IOS Press

Model Checking for Graded CTL

Alessandro Ferrante
Dip.to di Informatica ed Applicazioni
Università di Salerno
ferrante@dia.unisa.it

Margherita Napoli
Dip.to di Informatica ed Applicazioni
Università di Salerno
napoli@dia.unisa.it

Mimmo Parente
Dip.to di Informatica ed Applicazioni
Università di Salerno
parente@dia.unisa.it

Abstract. Recently, complexity issues related to the decidability of the μ-calculus, when the universal and existential quantifiers are augmented with *graded modalities*, have been investigated by Kupfermann, Sattler and Vardi ([19]). Graded modalities refer to the use of the universal and existential quantifiers with the added capability to express the concept of *at least* k or *all but* k, for a non-negative integer k. In this paper we study the Computational Tree Logic CTL, a branching time extension of classical modal logic, augmented with graded modalities and investigate the complexity issues with respect to the model-checking problem. We consider a system model represented by a Kripke structure $\mathcal{K}$ and give an algorithm to solve the model-checking problem running in time $O(|\mathcal{K}| \cdot |\varphi|)$ which is hence tight for the problem (here $|\varphi|$ is the number of temporal and boolean operators and does not include the values occurring in the graded modalities). In this framework, the graded modalities express the ability to generate a user-defined number of counterexamples to a specification φ given in CTL. However, these multiple counterexamples can partially overlap, that is they may share some behavior. We have hence investigated the case when all of them are completely disjoint. In this case we prove that the model-checking problem is both NP-hard and coNP-hard and give an algorithm for solving it running in polynomial space. We have thus studied a fragment of graded-CTL, and have proved that the model-checking problem is solvable in polynomial time.

Keywords: Model-checking, Computational Tree Logic, Graded Modalities.

Address for correspondence: Dipartimento di Informatica ed Applicazioni "R.M. Capocelli", Università di Salerno
Via Ponte don Melillo - 84084 - Fisciano (SA) - Italy
A preliminary version of the paper appeared in the proceedings of ATVA '08 [12], Automated Technology for Verification and Analysis, 6th International Symposium.

1. Introduction

Computational Tree Logic, CTL, is a well known branching-time temporal (modal) logic which let us reason about how the behaviour of a system can evolve over time by considering either *all the possible futures* or *at least one possible future*. Here we introduce the *graded*-CTL, obtained by augmenting CTL with graded modalities that allow to reason about either *at least* or *all but* any number of futures. In literature, the capability to express *at least* k and *all but* k, has been intensively studied in various logic frameworks. In classical logics $\exists^{>k}$ and $\forall^{\leq k}$ are called *counting quantifiers*, see e.g. [16, 15, 22], in modal logics they are called *graded* modalities, see e.g. [14, 24], and in description logics one speaks about *number restriction* of properties describing systems, see e.g. [17].

Recently complexity issues related to the satisfiability problem for the μ-calculus when the universal and existential quantifiers are augmented with graded modalities, have been investigated in [19]. They have shown that this problem is EXPTIME-complete, retaining thus the same complexity as in the case of classical μ-calculus, though strictly extending it.

Here we consider model-checking problems using formulas expressed in graded-CTL. Model-checking is an established process to check whether a model, representing a system, satisfies a given logical formula, which expresses some behaviors of the system [5, 23]. The focus of the paper is on the complexities involved in the process of model-checking system models against specifications given in this new logic. For example, the formula $\exists^{>k}\mathcal{F}\neg(wait \Rightarrow \forall\mathcal{F}criticSection)$ expresses the fact that in several cases it is possible that a waiting process never obtains the requested resource. This new logic allows us also to use *nested* graded quantifiers to express other interesting properties, such as the safety property that "a system always has at least two ways to reach a *safe state*" ($\forall\mathcal{G}\exists^{>1}\mathcal{F}$ safe). Clearly formulas of this type cannot be expressed in CTL and not even in classical μ-calculus.

The motivation in the use of these graded modalities mainly arises from the fact that during the verification of a system design, a central feature of the technique of model-checking is the generation of counterexamples. In fact the realization process for a system passes through the "Check/Analyze/Fix" cycle: model-check the design of the system against some desired properties φ, analyze the generated counterexamples to the properties, and re-design the system, trying to fix the errors. The analysis of the counterexamples usually gives clues to that part of the system model where the specification failed. It is therefore highly desirable to have as many significative counterexamples as possible simultaneously, c.f. [3, 9, 11]. Up-to-date model-checkers, as NuSMV and SPIN [4, 18], return only one counterexample of φ. Our aim here is first to efficiently get an answer to whether there are or not more counterexamples (without explicitly generating them) and then getting more *significative* ones, in the sense that by nesting the graded quantifiers we can concentrate ourselves on more interesting zones of the model. Actual model checkers could generate more counterexamples to a formula, but in a "blind" way, that is without user guidance. On the other side, the investigation of the complexities involved in the generation and the analysis of the counterexamples is a central issue, as explained also in the survey [7] where the role and the structure of counterexamples is investigated putting an emphasis on the complexities related to the generation problem.

Given a graded-CTL formula φ and a system model represented by a Kripke structure $\mathcal{K}$, our first result is an algorithm to solve the model-checking problem in time $O(|R| \cdot |\varphi|)$, the same running time of the algorithm for classical CTL. Let us remark that this complexity does not depend at all on the values representing the grading of the modalities in fact the size $|\varphi|$ of the formula does not depend on the representation of these values and is simply the number of the temporal and boolean operators.

The existential and universal quantifiers are dual operators, but differently from what happens in classical CTL, in our logic to rewrite equivalent formulas in terms of the dual quantifier implies an increase of the size of the same formula. Thus to show the above result we actually have first given an algorithm for formulas with only existential graded quantifiers and then have proved how to deal with universal graded quantifier to retain the same complexity.

However, the multiple counterexamples returned by this algorithm may partially overlap, while it can be desirable in the analysis phase to detect completely independent traces where the specification fails. To deal with this case, we have introduced a semantic for temporal operators to require *edge-disjointness* of the paths representing the counterexamples. The same setting can be applied also, for example, to ensure that a “correct” system behavior tolerates a given number of faults of the system. We have proved that to model-check a system model against such specifications is both NP-hard and coNP-hard. The reduction has been done from the cycle-packing problem (the problem to check whether there are k disjoint cycles in a graph). This has suggested that formulas expressing the existence of at least k infinite edge-disjoint paths globally satisfying φ) are *hard* to verify. We have then defined the still interesting fragment of the logic obtained by dropping this kind of formulas and proved that the model-checking problem can be solved in polynomial time in this case. In the full graded logic, unless NP $=$ coNP, the problem does not belong to NP. We have thus given an algorithm for the fragment, showing that however it is in PSPACE. Finally, we have considered the scenario in which only a given number of behaviors need to be disjoint and all the remaining may overlap. In this case we have proved that the problem is *fixed parameter* tractable.

The paper is organized as follows: in Section 2 we recall the basic definition and results of CTL; in Section 3 we introduce graded-CTL and define the model-checking problem for it; in Section 4 we prove that this problem is solvable in polynomial time; in Section 5 we study the edge-disjoint graded-CTL model-checking problem. Moreover, we show that the same problem restricted to a fragment of graded-CTL is solvable in polynomial time, and that we can obtain a good algorithm for practical cases by relaxing the edge-disjointness requirement; finally in Section 6 we give some conclusions and open problems.

2. Computation Tree Logic

The temporal logic CTL [5] is a branching-time logic in which each temporal operator, expressing properties about a possible future, has to be preceded by a quantifier that specifies in how many possible futures the property has to hold. So, in CTL one can express properties that have to be true either *immediately after now* ($\mathcal{X}$), or *each time from now* ($\mathcal{G}$), or *from now until something happens* ($\mathcal{U}$), and it is possible to specify that each property must hold either in *some possible futures* (E) or in *each possible future* (A). Formally, given a finite set of *atomic propositions* AP, CTL is the set of formulas φ defined as follows:

$$\varphi := p \mid \neg\psi_1 \mid \psi_1 \wedge \psi_2 \mid E\mathcal{X}\psi_1 \mid E\mathcal{G}\psi_1 \mid E\psi_1\mathcal{U}\psi_2$$

where $p \in AP$ is an atomic proposition and ψ_1 and ψ_2 are CTL formulas.

The semantics of a CTL formula is defined with respect to a *Kripke Structure* by means of the classical relation $\models$. As usual, a Kripke structure over a set of atomic propositions AP, is a tuple $\mathcal{K} = \langle S, s_{in}, R, L\rangle$, where S is a finite set of states, $s_{in} \in S$ is the initial state, $R \subseteq S \times S$ is a transition relation with the property that for each $s \in S$ there is $t \in S$ such that $(s,t) \in R$, and $L : S \rightarrow 2^{AP}$ is a labeling function.

A path in $\mathcal{K}$ is denoted by the sequence of states $\pi = \langle s_0, s_1, \ldots s_n\rangle$ or by $\pi = \langle s_0, s_1, \ldots\rangle$, if it is infinite. The length of a path, denoted by $|\pi|$, is the number of states in the sequence, and $\pi[i]$ denotes the i-th state s_i.

Then, the relation $\models$ for a state $s \in S$ of $\mathcal{K}$ is iteratively defined as follows:

- $(\mathcal{K}, s) \models p \in AP$ iff $p \in L(s)$;
- $(\mathcal{K}, s) \models \neg\psi_1$ iff $\neg((\mathcal{K}, s) \models \psi_1)$ (in short, $(\mathcal{K}, s) \not\models \psi_1$);
- $(\mathcal{K}, s) \models \psi_1 \wedge \psi_2$ iff $(\mathcal{K}, s) \models \psi_1$ and $(\mathcal{K}, s) \models \psi_2$;
- $(\mathcal{K}, s) \models E\mathcal{X}\psi_1$ iff there exists $s' \in S$ such that $(s, s') \in R$ and $(\mathcal{K}, s') \models \psi_1$ (the path $\langle s, s'\rangle$ is called an *evidence* of the formula $\mathcal{X}\psi_1$);
- $(\mathcal{K}, s) \models E\mathcal{G}\psi_1$ iff there exists an infinite path π starting from s (i.e., $\pi[0] = s$) such that for all $j \geq 0$, $(\mathcal{K}, \pi[j]) \models \psi_1$ (the path π is called an *evidence* of the formula $\mathcal{G}\psi_1$);
- $(\mathcal{K}, s) \models E\psi_1\mathcal{U}\psi_2$ iff there exists a finite path π with length $|\pi| = r + 1$ starting from s such that $(\mathcal{K}, \pi[r]) \models \psi_2$ and, for all $0 \leq j < r$, $(\mathcal{K}, \pi[j]) \models \psi_1$ (the path π is called an *evidence* of the formula $\psi_1\mathcal{U}\psi_2$);

We say that a Kripke structure $\mathcal{K} = \langle S, s_{in}, R, L\rangle$ *models* a CTL formula φ iff $(\mathcal{K}, s_{in}) \models \varphi$.

Note that the CTL formulas are *state-formulas*, with the meaning that they have to be satisfied in a single state of the system, while formulas of the form $\mathcal{X}\psi_1$, $\mathcal{G}\psi_1$ and $\psi_1\mathcal{U}\psi_2$ are called *path-formulas* (denoted generically in the rest of the paper with the symbol θ).

Observe that we have expressed the syntax of CTL with one of the possible minimal sets of operators. Other temporal operators and the universal path quantifier A, in fact, can be easily derived from those. For example, the classical abbreviation $\mathcal{F}$ (*eventually*) can be expressed as $\mathcal{F}\psi_1 = \text{TRUE}\ \mathcal{U}\ \psi_1$, and for the universal path quantifier we have that $A\mathcal{X}\psi_1 = \neg E\mathcal{X}\neg\psi_1$, $A\mathcal{G}\psi_1 = \neg E\mathcal{F}\neg\psi_1$, $A\psi_1\mathcal{U}\psi_2 = (\neg E\mathcal{G}\neg\psi_2) \wedge (\neg E(\neg\psi_2)\mathcal{U}(\neg\psi_1 \wedge \neg\psi_2))$.

The **CTL model-checking** is the problem of verifying whether a Kripke structure $\mathcal{K}$ models a CTL formula φ. It is known that the CTL model-checking problem can be solved in linear time, as asserted in the following theorem.

Theorem 2.1. ([8])
Let $\mathcal{K} = \langle S, s_{in}, R, L\rangle$ be a Kripke structure and φ be a CTL formula. The CTL model-checking problem can be solved in time $\mathcal{O}(|R| \cdot |\varphi|)$.

3. Graded-CTL

In this section we introduce the graded-CTL which extends the classical CTL by adding graded modalities on the quantifier operators. As we have seen in the previous section, classical CTL can be used for reasoning about the temporal behavior of systems considering either “all the possible futures” or “at least one possible future”. Graded modalities generalize CTL allowing to reason about more than a given number of possible distinct future behaviors. Let us first define the notion of *distinct*.

Let $\mathcal{K} = \langle S, s_{in}, R, L\rangle$ be a Kripke structure. We say that two paths π_1 and π_2 on $\mathcal{K}$ are *distinct* if there exists an index $0 \leq i < \min\{|\pi_1|, |\pi_2|\}$ such that $\pi_1[i] \neq \pi_2[i]$. Observe that from this definition if a path is the prefix of another path, then they are not distinct.

We introduce the *graded existential path quantifier* $E^{>k}$, that requires the existence of $k+1$ pairwise distinct evidences of a path-formula. Therefore, given a set of atomic proposition AP, the syntax of graded-CTL is defined as follows:

$$\varphi := p \mid \neg\psi_1 \mid \psi_1 \wedge \psi_2 \mid E^{>k}\mathcal{X}\psi_1 \mid E^{>k}\mathcal{G}\psi_1 \mid E^{>k}\psi_1\mathcal{U}\psi_2$$

where $p \in AP$, k is a non-negative integer and ψ_1 and ψ_2 are graded-CTL formulas.

The semantics of graded-CTL is still defined with respect to a Kripke structure $\mathcal{K} = \langle S, s_{in}, R, L\rangle$ on the set of atomic propositions AP. In particular, for formulas of the form p, $\neg\psi_1$ and $\psi_1 \wedge \psi_2$ the semantics is the same as in the classical CTL (see Section 2). For the remaining formulas, the semantics is defined as follows:

- $(\mathcal{K}, s) \models E^{>k}\theta$, with $k \geq 0$ and either $\theta = \mathcal{X}\psi_1$ or $\theta = \mathcal{G}\psi_1$ or $\theta = \psi_1\mathcal{U}\psi_2$, iff there exist $k+1$ pairwise distinct evidences of θ starting from s (note that $E^{>0}\theta$ is equivalent to $E\theta$).

It is easy to observe that classical CTL is a proper fragment of graded-CTL since the simple graded formula $E^{>1}\mathcal{X}p$ cannot be expressed in CTL, whereas any CTL formula is also a graded-CTL formula.

We also consider the graded extension of the universal quantifier, $A^{\leq k}$, with the meaning that *all the paths starting from a node s, but at most k pairwise distinct paths, are evidences of a given path-formula*. The quantifier $A^{\leq k}$ is the dual operator of $E^{>k}$ and can obviously be re-written in terms of $\neg E^{>k}$. However, while $A^{\leq k}\mathcal{X}\psi_1$ and $A^{\leq k}\mathcal{G}\psi_1$ can be easily re-written respectively as $\neg E^{>k}\mathcal{X}\neg\psi_1$ and $\neg E^{>k}\mathcal{F}\neg\psi_1$, the transformation of the formula $A^{\leq k}\psi_1\mathcal{U}\psi_2$ with $k > 0$ in terms of $\neg E^{>k}$ deserves more attention. In fact, we have that $A^{\leq k}\psi_1\mathcal{U}\psi_2$ is equivalent to $\neg E^{>k}\neg(\psi_1\mathcal{U}\psi_2)$ (note that this formula is not a graded-CTL formula because of the occurrence of the innermost negation), that can be translated in graded-CTL in the following way:

$$\begin{aligned} A^{\leq k}\psi_1\mathcal{U}\psi_2 \iff\ & \neg E^{>k}\mathcal{G}(\psi_1 \wedge \neg\psi_2)\ \wedge \neg E^{>k}(\psi_1 \wedge \neg\psi_2)\mathcal{U}(\neg\psi_1 \wedge \neg\psi_2) \wedge \\ & \bigwedge_{i=0}^{k-1}(\ \neg E^{>k-1-i}\mathcal{G}(\psi_1 \wedge \neg\psi_2)\ \vee\ \neg E^{>i}(\psi_1 \wedge \neg\psi_2)\mathcal{U}(\neg\psi_1 \wedge \neg\psi_2)\ ) \end{aligned} \quad (1)$$

In fact observe that a path not satisfying $\psi_1\mathcal{U}\psi_2$ is a path that satisfies either $\theta_1 = \mathcal{G}(\psi_1 \wedge \neg\psi_2)$ or $\theta_2 = (\psi_1 \wedge \neg\psi_2)\mathcal{U}(\neg\psi_1 \wedge \neg\psi_2)$ (clearly, the paths satisfying θ_1 are all distinct from the paths satisfying θ_2). Therefore the formula $E^{>k}\neg(\psi_1\mathcal{U}\psi_2)$ holds in s, if $k+1$ pairwise distinct paths stem from this, each satisfying either θ_1 or θ_2.

The **graded-CTL model-checking** is the problem of verifying whether a Kripke structure $\mathcal{K}$ models a graded-CTL formula φ. In the next sections we study the complexity of the graded-CTL model-checking problem with respect to the size of the Kripke structure (expressed in terms of the number of edges, as by our definition $|R| \geq |S|$), and to the size of the graded-CTL formula, where the size $|\varphi|$ of a graded-CTL formula is the number of the temporal and the boolean operators occurring in it.

4. Graded-CTL Model-Checking

In this section we solve the graded-CTL model-checking problem by showing an algorithm running in linear time in the size of both the Kripke structure and the formula φ. It is important to note here that this complexity is independent from the values occurring in the graded quantifiers of φ. We then discuss possible applications of model-checking system properties using graded-CTL formulas to generate more than one counterexample. We will give an example of a mutual-exclusion system for which we model-check some liveness properties.

Let us first give a definition that we will use in the next proofs. Given a graph, a *sink-cycle* is a cycle not containing nodes with out-degree greater than 1, called *exit nodes*.

Now we prove two technical lemmas, that will be exploited in the main theorem of the section. Let $\mathcal{K} = \langle S, s_{in}, R, L\rangle$ be a Kripke structure.

Lemma 4.1. Let $\psi = E^{>k}\mathcal{G}\psi_1$, $k > 0$, and G_ψ be the graph induced by the states of $\mathcal{K}$ where $E^{>0}\mathcal{G}\psi_1$ holds. Then, given a state s in G_ψ, $(\mathcal{K}, s) \models \psi$ iff either there is a *non-sink-cycle* reachable from s, or there are $k + 1$ pairwise distinct finite paths connecting s to *sink-cycles* in G_ψ.

Proof:
(if): Let s be a state in G_ψ and $C = \langle v_0, \ldots, v_{h-1}\rangle$ be a non-sink-cycle in G_ψ reachable from s via a finite path $\langle s, u_0, \ldots, u_i, v_0\rangle$ in G_ψ. Consider an exit-node, say v_j, $0 \le j \le h-1$ and a node w_0 in G_ψ such that $w_0 \neq v_{(j+1) \bmod h}$ and (v_j, w_0) is an edge of G_ψ. Since $(\mathcal{K}, w_0) \models E^{>0}\mathcal{G}\psi_1$, there is an infinite path $\langle w_0, w_1, \ldots\rangle$ starting from w_0 and satisfying $\mathcal{G}\psi_1$. There are $k + 1$ pairwise distinct infinite paths π_l, $0 \le l \le k$, each satisfying $\mathcal{G}\psi_1$, defined as $\pi_l = \langle s, u_0, \ldots, u_i, (C)^l, v_0, \ldots, v_j, w_0, \ldots\rangle$, where $(C)^l$ denotes the fact that π_l cycles l times on C. Thus $(\mathcal{K}, s) \models \psi$. Finally, suppose there are $k + 1$ pairwise distinct finite paths connecting s to sink-cycles in G_ψ. Since each of these such paths constitutes an infinite path satisfying $\mathcal{G}\psi_1$, then $(\mathcal{K}, s) \models \psi$.
(only if): If $(\mathcal{K}, s) \models \psi$ then there are $k + 1$ pairwise distinct infinite paths $\pi_0, \ldots, \pi_k$ starting from s and satisfying $\mathcal{G}\psi_1$. Since an infinite path on a Kripke structure either contains a non-sink-cycle, or ends in a sink-cycle, the lemma follows from the fact that each state in $\pi_0, \ldots, \pi_k$ belongs to G_ψ. □

Lemma 4.2. Let $\psi = E^{>k}\psi_1\mathcal{U}\psi_2$, $k > 0$, and G_ψ be the graph induced by considering the states of $\mathcal{K}$ where $E^{>0}\psi_1\mathcal{U}\psi_2$ holds and by deleting the edges outgoing from states where ψ_1 is not satisfied. Then, given a state s in G_ψ, $(\mathcal{K}, s) \models \psi$ iff either there is a *non-sink-cycle* reachable from s, or there are $k + 1$ pairwise distinct finite simple paths from s to states where ψ_2 holds.

Proof:
(if): Let s be a state in G_ψ and $C = \langle v_0, \ldots, v_{h-1}\rangle$ be a non-sink-cycle in G_ψ, reachable from s via a finite path $\langle s, u_0, \ldots, u_i, v_0\rangle$ in G_ψ. Consider an exit-node v_j, for $0 \le j \le h-1$, and a node w_0 in G_ψ such that $w_0 \neq v_{(j+1) \bmod h}$ and (v_j, w_0) is an edge in G_ψ. Since $(\mathcal{K}, w_0) \models E^{>0}\psi_1\mathcal{U}\psi_2$, then in G_ψ there is a finite path $\langle w_0, \ldots, w_r\rangle$ starting from w_0 and ending in a w_r such that $(\mathcal{K}, w_r) \models \psi_2$. Consider the $k+1$ pairwise distinct finite paths π_l, $0 \le l \le k$, defined as $\pi_l = \langle s, u_0, \ldots, u_i, (C)^l, v_0, \ldots, v_j, w_0, \ldots w_r\rangle$, where $(C)^l$ denotes the fact that π_l cycles l times on C. Since G_ψ does not contain edges out-going from nodes where ψ_1 is not satisfied, then $(\mathcal{K}, x) \models \psi_1$ for all x in π_l, except at most w_r, and therefore each π_l is an *evidence* of $\psi_1\mathcal{U}\psi_2$. Thus $(\mathcal{K}, s) \models \psi$. Now, let $\pi_0, \ldots, \pi_k$ be $k + 1$ pairwise distinct finite simple

paths (i.e. paths without cycles) connecting s to nodes where ψ_2 holds; from the definition of G_ψ, π_i is an *evidence* of $\psi_1 \mathcal{U} \psi_2$ for all $0 \leq i \leq k$, and therefore $(\mathcal{K}, s) \models \psi$.
(only if): If $(\mathcal{K}, s) \models \psi$ then there are $k+1$ pairwise distinct finite paths $\pi_0, \ldots, \pi_k$ starting from s and ending in states satisfying ψ_2, and these are also paths in G_ψ. If the paths are either all simple or one of them contains a non-sink cycle, then the proof is complete. Otherwise let π'_i $(0 \leq i \leq k)$ be the longest simple prefix of π_i ending in a state where ψ_2 holds. Obviously, π'_i satisfies $\psi_1 \mathcal{U} \psi_2$ and is distinct from π'_j for all $j \neq i$ and this completes the proof of the lemma. □

An example of a system where conditions of Lemma 4.2 hold, is the model of Figure 1. It satisfies the formula $E^{>k}\mathcal{F}(wait1 \wedge E\mathcal{G}\neg critic1)$, for all $k \geq 0$, as it contains reachable non-sink-cycles (one is depicted with bold-faced edges).

Now we give the main result of the section showing that the graded-CTL model-checking can be solved in linear time independently from the values of the constants occurring in the formula.

Theorem 4.1. Let $\mathcal{K} = \langle S, s_{in}, R, L \rangle$ be a Kripke structure and φ be a graded-CTL formula. The graded-CTL model-checking problem can be solved in time $\mathcal{O}(|R| \cdot |\varphi|)$.

Proof:
To solve the model-checking problem for a given Kripke structure $\mathcal{K}$ and a given formula φ an algorithm has to compute the subset $\{s \in S \text{ s.t. } (\mathcal{K}, s) \models \varphi\}$. Our algorithm works on the sub-formulas ψ of φ and for each state s determines whether $(\mathcal{K}, s) \models \psi$ (and sets a boolean variable $s.\psi$ to TRUE), (see Algorithm 1). The algorithm uses a primitive function Sub which returns all the sub-formulas of a given formula φ and moreover for a path-formula θ, if $E^{>k}\theta$ is in $Sub(\varphi)$, then $E^{>0}\theta$ is in $Sub(\varphi)$ as well. In particular we assume that such formulas are returned in non-decreasing order of complexity, with $E^{>0}\theta$ preceding $E^{>k}\theta$ in the sequence.

If a sub-formula ψ is of type $p \in AP$, $\neg\psi_1$, $\psi_1 \wedge \psi_2$, $E^{>0}\mathcal{G}\psi_1$, $E^{>0}\psi_1\mathcal{U}\psi_2$, then the algorithm (lines $3-13$) works as the classical CTL model-checking algorithm (see e.g. [6]), and, if a sub-formula is of type $E^{>k}\mathcal{X}\psi_1$, then the algorithm checks, for each state s whether $|\{t \in S \mid (s,t) \in R$ and $(\mathcal{K}, t) \models \psi_1\}| > k$, (lines $14-16$).

Consider now a sub-formula $\psi = E^{>k}\mathcal{G}\psi_1$ with $k > 0$ (line 17). This case is based on Lemma 4.1 In fact the algorithm looks for the states in $G_\psi = (S', R')$ (as defined in the statement of the lemma) from which it is possible to reach a non-sink-cycle (line 19) and then looks for the states from which $k+1$ pairwise distinct finite paths start, each ending in sink-cycles (line 21), after the deletion of the nodes from which it is possible to reach a non-sink-cycle (line 20).

Let us now consider a sub-formula $\psi = E^{>k}\psi_1\mathcal{U}\psi_2$ (line 23). In this case, the algorithm is based on Lemma 4.2. In fact here too, similarly to what has been done for the case of the operator $\mathcal{G}$, the algorithm looks for the states in $G_\psi = (S', R')$ (as defined in the statement of the lemma) from which it is possible to reach a non-sink-cycle (line 26), and then deletes these nodes (line 27). Finally it looks for the states from which $k+1$ pairwise distinct finite paths start, each ending in states where ψ_2 holds, (line 28),

The proof of the correctness of the algorithm can be easily done by induction on the length of the formulas. Let us now evaluate the running-time of the algorithm. It is easy to see that to check a sub-formula of type $p \in AP$, $\neg\psi_1$, $\psi_1 \wedge \psi_2$, requires $\mathcal{O}(|S|)$ and for a sub-formula $E^{>k}\mathcal{X}\psi_1$, $E^{>0}\mathcal{G}\psi_1$, $E^{>0}\psi_1\mathcal{U}\psi_2$ the algorithm requires time $\mathcal{O}(|R|)$. For a sub-formula $E^{>k}\mathcal{G}\psi_1$, note that the set of nodes from which it is possible to reach a non-sink-cycle can be globally calculated in time $\mathcal{O}(|R|)$ by using

Algorithm 1: The algorithm $GradedCTL(\mathcal{K}, \varphi)$.

```
Input: A Kripke Structure $\mathcal{K} = \langle S, s_{in}, R, L\rangle$ and a graded-CTL formula $\varphi$.
Output: For each state $s$, $s.\varphi = $ TRUE if $(\mathcal{K}, s) \models \varphi$ and $s.\varphi = $ FALSE otherwise

Let $s.\psi = $ FALSE for all $s \in S$ and $\psi \in Sub(\varphi)$;
foreach $\psi \in Sub(\varphi)$ do
    case $\psi = p \in AP$: foreach $s \in S$ s.t. $p \in L(s)$ do $s.\psi \leftarrow$ TRUE;
    case $\psi = \neg\psi_1$: foreach $s \in S$ do $s.\psi \leftarrow \neg s.\psi_1$;
    case $\psi = \psi_1 \wedge \psi_2$: foreach $s \in S$ do $s.\psi \leftarrow (s.\psi_1 \wedge s.\psi_2)$;
    case $\psi = E^{>0}\mathcal{G}\psi_1$:
        $S' \leftarrow \{s \in S \mid s.\psi_1 = $ TRUE$\}$; $R' \leftarrow R \cap (S' \times S')$;
        foreach $s \in S'$ s.t. $\exists$ a cycle reachable from $s$ in $(S', R')$ do $s.\psi \leftarrow$ TRUE;
    end
    case $\psi = E^{>0}\psi_1\mathcal{U}\psi_2$:
        $S' \leftarrow \{s \in S \mid s.\psi_1 = $ TRUE or $s.\psi_2 = $ TRUE$\}$; $R' \leftarrow R \cap (S' \times S')$;
        foreach $s \in S'$ s.t. $\exists\, t$ with $t.\psi_2 = $ TRUE reachable from $s$ in $(S', R')$ do $s.\psi \leftarrow$ TRUE;
    end
    case $\psi = E^{>k}\mathcal{X}\psi_1$ with $k \geq 0$:
        foreach $s \in S$ s.t. $|\{(s,t) \in R \mid t.\psi_1 = $ TRUE$\}| > k$ do $s.\psi \leftarrow$ TRUE;
    end
    case $\psi = E^{>k}\mathcal{G}\psi_1$ with $k > 0$:
        $S' \leftarrow \{s \in S \mid s.E^{>0}\mathcal{G}\psi_1 = $ TRUE$\}$; $R' \leftarrow R \cap (S' \times S')$;
        foreach $s \in S'$ s.t. $\exists$ a non-sink-cycle reachable from $s$ in $(S', R')$ do $s.\psi \leftarrow$ TRUE;
        $S' \leftarrow S' \setminus \{s \in S$ s.t. $s.\psi = $ TRUE$\}$; $R' \leftarrow R' \setminus \{(s,t)$ s.t. $s.\psi = $ TRUE or $t.\psi = $ TRUE$\}$;
        foreach $s \in S'$ s.t. $\exists\, k+1$ pairwise distinct finite paths from $s$ to sink-cycles in $(S', R')$
        do $s.\psi \leftarrow$ TRUE;
    end
    case $\psi = E^{>k}\psi_1\mathcal{U}\psi_2$ with $k > 0$:
        $S' \leftarrow \{s \in S \mid s.E^{>0}\psi_1\mathcal{U}\psi_2 = $ TRUE$\}$;
        $R' \leftarrow (R \cap (S' \times S')) \setminus \{(s,t) \in R \mid s.\psi_1 = $ FALSE$\}$;
        foreach $s \in S'$ s.t. $\exists$ a non-sink-cycle reachable from $s$ in $(S', R')$ do $s.\psi \leftarrow$ TRUE;
        $S' \leftarrow S' \setminus \{s \in S$ s.t. $s.\psi = $ TRUE$\}$; $R' \leftarrow R' \setminus \{(s,t)$ s.t. $s.\psi = $ TRUE or $t.\psi = $ TRUE$\}$;
        foreach $s \in S'$ s.t. $\exists\, k+1$ pairwise distinct finite paths from $s$ to states where $\psi_2$ holds in
        $(S', R')$ do $s.\psi \leftarrow$ TRUE;
    end
end
```

a Depth First Search algorithm and, as soon as a cycle is detected, checking whether the cycle contains an exit-node. Finally, also the set of nodes from which $k+1$ paths leading to sink-cycles exist, can be globally calculated in time $\mathcal{O}(|R|)$ by executing a standard DFS algorithm on the graph with no nodes from which it is possible to reach a non-sink-cycle. The analysis for $E^{>k}\psi_1\mathcal{U}\psi_2$ is essentially the same as that of the case $E^{>k}\mathcal{G}\psi_1$. Since the size of $Sub(\varphi)$ is $O(|\varphi|)$, then the overall complexity of the algorithm is $\mathcal{O}(|R|\cdot|\varphi|)$. □

Let us observe that in section 3 we have presented an extension of our graded logic to include also the graded universal quantifier, $A^{\leq k}$. There we have shown that it is possible to express formulas containing $A^{\leq k}$ in terms of $\neg E^{>k}$, but the size of the transformed formula is increased by an extra factor proportional to k. Despite of this, in the following theorem we show that the model-checking problem for this extended logic can be still solved in time $\mathcal{O}(|R|\cdot|\varphi|)$ (here also the complexity is independent from the constants in the formula).

Theorem 4.2. Let $\mathcal{K}=\langle S, s_{in}, R, L\rangle$ be a Kripke structure and φ be a graded-CTL formula with possibly graded universal quantifiers. The graded-CTL model-checking problem can be solved in time $\mathcal{O}(|R|\cdot|\varphi|)$.

Proof:
Similarly to Theorem 4.1, to solve the model-checking problem for a given Kripke structure $\mathcal{K}$ and a given formula φ the algorithm works on the sub-formulas ψ of φ and for each state s determines whether $(\mathcal{K},s)\models\psi$. If either $\psi=A^{\leq k}\mathcal{X}\psi_1$ or $\psi=A^{\leq k}\mathcal{G}\psi_1$, then the problem can be solved with Algorithm 1 for the equivalent formula using the dual graded existential quantifier. Let $\psi=A^{\leq k}\psi_1\mathcal{U}\psi_2$ and consider a state $s\in S$. Let $\theta_1=\mathcal{G}(\psi_1\wedge\neg\psi_2)$, $\theta_2=(\psi_1\wedge\neg\psi_2)\mathcal{U}(\neg\psi_1\wedge\neg\psi_2)$ and $max_1(s)$ and $max_2(s)$ be defined as follows

$$\begin{aligned} max_1(s) &= \max\{0\leq i\leq k+1 \text{ s.t. from } s \text{ stem } i \text{ pairwise distinct evidences of } \theta_1\} \\ max_2(s) &= \max\{0\leq i\leq k+1 \text{ s.t. from } s \text{ stem } i \text{ pairwise distinct evidences of } \theta_2\}. \end{aligned}$$

From relation (1) given in section 3, it is easy to see that $(\mathcal{K},s)\models\psi$ iff $max_1(s)+max_2(s)\leq k$.

The lines $19-21$, $26-28$ of Algorithm 1 can be easily modified to calculate, in time $\mathcal{O}(|R|)$, the sets $M_1=\{max_1(s) \text{ s.t. } s\in S\}$ and $M_2=\{max_2(s) \text{ s.t. } s\in S\}$. To calculate M_1, in fact, consider the graph induced by the states where θ_1 holds. We first assign $max_1(s)=k+1$ for all the states from which it is possible to reach a non-sink-cycle and then we use a DFS on the remaining states to calculate the number of pairwise distinct paths from each state to sink-cycles. In a similar way we can calculate the set M_2. □

The graded-CTL model-checking can be used to obtain simultaneously more than one counterexample for a formula. For example, consider the formula $A\mathcal{F}p$ expressing a simple liveness property: in all behaviors something good eventually happens. Given a model $\mathcal{K}$, a counterexample is a path in $\mathcal{K}$ where $\neg p$ always holds. It can be useful to detect whether there are more than a fixed number k of behaviors in which the desired property fails. To get that, we can test whether $(\mathcal{K},s_{in})\models E^{>k}\mathcal{G}\neg p$. Analogously, we can consider a safety property expressed by $\neg E\mathcal{F}\ error$: once fixed a number k, if $(\mathcal{K},s_{in})\models E^{>k}\mathcal{F}\ error$ then there are more than k wrong behaviors, each leading to an error. Note that the algorithm we introduced in Theorem 4.1 can be modified to return the required counterexamples.

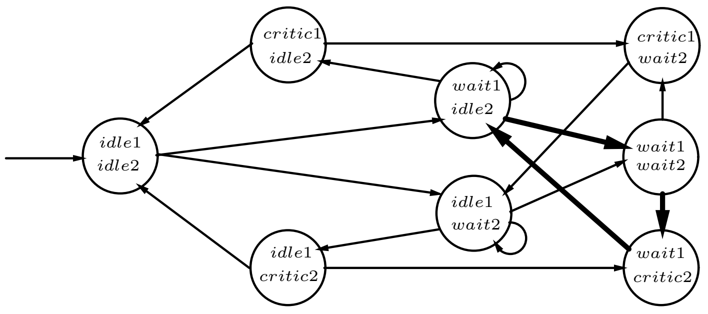

Figure 1. A mutual exclusion system.

Let us also consider the formula $A\mathcal{G}(wait \Rightarrow A\mathcal{F}critic)$ for the access control to a critical section of a system. A counterexample for this formula is an "unfair" path which is an evidence for the formula $E\mathcal{F}(wait \wedge E\mathcal{G}\neg critic)$. In this case, it is useful to detect whether the model can generate more bad behaviors. By using graded-CTL model-checking it is possible to analyze three bad situations: the first is to detect whether there are more "unfair" paths from the initial state, by verifying the formula $E^{>k_1}\mathcal{F}(wait \wedge E\mathcal{G}\neg critic)$; the second is to verify whether there is a finite path from the initial state to a state where $wait$ holds, from which more "unfair" paths stem, and this can be done by testing the formula $E\mathcal{F}(wait \wedge E^{>k_2}\mathcal{G}\neg critic)$, or, third, by using the formula $E^{>k_1}\mathcal{F}(wait \wedge E^{>k_2}\mathcal{G}\neg critic)$.

The following example shows the result of running NuSMV and SMV Cadence for a system model implementing mutual exclusion and having more than one unfair path.

Example 4.1. Consider the model in Figure 1 which violates the graded-CTL formula $\varphi = A^{\leq 1}\mathcal{G}(wait1 \Rightarrow A\mathcal{F}critic1)$.

When NuSMV (or also SMV Cadence [4, 21]) runs on this model and on the classical CTL formula corresponding to φ, then it generates as a counterexample the path:

$$\langle (idle1, idle2), (wait1, idle2), (wait1, idle2), \ldots \rangle$$

Then, if the user corrects this error by removing the self-loop on the state labeled $(wait1, idle2)$, the model-checker reports the second path

$$\langle (idle1, idle2), (wait1, idle2), (wait1, wait2), (wait1, critic2), (wait1, idle2), \ldots \rangle.$$

In practice most model-checkers implement *symbolic* algorithms which manipulates state sets represented by BDD. In [13] we give a symbolic algorithm for our setting (with both existential and universal quantifiers) whose complexity is $O(2^{|AP|} \cdot |S| \cdot k \cdot |\varphi|)$, where k is the maximum value occurring in φ. The extra factor k is due to the fact that when we consider state sets represented symbolically one has to take into account also all sub-formulas of the type $E^{>i}\theta$, $0 < i < k$, for each $E^{>k}\theta$ occurring in the given formula φ.

Theorem 4.3. ([13])
Let $\mathcal{K} = \langle S, s_{in}, R, L \rangle$ be a Kripke structure represented symbolically on a set of atomic propositions AP and let φ be a graded-CTL formula. The graded-CTL model-checking problem can be solved by a symbolic algorithm in time $\mathcal{O}(2^{|AP|} \cdot |S| \cdot k \cdot |\varphi|)$, where k is the maximum value occurring in φ.

In [13] some experimental results are reported by executing tests on the symbolic algorithm. The tests have shown that there is no increasing of both the time and the BDDs size, when compared with those of the classical symbolic-model checker NuSMV (see http://gradedctl.dia.unisa.it).

5. Edge-Disjoint Graded-CTL Model-Checking

In this section we introduce a different semantics of graded-CTL to distinguish whether different behaviors of the system, satisfying a graded-CTL formula, are completely disjoint. This setting can be applied also to ensure that a "correct" system behavior tolerates a given number of faults of the system.

The edge-disjoint semantics of graded-CTL is given by the relation $\models_{ed}$, which differs from the previous $\models$ relation only for the formulas of the following two types $E^{>k}\mathcal{G}\psi_1$ and $E^{>k}\psi_1\mathcal{U}\psi_2$. In these two cases it is required the edge-disjointness of the *evidences*, that is of the infinite paths satisfying $\mathcal{G}\psi_1$ and of the finite paths satisfying $\psi_1\mathcal{U}\psi_2$. Let us note that the model of Figure 1 does no longer satisfy the formula $E^{>2}\mathcal{F}(wait1 \wedge E\mathcal{G}\neg critic1)$ now as there are only two disjoint paths that violate the formula.

The **edge-disjoint graded-CTL model-checking** is defined as the problem of determining whether $(\mathcal{K}, s_{in}) \models_{ed} \varphi$, for a Kripke structure $\mathcal{K}$ with initial state s_{in} and a graded-CTL formula φ.

We first prove that the problem is both NP-hard and coNP-hard, and we give an upper bound showing that it lies in PSPACE. Then we introduce a fragment of our logic for which the problem has a polynomial time solution. To show this, we use techniques which are standards for flow network problems, see e.g. [10]. Finally we give a polynomial time algorithm for the case in which only a given number of single actions of behaviors (edges) must be disjoint and all the others may overlap. Note that this problem is a generalization both of the graded-CTL model-checking and of the edge-disjoint graded-CTL model-checking, since it is equivalent to the former (resp. to the latter) when no actions (all the actions) have to be disjoint.

5.1. Complexity

The proof of the hardness is given by a reduction from the Cycle-Packing problem, defined as follows: given a directed graph G and an integer $n \geq 2$, check whether in G there are at least n edge-disjoint cycles. The Cycle-Packing problem is known to be NP-complete (see [2]).

Theorem 5.1. The edge-disjoint graded-CTL model-checking problem is both NP-hard and coNP-hard.

Proof:
We first prove that edge-disjoint model-checking problem is NP-hard for specifications in the graded-CTL fragment $FRAG$ containing only formulas $E^{>k}\mathcal{G}p$, for an atomic proposition p and $k > 0$.

Given a graph $G = (\mathcal{V}, E)$ and an instance (G, n), $n \geq 2$, of the Cycle-Packing problem, let $\mathcal{K} = \langle \mathcal{V} \cup \{\hat{s}\}, \hat{s}, R, L\rangle$ be the Kripke structure obtained from G by adding an initial state $\hat{s} \notin \mathcal{V}$, connected to all the other nodes, and by labeling each state of $\mathcal{K}$ with a single atomic proposition p. Formally, $\mathcal{K}$ is defined on the atomic propositions $AP = \{p\}$ in such a way that $R = E \cup \{(\hat{s}, s)$ s.t. $s \in \mathcal{V}\}$ and $L(s) = \{p\}$ for all $s \in \mathcal{V} \cup \{\hat{s}\}$. Moreover, let us consider the graded-CTL formula $\varphi = E^{>n-1}\mathcal{G}p$. Since $\hat{s}$ is connected to each node of G and has no incoming edges, and since p holds in every node, then it follows that $(\mathcal{K}, \hat{s}) \models_{ed} \varphi$ iff G contains at least n edge-disjoint cycles. From the NP-hardness of the Cycle-Packing problem, the edge-disjoint FRAG model-checking problem is NP-hard as well. The edge-disjoint model-checking problem for specifications expressed with formulas of the type $\neg E^{>k}\mathcal{G}p$ hence turns out to be CONP-hard. Thus the theorem holds. $\square$

From the previous theorem, we have that the edge-disjoint graded-CTL model-checking problem is not in NP (and not in CONP as well) unless NP $=$ CONP. However, we show an upper bound for this problem.

Theorem 5.2. There is an algorithm to solve the edge-disjoint graded-CTL model-checking problem in space $\mathcal{O}(|R| \cdot |S| + |\varphi|)$.

Proof:
Consider the following simple algorithm to model-check formulas $E^{>k}\theta$ with either $\theta = \mathcal{G}\psi_1$ or $\theta = \psi_1\mathcal{U}\psi_2$: if $k \geq |R|$ then simply answer *false*, otherwise the Kripke structure is visited to look for paths satisfying θ and, each time a path is found, a new visit is recursively started, looking for other paths in the remaining graph, until $k+1$ edge-disjoint paths are found. This algorithm can be easily implemented by using polynomial space, as the overall size of the $k+1$ paths is bounded by $|R|$.

Note that this algorithm works in time exponential in k if $k < |R|$ and in time $\mathcal{O}(1)$ otherwise, therefore its running time is exponential in the size of $\mathcal{K}$ and still independent from k. $\square$

5.2. A fragment

One question that naturally arises from Theorem 5.1 is whether it is possible to define interesting fragments of graded-CTL for which the edge-disjoint graded-CTL model-checking problem can be solved in polynomial-time. In particular, the proof of Theorem 5.1 suggests that formulas of the type $E^{>k}\mathcal{G}\varphi$, with $k > 0$, are "hard" to verify. In this section we introduce a fragment, called graded-RCTL, of graded-CTL not containing formulas of the type $E^{>k}\mathcal{G}\varphi$, with $k > 0$ and show that there is a polynomial-time algorithm for the model-checking problem. Note that the fragment still is an extension of CTL and that many significant non CTL properties can be expressed within it. For example, consider the property stating that *do not exist more than k bad behaviors such that a device does not start unless a key is pressed*: such a property can be expressed in graded-RCTL with the formula $\neg E^{>k}(\neg key\, \mathcal{U}(start \wedge \neg key))$.

Theorem 5.3. Let $\mathcal{K} = \langle S, s_{in}, R, L\rangle$ be a Kripke structure and φ be a graded-RCTL formula. The edge-disjoint graded-RCTL model-checking problem, for $\mathcal{K}$ and φ, can be solved in time $\mathcal{O}(|R|^2 \cdot |S| \cdot |\varphi|)$.

Proof:
Since in graded-RCTL there are no $E^{>k}\mathcal{G}\psi_1$ formulas, we have only to show how to check sub-formulas $\psi = E^{>k}\psi_1\mathcal{U}\psi_2$ with $k > 0$. To this aim we will use ideas from flow networks of the graph theory.

Let us recall that a *flow network* is a directed graph with a *source* node, a *destination* node, and with edges having a non-negative capacity representing the amount of data that can be moved through the edge. A *maximum flow* from the source to the destination is the maximum amount of data that a network can move from the source to the destination.

The algorithm is identical to Algorithm 1 for graded-CTL, with the lines $17-29$ rewritten as follows (where $d \notin S$ is a new node used as destination node, $inDegree(t)$ returns the in-degree of a state t, c is the capacity function on the edges and $MaxFlow(S, R, c, s, d)$ returns the maximum flow from s to d on the graph (S, R)):

```
17 case $\psi = E^{>k}\psi_1\mathcal{U}\psi_2$ with $k > 0$:
18     $S' \leftarrow \{s \in S \mid s.E^{>0}\psi_1\mathcal{U}\psi_2 = \text{TRUE}\}$;
19     $R' \leftarrow (R \cap (S' \times S')) \setminus \{(s, s') \mid s.\psi_1 = \text{FALSE}\}$;
20     $S' \leftarrow S' \cup \{d\}$;   $R' \leftarrow R' \cup \{(s, d) \mid s.\psi_2 = \text{TRUE}\}$;
21     forall $e \in R'$ do $c(e) = inDegree(s)$ if $e = (s, d)$ and $c(e) = 1$ otherwise;
22     forall $s \in S' \setminus \{d\}$ s.t. $MaxFlow(S', R', c, s, d) > k$ do $s.\psi \leftarrow \text{TRUE}$;
23 end
```

Our algorithm considers in lines $18, 19$, the graph G_ψ, subgraph of $\mathcal{K}$, of the states where the formula $E^{>0}\psi_1\mathcal{U}\psi_2$ holds (without the edges outgoing from states where ψ_1 doesn't hold). Now one should verify, for each state $s \in S$, the existence of $k + 1$ edge-disjoint paths in G_ψ each starting from s and ending in a state where ψ_2 holds. To do this, the algorithm creates a network (S', R') by adding to G_ψ a new destination node d and, for each state s where ψ_2 holds, an edge (s, d) with capacity equal to the in-degree of s (the remaining edges have capacity 1, see Figure 2). Finally, for each $s \in S' \setminus \{d\}$ verifies whether the maximum flow from s to d in this network is greater than k.

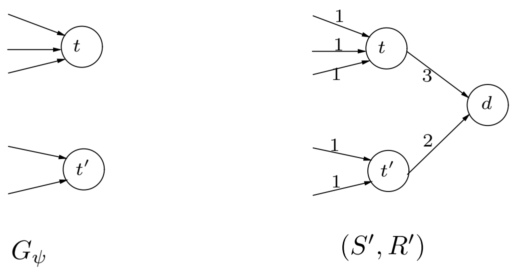

Figure 2. Transformation of G_ψ into (S', R'). In the states t and t' the formula ψ_2 is satisfied.

To prove the correctness of the algorithm we have to show that, for all $s \in S' \setminus \{d\}$, the maximum flow from s to d in (S', R') is equal to the maximum number of edge-disjoint paths starting from s and ending in a state where ψ_2 holds in G_ψ. Let us consider the network (S'', R'') derived from (S', R') by substituting each edge (t, d) with capacity $c > 1$ with c edge-disjoint new paths $\langle t, t_i, d\rangle$, $1 \leq i \leq c$, with $t_i \notin S'$ and $c(t, t_i) = c(t_i, d) = 1$ (see Figure 3).

It is easy to see that the maximum flow from s to d in (S', R') is equal to the maximum flow from s to d in (S'', R''). It is known that the maximum flow from s to d in a network with all unitary capacity is equal to the maximum number of edge-disjoint paths from s to d, see e.g. [10]. Let $f(s, d)$ be the maximum flow from s to d in (S'', R''). Since, from the construction, each path from s to d in (S'', R'') is

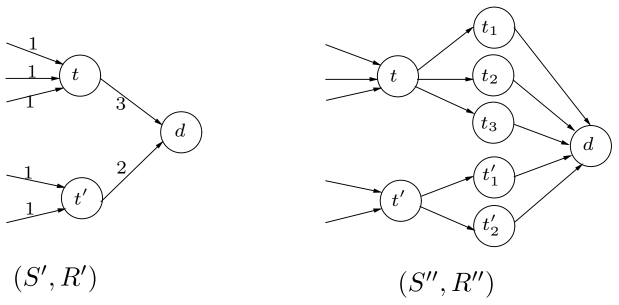

Figure 3. Transformation of (S', R') into (S'', R'').

of the form $\langle s, \ldots, t, t_i, d\rangle$ where $\langle s, \ldots, t\rangle$ is a path in (S', R') starting from s and such that $(\mathcal{K}, t) \models \psi_2$, then $f(s, d)$ is also the maximum number of pairwise edge-disjoint paths in (S', R') starting from s and ending in a state where ψ_2 holds.

The running-time of the algorithm on a sub-formula $E^{>k}\psi_1\mathcal{U}\psi_2$ depends on the time required to calculate the maximum flow. Note that the total capacity of the edges entering in d is at most $|R|$, therefore the maximum flow from any state to d is upper bounded by $|R|$. Then, by using for example the classical Ford-Fulkerson algorithm (see e.g. [10]), that works in time $\mathcal{O}(f \cdot |R|)$ (where f is the value of the maximum flow), the overall time complexity of the algorithm is $\mathcal{O}(|R|^2 \cdot |S| \cdot |\varphi|)$. □

5.3. A parameterized version of the problem

Let $\mathcal{K} = \langle S, s_{in}, R, L\rangle$ and $\hat{R}$ be a subset of R. We say that two paths π_1 and π_2 in $\mathcal{K}$ are *$\hat{R}$-edge-disjoint* if there are no edges in $\hat{R}$ belonging to both π_1 and π_2. We introduce the relation $\models_{ed}^{\hat{R}}$ which differs from the finer relation $\models_{ed}$ only for the formulas of the type $E^{>k}\mathcal{G}\psi_1$ and $E^{>k}\psi_1\mathcal{U}\psi_2$. In particular, we require the existence of $k+1$ pairwise $\hat{R}$-edge-disjoint paths satisfying $\mathcal{G}\psi_1$ or $\psi_1\mathcal{U}\psi_2$. Then, the **subset-edge-disjoint graded-CTL model-checking** requires to verify whether $(\mathcal{K}, s_{in}) \models_{ed}^{\hat{R}} \varphi$, for a Kripke structure $\mathcal{K}$, a set $\hat{R} \subseteq R$, and a graded-CTL formula φ.

The lower bound to this problem obviously matches the lower bound of the edge-disjoint graded-CTL model-checking problem. However, in the following theorem we prove that the problem is *fixed parameter* tractable, in fact we solve it in exponential time only in the size of $\hat{R}$, obtaining thus a good algorithm for practical cases.

Theorem 5.4. Let $\mathcal{K} = \langle S, s_{in}, R, L\rangle$ be a Kripke structure, $\hat{R} \subseteq R$ and φ be a graded-CTL formula. The subset-edge-disjoint graded-CTL model-checking problem can be solved with an algorithm whose running time is $\mathcal{O}((4^{|\hat{R}|} \cdot |R| + 2^{|\hat{R}|^2}) \cdot |S| \cdot |\varphi|)$ and with space complexity $\mathcal{O}(4^{|\hat{R}|} \cdot |\hat{R}| + |R| + |\varphi|)$.

Proof:
Since the difference between graded-CTL and subset-edge-disjoint graded-CTL model-checking is only in the satisfiability of formulas $E^{>k}\theta$, with the path-formula θ being either $\theta = \mathcal{G}\psi_1$ or $\theta = \psi_1\mathcal{U}\psi_2$ and $k > 0$, the algorithm to solve our problem is identical to Algorithm 1, but for the extra input value $\hat{R}$, and for the lines 17-29 replaced by these:

```
17 case $\psi = E^{>k}\theta$ with $\theta = \mathcal{G}\psi_1$ or $\theta = \psi_1\mathcal{U}\psi_2$ and $k > 0$:
18     forall $s \in S$ do
19         $I \leftarrow \{i \in \{k+1-|\hat{R}|, \dots, k+1\} \mid i \geq 0$ and $\exists\, i$ pairwise distinct paths from $s$
           satisfying $\theta$ without using edges in $\hat{R}\}$;
20         if $I \neq \emptyset$ then
21             $\hat{k} \leftarrow k+1-\max\{i \mid i \in I\}$;
22             if $\hat{k} = 0$ then $s.\psi \leftarrow$ TRUE; continue;
23             $\mathcal{V} \leftarrow \{T \mid T$ is the set of edges of $\hat{R}$ occurring in an evidence of $\theta\}$;
24             $E \leftarrow \{(T, T') \in \mathcal{V}^2 \mid T \cap T' = \emptyset\}$;
25             if $\exists$ a clique with size $\hat{k}$ in $(\mathcal{V}, E)$ then $s.\psi \leftarrow$ TRUE;
26         end
27     end
28 end
```

This part of the algorithm works as follows. Consider a state $s \in S$. As the number of $\hat{R}$-edge-disjoint evidences of θ which use at least one edge belonging to $\hat{R}$ is bounded by $|\hat{R}|$ itself, the number of the remaining evidences of θ (not using edges of $\hat{R}$) must be greater than $k+1-|\hat{R}|$ (otherwise $(\mathcal{K}, s) \not\models^{\hat{R}}_{ed} E^{>k}\theta$). Thus the algorithm first determines a number $\hat{k}$, lines 19-21, with the property that: $(\mathcal{K}, s) \models^{\hat{R}}_{ed} E^{>k}\theta$ if and only if there are $\hat{k}$ $\hat{R}$-edge-disjoint evidences of θ which use at least one edge belonging to $\hat{R}$. Then the graph $(\mathcal{V}, E)$, described in lines 23 and 24, is computed, such that a node in $\mathcal{V}$ is a set of edges of $\hat{R}$ which occur in an evidence of θ in $\mathcal{K}$ and an edge in E connects two disjoint such sets. Thus, $(\mathcal{K}, s) \models^{\hat{R}}_{ed} E^{>k}\theta$ iff in the graph $(\mathcal{V}, E)$ there is a clique of size $\hat{k}$.

Let us evaluate the running time and the space required by the algorithm. Since the set I described in line 19 is such that $|I| \leq |\hat{R}|$, the lines 19-21 can be easily computed in time $\mathcal{O}(|R| \cdot |\hat{R}|)$ by using a simple variation of Algorithm 1. Moreover, for a given subset T of $\hat{R}$, the existence of an evidence of θ which uses *all* the edges in T and possibly edges of $R \setminus \hat{R}$, can be verified in time $\mathcal{O}(|R|)$, while the set of edges outgoing from T can be computed in time $\mathcal{O}(2^{|\hat{R}|} \cdot |\hat{R}|)$; therefore the graph $(\mathcal{V}, E)$ can be computed in time $\mathcal{O}(4^{|\hat{R}|} \cdot |R|)$. Finally, the existence of a clique of size $\hat{k} \leq |\hat{R}|$ can be verified in time $\mathcal{O}(2^{|\hat{R}|^2})$.

The algorithm needs, to model-check a formula $E^{>k}\theta$ in a state $s \in S$, space $\mathcal{O}(4^{|\hat{R}|} \cdot |\hat{R}|)$ to store the graph $(\mathcal{V}, E)$ and space $\mathcal{O}(|R|)$ to calculate the path needed to verify whether a non-empty subset T of $\hat{R}$ is in $\mathcal{V}$. Moreover, the algorithm globally needs only $3 \cdot |S|$ truth values for the sub-formulas (two for the operands and one for the operator in each state). Therefore the space required by the algorithm is $\mathcal{O}(4^{|\hat{R}|} \cdot |\hat{R}| + |R| + |\varphi|)$. □

6. Conclusions

In this paper we have introduced graded-CTL as a more expressive extension of classical CTL. The results presented are in the model-checking setting with specifications in this new logic. We have investigated the complexities involved in various scenarios, all from a theoretical perspective. In [13] we describe a model-checker tool obtained by augmenting NuSmv with these grading modalities: we report test results showing that the classical performances are retained. We believe that this framework could turn out to be useful also in the verification of fault tolerant physical properties of networks. Let us mention also a drawback of our setting: as said in the introduction the generation of more than one counterexample is highly desirable, however the analyze stage (of the realization process of a system) is critical also for the size of the counterexamples and the poor human-readability of it.

One of the main concerns of the model checking setting, is the so-called *state explosion problem* which leads to manage systems with a huge number of states. An interesting approach to this, is that of *Hierarchical State Machines* [1, 20], where each state of the Kripke structure can also be a super-node representing in turn a smaller structure. One possible future direction to work on, is to study the graded-CTL in this setting.

References

[1] Alur, R., Yannakakis, M.: Model checking of hierarchical state machines, *ACM Trans. Program. Lang. Syst.*, **23**(3), 2001, 273–303.

[2] Caprara, A., Panconesi, A., Rizzi, R.: Packing cycles in undirected graphs, *Journal of Algorithms*, **48**(1), 2003, 239–256.

[3] Chechik, M., Gurfinkel, A.: A framework for counterexample generation and exploration, *Int. J. Softw. Tools Technol. Transf.*, **9**(5), 2007, 429–445, ISSN 1433-2779.

[4] Cimatti, A., Clarke, E. M., Giunchiglia, F., Roveri, M.: NUSMV: A New Symbolic Model Verifier, *Computer Aided Verification*, 1999, 495–499.

[5] Clarke, E., Emerson, E.: Usig Branching Time Temporal Logic to Synthesize Synchronization Skeletons, *Science of Computer Programming*, **2**, 1982, 241–266.

[6] Clarke, E., Grumberg, O., Peled, D.: *Model Checking*, The MIT Press, 1999.

[7] Clarke, E., Veith, H.: Counterexamples Revisited: Principles, Algorithms, Applications, *Verification: Theory and Practice*, 2003, 208–224.

[8] Clarke, E. M., Emerson, E. A., Sistla, A. P.: Automatic verification of finite-state concurrent systems using temporal logic specifications, *ACM Transactions on Programming Languages and Systems*, **8**, 1986, 244–263.

[9] Copty, F., Irron, A., Weissberg, O., Kropp, N., Kamhi, G.: Efficient Debugging in a Formal Verification Environment, *CHARME '01*, Springer-Verlag, London, UK, 2001, ISBN 3-540-42541-1, 275–292.

[10] Cormen, T., Leiserson, C., Rivest, R., Stein, C.: *Introduction to Algorithms - Second Edition*, The MIT Press, 2001.

[11] Dong, Y., Ramakrishnan, C., Smolka, S.: Model Checking and Evidence Exploration, *ECBS*, 2003, 214–223.

[12] Ferrante, A., Napoli, M., Parente, M.: CTLModel-Checking with Graded Quantifiers, *ATVA*, 2008, 18–32.

[13] Ferrante, A., Napoli, M., Parente, M.: Symbolic Graded-CTL Model Checking, *Submitted for publication*, 2009, http://gradedctl.dia.unisa.it/publications/fnp09.pdf.

[14] Fine, K.: In So Many Possible Worlds, *Notre Dame Journal of Formal Logic*, **13**(4), 1972, 516–520.

[15] Ganzinger, H., Meyer, C., Veanes, M.: The Two–Variable Guarded Fragment with Transitive Relations, *LICS*, 1999, 24–34.

[16] Grädel, E., Otto, M., Rosen, E.: Two–Variable Logic with Counting is Decidable, *LICS*, 1997, 306–317.

[17] Hollunder, B., Baader, F.: Qualifying Number Restrictions in Concept Languages, *KR*, 1991, 335–346.

[18] Holzmann, G.: The Model Checker SPIN, *IEEE Transactions on Software Engineering*, **23**(5), 1997, 279–295, ISSN 0098-5589.

[19] Kupferman, O., Sattler, U., Vardi, M.: The Complexity of the Graded μ–Calculus, *CADE-18: Proceedings of the 18th International Conference on Automated Deduction*, Springer-Verlag, London, UK, 2002, ISBN 3-540-43931-5, 423–437.

[20] La Torre, S., Napoli, M., Parente, M., Parlato, G.: Verification of scope-dependent hierarchical state machines, *Inf. Comput.*, **206**(9-10), 2008, 1161–1177, ISSN 0890-5401.

[21] McMillan, K.: The Cadence SMV Model Checker, http://www.kenmcmil.com/psdoc.html.

[22] Pacholski, L., Szwast, W., Tendera, L.: Complexity Results for First–Order Two–Variable Logic with Counting, *SIAM Journal of Computing*, **29**(4), 2000, 1083–1117.

[23] Queille, J., Sifakis, J.: Specification and verification of concurrent systems in CESAR, *Proc. of the 5th Colloquium on Intern. Symposium on Programming*, Springer-Verlag, London, UK, 1982, ISBN 3-540-11494-7, 337–351.

[24] Tobies, S.: PSPACE Reasoning for Graded Modal Logics, *Journal Log. Comput.*, **11**(1), 2001, 85–106.