CTL Model-Checking with Graded Quantifiers[*]

Alessandro Ferrante, Margherita Napoli, and Mimmo Parente

Dipartimento di Informatica ed Applicazioni "R.M. Capocelli", University of Salerno
Via Ponte don Melillo - 84084 - Fisciano (SA) - Italy
{ferrante,napoli,parente}@dia.unisa.it

Abstract. The use of the universal and existential quantifiers with the capability to express the concept of *at least k* or *all but k*, for a non-negative integer k, has been thoroughly studied in various kinds of logics. In classical logic there are *counting quantifiers*, in modal logics *graded modalities*, in description logics *number restrictions*.

Recently, the 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. They have shown that this problem is ExpTime-complete.

In this paper we consider another extension of modal logic, the Computational Tree Logic CTL, augmented with graded modalities generalizing standard quantifiers and investigate the complexity issues, with respect to the model-checking problem. We consider a system model represented by a pointed 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 (where $|\varphi|$ 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 (or evidences) 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 this graded-CTL logic, and have proved that the model-checking problem is solvable in polynomial time.

1 Introduction

Model-checking is the process, which is now becoming widely accepted, to check whether a given model satisfies a given logical formula [CGP99, QS82], and it can be applied to all kinds of logics. In this paper we consider model-checking of formulas expressed in a logic which extends the classical Computational Tree Logic, CTL, with graded modalities. Classical CTL can be used for reasoning

[*] Work partially supported by M.I.U.R. grant ex-60%.

about the temporal behavior of systems considering either *all the possible futures* or *at least one possible future*. Here we use *graded* extensions on the existential and universal quantifiers.

In the literature the capability to express *at least k* and *all but k*, given a non-negative integer 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. [GOR97, GMV99, PST00], in modal logics they are called *graded* modalities, see e.g. [Fin72, Tob01], and in description logics one speaks about *number restriction* of properties describing systems, see e.g. [HB91]. Recently the complexity issues related to the decidability of the μ-calculus when the universal and existential quantifiers are augmented with graded modalities, have been investigated in [KSV02]. 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.

In this paper we introduce the *graded*-CTL, obtained by augmenting CTL with graded modalities that generalize standard path quantifiers and this logic, here too, strictly extends classical CTL. With graded-CTL formulas we can reason about more than any constant number of futures. For example, the formula $E^{>k}\mathcal{F}\neg(wait \Rightarrow A\mathcal{F}criticSection)$ expresses the fact that in several cases it is possible that a waiting process never obtains the requested resource. Note that this logic allows also to grade nested path 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*" ($A\mathcal{G}E^{>1}\mathcal{F}safe$). Clearly formulas of this type cannot be expressed in CTL and not even in classical μ-calculus. The focus in the paper is on the complexities involved in the process of model-checking system models against specifications given in this logic. In this framework 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. [CG07, CIW$^+$01, DRS03]. Usually up-to-date model-checkers, as NuSMV and SPIN [CCGR99, Hol97], return only one counterexample of φ or, by using the so-called *onion ring* technique, may determine all the counterexamples to a given non-graded formula. Here we aim at getting more *significative* counterexamples, in the sense that by nesting the graded quantifiers we can concentrate ourselves on zones of the model for which we are more interested in. In other words with the actual model checkers we can obtain counterexamples to a formula with only the first quantifier which, in a sense, is graded, while in our scenario we can have also the inner quantifiers which are graded. 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 [CV03] where the role and the structure of counterexamples is nicely 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 pointed Kripke structure $\mathcal{K}$, our first result is an algorithm to solve the model-checking problem in time $O(|\mathcal{K}| \cdot |\varphi|)$, 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 and 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. However the multiple counterexamples returned by this algorithm may overlap, while it can be desirable in the analysis phase to detect independent traces where the specification fails. To deal with this case, we have introduced a semantic for temporal operators to require the 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 of the type $E^{>k}\mathcal{G}\varphi$ (there exist at least $k+1$ 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. Clearly, unless NP = coNP, the problem, in the full logic, does not belong to NP. We have then given an algorithm for it, 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 give some preliminary definitions and introduce the model-checking problem for graded-CTL; in Section 3 we prove that the graded-CTL model-checking is solvable in polynomial time; in Section 4 we study the edge-disjoint graded-CTL model-checking. 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 in practical cases by relaxing the edge-disjointness requirement; finally in Section 5 we give some conclusions and open problems.

2 Graded-CTL Logic

In this section we introduce the graded-CTL logic which extends the classical CTL logic with graded quantifiers. 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 extension generalizes CTL to reasoning about more than a given number of possible future behaviors.

Let us start by giving the syntax of the logic. The graded-CTL operators consist of the temporal operators $\mathcal{U}$ ("until"), $\mathcal{G}$ ("globally") and $\mathcal{X}$ ("next"), the boolean connectives $\wedge$ and $\neg$, and the graded path quantifier $E^{>k}$ ("for at least k+1 futures").

Given a set of atomic propositions AP, the syntax of the graded-CTL formulas is:

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

where $p \in AP$ and k is a non-negative integer.

We define the semantics of graded-CTL with respect to *Kripke Structures*. 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 state s_i, $0 \leq i < |\pi|$. Two paths π_1 and π_2 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.

Let $\mathcal{K} = \langle S, s_{in}, R, L\rangle$ be a Kripke structure and $s \in S$ be a state of $\mathcal{K}$. The concept of satisfiability for graded-CTL formulas is established by the relation $\models$, defined as follows:

- $(\mathcal{K}, s) \models p$, $p \in AP$, iff $p \in L(s)$;
- $(\mathcal{K}, s) \models \varphi_1 \wedge \varphi_2$ iff $(\mathcal{K}, s) \models \varphi_1$ and $(\mathcal{K}, s) \models \varphi_2$;
- $(\mathcal{K}, s) \models \neg\varphi$ iff $\neg((\mathcal{K}, s) \models \varphi)$ (shortly written, $(\mathcal{K}, s) \not\models \varphi$);
- $(\mathcal{K}, s) \models E^{>k}\mathcal{X}\varphi$ iff there exist $k+1$ different states $s_0, \ldots, s_k$ such that
  1. $(s, s_i) \in R$ and
  2. $(\mathcal{K}, s_i) \models \varphi$ for all $0 \leq i \leq k$;
- $(\mathcal{K}, s) \models E^{>k}\mathcal{G}\varphi$ iff there exist $k+1$ pairwise distinct infinite paths $\pi_0, \ldots, \pi_k$ such that for every $0 \leq j \leq k$,
  1. $\pi_j[0] = s$ and
  2. for all $h \geq 0$, $(\mathcal{K}, \pi_j[h]) \models \varphi$.
- $(\mathcal{K}, s) \models E^{>k}\varphi_1\mathcal{U}\varphi_2$ iff there exist $k+1$ pairwise distinct finite paths $\pi_0, \ldots, \pi_k$ of length $i_0, \ldots, i_k$, respectively, such that for all $0 \leq j \leq k$
  1. $\pi_j[0] = s$,
  2. $(\mathcal{K}, \pi_j[i_j - 1]) \models \varphi_2$, and
  3. for every $0 \leq h < i_j - 1$, $(\mathcal{K}, \pi_j[h]) \models \varphi_1$;

The graded-CTL formulas (as in the standard non-graded CTL) are also called *state-formulas* and a state s in $\mathcal{K}$ satisfies a state-formula φ if $(\mathcal{K}, s) \models \varphi$. On the other side, $\mathcal{X}\varphi$, $\mathcal{G}\varphi$ and $\varphi_1\mathcal{U}\varphi_2$ are called as usual *path-formulas*. In particular a path satisfying a path-formula θ is called an *evidence* of θ (note that the evidences for $\mathcal{X}$ and $\mathcal{U}$ are finite paths). Then, for the fulfillment of a formula $E^{>k}\theta$ in a state s, it is required the existence of $k+1$ distinct evidences of θ, starting from s.

As usual, in our logic the temporal operator $\mathcal{F}$ ("eventually") can be defined in terms of the operators given above: $E^{>k}\mathcal{F}\varphi \Leftrightarrow E^{>k}$ True $\mathcal{U}$ φ. Moreover, it is easy to observe that CTL is a proper fragment of graded-CTL since the simple formula $E^{>1}\mathcal{X}p$ cannot be expressed in CTL, whereas any CTL formula is also a graded-CTL formula since the quantifier E is equivalent to $E^{>0}$ and for the universal quantifier A the standard relations hold, recalled here:

– $A\mathcal{X}\varphi \Longleftrightarrow \neg E\mathcal{X}\neg\varphi$;
– $A\mathcal{G}\varphi \Longleftrightarrow \neg E\mathcal{F}\neg\varphi$;
– $A\varphi_1\mathcal{U}\varphi_2 \Longleftrightarrow \neg E(\neg\varphi_2\mathcal{U}(\neg\varphi_1 \wedge \neg\varphi_2)) \wedge \neg E\mathcal{G}\neg\varphi_2$.

Finally, we also consider the graded extension of the quantifier A, $A^{\leq k}$, with the meaning that *all the paths starting from a node s, except for at most k, satisfy a given path-formula*. The quantifier $A^{\leq k}$ is the dual of $E^{>k}$ and can obviously be re-written in terms of $\neg E^{>k}$. We now formally define the model-checking problem.

Given a Kripke structure $\mathcal{K} = \langle S, s_{in}, R, L\rangle$, and a graded-CTL formula φ, the **graded-CTL model-checking** is the problem to verify whether $(\mathcal{K}, s_{in}) \models \varphi$.

In the next sections we study the complexity of the 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 CTL formula, where the size $|\varphi|$ of a CTL formula φ is the number of the temporal and the boolean operators occurring in it.

3 Graded-CTL Model-Checking

In this section we show that the graded-CTL model-checking problem can be solved in polynomial time and independently from the values occurring in the graded modalities, involved in the formulas. Then we discuss possible applications of our result to the generation of counterexamples.

Let us recall that an algorithm to solve the model-checking problem for a given Kripke structure $\mathcal{K}$ and a given formula φ computes the subset $\{s \in S$ s.t. $(\mathcal{K}, s) \models \varphi\}$.

Theorem 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 problem we give an algorithm that 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 $Sub(\varphi)$ which returns all the sub-formulas of 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

[CGP99], and, if a sub-formula is of type $E^{>k}\mathcal{X}\psi_1$, then the algorithm checks, for each state s whether $|\{t \in S \mid (s,t) \in R \text{ and } (\mathcal{K}, t) \models \psi_1\}| > k$, (lines 14−16).

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

1  Let $s.\psi = $ FALSE for all $s \in S$ and $\psi \in Sub(\varphi)$;
2  forall $\psi \in Sub(\varphi)$ do
3      case $\psi = p \in AP$: forall $s \in S$ s.t. $p \in L(s)$ do $s.\psi \leftarrow$ TRUE;
4      case $\psi = \neg\psi_1$: forall $s \in S$ do $s.\psi \leftarrow \neg s.\psi_1$;
5      case $\psi = \psi_1 \wedge \psi_2$: forall $s \in S$ do $s.\psi \leftarrow (s.\psi_1 \wedge s.\psi_2)$;
6      case $\psi = E^{>0}\mathcal{G}\psi_1$:
7          $S' \leftarrow \{s \in S \mid s.\psi_1 = $ TRUE$\}$; $R' \leftarrow R \cap (S' \times S')$;
8          forall $s \in S'$ s.t. $\exists$ a cycle reachable from $s$ in $(S', R')$ do $s.\psi \leftarrow$ TRUE;
9      end
10     case $\psi = E^{>0}\psi_1\mathcal{U}\psi_2$:
11         $S' \leftarrow \{s \in S \mid s.\psi_1 = $ TRUE or $s.\psi_2 = $ TRUE$\}$; $R' \leftarrow R \cap (S' \times S')$;
12         forall $s \in S'$ s.t. $\exists$ $t$ with $t.\psi_2 = $ TRUE reachable from $s$ in $(S', R')$ do
           $s.\psi \leftarrow$ TRUE;
13     end
14     case $\psi = E^{>k}\mathcal{X}\psi_1$ with $k \geq 0$:
15         forall $s \in S$ s.t. $|\{(s,t) \in R \mid t.\psi_1 = $ TRUE$\}| > k$ do $s.\psi \leftarrow$ TRUE;
16     end
17     case $\psi = E^{>k}\mathcal{G}\psi_1$ with $k > 0$:
18         $S' \leftarrow \{s \in S \mid s.E^{>0}\mathcal{G}\psi_1 = $ TRUE$\}$; $R' \leftarrow R \cap (S' \times S')$;
19         forall $s \in S'$ s.t. $\exists$ a non-sink-cycle reachable from $s$ in $(S', R')$ do
           $s.\psi \leftarrow$ TRUE;
20         forall $s \in S'$ s.t. $\exists$ $k+1$ distinct finite paths from $s$ to sink-cycles in
           $(S', R')$ do $s.\psi \leftarrow$ TRUE;
21     end
22     case $\psi = E^{>k}\psi_1\mathcal{U}\psi_2$ with $k > 0$:
23         $S' \leftarrow \{s \in S \mid s.E^{>0}\psi_1\mathcal{U}\psi_2 = $ TRUE$\}$;
24         $R' \leftarrow (R \cap (S' \times S')) \setminus \{(s,t) \in R \mid s.\psi_1 = $ FALSE$\}$;
25         forall $s \in S'$ s.t. $\exists$ a non-sink-cycle reachable from $s$ in $(S', R')$ do
           $s.\psi \leftarrow$ TRUE;
26         forall $s \in S'$ s.t. $\exists$ $k+1$ distinct finite paths from $s$ to states where $\psi_2$
           holds in $(S', R')$ do $s.\psi \leftarrow$ TRUE;
27     end
28 end
```

Consider now a sub-formula $\psi = E^{>k}\mathcal{G}\psi_1$ with $k > 0$ (line 17). Let a *sink-cycle* be a cycle not containing *exit-nodes*, that is nodes with out-degree at least 2. The algorithm is based on the following claim, that we will prove later:

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

The algorithm looks for the states in G_ψ from which it is possible to reach a non-sink-cycle (line 19) and then looks for the states from which $k+1$ distinct finite paths start, each ending in sink-cycles (line 20).

Let us now consider a sub-formula $\psi = E^{>k}\psi_1\mathcal{U}\psi_2$ (line 22). In this case, the algorithm is based on the following claim:

Claim 2: *Let $G_\psi = (S_\psi, R_\psi)$ be the graph induced by considering the states 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 $s \in S$, $(\mathcal{K}, s) \models \psi$ iff $s \in G_\psi$ and either there is a* non-sink-cycle *reachable from s, or there are $k+1$ distinct finite paths from s to states where ψ_2 holds.*

Similarly to what has been done for the case of the operator $\mathcal{G}$, now the algorithm looks for the states in G_ψ from which it is possible to reach a non-sink-cycle (line 25), and then looks for the states from which $k+1$ distinct finite paths start, each ending in states where ψ_2 holds, (line 26). The proof of the correctness of the algorithm can be easily done by induction on the length of the formulas.

To complete the proof let us first prove Claim 1.

(if): Let $s \in S_\psi$ and $C = \langle v_0, \ldots, v_{h-1}\rangle$ be a cycle in G_ψ reachable from s via a finite path $\langle s, u_0, \ldots, u_i, v_0\rangle$ and containing at least one exit-node, say v_j, $0 \leq j \leq h-1$ connected to a node $w_0 \in S_\psi$ such that $w_0 \neq v_{(j+1) \bmod h}$ and $(v_j, w_0) \in R_\psi$. 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 $\mathcal{G}\psi_1$ and there are $k+1$ pairwise distinct infinite paths π_l, $0 \leq l \leq k$, 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, since a finite path from s to a sink-cycle in G_ψ constitutes an infinite path satisfying $\mathcal{G}\psi_1$, if there are $k+1$ distinct finite paths connecting s to sink-cycles in G_ψ then $(\mathcal{K}, s) \models \psi$.

(only if): If $(\mathcal{K}, s) \models E^{>k}\mathcal{G}\psi_1$ then obviously $(\mathcal{K}, s) \models E^{>0}\mathcal{G}\psi_1$, therefore $s \in S_\psi$. Let us consider $k+1$ distinct infinite paths $\pi_0, \ldots, \pi_k$ starting from s and satisfying $\mathcal{G}\psi_1$. Since an infinite path on a finite Kripke structure either contains a non-sink-cycle, or ends in a sink-cycle, the claim follows from the fact that each state in $\pi_0, \ldots, \pi_k$ belongs to S_ψ.

Finally let us now prove Claim 2.

(if): Let $s \in S_\psi$ and $C = \langle v_0, \ldots, v_{h-1}\rangle$ be a non-sink-cycle, reachable from s via a finite path $\langle s, u_0, \ldots, u_i, v_0\rangle$. Let v_j, for $0 \leq j \leq h-1$, be an exit-node of C connected to a node $w_0 \in S_\psi$ such that $w_0 \neq v_{(j+1) \bmod h}$ and $(v_j, w_0) \in R_\psi$. 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, $0 \leq l \leq k$, 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 R_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 $k+1$ distinct finite paths 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 $(\mathcal{K}, s) \models \psi$, as well.

(only if): If $(\mathcal{K}, s) \models E^{>k} \psi_1 \mathcal{U} \psi_2$ then obviously $(\mathcal{K}, s) \models E^{>0} \psi_1 \mathcal{U} \psi_2$, therefore $s \in S_\psi$. On the other side, from the semantics of $E^{>k} \psi_1 \mathcal{U} \psi_2$, there are $k+1$ distinct finite paths starting from s and ending in states satisfying ψ_2 and these are also paths in G_ψ, and this completes the proof of the claim.

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 vertices from which it is possible to reach a non-sink-cycle can be globally calculated in time $\mathcal{O}(|R|)$ by using 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 vertices from which $k+1$ paths leading to sink-cycles exist, can be globally calculated in time $\mathcal{O}(|R|)$ by using a standard DFS algorithm. 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 $\mathcal{O}(|R| \cdot |\varphi|)$. □

An example of Claim 2 is the model in Figure 1 which satisfies the formula $E^{>k} \mathcal{F}(wait1 \wedge E\mathcal{G}\neg critic1)$, for all $k \geq 0$, as contains reachable non-sink-cycles (one is depicted with bold-faced edges).

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 1 can be modified to return the required counterexamples.

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

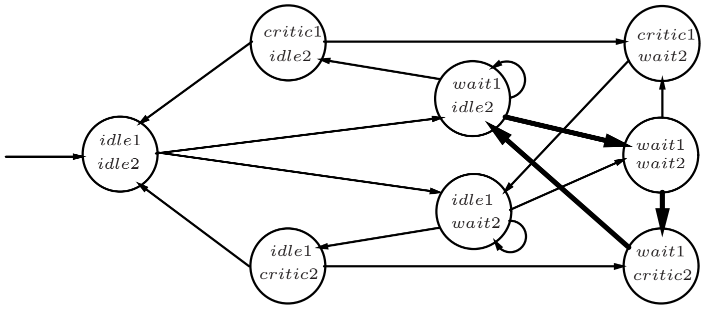

Fig. 1. A mutual exclusion system

When NuSMV (or also SMV Cadence [CCGR99, McM]) 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. We have hence studied a symbolic algorithm for our setting whose complexity turns out to be $O(2^{|AP|} \cdot k \cdot |\varphi|)$, 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. Thus we have the following theorem (whose full proof is in the extended version of the paper [FNP08]).

Theorem 2. *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 $\mathcal{O}(2^{|AP|} \cdot k \cdot |\varphi|)$, where k is the maximum value occurring in φ.*

4 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. [CLRS01]. 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.

4.1 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 [CPR03]).

Theorem 3. *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 $k \geq 0$.

Given a graph $G = (\mathcal{V}, \mathcal{E})$ 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 $R = \mathcal{E} \cup \{(\hat{s}, s) \text{ s.t. } s \in \mathcal{V}\}$ 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. □

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 now show an upper bound for this problem. In fact let us consider the following simple algorithm to model-check formulas $E^{>k}\theta$ with either $\theta = \mathcal{G}\psi_1$ or $\theta = \psi_1\mathcal{U}\psi_2$: the Kripke structure is visited to find 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 $|R|$. Therefore we obtain the following theorem.

Theorem 4. *There is an algorithm to solve the edge-disjoint graded-*CTL *model-checking problem in space* $\mathcal{O}(|R| \cdot |S| + |\varphi|)$.

4.2 A Fragment

One question that naturally arises from Theorem 3 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 3 suggests that only 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 for it there is a polynomial-time algorithm for the model-checking problem. Note that the fragment is an extension of CTL and that still many significant 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. *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 the graded-RCTL there are no $E^{>k}\mathcal{G}\psi_1$ formulas, we have only to show how to check sub-formulas $E^{>k}\psi_1\mathcal{U}\psi_2$ 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 in the time unit.

The algorithm is identical to Algorithm 1 for graded-CTL, with the lines $17-27$ rewritten as follows (where $d \notin S$ is the destination node, $inDegree(s)$ returns the in-degree of a state s and $MaxFlow(S, R, c, s, d)$ returns the maximum flow from s to d on the graph (S, R) with c as the capacity function on the edges):

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

Our algorithm considers the graph (S', R'), 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), and adds a new destination node d with incoming edges from all the nodes where ψ_2 holds (the capacity of the link (s,d) is the in-degree of s, while the remaining edges have capacity 1). It is known that in graphs with all unitary edge capacities, the maximum flow is equal to the maximum number of edge-disjoint paths from the source to the destination node, see e.g. [CLRS01]. However, it is easy to see that in our network the maximum flow from a node s to d is equal to the maximum number of edge-disjoint paths from s to the set $\{t \in S' \setminus \{d\} \mid (t,d) \in R'\}$, therefore our algorithm has only to evaluate the maximum flow from each state to d.

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 $|R|$. Since in this case, the maximum flow can be calculated in time $\mathcal{O}(|R|^2)$, see e.g. [CLRS01], the overall time complexity of the algorithm is $\mathcal{O}(|R|^2 \cdot |S| \cdot |\varphi|)$. □

4.3 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 time exponential only in the size of $\hat{R}$, obtaining thus a good algorithm for practical cases.

Theorem 6. *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 in time $\mathcal{O}((4^{|\hat{R}|} \cdot |R| + 2^{|\hat{R}|^2}) \cdot |S| \cdot |\varphi|)$ and in space $\mathcal{O}(4^{|\hat{R}|} \cdot |\hat{R}| + |R| + |\varphi|)$.*

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 17-27 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         $I \leftarrow \{i \in \{k+1-|\hat{R}|, \ldots, k+1\} \mid i \geq 0$ and $\exists$ $i$ 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             $\mathcal{E} \leftarrow \{(T, T') \in \mathcal{V}^2 \mid T \cap T' = \emptyset\}$;
25             if $\exists$ a clique with size $\hat{k}$ in $(\mathcal{V}, \mathcal{E})$ 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 $(\mathcal{K}, s) \not\models_{ed}^{\hat{R}} E^{>k}\theta$). Thus the algorithm first determines a number $\hat{k}$, lines 19-21, with the property that: $(\mathcal{K}, s) \models_{ed}^{\hat{R}} E^{>k}\theta$ 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 $(\mathcal{V},\mathcal{E})$, described in lines 23 and 24, is computed, such that a vertex 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 $\mathcal{E}$ connects two disjoint such sets. Thus, $(\mathcal{K}, s) \models_{ed}^{\hat{R}} E^{>k}\theta$ iff in the graph $(\mathcal{V}, \mathcal{E})$ 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 $(\mathcal{V}, \mathcal{E})$ 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 $(\mathcal{V}, \mathcal{E})$ 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 $\mathcal{O}(4^{|\hat{R}|} \cdot |\hat{R}| + |R| + |\varphi|)$. □

In the extended version of the paper [FNP08] we show how to modify this algorithm to fit in polynomial space.

5 Discussion

In this paper we have introduced the graded-CTL as a more expressive extension of classical CTL. The results presented are in the model-checking setting with specifications in this logic. We have investigated the complexities involved in various scenarios, all from a theoretical perspective. One possible future direction to work on, is to verify in practice whether an existing model-checker tool could be augmented with these grading modalities, retaining the usual performances. We believe that this framework could turn out to be useful also in the verification of fault tolerant physical properties of networks.

As said in the introduction, in [KSV02] the satisfiability problem has been studied for the graded μ-calculus obtaining the same complexity as for the non-graded logic. We have investigated the problem in our setting of graded-CTL (reported in the extended version [FNP08]) and have proved that it is EXPTIME-complete, when the values in the formula are represented in unary. An open problem is hence to establish the complexity when the values are in binary.

Another theoretical aspect to investigate is also with respect to the Linear Temporal Logic LTL. Also here this *graded* framework is a strict extension of the standard logic, but, differently to what happens for graded CTL, a straightforward algorithm to solve the model-checking problem, seems here to involve the values representing the graded modalities.

Finally let us mention 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.

References

[CCGR99] Cimatti, A., Clarke, E.M., Giunchiglia, F., Roveri, M.: NUSMV: A new symbolic model verifier. In: Halbwachs, N., Peled, D.A. (eds.) CAV 1999. LNCS, vol. 1633, pp. 495–499. Springer, Heidelberg (1999)

[CG07] Chechik, M., Gurfinkel, A.: A framework for counterexample generation and exploration. Int. J. Softw. Tools Technol. Transf. 9(5), 429–445 (2007)

[CGP99] Clarke, E.M., Grumberg, O., Peled, D.: Model Checking. MIT Press, Cambridge (1999)

[CIW$^+$01] Copty, F., Irron, A., Weissberg, O., Kropp, N.P., Kamhi, G.: Efficient debugging in a formal verification environment. In: Margaria, T., Melham, T.F. (eds.) CHARME 2001. LNCS, vol. 2144, pp. 275–292. Springer, Heidelberg (2001)

[CLRS01] Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. MIT Press, Cambridge (2001)

[CPR03] Caprara, A., Panconesi, A., Rizzi, R.: Packing cycles in undirected graphs. Journal of Algorithms 48(1), 239–256 (2003)

[CV03] Clarke, E.M., Veith, H.: Counterexamples revisited: Principles, algorithms, applications. In: Verification: Theory and Practice, pp. 208–224 (2003)

[DRS03] Dong, Y., Ramakrishnan, C.R., Smolka, S.A.: Model checking and evidence exploration. In: ECBS, pp. 214–223 (2003)

[Fin72] Fine, K.: In so many possible worlds. Notre Dame Journal of Formal Logic 13(4), 516–520 (1972)

[FNP08] Ferrante, A., Napoli, M., Parente, M.: Graded CTL. In: Preparation (2008)

[GMV99] Ganzinger, H., Meyer, C., Veanes, M.: The two–variable guarded fragment with transitive relations. In: LICS, pp. 24–34 (1999)

[GOR97] Grädel, E., Otto, M., Rosen, E.: Two–variable logic with counting is decidable. In: LICS, pp. 306–317 (1997)

[HB91] Hollunder, B., Baader, F.: Qualifying number restrictions in concept languages. In: KR, pp. 335–346 (1991)

[Hol97] Holzmann, G.J.: The model checker SPIN. IEEE Transactions on Software Engineering 23(5), 279–295 (1997)

[KSV02] Kupferman, O., Sattler, U., Vardi, M.Y.: The complexity of the graded μ–calculus. In: CADE-18: Proceedings of the 18th International Conference on Automated Deduction, London, UK, pp. 423–437. Springer, Heidelberg (2002)

[McM] McMillan, K.: The Cadence smv model checker, `http://www.kenmcmil.com/psdoc.html`

[PST00] Pacholski, L., Szwast, W., Tendera, L.: Complexity results for first–order two–variable logic with counting. SIAM Journal of Computing 29(4), 1083–1117 (2000)

[QS82] Queille, J.P., Sifakis, J.: Specification and verification of concurrent systems in cesar. In: Proc. of the 5th Colloquium on Intern. Symposium on Programming, London, UK, pp. 337–351. Springer, Heidelberg (1982)

[Tob01] Tobies, S.: PSPACE reasoning for graded modal logics. Journal Log. Comput. 11(1), 85–106 (2001)