Submodel Enumeration for CTL Is Hard

Nicolas Fröhlich, Arne Meier

Leibniz Universität Hannover, Institut für Theoretische Informatik, Appelstrasse 9a, 30167 Hannover, Germany
nicolas.froehlich@thi.uni-hannover.de, meier@thi.uni-hannover.de

Abstract

Expressing system specifications using Computation Tree Logic (CTL) formulas, formalising programs using Kripke structures, and then model checking the system is an established workflow in program verification and has wide applications in AI. In this paper, we consider the task of model enumeration, which asks for a uniform stream of output systems that satisfy the given specification. We show that, given a CTL formula and a system (potentially falsified by the formula), enumerating satisfying submodels is always hard for CTL—regardless of which subset of CTL operators is considered. As a silver lining on the horizon, we present fragments via restrictions on the allowed Boolean functions that still allow for fast enumeration.

Introduction

In artificial intelligence, temporal logic is used as a formal language to describe and reason about the temporal behaviour of systems and processes (Barringer et al. 2000; Bérard et al. 2001). One of the key applications of temporal logic in artificial intelligence is the formal specification and verification of temporal properties of software systems, such as real-time systems (Bellini, Mattonlini, and Nesi 2000; Konur 2013; Blom 1996), reactive systems (Finger, Fisher, and Owens 1993), and hybrid systems (da Silva, Kurtz, and Lin 2021). Temporal logic can be used to specify the desired behaviour of these systems and to check that systems of that kind satisfy the specified properties. This task is known as *model checking* (MC) and is one of the most important reasoning tasks (Schnoebelen 2002). In this context, the search for satisfying submodels is a useful approach to debugging faulty systems.

One of the central temporal logics for which the model checking problem is efficiently solvable (more precisely, the problem is complete for polynomial time) is the Computation Tree Logic CTL (Clarke et al. 2018). The logic is often used in the context of program verification and, accordingly, is well suited to our study. CTL formulas enrich classical propositional logic with a variety of modal operators (next, until, global, future, release) that combine with so-called path quantifiers (existential and universal) to form CTL operators.

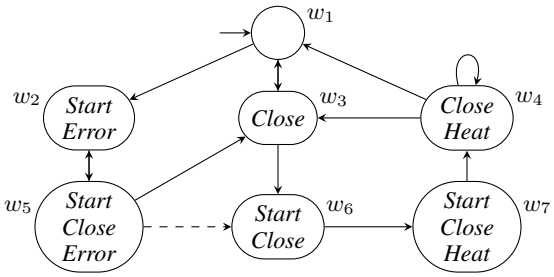

Figure 1: Kripke model of the microwave oven example. While the structure containing the dashed edge (w_5, w_6) does not satisfy the constraint $\varphi = \mathsf{AG}(\mathit{Error} \rightarrow \neg \mathit{Heat}\ \mathsf{AU}\ \neg \mathit{Start})$, the submodel without the edge does.

Kripke structures, which model the software system of interest, are essentially labelled directed graphs that have a total transition relation (Kripke 1963). A submodel of a Kripke structure is defined with respect to all possible subsets for the state set and transition relation. Note that the subset for the transition relation must still be total, otherwise it is not a valid submodel. More formally, the problem we are interested in is defined as follows. For a Kripke structure $\mathcal{M}$ and a CTL formula φ, list of all submodels $\mathcal{M}'$ of $\mathcal{M}$ such that $\mathcal{M}'$ satisfies φ. Let us illustrate the idea with an example.

Example 1. Consider the Kripke model $\mathcal{M}$ shown in Figure 1, which models the behavior of a microwave oven, a well-known example from (Clarke et al. 2018). Next, consider the constraint $\varphi = \mathsf{AG}(\mathit{Error} \rightarrow \neg \mathit{Heat}\ \mathsf{AU}\ \neg \mathit{Start})$, which says that any path starting in a world labeled with the *Error* proposition must first reach a world where $\neg Start$ holds, before *Heat* becomes true. With the dashed edge from w_5 to w_6 this constraint obviously does not hold in $\mathcal{M}$. In contrast, the submodel $\mathcal{M}'$ of $\mathcal{M}$ without the dashed edge satisfies φ. Thus, as an automated repair or a suggestion in debugging one might want to consider the submodels of $\mathcal{M}$.

Also other areas of research benefit from this kind of approach as follows. For bounded model checking (Biere et al. 2003), the size of the state space can easily become very large for complex systems. For such systems, Biere et al. suggest combining model checking with SAT solvers, which allow faster exploration of the state space. Similarly, Gupta et al. 2000 showed that similar things have been ob-

served in the context of BDD-based symbolic algorithms for image processing. While one might think that the work of Sullivan et al. 2019 is closely related to our setting, the authors work with propositional logic in the setting of the specification language Alloy, which is based on first-order logic. This is somewhat different from us, as we work with CTL. However, there is work on CTL-live model checking for first-order logic validity checking (Vakili and Day 2014), but this would be a different direction to our approach. Lauri and Dutta 2019, following an ML perspective, devise an ML framework that attempts to shrink the search space and augment the solver with some help. Recently, this topic has been investigated in the context of plain modal logic (Fröhlich and Meier 2022).

Classically, the task of enumerating models is very different from counting the number of existing models or deciding on the existence of such models. Although enumeration algorithms are usually exponential time algorithms, this does not preclude practical applications. In addition to theoretical studies, there are a number of application scenarios (Fomin and Kratsch 2010), e.g., recommender systems (Friedrich and Zanker 2011), ASP (Alviano and Dodaro 2016), or ML (Lauri and Dutta 2019). The formal foundations were originally laid by Johnson et al. (1988). Intuitively, an enumeration algorithm is deterministic and produces a uniform stream of output solutions avoiding duplicates. This solution flow is mathematically modelled by the notion of the *delay*, i.e., an upper bound on the elapsed time between printing two successive solutions (or the time before the first, resp., after the last, solution is returned). In 2019, Creignou et al. introduced a framework for intrinsically hard enumeration problems. Here, the polynomial hierarchy and the concept of oracle machines have been utilised to present notions that allow for proving intractability bounds for enumeration problems. The complexity class DelP describes "efficient" enumeration, that is, a delay that is polynomially bounded in the input length, while the complexity class DelNP contains intractable and, accordingly, difficult enumeration problems. Solutions to instances of problems in DelNP cannot efficiently be produced unless the (classical) complexity classes P and NP coincide.

While the tractability of MC for CTL formulas is known to be P-complete, the complexity of enumerating satisfying submodels is still open.

Contributions. In this paper, we fill this gap and present a thorough study of the complexity of the submodel enumeration problem in the context of CTL. We will see that in general the problem is complete for the class DelNP; so it is reasonable to consider restrictions that aim for tractable enumeration cases. However, our answer in this direction is on the negative side, showing that any restriction on the CTL operator side does not allow faster enumeration algorithms (assuming $\mathsf{P} \neq \mathsf{NP}$). Finally, we identify some further Boolean restrictions that still allow for DelP algorithms.

Related works. The work of Schnoebelen (2002) considers the classical model checking question for temporal logics. There is a study on the complexity of fragments of the model checking problem (Krebs, Meier, and Mundhenk 2019) for CTL, but it has no direct impact on our results as the problems are situated in P (or classes within).

Organisation. At first, we introduce the necessary preliminaries on temporal logic and enumeration complexity. Then we prove our dichotomy theorem. Finally, we conclude. Due to space constraints some proof details are omitted (marked with $\star$) and can be found in the technical report (Fröhlich and Meier 2023).

Preliminaries

In this section, we assume basic familiarity with computational complexity (Papadimitriou 2007) and will make use of the framework for hard enumeration problems (Creignou et al. 2019). Furthermore, we will define the temporal logic CTL, introduce submodels, and enumeration complexity.

Computational Tree Logic. We follow standard notation of model checking (Clarke et al. 2018). Let PROP be an infinite, countable set of propositions. The set of well-formed CTL formulas is defined with the following BNF

$$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathcal{P}\mathcal{T}\,\varphi \mid \varphi\,\mathcal{P}\mathcal{T}'\,\varphi,$$

where $p \in \mathrm{PROP}, \mathcal{P} \in \{\mathsf{E}, \mathsf{A}\}, \mathcal{T} \in \{\mathsf{X}, \mathsf{F}, \mathsf{G}\}, \mathcal{T}' \in \{\mathsf{U}, \mathsf{R}\}$. This results in ten CTL operators, consisting of six unary operators EX, EX, EF, AF, EG, AG and four binary operators EU, AU, ER, AR. The set $\mathcal{ALL}$ contains all ten CTL operators. We will call $\wedge, \vee, \neg$ Boolean connectors.

Now, we defined a special version of Kripke models.

Definition 2. A *rooted Kripke model* is a tuple $\mathcal{M} = (W, R, \eta, r)$ where

- W is a non-empty set of *worlds* (or *states*),
- $R \subseteq W \times W$ is a total, binary transition relation on W
- $\eta\colon W \to 2^{\mathrm{PROP}}$ is an assignment function, that maps each world w to a set $\eta(w)$ of propositions, and
- $r \in W$ is the *root*.

Definition 3. Let $\mathcal{M} = (W, R, \eta, r)$ be a rooted Kripke model. A *path* π in $\mathcal{M}$ is an infinite sequence of worlds $w_1, w_2, \ldots$ such that $(w_i, w_{i+1}) \in R$ for all $i \geq 1$. We write $\pi[i]$ to denote the ith world on the path π. For a world $w \in W$ we define $\Pi(w) \coloneqq \{\, \pi \mid \pi[1] = w \,\}$ as the (possibly infinite) set of all infinite paths of $\mathcal{M}$ starting with w.

Definition 4. Let $\mathcal{M}$ be a rooted Kripke model and φ, ψ be CTL formulas.

$\mathcal{M}, w \models \top$	always,
$\mathcal{M}, w \models p$	iff $p \in \eta(w)$ with $p \in \mathrm{PROP}$,
$\mathcal{M}, w \models \neg\varphi$	iff $\mathcal{M}, w \not\models \varphi$,
$\mathcal{M}, w \models \varphi \wedge \psi$	iff $\mathcal{M}, w \models \varphi$ and $\mathcal{M}, w \models \psi$,
$\mathcal{M}, w \models \varphi \vee \psi$	iff $\mathcal{M}, w \models \varphi$ or $\mathcal{M}, w \models \psi$,
$\mathcal{M}, w \models \mathsf{EX}\,\varphi$	iff $\exists \pi \in \Pi(w) : \mathcal{M}, \pi[2] \models \varphi$,
$\mathcal{M}, w \models \mathsf{AX}\,\varphi$	iff $\forall \pi \in \Pi(w) : \mathcal{M}, \pi[2] \models \varphi$,
$\mathcal{M}, w \models \mathsf{EF}\,\varphi$	iff $\exists \pi \in \Pi(w)\ \exists k \geq 1 : \mathcal{M}, \pi[k] \models \varphi$,
$\mathcal{M}, w \models \mathsf{AF}\,\varphi$	iff $\forall \pi \in \Pi(w)\ \exists k \geq 1 : \mathcal{M}, \pi[k] \models \varphi$,
$\mathcal{M}, w \models \mathsf{EG}\,\varphi$	iff $\exists \pi \in \Pi(w)\ \forall k \geq 1 : \mathcal{M}, \pi[k] \models \varphi$,
$\mathcal{M}, w \models \mathsf{AG}\,\varphi$	iff $\forall \pi \in \Pi(w)\ \forall k \geq 1 : \mathcal{M}, \pi[k] \models \varphi$,
$\mathcal{M}, w \models \varphi\,\mathsf{EU}\,\psi$	iff $\exists \pi \in \Pi(w)\ \exists k \geq 1 : \mathcal{M}, \pi[k] \models \psi$ and $\forall i < k : \mathcal{M}, \pi[i] \models \varphi$,

$$\mathcal{M}, w \models \varphi \,\mathsf{AU}\, \psi \text{ iff } \forall \pi \in \Pi(w)\ \exists k \geq 1 : \mathcal{M}, \pi[k] \models \psi \text{ and } \forall i < k : \mathcal{M}, \pi[i] \models \varphi,$$
$$\mathcal{M}, w \models \varphi \,\mathsf{ER}\, \psi \text{ iff } \exists \pi \in \Pi(w)\ \forall k \geq 1 : \mathcal{M}, \pi[k] \models \psi \text{ or } \exists i < k : \mathcal{M}, \pi[i] \models \varphi,$$
$$\mathcal{M}, w \models \varphi \,\mathsf{AR}\, \psi \text{ iff } \forall \pi \in \Pi(w)\ \forall k \geq 1 : \mathcal{M}, \pi[k] \models \psi \text{ or } \exists i < k : \mathcal{M}, \pi[i] \models \varphi.$$

Furthermore, $\bot \coloneqq \neg\top$ is constant false. Also omit the root in $\mathcal{M}, r \models \varphi$ and just write $\mathcal{M} \models \varphi$ instead. A formula φ is then said to be *satisfied by model* $\mathcal{M}$, if $\mathcal{M} \models \varphi$ is true.

Notice an observation regarding the semantics of CTL.

Observation 5. The following equivalences are true:

$$\mathsf{EX}\,\varphi \equiv \neg\,\mathsf{AX}(\neg\varphi), \mathsf{AG}\,\varphi \equiv \neg\,\mathsf{EF}(\neg\varphi),$$
$$\mathsf{EG}\,\varphi \equiv \neg\,\mathsf{AF}(\neg\varphi), \mathsf{EG}\,\varphi \equiv \bot\,\mathsf{ER}\,\varphi, \mathsf{AG}\,\varphi \equiv \bot\,\mathsf{AR}\,\varphi,$$
$$\mathsf{EF}\,\varphi \equiv \top\,\mathsf{EU}\,\varphi, \mathsf{AF}\,\varphi \equiv \top\,\mathsf{AU}\,\varphi,$$
$$\varphi\,\mathsf{ER}\,\psi \equiv \neg(\neg\varphi\,\mathsf{AU}\,\neg\psi), \varphi\,\mathsf{AR}\,\psi \equiv \neg(\neg\varphi\,\mathsf{EU}\,\neg\psi).$$

Now, we formally introduce submodels of Kripke models. Given two Kripke models $\mathcal{M} = (W, R, \eta, r)$ and $\mathcal{M}' = (W', R', \eta, r)$. We call $\mathcal{M}'$ a *submodel (of* $\mathcal{M}$*)*, if $W' \subseteq W$, $R' \subseteq R$, and R' is total. For a function $f\colon A \to B$ we write $f\restriction_C$, given $C \subseteq A$, for the restriction of f to domain C.

Definition 6. Let $\mathcal{M} = (W, R, \eta, r)$ be a Kripke model. $\mathcal{M}' = (W', R', \eta\restriction_{W'}, r)$ is a *connected submodel* of $\mathcal{M}$, denoted by $\mathcal{M}' \subseteq \mathcal{M}$, if (1.) $W' \neq \emptyset$, (2.) $\mathcal{M}'$ is a submodel of $\mathcal{M}$, and (3.) for all $w \in W'$ there exists a path $\pi \in \Pi(r)$ and $i \geq 1$ with $\pi[i] = w$.

Clearly, worlds that violate (3.) cannot have influence on the satisfiability of CTL formulas. Yet, an enumeration algorithm printing connected submodels could trivially be extended to include non-connected submodels.

Additionally we want to introduce an alternative notation for submodels, $\mathcal{M}' = \mathcal{M} - D$, with $D = (D_W, D_R)$ for $r \notin D_W$ a tuple consisting of a set of worlds and a set of tuples, and $W' = W \setminus D_W$ and $R' = R \setminus D_R$, for $\mathcal{M} = (W, R, \eta, r)$ and $\mathcal{M}' = (W', R', \eta\restriction_{W'}, r)$. Here, D is called the set of *deletions*.

A submodel $\mathcal{M}'$ is *satisfying* φ if $\mathcal{M}' \models \varphi$. The formula φ is often omitted, if it can be deduced from the context.

Enumeration Complexity. The Turing machine, as one of the standard machine models used in complexity theory, proves to be problematic for the setting of enumeration algorithms. Its linear nature in accessing data prevents a polynomial delay when traversing exponentially large data sets, even if the actual data read is small. As a result, random access machines (RAMs) are the common machine model of choice (Strozecki 2019).

Definition 7. Let Σ be a finite alphabet. An *enumeration problem (EP)* is a tuple $\mathcal{E} = (I, \mathrm{Sol})$, where

- $I \subseteq \Sigma^*$ is the set of *instances*,
- $\mathrm{Sol}\colon I \to \mathcal{P}(\Sigma^*)$ is a function that maps each instance $x \in I$ to a set of *solutions (of* x*)*, and
- there exists a polynomial p such that $\forall x \in I\ \forall y \in \mathrm{Sol}(x)$ we have that $|y| \leq p(|x|)$.

Note that sometimes one is interested in dropping the last requirement of the previous definition (Strozecki 2019).

Definition 8. Let $\mathcal{E} = (I, \mathrm{Sol})$ be an EP. An algorithm $\mathcal{A}$ is called an *enumeration algorithm* for $\mathcal{E}$, if for every instance $x \in I$: $\mathcal{A}(x)$ terminates after a finite sequence of steps and $\mathcal{A}(x)$ prints exactly $\mathrm{Sol}(x)$ without duplicates, where $\mathcal{A}(x)$ denotes the *computation of* $\mathcal{A}$ *on input* x.

We now define the mentioned delay of an enumeration algorithm.

Definition 9. Let $\mathcal{E} = (I, \mathrm{Sol})$ be an EP, $\mathcal{A}$ be an enumeration algorithm for $\mathcal{E}$, $x \in I$ be an instance and $n = |\mathrm{Sol}(x)|$ the number of solutions of x. We define the

- i*th delay* of $\mathcal{A}(x)$ as the elapsed time between the output of the ith and $(i+1)$st solution of $\mathrm{Sol}(x)$,
- 0*th delay* as the *precomputation time*, i.e, the elapsed time before the first output of $\mathcal{A}(x)$, and
- n*th delay* as the *postcomputation time*, i.e., the elapsed time after the last output of $\mathcal{A}(x)$ until it terminates.

We say that $\mathcal{A}$ has *delay* f, for $f\colon \mathbb{N} \to \mathbb{N}$, if for all $x \in I$ and all $0 \leq i \leq n$ the ith delay of $\mathcal{A}(x)$ is in $O(f(|x|))$.

Hard enumeration. We will shortly introduce the framework of hard enumeration by Creignou et al. (2019). The idea is to analyse EPs beyond polynomial delay by introducing a hierarchy of complexity classes similar to the polynomial-time hierarchy and reduction notions for EPs.

We begin by defining two decision problems that naturally arise in the context of enumeration. Let $\mathcal{E} = (I, \mathrm{Sol})$ be an EP over the alphabet Σ. The first decision problem EXIST_$\mathcal{E}$ asks, given an instance x, for the existence of any solutions, that is, $\mathrm{Sol}(x)$ is nonempty.

The second decision problem is concerned with obtaining new solutions. This is the question whether, given an instance x and a partial solution y, can we extend the partial solution by a word $y' \subseteq \Sigma^*$ such that yy' is a solution of $\mathcal{E}$, where yy' denotes the concatenation of y and y'.

Problem:	EXTENDSOL_$\mathcal{E}$
Input:	Instance x, partial solution y
Question:	Is there some y' such that $yy' \in \mathrm{Sol}(x)$?

As mentioned before, we use RAMs instead of Turing machines in the context of enumeration complexity. We now want to further extend the underlying machine model, by introducing decision oracles. Classically, when analysing runtime, or in this case delay, algorithm calls to its oracle are always charged as a single step, regardless of the time the oracle takes. Our machines can write into special registers and the oracle will consider these as well as all consecutive non-empty registers as its input. A query to the oracle then occurs when the machine enters a special question state and will transition into a positive/negative state if the oracle answers "yes"/"no". Now, start with enumeration complexity classes with oracles.

Definition 10 (Creignou et al. 2019, Def. 2)**.** Let $\mathcal{E}$ be an EP, and $\mathcal{C}$ a decision complexity class. Then we say that $\mathcal{E} \in \mathsf{Del}\mathcal{C}$ if there is a RAM M with oracle L in $\mathcal{C}$ and a polynomial p, such that for any instance x, the RAM M enumerates $\mathrm{Sol}(x)$ with delay $p(|x|)$. Moreover, the size of every oracle call is bound by $p(|x|)$.

In this paper, the enumeration classes of interest are when $\mathcal{C}$ is either P, or NP; so DelP and DelNP.

The following Proposition 11 as well as Proposition 15 are both simplified versions of results presented by Creignou et al. (2019). While Creignou et al. considered the full polynomial hierarchies in their proofs, here we are only concerned with the P and NP cases.

Proposition 11 (Creignou et al. 2019, Prop. 6)**.** *Let $\mathcal{E} = (I, \mathrm{Sol})$ be an EP and $\mathcal{C} \in \{\mathsf{P}, \mathsf{NP}\}$. If $\textsc{ExtendSol}\_\mathcal{E} \in \mathcal{C}$ then $\mathcal{E} \in \mathsf{Del}\mathcal{C}$.*

Proposition 11 allows for membership results for EPs using the corresponding decision problem ExtendSol. This technique will prove particularly useful when showing membership in DelNP, as constructing enumeration algorithms with oracles can be quite difficult.

We now give the necessary definitions to show hardness results for EPs. The first definition introduces yet another machine model, which can then be used to define a reduction from one EP to another.

Definition 12 (Creignou et al. 2019, Def. 6)**.** Let $\mathcal{E}$ be an EP. An *Enumeration Oracle Machine with an enumeration oracle $\mathcal{E}$*, abbreviated as EOM_$\mathcal{E}$, is a RAM that has a sequence of new registers $A_e, O^e(0), O^e(1), \ldots$ and a new instruction NOO (next oracle output). An EOM_$\mathcal{E}$ is *oracle-bounded*, if the size of all inputs to the oracle is at most polynomial in the size of the input to the EOM_$\mathcal{E}$.

Note that the sequence of registers as input is only necessary for EOM_$\mathcal{E}$ that are not oracle-bounded, to allow input sizes larger than polynomial.

Definition 13 (Creignou et al. 2019, Def. 7)**.** Let $\mathcal{E} = (I, \mathrm{Sol})$ be an EP and $\rho_1, \rho_2, \ldots$ be the run of an EOM_$\mathcal{E}$ and assume that the kth instruction is NOO, that is, $\rho_k = \text{NOO}$. Denote with x_i the word stored in $O^e(0), O^e(1), \ldots$ at step i. Let $K = \{\rho_j \in \{\rho_1, \ldots, \rho_{k-1}\} \mid \rho_j = \text{NOO} \text{ and } x_j = x_k\}$. Then the *oracle output y_k in ρ_k* is defined as an arbitrary $y_k \in \mathrm{Sol}(x_k)$ such that y_k has not been the oracle output in any $\rho_j \in K$. If no such y_k exists, then the oracle output in ρ_k is undefined.

On executing NOO in step ρ_k, if the oracle output y_k is undefined, then register A_e contains some special symbol in step ρ_{k+1}; otherwise it contains y_k.

Definition 14 (D-reductions)**.** Let $\mathcal{E}$ and $\mathcal{E}'$ be EPs. We say that $\mathcal{E}$ reduces to $\mathcal{E}'$ via *D-reduction*, $\mathcal{E} \leq_D \mathcal{E}'$, if there is an oracle-bounded EOM_$\mathcal{E}'$ that enumerates $\mathcal{E}$ in DelP and is independent of the order in which the $\mathcal{E}'$-oracle enumerates its answers.

The next result shows that one can use the decision problem Exist_$\mathcal{E}$ to show hardness of the corresponding EP $\mathcal{E}$.

Proposition 15 (Creignou et al. 2019, Theorem 13)**.** *Let $\mathcal{E} = (I, \mathrm{Sol})$ be an EP. If $\textsc{Exist}\_\mathcal{E}$ is NP-hard, then $\mathcal{E}$ is DelNP-hard via D-reductions.*

Any following result that states DelNP-hardness for an EP will be with respect to D-reductions.

Complexity of Submodel Enumeration

In this section, we will present our results regarding the submodel EP with respect to CTL formulas.

Problem:	E-Submodels
Input:	Kripke model $\mathcal{M}$, CTL formula φ
Task:	Output all $\mathcal{M}' \subseteq \mathcal{M}$ such that $\mathcal{M}' \models \varphi$?

Let $\mathcal{O}$ be a set of CTL operators. Then E-Submodels($\mathcal{O}$) is E-Submodels but only for CTL formulas using operators from $\mathcal{O}$ (besides any of the Boolean connectors). The same applies to the next two auxiliary decision problems.

Problem:	$\exists$Submodel
Input:	Kripke model $\mathcal{M}$, CTL formula φ
Question:	Does $\mathcal{M}' \subseteq \mathcal{M}$ exist such that $\mathcal{M}' \models \varphi$?

Problem:	ExtSubmodel
Input:	Kripke model $\mathcal{M}$, CTL formula φ, set of deletions D
Question:	Does an extension $D' \supseteq D$ exist such that $\mathcal{M} - D' \models \varphi$?

The first result will show membership in the class DelNP for the unrestricted version and will make use of the auxiliary problem ExtSubmodel.

Theorem 16. E-Submodels $\in$ DelNP.

Proof. The algorithm deciding ExtSubmodel works as follows. For given Kripke model $\mathcal{M} = (W, R, \eta, r)$, CTL formula φ, and set of deletions $D = (D_W, D_R)$, guess $W' \subseteq W$ and $R' \subseteq R$. Afterwards compute $D' \coloneqq (W' \cup D_W, R' \cup D_R)$ and accept if and only if $\mathcal{M} - D' \models \varphi$.

For correctness, consider that if an extension D' exists such that $\mathcal{M} - D' \models \varphi$, it can be computed by nondeterministically guessing the worlds and relations of that extension. Guessing W' and R', computing D' and checking if $\mathcal{M} - D' \models \varphi$ can all clearly be done in polynomial time (MC is in P for CTL). By Proposition 11 this is sufficient to prove that E-Submodels $\in$ DelNP. □

Fragment AG. We will show hardness by relating submodels to assignments of propositional formulas, such that a submodel is satisfying, if and only if the corresponding assignment satisfies the given propositional formulas. Formally this is a reduction from the well-known NP-complete problem SAT (Cook 1971; Levin 1973).

Definition 17. Let φ be a propositional formula with propositions $\mathrm{PROP}(\varphi) = \{x_1, x_2, \ldots, x_n\}$. We define the Kripke model $\mathcal{M}(\varphi) \coloneqq (W, R, \eta, w_0)$ as follows:

$$
\begin{aligned}
W &\coloneqq \{w_0\} \cup \{w_i^0, w_i^1 \mid 1 \leq i \leq n\},\\
R &\coloneqq \{(w_0, w_1^k) \mid k \in \{0,1\}\}\\
&\quad \cup \{(w_i^k, w_{i+1}^l) \mid k, l \in \{0,1\}, 1 \leq i < n\}\\
&\quad \cup \{(w_n^k, w_n^k) \mid k \in \{0,1\}\},\\
\eta(w_i^k) &\coloneqq \{x_i, x_i^k\} \text{ for } 1 \leq i \leq n, k \in \{0,1\}.
\end{aligned}
$$

Figure 2 depicts such a Kripke model together with one of its submodels. Notice that the formula $\varphi = (x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2)$ of $\mathcal{M}$ is satisfied given the assignment $\mathfrak{I}(x_1) = 1, \mathfrak{I}(x_2) = 0$, which the submodel $\mathcal{M}'$ "encodes" by containing the worlds w_1^1, w_2^0 and not w_1^0, w_2^1. The proof of the

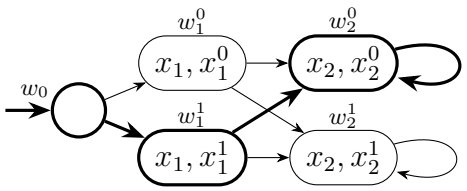

Figure 2: Kripke model $\mathcal{M}((x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2))$ and a submodel $\mathcal{M}'$ of $\mathcal{M}$ in bold.

following theorem uses this connection by constructing formulas that are satisfied in a submodel if and only if the corresponding assignment evaluates to 1, giving rise to a nice reduction from SAT to $\exists$SUBMODEL(AG).

Theorem 18. E-SUBMODELS(AG) *is* DelNP*-complete.*

Proof. The upper bound follows from Theorem 16. By Proposition 15, showing NP-hardness of $\exists$SUBMODEL(AG) implies DelNP-hardness of E-SUBMODELS(AG).

Let φ be propositional formula in negation normal form. φ_{AG} is constructed by substituting x_i with $\mathsf{AG}(x_i \to x_i^1)$ and $\neg x_i$ with $\mathsf{AG}(x_i \to x_i^0)$ in φ for all $x_i \in \mathrm{PROP}(\varphi)$. Note that while we have not formally introduced implication, it can simply be taken as $\neg x_i \vee x_i^1$. Also recall that atomic negation can always be simulated by introducing new propositions and labeling the model accordingly.

We now show that $\langle\varphi\rangle \in$ SAT, if and only if we have that $\langle \mathcal{M}(\varphi), \varphi_{\mathsf{AG}}\rangle \in \exists$SUBMODEL(AF).

Suppose $\varphi \in$ SAT. Then there exists an assignment $\mathfrak{I}$ such that $\mathfrak{I}(\varphi) = 1$. Using $\mathfrak{I}$, we construct a submodel $\mathcal{M}' = (W', R', \eta, w_0)$ as follows:

$$W' \coloneqq W \setminus \{w_i^k \mid 1 \leq i \leq n, k = 1 - \mathfrak{I}(x_i)\},$$
$$R' \coloneqq R \cap (W' \times W').$$

That is, we remove the worlds w_i^1, if $\mathfrak{I}(x_i) = 0$ and w_i^0, if $\mathfrak{I}(x_i) = 1$.

Observe that $\mathcal{M}' \models (x_i \to x_i^1)$, if and only if $\mathfrak{I}(x_i) = 1$, since all worlds of $\mathcal{M}'$ labeled with x_i are also labeled with x_i^1. Analogously, $\mathcal{M}' \models (x_i \to x_i^0)$, if and only if $\mathfrak{I}(x_i) = 0$. Because φ_{AG} differs from φ only in its atoms, it follows that $\mathcal{M}' \models \varphi_{\mathsf{AG}}$ must be true.

In the same way, if there is a submodel $\mathcal{M}'$ such that $\mathcal{M}' \models \varphi_{\mathsf{AG}}$, we can construct an assignment $\mathfrak{I}$ from a path $\pi \in \Pi(\mathcal{M}')$ such that $\mathfrak{I}(\varphi)$ evaluates to 1.

To conclude the reduction, observe that the construction of $\mathcal{M}(\varphi)$ and φ_{AG} can both clearly be done in polynomial time, showing SAT $\leq_m^{\mathsf{P}}$ $\exists$SUBMODEL(AG) and proving DelNP-hardness of E-SUBMODELS(AG). □

Notice that the reduction requires AG as operator and only the binary Boolean connectors $\wedge$, $\vee$ and atomic negations, which can be removed by a simple relabeling.

Fragment AF. We will show hardness via relating submodels to deciding the problem HAMPATH (Karp 1972).

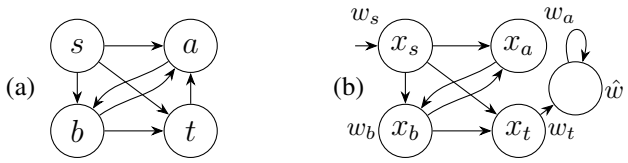

Figure 3: (a) G with Hamiltonian path s, a, b, t. (b) Kripke model $\mathcal{M}(H)$ of $H = \langle G, s, t\rangle$.

Definition 19. Let $H = \langle G, s, t\rangle$ be a HAMPATH instance, with $G = (V, E)$ a graph and $s, t \in V$. We define the Kripke model $\mathcal{M}(H) \coloneqq (W, R, \eta, w_s)$ as follows:

$$W \coloneqq \{w_v \mid v \in V\} \cup \{\hat{w}\}, \quad \eta(w_v) \coloneqq \{x_v\}, \text{ for } v \in V,$$
$$R \coloneqq \{(w_u, w_v) \mid (u, v) \in E, u \neq t\} \cup \{(w_t, \hat{w}), (\hat{w}, \hat{w})\}.$$

The underlying graph of this model is almost G itself, except that a new world $\hat{w}$ is added, which became the only successor of w_t and has only one relation to itself. Figure 3 (b) depicts such a model for the graph in Figure 3 (a).

Theorem 20. E-SUBMODELS(AF) *is* DelNP*-complete.*

Proof. The upper bound follows directly from Theorem 16.

Let $H = \langle G, s, t\rangle$ be an instance of HAMPATH with $G = (V, E)$, $s, t \in V$ and $n = |V|$. Further let $\mathcal{M}(H)$ be the Kripke model obtained from H as described in Definition 19. Now, construct the formula $\varphi \coloneqq \bigwedge_{v \in V} \mathsf{AF}\, x_v$.

Notice that a submodel $\mathcal{M}' \subseteq \mathcal{M}$ satisfies φ only if it is acyclic. That is because all paths have to contain a world labeled with t, which only holds at w_t. The world w_t has a single outgoing edge to $\hat{w}$, where all paths "end" in an infinite loop, making other cycles impossible.

Next, we have that paths must contain worlds where v for $v \in V$ holds. This can only be achieved if all path contain the worlds w_v for $v \in V$. It follows that satisfying submodels must contain each world at least once, but because of acyclicity they can also only contain each world at most once. Thus satisfying submodels must be single path from s to t containing each world exactly once, i.e., they must be Hamiltonian. Construction of $\mathcal{M}(H)$ and φ is in P. □

Notice that the reduction requires AF as operator and the Boolean connector $\wedge$.

Fragment AX. We again use HAMPATH to show hardness. By concatenating the AX operator $n-1$ times followed by x_t, we enforce that submodels must satisfy x_t on all path at position n. Considering the construction of $\mathcal{M}(H)$ this is only possible, if paths are acyclic and contain w_t only at position n. This implies that all satisfying submodels describe Hamiltonian paths from s to t.

Theorem 21. E-SUBMODELS(AX) *is* DelNP*-complete.*

Proof. The upper bound follows from Theorem 16.

Suppose H is an instance of HAMPATH and $n = |V|$. Then let $\mathcal{M}(H)$ be the Kripke model as defined in Definition 19 and let $\varphi \coloneqq \mathsf{AX}^{n-1}\, x_t$ be a formula, where AX^{n-1} denotes the $n-1$-times concatenation of the AX operator. Furthermore, let $\mathcal{M}' \subseteq \mathcal{M}(H)$ be a satisfying submodel.

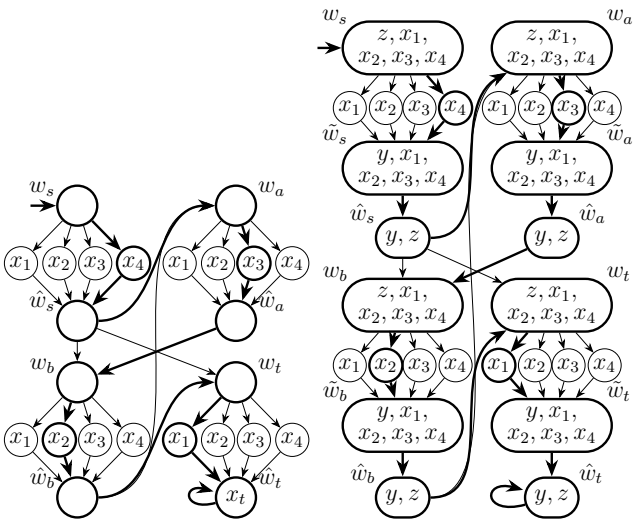

Figure 4: Kripke model (left) of $\mathcal{M}_{\mathsf{AU}}(H)$ and (right) of $\mathcal{M}_{\mathsf{AR}}(H)$ for the graph in Figure 3 (a). The highlighted worlds and relations form a submodel that induces a Hamiltonian path for the instance $H = \langle G, s, t\rangle$ and satisfy (left) $\varphi_4 = ((((\top\,\mathsf{AU}\,x_t)\,\mathsf{AU}\,x_1)\,\mathsf{AU}\,x_2)\,\mathsf{AU}\,x_3)\,\mathsf{AU}\,x_4$ and (right) $\varphi_4 = (\cdots(\top\,\mathsf{AR}\,z)\,\mathsf{AR}\,y)\,\mathsf{AR}\,x_1)\,\mathsf{AR}\,z)\,\mathsf{AR}\,y)\,\mathsf{AR}\,x_2)\,\mathsf{AR}\,z)\,\mathsf{AR}\,y)\,\mathsf{AR}\,x_3)\,\mathsf{AR}\,z)\,\mathsf{AR}\,y)\,\mathsf{AR}\,x_4$.

First, show that $\pi[n] = w_t$ for all paths $\pi \in \Pi(\mathcal{M}')$.

$$
\begin{aligned}
&\mathcal{M}', w_s \models \mathsf{AX}^{n-1}\,x_t \\
&\Leftrightarrow \forall \pi \in \Pi(w_s) : \mathcal{M}', \pi[2] \models \mathsf{AX}^{n-2}\,x_t && \text{(Def.)} \\
&\Leftrightarrow \forall \pi \in \Pi(w_s) \forall \sigma \in \Pi(\pi[2]) : \mathcal{M}', \sigma[2] \models \mathsf{AX}^{n-3}\,x_t \\
&\Leftrightarrow \forall \pi \in \Pi(w_s) : \mathcal{M}', \pi[3] \models \mathsf{AX}^{n-3}\,x_t && \text{(prefix)} \\
&\Leftrightarrow \forall \pi \in \Pi(w_s) : \mathcal{M}', \pi[n-1] \models \mathsf{AX}\,x_t && \text{(repeat)} \\
&\Leftrightarrow \forall \pi \in \Pi(w_s) : \mathcal{M}', \pi[n] \models x_t
\end{aligned}
$$

By the definition of $\mathcal{M}(H)$, only $\eta(w_t) = x_t$. Thus $\forall \pi \in \Pi(w_s)$ we have $\pi[n] = w_t$.

Note that w_t cannot be on any path before that. Otherwise the path could only continue to $\hat{w}$ and "end" there. Also, submodels again cannot have cycles, otherwise there would be a path that never reaches w_t. So we can conclude that on all paths in $\mathcal{M}'$ the first n elements must be different. With n worlds other than $\hat{w}$, this leads to satisfying submodels that are Hamiltonian paths from w_s to w_t, showing correctness of the reduction. The reduction can be computed in P. □

Notice that the reduction merely requires AX as operator and no Boolean connectors are used.

Fragment AU. We continue to use HAMPATH. For fragment AU, we construct a new submodel $\mathcal{M}_{\mathsf{AU}}(H)$ that expands each node into a "diamond" construct with a world for the incoming relations and a world for the outgoing relations, as well as a number of intermediate worlds equal to the total number of vertices in G. We then construct an AU formula such that all paths in a satisfying submodel are acyclic and contain a different intermediate world at each "diamond", thereby describing a Hamiltonian path of G.

Definition 22. Let $G = (V, E)$ be a graph, $s, t \in V$, $n = |V|$, and $H = \langle G, s, t\rangle$ be an instance of HAMPATH. We define the model $\mathcal{M}_{\mathsf{AU}}(H) \coloneqq (W, R, \eta, w_s)$ as follows:

$$
\begin{aligned}
W &\coloneqq \{w_v, \hat{w}_v, w_{v,i} \mid v \in V, 1 \le i \le n\}, \\
R &\coloneqq \{(w_v, w_{v,i}), (w_{v,i}, \hat{w}_v) \mid v \in V, 1 \le i \le n\}, \\
&\quad \cup \{(\hat{w}_u, w_v) \mid (u, v) \in E, u \neq t\} \cup \{(\hat{w}_t, \hat{w}_t)\}, \\
\eta(w_{v,i}) &\coloneqq \{x_i\} \text{ for } 1 \le i \le n, \quad \eta(\hat{w}_t) \coloneqq \{x_t\}.
\end{aligned}
$$

Figure 4 depicts the submodel $\mathcal{M}_{\mathsf{AU}}(H)$ constructed from the graph in Figure 3 (a), with $H = \langle G, s, t\rangle$.

Theorem 23 (⋆). E-SUBMODELS(AU) *is* DelNP-*complete.*

Proof. The upper bound follows directly from Theorem 16.

Let $H = \langle G, s, t\rangle$ be an instance of HAMPATH and φ_n be the formula of interest here, for

$$
\varphi_i \coloneqq \begin{cases} \varphi_{i-1}\,\mathsf{AU}\,x_i & \text{if } i > 0, \\ \top\,\mathsf{AU}\,x_t & \text{if } i = 0. \end{cases}
$$

The reduction then is $H \mapsto \langle \mathcal{M}_{\mathsf{AU}}(H), \varphi_n\rangle$. □

Notice that the reduction merely requires AU as operator and no Boolean connectors are used.

Fragment AR. We use a similar "diamond" expansion of the nodes in G, as in the proof for AU. Here an extra world is added to the construct between the middle worlds and the world for outgoing relations. In addition, the labeling is extended to make AR behave in the indented way. That is, we want to repeatedly force the left hand side of the AR operator in our constructed formula φ to only hold in specific subsequent worlds to simulate the behavior of the AX operator. The construction of φ, also requires that worlds labeled with $x_1, x_2, \ldots, x_n$ are on the paths, similar to the proof of the AU fragment.

Definition 24. Let $G = (V, E)$ be a graph with $n = |V|$ and $H \coloneqq \langle G, s, t\rangle$ an instance of HAMPATH. Define $\mathcal{M}_{\mathsf{AR}}(H) \coloneqq (W, R, \eta, w_s)$ as follows (see Figure 4 for an example):

$$
\begin{aligned}
W &\coloneqq \{w_v, \tilde{w}_v, \hat{w}_v, w_{v,i} \mid v \in V, 1 \le i \le n\}, \\
R &\coloneqq \left\{ \begin{matrix} (w_v, w_{v,i}), \\ (w_{v,i}, \tilde{w}_v), \\ (\tilde{w}_v, \hat{w}_v) \end{matrix} \;\middle|\; v \in V, 1 \le i \le n \right\} \\
&\quad \cup \{(\hat{w}_u, w_v) \mid (u, v) \in E \text{ and } u \neq t\} \\
&\quad \cup \{(\hat{w}_t, \hat{w}_t)\}, \\
\eta(w_{v,i}) &\coloneqq \{x_i\} \text{ for } 1 \le i \le n, \quad \eta(\hat{w}_v) \coloneqq \{z, y\}, \\
\eta(w_v) &\coloneqq \{z, x_1, \ldots, x_n\}, \eta(\tilde{w}_v) \coloneqq \{y, x_1, \ldots, x_n\}.
\end{aligned}
$$

Theorem 25 (⋆). E-SUBMODELS(AR) *is* DelNP-*complete.*

Proof. The upper bound follows from Theorem 16 again. For the lower bound reduce from HAMPATH, using the Kripke model $\mathcal{M}_{\mathsf{AR}}(H)$ defined in Definition 24 and φ_n with

$$
\varphi_i \coloneqq \begin{cases} ((\varphi_{i-1}\,\mathsf{AR}\,z)\,\mathsf{AR}\,y)\,\mathsf{AR}\,x_i & \text{if } i > 0, \\ \top & \text{if } i = 0. \end{cases}
$$

□

Notice that the reduction merely requires AR as operator and no Boolean connectors are used.

Fragment EX, EF, EG, EU & ER. DelNP-hardness of existentially quantified operators follows immediately, when considering negation.

Corollary 26. *For* $\emptyset \neq \mathcal{O} \subseteq \{\mathsf{EX}, \mathsf{EF}, \mathsf{EG}, \mathsf{EU}, \mathsf{ER}\}$ *we have that* E-SUBMODELS$(\mathcal{O})$ *is* NP*-complete.*

Proof. Follows directly form the duality between the existential and universal path quantifiers (see Observation 5) and our results for the universally quantified cases. □

Notice that for the fragments EX, EU and ER negation suffices as the only Boolean connector to archive intractability. In contrast the fragments EF and EG require all Boolean connectors. We summarise all results in one statement.

Theorem 27. *Let* $\emptyset \neq \mathcal{O} \subseteq \mathcal{ALL}$ *be a set of* CTL *operators. Then* E-SUBMODELS$(\mathcal{O})$ *is* DelNP*-complete.*

The silver lining

In this section, we strive for tractability results. For this we restrict also the allowed Boolean connectors and accordingly require to extend the problem notion a bit as follows. For instance, we will write EXTSUBMODEL$(\mathsf{EX}, \mathsf{ER}, \mathsf{EU})$ whenever we restrict the formulas to *only* the operators $\mathsf{EX}, \mathsf{ER}, \mathsf{EU}$ without *any* Boolean connectors.

Fragment EX, ER, EU & Conjunction, Disjunction. The first tractability result we present is a restriction to formulas only containing existentially quantified CTL operators and no negation. That is, we show DelP membership of E-SUBMODELS$(\mathsf{EX}, \mathsf{ER}, \mathsf{EU}, \wedge, \vee)$. Recall that we have $\mathsf{EF}\,\varphi = \top\,\mathsf{EU}\,\varphi$ and $\mathsf{EG}\,\varphi = \varphi\,\mathsf{ER}\,\bot$.

The following Lemma 28 gives a straightforward way to decide EXTSUBMODEL$(\mathsf{EX}, \mathsf{ER}, \mathsf{EU}, \wedge, \vee)$, by only having to consider the model and partial solution.

Lemma 28. *Let* $\mathcal{M}' \subseteq \mathcal{M}$ *be a submodel. If* $\mathcal{M} \not\models \varphi$*, for any* $\{\mathsf{EX}, \mathsf{EU}, \mathsf{ER}, \wedge, \vee\}$*-formula* φ*, then* $\mathcal{M}' \not\models \varphi$*.*

Proof. To prove this lemma consider its contraposition, i.e., $\mathcal{M}' \models \varphi$ implies $\mathcal{M} \models \varphi$. Note that the set of paths that satisfy φ in $\mathcal{M}'$ also exist in $\mathcal{M}$. Since φ does not contain negation, the same set of paths must satisfy φ in $\mathcal{M}$. □

Theorem 29. E-SUBMODELS$(\mathsf{EX}, \mathsf{ER}, \mathsf{EU}, \wedge, \vee) \in$ DelP

Proof. We describe a deterministic polynomial time algorithm for EXTSUBMODEL$(\mathsf{EX}, \mathsf{ER}, \mathsf{EU}, \wedge, \vee)$. By Lem. 28, if, for a partial solution, we have that $\mathcal{M} - D \not\models \varphi$, then it cannot be extended to an actual solution. Conversely, if $\mathcal{M} - D \models \varphi$ is true, then the empty extension is sufficient. Thus, any polynomial time model checking algorithm on an instance $\langle \mathcal{M} - D, \varphi \rangle$ can be used to decide EXTSUBMODEL$(\mathsf{EX}, \mathsf{ER}, \mathsf{EU}, \wedge, \vee)$. □

Fragment AF & AG. We adapted a result presented by Krebs et al. (Krebs, Meier, and Mundhenk 2019, Lemma 10), showing that every $\{\mathsf{AF}, \mathsf{AG}\}$-formula can be reduced to contain at most two temporal operators.

Lemma 30 ($\star$)**.** *For any formula* φ *we have that (1.)* $\mathsf{AF}\,\mathsf{AF}\,\varphi \equiv \mathsf{AF}\,\varphi$ *(2.)* $\mathsf{AG}\,\mathsf{AG}\,\varphi \equiv \mathsf{AG}\,\varphi$ *(3.)* $\mathsf{AG}\,\mathsf{AF}\,\mathsf{AG}\,\varphi \equiv \mathsf{AF}\,\mathsf{AG}\,\varphi$ *(4.)* $\mathsf{AF}\,\mathsf{AG}\,\mathsf{AF}\,\varphi \equiv \mathsf{AG}\,\mathsf{AF}\,\varphi$

Theorem 31 ($\star$)**.** E-SUBMODELS$(\mathsf{AF}, \mathsf{AG})$ *is in* DelP.

Proof. The following deterministic polynomial time algorithm decides EXTSUBMODEL$(\mathsf{AF}, \mathsf{AG})$.

The input is $\langle \mathcal{M}, \varphi, D \rangle$, where $\mathcal{M} = (W, R, \eta, r)$ is a Kripke model, φ is a $\{\mathsf{AF}, \mathsf{AG}\}$-formula, and D is a set of deletions. Let $\mathcal{M}' = (W', R', \eta, r) \coloneqq \mathcal{M} - D$ be current submodel and φ' be the shortened formula obtained from φ using Lemma 30. Notice that φ' can only have one of four forms.

Now, the algorithm has the following behaviour, depending on φ', where x is in PROP:

- $\varphi' = \mathsf{AF}\,x$: if $\mathcal{M}' \models \mathsf{EF}\,x$ accept, else reject.
- $\varphi' = \mathsf{AG}\,x$: if $\mathcal{M}' \models \mathsf{EG}\,x$ accept, else reject.
- $\varphi' = \mathsf{AF}\,\mathsf{AG}\,x$: if $\mathcal{M}' \models \mathsf{EF}\,\mathsf{EG}\,x$ accept, else reject.
- $\varphi' = \mathsf{AG}\,\mathsf{AF}\,x$: let $\hat{\mathcal{M}} = (W', R', \hat{\eta}, r)$ be the submodel $\mathcal{M}'$ but with a new labeling function $\hat{\eta}$ defined as $\hat{\eta}(w') \coloneqq \{x_{w'}\}$ for all $w' \in W'$ with $x \in \eta(w')$.
  Accept if $\hat{\mathcal{M}} \models \mathsf{EF}(x_{w'} \wedge \mathsf{EX}\,\mathsf{EF}\,x_{w'})$ for some $w' \in W'$, else reject. □

Conclusion and outlook

In this paper, we have presented a complete study of the submodel enumeration problem for the temporal logic CTL with respect to restrictions on the allowed CTL operators. We have examined all CTL operator fragments and show DelNP-completeness for every possible fragment in the presence of all Boolean connectors. This paints a completely negative picture and precludes using the debugging approach as motivated in this setting. As a silver lining on the horizon, we presented fragments obtained by constraints on Boolean functions, allowing for fast DelP algorithms that could be used for bugfix recommendations. We are currently planning to extend this approach to a complete picture for all Boolean fragments and combinations with CTL operator fragments. In particular, this leads to a very large number of possible fragments: as a rough estimate, one has to consider seven Boolean fragments, which, combined with ten CTL operators, lead to an astonishing number of $7 \cdot 2^{10} = 7168$ cases. As future work, it would be worthwhile to apply the framework of parameterised complexity (Downey and Fellows 1999) aiming at more efficient subcases. Another pressing issue is to investigate the motivated debugging approach using enumeration algorithms in a feasibility study. Furthermore, submodel enumeration is just one of many possible enumeration problems for CTL. Other variants worth investigating in this context include (minimal) modifications to η instead of, or in addition to, frame modifications.

Acknowledgments

The authors thank the anonymous reviewers for their valuable feedback and appreciate funding by the German Research Foundation (DFG) under the project id ME4279/3-1.

References

Alviano, M.; and Dodaro, C. 2016. Answer Set Enumeration via Assumption Literals. In *AI*IA*, volume 10037 of *Lecture Notes in Computer Science*, 149–163. Springer.

Barringer, H.; Fisher, M.; Gabbay, D.; and Gough, G., eds. 2000. *Advances in Temporal Logic*. Applied Logic Series. Springer Dordrecht.

Bellini, P.; Mattonlini, R.; and Nesi, P. 2000. Temporal logics for real-time system specification. *ACM Comput. Surv.*, 32(1): 12–42.

Bérard, B.; Bidoit, M.; Finkel, A.; Laroussinie, F.; Petit, A.; Petrucci, L.; Schnoebelen, P.; and McKenzie, P. 2001. *Systems and Software Verification, Model-Checking Techniques and Tools*. Springer.

Biere, A.; Cimatti, A.; Clarke, E. M.; Strichman, O.; and Zhu, Y. 2003. Bounded model checking. *Adv. Comput.*, 58: 117–148.

Blom, J. 1996. Temporal logics and real time expert systems. *Computer Methods and Programs in Biomedicine*, 51(1): 35–49. Improving Control of Patient Status in Critical Care: The IMPROVE Project.

Clarke, E. M.; Grumberg, O.; Kroening, D.; Peled, D. A.; and Veith, H. 2018. *Model checking, 2nd Edition*. MIT Press. ISBN 978-0-262-03883-6.

Cook, S. A. 1971. The Complexity of Theorem-Proving Procedures. In *STOC*, 151–158. ACM.

Creignou, N.; Kröll, M.; Pichler, R.; Skritek, S.; and Vollmer, H. 2019. A complexity theory for hard enumeration problems. *Discret. Appl. Math.*, 268: 191–209.

da Silva, R. R.; Kurtz, V.; and Lin, H. 2021. Symbolic control of hybrid systems from signal temporal logic specifications. *Guidance, Navigation and Control*, 1(02): 2150008.

Downey, R. G.; and Fellows, M. R. 1999. *Parameterized Complexity*. Monographs in Computer Science. Springer.

Finger, M.; Fisher, M.; and Owens, R. 1993. Metatem at work: Modelling reactive systems using executable temporal logic. In *IEA/AIE-93*. Gordon and Breach Publishers Edinburgh, UK.

Fomin, F. V.; and Kratsch, D. 2010. *Exact Exponential Algorithms*. Springer Berlin, Heidelberg.

Friedrich, G.; and Zanker, M. 2011. A Taxonomy for Generating Explanations in Recommender Systems. *AI Mag.*, 32(3): 90–98.

Fröhlich, N.; and Meier, A. 2022. Submodel Enumeration of Kripke Structures in Modal Logic. In *AiML*, 391–406. College Publications.

Fröhlich, N.; and Meier, A. 2023. Submodel Enumeration for CTL Is Hard. arXiv:2312.09868.

Gupta, A.; Yang, Z.; Ashar, P.; and Gupta, A. 2000. SAT-Based Image Computation with Application in Reachability Analysis. In *FMCAD*, volume 1954 of *Lecture Notes in Computer Science*, 354–371. Springer.

Johnson, D. S.; Yannakakis, M.; and Papadimitriou, C. H. 1988. On generating all maximal independent sets. *Information Processing Letters*, 27(3): 119–123.

Karp, R. M. 1972. Reducibility Among Combinatorial Problems. In *Complexity of Computer Computations*, The IBM Research Symposia Series, 85–103. Plenum Press, New York.

Konur, S. 2013. A survey on temporal logics for specifying and verifying real-time systems. *Frontiers Comput. Sci.*, 7(3): 370–403.

Krebs, A.; Meier, A.; and Mundhenk, M. 2019. The model checking fingerprints of CTL operators. *Acta Informatica*, 56(6): 487–519.

Kripke, S. 1963. Semantical Considerations on Modal Logic. *Acta Philosophica Fennica*, 16: 83–94.

Lauri, J.; and Dutta, S. 2019. Fine-Grained Search Space Classification for Hard Enumeration Variants of Subset Problems. In *AAAI*, 2314–2321. AAAI Press.

Levin, L. 1973. Universal sorting problems. *Problems of Information Transmission*, 9: 265–266.

Papadimitriou, C. H. 2007. *Computational complexity*. Academic Internet Publ.

Schnoebelen, P. 2002. The Complexity of Temporal Logic Model Checking. In *Advances in Modal Logic*, 393–436. King's College Publications.

Strozecki, Y. 2019. Enumeration Complexity. *Bull. EATCS*, 129.

Sullivan, A.; Marinov, D.; and Khurshid, S. 2019. Solution Enumeration Abstraction: A Modeling Idiom to Enhance a Lightweight Formal Method. In *ICFEM*, volume 11852 of *Lecture Notes in Computer Science*, 336–352. Springer.

Vakili, A.; and Day, N. A. 2014. Reducing CTL-live model checking to first-order logic validity checking. In *Formal Methods in Computer-Aided Design, FMCAD 2014, Lausanne, Switzerland, October 21-24, 2014*, 215–218. IEEE.