

# Graded Computation Tree Logic with Binary Coding<sup>\*</sup>

Alessandro Bianco, Fabio Mogavero, and Aniello Murano

Università degli Studi di Napoli "Federico II", via Cinthia, I-80126, Napoli, Italy  
{alessandrobianco,mogavero,murano}@na.infn.it  
<http://people.na.infn.it/{alessandrobianco,mogavero,murano}>

**Abstract.** *Graded path quantifiers* have been recently introduced and investigated as a useful framework for generalizing standard existential and universal path quantifiers in the branching-time temporal logic CTL (GCTL), in such a way that they can express statements about a minimal and conservative number of accessible paths. These quantifiers naturally extend to paths the concept of *graded world modalities*, which has been deeply investigated for the  $\mu$ -CALCULUS ( $G\mu$ -CALCULUS) where it allows to express statements about a given number of immediately accessible worlds. As for the “non-graded” case, it has been shown that the satisfiability problem for GCTL and the  $G\mu$ -CALCULUS coincides and, in particular, it remains solvable in EXPTIME. However, GCTL has been only investigated w.r.t. graded numbers coded in unary, while  $G\mu$ -CALCULUS uses for this a binary coding, and it was left open the problem to decide whether the same result may or may not hold for binary GCTL. In this paper, by exploiting an automata theoretic-approach, which involves a model of alternating automata with satellites, we answer positively to this question. We further investigate the succinctness of binary GCTL and show that it is at least exponentially more succinct than  $G\mu$ -CALCULUS.

## 1 Introduction

*Temporal logic* is a suitable framework for reasoning about the correctness of concurrent programs [19, 20]. Depending on the view of the underlying nature of time, two types of temporal logics are mainly considered [14]. In *linear-time temporal logics*, such as LTL [19], time is treated as if each moment in time has a unique possible future. Conversely, in *branching-time temporal logics*, such as CTL [4] and CTL\* [5], each moment in time may split into various possible futures and *existential* and *universal quantifiers* are used to express properties along one or all the possible futures. Recently in [2], *graded path modalities* have been introduced as a useful extension of these branching quantifiers in such a way that they can express statements about a *minimal* and *conservative* number of accessible paths. In particular, they allow to express properties such as “there are at least  $n$  minimal and conservative paths satisfying a formula  $\psi$ ”, by formally writing  $E^{\geq n}\psi$ , for suitable and well-formed concepts of minimality and conservativeness among paths. This generalization has been deeply investigated in [2] for the logic CTL, where the extended logic has been named GCTL. In particular, GCTL has been proved to be very powerful as it results in a logic more expressive

---

<sup>\*</sup> Work partially supported by MIUR PRIN Project n.2007-9E5KM8.

than CTL where system specifications can be expressed in a very succinct way, without affecting the complexity of the logic. Indeed, the satisfiability problem for GCTL is EXPTIME-COMPLETE, as for CTL. There are several practical examples that show the usefulness of GCTL and we refer to [2, 6, 7] for a significant list.

Graded path modalities extend to paths the concept of *graded word modalities*, which has been investigated for several logics such as ALC,  $\mu$ -CALCULUS, and the *first order logic* [3, 8, 10, 11, 22]. In particular, as for GCTL, the graded word modalities allow to extend the  $\mu$ -CALCULUS ( $G\mu$ -CALCULUS, for short) in a very powerful logic, without increasing its computational complexity. Indeed, the satisfiability problem for the  $G\mu$ -CALCULUS remains solvable in EXPTIME. Despite its high expressive power, the  $G\mu$ -CALCULUS is considered in some sense a low-level logic (as this is intrinsic in the  $\mu$ -CALCULUS), making it an “unfriendly” logic for users. On the contrary, although less expressive than the  $G\mu$ -CALCULUS, GCTL can easily and naturally express complex graded properties of computation trees. However, we recall that the logic introduced in [2] considers every number  $n$  appearing in a graded quantifier  $E^{\geq n}\psi$  as coded in unary (*unary GCTL*), while the  $G\mu$ -CALCULUS, as it has been introduced and studied in [3, 11], considers these numbers as coded in binary<sup>1</sup>. In particular, the technique developed in [2] to solve the satisfiability problem for GCTL gives an exponential upper bound only in the unary case, while it gives a double-exponential upper bound if applied to the binary case. In [2], it was left as an open problem to check whether a single exponential upper bound may also hold for binary GCTL. We further remark that this problem was left open also in the case of “non-minimal” and “non-conservative” graded path quantifiers [7]. In this paper, we positively answer to this question. As for unary GCTL, we show an upper bound for the satisfiability of the binary GCTL, by exploiting an automata-theoretic approach [13, 23]. Before describing the technique we develop here, let us first recall the one we used in [2] for unary GCTL and discuss in detail the points that lead to a double-exponential upper bound when it is applied to binary GCTL. Then, we explain as we have managed these aspects for gaining the desired upper bound.

Recall that to develop a decision procedure automata-theoretic based for a logic with the tree model property, one first develops an appropriate notion of tree automata and studies their emptiness problem. Then, the satisfiability problem for the logic is reduced to the emptiness problem of the automata. To this aim, in [2], it has first shown that the tree model property for GCTL holds, by showing that each unary GCTL formula  $\varphi$  is satisfiable on a Kripke structure iff it has a tree model whose branching degree is polynomial in the size of  $\varphi$ . Then, a corresponding tree automaton model named *partitioning alternating Büchi tree automata* (PABT) has been introduced and shown that, for each unary GCTL formula  $\varphi$ , it is always possible to build in linear time a PABT accepting all tree models of  $\varphi$ . Then, by using a nontrivial extension of the Miyano and Hayashi technique [17] it has been shown an exponential translation of a PABT into a non-deterministic Büchi tree automata (NBT). Since the emptiness problem for NBT is solvable in polynomial time (in the size of the transition function that is polynomial in

---

<sup>1</sup> The  $G\mu$ -CALCULUS was first considered in the unary case [9] and it required much effort to be solved efficiently for the binary coding as well. This further gives an evidence that working with binary logics is rather than an easy task.

the number of states and exponential in the width of the tree in input) [24], we obtain that the satisfiability problem for unary GCTL is solvable in EXPTIME.

A detailed analysis on the above technique shows two points where it fails to give a single exponential-time algorithm when applied to binary GCTL. First, the tree model property shows for binary GCTL the necessity to consider also tree models with a branching degree exponential in the highest degree of the formula. Second, the number of states of the NBT derived from the PABT is double-exponential in the coding of the highest degree  $g$  of the formula. These two points reflect directly in the transition relation of the NBT, which turns to be double exponential in the coding of the degree  $g$ . To take care of the first point, we develop a sharp binary encoding of each tree model. In practice, for a given model  $\mathcal{T}$  of  $\varphi$  we build a binary encoding  $\mathcal{T}_D$  of  $\mathcal{T}$ , called *delayed generation tree*, such that, for each node  $x$  in  $\mathcal{T}$  having  $m + 1$  children  $x \cdot 0, \dots, x \cdot m$ , there is a corresponding node  $y$  of  $x$  in  $\mathcal{T}_D$  and nodes  $y \cdot 0^i$  having  $x \cdot i$  as right child and  $y \cdot 0^{(i+1)}$  as left child, for  $0 \leq i \leq m$ . To address the second point, we exploit a careful construction of the alternating automaton accepting all models of the formula, in a way that the graded numbers do not give any exponential blow-up in the translating of the automaton into an NBT.

We now describe the main idea behind the automata construction. Basically, we use alternating tree automata enriched with *satellites* (ATAS) as an extension of that introduced in [12]. In particular, we use the Büchi acceptance condition (ABTS). The satellite is a nondeterministic tree automaton and is used to ensure that the tree model satisfies some structural properties along its paths and it is kept apart from the main automaton. This separation, as it has been proved in [12], allows to solve the emptiness problem for Büchi automata in a time exponential in the number of states of the main automaton and polynomial in the number of states of the satellite. Then, we obtain the desired complexity by forcing the satellite to take care of the graded modalities and by noting that the main automaton is polynomial in the size of the formula.

The achieved result is even more appealing as we also show here that binary GCTL is much more succinct than  $G\mu$ -CALCULUS. We recall that some preliminary studies on this aspect were already carried out in [2], but only for the unary case. There, some examples in which unary GCTL is at least exponentially more succinct than binary  $G\mu$ -CALCULUS were also shown, but this does not hold in general as there are formulas from the latter that show the opposite<sup>2</sup>. In this paper, we show that binary GCTL is at least exponentially more succinct than binary  $G\mu$ -CALCULUS.

Finally, we report that in the full version we also discuss the application of the technique we have exploited for GCTL to the more expressive logic case of the binary *graded* CTL\* (GCTL\*, for short), i.e., CTL\* augmented with graded path quantifiers. Clearly, we cannot simply use ABTS for GCTL\*, as the Büchi acceptance condition is too weak already for CTL\*. We use instead ATAS along with the hesitant condition (AHTS). Due to the particular semantics of the logic, based on minimality and conservativeness, we cannot use as automata states either formulas (as for GCTL) or atoms (i.e., consistent sets of formulas, as for CTL\*). In fact, we need sets of atoms, instead. All these peculiarities lead to a 3EXPTIME satisfiability procedure for GCTL\*.

---

<sup>2</sup> Note that in [2] it was erroneously stated that unary GCTL is in general exponentially more succinct than  $G\mu$ -CALCULUS.

*Related work.* Graded modalities along with CTL have been also studied in [6, 7], but under a different semantics. There, the authors consider overlapping paths (as we do) as well as disjoint paths, but they do not consider the concepts of minimality and conservativeness, which we deeply use in our logics, as it is well described and motivated in [2]. In [6] the model checking problem for non-minimal and non-conservative unary GCTL has been investigated. In particular, by opportunely extending the classical algorithm for CTL [4], they show that, in the case of overlapping paths, the model checking problem is PTIME-COMPLETE (thus not harder than CTL), while in the case of disjoint paths, it is in PSPACE and both NPTIME-HARD and CONPTIME-HARD. The work continues in [7], by showing a symbolic model checking algorithm for the binary coding and, limited to the unary case, a satisfiability procedure. Regarding the comparison between GCTL and graded CTL with overlapping paths studied in [6], it can be shown that they are equivalent by using an exponential reduction in both ways, whereas we do not know whether any of the two blow-up can be avoid. However, it is important to note that our technique can be also adapted to obtain an EXPTIME satisfiability procedure for the binary graded CTL under the semantics proposed in [6]. Indeed, it is needed only to slightly modify the transition function of the main automaton (w.r.t. until and release formulas), without changing the structure of the whole satellite. Moreover, it can be used to prove that, in the case of unary GCTL, the complexity of the satisfiability problem is only polynomial in the degree. Finally, our method can be also applied to the satisfiability of the  $G\mu$ -CALCULUS while the technique developed in [11] cannot be used for GCTL.

Due to space limitation, all proofs are omitted and reported in a full version of the paper. Preliminary materials on the subject can be found in [2].

## 2 Preliminaries

Given two sets  $X$  and  $Y$  of *objects*, we denote by  $|X|$  the *size* of  $X$ , i.e., the number of its elements, by  $2^X$  the *powerset* of  $X$ , i.e., the set of all its subsets, and by  $Y^X \subseteq 2^{X \times Y}$  the set of *total functions*  $f : X \mapsto Y$ . By  $X^n$  we denote the set of all  $n$ -*tuples* of elements from  $X$ , by  $X^* = \bigcup_{n=0}^{\omega} X^n$  the set of *finite words* on the *alphabet*  $X$ , and by  $X^\omega$  the set of *infinite words*, where as usual,  $\omega$  is the *numerable infinity* and  $\varepsilon$  is the *empty word*. Moreover, by  $|x|$  we denote the *length* of a word  $x \in X^\infty = X^* \cup X^\omega$ . As special sets,  $\mathbb{N}$  is the sets of *natural numbers* and  $[n]$  is its subset  $\{k \in \mathbb{N} \mid k \leq n\}$ , with  $n \in \mathbb{N} \cup \{\omega\}$ .

For a set  $\Delta$ , we define a  $\Delta$ -*tree* as a set  $T \subseteq \Delta^*$  closed under *prefix*, i.e., if  $w \cdot w' \in T$ , with  $w' \in \Delta$ , then also  $w \in T$ , and we say that it is *full* iff it also holds that  $w \cdot w'' \in T$ , for all  $w'' < w'$ , where  $< \subseteq \Delta \times \Delta$  is a strict order on the directions. The elements of  $T$  are called *nodes* and the empty word  $\varepsilon$  is the *root* of  $T$ . For every  $w \in T$  and  $w' \in \Delta$ , the node  $w \cdot w' \in T$  is a *successor* of  $w$  in  $T$ . For a finite set  $\Sigma$ , a  $\Sigma$ -*labeled*  $\Delta$ -*tree* is a pair  $\langle T, \nu \rangle$ , where  $T$  is a  $\Delta$ -tree and  $\nu : T \mapsto \Sigma$  is a *labeling* function. When  $\Delta$  and  $\Sigma$  are clear from the context, we call  $\mathcal{T} = \langle T, \nu \rangle$  simply a (labeled) tree.

A *Kripke structure* (KRIPKE, for short) is a tuple  $\mathcal{K} = \langle AP, W, R, L \rangle$ , where  $AP$  is a finite non-empty set of *atomic propositions*,  $W$  is an enumerable non-empty set of *worlds*,  $R \subseteq W \times W$  is a *transition* relation, and  $L : W \mapsto 2^{AP}$  is a *labeling* function that maps each world to the set of atomic propositions true in that world. By  $|\mathcal{K}| =$

$|R| \leq |W|^2$  we denote the *size* of  $\mathcal{K}$ , which is also the size of the transition relation. A finite Kripke structure is a structure of finite size. A *path* of  $\mathcal{K}$  is a finite or infinite sequence of worlds  $\pi \in W^+ \cup W^\omega$  such that  $(\pi_i, \pi_{i+1}) \in R$ , for all  $0 \leq i < |\pi| - 1$ . By  $\text{Pth}^\infty(w) \subseteq W^+ \cup W^\omega$  (resp.,  $\text{Pth}(w) \subseteq W^+$ ) we denote the sets of all (resp., finite) paths starting at the world  $w \in W$ , i.e.,  $\pi \in \text{Pth}^\infty(w)$  implies  $\pi_0 = w$ . Let  $\pi$  and  $\pi'$  be two paths. We say that  $\pi'$  is a *subpath* of  $\pi$ , in symbols  $\pi' \leq \pi$ , iff  $\pi'$  is a non-necessarily proper prefix of  $\pi$ . Moreover, we say that  $\pi$  and  $\pi'$  are *comparable* iff (i)  $\pi \leq \pi'$  or (ii)  $\pi' \leq \pi$  holds, otherwise they are *incomparable*. For a set of paths  $X$ , we define the set of *minimal subpaths* (antichain)  $\min(X)$  as the set consisting of the  $\leq$ -minimal elements of  $X$ , i.e., it is the set containing all and only the paths  $\pi \in X$  such that for all  $\pi' \in X$ , it holds that (i)  $\pi \leq \pi'$  or (ii)  $\pi' \not\leq \pi$ . Note that all paths in  $\min(X)$  are incomparable among them. A path  $\pi$  is *minimal* w.r.t. a set  $X$  (or simply minimal, when the context clarify the set  $X$ ) iff  $\pi \in \min(X)$ . A set of paths  $X$  is minimal iff  $X = \min(X)$ .

The *unwinding* of a KRIPKE  $\mathcal{K}$  starting at the world  $w$  is a  $2^{\text{AP}}$ -labeled  $W$ -tree  $\mathcal{T}_{\mathcal{K},w} = \langle T, \nu \rangle$  for which there exists a bijective function  $\text{unw}$ , called *unwinding function*, such that (i)  $\text{unw}(w) = \varepsilon$ , (ii)  $\text{unw}(\pi) = \text{unw}(\pi_0 \dots \pi_{l-2}) \cdot \pi_{l-1}$ , and (iii)  $\nu(\text{unw}(\pi_i)) = L(\pi_i)$ , for all  $\pi \in \text{Pth}(w)$  and  $0 \leq i \leq l = |\pi| > 1$ . In this work, we also consider as unwindings of  $\mathcal{K}$  also all  $2^{\text{AP}}$ -labeled  $\mathbb{N}$ -tree that are isomorph to  $\mathcal{T}_{\mathcal{K},w}$ .

Finally, let  $n \in \mathbb{N} \setminus \{0\}$ . Then, we define  $P(n)$  as the set of all *solutions*  $\{p_i\}$  of the *linear Diophantine equation*  $1 * p_1 + 2 * p_2 + \dots + n * p_n = n$  and  $C(n)$  as the set of all the *cumulative solutions*  $\{c_i\}$  obtained by summing increasing sets of elements from  $\{p_i\}$ . Formally,  $P(n) = \{\{p_i\} \in \mathbb{N}^n \mid \sum_{i=1}^n i * p_i = n\}$  and  $C(n) = \{\{c_i\} \in \mathbb{N}^n \mid \exists \{p_i\} \in P(n). \forall 1 \leq i \leq n. c_i = \sum_{j=i}^n p_j\}$ . Note that  $|C(n)| = |P(n)|$  and, since for each solution  $\{p_i\}$  of the above Diophantine equation there is exactly one *partition* of  $n$ , we have that  $|C(n)| = p(n)$ , where  $p(n)$  is the number of partitions of  $n$ . By [1], it holds that  $|C(n)| = \Theta(\frac{1}{n} \cdot 2^{k \cdot \sqrt{n}})$ , with  $k = \pi \cdot \log e \cdot \sqrt{2/3}$ .

### 3 Full Graded Computation Tree Logic

We now define syntax and semantics of GCTL\*.

*Syntax.* The *graded computation tree logic* (GCTL\*) extends CTL\* by using two special path quantifiers, the existential  $E^{\geq g}$  and the universal  $A^{<g}$ , where  $g$  denotes the corresponding *degree*. As in CTL\*, these quantifiers can prefix a linear-time formula composed of an arbitrary combination and nesting of the temporal operators  $X$  (“*effective next*”),  $\tilde{X}$  (“*hypothetical next*”),  $U$  (“*until*”), and  $R$  (“*release*”). The quantifiers  $A^{<g}$  and  $E^{\geq g}$  can be respectively read as “*all but  $g$  minimal paths*” and “*there exist at least  $g$  minimal paths*”. The formal syntax of GCTL\* follows.

**Definition 1. (Syntax)** GCTL\* state ( $\varphi$ ) and path ( $\psi$ ) formulas are built inductively from AP using the following context-free grammar, where  $p \in \text{AP}$  and  $g \in \mathbb{N} \setminus \{0\}$ :

1.  $\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid E^{\geq g}\psi \mid A^{<g}\psi$ ,
2.  $\psi ::= \varphi \mid \neg\psi \mid \psi \wedge \psi \mid \psi \vee \psi \mid X\psi \mid \tilde{X}\psi \mid \psi U \psi \mid \psi R \psi$ .

The class of GCTL\* formulas is the set of state formulas generated by the above grammar. In addition, the simpler class of GCTL formulas is obtained by forcing each

temporal operator, occurring in a formula, to be coupled with a path quantifier, as in the classical definition of CTL.

For a state formula  $\varphi$ , we define the *degree*  $\hat{\varphi}$  of  $\varphi$  as the maximum natural number  $g$  occurring among the degrees of all its path quantifiers. We assume that all such degrees are coded in binary. The *length* of  $\varphi$ , denoted by  $|\varphi|$ , is defined as for CTL\* and does not consider the degrees at all. Accordingly, the *size* of  $\varphi$ , denoted by  $\|\varphi\|$ , is defined in the same way that the length, by considering  $\|E^{\geq g}\psi\|$  and  $\|A^{<g}\psi\|$  to be equal to  $\log(g+1) + \|\psi\|$ . Clearly, it holds that  $\log(\hat{\varphi}) \leq \|\varphi\|$  and  $|\varphi| \leq \|\varphi\|$ . We use  $\text{cl}(\varphi)$  to denote the classical Fischer-Ladner closure of  $\varphi$  augmented in a way that if  $E^{\geq g}\varphi_1 R \varphi_2 \in \text{cl}(\varphi)$  (resp.,  $A^{<g}\varphi_1 U \varphi_2 \in \text{cl}(\varphi)$ ) then also  $E^{\geq 1}\varphi_1 R \varphi_2 \in \text{cl}(\varphi)$  (resp.,  $A^{<\hat{\varphi}}\varphi_1 U \varphi_2 \in \text{cl}(\varphi)$ ). Moreover, by  $\text{ecl}(\varphi)$  (resp.,  $\text{ecl}(\varphi)^\exists$ ,  $\text{ecl}(\varphi)^\forall$ ) we denote the set of all the (resp., existential, universal) quantification formulas in  $\text{cl}(\varphi)$  deprived of the degree. Finally, by  $\text{rcl}(\varphi)$  we denote the set of quantification formulas in  $\text{cl}(\varphi) \cup \text{ecl}(\varphi)$  of the form  $E^{\geq 1}\varphi_1 R \varphi_2$ ,  $A^{<\hat{\varphi}}\varphi_1 U \varphi_2$ , and  $A\varphi_1 R \varphi_2$ .

*Semantics.* We now define the semantics of GCTL\* w.r.t. a KRIPKE  $\mathcal{K} = \langle AP, W, R, L \rangle$ . For a world  $w \in W$ , we write  $\mathcal{K}, w \models \varphi$  to indicate that a state formula  $\varphi$  holds at  $w$ , and, for a path  $\pi \in \text{Pth}^\infty(w)$ , we write  $\mathcal{K}, \pi, k \models \psi$  to indicate that a path formula  $\psi$  holds on  $\pi$  at position  $k \in [|\pi| - 1]$ . Note that, the relation  $\mathcal{K}, \pi, k \models \psi$  does not hold for any point  $k \in \mathbb{N}$ , with  $k \geq |\pi|$ . For a better readability, in the semantics definition of GCTL\* we use the special set  $P_A(\psi, w)$  and its dual  $P_E(\psi, w)$ , with the following meaning:  $P_A(\psi, w)$  contains every path  $\pi$  starting in  $w$  such that all its extensions  $\pi'$  (including  $\pi$ ) satisfy the path formula  $\psi$ . The semantics of GCTL\* state formulas of the form  $A^{<g}\psi$  and  $E^{\geq g}\psi$  follows. The semantics of the remaining GCTL\* state formulas and all GCTL\* path formulas is defined as usual in CTL\*.

**Definition 2. (Semantics of  $E^{\geq g}$  and  $A^{<g}$ )** Given a KRIPKE  $\mathcal{K} = \langle AP, W, R, L \rangle$ , a world  $w \in W$ , a natural number  $g \in \mathbb{N} \setminus \{0\}$ , and a GCTL\* path formula  $\psi$ , we have:

1.  $\mathcal{K}, w \models E^{\geq g}\psi$  iff  $|\min(P_A(\psi, w))| \geq g$ ;
2.  $\mathcal{K}, w \models A^{<g}\psi$  iff  $|\min(\text{Pth}^\infty(w) \setminus P_E(\psi, w))| < g$ ;

where  $P_A(\psi, w) = \{\pi \in \text{Pth}^\infty(w) \mid \forall \pi' \in \text{Pth}^\infty(w). \pi \leq \pi' \implies \mathcal{K}, \pi', 0 \models \psi\}$  and  $P_E(\psi, w) = \{\pi \in \text{Pth}^\infty(w) \mid \exists \pi' \in \text{Pth}^\infty(w) : \pi \leq \pi' \wedge \mathcal{K}, \pi', 0 \not\models \psi\}$ .

Note that GCTL\* formulas with degrees 1 are CTL\* formulas. Moreover, the above definition of  $P_A(\psi, w)$  and  $P_E(\psi, w)$  formally states that they are dual of each other, i.e.,  $P_A(\psi, w) = \text{Pth}^\infty(w) \setminus P_E(\neg\psi, w)$ .

In the rest of the paper, we only consider formulas in *positive normal form* (pnf, for short), i.e., the negation is applied only to atomic propositions. Under this assumption, we consider  $\neg\varphi$  as the pnf formula equivalent to the negation of  $\varphi$ . Moreover, we only consider formulas that do not contain any subformula of the form  $E^{\geq g}\bar{X}\varphi$  or  $A^{<g}X\varphi$ . This can be done w.l.o.g. since each formula can be linearly translated into another formula not containing the above quantifications (see [2] for more). Finally, as abbreviation we use the boolean values  $\text{t}$  (“true”) and  $\text{f}$  (“false”).

We now give the formal definition of *conservativeness* and then, by means of two examples, we clarify the need of the concepts of minimality and conservativeness. A path  $\pi$  of  $\mathcal{K}$  is conservative w.r.t. a path formula  $\psi$  iff, for all paths  $\pi'$  extending  $\pi$ , i.e.,

with  $\pi \leq \pi'$ , it holds that  $\mathcal{K}, \pi', 0 \models \psi$ . Note that this concept of conservativeness is automatically embedded in the definition of the set  $P_A(\psi, w)$ , since we consider only paths  $\pi \in \text{Pth}^\infty(w)$  that, if extended, continue to satisfy the formula  $\psi$ . Now, for the minimality, consider a finite tree  $\mathcal{T}$  having just three nodes all labeled by  $p$ , the root and two of its successors. Also, consider the formula  $\phi = E^{\geq 2}Fp$ . Because of the minimality, the two paths of length two that satisfy  $Fp$  collapse into the path containing just the root, hence  $\mathcal{T} \not\models \phi$ . For the conservativeness, consider another tree  $\mathcal{T}'$  equal to  $\mathcal{T}$ , but with one of the two root successors not labeled with  $p$ . Also, consider the formula  $\phi' = E^{\geq 2}Gp$ . At this point, by the conservativeness, we have that  $\mathcal{T}' \not\models \phi'$  even if there are two paths satisfying the formula  $Gp$ , the root alone and its extension with one of the children, since the former is not conservative. Indeed, this path can be extended into a path that does not satisfy  $Gp$ .

Let  $\mathcal{K}$  be a Kripke structure and  $\phi$  be a GCTL\* formula. Then,  $\mathcal{K}$  is a *model* for  $\phi$ , denoted by  $\mathcal{K} \models \phi$ , iff there is  $w \in W$  such that  $\mathcal{K}, w \models \phi$ . In this case, we also say that  $\mathcal{K}$  is a model for  $\phi$  on  $w$ . A GCTL\* formula  $\phi$  is said *satisfiable* iff there exists a model for it. For all state formulas  $\phi_1$  and  $\phi_2$  (resp., path formulas  $\psi_1$  and  $\psi_2$ ), we say that  $\phi_1$  is *equivalent* to  $\phi_2$ , formally  $\phi_1 \equiv \phi_2$ , (resp.,  $\psi_1$  is *equivalent* to  $\psi_2$ , formally  $\psi_1 \equiv \psi_2$ ) iff for all Kripke structures  $\mathcal{K}$  and worlds  $w \in W$  it holds that  $\mathcal{K}, w \models \phi_1$  iff  $\mathcal{K}, w \models \phi_2$  (resp.,  $\min(P_A(\psi_1, w)) = \min(P_A(\psi_2, w))$ ).

The following lemma shows two exponential fixed point equivalences that extend to “graded” formulas the correspondign well-known result for “ungraded” formulas. These interesting equivalences among GCTL\* formulas, are useful to describe important properties of the GCTL semantics.

**Lemma 1 ([2]).** *Let  $\phi_1$  and  $\phi_2$  be state formulas,  $g > 1$ , and  $\text{ex}\langle \psi, g \rangle = \bigvee_{\{c_i\} \in C(g)} \bigwedge_{i=1}^g E^{\geq c_i} X E^{\geq i} \psi$ . Then, the following equivalences hold:*

1.  $E\phi_1 \cup \phi_2 \equiv \phi_2 \vee \phi_1 \wedge \text{ex}\langle \phi_1 \cup \phi_2, 1 \rangle$
2.  $E^{\geq g} \phi_1 \cup \phi_2 \equiv \neg \phi_2 \wedge \phi_1 \wedge \text{ex}\langle \phi_1 \cup \phi_2, g \rangle$
3.  $E\phi_1 R \phi_2 \equiv \phi_2 \wedge (\phi_1 \vee E \bar{X} \bar{f} \vee \text{ex}\langle \phi_1 R \phi_2, 1 \rangle)$
4.  $E^{\geq g} \phi_1 R \phi_2 \equiv \phi_2 \wedge \neg \phi_1 \wedge E X E \neg (\phi_1 R \phi_2) \wedge \text{ex}\langle \phi_1 R \phi_2, g \rangle$

Finally, by using a simple proof by induction, it is possible to show that GCTL\* is invariant under the unwinding of a model. Hence, the next theorem follows.

**Theorem 1.** *GCTL\* satisfies the tree model property.*

## 4 Succinctness

In this section, we show that binary GCTL is at least exponentially more succinct than binary  $G\mu$ -CALCULUS. We prove the statement by showing a class of GCTL formulas  $\phi_g$  whose minimal equivalent  $G\mu$ -CALCULUS formulas  $\chi_g$  needs to be in size exponentially bigger than (the size of)  $\phi_g$ . Classical techniques ([15, 16, 25]) rely on the fact that in the more succinct logic there exists a formula having a *least finite model* whose size is double exponential in the size of the formula, while in the less succinct logic every formula has finite models of size at most exponential in its size. Unfortunately, in our case we cannot apply this idea, since, as far as we know, both GCTL and the

$G\mu$ -CALCULUS satisfy the small model property, i.e., all their satisfiable formulas have always a model at most exponential in their size. To prove the succinctness of GCTL, hence, we explore a technique based on a characteristic property of our logic. Specifically, it is based on the fact that, using GCTL, we can write a set of formulas  $\varphi_g$  each one having a number of “characterizing models” that is exponential in the degree  $g$  of  $\varphi_g$ , while every  $G\mu$ -CALCULUS formula has at most a polynomial number of those models in its degree.

Consider the property “in a tree, there are exactly  $g$  grandchildren of the root labeled with  $p$  and having only one path leading from them, while all other nodes are not”. Such a property can be easily described by the GCTL formula  $\varphi_g = \varphi' \wedge \varphi''$ , where  $\varphi' = \neg p \wedge AX(\neg p \wedge AX(p \wedge AXAG(\neg p \wedge A^{<2}\tilde{X}f)))$  and  $\varphi'' = E^{=g}F p$ . Its size is  $\Theta(\lceil \log(g+1) \rceil)$ . We claim that a  $G\mu$ -CALCULUS formula  $\chi_g$  requires exponential size to express the same property. More formally, our aim is to prove the following theorem.

**Theorem 2.** *Let  $\varphi_g = (E^{=g}F p) \wedge \varphi'$ , with  $g \in \mathbb{N}$  and  $\varphi' = \neg p \wedge AX(\neg p \wedge AX(p \wedge AXAG(\neg p \wedge A^{<2}\tilde{X}f)))$ . Then, each  $G\mu$ -CALCULUS formula  $\chi_g$  equivalent to  $\varphi_g$  has size  $\Omega(2^{\|\varphi_g\|})$ .*

The proof of this theorem proceeds directly by proving the following lemma and observing that, since  $\|\varphi_g\| = \Theta(\lceil \log(g+1) \rceil)$ , we can easily derive that  $\|\chi_g\| = \Omega(2^{\|\varphi_g\|})$ .

**Lemma 2.** *Every  $G\mu$ -CALCULUS formula  $\chi_g$  equivalent to  $\varphi_g$  is of size  $\Omega(g)$ .*

## 5 GCTL Satisfiability

In this section, we describe the satisfiability procedure for GCTL. As we discussed in the introduction, we exploit an automata-theoretic approach by using satellites that are used to accept binary tree-encodings of tree models of a formula. So, we first introduce the automata model, then we discuss the binary tree encoding, and finally, we show how to build the automaton accepting all tree-model encodings of a given formula.

*Tree automata with satellites.* Alternating tree automata (ATA) [18] are a generalization of nondeterministic tree automata. Intuitively, on visiting a node of the input tree, while the latter sends exactly one copy of itself to each of the successors of the node, an ATA can send several copies of itself to the same successor. As a generalization of ATA, here we consider *alternating tree automata with satellites* (ATAS), in a similar way it has been done in [12], with the main difference that our satellites are nondeterministic and can work on trees and not only on words. The satellite is used to ensure that the input tree satisfies some structural properties and it is kept apart from the main automaton as it allows to show a tight complexity for the satisfiability problem. The formal definitions follow.

**Definition 3.** *An ATA is a tuple  $\mathcal{A} = \langle \Sigma, \Delta, Q, \delta, q_0, F \rangle$ , where  $\Sigma$ ,  $\Delta$ , and  $Q$  are non-empty finite sets of input symbols, directions, and states, respectively,  $q_0 \in Q$  is an initial state,  $F$  is an acceptance condition to be defined later, and  $\delta : Q \times \Sigma \mapsto B^+(\Delta \times Q)$  is an alternating transition function that maps a state and an input symbol to a positive boolean combination of moves in  $\Delta \times Q$ .*

**Definition 4.** A run of an ATA  $\mathcal{A}$  on a  $\Sigma$ -labeled  $\Delta$ -Tree  $\mathcal{T} = \langle T, \nu \rangle$  is a  $(Q \times T)$ -labeled  $\mathbb{N}$ -tree  $\langle \text{Tr}, r \rangle$  such that (i)  $r(\varepsilon) = (q_0, \varepsilon)$  and (ii) for all  $y \in \text{Tr}$  with  $r(y) = (q, x)$ , there is a set  $S \subseteq \Delta \times Q$  with  $S \models \delta(q, \nu(x))$ , such that, for all  $(i, q') \in S$ , there is a  $j \in \mathbb{N}$  for which  $r(y \cdot j) = (q', x \cdot i)$  holds.

In the following, we consider ATA along with the Büchi acceptance condition  $F \subseteq Q$  (ABT) (see [13] for more). By  $\mathcal{L}(\mathcal{A})$  we denote the language accepted by the automaton  $\mathcal{A}$ , i.e., the set of all trees  $\mathcal{T}$  accepted by  $\mathcal{A}$ . Moreover,  $\mathcal{A}$  is said to be *empty* if  $\mathcal{L}(\mathcal{A}) = \emptyset$ . The emptiness problem for  $\mathcal{A}$  is to decide whether  $\mathcal{L}(\mathcal{A}) = \emptyset$ .

We now define automata with satellite.

**Definition 5.** An ATAS is a tuple  $\langle \mathcal{A}, S \rangle$ , where  $\mathcal{A} = \langle \Sigma \times P', \Delta, Q, \delta, q_0, F \rangle$  is an ATA and  $S = \langle \Sigma, \Delta, P, \zeta, P_0 \rangle$  is a satellite, where  $P = P' \times P''$  is a non-empty finite set of states,  $P_0 \subseteq P$  is a set of initial states, and  $\zeta : P \times \Sigma \mapsto 2^{P^\Delta}$  is a nondeterministic transition function that maps a state and an input symbol to a set of functions from directions to states.

For the coming definition we need an extra notation. Given a  $(\Sigma' \times \Sigma'')$ -labeled  $\Delta$ -tree  $\mathcal{T} = \langle T, \nu \rangle$ , we define the *projection* of  $\mathcal{T}$  on  $\Sigma'$  as the  $\Sigma'$ -labeled  $\Delta$ -tree  $\mathcal{T}' = \langle T, \nu' \rangle$  such that, for all nodes  $x \in T$ , we have  $\nu(x) = (\nu'(x), \sigma)$ , for some  $\sigma \in \Sigma''$ .

**Definition 6.** A tree  $\mathcal{T}$  is accepted by an ATAS  $\langle \mathcal{A}, S \rangle$ , where  $\mathcal{A} = \langle \Sigma \times P', \Delta, Q, \delta, q_0, F \rangle$ ,  $S = \langle \Sigma, \Delta, P, \zeta, P_0 \rangle$ , and  $P = P' \times P''$ , iff there exists a run  $\mathcal{R}$  of  $S$  on  $\mathcal{T}$ , i.e., a  $(\Sigma \times P)$ -labeled  $\Delta$ -tree, whose projection on  $\Sigma \times P'$  is accepted by the ATA  $\mathcal{A}$ .

In words, first the satellite  $S$  guesses and adds to the input tree  $\mathcal{T}$  and additional labeling on the set  $P'$ , thus returning the augmented tree  $\mathcal{R}$ . Then, the main automaton  $\mathcal{A}$  computes a new run on  $\mathcal{R}$  as input.

In the following, we also consider ATAS along with the Büchi condition (ABTS).

Note that satellites are just a convenient way to describe an ATA in which the state space can be partitioned into two components, one of which is nondeterministic and independent from the other, and has no influence on the acceptance. Indeed, it is just a matter of technicality to see that automata with satellites inherit all the closure properties of alternating automata. In particular, the following theorem, directly derived by the proof idea of [12], shows how the separation between  $\mathcal{A}$  and  $S$  gives a tight analysis of the complexity of the relative emptiness problem.

**Theorem 3.** The emptiness problem for an ABTS  $\langle \mathcal{A}, S \rangle$ , where  $\mathcal{A}$  has  $n$  states and  $d$  directions and  $S$  has  $m$  states, can be decided in time  $2^{O(n \cdot \log(m) \cdot d)}$ .

*Binary tree model encoding.* As first step, we define the infinite widening of a formula tree model  $\mathcal{T}$ , i.e., a transformation that, taken  $\mathcal{T}$ , returns a full infinite tree  $\mathcal{T}_W$  having infinite branching degree and embedding  $\mathcal{T}$ . This transformation ensures that in  $\mathcal{T}$  all nodes have the same branching degree and all branches are infinite. To this aim, we use a fresh label  $\#$  as described in the following definition.

**Definition 7 (Infinite Widening).** Let  $\mathcal{T} = \langle T, \nu \rangle$  be a  $\Sigma$ -labeled  $\Delta$ -tree, with  $\Delta \subseteq \mathbb{N}$  and such that  $\# \notin \Sigma$ . Then, the infinite widening of  $\mathcal{T}$  is the  $\Sigma_W$ -labeled  $\mathbb{N}$ -tree  $\mathcal{T}_W = \langle \mathbb{N}^*, \nu_W \rangle$  such that (i)  $\Sigma_W = \Sigma \cup \{\#\}$ , (ii) for  $x \in T$ ,  $\nu_W(x) = \nu(x)$ , and (iii) for  $y \in \mathbb{N}^* \setminus T$ ,  $\nu_W(y) = \#$ .

Now, we define a sharp transformation of  $\mathcal{T}_W$  in a full binary tree  $\mathcal{T}_D$ . This is inspired but different from that used to embed the logic  $S\omega S$  into  $S2S$  [21]. Intuitively, the transformation allows to delay  $n$  decisions, to be taken at a node  $y$  in  $\mathcal{T}_W$  and corresponding to its successors  $y \cdot i$ , along some corresponding nodes  $x, x \cdot 0, x \cdot 00, \dots$  in  $\mathcal{T}_D$ . In particular, when we are on a node  $x \cdot 0^i$ , we are able to split the decision on  $y \cdot i$  into an immediate action, which is sent to the right (effective) successor  $x \cdot 0^i \cdot 1$ , while the remaining actions are sent to its copy  $x \cdot 0^{i+1}$ . To differentiate the meaning of left and right successors we use the fresh symbol  $\perp$ .

**Definition 8 (Delayed Generation).** *Let  $\mathcal{T}_W = \langle \mathbb{N}^*, \nu_W \rangle$  be the infinite widening of a  $\Sigma$ -labeled tree  $\mathcal{T}$  such that  $\perp \notin \Sigma$ . Then, the delayed generation of  $\mathcal{T}$  is the  $\Sigma_D$ -labeled  $\{0, 1\}$ -tree  $\mathcal{T}_D = \langle \{0, 1\}^*, \nu_D \rangle$  such that (i)  $\Sigma_D = \Sigma_W \cup \{\perp\}$  and (ii) there exists a surjective function  $f : \{0, 1\}^* \mapsto \mathbb{N}^*$ , with  $f(\varepsilon) = \varepsilon$ ,  $f(x \cdot 0^i) = f(x)$ , and  $f(x \cdot 0^i \cdot 1) = f(x) \cdot i$ , where  $x \in \{0, 1\}^*$  and  $i \in \mathbb{N}$ , such that (ii.i)  $\nu_D(x) = \nu_W(f(x))$ , for all  $x \in \{\varepsilon\} \cup \{0, 1\}^* \cdot \{1\}$ , and (ii.ii) if  $\nu_W(f(x)) \neq \#$ , then either  $\nu_D(x \cdot 0) = \perp$  and  $\nu_D(x \cdot 1) \neq \#$  or  $\nu_D(x \cdot 0) = \nu_D(x \cdot 1) = \#$ , else  $\nu_D(x \cdot 0) = \#$ , for all  $x \in \{0, 1\}^*$ .*

To complete the tree encoding, we have also to delay the degree associated to each node in the input tree model. We recall that, an original tree model of a graded formula may require a fixed number of paths satisfying the formula going through the same node. Such a number is the degree associated to that node and which we need to delay. To this aim, we enrich the label of a node with a function mapping a set of elements, named *bases*, into triples of numbers representing the splitting of the node degree into two components. The first is the delayed degree, while the second is the degree associated to one of the effective successors of the node. Such a splitting is the delayed action mentioned above customized to the need of having information on the degrees. This is formalized in the following four definitions.

**Definition 9 (( $\Sigma, B$ )-Enriched  $g$ -Degree Tree).** *Let  $\Sigma$  and  $B$  be two sets,  $g \in \mathbb{N}$ , and  $H(g) \subset \mathbb{N}^3$  be the set of triples  $(d, d_1, d_2)$  such that  $d = d_1 + d_2 \leq g$ . Then, a  $(\Sigma, B)$ -enriched  $g$ -degree tree is a  $(\Sigma \times H(g)^B)$ -labeled  $\{0, 1\}$ -tree  $\mathcal{T} = \langle \{0, 1\}^*, \nu \rangle$ .*

We now introduce  $(\Sigma_D, B)$ -enriched  $g$ -degree trees  $\mathcal{T}_{D, B, g}$  as the extension of the delayed generation  $\mathcal{T}_D$  of  $\mathcal{T}$  with degree functions in its labeling. Intuitively, each function in a node represents how to distribute and propagate an information on the degrees along its successors.

**Definition 10 ( $B$ -Based  $g$ -Degree Delayed Generation).** *Let  $B$  be a set,  $g \in \mathbb{N}$ , and  $\mathcal{T}_D = \langle \{0, 1\}^*, \nu_D \rangle$  be the delayed generation of a  $\Sigma$ -labeled tree  $\mathcal{T}$ . Then, the  $B$ -based  $g$ -degree delayed generation of  $\mathcal{T}$  is the  $(\Sigma_D, B)$ -enriched  $g$ -degree tree  $\mathcal{T}_{D, B, g} = \langle \{0, 1\}^*, \nu_{D, B, g} \rangle$  such that there is an  $h \in H(g)^B$  with  $\nu_{D, B, g}(x) = (\nu_D(x), h)$ , for all  $x \in \{0, 1\}^*$ .*

In order to have a sound construction for  $\mathcal{T}_{D, B, g}$ , we need to impose a coherence on the information between a node and its two successors. In particular, whenever we enter a node  $x$  labeled with  $\#$  in its first part, as it represents that the node is fictitious, we have to take no splitting of the degree by sending to  $x$  the value 0. On the other nodes, we have to match the value of the first (resp., second) component of the splitting with the degree of the left (resp., right) successor.

**Definition 11 (Sup/Inf Coherence).** Let  $\mathcal{T} = \langle \{0, 1\}^*, \nu \rangle$  be a  $(\Sigma \cup \{\#\}, B)$ -enriched  $g$ -degree tree. Then,  $\mathcal{T}$  is superiorly (resp., inferiorly) coherent w.r.t. a base  $b \in B$  iff, for  $x \in \{0, 1\}^*$  and  $i \in \{0, 1\}$  with  $\nu(x) = (\sigma, h)$ ,  $h(b) = (d, d_0, d_1)$ ,  $\nu(x \cdot i) = (\sigma_i, h_i)$ , and  $h_i(b) = (d^i, d_0^i, d_1^i)$ , it holds that (i) if  $\sigma_i = \#$  then  $d_i = 0$  and (ii)  $d_i \leq d^i$  (resp.,  $d_i \geq d^i$ ).

Finally, with the following definition, we extend the local concept of sup/inf coherence of a particular base to a pair of sets of bases.

**Definition 12 (Full Coherence).** A  $(\Sigma \cup \{\#\}, B)$ -enriched  $g$ -degree tree  $\mathcal{T}$  is full coherent w.r.t. a pair  $(B_{\text{sup}}, B_{\text{inf}})$ , where  $B_{\text{sup}} \cup B_{\text{inf}} \subseteq B$  iff it is superiorly (resp., inferiorly) coherent w.r.t. all bases  $b \in B_{\text{sup}}$  (resp.,  $b \in B_{\text{inf}}$ ).

*The coherent structure satellites.* We now define the satellites we use to verify that the tree encoding the model of the formula has a correct shape w.r.t. the whole transformation described in the previous paragraph. In particular, we first introduce a satellite that checks if the “enriched degree tree” in input is the result of a “based degree delayed generation” of the model of the formula. Then, we show how to create the additional labeling of the tree that satisfies the coherence properties on the degrees required by the semantics of the logic. The following automaton checks if the  $\#$  and  $\perp$  labels of the input tree are correct w.r.t. Definitions 7 and 8.

**Definition 13 (Structure Satellite).** The structure satellite is the binary satellite  $\mathcal{S}^* = \langle \Sigma_D, \{0, 1\}, \{\#, \perp, @\}, \zeta, \{@\} \rangle$ , where  $\zeta$  is as follows: if  $p = \sigma = \#$  then  $\zeta(p, \sigma) = \{(\#, \#)\}$  else if either  $p = \sigma = \perp$  or  $p = @$  and  $\sigma \in \Sigma$  then  $\zeta(p, \sigma) = \{(\perp, @), (\#, \#)\}$ , otherwise  $\zeta(p, \sigma) = \emptyset$ .

The satellite  $\mathcal{S}^*$  has constant size 3. Its transition function  $\zeta$  is defined to directly represent the constraints on the  $\#$  and  $\perp$  labels, so the next lemma easily follows.

**Lemma 3.** The satellite  $\mathcal{S}^*$  accepts all and only the  $\Sigma_D$ -labeled  $\{0, 1\}$ -trees  $\mathcal{T}_D$  that can be obtained as delayed generation of  $\Sigma$ -labeled trees  $\mathcal{T}$ .

The next satellite creates the additional labeling of the input tree of the main automaton in such a way that it is sup/inf coherent by using the properties (i) and (ii) of Definition 11. Precisely, if the satellite accepts the input tree, the additional labeling is given by its states.

**Definition 14 (Sup/Inf Coherence Satellite).** The  $b$ -base  $g$ -degree sup (resp., inf) coherence satellite is the binary satellite  $\mathcal{S}_{b,g}^\Sigma = \langle \Sigma \cup \{\#\}, \{0, 1\}, H(g), \zeta, H(g) \rangle$ , where  $\zeta$  is as follows: (i) if  $\sigma = \#$  then  $\zeta(p, \sigma)$  is set to  $\{(p, p)\}$  if  $p = (0, 0, 0)$  and to  $\emptyset$  otherwise; (ii) if  $\sigma \neq \#$  then  $\zeta(p, \sigma)$  contains all and only the pairs of states  $(p_0, p_1) \in H(g)^{\{0,1\}}$  with  $p_i = (d^i, d_0^i, d_1^i)$ , such that  $d_i \leq d^i$  (resp.,  $d_i \geq d^i$ ), for all  $i \in \{0, 1\}$ , where  $p = (d, d_0, d_1)$ .

Note that the satellite  $\mathcal{S}_{b,g}^\Sigma$  has size quadratic in its degree  $g$ .

Finally, we introduce the satellite that checks if the tree in input is coherent or not by merging the behavior of the two previous described satellites.

**Definition 15 (Coherent Structure Satellite).** *The  $g$ -degree structure  $(B_{\text{sup}}, B_{\text{inf}})$ -coherent satellite is the binary satellite  $\mathcal{S}_g^{B_{\text{sup}}, B_{\text{inf}}} = \langle \Sigma_D, \{0, 1\}, H(g)^{(B_{\text{sup}} \cup B_{\text{inf}})} \times \{\#, \perp, @\}, \zeta, H(g)^{(B_{\text{sup}} \cup B_{\text{inf}})} \times \{@\} \rangle$  obtained as the product of the structure satellite  $\mathcal{S}^*$  with all the  $b$ -base  $g$ -degree sup (resp., inf) coherence satellites  $\mathcal{S}_{b,g}^{\Sigma \cup \{\perp\}}$ , with  $b \in B_{\text{sup}}$  (resp.,  $b \in B_{\text{inf}}$ ).*

Clearly, the size of  $\mathcal{S}_g^{B_{\text{sup}}, B_{\text{inf}}}$  is polynomial in  $g$  and exponential in  $|B_{\text{sup}} \cup B_{\text{inf}}|$ , since its number of states is equal to  $\frac{3}{2}((g+1)(g+2))^{|B_{\text{sup}} \cup B_{\text{inf}}|}$ . Due to the product structure of the automaton, next result directly follows from Lemma 3, and Definitions 10 and 12.

**Theorem 4.** *The main automaton  $\mathcal{A}$  of an ATAS  $\langle \mathcal{A}, \mathcal{S}_g^{B_{\text{sup}}, B_{\text{inf}}} \rangle$  having satellite state space  $P = P' \times P''$ , with  $P' = H(g)^B$  and  $P'' = \{\#, \perp, @\}$ , accepts only the  $B$ -based  $g$ -degree delayed generation  $\mathcal{T}_{D_{B,g}}$  of  $\Sigma$ -labeled trees  $\mathcal{T}$  that are coherent w.r.t. the pair  $(B_{\text{sup}}, B_{\text{inf}})$ , with  $B = B_{\text{sup}} \cup B_{\text{inf}}$ .*

*The formula automaton.* In this paragraph we introduce a Büchi tree automaton  $\mathcal{A}_\varphi$  that checks whether a full bounded-width tree  $\mathcal{T}$  satisfies a given formula  $\varphi$  by evaluating all  $B$ -based  $g$ -degree delayed generation trees  $\mathcal{T}_{D_{B,g}}$  associated with  $\mathcal{T}$ , where  $g = \hat{\varphi}$  is the maximum degree of  $\varphi$  and  $B = \text{ecl}(\varphi)$  is the extended closure of  $\varphi$ . The automaton works on any  $B$ -based  $g$ -degree generation tree, even those that are not associated to a full bounded-width tree. However, we make the assumptions that the trees in input are really associated to this kind of trees and that they are coherent with respect to  $(B_{\text{sup}}, B_{\text{inf}})$ , where  $B_{\text{sup}} = \text{ecl}(\varphi)^\exists$  and  $B_{\text{inf}} = \text{ecl}(\varphi)^\forall$ . By Theorem 4, we are able to enforce such properties by using  $\mathcal{A}_\varphi$  as a part of an ATAS having the  $g$ -degree structure  $(B_{\text{sup}}, B_{\text{inf}})$ -coherent satellite  $\mathcal{S}_g^{B_{\text{sup}}, B_{\text{inf}}}$ .

In order to understand how the formula automaton works, it is useful to gain more insights on the meaning of the tree  $\mathcal{T}_{D_{B,g}}$  associated with  $\mathcal{T}$ . First of all, the widening operation has the purpose to make the tree complete by adding fake nodes labeled with  $\#$ . Through this, we obtain the tree  $\mathcal{T}_W$ . Then, the delaying operation transforms  $\mathcal{T}_W$  into a binary tree  $\mathcal{T}_D$ , such that at every level a node  $x$  associated to a node  $y$  in  $\mathcal{T}$  generates only one of the successor of  $y$  in the direction 1, meanwhile it sends a duplicate of itself on the direction 0 labeled with  $\perp$ . The following duplicates have to generate the remaining successors in a recursive way. However, when there are no more successors to generate, the node  $x$  does not send in the direction 0 a duplicate of itself anymore, but just a fake node labeled with  $\#$ . At this point, to obtain the tree  $\mathcal{T}_{D_{B,g}}$ , we enrich the labeling of the delayed generation tree, by adding a degree function  $h: B \mapsto H(g)$ . In the hypothesis that  $\mathcal{T}$  satisfies  $\varphi$ , for every formula  $\varphi' \in B$  and node  $x \in \{0, 1\}^*$  with  $\nu_{D_{B,g}}(x) = (\sigma, h)$ , we have that  $h(\varphi') = (d, d_0, d_1)$  describes the degree with which the formula  $\varphi'$  is supposed to be satisfied on  $x$ . In particular  $d$  is the degree in the current node, and  $d_0 + d_1$  explains how this degree is partitioned in the following nodes. More precisely  $d_1$  represents the degree sent to the direction 1, which usually corresponds to a concrete node in  $\mathcal{T}$ , hence it is the degree sent to that node. Meanwhile,  $d_0$  represents the degree sent to the direction 0, which usually corresponds to a duplicate of the previous node. Hence  $d_0$  represents the degree that had yet to be partitioned among the remaining successors of the node  $y$  associated to  $x$ . To this aim, the coherence requirement asks:

(i) for an existential formula, the degree found in a successor node is not lower than the degree the father sent to that node (it may be higher as the node may satisfy the existence by finding more paths with a certain property, so it surely satisfies what the formula requires); (ii) for a universal formula, the degree found in a successor node is not greater than the degree the father sent to that node (it may be smaller as the node may satisfy the universality by finding less paths with a certain negated property, so it surely satisfies what the formula requires).

In the hypothesis of coherence, the formula automaton needs only to check that (i) the degree of every existential and universal formula is initiated correctly on the node in which the formula first appears in (e.g., for an existential formula it needs to check that the degree in the label of the node is not lower than the degree required by the formula), and (ii) that every node of the tree satisfies the existential or universal formula with the degree specified in the node labeling. To do this, the automaton has as state space  $\text{cl}(\varphi) \cup \text{ecl}(\varphi) \cup \text{rcl}(\varphi)$ : on one hand, the formulas in  $\text{ecl}(\varphi)$  ask the automaton to verify them completely relying on the degree of the label, on the other hand, the existential and universal formulas in  $\text{cl}(\varphi)$  ask the automaton even to check that the degree of them agrees with that contained in the label. Finally, states in  $\text{rcl}(\varphi)$  are used for the Büchi acceptance condition.

**Definition 16 (Formula Automaton).** *The formula automaton for  $\varphi$  is the binary ABT  $\mathcal{A}_\varphi = \langle \Sigma_\varphi \times P'_\varphi, \{0, 1\}, Q_\varphi, \delta, \varphi, F_\varphi \rangle$ , where  $\Sigma_\varphi = 2^{\text{AP}} \cup \{\#, \perp\}$ ,  $P'_\varphi = H(\hat{\varphi})^{\text{ecl}(\varphi)}$ ,  $Q_\varphi = \text{cl}(\varphi) \cup \text{ecl}(\varphi) \cup \text{rcl}(\varphi)$ ,  $F_\varphi = \text{rcl}(\varphi)$ , and  $\delta : Q_\varphi \times (\Sigma_\varphi \times P'_\varphi) \mapsto \mathbb{B}^+(\{0, 1\} \times Q_\varphi)$  is described in the body of the article.*

We now describe the structure of the transition transition  $\delta(q, (\sigma, h))$  through a case analysis on the state space.

As first thing, when  $\sigma = \#$ , the automaton is on a fake node  $x = x' \cdot i$  of the the input tree  $\mathcal{T}_{D_{B,g}}$ , so every formula should be false on it. However, in the instant the automaton reaches such a node, by passing through its antecedent  $x'$ , it is not asking to verify the formula represented by the state  $q$ . Indeed, we have that it is sent by another state  $q'$  on  $x'$  which corresponds to a universal formula. In that case, we are checking that the “core” of it is satisfied on all the successors (but a given number of them). Hence, since  $x$  does not exist in the original tree  $\mathcal{T}$ , we do not have to verify the property of  $q$  on it. Moreover, we are sure that  $q'$  does not represent any existential property. This is due to the fact that (i) the degree  $d_i$  related to the state  $q'$  in the labeling of  $x'$  needs to be 0 by the coherence requirements of Definition 11 and (ii) that, as we show later, the transition on existential formulas do not send any new state to a direction  $j$  having  $d_j = 0$ . For this reason, we set  $\delta(q, (\sigma, h)) = \text{t}$ . In the rest of this paragraph, we suppose  $\sigma \neq \#$ .

As we show later, the structure of the transition function does not allow to reach, at the same time, a state  $q$  belonging to the set  $\text{cl}(\varphi) \cup \text{rcl}(\varphi)$  and a labeling  $\sigma = \perp$ . For this reason, w.l.o.g., we can set  $\delta(q, (\sigma, h)) = \text{f}$ .

Due to the lack of space, here we partially describe the transition function of  $\mathcal{A}_\varphi$  (i.e., the cases EX and EU), and send the reader to the full version for the complete definition.

Let  $h(\text{EX } \varphi) = (d, d_0, d_1)$ . For a state of the form EX  $\varphi$ , we verify that this formula holds with degree  $d$ . Recall that in the input tree the degrees  $(d_0, d_1)$  describe the distribution of the nodes, which need to satisfy  $\varphi$ , among the successors of the current node.

Since the nodes on direction 1 are real successors of the node in the original input tree  $\mathcal{T}$ , we need that the state formula  $\varphi$  holds on them iff  $d_1 = 1$ . However, we cannot ask that a state formula holds more than one time, so, if  $d_1 > 1$ , the input tree cannot be accepted. Finally, on direction 0, we send the same state  $\text{EX } \varphi$  if  $d_0 > 0$ , in order to ask that the residual degree  $d_0$  is distributed on the remaining successors. For a state of the form  $\text{E}^{\geq i} \text{X } \varphi$  we have only to further verify that the degree  $i$  agrees with the value  $d$ , i.e.,  $d \geq i$ . Hence,  $\delta(\text{E}^{\geq i} \text{X } \varphi, (\sigma, h))$  is set to  $\text{f}$ , if  $d < i$ , and to  $\delta(\text{EX } \varphi, (\sigma, h))$ , otherwise.

Let now  $h(\text{E}\varphi_1 \cup \varphi_2) = (d, d_0, d_1)$ . For a state of the form  $\text{E}\varphi_1 \cup \varphi_2$  we check that this formula holds with degree  $d$ . If the node is not a duplicate of a previous node, i.e.,  $\sigma \neq \perp$ , we check the formula that should hold in the current node by applying the one-step unfolding property derived by the semantics (see Lemma 1). If  $d = 1$ , then  $\text{E}\varphi_1 \cup \varphi_2$  may be satisfied on the current node. Indeed, if  $\varphi_2$  is true on it, we already have one and only one minimal path satisfying  $\varphi_1 \cup \varphi_2$ . Otherwise, we need  $\varphi_1$  to holds. If  $d > 1$ , we have to force  $\varphi_1 \cup \varphi_2$  to be not yet satisfied. So, we require  $\neg\varphi_2$  to holds in the current node. These two cases, may also require  $\text{E}\varphi_1 \cup \varphi_2$  to hold on some of the successors, so we may need to propagate the formula itself, by using the requirement  $\gamma = (0, \text{E}\varphi_1 \cup \varphi_2) \wedge (1, \text{E}\varphi_1 \cup \varphi_2)$ . In the case  $d = 0$ , we do not require anything, so we set the transition function to be  $\text{t}$ . On the other hand, if  $\sigma = \perp$ , we do not have to verify again what is check on a previous node, but only to propagate the formula on both the directions 0 and 1, by using the  $\gamma$  requirement. As for the EX case, for a state of the form  $\text{E}^{\geq i} \varphi_1 \cup \varphi_2$  we have to do the same consideration on the relation between the degrees  $i$  and  $d$ . Moreover, we do not have to deal with the  $\perp$  labeling, since the current state is never sent through the 0 direction. Hence, we do not consider it as a separate case.

We now briefly discuss the acceptance condition for the illustrated cases. In the full version we show that  $F_\varphi$  equals to  $\text{rcl}(\varphi)$ . Here, we only show that the states EX and EU discussed above are not contained in  $F_\varphi$ . First note that states EX may generate themselves only along direction 0 but, in the hypothesis that  $\mathcal{T}$  has bounded-width, this cannot occur infinitely often. Indeed,  $\mathcal{T}_{D_{B,g}}$  can only have as many duplicates of a node  $x$  in the direction 0 labeled with  $\perp$  as the number of successors of the related node in  $\mathcal{T}$ . Thus, along a direction 0, the automaton eventually meets a node labeled with  $\#$  that does not allow the states to further propagate. For the same reason, also the states EU cannot progress infinitely often along direction 0. As concern direction 1, a state EU needs to follow the branches on  $\mathcal{T}_{D_{B,g}}$  that satisfy the until. Since, on those branches, it needs to be satisfied within a finite path, we cannot allow it to progress infinitely often. Consequently, we do not add EU in the acceptance set  $F_\varphi$ .

**Theorem 5.** *Let  $\varphi$  be a GCTL formula, with  $g = \hat{\phi}$ ,  $B_{\text{sup}} = \text{ecl}(\varphi)^\exists$ , and  $B_{\text{inf}} = \text{ecl}(\varphi)^\forall$ . Then,  $\varphi$  is satisfiable iff  $\mathcal{L}(\langle \mathcal{A}_\varphi, \mathcal{S}_g^{B_{\text{sup}}, B_{\text{inf}}} \rangle) \neq \emptyset$ .*

By a matter of calculation, it holds that  $|\mathcal{A}_\varphi| = O(|\varphi|)$  and  $|\mathcal{S}_g^{B_{\text{sup}}, B_{\text{inf}}}| = \hat{\phi}^{O(|\varphi|)}$ . By Theorem 3, we obtain that the emptiness problem for  $\langle \mathcal{A}_\varphi, \mathcal{S}_g^{B_{\text{sup}}, B_{\text{inf}}} \rangle$  can be solved in time  $2^{O(|\varphi|^2 \cdot \log(\hat{\phi}))} \leq 2^{O(\|\varphi\|^3)}$ . Moreover, by recalling that GCTL subsumes CTL, the following result follows.

**Theorem 6.** *The satisfiability problem for GCTL with binary codings is EXPTIME-COMplete.*

## References

- [1] Apostol, T.M.: Introduction to Analytic Number Theory. Springer, Heidelberg (1976)
- [2] Bianco, A., Mogavero, F., Murano, A.: Graded Computation Tree Logic. In: LICS 2009, pp. 342–351 (2009)
- [3] Bonatti, P.A., Lutz, C., Murano, A., Vardi, M.Y.: The Complexity of Enriched  $\mu$ -Calculi. LMCS 4(3:11), 1–27 (2008)
- [4] Clarke, E.M., Emerson, E.A.: Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic. In: Kozen, D. (ed.) Logic of Programs 1981. LNCS, vol. 131, pp. 52–71. Springer, Heidelberg (1982)
- [5] Emerson, E.A., Halpern, J.Y.: “Sometimes” and “Not Never” Revisited: On Branching Versus Linear Time. JACM 33(1), 151–178 (1986)
- [6] Ferrante, A., Napoli, M., Parente, M.: CTL Model-Checking with Graded Quantifiers. In: Cha, S(S.), Choi, J.-Y., Kim, M., Lee, I., Viswanathan, M. (eds.) ATVA 2008. LNCS, vol. 5311, pp. 18–32. Springer, Heidelberg (2008)
- [7] Ferrante, A., Napoli, M., Parente, M.: Graded-CTL: Satisfiability and Symbolic Model Checking. In: Breitman, K., Cavalcanti, A. (eds.) ICFEM 2009. LNCS, vol. 5885, pp. 306–325. Springer, Heidelberg (2009)
- [8] Fine, K.: In So Many Possible Worlds. NDJFL 13, 516–520 (1972)
- [9] De Giacomo, G., Lenzerini, M.: Concept Language with Number Restrictions and Fix-points, and its Relationship with Mu-calculus. In: ECAI 1994, pp. 411–415 (1994)
- [10] Grädel, E.: On The Restraining Power of Guards. JSL 64(4), 1719–1742 (1999)
- [11] Kupferman, O., Sattler, U., Vardi, M.Y.: The Complexity of the Graded  $\mu$ -Calculus. In: Voronkov, A. (ed.) CADE 2002. LNCS (LNAI), vol. 2392, pp. 423–437. Springer, Heidelberg (2002)
- [12] Kupferman, O., Vardi, M.Y.: Memoryful Branching-Time Logic. In: LICS 2006, pp. 265–274. IEEE Computer S., Los Alamitos (2006)
- [13] Kupferman, O., Vardi, M.Y., Wolper, P.: An Automata Theoretic Approach to Branching-Time Model Checking. JACM 47(2), 312–360 (2000)
- [14] Lamport, L.: “Sometime” is Sometimes “Not Never”: On the Temporal Logic of Programs. In: POPL 1980, pp. 174–185 (1980)
- [15] Lange, M.: A Purely Model-Theoretic Proof of the Exponential Succinctness Gap between CTL+ and CTL. IPL 108(5), 308–312 (2008)
- [16] Lutz, C.: Complexity and Succinctness of Public Announcement Logic. In: AAMAS 2006, pp. 137–143 (2006)
- [17] Miyano, S., Hayashi, T.: Alternating Finite Automata on  $\omega$ -Words. TCS 32, 321–330 (1984)
- [18] Muller, D.E., Schupp, P.E.: Alternating Automata on Infinite Trees. TCS 54(2-3), 267–276 (1987)
- [19] Pnueli, A.: The Temporal Logic of Programs. In: FOCS 1977, pp. 46–57 (1977)
- [20] Pnueli, A.: The Temporal Semantics of Concurrent Programs. TCS 13, 45–60 (1981)
- [21] Rabin, M.O.: Decidability of Second-Order Theories and Automata on Infinite Trees. BAMS 74, 1025–1029 (1968)
- [22] Tobies, S.: PSpace Reasoning for Graded Modal Logics. JLC 11(1), 85–106 (2001)
- [23] Vardi, M.Y., Wolper, P.: An Automata-Theoretic Approach to Automatic Program Verification. In: LICS 1986, pp. 332–344 (1986)
- [24] Vardi, M.Y., Wolper, P.: Automata-Theoretic Techniques for Modal Logics of Programs. JCSS 32(2), 183–221 (1986)
- [25] Wilke, T.: CTL+ is Exponentially More Succinct than CTL. In: Pandu Rangan, C., Raman, V., Sarukkai, S. (eds.) FST TCS 1999. LNCS, vol. 1738, pp. 110–121. Springer, Heidelberg (1999)