_j \models \psi \\ & \quad \text{and for all } i \in [0, j-1], \mathcal{K}, w_i \models \phi \\ \mathcal{K}, w \models \exists p \phi & \quad \text{iff} \quad \exists \mathcal{K}' \text{ s.t. } \mathcal{K} \approx_{AP \setminus \{p\}} \mathcal{K}' \ \& \ \mathcal{K}', w \models \phi \end{aligned}$$

Note that $\mathcal{K}, w \models \mathbf{EF}\phi$ iff there is $w' \in R^*(w)$ such that $\mathcal{K}, w' \models \phi$. Stating that there is a unique successor of w satisfying x can be expressed by $\mathbf{EX} x \wedge \neg(\exists p \mathbf{EX}(x \wedge p)) \wedge \mathbf{EX}(x \wedge \neg p)$, where p is distinct from x .

The satisfiability problem for the logic QCTL^s (under the *structure semantics*) takes as input a formula ϕ in QCTL and asks whether there is a finite (total) Kripke structure \mathcal{K} and w such that $\mathcal{K}, w \models \phi$. The *tree semantics* introduced in [13] is obtained by considering as only admissible models the computation trees of finite (total) Kripke structures. As noted in [13, Remark 5.7], an equivalent formulation can be provided: the satisfiability problem for the logic QCTL^t (under the *tree semantics*) takes as input a formula in QCTL and asks whether there is a finite-branching tree model \mathfrak{T} in which all the maximal branches are infinite such that $\mathfrak{T}, \varepsilon \models \phi$ and ε is the root of \mathfrak{T} . The equivalence is mainly due to the fact that QCTL^t can be translated into MSO over tree models with arbitrary finite branching (getting decidability by Rabin’s Theorem [33]) and by using the regular tree model property, see explanations in [38, Section 6.3]. In the sequel, we operate with the latter definition and we write $\text{SAT}(\mathfrak{L})$ to denote the satisfiability problem for the logic \mathfrak{L} .

The distinction between the tree semantics and the structure semantics is crucial and affects the computational properties of the satisfiability problems.

Proposition 1. (I) $\text{SAT}(\text{QCTL}^s)$ is undecidable [13, Theorem 5.1]. (II) $\text{SAT}(\text{QCTL}^t)$ is decidable and TOWER-complete [13, Theorem 5.3]. (III) The satisfiability problem for QCTL^t is TOWER-hard even if restricted to ω -sequences [12], [11].

All our TOWER upper bound results are based on translations into QCTL^t , sometimes via intermediate problems, which eventually uses Rabin’s Theorem [33] in some essential way. This is not surprising, as considering tree-like models and propositional quantification naturally leads to invoke the decidability of $\text{S}\omega\text{S}$ [33] or its linear version S1S [39]. Decidability of other logics with propositional quantification is shown that way in [40], [41], [16].

To conclude this section, we write QCTL^{ft} to denote the variant of QCTL^t in which the tree models are assumed to be finite. Of course, the computation trees of (total) finite Kripke structures do not lead to finite trees, but several standard modal logics such as \mathbf{K} and \mathbf{GL} , can be characterised by classes of finite tree-like structures, see Section V for further developments.

C. Complexity classes and tiling problems

In this section, we introduce tiling problems that are mainly used in Sections III-C and IV-D. Let t be a tetration function defined for integers $n, k \geq 0$, inductively

as $\mathfrak{t}(0, n) = n$ and $\mathfrak{t}(k+1, n) = 2^{\mathfrak{t}(k, n)}$. Intuitively the function \mathfrak{t} defines the tower of exponentials of height k , i.e., $\mathfrak{t}(1, n) = 2^n$, $\mathfrak{t}(2, n) = 2^{2^n}$, and so on. By k -NEXPTIME we denote the class of all problems decidable with a nondeterministic Turing machines of working time in $\mathcal{O}(\mathfrak{t}(k, p(n)))$ for some polynomial $p(\cdot)$, on each input of length n . We define TOWER as the class of all problems of time complexity bounded by a tower of exponentials, whose height is an elementary function [15]. Thus, to show TOWER-hardness (using elementary reductions [15]), it is sufficient to prove k -NEXPTIME-hardness for all k using uniform reductions [15]. For proving hardness results, we make extensive use of tiling problems, see e.g. [42].

The *tiling problem* Tiling_k takes as inputs a triple $\langle \mathcal{T}, \mathcal{H}, \mathcal{V} \rangle$ and $c \in \mathcal{T}^n$ for some $n \geq 1$ such that \mathcal{T} is a finite set of *tile types*, $\mathcal{H} \subseteq \mathcal{T} \times \mathcal{T}$ (resp. $\mathcal{V} \subseteq \mathcal{T} \times \mathcal{T}$) represents the *horizontal* (resp. *vertical*) *matching relation*, and $c = t_0, t_1, \dots, t_{n-1} \in \mathcal{T}^n$ is the *initial condition*. A *solution* for the instance $\langle \mathcal{T}, \mathcal{H}, \mathcal{V} \rangle, c$ is a mapping $\tau : [0, \mathfrak{t}(k, n) - 1] \times [0, \mathfrak{t}(k, n) - 1] \rightarrow \mathcal{T}$ such that

- (**init**) For all $i \in [0, n - 1]$, $\tau(0, i) = t_i$.
- (**hori**) For all $i, j \in [0, \mathfrak{t}(k, n) - 1]$, $i < \mathfrak{t}(k, n) - 1$ implies $(\tau(i, j), \tau(i+1, j)) \in \mathcal{H}$.
- (**verti**) For all $i, j \in [0, \mathfrak{t}(k, n) - 1]$, $j < \mathfrak{t}(k, n) - 1$ implies $(\tau(i, j), \tau(i, j+1)) \in \mathcal{V}$.

A mapping τ satisfying (hori) and (verti) is called a *tiling*. So, the problem of checking whether an instance of Tiling_k has a solution (note that k does not appear in the instance and it influences only the size of the grid) is k -NEXPTIME-complete, see e.g. [42].

Given $N \geq 1$, let us consider the satisfiability problem for $\text{QCTL}_{\mathbf{x}, \leq N}^t$ in which the structures are tree models where all the maximal branches are infinite but each node has at most N children (and at least one child). To characterise the complexity of $\text{SAT}(\text{QCTL}_{\mathbf{x}, \leq N}^t)$, we consider the complexity class AEXP_{POL} that consists of all problems decidable by an alternating Turing machine (ATM) [43] working in exponential-time and using only polynomially many alternations [31], [32]. We stress here that allowing unbounded number of alternations gives us the class EXPSPACE and classes similar to AEXP_{POL} have been considered in [44], typically $\text{STA}(\mathfrak{f}(n), \mathfrak{g}(n), \mathfrak{h}(n))$. The complexity of several logical problems has been captured by the class AEXP_{POL} , see e.g. [45], [20], [31]. For proving AEXP_{POL} -hardness, we use an elegant modification of Tiling_1 , introduced in [31], [32]. The extension amounts to consider a stack of n tilings, with a matching relation between two consecutive tile types on the same position of the grid, and quantifications over the tile types on the first row (initial conditions). Details follow below.

The *alternating multi-tiling problem* AMTP takes as inputs an even number n (in unary), $\langle \mathcal{T}, \mathcal{H}, \mathcal{V} \rangle$ (as for defining Tiling_1), $\mathcal{T}_0 \subseteq \mathcal{T}$, $\mathcal{T}_{acc} \subseteq \mathcal{T}$ and $\mathcal{T}_{multi} \subseteq \mathcal{T} \times \mathcal{T}$. Given an initial condition $c = (w_1, \dots, w_n) \in (\mathcal{T}_0^{2^n})^n$, a *solution* for c is a multi-tiling (τ_1, \dots, τ_n) on the grid $[0, 2^n - 1] \times [0, 2^n - 1]$ such that:

- (**m-init**) For all $\alpha \in [1, n]$, for all $j \in [0, 2^n - 1]$, $\tau_\alpha(0, j) = w_\alpha(j)$ (i.e., the first row of τ_α is w_α).
 - (**m-tiling**) For $\alpha \in [1, n]$, τ_α satisfies (hori) and (verti).
 - (**m-multi**) For $\alpha \in [1, n - 1]$, for all $i, j \in [0, 2^n - 1]$, $(\tau_\alpha(i, j), \tau_{\alpha+1}(i, j)) \in \mathcal{T}_{multi}$.
 - (**m-accept**) For some $j \in [0, 2^n - 1]$, $\tau_n(2^n - 1, j) \in \mathcal{T}_{acc}$.
- An instance \mathcal{I} for AMTP made of $n, \langle \mathcal{T}, \mathcal{H}, \mathcal{V} \rangle, \mathcal{T}_0, \mathcal{T}_{acc} \subseteq \mathcal{T}, \mathcal{T}_{multi} \subseteq \mathcal{T} \times \mathcal{T}$ is positive iff for all $w_1 \in \mathcal{T}_0^{2^n}$, there is $w_2 \in \mathcal{T}_0^{2^n}$ such that \dots for all $w_{n-1} \in \mathcal{T}_0^{2^n}$, there is $w_n \in \mathcal{T}_0^{2^n}$ such that there is a solution (τ_1, \dots, τ_n) for (w_1, \dots, w_n) . AMTP is shown AEXP_{POL} -complete in [31], [32].

III. WHAT HAPPENS WHEN TREES ARE BOUNDED?

In this section, we study $\text{SAT}(\text{QCTL}_{\mathbf{x}, \leq N}^t)$ with $N \geq 1$.

A. A toolkit for introducing local nominals

Below, we introduce formulae to simulate partially the use of nominals from hybrid modal logics [26]. A nominal x is usually understood as a propositional variable true at exactly one world of the model (a global property). In $\text{QCTL}_{\mathbf{x}}^t$, such a property cannot be enforced but it can be done with respect to nodes at a bounded depth from the evaluation node, whence the adjective ‘local’ for the nominals. The use of local nominals is essential in all our hardness proofs, as it allows us to simulate first-order quantification on a given set of nodes of bounded depth.

Given a tree model \mathfrak{T} and a node v , the variable x is a *nominal for the depth $k \geq 1$ from v* iff there is $v' \in V$ such that $\mathfrak{T}, v' \models x$, $vE^k v'$, and for all $v'' \neq v'$ such that $vE^k v''$, we have $\mathfrak{T}, v'' \not\models x$ (E^k is the relation obtained by composing k times E). The formula $\text{nom}(x, k)$ defined as $\mathbf{EX}^k x \wedge \neg \exists p (\mathbf{EX}^k (x \wedge p) \wedge \mathbf{EX}^k (x \wedge \neg p))$, states that x is a nominal for the depth k (\mathbf{EX}^k denotes k copies of \mathbf{EX}).

Lemma 2. *x is a nominal for the depth $k \geq 1$ from v iff $\mathfrak{T}, v \models \text{nom}(x, k)$.*

Let us define $@_x^k \phi$ as the formula $\mathbf{EX}^k (x \wedge \phi)$.

Lemma 3. *Assuming that x is a nominal for the depth $k \geq 0$ from v such that $vE^k v'$ and $\mathfrak{T}, v' \models x$, we have $\mathfrak{T}, v \models @_x^k \phi$ iff $\mathfrak{T}, v' \models \phi$.*

Given $d \geq 1$ and variables x_1, \dots, x_d (that play the role of nominals), we write $@_{x_1, \dots, x_d} \phi$ to denote the formula $@_{x_1}^1 @_{x_2}^1 \dots @_{x_d}^1 \phi$ (usually assuming that $\text{nom}(x_1, 1)$ holds and for all $i \in [2, d]$, $@_{x_1}^1 @_{x_2}^1 \dots @_{x_{i-1}}^1 \text{nom}(x_i, 1)$ holds true). We write $@_{\bar{x}} \phi$ instead of $@_{x_1, \dots, x_d} \phi$ when \bar{x} is understood as x_1, \dots, x_d . Given a node v_0 such that $\mathfrak{T}, v_0 \models \text{nom}(x_1, 1) \wedge \bigwedge_{i \in [2, d]} @_{x_1}^1 @_{x_2}^1 \dots @_{x_{i-1}}^1 \text{nom}(x_i, 1)$, we write v_1, \dots, v_d to denote the unique sequence of nodes such that for all $i \in [1, d]$, we have $v_{i-1} E v_i$ and $\mathfrak{T}, v_i \models x_i$. Existence and unicity of the nodes v_1, \dots, v_d follow from Lemma 2 and Lemma 3. Here is another useful lemma.

Lemma 4. *Assume that $\mathfrak{T}, v_0 \models \text{nom}(x_1, 1) \wedge \bigwedge_{i \in [2, d]} @_{x_1}^1 @_{x_2}^1 \dots @_{x_{i-1}}^1 \text{nom}(x_i, 1)$ and v_1, \dots, v_d is associated to x_1, \dots, x_d . Then, $\mathfrak{T}, v_0 \models @_{\bar{x}} \phi$ iff $\mathfrak{T}, v_d \models \phi$.*

Let $\text{diff-nom}(x_1, \dots, x_\alpha, k)$ be $\bigwedge_{i \in [1, \alpha]} \text{nom}(x_i, k) \wedge \bigwedge_{i < j \in [1, \alpha]} \neg @_{x_i}^k x_j$. It allows us to name α distinct nodes at the depth k . Hence, the respective nodes interpreting the nominals x_1, \dots, x_α are pairwise distinct.

Lemma 5. *Given a tree model \mathfrak{T} and a node v , we have $\mathfrak{T}, v \models \text{diff-nom}(x_1, \dots, x_\alpha, k)$ iff there are α distinct nodes v_1, \dots, v_α such that for all $i \in [1, \alpha]$, x_i is a nominal for the depth $k \geq 1$ from v .*

The proof is by an easy verification and is a variant of the proof of Lemma 2. Let us illustrate the use of $\text{diff-nom}(x_1, \dots, x_\alpha, k)$ to specify that a node has at most 2^n children, a property that can be stated in graded modal logics [46], [47], [48]. Given a finite set X of propositional variables, let us design a formula that expresses that no pair of distinct children agree on every propositional variable from X , as done in [14]. Here it is: $\text{Uni}(X) \stackrel{\text{def}}{=} \forall x, y \text{ diff-nom}(x, y, 1) \rightarrow \neg(\bigwedge_{p \in X} @_x^1 p \leftrightarrow @_y^1 p)$. So, $\diamond_{\leq 2^n} \top$ from graded modal logics can be expressed with $\exists p_0, \dots, p_{n-1} \text{ Uni}(\{p_0, \dots, p_{n-1}\})$. In Section IV, we show how to express succinctly hyperexponential bounds.

B. Beyond the EXPSPACE upper bound: AEXP_{POL}

Given ϕ in $\text{QCTL}_{\mathbf{X}, \leq N}^t$, it is clear that for a N -bounded tree model \mathfrak{T} satisfying ϕ at its root node ε , it is irrelevant what happens at nodes of depth strictly more than $md(\phi)$. Hence, ϕ is satisfiable iff there is a finite N -bounded tree structure \mathfrak{T} with all the branches of length exactly $md(\phi)$ satisfying ϕ at its root ε (as the branches of tree models are infinite, we need to consider branches of length exactly $md(\phi)$). So, \mathfrak{T} has at most $|\phi|N^{|\phi|}$ nodes. To get an algorithm in NEXPSPACE, guess such an exponential-size finite tree structure, and perform model-checking on it with an algorithm inherently in PSPACE (as model-checking finite structures with MSO is PSPACE-complete [49], [50] and ϕ can be translated to MSO in the standard way), which leads to NEXPSPACE. By Savitch's Theorem, we get the EXPSPACE upper bound. This bound is not completely satisfactory as it does not use much of $\text{QCTL}_{\mathbf{X}, \leq N}^t$ and Section III-C proves AEXP_{POL}-hardness of $\text{SAT}(\text{QCTL}_{\mathbf{X}, \leq N}^t)$ as soon as $N \geq 2$. So, the goal of this section is to establish an AEXP_{POL} upper bound. The tight upper bound for $\text{SAT}(\text{QCTL}_{\mathbf{X}, \leq N}^t)$ relies on the following ingredients.

- (i) Every formula ϕ of $\text{QCTL}_{\mathbf{X}}^t$ is logically equivalent to a $\text{QCTL}_{\mathbf{X}}^t$ formula ϕ' in prenex normal form (PNF) such that ϕ' can be computed in polynomial-time in $|\phi|$. Formulae in PNF are of the form $\mathcal{Q}_1 p_1 \cdots \mathcal{Q}_\beta p_\beta \psi$ where $\{\mathcal{Q}_1, \dots, \mathcal{Q}_\beta\} \subseteq \{\exists, \forall\}$ and ψ is quantifier-free.
- (ii) Existence of an N -bounded tree model for ϕ is equivalent to the existence of an N -bounded finite tree structure such that all branches are of length $md(\phi)$.
- (iii) Checking whether $\mathfrak{T}, \varepsilon \models \mathcal{Q}_1 p_1 \cdots \mathcal{Q}_\beta p_\beta \psi$ (involving a N -bounded finite tree with branches of length $md(\psi)$ and the input formula in PNF) can

be done with an alternating Turing machine in time $\mathcal{O}((|\psi| + \beta)|\mathfrak{T}|)$ and with at most β alternations.

To establish (i), we cannot rely on [13, Prop. 3.1] as the translation in [13, Prop. 3.1] involves temporal operators beyond the language of $\text{QCTL}_{\mathbf{X}}^t$. Fortunately, on tree models, $\mathbf{EX} \mathcal{Q} p \psi \leftrightarrow \mathcal{Q} p \mathbf{EX} \psi$ and $(\mathcal{Q} p \psi) \wedge \psi' \leftrightarrow \mathcal{Q} p (\psi \wedge \psi')$ are valid ($\mathcal{Q} \in \{\exists, \forall\}$), assuming that p does not occur in ψ' (otherwise, rename the quantified variable).

Lemma 6. *For every formula ϕ in $\text{QCTL}_{\mathbf{X}}^t$, one can compute in polynomial-time in $|\phi|$ a logically equivalent formula in PNF $\mathcal{Q}_1 p_1 \cdots \mathcal{Q}_\beta p_\beta \psi$ with $\beta \leq |\phi|$.*

So, the equivalent formula in PNF remains equivalent in $\text{QCTL}_{\mathbf{X}, \leq N}^t$. Let us be a bit more precise about proving (ii). Given a tree model \mathfrak{T} , we write $\mathfrak{T} \upharpoonright_n$ to denote its subtree obtained by taking only nodes on the depth at most n from the root. A *completion* of a finite tree \mathfrak{T}' of maximal depth n is an infinite tree \mathfrak{T} (finite-branching and all the maximal branches are infinite) such that $\mathfrak{T}' = \mathfrak{T} \upharpoonright_n$.

Lemma 7. *Let $\mathfrak{T}, \varepsilon$ be a model for the $\text{QCTL}_{\mathbf{X}}^t$ -formula ϕ . Then, any completion of $\mathfrak{T} \upharpoonright_{md(\phi)}, \varepsilon$ is also a model for ϕ .*

The construction is standard and goes in exactly the same way as in the literature, e.g. [35, Theo. 2.3]. Since we are interested in the satisfiability problem over N -bounded trees, the overall size of a structure $\mathfrak{T} \upharpoonright_{md(\phi)}$ can be bounded, typically $|\mathfrak{T} \upharpoonright_{md(\phi)}| < |\phi|N^{|\phi|}$. Consequently:

Lemma 8. *For any ϕ , ϕ is satisfiable for $\text{QCTL}_{\mathbf{X}, \leq N}^t$ iff ϕ is satisfiable in a finite N -bounded tree structure of size bounded by $|\phi|N^{|\phi|}$ and each branch is of length $md(\phi)$.*

In order to establish (iii), the details are omitted but we apply the naive model-checking algorithm for MSO with an ATM: existential (resp. universal) quantification $\exists p$ (resp. $\forall p$) requires time $\mathcal{O}(|\mathfrak{T}|)$ and the machine enters in a sequence of existential (resp. universal) states. The quantifier-free formula ψ is evaluated as a first-order formula by the standard translation for modal logic. Note also that checking $\mathfrak{T}', v \models \psi$ can be done in polynomial time in $|\psi| + |\mathfrak{T}'|$ (see [36], [51]). All is in place to establish our improved upper bound.

Theorem 9. *For $N \geq 1$, $\text{SAT}(\text{QCTL}_{\mathbf{X}, \leq N}^t) \in \text{AEXP}_{\text{POL}}$.*

When $N = 1$, the upper bound can be improved as the number of alternations is linear and the size of the finite witness “tree” is polynomial in $|\phi|$, and therefore the whole procedure can be implemented with a polynomial-time ATM (thus in PSPACE [43]). The matching lower bound is inherited from the problem QBF.

Corollary 10. *$\text{SAT}(\text{QCTL}_{\mathbf{X}, \leq 1}^t)$ is PSPACE-complete.*

C. A reduction from AMTP (with fixed $N \geq 2$)

In order to show that the problem is AEXP_{POL}-hard, we define below a reduction from AMTP presented in Section II and introduced in [31]. To define a grid $[0, 2^n - 1] \times [0, 2^n -$

1], a major part in the solution of an instance of **AMTP**, we specify a tree such that every node at distance less than $2n$ from the root ε has exactly two children (so there are exactly 2^{2n} nodes at distance $2n$ from ε) and each node at distance $2n$ encodes a position $(\mathfrak{H}, \mathfrak{V})$ in $[0, 2^n - 1] \times [0, 2^n - 1]$, thanks to the use of the propositional variables $\mathbf{h}_{n-1}, \dots, \mathbf{h}_0$ and $\mathbf{v}_{n-1}, \dots, \mathbf{v}_0$. The i th bit of \mathfrak{H} (resp. \mathfrak{V}) is taken care by the truth value of \mathbf{h}_i (resp. \mathbf{v}_i) and the least significant bit is encoded by $\mathbf{h}_0/\mathbf{v}_0$. The formula $\text{grid}(2n)$ below is dedicated to encode this grid.

$$\begin{aligned} & \left(\bigwedge_{i \in [0, 2n-1]} \mathbf{AX}^i \mathbf{EX}_{=2} \top \right) \wedge \forall x, y \text{ diff-nom}(x, y, 2n) \rightarrow \\ & \left(\bigvee_{j \in [0, n-1]} \neg(\textcircled{x}^{2n} \mathbf{h}_j \leftrightarrow \textcircled{y}^{2n} \mathbf{h}_j) \vee \neg(\textcircled{x}^{2n} \mathbf{v}_j \leftrightarrow \textcircled{y}^{2n} \mathbf{v}_j) \right), \end{aligned}$$

where $\mathbf{EX}_{=2} \top = \exists x_1, x_2 \text{ diff-nom}(x_1, x_2, 1) \wedge \mathbf{AX}(x_1 \vee x_2)$ and states that there are exactly two children. Moreover, $\mathbf{AX}^0 \psi \stackrel{\text{def}}{=} \psi$ and $\mathbf{AX}^{i+1} \psi \stackrel{\text{def}}{=} \mathbf{AX} \mathbf{AX}^i \psi$. As there are exactly 2^{2n} descendants at distance $2n$ from the root, if two such distinct descendants differ by at least one propositional variable from $\mathbf{h}_{n-1}, \dots, \mathbf{h}_0, \mathbf{v}_{n-1}, \dots, \mathbf{v}_0$ (exactly what is expressed by the second conjunct of $\text{grid}(2n)$), then the full grid $[0, 2^n - 1] \times [0, 2^n - 1]$ is encoded. The correctness of $\text{grid}(2n)$ obviously relies on Lemma 3.

Let $\langle \mathcal{T}, \mathcal{H}, \mathcal{V} \rangle$ be a triple from an instance of **AMTP** and $j \in \mathbb{N}$. Each tile type $t \in \mathcal{T}$ will be represented by a fresh propositional variable t^j . So, $\{t^j : t \in \mathcal{T}\}$ (written below \mathcal{T}^j) is understood as the j th copy of \mathcal{T} . First, let us define the formulae ϕ_{cov}^j , $\phi_{\mathcal{H}}^j$ and $\phi_{\mathcal{V}}^j$ whose conjunction states that every position of the grid has a unique tile type in \mathcal{T}^j , and the horizontal and vertical matching conditions are satisfied. The formula ϕ_{cov}^j expresses that every position of the grid has a unique tile type:

$$\phi_{\text{cov}}^j \stackrel{\text{def}}{=} \forall x \text{ nom}(x, 2n) \rightarrow \textcircled{x}^{2n} \left(\bigvee_{t \in \mathcal{T}} t^j \wedge \bigwedge_{t' \neq t \in \mathcal{T}} \neg(t^j \wedge t'^j) \right).$$

For the horizontal matching constraints, we need to express when two nodes at distance $2n$ interpreted respectively by x and y and representing respectively the position $(\mathfrak{H}, \mathfrak{V})$ and $(\mathfrak{H}', \mathfrak{V}')$, satisfy $\mathfrak{V} = \mathfrak{V}'$ and $\mathfrak{H}' = \mathfrak{H} + 1$. The formula $\text{HN}(x, y)$ ('HN' for 'horizontal neighbours') does the job using standard arithmetical reasoning on numbers encoded in binary.

$$\begin{aligned} & \bigvee_{i \in [0, n-1]} (\textcircled{x}^{2n} \neg \mathbf{h}_i \wedge \textcircled{y}^{2n} \mathbf{h}_i \wedge \left(\bigwedge_{\alpha \in [0, i-1]} \textcircled{x}^{2n} \mathbf{h}_\alpha \wedge \textcircled{y}^{2n} \neg \mathbf{h}_\alpha \right) \wedge \\ & \left(\bigwedge_{\alpha \in [i+1, n]} (\textcircled{x}^{2n} \mathbf{h}_\alpha \leftrightarrow \textcircled{y}^{2n} \mathbf{h}_\alpha) \right) \wedge \left(\bigwedge_{\alpha \in [0, n-1]} \textcircled{x}^{2n} \mathbf{v}_\alpha \leftrightarrow \textcircled{y}^{2n} \mathbf{v}_\alpha \right). \end{aligned}$$

$\phi_{\mathcal{H}}^j$ below encodes horizontal matching constraints:

$$\forall x, y (\text{nom}(x, 2n) \wedge \text{nom}(y, 2n) \wedge \text{HN}(x, y)) \rightarrow \bigvee_{(t, t') \in \mathcal{H}} \textcircled{x}^{2n} t^j \wedge \textcircled{y}^{2n} t'^j.$$

Let $\text{VN}(x, y)$ be the formula obtained from $\text{HN}(x, y)$ by replacing each occurrence of \mathbf{h}_α (resp. \mathbf{v}_α) by \mathbf{v}_α (resp. \mathbf{h}_α). The following formula $\phi_{\mathcal{V}}^j$ encodes the vertical matching

constraints: $\forall x, y (\text{nom}(x, 2n) \wedge \text{nom}(y, 2n) \wedge \text{VN}(x, y)) \rightarrow \bigvee_{(t, t') \in \mathcal{V}} \textcircled{x}^{2n} t^j \wedge \textcircled{y}^{2n} t'^j$. To state that a root node satisfying $\text{grid}(2n)$ encodes a tiling with respect to \mathcal{T}^j , we consider $\phi_{\text{tiling}}^j \stackrel{\text{def}}{=} \phi_{\text{cov}}^j \wedge \phi_{\mathcal{H}}^j \wedge \phi_{\mathcal{V}}^j$. To encode an instance of **AMTP**, there are still properties that need to be expressed. Let us assume that the root node ε satisfies $\text{grid}(2n)$.

- Given $\mathcal{T}_0 \subseteq \mathcal{T}$, for each position of the first row of the grid, exactly one tile type in \mathcal{T}_0^j holds.

$$\begin{aligned} \phi_{\text{init}}^j & \stackrel{\text{def}}{=} \forall x (\text{nom}(x, 2n) \wedge \textcircled{x}^{2n} \left(\bigwedge_{\alpha \in [0, n-1]} \neg \mathbf{h}_\alpha \right) \rightarrow \\ & \textcircled{x}^{2n} \left(\bigvee_{t \in \mathcal{T}_0} t^j \wedge \bigwedge_{t' \neq t \in \mathcal{T}_0} \neg(t^j \wedge t'^j) \right). \end{aligned}$$

- Assuming that ε satisfies $\phi_{\text{tiling}}^j \wedge \phi_{\text{init}}^j$, for each position of the first row of the grid, the tile type in \mathcal{T}_0^j coincides with the tile type in $\mathcal{T}_0^{j'}$ (corresponding to (m-init)):

$$\begin{aligned} \phi_{\text{coinci}}^{j, j'} & \stackrel{\text{def}}{=} \forall x (\text{nom}(x, 2n) \wedge \textcircled{x}^{2n} \left(\bigwedge_{\alpha \in [0, n-1]} \neg \mathbf{h}_\alpha \right) \rightarrow \\ & \textcircled{x}^{2n} \left(\bigvee_{t \in \mathcal{T}_0} t^j \wedge t'^{j'} \right). \end{aligned}$$

- Given $\mathcal{T}_{\text{acc}} \subseteq \mathcal{T}$ and assuming that ε satisfies ϕ_{tiling}^j , there is a position on the last row with a tile type in \mathcal{T}_{acc} (corresponding to (m-accept)):

$$\phi_{\text{acc}}^j \stackrel{\text{def}}{=} \exists x \text{ nom}(x, 2n) \wedge \textcircled{x}^{2n} \left(\left(\bigwedge_{\alpha \in [0, n-1]} \mathbf{h}_\alpha \right) \wedge \bigvee_{t \in \mathcal{T}_{\text{acc}}} t^j \right).$$

- Given $\mathcal{T}_{\text{multi}} \subseteq \mathcal{T} \times \mathcal{T}$, and assuming ε satisfies $\phi_{\text{tiling}}^j \wedge \phi_{\text{tiling}}^{j+1}$, on every position, the tile type in \mathcal{T}^j and the tile type in \mathcal{T}^{j+1} are in the relation $\mathcal{T}_{\text{multi}}$ (corresponding to (m-multi)): $\phi_{\text{multi}}^j \stackrel{\text{def}}{=} \forall x \text{ nom}(x, 2n) \rightarrow \textcircled{x}^{2n} \left(\bigvee_{(t, t') \in \mathcal{T}_{\text{multi}}} t^j \wedge t'^{j+1} \right)$.

Given a set of propositional variables $X = \{r_1, \dots, r_\beta\}$, we write $\exists X \psi$ to denote $\exists r_1 \exists r_2 \dots \exists r_\beta \psi$. $\forall X \psi$ is defined similarly. Given an instance \mathcal{I} of **AMTP** made of n , $\langle \mathcal{T}, \mathcal{H}, \mathcal{V} \rangle$, \mathcal{T}_0 , \mathcal{T}_{acc} , $\mathcal{T}_{\text{multi}}$, let us define $\phi_{\mathcal{I}}$ below:

$$\begin{aligned} & \text{grid}(2n) \wedge \forall \mathcal{T}_0^1 \exists \mathcal{T}_0^2 \forall \mathcal{T}_0^3 \dots \exists \mathcal{T}_0^n \bigwedge_{j \in [1, n]} \phi_{\text{init}}^j \rightarrow \\ & (\exists \{t^j : t \in \mathcal{T}, j \in [n+1, 2n]\}) \left(\bigwedge_{j \in [n+1, 2n]} \phi_{\text{tiling}}^j \wedge \phi_{\text{coinci}}^{j, (j-n)} \right) \wedge \\ & \left(\bigwedge_{j \in [n+1, 2n-1]} \phi_{\text{multi}}^j \wedge \phi_{\text{acc}}^{2n} \right). \end{aligned}$$

Now, we can state the correctness of the reduction.

Lemma 11. \mathcal{I} is a positive instance of **AMTP** iff $\phi_{\mathcal{I}}$ is satisfiable in $\text{QCTL}_{\mathbf{X}, \leq N}^t$.

The proof is a bit tedious but has no serious difficulties, as all the conditions for being a solution of \mathcal{I} can be easily expressed, as soon as the grid $[0, 2^n - 1] \times [0, 2^n - 1]$ is encoded. Moreover, the quantifications involved in **AMTP** are straightforwardly taking care of in $\text{QCTL}_{\mathbf{X}, \leq N}^t$ thanks to the presence of propositional quantification. This leads us to one of the main results of the paper.

Theorem 12. For all $N \geq 2$, $\text{SAT}(\text{QCTL}_{\mathbf{X}, \leq N}^t)$ is AEXP_{POL} -hard.

IV. TOWER-HARDNESS OF SAT(QCTL_X^t)

We are back to the (general) problem SAT(QCTL_X^t).

A. Overview of the method

In order to show TOWER-hardness, we shall reduce the k -NEXPTIME-complete tiling problem **Tiling_k** introduced in Section II-C to SAT(QCTL_X^t) and this should be done in a uniform way so that TOWER-hardness can be concluded (see the discussion in [15, Section 3.1.2]). Hence, we need to encode concisely a grid $t(k, n) \times t(k, n)$ and to do so, the main task consists in enforcing that a node has $t(k, n)$ children with a formula of elementary size in $k+n$ (bounded by a tower of exponentials of fixed height). Actually, our method produces a formula of exponential size in $k+n$. Of course, this is not the end of the story as we need to encode the grid $t(k, n) \times t(k, n)$ and to express on it constraints about the tiling τ . First, let us explain how to enforce that a node has exactly $t(k, n)$ children, by partly taking advantage of the proof technique of local nominals (see Section III-A).

For the forthcoming subsections, we assume that n is fixed. Below, we classify the nodes of a tree model by their *type* (a value in \mathbb{N}) such that any node is of type 0, and if a node is of type $k > 0$, then it has exactly $t(k, n)$ children and all the children are of type $k-1$. To be more precise, a node may have two types (as one of them is always zero). So, a node of type 1 has exactly 2^n children. As every node is of type 0 and for $k > 0$, being of type k entails a number of children equal to $t(k, n)$, if a node is of type $k > 0$, this value k is unique. Additional conditions apply for being of type $k > 0$ but we can already notice that a node v of type k implicitly defines a balanced subtree of depth k with root v . In order to enforce that a node is of type $k \geq 1$, and therefore has exactly $t(k, n)$ children, with each node v of type $k \geq 0$ is associated a *number* in $[0, t(k+1, n) - 1]$. Such a number is written $\mathbf{nb}_{\mathfrak{T}}(v)$. The subscript ‘ \mathfrak{T} ’ may be omitted when the context is clear. When the type of the node v is zero, its number is defined as the unique m such that the number represented by the truth values of $p_{n-1}, p_{n-2}, \dots, p_0$ is equal to m . As usual, the propositional variable p_i is responsible for the i th bit of m and by convention, the least significant bit is encoded by the truth value of p_0 . Otherwise, when the type of v is equal to some $k > 0$, its number is represented by the binary encoding of the propositional variable val on its children assuming that there are $t(k, n)$ children whose respective (bit) numbers span all over $[0, t(k, n) - 1]$ and therefore all the children are implicitly ordered.

This principle makes sense conceptually but it remains to express it in QCTL_X^t. That is why, in the table below, we present a list of formulae to be defined. All of them are interpreted on a node v_0 of level $k \geq 0$, $1 \leq d \leq k$, and \widehat{x}, \widehat{y} denotes $\mathbf{nom}(x_1, 1) \wedge \mathbf{nom}(y_1, 1) \wedge \bigwedge_{i \in [2, d]} (\@_{x_1}^1 \@_{x_2}^1 \dots \@_{x_{i-1}}^1 \mathbf{nom}(x_i, 1) \wedge \@_{y_1}^1 \@_{y_2}^1 \dots \@_{y_{i-1}}^1 \mathbf{nom}(y_i, 1))$. Assuming that the nodes v_1, \dots, v_d are associated with $\bar{x} = x_1, \dots, x_d$ (resp.

v'_1, \dots, v'_d are associated with $\bar{y} = y_1, \dots, y_d$), v_0, v_1, \dots, v_d and v_0, v'_1, \dots, v'_d can be understood as two branches rooted at v_0 ending at v_d and at v'_d respectively. The formula \widehat{x}, \widehat{y} uses subformulae introduced in Section III-A and the wide hat symbol in \widehat{x}, \widehat{y} is a graphical reminder of these two branches. By contrast, \widehat{x}, \bar{x} states the existence of a single branch with nodes v_0, \dots, v_d .

Formulae to be defined	Intuitive meaning
$\text{type}(k)$	v_0 is of type k
$\text{first}(k)$	$\mathbf{nb}(v_0) = 0$
$\text{last}(k)$	$\mathbf{nb}(v_0) = t(k+1, n) - 1$
$\text{uniq}(k)$	$\forall v, v' ((v_0 E v) \ \& \ (v_0 E v') \ \& \ v \neq v') \rightarrow \mathbf{nb}(v) \neq \mathbf{nb}(v')$
$\text{compl}(k)$	$\forall v ((v_0 E v) \ \& \ \mathbf{nb}(v) < t(k, n) - 1) \rightarrow \exists v' (v_0 E v') \ \wedge \ \mathbf{nb}(v') = \mathbf{nb}(v) + 1$
$\mathbf{nb}(\bar{x}) \stackrel{k}{=} \mathbf{nb}(\bar{y})$	$\mathbf{nb}(v_d) = \mathbf{nb}(v'_d)$
$\mathbf{nb}(\bar{y}) \stackrel{k}{=} \mathbf{nb}(\bar{x}) + 1$	$\mathbf{nb}(v'_d) = 1 + \mathbf{nb}(v_d)$
$\mathbf{nb}(\bar{x}) \stackrel{k}{<} \mathbf{nb}(\bar{y})$	$\mathbf{nb}(v_d) < \mathbf{nb}(v'_d)$

In the last 3 lines of the table, the superscript ‘ k ’ above ‘ $=$ ’ and ‘ $<$ ’ allows us to remember that the formula is evaluated at a node of type k ; as \bar{x} and \bar{y} are of length d , the number comparison is done on nodes of type $k-d$.

In order to define $\text{type}(k)$ ($k \geq 1$), we specify that every child is of type $k-1$, there is a child with number equal to zero, and if a child has number $m < t(k, n) - 1$, then there is a child with number equal to $m+1$. Moreover, two distinct children have distinct numbers in $[0, t(k, n) - 1]$. Satisfying these conditions guarantees that the number for the children span all over $[0, t(k, n) - 1]$. The formula $\text{type}(k)$ ($k \geq 1$) is defined as $\mathbf{AX}(\text{type}(k-1)) \wedge \mathbf{EX}(\text{first}(k-1)) \wedge \text{uniq}(k) \wedge \text{compl}(k)$. Note that the formula is built over the propositional variables p_0, \dots, p_{n-1}, val (only). Let us explain how we shall proceed below to define all these formulae. For successive values $N \in \mathbb{N}$, we define inductively the formulae $\text{type}(N)$, $\text{first}(N)$ and $\text{last}(N)$, and $\mathbf{nb}(x_1, \dots, x_d) \stackrel{k}{=} \mathbf{nb}(y_1, \dots, y_d)$, $\mathbf{nb}(x_1, \dots, x_d) \stackrel{k}{<} \mathbf{nb}(y_1, \dots, y_d)$ and $\mathbf{nb}(y_1, \dots, y_d) \stackrel{k}{=} \mathbf{nb}(x_1, \dots, x_d) + 1$ for $k-d = N-1$. For $N=0$, only the formulae $\text{type}(0)$, $\text{first}(0)$ and $\text{last}(0)$ make sense. The case $N=1$ is not yet an instance of the general case. We first treat the cases for $N \in \{0, 1\}$ and then the general case with $N \geq 2$.

B. Formulae for $N \in \{0, 1\}$

When $k=0$ (i.e. $k=N$ and $N=0$), only the intended properties for the formulae $\text{type}(0)$, $\text{first}(0)$ and $\text{last}(0)$ are meaningful: $\text{type}(0) \stackrel{\text{def}}{=} \top$, $\text{first}(0) \stackrel{\text{def}}{=} \neg p_{n-1} \wedge \dots \wedge \neg p_0$ and $\text{last}(0) \stackrel{\text{def}}{=} p_{n-1} \wedge \dots \wedge p_0$. One can check that $\mathfrak{T}, v \models \text{type}(0)$ iff v is of type 0, and assuming that v is understood as a node of type 0, we have $\mathfrak{T}, v \models \text{first}(0)$ iff $\mathbf{nb}_{\mathfrak{T}}(v) = 0$ and $\mathfrak{T}, v \models \text{last}(0)$ iff $\mathbf{nb}_{\mathfrak{T}}(v) = t(1, n) - 1 = 2^n - 1$.

Let us move to the case $N=1$ and we define the formulae $\text{type}(k)$, $\text{first}(k)$ and $\text{last}(k)$ with $k=1$ (i.e. $k=N=1$) as well as $\mathbf{nb}(x_1, \dots, x_d) \stackrel{k}{<} \mathbf{nb}(y_1, \dots, y_d)$ and $\mathbf{nb}(y_1, \dots, y_d) \stackrel{k}{=} \mathbf{nb}(x_1, \dots, x_d) + 1$ with $k-d=0$ (i.e.

$k - d = N - 1$ with $N = 1$). So, k and d can be arbitrarily large as soon as $k = d$. The formula $\text{nb}(x_1, \dots, x_d) \stackrel{k}{=} \text{nb}(y_1, \dots, y_d)$ needs also to be defined and is simply equal to $\neg(\text{nb}(x_1, \dots, x_d) \stackrel{k}{<} \text{nb}(y_1, \dots, y_d)) \wedge \neg(\text{nb}(y_1, \dots, y_d) \stackrel{k}{<} \text{nb}(x_1, \dots, x_d))$.

The formula $\text{type}(1)$ defined below states that there is a child with number equal to zero, if a child has number $m < 2^n - 1$, then there is a child with number equal to $m + 1$, all the children are of type 0, and two distinct children have distinct numbers in $[0, 2^n - 1]$. Remember that the number of each child (of type 0) is computed from the propositional variables in $\{p_{n-1}, \dots, p_0\}$. The arithmetical reasoning between children, leading to the fact that there are exactly 2^n children whose numbers span all over $[0, 2^n - 1]$ takes advantage of standard arithmetical properties on numbers encoded by n bits. Here is the formula $\text{type}(1)$: $\mathbf{AX}(\text{type}(0)) \wedge \mathbf{EX}(\text{first}(0)) \wedge \text{uniq}(1) \wedge \text{compl}(1)$. It remains to specify what are $\text{uniq}(1)$ and $\text{compl}(1)$. In order to define $\text{uniq}(1)$, we simply state that there are no two distinct children (of type 0) with the same number: $\forall x, y \text{ diff-nom}(x, y, 1) \rightarrow \neg(\text{nb}(x) \stackrel{1}{=} \text{nb}(y))$. The formula $\text{diff-nom}(x, y, 1)$ guarantees that we pick two distinct children and the nominals x and y allow to access to them (and check the values of the propositional variables in $\{p_{n-1}, \dots, p_0\}$). The formula $\text{compl}(1)$ below states that for each child v (of type 0) that is not the last one, there is also a child v' (of type 0) such that $\text{nb}_{\mathfrak{T}}(v') = \text{nb}_{\mathfrak{T}}(v) + 1$. Here is the formula $\text{compl}(1)$: $\forall x (\text{nom}(x, 1) \wedge @_{\bar{x}}(\neg \text{last}(0))) \rightarrow \exists y \text{ nom}(y, 1) \wedge \text{nb}(y) \stackrel{1}{=} \text{nb}(x) + 1$. Finally, it remains to define the formulae $\text{nb}(x) \stackrel{1}{=} \text{nb}(y)$ and $\text{nb}(y) \stackrel{1}{=} \text{nb}(x) + 1$ used respectively in $\text{uniq}(1)$ and in $\text{compl}(1)$. Below, we treat the more general situation with $k = d$ (k is not necessarily equal to 1), and $\text{nb}(x) \stackrel{1}{=} \text{nb}(y)$ and $\text{nb}(y) \stackrel{1}{=} \text{nb}(x) + 1$ are specific instances with $k = d = 1$. Let assume that $\bar{x} = x_1, \dots, x_k$ and $\bar{y} = y_1, \dots, y_k$ (so $d = k$).

- $\text{nb}(y_1, \dots, y_k) \stackrel{k}{=} \text{nb}(x_1, \dots, x_k) + 1$ is defined as

$$\bigvee_{i=0}^{n-1} @_{\bar{x}}(\neg p_i \wedge \bigwedge_{j=0}^{i-1} p_j) \wedge @_{\bar{y}}(\bigwedge_{j=0}^{i-1} \neg p_j \wedge p_i) \wedge (\bigwedge_{j=i+1}^{n-1} @_{\bar{x}} p_j \Leftrightarrow @_{\bar{y}} p_j).$$

- $\text{nb}(x_1, \dots, x_k) \stackrel{k}{<} \text{nb}(y_1, \dots, y_k)$ is defined as

$$\bigvee_{i=0}^{n-1} @_{\bar{y}} p_i \wedge @_{\bar{x}} \neg p_i \wedge (\bigwedge_{j=i+1}^{n-1} @_{\bar{x}} p_j \Leftrightarrow @_{\bar{y}} p_j).$$

The above definitions use standard properties about numbers encoded in binary representation with n bits (the least significant bit is represented by the truth value of p_0). For the sake of completeness, let us define $\text{first}(1)$, $\text{last}(1)$ as follows: $\text{first}(1) \stackrel{\text{def}}{=} \mathbf{AX}(\neg \text{val})$, $\text{last}(1) \stackrel{\text{def}}{=} \mathbf{AX}(\text{val})$.

Here is a preliminary lemma that states that we have properly proceeded for the binary encoding of numbers with the propositional variables in p_{n-1}, \dots, p_0 (and Lemma 4 needs to be used).

Lemma 13. *Let \mathfrak{T} be a tree model and v be one of its nodes such that v satisfies \bar{x}, \bar{y} (\bar{x} and \bar{y} are both of length k) and, v_k and v'_k are understood as nodes of type 0.*

- (I) $\mathfrak{T}, v \models \text{nb}(\bar{y}) \stackrel{k}{=} \text{nb}(\bar{x}) + 1$ iff $\text{nb}_{\mathfrak{T}}(v'_k) = 1 + \text{nb}_{\mathfrak{T}}(v_k)$.
- (II) $\mathfrak{T}, v \models \text{nb}(\bar{x}) \stackrel{k}{<} \text{nb}(\bar{y})$ iff $\text{nb}_{\mathfrak{T}}(v_k) < \text{nb}_{\mathfrak{T}}(v'_k)$.
- (III) $\mathfrak{T}, v \models \text{nb}(\bar{x}) \stackrel{k}{=} \text{nb}(\bar{y})$ iff $\text{nb}_{\mathfrak{T}}(v_k) = \text{nb}_{\mathfrak{T}}(v'_k)$.

Here is the main lemma of this subsection.

Lemma 14. *Let \mathfrak{T} be a tree model and v be one of its nodes. (I) $\mathfrak{T}, v \models \text{type}(1)$ iff v is of type 1, (II) Assuming v satisfies $\text{type}(1)$, we have $\mathfrak{T}, v \models \text{first}(1)$ (resp. $\mathfrak{T}, v \models \text{last}(1)$) iff $\text{nb}_{\mathfrak{T}}(v) = 0$ (resp. $\text{nb}_{\mathfrak{T}}(v) = \mathfrak{t}(2, n) - 1$).*

C. Formulae with arbitrary $N \geq 2$

Let us consider the arbitrary case $N \geq 2$. Below, we define the formulae $\text{type}(N)$, $\text{first}(N)$ and $\text{last}(N)$ as well as $\text{nb}(x_1, \dots, x_d) \stackrel{k}{<} \text{nb}(y_1, \dots, y_d)$, and $\text{nb}(y_1, \dots, y_d) \stackrel{k}{=} \text{nb}(x_1, \dots, x_d) + 1$ with $k - d = N - 1$. The formula $\text{nb}(x_1, \dots, x_d) \stackrel{k}{=} \text{nb}(y_1, \dots, y_d)$ is also defined but does not require much developments as by definition, it is equal to $\neg(\text{nb}(x_1, \dots, x_d) \stackrel{k}{<} \text{nb}(y_1, \dots, y_d)) \wedge \neg(\text{nb}(y_1, \dots, y_d) \stackrel{k}{<} \text{nb}(x_1, \dots, x_d))$. We assume that for all $k < N$, the formulae $\text{type}(k)$, $\text{last}(k)$ and $\text{first}(k)$ are already defined and for $k - d \leq N - 2$, the formulae $\text{nb}(\bar{x}) \stackrel{k}{<} \text{nb}(\bar{y})$, $\text{nb}(\bar{x}) \stackrel{k}{=} \text{nb}(\bar{y})$ and $\text{nb}(\bar{y}) \stackrel{k}{=} \text{nb}(\bar{x}) + 1$ are already defined too (\bar{x} and \bar{y} are of length d). This can be understood as an implicit induction hypothesis when proving the correctness of the formulae built for N .

As for the case $N = 1$, the formula $\text{type}(N)$ follows the general schema: $\text{type}(N) \stackrel{\text{def}}{=} \mathbf{AX}(\text{type}(N - 1)) \wedge \mathbf{EX}(\text{first}(N - 1)) \wedge \text{uniq}(N) \wedge \text{compl}(N)$. It remains to specify what are $\text{uniq}(N)$ and $\text{compl}(N)$. In order to define $\text{uniq}(N)$, we simply state that there are no two distinct children (of type $N - 1$) with the same number:

$$\text{uniq}(N) \stackrel{\text{def}}{=} \forall x, y \text{ diff-nom}(x, y, 1) \rightarrow \neg(\text{nb}(x) \stackrel{N}{=} \text{nb}(y)).$$

Again, $\text{diff-nom}(x, y, 1)$ allows us to select two distinct children (of type $N - 1$). The formula $\text{compl}(N)$ below states that for each child v (of type $N - 1$) that is not the last one, there is also a child v' (of type $N - 1$ too) such that $\text{nb}_{\mathfrak{T}}(v') = \text{nb}_{\mathfrak{T}}(v) + 1$. Here is the formula $\text{compl}(N)$:

$$\forall x (\text{nom}(x, 1) \wedge @_{\bar{x}}(\neg \text{last}(N - 1))) \rightarrow \exists y \text{ nom}(y, 1) \wedge \text{nb}(y) \stackrel{N}{=} \text{nb}(x) + 1.$$

It remains to define $\text{nb}(x) \stackrel{N}{=} \text{nb}(y)$ and $\text{nb}(y) \stackrel{N}{=} \text{nb}(x) + 1$ used respectively in $\text{uniq}(N)$ and in $\text{compl}(N)$. This time this requires much lengthier developments, apart from using the properties of the formulae constructed for $N - 1$ and for smaller values (implicit induction hypothesis). Below, we treat the more general situation with $k - d = N - 1$, and $\text{nb}(x) \stackrel{N}{=} \text{nb}(y)$ and $\text{nb}(y) \stackrel{N}{=} \text{nb}(x) + 1$ are just particular instances with $k = N$ and $d = 1$. So, let assume that $\bar{x} = x_1, \dots, x_d$ and $\bar{y} = y_1, \dots, y_d$.

For defining $\text{nb}(y_1, \dots, y_k) \stackrel{k}{=} \text{nb}(x_1, \dots, x_k) + 1$ (see Section IV-B), we have compared the respective truth values of p_{n-1}, \dots, p_0 for the node v_k (interpretation of x_k) and for the node v'_k (interpretation of y_k). The same principle applied for defining $\text{nb}(x_1, \dots, x_k) \stackrel{k}{=} \text{nb}(y_1, \dots, y_k)$. Typically, $\text{nb}(y_1, \dots, y_k) \stackrel{k}{=} \text{nb}(x_1, \dots, x_k) + 1$ holds iff there is $i \in [0, n-1]$, such that

- for every $j \in [i+1, n-1]$, v_k and v'_k agree on p_j ,
- v_k does not satisfy p_i and v'_k satisfies p_i ,
- for $j \in [0, i-1]$, v_k satisfies p_j , v'_k does not satisfy p_j .

So, we needed to define a partition $\{[0, i-1], \{i\}, [i+1, n-1]\}$ of $[0, n-1]$. The same principle applies when the bit numbers are among $[0, \mathfrak{t}(k-d, n) - 1]$ to encode a value in $[0, \mathfrak{t}(k-d+1, n) - 1]$. This needs to be done concisely as we cannot go through all the $\mathfrak{t}(k-d, n)$ bit numbers because the reduction has to be of elementary complexity.

Now, when attempting to define $\text{nb}(y_1, \dots, y_d) \stackrel{k}{=} \text{nb}(x_1, \dots, x_d) + 1$, the nodes v_d and v'_d are of type $k-d > 0$ with $\mathfrak{t}(k-d, n)$ children each. The truth values of val on their respective children determine precisely the numbers $\text{nb}(v_d)$ and $\text{nb}(v'_d)$.

Let $u_0, \dots, u_{\mathfrak{t}(k-d, n)-1}$ be the children of v_k such that $\text{nb}(u_j) = j$ for all j . Similarly, let $u'_0, \dots, u'_{\mathfrak{t}(k-d, n)-1}$ be the children of v'_k such that $\text{nb}(u'_j) = j$ for all j . So, $\text{nb}(y_1, \dots, y_d) \stackrel{k}{=} \text{nb}(x_1, \dots, x_d) + 1$ holds iff there is $i \in [0, \mathfrak{t}(k-d, n) - 1]$, such that

- for $j \in [i+1, \mathfrak{t}(k-d, n) - 1]$, u_j and u'_j agree on val ,
- u_i does not satisfy val and u'_i satisfies val ,
- for every $j \in [0, i-1]$, u_j satisfies val and u'_j does not satisfy val .

We have a partition $\{u_0, \dots, u_{i-1}\}, \{u_i\}, \{u_{i+1}, \dots, u_{\mathfrak{t}(k-d, n)-1}\}$ of $\{u_0, \dots, u_{\mathfrak{t}(k-d, n)-1}\}$ (and similarly for the children of v'_d). To do so, we use an existential quantification on the fresh propositional variables l (left), s (selected bit), r (right) such that

- for every child of v_d (resp. v'_d), exactly one propositional variable among $\{l, s, r\}$ holds true,
- exactly one child of v_d (resp. v'_d) satisfies s ,
- if v is a child of v_d satisfying l (resp. s) and v' is child of v_d satisfying s (resp. r), then $\text{nb}_{\bar{x}}(v) < \text{nb}_{\bar{x}}(v')$. The same condition holds with v'_d .

Additional arithmetical constraints are needed to relate the partition of \bar{x} with the partition of \bar{y} (see below the details) but in a way, it is independent of the partition itself. For instance, the unique child of v_d satisfying s and the unique child of v'_d satisfying s should have the same (bit) number. Nevertheless, it is clear that we need, at least, to be able to state in $\text{QCTL}_{\bar{x}}^t$ the existence of a partition satisfying (a), (b) and (c). In the sequel, such partitions are called *lsr-partitions*. The formula $\text{LSR}_{\bar{x}}(k)$ below does the job for v_d (then use $\text{LSR}_{\bar{y}}(k)$ for v'_d).

So, let $\bar{x} = x_1, \dots, x_d$. In the context of the definition of $\text{LSR}_{\bar{x}}(k)$, we allow the limit case $d = 0$, with empty sequence ϵ , assuming that $@_{\epsilon}\psi \stackrel{\text{def}}{=} \psi$ and $\widehat{\epsilon, \epsilon} \stackrel{\text{def}}{=} \top$.

Below $0 \leq d < k$ and the formula $\text{LSR}_{\bar{x}}(k)$ is defined as the conjunction $\text{LSR}_{\bar{x}}^1(k) \wedge \text{LSR}_{\bar{x}}^2(k) \wedge \text{LSR}_{\bar{x}}^3(k)$ and is interpreted on a node v_0 of type k satisfying $\widehat{\bar{x}, \bar{x}}$, and therefore this satisfaction is witnessed by the branch v_0, \dots, v_d (notations for developments below).

First, $\text{LSR}_{\bar{x}}^2(k) \stackrel{\text{def}}{=} @_{\bar{x}}(\mathbf{EX}_{=1}(s))$, with $\mathbf{EX}_{=1}(\psi)$ defined as $\mathbf{EX}\psi \wedge \neg \exists p (\mathbf{EX}(\psi \wedge p) \wedge \mathbf{EX}(\psi \wedge \neg p))$ assuming that p does not occur in ψ . The formula $\text{LSR}_{\bar{x}}^2(N)$ simply states that there is a unique child of v_d satisfying s .

Let $\text{LSR}_{\bar{x}}^1(k)$ be defined below, stating that for every child of v_d , exactly one variable among $\{l, s, r\}$ holds true:

$$@_{\bar{x}}(\mathbf{AX}((s \vee l \vee r) \wedge \neg(s \wedge l) \wedge \neg(s \wedge r) \wedge \neg(l \wedge r))).$$

Finally, $\text{LSR}_{\bar{x}}^3(k)$ is defined as follows.

$$@_{\bar{x}}(\forall w \forall w' \text{diff-nom}(w, w', 1) \wedge ((@_w^1(s) \wedge @_{w'}^1(r)) \vee (@_w^1(l) \wedge @_{w'}^1(s))) \rightarrow \text{nb}(w') \stackrel{k-d}{<} \text{nb}(w)).$$

The formula $\text{LSR}_{\bar{x}}^3(k)$ states if v is a child of v_d satisfying l (resp. s) and v' is another child of v_d satisfying s (resp. r), then $\text{nb}_{\bar{x}}(v) < \text{nb}_{\bar{x}}(v')$. The nodes v and v' are obviously of type $k-d-1$ and their respective numbers belong to $[0, \mathfrak{t}(k-d, n) - 1]$. It is important to observe that $\text{nb}(w') \stackrel{k-d}{<} \text{nb}(w)$ is well-defined recursively as soon as $k-d \leq N-1$.

Lemma 15. *Let \mathfrak{T} be a tree model, v_0 be a node of type $k \geq 0$, and \bar{x} be a (possibly empty) sequence of nominals x_1, \dots, x_d for some $d \in [0, k-1]$ such that $k-d \leq N-1$, $\mathfrak{T}, v_0 \models \widehat{\bar{x}, \bar{x}}$, and its witness branch is v_0, \dots, v_d . Then $\mathcal{M}, v_0 \models \text{LSR}_{\bar{x}}(k)$ iff the conditions below:*

- For every child of v_d , exactly one propositional variable among $\{l, s, r\}$ holds true.
- Exactly one child of v_d satisfies s .
- If v is a child of v_d satisfying l (resp. s) and v' is a child of v_d satisfying s (resp. r), then $\text{nb}_{\bar{x}}(v) < \text{nb}_{\bar{x}}(v')$.

We come back to the question of defining formulae expressing number comparisons. The formula $\text{nb}(\bar{y}) \stackrel{k}{=} \text{nb}(\bar{x}) + 1$ (remember $k-d = N-1$) is defined as

$$\exists l, s, r \text{LSR}_{\bar{x}}(k) \wedge \text{LSR}_{\bar{y}}(k) \wedge \phi_{\text{left}}(k) \wedge \phi_{\text{select}}(k) \wedge \phi_{\text{right}}(k).$$

The conjunction $\phi_{\text{left}}(k) \wedge \phi_{\text{select}}(k) \wedge \phi_{\text{right}}(k)$ takes care of the arithmetical constraints. $\phi_{\text{select}}(k)$ states that the unique v_d 's child satisfying s (whose number is the pivot bit i) does not satisfy val , and the unique v'_d 's child satisfying s (whose number is also the pivot bit i) satisfies val : $\phi_{\text{select}}(k) \stackrel{\text{def}}{=} @_{\bar{x}}(\mathbf{AX}(s \rightarrow \neg val)) \wedge @_{\bar{y}}(\mathbf{AX}(s \rightarrow val))$. The formula $\phi_{\text{right}}(k)$ states that for all the children of v_d satisfying r (and therefore with bit number strictly smaller than i), the bit value is 1, and for all the children of v'_d satisfying r (and therefore with bit number strictly smaller than i), the bit value is 0.

$$\phi_{\text{right}}(k) \stackrel{\text{def}}{=} @_{\bar{x}}(\mathbf{AX}(r \rightarrow val)) \wedge @_{\bar{y}}(\mathbf{AX}(r \rightarrow \neg val)).$$

$\phi_{\text{left}}(k)$ states that the children of v_d satisfying l induce a set of bit numbers equal to the set of bit numbers induced

by the children of v'_d satisfying l . This entails also that the unique respective children of v_d and v'_d satisfying s have the same (bit) number. Moreover, we require that children with the same bit number satisfying l (taken from v_d and from v'_d) have the same bit value witnessed by the truth value of val . The formula $\phi_{\text{left}}(k)$ is equal to $\phi_{\text{left}}^{\bar{x}, \bar{y}}(k) \wedge \phi_{\text{left}}^{\bar{y}, \bar{x}}(k)$ with $\phi_{\text{left}}^{\bar{x}, \bar{y}}(k)$ defined below:

$$\forall w @_{\bar{x}}(\text{nom}(w, 1) \wedge @_{\bar{w}}^1(l)) \rightarrow \exists w' @_{\bar{y}}(\text{nom}(w', 1) \wedge @_{\bar{w}'}^1(l)) \wedge \text{nb}(\bar{x}, w) \stackrel{k}{=} \text{nb}(\bar{y}, w') \wedge (@_{\bar{x}, w} val \leftrightarrow @_{\bar{y}, w'} val).$$

Note that $\text{nb}(\bar{x}, w) \stackrel{k}{=} \text{nb}(\bar{y}, w')$ is well-defined as $k - (d + 1) < N - 1$. Below, we define the formula $\text{nb}(\bar{x}) < \text{nb}(\bar{y})$ with $k - d = N - 1$, $\bar{x} = x_1, \dots, x_d$ and $\bar{y} = y_1, \dots, y_d$. Based on previous developments and on standard arithmetical properties of numbers encoded in binary with $k - d$ bits, we define the formula $\text{nb}(\bar{x}) < \text{nb}(\bar{y})$ as the expression $\exists s, l, r \text{LSR}_{\bar{x}}(k) \wedge \text{LSR}_{\bar{y}}(k) \wedge \phi_{\text{left}}(k) \wedge \phi_{\text{select}}(k)$.

To conclude, $\text{first}(N) \stackrel{\text{def}}{=} \mathbf{AX}(\neg val)$ and $\text{last}(N) \stackrel{\text{def}}{=} \mathbf{AX}(val)$.

Lemma 16. *Let \mathfrak{T} be a tree model and v be one of its nodes such that v satisfies \bar{x}, \bar{y} and, v_d/v'_d are of type $k - d$.*

- (I) $\mathfrak{T}, v \models \text{nb}(\bar{y}) \stackrel{k}{=} \text{nb}(\bar{x}) + 1$ iff $\text{nb}_{\mathfrak{T}}(v'_d) = 1 + \text{nb}_{\mathfrak{T}}(v_d)$.
- (II) $\mathfrak{T}, v \models \text{nb}(\bar{x}) < \text{nb}(\bar{y})$ iff $\text{nb}_{\mathfrak{T}}(v_d) < \text{nb}_{\mathfrak{T}}(v'_d)$.
- (III) $\mathfrak{T}, v \models \text{nb}(\bar{x}) \stackrel{k}{=} \text{nb}(\bar{y})$ iff $\text{nb}_{\mathfrak{T}}(v_d) = \text{nb}_{\mathfrak{T}}(v'_d)$.

Mainly, this allows us to prove Lemma 17(I).

Lemma 17. *Let $N \geq 2$, \mathfrak{T} be a tree model and v be one of its nodes.*

- (I) $\mathfrak{T}, v \models \text{type}(N)$ iff v is of type N ,
- (II) Assuming that v satisfies $\text{type}(N)$, we have $\mathfrak{T}, v \models \text{first}(N)$ iff $\text{nb}_{\mathfrak{T}}(v) = 0$.
- (III) Assuming that v satisfies $\text{type}(N)$, we have $\mathfrak{T}, v \models \text{last}(N)$ iff $\text{nb}_{\mathfrak{T}}(v) = \text{t}(N + 1, n) - 1$.

Consequently, for all $k \geq 0$, $\text{type}(k)$, $\text{first}(k)$ and $\text{last}(k)$ characterise exactly the discussed properties, and similarly for the formulae of form $\text{nb}(\bar{y}) \stackrel{k}{=} \text{nb}(\bar{x}) + 1$, $\text{nb}(\bar{x}) < \text{nb}(\bar{y})$, $\text{nb}(\bar{x}) \stackrel{k}{=} \text{nb}(\bar{y})$ where \bar{x} and \bar{y} are of length d in $[1, k]$ and $k \geq 1$. It is natural to wonder what is the size of $\text{type}(k)$, using a reasonably succinct encoding for formulae. As the definition of $\text{type}(k)$ requires the subformulae $\text{type}(k-1)$, $\text{nb}(\bar{x}) \stackrel{k}{=} \text{nb}(\bar{y})$ and $\text{nb}(\bar{y}) \stackrel{k}{=} \text{nb}(\bar{x}) + 1$ (the other subformulae are of constant size), and $\text{type}(1)$ is quadratic in n , one can show that $\text{type}(k)$ is of size $2^{\mathcal{O}(k+n)}$.

D. Uniform reduction leading to TOWER-hardness

Let $\mathcal{P} = (\mathcal{T}, \mathcal{H}, \mathcal{V})$, $c = t_0, t_1, \dots, t_{n-1}$ be an instance of Tiling_k . Let us reduce the existence of a tiling $\tau : [0, \text{t}(k, n) - 1] \times [0, \text{t}(k, n) - 1] \rightarrow \mathcal{T}$ respecting the initial condition to the satisfiability of a formula ϕ in $\text{QCTL}_{\mathbf{X}}^t$.

To encode the grid $[0, \text{t}(k, n) - 1] \times [0, \text{t}(k, n) - 1]$, we consider a root node ε of type $k + 1$, and we distinguish $\text{t}(k, n)$ children among the totality of $\text{t}(k + 1, n)$ children.

Each child of ε has exactly $\text{t}(k, n)$ children as it is a node of type k . In order to identify the $\text{t}(k, n)$ first children of ε , we use an lsr -partition so that the unique child satisfying s has precisely the number $\text{t}(k, n)$. This guarantees that exactly the children of ε whose numbers are in $[0, \text{t}(k, n) - 1]$ satisfy r . So, the lsr -partition is used in a new context.

Below, we define $\text{nb} \stackrel{k}{=} \text{t}(k, n)$ that expresses that the number of a node of type k is $\text{t}(k, n)$ ($k \geq 1$). We recall that a node of type k takes its values in $[0, \text{t}(k + 1, n) - 1]$. Let us provide an inductive definition for $\text{nb} \stackrel{k}{=} \text{t}(k, n)$. $\text{nb} \stackrel{1}{=} \text{t}(1, n) \stackrel{\text{def}}{=} \mathbf{AX}(val \leftrightarrow X_n = n)$, where $X_n = n$ is an abbreviation for the formula stating that the truth values for p_{n-1}, \dots, p_0 encode n in binary. For $k \geq 2$, $\text{nb} \stackrel{k}{=} \text{t}(k, n) \stackrel{\text{def}}{=} \mathbf{AX}(val \leftrightarrow (\text{nb} \stackrel{k-1}{=} \text{t}(k-1, n)))$.

Lemma 18. *Assume that $\mathfrak{T}, v \models \text{type}(k)$, we have $\text{nb}_{\mathfrak{T}}(v) = \text{t}(k, n)$ iff $\mathfrak{T}, v \models \text{nb} \stackrel{k}{=} \text{t}(k, n)$.*

The formula ϕ is defined as $\text{type}(k+1) \wedge \exists l, s, r \text{LSR}_{\varepsilon}(k+1) \wedge \mathbf{EX}(s \wedge \text{nb} \stackrel{k}{=} \text{t}(k, n)) \wedge \phi_{\text{cov}} \wedge \phi_{\text{init}} \wedge \phi_{\mathcal{H}} \wedge \phi_{\mathcal{V}}$. Definitions and explanations for ϕ_{cov} , ϕ_{init} , $\phi_{\mathcal{H}}$ and $\phi_{\mathcal{V}}$ follow but observe that an lsr -partition is performed for a node of type $k + 1$ and exactly $\text{t}(k, n)$ children satisfy r thanks to $\mathbf{EX}(s \wedge \text{nb} \stackrel{k}{=} \text{t}(k, n))$. The formula ϕ_{cov} states that every position in $[0, \text{t}(k, n) - 1] \times [0, \text{t}(k, n) - 1]$ has a unique tile:

$$\forall x, y \text{nom}(x, 1) \wedge @_{\bar{x}}^1 r \wedge @_{\bar{y}}^1 \text{nom}(y, 1) \rightarrow$$

$$@_{x, y} \left(\bigvee_{t \in \mathcal{T}} t \wedge \bigwedge_{t' \neq t \in \mathcal{T}} \neg(t \wedge t') \right).$$

The tile types in \mathcal{T} are understood as propositional variables. In order to access from the root node ε to a node encoding a position of the grid, one needs first to access to a child v of ε satisfying r (and this is done with the help of the local nominal x) and then to access to any child v' of v (done with the local nominal y). Then, to reason propositionally on v' , it is sufficient to consider subformulae of the form $@_{x, y} \psi$. This principle is applied for all the formulae below. The formula $\phi_{\mathcal{H}}$ defined below encodes the horizontal matching constraints.

$$\forall x, x', y, y' (\text{nom}(x, 1) \wedge @_{\bar{x}}^1 r \wedge \text{nom}(x', 1) \wedge @_{\bar{x}'}^1 r \wedge$$

$$@_{\bar{x}}^1 \text{nom}(y, 1) \wedge @_{\bar{x}'}^1 \text{nom}(y', 1) \wedge \text{nb}(x') \stackrel{k+1}{=} \text{nb}(x) + 1 \wedge$$

$$\text{nb}(x, y) \stackrel{k+1}{=} \text{nb}(x, y')) \rightarrow \bigvee_{(t, t') \in \mathcal{H}} @_{x, y} t \wedge @_{x', y'} t'$$

The formula $\phi_{\mathcal{V}}$ for vertical matching constraints is defined similarly. It remains to express the initial conditions. It is sufficient to identify the n first children of the first child of ε (identified by the satisfaction of $\text{first}(k)$). For example, to express that the j th child of the first child of ε (say v is this first child of ε) satisfies t_j , perform an lsr -partition on v , enforce that the unique child satisfying s also satisfies t_j and express that there are exactly $j - 1$ children of v satisfying r . This is a condition from graded

modal logic that is easy to express. Let $\mathbf{EX}_{=i} \psi$ be the formula below stating that exactly $i \geq 1$ children satisfy ψ : $\exists q_1, \dots, q_i \text{ diff-nom}(q_1, \dots, q_i, 1) \wedge \mathbf{AX}((q_1 \vee \dots \vee q_i) \leftrightarrow \psi)$, where q_1, \dots, q_i are fresh variables. By convention $\mathbf{EX}_{=0} \psi$ is defined as $\mathbf{AX}\neg\psi$. The formula ϕ_{init} is defined as $\forall x (\text{nom}(x, 1) \wedge @_x^1(\text{first}(k))) \rightarrow @_x^1(\bigwedge_{i \in [0, n-1]} \exists l, s, r \text{LSR}_\varepsilon(k) \wedge \mathbf{EX}_{=i} r \wedge \mathbf{EX}(s \wedge t_i))$. The correctness of the reduction is stated below.

Lemma 19. $\mathcal{P} = (\mathcal{T}, \mathcal{H}, \mathcal{V})$, $c = t_0, t_1, \dots, t_{n-1}$ is a positive instance of Tiling_k iff ϕ is satisfiable in $\text{QCTL}_{\mathbf{X}}^t$.

Here is one of the main results of the paper.

Theorem 20. $\text{SAT}(\text{QCTL}_{\mathbf{X}}^t)$ is TOWER-complete.

Theorem 20 improves the TOWER lower bound from [13, Cor. 5.6] by considering as only temporal operator, the (local) modality \mathbf{EX} . TOWER-hardness can be also obtained with arbitrary countable trees. In Section V, we show that this entails more TOWER-hardness results for other fragments of $\text{QCTL}_{\mathbf{X}}^t$ and for modal logics with propositional quantification under appropriate tree semantics.

V. A HARVEST OF TOWER-COMPLETE LOGICS

Now, we capitalise on the TOWER-hardness of $\text{SAT}(\text{QCTL}_{\mathbf{X}}^t)$, by showing TOWER-hardness of other fragments of QCTL^t that involve only \mathbf{EF} or its strict variant \mathbf{EXEF} (Section V-A). TOWER-hardness is obtained by reduction from $\text{SAT}(\text{QCTL}_{\mathbf{X}}^t)$ by introducing propositional variables that enforce layers from the root in the tree model and therefore this allows us to simulate \mathbf{EX} . In Section V-B, we consider well-known modal logics that are complete for classes of tree-like Kripke structures, and we show that their extension with propositional quantification for such classes is decidable in TOWER, but more importantly TOWER-hard. Some of such classes involve finite trees and therefore, we take also the opportunity to study $\text{QCTL}_{\mathbf{X}}^{ft}$ and $\text{QCTL}_{\mathbf{XF}}^{ft}$ that happen, for instance, to be closely related to the modal logics \mathbf{K} and \mathbf{GL} .

A. $\text{SAT}(\text{QCTL}_{\mathbf{F}}^t)$ and $\text{SAT}(\text{QCTL}_{\mathbf{XF}}^t)$ are TOWER-hard!

The fragment $\text{QCTL}_{\mathbf{F}}^t$ of QCTL^t is defined according to $\phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid \mathbf{EF}\phi \mid \exists p \phi$. We recall the standard semantics for \mathbf{EF} -formulae: $\mathfrak{T}, v \models \mathbf{EF}\phi \stackrel{\text{def}}{\iff}$ there is $j \geq 0$ such that $vE^j v'$ and $\mathfrak{T}, v' \models \phi$, and $\mathbf{AG}\phi \stackrel{\text{def}}{\iff} \neg\mathbf{EF}\neg\phi$.

In order to show that $\text{SAT}(\text{QCTL}_{\mathbf{F}}^t)$ is TOWER-hard, we design a logspace reduction from $\text{SAT}(\text{QCTL}_{\mathbf{X}}^t)$. Let ϕ be in $\text{QCTL}_{\mathbf{X}}^t$ with $md(\phi) = k$. W.l.o.g., we assume that ϕ may contain occurrences of \mathbf{EX} and no occurrences of \mathbf{AX} . Let us define $\text{trans}(k, \phi) \wedge \text{shape}(k)$ in $\text{QCTL}_{\mathbf{F}}^t$, where $\text{shape}(k)$ enforces a discipline for layers (explained below) and $\text{trans}(k, \phi)$ admits a recursive definition, by relativising the occurrences of \mathbf{EX} . Let us consider the set of propositional variables $Y_k = \{\text{layer}_i \mid i \in [-1, k]\}$ with the intended meaning that a node satisfying layer_i is of “layer i ”, the root node being of layer k . Indeed, there is a need for such propositional variables, as unlike with

$\text{QCTL}_{\mathbf{X}}^t$, we have to enforce that moving with \mathbf{EF} leads to a lower layer. The formula $\text{shape}(k)$ is the conjunction of the following formulae:

- Every node satisfies exactly one propositional variable from Y_k (layer unicity): $\mathbf{AG}((\text{layer}_{-1} \vee \text{layer}_0 \vee \dots \vee \text{layer}_k) \wedge \bigwedge_{-1 \leq i \neq j \leq k} \neg(\text{layer}_i \wedge \text{layer}_j))$.
- When a node satisfies layer_i with $i \leq k$, none of its descendants satisfies some layer_j with $j > i$ (monotonicity of layer numbers): $\bigwedge_{i \leq k} \mathbf{AG}(\text{layer}_i \rightarrow \mathbf{AG}(\text{layer}_{-1} \vee \text{layer}_0 \vee \dots \vee \text{layer}_i))$.
- When a node satisfies layer_i with $0 \leq i \leq k$, there is a descendant satisfying layer_{i-1} (weak progress): $\bigwedge_{0 \leq i \leq k} \mathbf{AG}(\text{layer}_i \rightarrow \mathbf{EF} \text{layer}_{i-1})$.
- When a node satisfies layer_i with $0 \leq i \leq k$, it has no (strict) descendant satisfying layer_i (no stuttering): $\bigwedge_{0 \leq i \leq k} \mathbf{AG}(\text{layer}_i \rightarrow \neg \exists p (p \wedge \mathbf{EF}(\text{layer}_i \wedge \neg p)))$. This type of constraints does not apply to layer_{-1} .
- The root node is at layer k : layer_k .

Let $\mathfrak{T} = \langle V, E, l \rangle$ be a tree model for QCTL^t with root node ε . \mathfrak{T} is k -layered $\stackrel{\text{def}}{\iff}$ the conditions below hold:

- For every node $v \in V$, $\text{card}(l(v) \cap Y_k) = 1$.
- For all $v \in V$ s.t. $\text{layer}_j \in l(v)$ for some $j \in [-1, k]$,
 - if $j \geq 0$, then there is v' such that vEv' and $\text{layer}_{j-1} \in l(v')$ and,
 - for all v' such that vE^+v' and $\text{layer}_{j'} \in l(v')$, we have $j' \leq j$.
- For all $j \in [0, k]$, there are no v and v' such that vE^+v' and $\text{layer}_j \in l(v) \cap l(v')$.
- $\text{layer}_k \in l(\varepsilon)$.

This means that the only propositional variable from Y_k satisfied by a node reachable in $j \in [1, k]$ steps from ε is layer_m for some $m \leq k-j$, and the only propositional variable from Y_k satisfied by a node reachable in strictly more than k steps from ε is layer_{-1} . Moreover, once layer_{-1} holds true, it holds for all its descendants. Actually, the formula $\text{shape}(k)$ characterises k -layered structures.

Lemma 21. Let $\mathfrak{T} = \langle V, E, l \rangle$ be a tree model for QCTL^t with root node ε . $\mathfrak{T}, \varepsilon \models \text{shape}(k)$ iff \mathfrak{T} is k -layered.

To define $\text{trans}(k, \phi)$, we define inductively $\text{trans}(i, \psi)$ where ψ is a subformula of ϕ and $md(\psi) \leq i$.

- $\text{trans}(i, p) \stackrel{\text{def}}{=} p$ for all propositional variables p ,
- trans is homomorphic for Boolean connectives and $\text{trans}(i, \exists p \psi) \stackrel{\text{def}}{=} \exists p \text{trans}(i, \psi)$,
- $\text{trans}(i, \mathbf{EX}\psi) \stackrel{\text{def}}{=} \mathbf{EF}(\text{layer}_{i-1} \wedge \text{trans}(i-1, \psi))$.

$\text{trans}(k, \phi)$ has no occurrence of layer_{-1} (because $md(\phi) = k$, and translating an \mathbf{EX} -formula decreases by exactly one the index of the layer). Correctness of the reduction can be now stated as follows.

Lemma 22. ϕ is satisfiable for $\text{QCTL}_{\mathbf{X}}^t$ iff $\text{trans}(k, \phi) \wedge \text{shape}(k)$ is satisfiable for $\text{QCTL}_{\mathbf{F}}^t$.

As a conclusion, we obtain another important result.

Theorem 23. $\text{SAT}(\text{QCTL}_{\mathbf{F}}^t)$ is TOWER-complete.

The complexity upper bound TOWER is established for QCTL^t in [13] and in particular for $\text{QCTL}_{\mathbf{F}}^t$. Theorem 23 admits a variant in which we only allow to move to proper descendants, which amounts to replace **EF** by **EXEF** in $\text{QCTL}_{\mathbf{F}}^t$, leading to the variant $\text{QCTL}_{\mathbf{XF}}^t$ (with formulae obtained from $\phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid \mathbf{EXEF}\phi \mid \exists p \phi$).

Theorem 24. $\text{SAT}(\text{QCTL}_{\mathbf{XF}}^t)$ is TOWER-complete.

As above, the complexity upper bound TOWER for $\text{SAT}(\text{QCTL}_{\mathbf{XF}}^t)$ is inherited from $\text{SAT}(\text{QCTL}^t)$ in [13]. By observing that in QCTL^t , $\mathbf{EF}p$ is logically equivalent to $p \vee \mathbf{EXEF}p$, this allows us to design an elementary reduction from $\text{SAT}(\text{QCTL}_{\mathbf{F}}^t)$ into $\text{SAT}(\text{QCTL}_{\mathbf{XF}}^t)$, whence the TOWER-hardness of $\text{SAT}(\text{QCTL}_{\mathbf{XF}}^t)$.

B. Standard modal logics on trees

Now, we focus on modal logics that are complete for classes of tree-like structures. For instance, the modal logic **K** is complete for the class of finite trees [52], [35], and its extension with propositional quantification corresponds to $\text{QCTL}_{\mathbf{X}}^{ft}$. It is worth noting that a given modal logic can be complete for different classes of Kripke models (e.g., **K** is complete for the class of all the Kripke models, but also complete for the class of finite Kripke models) and their extension to propositional quantification may lead to distinct logics. Typically, **K** with propositional quantification under the structure semantics is undecidable [2] whereas it is TOWER-complete under the finite tree semantics.

Theorem 25. *The satisfiability problem for the modal logic **K** with propositional quantification under the finite tree semantics is TOWER-complete.*

The satisfiability problem in Theorem 25 is exactly $\text{SAT}(\text{QCTL}_{\mathbf{X}}^{ft})$. Besides, the modal logic **KD** (**K** with seriality) is known to be complete for the class of finite-branching trees for which all the maximal branches are infinite (the models of QCTL^t). Hence, the satisfiability problem for **KD** with propositional quantification under the tree semantics is TOWER-complete as it corresponds exactly to $\text{SAT}(\text{QCTL}_{\mathbf{X}}^t)$. Similarly, the modal logic **GL** is known to be complete for the class of finite transitive trees (**GL** is complete with respect to finite irreflexive transitive Kripke models [53]), i.e. the class of Kripke structures $\langle V, E^+, l \rangle$ such that $\langle V, E, l \rangle$ is a finite tree model, see e.g. [35]. Hence, the modal logic **GL** with propositional quantification under the finite transitive tree semantics is equivalent to $\text{QCTL}_{\mathbf{XF}}^{ft}$. Adding propositional quantification to **GL** is studied in [40], where a fragment is shown decidable by translation into WS1S [39]. Theorem 26 provides a new decidability result by reduction into $\text{SAT}(\text{QCTL}^t)$ and a complexity characterisation with a remarkable lower bound.

Theorem 26. *The satisfiability problem for the modal logic **GL** with propositional quantification under the finite*

transitive tree semantics is TOWER-complete.

The proof of Theorem 26 amounts to show that $\text{SAT}(\text{QCTL}_{\mathbf{XF}}^{ft})$ is TOWER-complete. Furthermore, the modal logic **K4** is complete for the class of Kripke structures $\langle V, E^+, l \rangle$ such that $\langle V, E, l \rangle$ is a finite-branching tree model (some branches may be infinite, some others not). So, **K4** with propositional quantification under the general tree semantics is equivalent to $\text{QCTL}_{\mathbf{XF}}^{gt}$, understood as the variant of $\text{QCTL}_{\mathbf{XF}}^t$ in which the tree models are (only required to be) finite-branching.

Theorem 27. *The satisfiability problem for the modal logic **K4** with propositional quantification under the general tree semantics is TOWER-complete.*

S4 (resp. **D4**) is also known to be complete for the class of Kripke structures $\langle V, E^*, l \rangle$ (resp. $\langle V, E^+, l \rangle$) such that $\langle V, E, l \rangle$ is a finite-branching tree model with all branches infinite. Consequently, **S4** (resp. **D4**) with propositional quantification under the tree semantics is equivalent to $\text{QCTL}_{\mathbf{F}}^t$ (resp. $\text{QCTL}_{\mathbf{XF}}^t$). So, the satisfiability problem for **S4** (resp. **D4**) with propositional quantification under the tree semantics is TOWER-complete (see Theorem 23 and Theorem 24). By [2], **S4** with propositional quantification under the structure semantics is undecidable whereas **S4** with propositional quantification under the class of finite trees (resp. under the class of trees) is shown decidable in [16] by invoking Rabin’s Theorem [33]. Though our upper bound for **S4** leads to a similar upper bound, our TOWER-hardness result is significantly different for **S4**.

VI. CONCLUSION

We have shown that the satisfiability problems for $\text{QCTL}_{\mathbf{X}}^t$, $\text{QCTL}_{\mathbf{F}}^t$ and $\text{QCTL}_{\mathbf{XF}}^t$ are TOWER-complete, and for $N \geq 2$, $\text{SAT}(\text{QCTL}_{\mathbf{X}, \leq N}^t)$ is AEXP_{POL} -complete. Whereas AEXP_{POL} -hardness is established by reducing the alternating multi-tiling problem recently introduced in [31], the TOWER-hardness of $\text{SAT}(\text{QCTL}_{\mathbf{X}}^t)$ required to be able to express that a node has a number of children equal to some tower of exponentials of height k . Section V deals with the TOWER-completeness of $\text{SAT}(\text{QCTL}_{\mathbf{X}}^{ft})$ and $\text{SAT}(\text{QCTL}_{\mathbf{XF}}^{ft})$, as well as TOWER-completeness for the modal logics **K**, **KD**, **GL**, **K4**, **D4** and **S4** with propositional quantification but with adequate classes of tree-like structures. All our TOWER-hardness results significantly improve what was known so far for fragments of QCTL^t and for the above-mentioned well-known modal logics. This work can be continued in several directions, for instance to characterise the expressiveness of $\text{QCTL}_{\mathbf{X}}^t$ along the lines of [14] or to analyse whether our results can be lifted to modal separation logics where separating conjunction can be encoded by propositional quantification, see e.g. [54], [55].

ACKNOWLEDGEMENTS

B. Bednarczyk was supported by the Polish Ministry of Science and Higher Education program ”Diamentowy Grant” no. DI2017 006447.

REFERENCES

- [1] R. Bull, “On modal logic with propositional quantifiers,” *JSL*, vol. 34, no. 2, pp. 257–263, 1969.
- [2] K. Fine, “Propositional quantifiers in modal logic,” *Theoria*, vol. 36, pp. 336–346, 1970.
- [3] D. Kaplan, “S5 with quantifiable propositional variables,” *JSL*, vol. 35, no. 2, p. 355, 1970.
- [4] G. Antonelli and R. Thomason, “Representability in second-order propositional poly-modal logic,” *JSL*, vol. 67, no. 3, pp. 1039–1054, 2002.
- [5] K. Engelhardt, R. van der Meyden, and Y. Moses, “Knowledge and the logic of local propositions,” in *TARK’98*. Morgan Kaufmann, 1998, pp. 29–41.
- [6] S. Kuhn, “A simple embedding of T into double S5,” *NDJFL*, vol. 45, no. 1, pp. 13–18, 2004.
- [7] M. Kaminski and M. Tiomkin, “The expressive power of second order propositional modal logic,” *NDJFL*, vol. 37, no. 1, pp. 35–43, 1996.
- [8] P. Kremer, “Propositional quantification in the topological semantics for S4,” *NDJFL*, vol. 38, no. 2, pp. 295–313, 1997.
- [9] B. ten Cate, “Expressivity of second order propositional modal logic,” *JPL*, vol. 35, no. 2, pp. 209–223, 2006.
- [10] A. Sistla, “Theoretical issues in the design and verification of distributed systems,” Ph.D. dissertation, Harvard University, 1983.
- [11] A. Sistla, M. Vardi, and P. Wolper, “The complementation problem for Büchi automata with applications to temporal logic,” *TCS*, vol. 49, pp. 217–237, 1987.
- [12] A. Meyer, “Weak second order theory of successor is not elementary-recursive,” MIT, Tech. Rep. MAC TM-38, 1973, 26 pages.
- [13] F. Laroussinie and N. Markey, “Quantified CTL: expressiveness and complexity,” *LMCS*, vol. 10, no. 4, 2014.
- [14] A. David, F. Laroussinie, and N. Markey, “On the expressiveness of QCTL,” in *CONCUR’16*, ser. LIPIcs, vol. 59. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016, pp. 28:1–28:15.
- [15] S. Schmitz, “Complexity hierarchies beyond elementary,” *TOCT*, vol. 8, no. 1, pp. 3:1–3:36, 2016.
- [16] R. Zach, “Decidability of quantified propositional intuitionistic logic and S4 on trees of height and arity $\leq \omega$,” *JPL*, vol. 33, no. 2, pp. 155–164, 2004.
- [17] F. Laroussinie and N. Markey, “Augmenting ATL with strategy contexts,” *IC*, vol. 245, pp. 98–123, 2015.
- [18] A. Patthak, I. Bhattacharya, A. Dasgupta, P. Dasgupta, and P. Chakrabarti, “Quantified computation tree logic,” *IPL*, vol. 82, no. 3, pp. 123–129, 2002.
- [19] T. French, “Quantified propositional temporal logic with repeating states,” in *TIME-ICTL’03*. IEEE Computer Society, 2003, pp. 155–165.
- [20] L. Bozzelli, H. van Ditmarsch, and S. Pinchinat, “The complexity of one-agent refinement modal logic,” *TCS*, vol. 603, pp. 58–83, 2015.
- [21] F. Belardinelli and W. van der Hoek, “A semantical analysis of second-order propositional modal logic,” in *AAAI’16*. AAAI Press, 2016, pp. 886–892.
- [22] F. Belardinelli, W. van der Hoek, and L. Kuijter, “Second-order propositional modal logic: Expressiveness and completeness results,” *Artificial Intelligence*, vol. 263, pp. 3–45, 2018.
- [23] S. Riedweg and S. Pinchinat, “Quantified mu-calculus for control synthesis,” in *MFCSS’03*, ser. LNCS, vol. 2747. Springer, 2003, pp. 642–651.
- [24] O. Kupferman, “Augmenting branching temporal logics with existential quantification over atomic propositions,” *JLC*, vol. 9, no. 2, pp. 135–147, 1999.
- [25] C. Areces, P. Blackburn, and M. Marx, “Complexity of hybrid temporal logics,” *Logic Journal of the IGPL*, vol. 8, no. 5, pp. 653–679, 2000.
- [26] —, “Hybrid logics: characterization, interpolation and complexity,” *JSL*, vol. 66, no. 3, pp. 977–1010, 2001.
- [27] M. Marx, “Narcissists, stepmothers and spies,” in *Workshop on Description Logics (DL’02)*, ser. CEUR Workshop Proceedings, vol. 53, 2002.
- [28] V. Weber, “Hybrid branching-time logics,” *CoRR*, vol. abs/0708.1723, 2007.
- [29] V. Goranko and S. Passy, “Using the universal modality: gains and questions,” *JLC*, vol. 2, no. 1, pp. 5–30, 1992.
- [30] S. Demri, V. Goranko, and M. Lange, *Temporal Logics in Computer Science*. Cambridge University Press, 2016.
- [31] L. Bozzelli, A. Molinari, A. Montanari, and A. Peron, “On the complexity of model checking for syntactically maximal fragments of the interval temporal logic HS with regular expressions,” in *GandALF’17*, ser. EPTCS, vol. 256, 2017, pp. 31–45.
- [32] A. Molinari, “Model checking: The interval way,” Ph.D. dissertation, University of Udine, Italy, February 2019.
- [33] M. Rabin, “Decidability of second-order theories and automata on infinite trees,” *Transactions of the American Mathematical Society*, vol. 41, pp. 1–35, 1969.
- [34] A. Emerson and P. Sistla, “Deciding full branching time logic,” *Information and Control*, vol. 61, pp. 175–201, 1984.
- [35] P. Blackburn, M. de Rijke, and Y. Venema, *Modal Logic*. Cambridge University Press, 2001.
- [36] E. Clarke and A. Emerson, “Design and synthesis of synchronization skeletons using branching time temporal logic,” in *Workshop on Logic of Programs*, ser. LNCS, vol. 131. Springer, 1981, pp. 52–71.
- [37] T. French, “Decidability of quantified propositional branching time logics,” in *14th Australian Joint Conference on Artificial Intelligence*, ser. LNCS, vol. 2256. Springer, 2001, pp. 165–176.
- [38] W. Thomas, “Languages, automata, and logic,” in *Handbook of Formal Languages, Vol. 3: Beyond Words*, G. Rozenberg and A. Salomaa, Eds. Springer, 1997, pp. 389–455.
- [39] J. Büchi, “On a decision method in restricted second-order arithmetic,” in *Logic, Methodology, and Philosophy of Science*, 1960, pp. 1–11.
- [40] S. Artëmov and L. Beklemishev, “On propositional quantifiers in provability logic,” *NDJFL*, vol. 34, no. 3, pp. 401–419, 1993.
- [41] M. Baaz, A. Ciabattoni, and R. Zach, “Quantified propositional Gödel logics,” in *LPAR’00*, ser. LNCS, vol. 1955. Springer, 2000, pp. 240–256.
- [42] P. van Emde Boas, “The convenience of tilings,” in *Complexity, Logic, and recursion Theory*, ser. Lecture Notes in Pure and Applied Logic, A. Sorbi, Ed., vol. 187. Marcel Dekker, Inc., 1997, pp. 331–363.
- [43] A. Chandra, D. Kozen, and L. Stockmeyer, “Alternation,” *Journal of the ACM*, vol. 28, no. 1, pp. 114–133, 1981.
- [44] L. Berman, “The complexity of logical theories,” *TCS*, vol. 11, pp. 71–77, 1980.
- [45] J. Ferrante and C. Rackoff, “A decision procedure for the first order theory of real addition with order,” *SIAM Journal of Computing*, vol. 4, no. 1, pp. 69–76, 1975.
- [46] K. Fine, “In so many possible worlds,” *NDJFL*, vol. 13, no. 4, pp. 516–520, 1972.
- [47] M. Fattorosi Barnaba and F. De Caro, “Graded modalities,” *Studia Logica*, vol. 44, no. 2, pp. 197–221, 1985.
- [48] S. Tobies, “PSPACE reasoning for graded modal logics,” *JLC*, vol. 11, pp. 85–106, 2001.
- [49] L. Stockmeyer, “The complexity of decision problems in automata theory and logic,” Ph.D. dissertation, Department of Electrical Engineering, MIT, 1974.
- [50] M. Vardi, “The complexity of relational query languages,” in *STOC’82*, 1982, pp. 137–146.
- [51] P. Schnoebelen, “The complexity of temporal logic model checking,” in *AiML’02*. College Publications, 2003, pp. 437–459.
- [52] K. Segerberg, “An essay in classical modal logic (three vols.),” Uppsala Universitet, Tech. Rep. Filosofiska Studier nr 13, 1971.
- [53] C. Smoryński, *Self-reference and Modal Logic*. Springer-Verlag, 1985.
- [54] S. Demri and R. Fervari, “On the complexity of modal separation logics,” in *AiML’18*. College Publications, 2018, pp. 179–198.
- [55] S. Demri, “On temporal and separation logics (invited paper),” in *TIME’18*, ser. LIPIcs, vol. 120. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018, pp. 1:1–1:4.