Gottfried Wilhelm Leibniz Universität Hannover
Institut für Theoretische Informatik

# Enumeration in Temporal Logic

Master Thesis

**Nicolas Frederik Hamlet Fröhlich**
Matriculation Number: 10022886

May 2, 2023

First Examiner: PD Dr. Arne Meier
Second Examiner: Prof. Dr. Heribert Vollmer
Advisor: PD Dr. Arne Meier

# Erklärung der Selbstständigkeit

Hiermit versichere ich, die vorliegende Masterarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet zu haben. Die Arbeit hat in gleicher oder Ähnlicher Form noch keinem anderen Prüfungsamt vorgelegen.

Hannover, den 1. Mai 2023

Nicolas Fröhlich

# Contents

# 1 Introduction

Temporal logics extend classical logic, by introducing reasoning over time and modality [16, 17]. A well-known application is the specification and verification of systems and software through transition models. Other uses include automated reasoning and artificial intelligence as well as applications in physics, philosophy and linguistics. The main temporal logics are Computational Tree Logic $\mathsf{CTL}$, Linear Time Logic $\mathsf{LTL}$, and the superset of both $\mathsf{CTL}^*$. All three introduce two path quantifiers $\mathsf{A}$ and $\mathsf{E}$, together with the temporal operators ne$\mathsf{X}$t, $\mathsf{U}$ntil, $\mathsf{F}$uture, $\mathsf{G}$lobally, and $\mathsf{R}$elease.

In $\mathsf{CTL}^*$ any combination of path quantifiers and temporal operators is allowed, making $\mathsf{CTL}^*$ the most general temporal logic. In contrast, $\mathsf{LTL}$ formulas only quantify over paths, i.e. they always start with $\mathsf{E}$, followed only by propositional and temporal operators. Finally, in $\mathsf{CTL}$ all path quantifiers must be followed by a temporal operator, resulting in ten different $\mathsf{CTL}$ operators.

The most common variant of a transition system used for temporal logic is the *Kripke model* [14]. Informally, a Kripke model consists of worlds $W$, a *total* binary relation $R$ connecting one world to another, and a function $\eta$ labelling each world with a set of propositions that hold in that world.

To familiarise the reader with Kripke models and temporal logic formulas, in particular $\mathsf{CTL}$ formulas, we give a short example. Figure 1.1 depicts a simple model for a traffic light, which is formally defined as

$$\mathcal{M} = (\{r, w_1, w_2\}, \{(r, w_2), (r, w_1), (w_1, w_2), (w_2, r)\}, \eta)$$

with $\eta(r) = \{green\}, \eta(w_1) = \{yellow\}$ and $\eta(w_3) = \{red\}$. To check that our model meets the requirements of a traffic light we can describe its behaviour using $\mathsf{CTL}$ formulas.

For example: "at least one light should be on at all times and after *green* always comes *red* at some point". A $\mathsf{CTL}$-formula describing this property would be

$$\varphi := \mathsf{AG}(green \lor yellow \lor red) \land \mathsf{AG}(green \to \mathsf{AF}\, red).$$

This leads directly to the first of two important decision problems in temporal logic, *model checking* [3, 2]. In model checking, you are given a Kripke model and a temporal formula, and

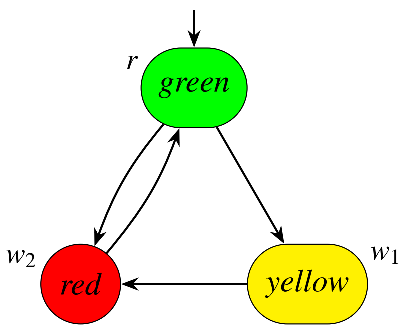


Figure 1.1: Kripke model $\mathcal{M}$ of a three signal traffic light.

have to decide whether the formula is true in the model or not. The model checking problem for temporal logics has been thoroughly studied, with results ranging from P-completeness for CTL to PSPACE-completeness for LTL and CTL$^*$ [18].

Satisfiability is the other important decision problem in temporal logic, asking whether for a given formula there exists a Kripke model such that the formula is true in that model. For CTL satisfiability is EXPTIME-complete, for LTL it is PSPACE-complete and for CTL$^*$ it is 2-EXPTIME-complete [10, 19, 20].

While decision problems ask whether solutions exist, a more natural approach may be to output the solutions themselves. Such problems are called *enumeration problems* and are concerned with finding and returning *all* solutions to a problem. Analysing the complexity of an enumeration algorithm often lead to exponential runtime, simply because the number of solutions is exponential. Therefore, one studies the time that elapses between the output of two solutions, the *delay* of an enumeration algorithm. Enumeration problems with an algorithm with polynomial delay are considered as the class of efficient enumeration problems.

Creignou et al. introduced a framework for enumeration problems beyond polynomial delay [6]. They present a hierarchy for enumeration problems analogous to the polynomial hierarchy and a notion of hardness using reduction. Here, we will use the enumeration complexity classes DelP and DelNP, where DelP is the aforementioned class of efficient enumeration problems.

A natural enumeration problem for CTL is *submodel enumeration*, i.e. the problem of finding and outputting all submodels of a Kripke model that satisfy a CTL formula. While submodel enumeration has been studied for modal logic, a weaker extension of classical logic with only modality [11], the goal of this thesis is to further investigate its complexity with respect to fragments of the temporal logic CTL. We will immediately see that this problem is hard for arbitrary CTL formulas, which motivates restrictions on the allowed Boolean and CTL operators.

For a better understanding, let us go back to the traffic light example from before and consider the following scenario. We want to simplify our notion of a traffic light to just two signals *red* and *green*. Since we already have a model of a traffic light with three signals

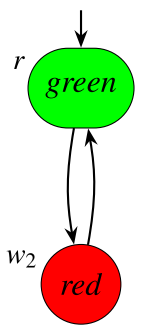


Figure 1.2: Traffic light submodel of Figure 1.1 with only two signals.

we do not want to start from scratch. Instead, we define a new formula that describes the properties of a traffic light with two signals, e.g.

$$\varphi_{RG} \coloneqq \mathsf{AG}(green \rightarrow \mathsf{AX}\, red \wedge red \rightarrow \mathsf{AX}\, green).$$

Using submodel enumeration, we get all submodels that satisfy $\varphi_{RG}$ allowing us to choose the one that is most useful to us. Here, the only suitable submodel that would be enumerated can be seen in Figure 1.2.

In Chapter 2 we first introduce the necessary definitions of CTL and enumeration complexity, as well as some results of hard enumeration. We then investigate the complexity of submodel enumeration in Chapter 3. First we consider CTL with all operators, which we will prove to be DelNP-complete, followed by restricting access to more and more operators. Finally, we summarise our results and give an outlook on possible future research in Chapter 4.

# 2 Preliminaries

In this chapter, we will define the temporal logic CTL and submodels, the notion of clones and fragments, enumeration complexity and some results of hard enumeration used in this thesis.

## 2.1 Computational Tree Logic

We follow the notation of Clarke et al. [3].

Let PROP be an infinite, countable set of propositions. The set of well-formed CTL formulas is

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

with $p \in \text{PROP}, \mathcal{P} \in \{\mathsf{E}, \mathsf{A}\}, \mathcal{T} \in \{\mathsf{X}, \mathsf{F}, \mathsf{G}\}, \mathcal{T}' \in \{\mathsf{U}, \mathsf{R}\}$. 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.

Next, we turn towards the structures used in Temporal Logic called *Kripke model*. Usually a Kripke model is triple $(W, R, \eta)$, where $W$ is a set of worlds, $R \subseteq W \times W$ is a total transition relation and $\eta$ is an assignment function. Here we add a fourth element $r \in W$ called *root*.

**Definition 1.** A *rooted Kripke Model* is a tuple $\mathcal{M} = (W, R, \eta, r)$ where

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

An example of a rooted Kripke model is depicted in Figure 2.1.

CTL formulas make statements about infinite paths. As such we need to define paths of rooted Kripke Models.

**Definition 2.** Let $\mathcal{M} = (W, R, \eta, r)$ be a rooted Kripke model. A *path* $\pi$ in $\mathcal{M}$ is an infinite sequence of worlds $w_1, w_2, \cdots$ such that $(w_i, w_{i+1}) \in R$ for all $i \geq 1$. We write $\pi[i]$ to denote

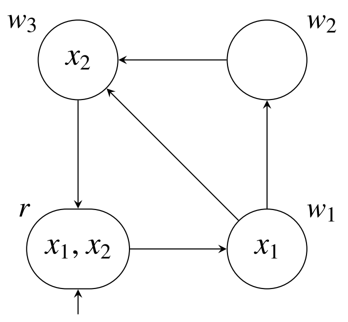


Figure 2.1: Example Kripke model.

the $i$th world on the path $\pi$. For a world $w \in W$ we define $\Pi(w) := \{\pi \mid \pi[1] = w\}$ as the set of all infinite paths of $\mathcal{M}$ starting with $w$.

We can now define the semantics of CTL.

**Definition 3.** Let $\mathcal{M}$ be a rooted Kripke model and $\varphi, \psi$ 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 $\mathcal{M}, w \models \psi$, |
| $\mathcal{M}, w \models \varphi \oplus \psi$ | iff ($\mathcal{M}, w \models \varphi$ and $\mathcal{M}, w \not\models \psi$) or ($\mathcal{M}, w \not\models \varphi$ and $\mathcal{M}, w \models \psi$), |
| $\mathcal{M}, w \models \mathsf{EX}\,\varphi$ | 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 $\forall i < k : \mathcal{M}, \pi[i] \models \varphi$, |
| $\mathcal{M}, w \models \varphi\,\mathsf{AU}\,\psi$ | iff $\forall \pi \in \Pi(w)\ \exists k \geq 1 : \mathcal{M}, \pi[k] \models \psi$ and $\forall i < k : \mathcal{M}, \pi[i] \models \varphi$, |
| $\mathcal{M}, w \models \varphi\,\mathsf{ER}\,\psi$ | iff $\exists \pi \in \Pi(w)\ \forall k \geq 1 : \mathcal{M}, \pi[k] \models \psi$ or $\exists i < k : \mathcal{M}, \pi[i] \models \varphi$, |
| $\mathcal{M}, w \models \varphi\,\mathsf{AR}\,\psi$ | iff $\forall \pi \in \Pi(w)\ \forall k \geq 1 : \mathcal{M}, \pi[k] \models \psi$ or $\exists i < k : \mathcal{M}, \pi[i] \models \varphi$. |

Additionally, take $\bot := \neg\top$ as constant false. We also omit the root in $\mathcal{M}, r \models \varphi$ and just write $\mathcal{M} \models \varphi$ instead. A formula $\varphi$ is then said to be *satisfied by model* $\mathcal{M}$, if $\mathcal{M} \models \varphi$ holds.

Before continuing let us make an observation regarding the semantics of CTL.

*Observation* 4. The following equivalences between $\mathsf{CTL}$ operators hold:

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

### 2.1.1 Submodels

Next, we introduce the notion of submodels. Given two Kripke models $\mathcal{M} = (W, R, \eta, r)$ and $\mathcal{M}' = (W', R', \eta, r)$. If $W' \subseteq W$ and $R' \subseteq R$ we call $\mathcal{M}'$ a *submodel* of $\mathcal{M}$, denoted by $\mathcal{M}' \subseteq \mathcal{M}$. Notice that a tuple $(W', R', \eta, r)$ with $W' \subseteq W$ and $R' \subseteq R$ is not necessarily a submodel, as for example $R'$ could be a not total relation.

We want to further narrow down the definition of a submodel, by introducing connected submodels.

**Definition 5.** Let $\mathcal{M} = (W, R, \eta, r)$ be a Kripke model. $\mathcal{M}' = (W', R', \eta, r)$ is a *connected submodel* of $\mathcal{M}$, denoted by $\mathcal{M}' \subseteq_c \mathcal{M}$, if

1. $W' \subseteq W$ and $W \neq \emptyset$,
2. $R' \subseteq R$ and $R' \subseteq W' \times W'$ is a total relation and
3. for all $w \in W'$ there exists a path $\pi \in \Pi(r)$ and $i \geq 1$ with $\pi[i] = w$.

The first two points ensure that $\mathcal{M}'$ is a Kripke model. The third point makes sure that all worlds in $\mathcal{M}'$ lie on a path starting at the root. This reduces redundancy in the enumeration of submodels. Worlds that violate this point cannot have influence on the satisfiability of CTL formulas. Thus an enumeration algorithm printing connected submodels can trivially be extended to include non-connected submodels.

Since we will only consider connected submodels in this thesis, we will simply write submodels and use $\subseteq$ instead of $\subseteq_c$. Additionally we want to introduce an alternative notation for submodels $\mathcal{M}' = \mathcal{M} - D$, with $D = (D_W, D_R)$ a tuple consisting of a set of worlds and a set of relations and where $W' = W \setminus D_W$ and $R' = R \setminus D_R$, for $\mathcal{M} = (W, R, \eta, r)$ and $\mathcal{M}' = (W', R', \eta, r)$.

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

Finally, we define the notion of an order over the relations of a Kripke model.

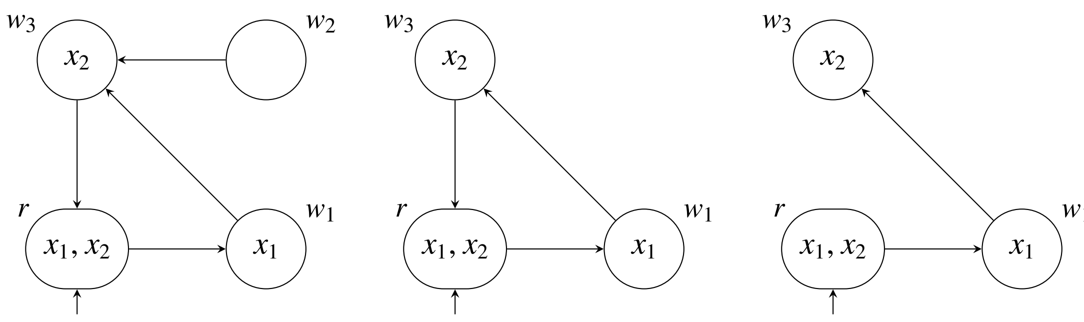


Figure 2.2: Three potential submodels of the Kripke model in Figure 2.1. The left structure is a submodel but not a connected submodel. The middle structure is a connected submodel. And the right structure is neither a connected submodel nor a submodel.

**Definition 6** ($R$-Order)**.** Let $\mathcal{M} = (W, R, \eta, r)$ be a Kripke model. We write $\prec_R$ to denote an arbitrary strict total order over the relations $R$.

Additionally we say that $\min(\mathcal{M}) := \min(R) := e \in R$ such that for all $e_0 \in R$ with $e_0 \neq e$ we have $e \prec_R e_0$.

The specifics of $\prec_R$ are not of relevance here. All that is required of $\prec_R$ is that it allows us to describe a unique sequence of relations in a model.

**Example 7.** Let us look at the Kripke model in Figure 2.1 again. Figure 2.2 shows three possible submodels. While the first model has a total relation and is therefore a submodel by the original definition, there is no path to world $w_2$. Therefore the model is not a connected submodel. The middle model fulfills all the requirements of a connected submodel. Although the last model is connected, the world $w_3$ has no outgoing relation, as such the model is not total and therefore not a submodel.

## 2.1.2 Post's Lattice in CTL

Let *id* be the identity function, i.e. $id(x) := x$. If $B$ is a set of Boolean functions, then $[B]$ is called a *clone*, if it contains all projections (i.e. all Boolean functions $I_k^n(a_1, \dots, a_n) = a_k$ for all $a_1, \dots, a_n \in \{0, 1\}$) and is the smallest set, which is derived by arbitrary composition of functions of $B \cup \{id\}$, such that $[B] = B$ holds. Any finite $B_0 \subseteq B$ with $[B_0] = B$ is called a *base* (or *basis*) of $B$. For a more in depth introduction into clones we refer the reader to [1].

While there is usually an infinite set of Boolean clones, in the context of this thesis we only need to consider seven. This is due to the fact that we can always simulate the constants $\top$ and $\bot$ by introducing new proportions that hold on either all worlds or none.

Figure 2.3 depicts the relevant clones described by their standard bases. Notice that $\{\land, \oplus\}$ is sufficient to express all Boolean functions, because of the "free" constants and the equivalences

$$\neg\varphi \equiv \varphi \oplus \top \text{ and } \varphi \lor \psi \equiv ((\varphi \oplus \top) \land (\psi \oplus \top)) \oplus \top.$$

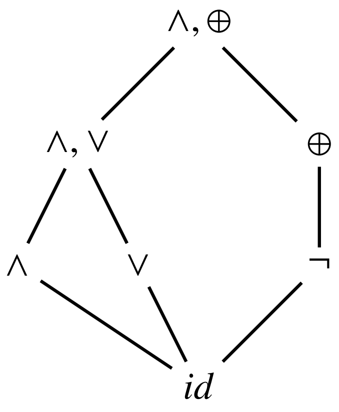


Figure 2.3: The relevant Boolean clones in this thesis.

For the CTL operators we also have equivalences as shown in Observation 4. These allow us to construct a similar lattice for CTL operators when there are no restrictions on the Boolean operators, as depicted in Figure 2.4. Here we can also see, that $\{\mathsf{AX}, \mathsf{AF}, \mathsf{AR}, \wedge, \oplus\}$-formulas have enough expressivity to formulate any CTL formula. Unfortunately, the lattices for temporal operators and clones with limited access to Boolean operators are impractical to visualise due to their sheer size.

## 2.2 Enumeration Complexity

In this section we will introduce Enumeration Complexity, unlike decision problems, which ask for the existence of solutions to a given instance, enumeration problems aim to output *all* solutions of an instance. The set of solutions is often much larger than the the size of the input instance, making the overall runtime of an enumeration algorithm overly dependent on the number of solutions. To avoid the problem of possible exponential total runtime, the elapsed time between outputting two solutions, called the *delay* of an enumeration algorithm, is commonly measured.

The Turing machine, as the standard machine model used in complexity theory, also proves to be problematic for enumeration. Its linear nature in accessing data prevents a polynomial delay when traversing exponentially large data sets, even if the actual data read is small. Therefore, one commonly uses random access machines (RAMs) as the machine model in Enumeration Complexity. RAMs, as their name suggests, allow direct access to bits of their data.

**Definition 8.** Let $\Sigma$ be a finite alphabet. An *enumeration problem* is a tuple $\mathcal{E} = (I, \mathrm{Sol})$, where

- $I \subseteq \Sigma^*$ is the set of *instances*, and
- $\mathrm{Sol}\colon I \to \mathcal{P}(\Sigma^*)$ is a function that maps to each instance $x \in I$ a set of *solutions (of $x$)*.

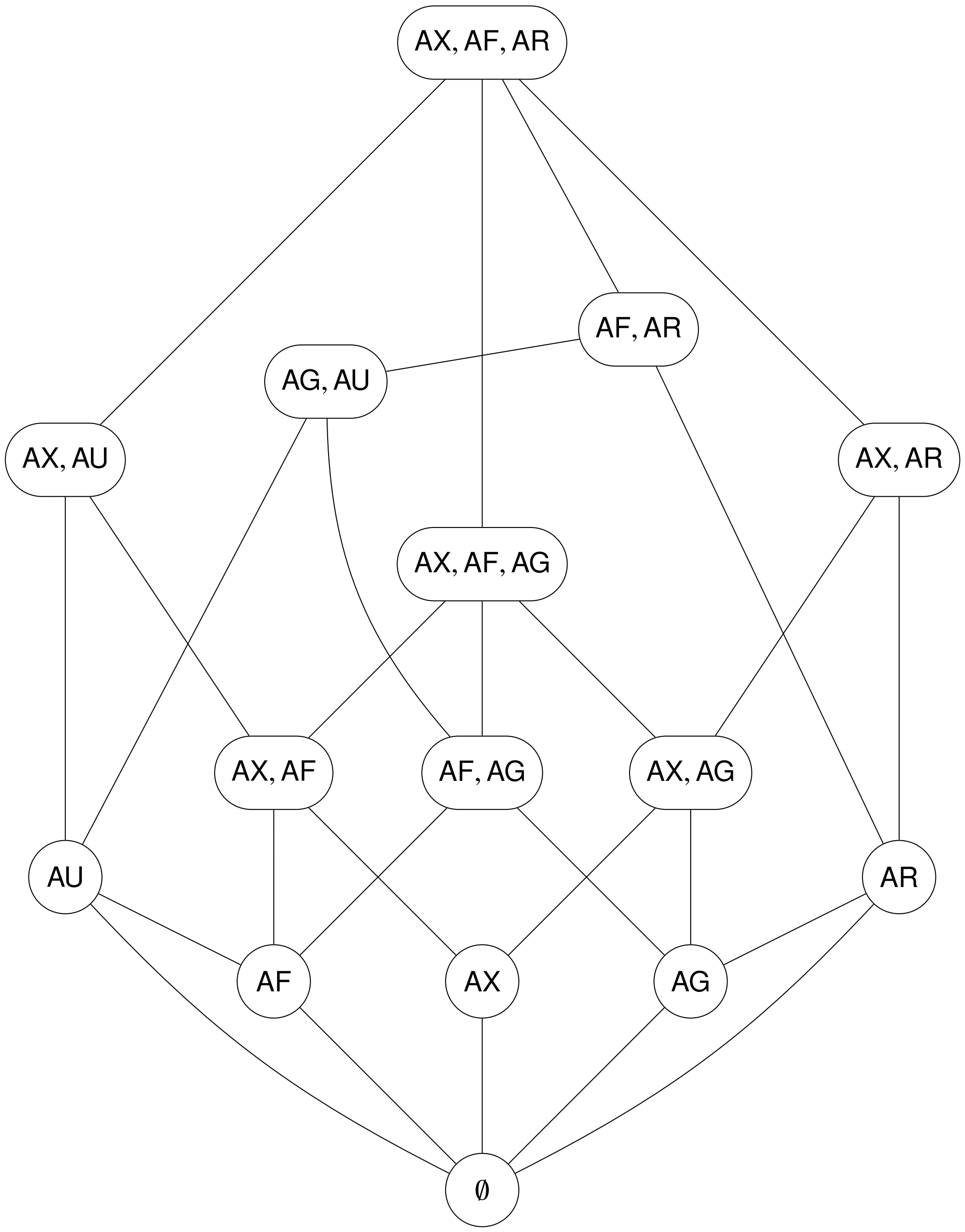


Figure 2.4: The lattice induced by all CTL operators with no restrictions on the Boolean operators. Each node is labeled with a minimal set of operators.

**Definition 9.** Let $\mathcal{E} = (I, \mathrm{Sol})$ be an enumeration problem. An algorithm $\mathcal{A}$ is called an *enumeration algorithm* for $\mathcal{E}$, if for every instance $x \in I$ the algorithm $\mathcal{A}$ obeys the following two properties, where $\mathcal{A}(x)$ denotes the *computation of $\mathcal{A}$ on input $x$*:

- $\mathcal{A}(x)$ terminates after a finite sequence of steps.
- $\mathcal{A}(x)$ prints exactly $\mathrm{Sol}(x)$ without duplicates.

Let us now formally define the previously mentioned delay of an enumeration algorithm.

**Definition 10.** Let $\mathcal{E} = (I, \mathrm{Sol})$ be an enumeration problem, $\mathcal{A}$ 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 now define

- the *ith delay* of $\mathcal{A}(x)$ as the elapsed time between the output of the $i$th and $(i+1)$st solution of $\mathrm{Sol}(x)$,
- the 0*th delay* as the *precomputation time*, i.e, the elapsed time before the first output of $\mathcal{A}(x)$, and
- the *nth 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 some function $f\colon \mathbb{N} \to \mathbb{N}$, if for all $x \in I$ and all $0 \leq i \leq n$ the $i$th delay of $\mathcal{A}(x)$ is in $O(f(|x|))$.

If $f$ is a polynomial, then this leads directly to the most common enumeration complexity class, i.e. the class of problems that can be enumerated by an algorithm with polynomial delay, called DelP.

### 2.2.1 Hard enumeration

In this subsection we will introduce the framework of hard enumeration by Creignou et al. [6]. Hard enumeration presents tools to analyse enumeration problems beyond polynomial delay by introducing a hierarchy of complexity classes similar to the polynomial-time hierarchy and a notion of reductions for enumeration problems.

We begin by defining two decision problems that naturally arise in the context of enumeration.

**Definition 11.** Let $\mathcal{E} = (I, \mathrm{Sol})$ be an enumeration problem over the alphabet $\Sigma$. The decision problem asking for the existence of solutions, i.e. the set $\mathrm{Sol}(x)$ is not empty for a given instance $x \in I$, is defined as follows:

| **Problem:** | EXIST_$\mathcal{E}$ |
|---:|:---|
| **Input:** | Instance $x$ |
| **Question:** | Is $\mathrm{Sol}(x) \neq \emptyset$? |

The decision problem concerned with finding new solutions, i.e. 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 $y'$, is defined as:

| **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 underling machine model, by introducing decision oracles. A decision oracle can be seen as a subroutine solving a specific decision problem. When analysing the runtime, or in this case the delay, of an algorithm calls to its oracle always count as a single step, regardless of the time the oracles needs. Our machines can now write into special registers and the oracle will consider these as well as all consecutive non-empty registers as its input. A request to the oracle then occurs if the machine enters a special question state and will go into either a positive state if the oracle answers "yes" or a negative state if the oracle answers "no".

We can now formally define enumeration complexity classes with oracles.

**Definition 12** (Enumeration Complexity Classes)**.** Let $\mathcal{E}$ be an enumeration problem, and $C$ a decision complexity class. Then we say that $\mathcal{E} \in \mathsf{Del}C$ if there is a RAM $M$ with oracle $L$ in $C$ and a polynomial $p$, such that for any instance $x$, $M$ enumerates $\mathrm{Sol}(x)$ with delay $p(|x|)$. Moreover, the size of every oracle call is bound by $p(|x|)$.

This definition directly entails the aforementioned enumeration complexity classes $\mathsf{DelP}$. Another enumeration complexity class we want to highlight here is $\mathsf{DelNP}$, which consists of enumeration problems that can be enumerated with polynomial delay by a RAM with an oracle in $\mathsf{NP}$. Both $\mathsf{DelP}$ and $\mathsf{DelNP}$ can be seen as the counterparts of $\mathsf{P}$ and $\mathsf{NP}$ respectively.

The following Corollary 13 as well as Corollary 17 are both simplified versions of results presented in [6]. While Creignou et al. considered the full polynomial hierarchies in their proofs, here we are only concerned with the $\mathsf{P}$ and $\mathsf{NP}$ cases.

**Corollary 13** ([6, Proposition 6])**.** *Let $\mathcal{E} = (I, \mathrm{Sol})$ be an enumeration problem and $C \in \{\mathsf{P}, \mathsf{NP}\}$. If* ExtendSol_$\mathcal{E} \in C$ *then* $\mathcal{E} \in \mathsf{Del}C$.

Corollary 13 allows membership results for enumeration problems, using the corresponding decision problem ExtendSol. This technique will prove particularly useful when showing membership in $\mathsf{DelNP}$, as constructing enumeration algorithms with oracles can be quite difficult.

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

**Definition 14.** Let $\mathcal{E}$ be an enumeration problem. 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 $A_e, O^e(0), O^e(1), \dots$ 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_$\mathcal{E}$.

**Definition 15.** Let $\mathcal{E}$ be an enumeration problem and $\pi_1, \pi_2, \dots$ be the run of an EOM_$\mathcal{E}$ and assume that the $k$th instruction is NOO, i.e. $\pi_k = \text{NOO}$. Denote with $x_i$ the word stored in $O^e(0), O^e(1), \dots$ at step $i$. Let $K = \{\pi_i \in \{\pi_1, \dots, \pi_{k-1}\} \mid \pi_i = \text{NOO and } x_i = x_k\}$. Then the *oracle output* $y_k$ *in* $\pi_k$ is defined as an arbitrary $y_k \in R(x_k)$ such that $y_k$ has not been the oracle output in any $\pi_i \in K$. If no such $y_k$ exists, then the oracle output in $\pi_k$ is undefined.

When executing NOO in step $\pi_k$, if the oracle output $y_k$ is undefined, then the register $A_e$ contains some special symbol in step $\pi_{k+1}$. Otherwise in step $\pi_{k+1}$ the register $A_e$ contains $y_k$.

**Definition 16** ($D$-reductions)**.** Let $\mathcal{E}$ and $\mathcal{E}'$ be enumeration problems. 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 enumeration problem $\mathcal{E}$. If an enumeration problem is shown to be both hard for and a member of an enumeration complexity class, we call it *complete* for that class.

**Corollary 17** ([6, Theorem 13])**.** *Let* $\mathcal{E} = (I, \text{Sol})$ *be an enumeration problem. If* Exist_$\mathcal{E}$ *is* NP*-hard, then* $\mathcal{E}$ *is* DelNP*-hard via D-reductions.*

## 2.3 Auxiliary Decision Problems

To make use of Corollary 17 we need to show the NP-hardness of Exist_$\mathcal{E}$. We will use the standard approach in complexity theory by presenting *polynomial time many-one reductions* ($\leq_m^{\mathsf{P}}$). That is, a problem $A$ is NP-hard if a problem $B$, already known to be NP-hard, can be reduced to problem $A$ in polynomial time, i.e. $B \leq_m^{\mathsf{P}} A$. Thus we now present the NP-complete (NP-hard and computable in NP) decision problems SAT and HAMPATH.

The problem SAT is concerned with the satisfiability of formulas of propositional logic. We will not fully introduce propositional logic here, instead we present the necessary aspect to work with SAT.

A *formula* $\varphi$ in propositional logic consists of Boolean connectors $\land, \lor, \neg$ and propositions $x_1, x_2, \dots, x_n$. The set of propositions of $\varphi$ is denoted by PROP. We call the total function $\mathfrak{I}\colon \text{PROP} \to \{0, 1\}$ an *assignment* of $\varphi$. The *evaluation function* $\hat{\mathfrak{I}}(\varphi)$ extends the assignment function to formulas in the following way:

| **Problem:** | SAT |
|---|---|
| **Input:** | A propositional formula $\varphi$ |
| **Question:** | Does an assignment $\mathfrak{I}$ exist such that $\mathfrak{I}(\varphi) = 1$? |

Figure 2.5: Satisfiability problem of propositional logic

| **Problem:** | HAMPATH |
|---|---|
| **Input:** | A graph $G = (V, E)$ with $s \in V$ and $t \in V$ |
| **Question:** | Does a Hamiltonian path from $s$ to $t$ exist in $G$? |

Figure 2.6: Hamiltonian path problem

1. If $\varphi$ is a proposition, then $\hat{\mathfrak{I}}(\varphi) = \mathfrak{I}(\varphi)$.
2. If $\varphi = \theta \wedge \phi$, then $\hat{\mathfrak{I}}(\varphi) = 1$, if $\hat{\mathfrak{I}}(\theta) = 1$ and $\hat{\mathfrak{I}}(\phi) = 1$, 0 otherwise.
3. If $\varphi = \theta \vee \phi$, then $\hat{\mathfrak{I}}(\varphi) = 1$, if $\hat{\mathfrak{I}}(\theta) = 1$ or $\hat{\mathfrak{I}}(\phi) = 1$, 0 otherwise.
4. If $\varphi = \neg\phi$, then $\hat{\mathfrak{I}}(\varphi) = 1$, if $\hat{\mathfrak{I}}(\phi) = 0$ and 0 otherwise.

We usually write $\mathfrak{I}$ for both, the assignment and evaluation function, and omit $\hat{\mathfrak{I}}$. If $\mathfrak{I}(\varphi) = 1$ we say that $\varphi$ is *satisfied by* $\mathfrak{I}$. A formula $\varphi$ is *satisfiable* if an assignment $\mathfrak{I}$ exists such that $\varphi$ is satisfied by $\mathfrak{I}$.

The problem SAT now consists of all satisfiable formulas. The formal definition can be seen in Figure 2.5.

**Theorem 18** ([4])**.** SAT *is* NP*-complete.*

The second decision problem we will be using in this thesis is HAMPATH. Given a graph $G = (V, E)$ containing the nodes $s$ and $t$, HAMPATH asks if there exits a Hamiltonian path from $s$ to $t$. A Hamiltonian path is defined as a path containing all nodes of the graph exactly once. Figure 2.6 formally defines this as a decision problem.

**Theorem 19** ([12])**.** HAMPATH *is* NP*-complete.*

# 3 Complexity of Submodel Enumeration

In this chapter we will formally define the problem of enumerating satisfying submodels. We then present complexity results for CTL and most of its fragments.

## 3.1 Submodel Enumeration

We begin by formally defining the enumeration problems we want to study in this thesis. That is the problem of enumerating all submodels of a given models satisfying the given CTL formula.

| | |
|---|---|
| **Problem:** | E-CTL-Submodel($T$) |
| **Input:** | A Kripke model $\mathcal{M} = (W, R, \eta, r)$ and a CTL formula $\varphi$ with operators in $T \subseteq \{\mathsf{EX}, \mathsf{AX}, \mathsf{EG}, \mathsf{AG}, \mathsf{EF}, \mathsf{AF}, \mathsf{EU}, \mathsf{AU}, \mathsf{ER}, \mathsf{AR}, \neg, \vee, \wedge, \oplus\}$. |
| **Output:** | All submodels $\mathcal{M}' \subseteq \mathcal{M}$ that satisfy $\varphi$, i.e. $\mathcal{M}' \models \varphi$. |

We will omit the set notation in favour of convenience and write E-CTL-Submodel(AF, $\wedge$) instead of E-CTL-Submodel({AF, $\wedge$}) for example. Also we write E-CTL-Submodel to refer to the problem that can express any CTL formula, i.e. E-CTL-Submodel(EX, EG, EU, $\wedge, \oplus$).

Next we define the decision problems Exist_$\mathcal{E}$ and ExtendSol_$\mathcal{E}$ of E-CTL-Submodel($T$).

| | |
|---|---|
| **Problem:** | ExistSubmodel($T$) |
| **Input:** | A Kripke model $\mathcal{M} = (W, R, \eta, r)$ and a CTL formula $\varphi$ with operators in $T \subseteq \{\mathsf{EX}, \mathsf{AX}, \mathsf{EG}, \mathsf{AG}, \mathsf{EF}, \mathsf{AF}, \mathsf{EU}, \mathsf{AU}, \mathsf{ER}, \mathsf{AR}, \neg, \vee, \wedge, \oplus\}$. |
| **Question:** | Does a submodels $\mathcal{M}' \subseteq \mathcal{M}$ with $\mathcal{M}' \models \varphi$ exist? |

| | |
|---|---|
| **Problem:** | ExtendSubmodel($T$) |
| **Input:** | A Kripke model $\mathcal{M} = (W, R, \eta, r)$, a CTL formula $\varphi$ with operators in $T \subseteq \{\mathsf{EX}, \mathsf{AX}, \mathsf{EG}, \mathsf{AG}, \mathsf{EF}, \mathsf{AF}, \mathsf{EU}, \mathsf{AU}, \mathsf{ER}, \mathsf{AR}, \neg, \vee, \wedge, \oplus\}$ and a set of deletions $D$. |
| **Question:** | Is there a $D'$ such that the submodel $\mathcal{M}' := \mathcal{M} - [D \cup D']$ satisfies $\varphi$, i.e. $\mathcal{M}' \models \varphi$? |

We will again omit ($T$) when the fragment in question is clear from the context.

## 3.2 Results for formulas with all operators

Let us now turn to our first complexity results. Enumerating satisfying submodels without restrictions on the operators is intractable. We prove this, by showing NP-hardness of ExistSubmodel. The core idea of the proof is to establish a connection between submodels and assignments of a propositional formula. The goal is to show, that finding a submodel that satisfies the CTL formula is equivalent to solving SAT.

**Theorem 20.** E-CTL-Submodel *is* DelNP*-hard.*

*Proof.* It follows from Corollary 17 that the showing NP-hardness of ExistSubmodel is sufficient to prove this theorem.

Let $\varphi$ be a propositional formula. Further let $\mathrm{PROP}(\varphi) = \{x_1, x_2, \dots, x_n\}$ be the finite set of propositions of $\varphi$. We define $\varphi_{CTL}$ by replacing all occurrences of $x$ in $\varphi$ with $\mathsf{EX}\, x$ for $x \in \mathrm{PROP}(\varphi)$. We then construct a Kripke model $\mathcal{M} = (W, R, \eta, r)$ as follows (see Figure 3.1 for a visual representation).

$$\begin{aligned} W &:= \{r, w_{x_1}, w_{x_2}, \dots, w_{x_n}\} \\ R &:= \{(r, w_{x_i}), (w_{x_i}, w_{x_i}) \mid x_i \in \mathrm{PROP}(\varphi)\} \\ \eta(w_{x_i}) &:= \{x_i\} \text{ for all } x_i \in \mathrm{PROP}(\varphi) \end{aligned}$$

We show that $\langle\varphi\rangle \mapsto \langle\mathcal{M}, \varphi_{CTL}\rangle$ is a valid reduction function from SAT to ExistSubmodel.

Suppose $\langle\varphi\rangle \in$ SAT. Then there is an assignment $\mathfrak{I}$ such that $\mathfrak{I}(\varphi) = 1$. Next let $\mathcal{M}' = (W', R', \eta, r)$ be a submodel of $\mathcal{M}$ as follows:

$$\begin{aligned} W' &:= W \setminus \{w_{x_i} \mid \mathfrak{I}(x_i) = 0\} \\ R' &:= R \setminus \{(r, w_{x_i}), (w_{x_i}, w_{x_i}) \mid \mathfrak{I}(x_i) = 0\}. \end{aligned}$$

That is, we remove all worlds as well as their adjacent edges labeled with a proposition set to

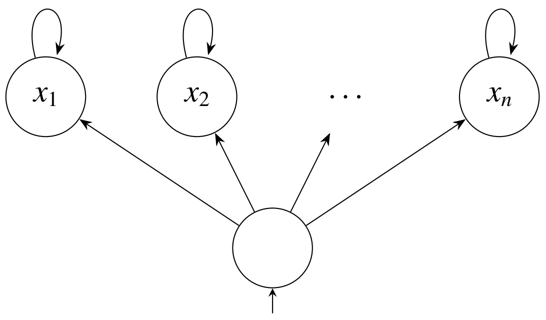


Figure 3.1: Kripke model $\mathcal{M}$ used in Theorem 20, with a root and worlds for each proposition of a propositional formula.

0 by the satisfying assignment of $\varphi$. This means that

$$\mathcal{M}' \models \mathsf{EX}\, x \iff \Im(x) = 1$$

and because the only difference between $\varphi$ and $\varphi_{CTL}$ is the temporal operator $\mathsf{EX}$ before propositions, it follows that $\mathcal{M}' \models \varphi_{CTL}$ and thus $\langle \mathcal{M}, \varphi_{CTL} \rangle \in$ ExistSubmodel.

Now suppose $\langle \mathcal{M}, \varphi_{CTL} \rangle \in$ ExistSubmodel. Then there exists a submodel of $\mathcal{M}$ that satisfies $\varphi_{CTL}$. Let $\mathcal{M}' = (W', R', \eta, r)$ be such a submodel. We construct an assignment $\Im$ from $\mathcal{M}'$ as follows:

$$\Im(x) := \begin{cases} 1, \text{if } w_x \in W' \\ 0, \text{otherwise.} \end{cases}$$

It should be clear that $\Im(x) = 1 \iff \mathcal{M}' \models x$ holds again. So by the same argument as above we have $\Im(\varphi) = 1$ and $\langle \varphi \rangle \in$ SAT.

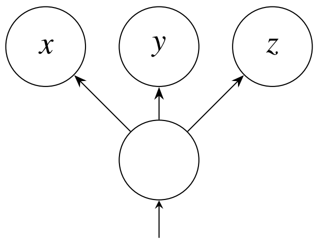
It follows that SAT $\leq_m^{\mathsf{P}}$ ExistSubmodel, since both the model $\mathcal{M}$ and the formula $\varphi_{CTL}$ can clearly be computed in polynomial time. □

To illustrate the proof, we give a short example, where we show the construction of the formula $\varphi_{CTL}$ and the Kripke model $\mathcal{M}$, as well as the connection between finding submodels and solving SAT.

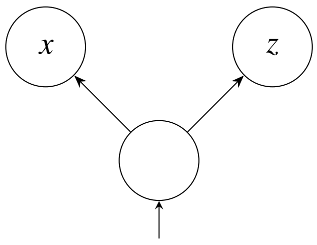
**Example 21.** Let $\varphi := (x \oplus y) \land \neg(y \lor \neg z)$ with PROP($\varphi$) = $\{x, y, z\}$. We then construct the CTL formula $\varphi_{CTL}$ as described above.

$$\varphi_{CTL} := (\mathsf{EX}\, x \oplus \mathsf{EX}\, y) \land \neg(\mathsf{EX}\, y \lor \neg\, \mathsf{EX}\, z)$$

The corresponding Kripke model $\mathcal{M} = (W, R, \eta, r)$ is as follows:



Assume we find the following satisfying submodel $\mathcal{M}' \subseteq \mathcal{M}$:



Now we construct the assignment for $\varphi$ as shown in the proof of Theorem 20. Since $w_x, w_z \in W'$ we have

$$\Im(x) = \Im(z) = 1$$

and with $w_y \notin W'$

$$\Im(y) = 0.$$

We can convince ourselves that $\Im$ indeed satisfies $\varphi$.

$$\begin{aligned}\varphi &= (x \oplus y) \wedge \neg(y \vee \neg z)\\ &= (1 \oplus 0) \wedge \neg(0 \vee \neg 1)\\ &= 1 \wedge \neg 0\\ &= 1\end{aligned}$$

Therefore $\varphi \in \mathrm{SAT}$ holds.

Having just shown DelNP-hardness, we now turn to membership. Once again, we use the tools of hard enumeration, this time to get around the need to give a DelNP algorithm. Instead we show that ExtendSubmodel $\in$ NP which, together with Corollary 13, gives the desired upper bound.

**Theorem 22.** E-CTL-Submodel $\in$ DelNP.

*Proof.* Algorithm 1 decides ExtendSubmodel and is in NP. The correctness should be obvious. If an extension $D'$ exists such that $\mathcal{M} - D' \models \varphi$, then we can nondeterministically guess the worlds and relations of that extension in line 1. Also guessing, computing $D'$ and checking $\mathcal{M} - D' \models \varphi$ can all clearly be done in polynomial time.

□

**Algorithm 1:** NP algorithm for ExtendSubmodel.

```
Input: Kripke model $\mathcal{M} = (W, R, \eta, r)$, CTL formula $\varphi$, set of deletions
       $D = (D_W, D_R)$
1 Guess a set of worlds $W' \subseteq W$ and a set of relations $R' \subseteq R$
2 $D' := (W' \cup D_W, R' \cup D_R)$
3 if $\mathcal{M} - D' \models \varphi$ then accept else reject
```

**Corollary 23.** E-CTL-Submodel *is* DelNP*-complete.*

*Proof.* Follows directly from Theorem 20 and Theorem 22. □

Notice that Theorem 22 is not only an upper bound for E-CTL-Submodel but also for all of its fragments too.

## 3.3 Results for monotone formulas

Since E-CTL-Submodel is intractable, we now want to restrict the access to some operators in hope of finding a tractable fragment. We start by studying monotone formulas. These are formulas that have no negation. Note that in the context of CTL, this restriction is actually not that harsh. Every CTL formula can be transformed into an equivalent formula in negation normal form, i.e. formulas with only atomic negations, using Observation 4. One can then introduce new propositions to simulate atomic negations. Thus, every formula-model pair has an equivalent formula-model pair without negations.

In this section we will show, that monotone CTL formulas with only E quantified temporal operators lead to our first tractability result, while A quantified temporal operators are still hard to enumerate.

Before showing tractability of E-CTL-Submodel(EX, EG, EF, EU, ER, $\wedge$, $\vee$) we first need two auxiliary results. Algorithm 2 will be used as a subroutine to "fix" a set of deletions to create an actual submodel that matches to our definition of a submodel. Lemma 24 then states that using this subroutine will not prevent polynomial delay.

The other Lemma 25 allows us to formulate a break continue, when searching through all possible submodels. It says that if a Kripke model does not satisfy a formula, then neither will one of its submodels, and therefore does not need to be considered by an enumeration algorithm.

The idea of the algorithm is to simply remove all "bad" worlds, i.e. worlds that violate the totality or connectivity of the submodel, and repeating to do so until either all worlds are removed or a valid submodel is found. The function of Line 7 will become clear later, when considering the full enumeration algorithm.

**Lemma 24.** *The function* `FixSubmodel()` *can be computed in polynomial time with respect to the size of the model.*

**Algorithm 2:** Repair a potential submodel

```
Input: A Kripke model $\mathcal{M} = (W, R, \eta, r)$ and a set of deletions $D = (D_W, D_R)$
Output: A set of deletions $D' = (D'_W, D'_R)$ such that $\mathcal{M} - D'$ is a submodel or empty
1 Function FixSubmodel($\mathcal{M}$, $D$):
2     $W' \leftarrow W \setminus D_W$, $R' \leftarrow R \setminus D_R$ and $D' \leftarrow D$
3     while $\exists w \in W'$ on no path in $\Pi(\mathcal{M})$ or $(w, \cdot) \cap R' = \emptyset$ do
4         $D'_W \leftarrow D'_W \cup \{w\}$
5         $D'_R \leftarrow \{(w, \cdot)\} \cup \{(\cdot, w)\}$, for all $(w, \cdot), (\cdot, w) \in R'$
6         $W' \leftarrow W \setminus D'_W$ and $R' \leftarrow R \setminus D'_R$
7         if $\exists r \in R'$ with $r <_R \min(D_R)$ then return $W(\mathcal{M}) \cup R(\mathcal{M})$    // no fix
8     return $D'$
```

*Proof.* The loop in line 2 iterates over each world at most once. Adding to $D'$, recalculation $W'$ and $R'$ and comparing all relations in $R'$ to the relations in $D$ can clearly all be done in polynomial time. Therefore the total runtime is polynomial. □

**Lemma 25.** *Let $\mathcal{M}' \subseteq \mathcal{M}$ be a submodel. If $\mathcal{M} \not\models \varphi$, for any* CTL *formula $\varphi$ with* CTL *operators in* $\{\mathsf{EX}, \mathsf{EG}, \mathsf{EF}, \mathsf{EU}, \mathsf{ER}\}$*, 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 $\varphi$ in $\mathcal{M}'$ also exist in $\mathcal{M}$. Since we only consider monotone CTL formulas, the same set of paths will satisfy $\varphi$ in $\mathcal{M}$. □

**Theorem 26.** E-CTL-Submodel($\mathsf{EX}, \mathsf{EG}, \mathsf{EF}, \mathsf{EU}, \mathsf{ER}, \wedge, \vee$) $\in$ DelP.

*Proof.* Using Lemma 25 as a break condition we construct a recursive enumeration algorithm (see Algorithm 3) starting with the original model $\mathcal{M}$ and successively removing relations (and nodes when necessary). We can guarantee no duplicate outputs by using an order $<_R$ over $R$, because for every submodel $\mathcal{M} - D$ that satisfies $\varphi$, there is a unique sequence of recursive calls to `EnumSubgraphRec()`. This must be a subsequence of the sequence given by the relations in $D$ and $<_R$. Any relation missing from the subsequence has been removed by a call to `FixSubmodel()`, which also respects the order. So there is only *one* way to output a submodel.

The algorithm also does not miss any possible submodels. A simple induction shows that for every satisfying submodel, there exists a satisfying submodel with at least one more relation up to the original model, and it is quite easy to see that our algorithm will consider and output all submodels in this chain.

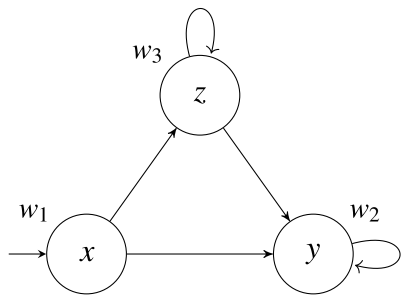
The delay of the algorithm mainly depends on the calls to `FixSubmodel()`, the model checking of the current submodel and the number of submodels not satisfying $\varphi$ considered between two solutions. We have shown in Lemma 24 that `FixSubmodel()` can be computed in polynomial time. Model checking is a well-studied problem and it is well known to be computable in polynomial time [18]. Finally, the number of "bad" submodels considered is bounded by the depth of the recursion and the number of iterations in the for-loop. Both are bounded by the number of relations in the model, giving an upper bound of $O(|\mathcal{M}|^2)$. Together the algorithm is then obviously in DelP. □

**Algorithm 3:** Enumerate the satisfying submodels for a monotone CTL formula with only E quantifiers

```
Input: A Kripke model $\mathcal{M} = (W, R, \eta, r)$ and a CTL formula $\varphi$ with operators
       $T \subseteq \{\mathsf{EX}, \mathsf{EG}, \mathsf{EF}, \mathsf{EU}, \land, \lor, \oplus\}$
Output: All submodels $\mathcal{M}' \subseteq M$ with $\mathcal{M}' \models \varphi$
1 EnumSubgraphRec($\mathcal{M}, \varphi, \emptyset$)
2
3 Procedure EnumSubgraphRec($\mathcal{M}, \varphi, D$):
4     if $\mathcal{M} - D$ is not a submodel then
5         $D \leftarrow$ FixSubmodel($\mathcal{M}, D$)
6     if $\mathcal{M} - D \models \varphi$ then
7         output $\mathcal{M} - D$
8         for $e \in R \setminus D_R$ and $\min(D_R) \prec_R e$ do        // prevents duplicates
9             EnumSubgraphRec($\mathcal{M}, \varphi, D \cup \{e\}$)
```

Let us now visualise the algorithms functionality with an example.

**Example 27.** Let $\mathcal{M}$ be a Kripke model as follows:



Next, consider the following CTL formula

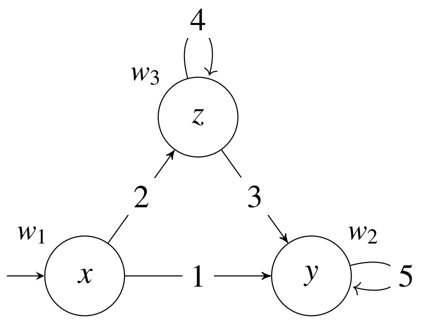
$$\varphi := x \land \mathsf{EF}\, z.$$

The goal is to enumerate all submodels $\mathcal{M}' \subseteq \mathcal{M}$ that satisfy $\varphi$.

First fix the order $\prec_R$ as

$$(w_1, w_2) \prec_R (w_1, w_3) \prec_R (w_3, w_2) \prec_R (w_3, w_3) \prec_R (w_2, w_2).$$

We can visualise it in the model by labeling the relations as follows:



Next, we start the algorithm by calling the procedure

$$\texttt{EnumSubgraphRec}(\mathcal{M}, x \wedge \mathsf{EF}\, z, \emptyset).$$

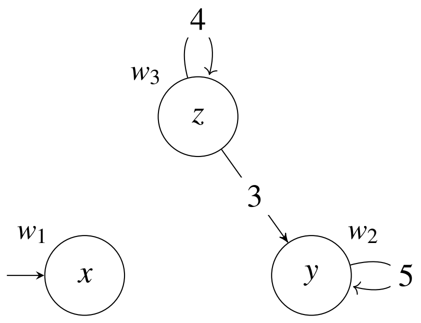
The algorithm tests if $\mathcal{M} - \emptyset \models x \wedge \mathsf{EF}\, z$, which is true. We therefore get our first solution $\mathcal{M} - \emptyset$, i.e. the original model $\mathcal{M}$. The algorithm then starts the recursion with

$$\texttt{EnumSubgraphRec}(\mathcal{M}, x \wedge \mathsf{EF}\, z, \{(w_1, w_2)\}).$$

$\mathcal{M} - \{(w_1, w_2)\} \models x \wedge \mathsf{EF}\, z$ is again true and $\mathcal{M} - \{(w_1, w_2)\}$ is the second solution. The algorithm continues deeper into the recursion with

$$\texttt{EnumSubgraphRec}(\mathcal{M}, x \wedge \mathsf{EF}\, z, \{(w_1, w_2), (w_1, w_3)\}).$$

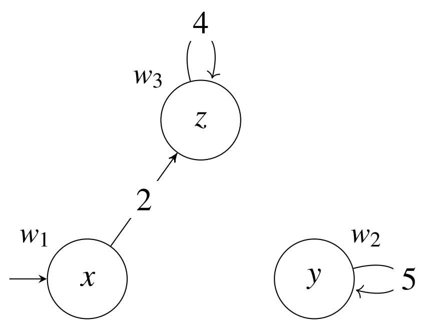
$\mathcal{M} - \{(w_1, w_2), (w_1, w_3)\}$ is not a submodel, because there is no path from the root $w_1$ to $w_2$ and $w_3$.



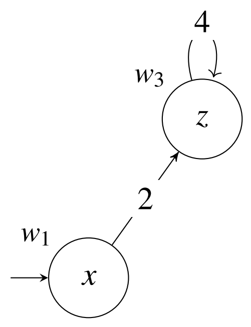
The algorithm therefore tries to repair the submodel, which results in the empty submodel, because the root has no outgoing relation and has to be removed. This branch of the recursions has reached a dead end and we need to backtrack. There are still relations to be considered and up next is

$$\texttt{EnumSubgraphRec}(\mathcal{M}, x \wedge \mathsf{EF}\, z, \{(w_1, w_2), (w_2, w_3)\}).$$

Resulting in the following submodel:



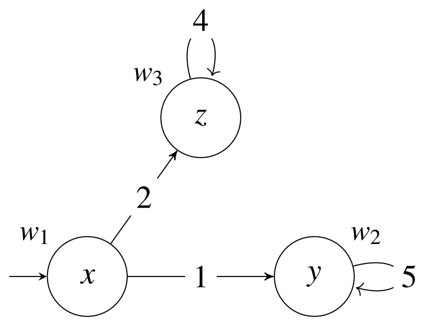
This submodel is also not connected and needs to be repaired. This time it is in fact possible, by removing the world $w_2$ and the relation $(w_2.w_2)$, resulting in the following submodel:



This submodel is total and connected and $\mathcal{M} - \{w_2, (w_1, w_2), (w_2, w_3), (w_2, w_2)\} \models x \wedge \mathsf{EF}\, z$ is true, thus being our third solution.

There are two relations left, but because $(w_2, w_2) \not<_R (w_1, w_3)$ and $(w_2, w_2) \not<_R (w_3, w_3)$, we cannot remove either.

Finally, let us take a look how the order $<_R$ over the relations prevents outputting duplicate submodels. Assume we just called `EnumSubgraphRec(`$\mathcal{M}, x \wedge \mathsf{EF}\, z, \{(w_3, w_2)\}$`)`. The submodel $\mathcal{M} - \{(w_3, w_2)\}$



satisfies $x \wedge \mathsf{EF}\, z$. Removing the relation $(w_1, w_2)$ would result in the same submodel as our second solution above. But because $(w_3, w_2) \not<_R (w_1, w_2)$, this is not allowed and we would continue with the relations $(w_3, w_3)$ and $(w_2, w_2)$. Lets assume we continue with $(w_2, w_2)$. The resulting submodel is not total, because $w_2$ has no outgoing relations. The algorithm therefore calls `FixSubmodel(`$\mathcal{M}, \{(w_3, w_2), (w_2, w_2)\}$`)`, but again we are prohibited from removing the relation $(w_1, w_2)$, and cannot repair the submodel resulting in no further outputs on this branch.

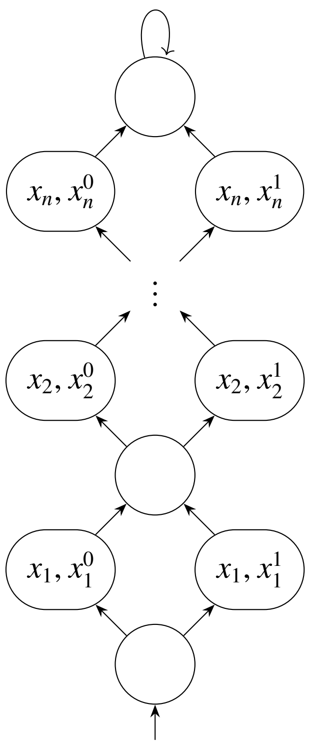


Figure 3.2: Kripke model $\mathcal{M}(\varphi)$ from Definition 28

Next we show that a single universal temporal operator leads to intractability.

**Definition 28.** Let $\varphi$ be a propositional formula with propositions $\mathrm{PROP}(\varphi) = \{x_1, x_2, \ldots, x_n\}$. We define the model $\mathcal{M}(\varphi) = (W, R, \eta, r)$ with the following properties.

$$\begin{aligned}
W :=& \{r, w_i, w_i^0, w_i^1\} && \\
R :=& \{(r, w_1^0), (r, w_1^1)\} && \text{Edge from the root to the first layer} \\
&\cup \{(w_i^0, w_i), (w_i^1, w_i)\} && \text{Edge from a layer to its joint} \\
&\cup \{(w_i, w_{i+1}^0), (w_i, w_{i+1}^1)\} && \text{Edge from the joint to the next layer} \\
&\cup \{(w_n, w_n)\} && \text{Loop on the last joint} \\
\eta(w_i^k) :=& \{x_i, x_i^k\} &&
\end{aligned}$$

for all $1 \leq i \leq n$.

A visualisation of the definition above can be seen in Figure 3.2, with the proposition labeling instead of the worlds names.

**Theorem 29.** *The following enumeration problems are* DelNP*-complete:*

1. E-CTL-Submodel($\mathsf{AX}, \wedge, \vee$)

2. E-CTL-Submodel($\mathsf{AG}, \wedge, \vee$)

3. E-CTL-Submodel($\mathsf{AF}, \wedge, \vee$)

*Proof.* The upper bound follows from Theorem 22.

We will detail the proof of the lower bound for E-CTL-Submodel($\mathsf{AF}, \wedge, \vee$). The other two cases will follow analogously.

By Corollary 17 we only need to show NP-hardness for Exist_E-CTL-Submodel($\mathsf{AF}, \wedge, \vee$), which is a decision problem asking, if there exists a submodel of the given model satisfying the given formula. For that we present a reduction from SAT.

Let $\varphi$ be propositional formula in negation normal form. We construct a formula $\varphi_{\mathsf{AF}}$ by replacing all occurrences of $x_i$ by $\mathsf{AF}\, x_i^1$ and $\neg x_i$ by $\mathsf{AF}\, x_i^0$. Next we need to show that

$$\langle\varphi\rangle \in \mathrm{SAT} \iff \langle\mathcal{M}(\varphi), \varphi_{\mathsf{AX}}\rangle \in \textsc{ExistSubmodel}(\mathsf{AF}, \wedge, \vee).$$

Assume that $\varphi \in \mathrm{SAT}$. Then there is an assignment $\mathfrak{I}$ such that $\mathfrak{I}(\varphi) = 1$. Using this assignment, we construct a submodel $\mathcal{M}' \subseteq \mathcal{M}(\varphi)$ by removing worlds and relations containing them as follows:

- if $\mathfrak{I}(x_i) = 1$, then remove $w_i^0$, likewise
- if $\mathfrak{I}(x_i) = 0$, then remove $w_i^1$.

Observe that $\mathcal{M}' \models \mathsf{AF}\, x_i^1$, iff $\mathfrak{I}(x_i) = 1$, since all paths of $\mathcal{M}'$ have to contain $w_i^1$, because it is the only successor world of $w_i$. Analogously, $\mathcal{M}' \models \mathsf{AF}\, x_i^0$, if and only if $\mathfrak{I}(x_i) = 0$. Recall that the formula $\varphi_{\mathsf{AX}}$ differs from $\varphi$ only in its atoms. It follows that $\mathcal{M}' \models \varphi_{\mathsf{AX}}$ must be true.

In the same way, if there is a submodel such that $\mathcal{M}' \models \varphi_{\mathsf{AX}}$, we can construct an assignment $\mathfrak{I}$ with $\mathfrak{I}(\varphi) = 1$, concluding the proof for DelNP-hardness of E-CTL-Submodel($\mathsf{AF}, \wedge, \vee$).

The fragments E-CTL-Submodel($\mathsf{AX}$, $\wedge$, $\vee$) and E-CTL-Submodel($\mathsf{AG}$, $\wedge$, $\vee$) follow the same approach using different CTL-formula. For E-CTL-Submodel($\mathsf{AG}, \wedge, \vee$) we construct the formula $\varphi_{\mathsf{AG}}$ by replacing all occurrences of $x_i$ with $\mathsf{AG}(\neg x_i \vee x_i^1)$ and $\neg x_i$ by $\mathsf{AG}(\neg x_i \vee x_i^0)$. These replacements work in the same way as in the $\mathsf{AF}$ case. $\mathsf{AG}(\neg x_i \vee x_i^1)$ only holds, if the world $w_i^0$ is removed, while $\mathsf{AG}(\neg x_i \vee x_i^0)$ only holds, if $w_i^1$ is removed.

For $\mathsf{AX}$, we need to consider how far away the worlds are from the root. Keeping this in mind we can simulate the behaviour of $\mathsf{AF}$ by concatenating multiple $\mathsf{AX}$ operators. the resulting formula $\varphi_{\mathsf{AX}}$ is constructed as follows: Replace all occurrences of $x_i$ with $\mathsf{AX}^{2i-1}\, x_i^1$ and $\neg x_i$ by $\mathsf{AX}^{2i-1}\, x_i^0$, using $\mathsf{AX}^n$ as an abbreviation for $n$ concatenated $\mathsf{AX}$ operators. □

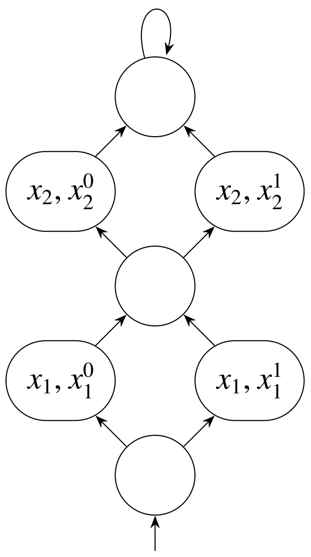


Figure 3.3: Model $\mathcal{M}(\varphi)$ in Example 32.

**Corollary 30.** E-CTL-Submodel($\mathsf{AU}$, $\wedge$, $\vee$) *and* E-CTL-Submodel($\mathsf{AR}$, $\wedge$, $\vee$) *are* $\mathsf{DelNP}$-*complete.*

*Proof.* We can simulate the behaviour of $\mathsf{AF}$ and $\mathsf{AG}$ respectively using the identities $\mathsf{AG}\,\varphi = \varphi\,\mathsf{AR}\,\bot$ and $\mathsf{AF}\,\varphi = \top\,\mathsf{AU}\,\varphi$. □

The next result shows how this affects the the clone with all Boolean operators. While we have already shown that E-CTL-Submodel with access to all temporal and Boolean operators is $\mathsf{DelNP}$-complete, we now prove that a single temporal operator already has enough expressivity to show $\mathsf{DelNP}$-hardness.

**Corollary 31.** E-CTL-Submodel($T, \wedge, \oplus$) *is* $\mathsf{DelNP}$-*complete, if $T$ contains at least one* $\mathsf{CTL}$ *operator.*

*Proof.* For $\mathsf{A}$ operator this follows directly from Theorem 29 and Corollary 30. If $T$ contains only $\mathsf{E}$ operators we can use Observation 4 to construct an equivalent formula using only $\mathsf{A}$ operator. □

**Example 32.** Let $\varphi = (x_1 \wedge \neg x_2) \vee (x_2 \wedge (\neg x_1 \vee 1) \wedge 0)$ be a propositional formula, with $\mathrm{PROP}(\varphi) = \{x_1, x_2\}$. Then $\mathcal{M}(\varphi)$ is as depicted in Figure 3.3 and $\varphi_T$ is, depending on the CTL operators, as follows:

$$\begin{aligned}
\varphi_{AX} &= (\mathsf{AX}\,x_1^1 \wedge \mathsf{AX}\,\mathsf{AX}\,\mathsf{AX}\,x_2^0) \vee (\mathsf{AX}\,\mathsf{AX}\,\mathsf{AX}\,x_2^1 \wedge (\mathsf{AX}\,x_1^0 \vee \top) \wedge \bot)\\
\varphi_{AF} &= (\mathsf{AF}\,x_1^1 \wedge \mathsf{AF}\,x_2^0) \vee (\mathsf{AF}\,x_2^1 \wedge (\mathsf{AF}\,x_1^0 \vee \top) \wedge \bot)\\
\varphi_{AG} &= (\mathsf{AG}(\neg x_1 \vee x_1^1) \wedge \mathsf{AG}(\neg x_2 \vee x_2^0)) \vee (\mathsf{AG}(\neg x_2 \vee x_2^1) \wedge (\mathsf{AG}(\neg x_2 \vee x_1^0) \vee \top) \wedge \bot)
\end{aligned}$$

The submodel $\mathcal{M}'$ as shown in Figure 3.4 satisfies $\varphi_T$ in all three cases. In the first case $\mathsf{AX}\,x_1^1$ and $\mathsf{AX}\,\mathsf{AX}\,\mathsf{AX}\,x_2^0$ are true and thus the whole formula. In the second case the

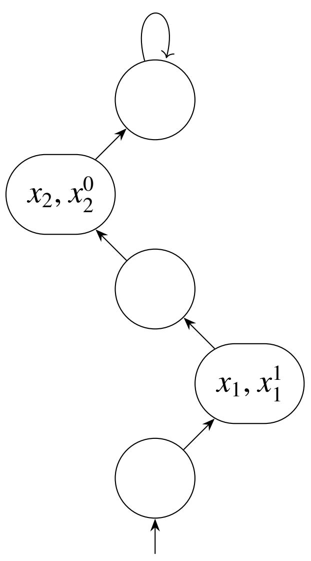


Figure 3.4: Submodel $\mathcal{M}'$ of the model in Figure 3.3 with $\mathcal{M}' \models \varphi_T$.

corresponding $\mathsf{AF}\, x_1^1$ and $\mathsf{AF}\, x_2^0$ are true. And lastly in the third case $\mathsf{AG}(x_1 \rightarrow x_1^1)$ and $\mathsf{AG}(x_2 \rightarrow x_2^0)$ are true. In all these cases we can deduce a satisfying assignment for the original propositional formula $\varphi$. In this example the assignment is $\Im(x_1) = 1$ and $\Im(x_2) = 0$, resulting in

$$\begin{aligned} \Im(\varphi) &= (1 \land \neg 0) \lor (0 \land (\neg 1 \lor 1) \\ &= 1 \lor 0 \\ &= 1 \end{aligned}$$

which shows $\varphi \in \text{SAT}$ as desired.

## 3.4 Results for formulas with only conjunction

Having fully classified the monotone case in the last section, we continue to restrict access to Boolean operators. Fortunately, the tractability results for $\mathsf{E}$ quantified temporal operators carry over to the less expressive clones. Thus, from here on out, we will mainly focus on $\mathsf{A}$ quantified $\mathsf{CTL}$ operators.

In this section we will show that the fragment containing only $\mathsf{AF}$ and the Boolean conjunction is already $\mathsf{DelNP}$-hard. So far we have been to use a fairly straightforward connection between finding satisfying assignments for propositional formulas and finding satisfying submodels. Without access to disjunction, this connection seems much less obvious. We will therefore consider another $\mathsf{NP}$-complete problem and make a similar connection between its solutions and satisfying submodels. Our problem of choice will be HAMPATH, which we have formally defined in Figure 2.6.

We begin by defining the Kripke model used in the proof of Theorem 35.

**Definition 33.** Let $G = (V, E)$ be a directed graph and $s, t \in V$ be some nodes of $G$. Further let $H := \langle G, s, t\rangle$.

We define the model $\mathcal{M}(H) := (W, R, \eta, w_s)$ as follows:

$$\begin{aligned}
W &:= \{w_v, \hat{w}_v, w_{v,i} \mid v \in V, 1 \le i \le |V|\} \\
R &:= \{(w_v, w_{v,i}), (w_{v,i}, \hat{w}_v) \mid v \in V, 1 \le i \le |V|\} \\
&\quad \cup \{(\hat{w}_u, w_v) \mid (u, v) \in E \text{ and } u \neq t\} && \text{($t$ has no outgoing relation)} \\
&\quad \cup \{(\hat{w}_t, \hat{w}_t)\} && \text{(except to itself)} \\
\eta(w_{v,i}) &:= \{i\} \text{ for } 1 \le i \le |V| \\
\eta(w_t) &:= \{t\}
\end{aligned}$$

For a better understanding of the above definition, let us consider the following example.

**Example 34.** Let $G = (V, E)$ be the graph depicted in Figure 3.5. The Kripke model constructed as described in Definition 33 can be seen in Figure 3.6.

**Theorem 35.** E-CTL-Submodel($\mathsf{AF}, \wedge$) *is* $\mathsf{DelNP}$*-complete.*

*Proof.* The upper bound follows directly from Theorem 20. For the lower bound we give a reduction from the $\mathsf{NP}$-complete problem HAMPATH to ExistSubmodel($\mathsf{AF}, \wedge$).

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 $G$ as described in Definition 33. We construct a $\{\mathsf{AF}, \wedge\}$-formula as follows:

$$\varphi := \mathsf{AF}(1 \wedge \mathsf{AF}(2 \wedge \mathsf{AF}(\cdots \wedge \mathsf{AF}(n \wedge \mathsf{AF}\, t))))$$

We now show that the Kripke model $\mathcal{M}(H)$ and the formula $\varphi$ are constructed in such a way that all submodels $\mathcal{M}'(H) \subseteq \mathcal{M}(H)$, with $\mathcal{M}'(H) \models \varphi$, must consist only of paths from $w_s$ to $\hat{w}_t$ containing all other worlds $w_v$ for $v \in V$. That is all submodels $\mathcal{M}'(H)$ describe Hamiltonian paths, with respect to the worlds $w_v$ in the Kripke model and thus Hamiltonian paths in the graph $G$. It is then obvious that a submodel of $\mathcal{M}(H)$ which satisfies $\varphi$ can only exist if the original graph has a Hamiltonian path.

Assume $\mathcal{M}'(H)$ is a satisfying submodel. We first show that $\varphi$ forces all infinite paths in $\mathcal{M}'(H)$ to end in a loop at $\hat{w}_t$. For that we start by showing that all paths $\pi \in \Pi(\mathcal{M}'(H))$ contain the world $\hat{w}_t$.

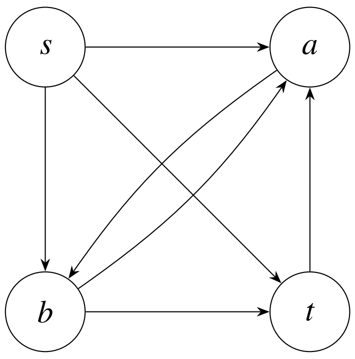


Figure 3.5: Example graph with the Hamiltonian path $s, a, b, t$

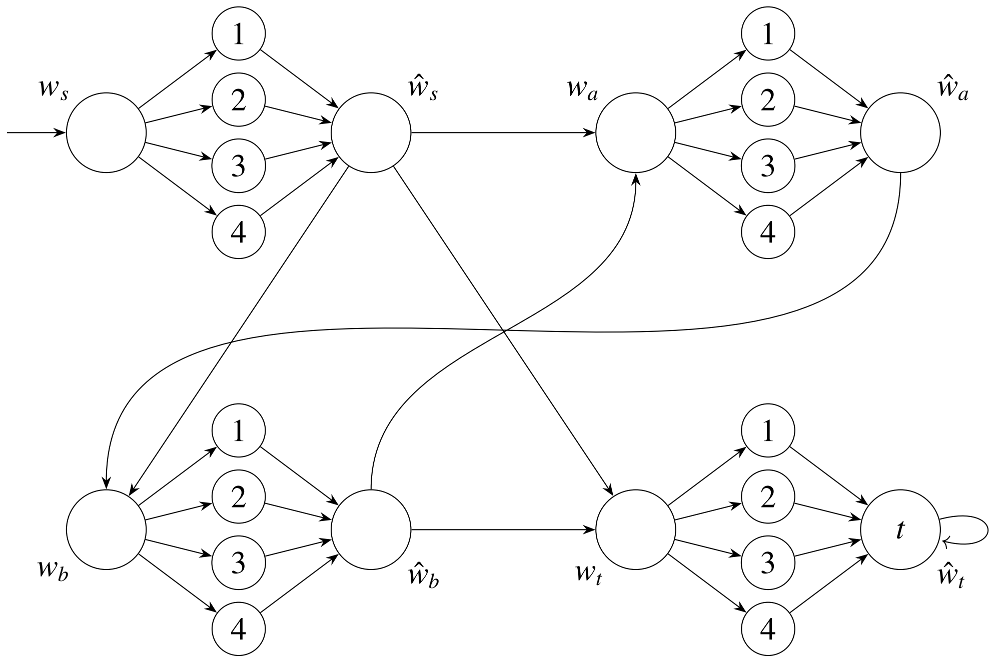


Figure 3.6: Kripke model $\mathcal{M}(H)$ of the graph in Figure 3.5.

$$
\begin{aligned}
&\mathcal{M}'(H), r \models \mathsf{AF}(1 \land \mathsf{AF}(2 \land \mathsf{AF}(\cdots \land \mathsf{AF}(n \land \mathsf{AF}\, t)))) && \text{(with } r \text{ root of } \mathcal{M}'(H)\text{)}\\
&\Rightarrow \forall \pi \in \Pi(r) \exists k \geq 1 \colon \mathcal{M}'(H), \pi[k] \models 1 \land \mathsf{AF}(2 \land \mathsf{AF}(\cdots \land \mathsf{AF}(n \land \mathsf{AF}\, t)))\\
& && \text{(Definition of } \mathsf{AF}\text{)}\\
&\Rightarrow \forall \pi \in \Pi(r) \exists k \geq 1 \colon \mathcal{M}'(H), \pi[k] \models \mathsf{AF}(2 \land \mathsf{AF}(\cdots \land \mathsf{AF}(n \land \mathsf{AF}\, t))) && \text{(Definition of } \land\text{)}\\
&\Rightarrow \forall \pi \in \Pi(r) \exists k \geq 1 \forall \pi' \in \Pi(\pi[k]) \exists k' \geq 1 \colon\\
&\qquad \mathcal{M}'(H), \pi'[k'] \models 2 \land \mathsf{AF}(\cdots \land \mathsf{AF}(n \land \mathsf{AF}\, t)) && \text{(}\mathsf{AF}\text{ again)}\\
&\Rightarrow \forall \pi \in \Pi(r) \exists k \geq 1 \exists k' \geq k \colon \mathcal{M}'(H), \pi[k'] \models 2 \land \mathsf{AF}(\cdots \land \mathsf{AF}(n \land \mathsf{AF}\, t))\\
& && \text{(all } \pi' \in \Pi(\pi[k]) \text{ also exist in } \Pi(r) \text{ with some prefix of length } k-1\text{)}\\
&\Rightarrow \forall \pi \in \Pi(r) \exists k \geq 1 \colon \mathcal{M}'(H), \pi[k] \models 2 \land \mathsf{AF}(\cdots \land \mathsf{AF}(n \land \mathsf{AF}\, t)) && \text{(take } k = k'\text{)}\\
&\Rightarrow \forall \pi \in \Pi(r) \exists k \geq 1 \colon \mathcal{M}'(H), \pi[k] \models t && \text{(repeat steps above)}\\
&\Rightarrow \mathcal{M}'(H), r \models \mathsf{AF}\, t && \text{(Definition of } \mathsf{AF}\text{)}
\end{aligned}
$$

Only one world in $\mathcal{M}(H)$ is labeled with the proposition $t$, that is $\hat{w}_t$. Because $\mathcal{M}'(H) \models \mathsf{AF}\, t$ is true if and only if all paths contain such a world, all paths must contain $\hat{w}_t$.

For any path $\pi$ of $\Pi(\mathcal{M}'(H))$ assume that $\pi[k] = \hat{w}_t$ with $k \in \mathbb{N}$. Now, the only outgoing relation of $\hat{w}_t$ in $\mathcal{M}'(H)$ is $(\hat{w}_t, \hat{w}_t)$, because it is the only outgoing relation of $\hat{w}_t$ in $\mathcal{M}(H)$. Therefore for all $i \geq k$ we have that $\pi[i] = \hat{w}_t$.

Next, we show that all paths of $\mathcal{M}'(H)$ must contain all worlds $w_v$ for $v \in V$. Analogously to above $\mathcal{M}'(H) \models \varphi$ implies

$$\mathcal{M}'(H) \models \mathsf{AF}\, 1, \mathcal{M}'(H) \models \mathsf{AF}\, 2, \ldots \text{ and } \mathcal{M}'(H) \models \mathsf{AF}\, n.$$

This means there are $k_1, k_2, \ldots, k_n \in \mathbb{N}$ with

$$\mathcal{M}'(H), \pi[k_1 + 1] \models 1, \mathcal{M}'(H), \pi[k_2 + 1] \models 2, \ldots \text{ and } \mathcal{M}'(H), \pi[k_n + 1] \models n.$$

on all paths $\pi \in \Pi(\mathcal{M}'(H))$. The worlds labeled with numbers in the submodel $\mathcal{M}'(H)$ are $w_{v,k}$ for $v \in V$ and $1 \leq k \leq n$, which have $w_v$ as predecessor respectively. Thus all paths must contain $\geq n$ worlds $w_v$. With a total number of $n$ such worlds in $\mathcal{M}'(H)$, it remains to show that all paths can visit these worlds only once.

A consequence of all paths leading to $\hat{w}_t$ is that the underlying graph of $\mathcal{M}'(H)$ must be acyclic, except for the loop at the world $\hat{w}_t$. Assume $\mathcal{M}'(H)$ is not acyclic. Then $\mathcal{M}'(H)$ would have a cycle. But the cycle obviously cannot contain $\hat{w}_t$, because $\hat{w}_t$ has no outgoing relations except to itself, which contradicts the above statement.

This acyclicity of the model, and the aforementioned necessity to visit at least $n$ worlds $w_v$,

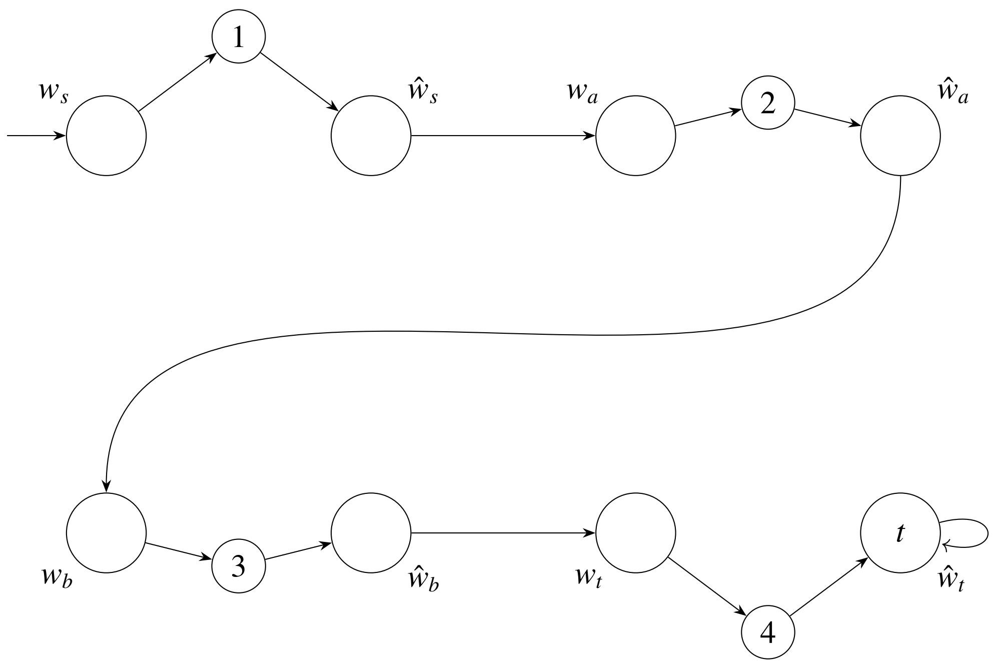


Figure 3.7: Submodel of the model in Figure 3.6 describing the Hamiltonian path $s, a, b, t$ of the graph in Figure 3.5.

leads to each world $w_v$ being on all paths *exactly* once.

The infinite paths of a satisfying submodel $\mathcal{M}'(H) \subseteq \mathcal{M}(H)$ start at $w_r$ by definition, we have shown that they must reach $\hat{w}_t$ and contain all $w_v$ for $v \in V$ exactly once, therefore describing a Hamiltonian path from $s$ to $t$ in $G$.

Finally, the reduction function $\langle G, s, t\rangle \mapsto \langle \mathcal{M}(H), \varphi\rangle$ is polynomial time computable, since both the Kripke model and the formula can be constructed in polynomial time. □

Let us again visualise the proof with an example.

**Example 36.** Let $G = (V, E)$ be the graph from Figure 3.5 and $\mathcal{M}(H)$ be the constructed Kripke model depicted in Figure 3.6. We construct the following $\{\mathsf{AF}, \wedge\}$-formula:

$$\varphi = \mathsf{AF}(1 \wedge \mathsf{AF}(2 \wedge \mathsf{AF}(3 \wedge \mathsf{AF}(4 \wedge \mathsf{AF}\, t))))$$

Figure 3.7 depicted a submodel of $\mathcal{M}(H)$ that satisfies this formula and clearly shows the corresponding Hamiltonian path of $G$.

We will continue using this concept of forcing submodels to describe Hamiltonian paths in the following section. Before that, we want to state the following corollary and remark.

**Corollary 37.** E-CTL-Submodel($\mathsf{AU}, \wedge$) *is* $\mathsf{DelNP}$*-complete*

*Proof.* Follows directly from the equivalence $\mathsf{AF}\,\varphi \equiv \top\, \mathsf{AU}\, \varphi$. □

*Remark* 38. E-CTL-Submodel($\mathsf{AX}, \wedge$) can also be shown to be $\mathsf{DelNP}$-complete, by using a similar reduction as above with the following formula:

$$\varphi := \mathsf{AX}(1 \wedge \mathsf{AX}\,\mathsf{AX}\,\mathsf{AX}(2 \wedge \ldots (n\,\mathsf{AX}\,t).$$

We won't fully prove this, as we give an even stronger result in the next section. But it should be clear that this formula simply mimics the behaviour of the formula used in the proof of Theorem 35, with the structure of the Kripke model in mind.

## 3.5 Results for formulas without Boolean operators

In this sections we will show that some $\mathsf{CTL}$ operators are inherently hard, when trying to enumerate satisfying submodels, by disallowing all Boolean operators. We will again relate submodels to Hamiltonian paths of graphs resulting analogous proves to the proof of Theorem 35.

**Theorem 39.** E-CTL-Submodel($\mathsf{AU}$) *is* $\mathsf{DelNP}$*-complete.*

*Proof.* The upper bound, as always, follows directly from Theorem 22.

For the lower bound we again present a reduction from HAMPATH. Let $G = (V, E)$ be a graph, $n = |V|$, $s \in V$ and $t \in V$. We make use of the same Kripke model $\mathcal{M}(H)$, with $H := \langle G, s, t\rangle$, introduced in Definition 33 together with the following $\{\mathsf{AU}\}$-formula

$$\varphi := ((((\top\,\mathsf{AU}\,t)\,\mathsf{AU}\,n)\,\mathsf{AU}\,n-1)\cdots\mathsf{AU}\,2)\,\mathsf{AU}\,1.$$

We show that this formula forces satisfying submodels $\mathcal{M}'(H) \subseteq \mathcal{M}(H)$ to consist of paths from $w_s$ to $\hat{w}_t$ over all worlds $w_v$ with $v \in V$ and each $w_v$ is on the path exactly once, i.e. they describe a Hamiltonian path of $G$ in $\mathcal{M}(H)$.

Let us start by showing that all paths of any $\mathcal{M}'(H)$ must reach $\hat{w}_t$.

$$\begin{aligned}
&\mathcal{M}'(H), w_s \models ((((\top\,\mathsf{AU}\,t)\,\mathsf{AU}\,n)\,\mathsf{AU}\,n-1)\cdots\mathsf{AU}\,2)\,\mathsf{AU}\,1 \\
&\Rightarrow \forall\pi \in \Pi(w_s)\exists k \geq 1 \forall i < k: \\
&\qquad \mathcal{M}'(H), \pi[i] \models (((\top\,\mathsf{AU}\,t)\,\mathsf{AU}\,n)\,\mathsf{AU}\,n-1)\cdots\mathsf{AU}\,2 && \text{(Second half of } \mathsf{AU}\text{)} \\
&\Rightarrow \forall\pi \in \Pi(w_s): \mathcal{M}'(H), \pi[1] \models (((\top\,\mathsf{AU}\,t)\,\mathsf{AU}\,n)\,\mathsf{AU}\,n-1)\cdots\mathsf{AU}\,2 && \text{(take } i = 1\text{)} \\
&\Rightarrow \mathcal{M}'(H), w_s \models (((\top\,\mathsf{AU}\,t)\,\mathsf{AU}\,n)\,\mathsf{AU}\,n-1)\cdots\mathsf{AU}\,2 && (\pi[1] = w_s) \\
&\Rightarrow \mathcal{M}'(H), w_s \models \top\,\mathsf{AU}\,t && \text{(repeat)} \\
&\Rightarrow \mathcal{M}'(H), w_s \models \mathsf{AF}\,t && \text{(Observation 4)}
\end{aligned}$$

It follows that the underlying graph of $\mathcal{M}'(H)$ must be acyclic as demonstrated in the proof of Theorem 35.

Next we prove that all paths in $\mathcal{M}'(H)$ visited all worlds $w_v$ for $v \in V$.

$$\begin{aligned}
&\mathcal{M}'(H), w_s \models (((\top \mathsf{AU}\ t) \mathsf{AU}\ n) \cdots \mathsf{AU}\ 2) \mathsf{AU}\ 1 \\
&\Rightarrow \forall \pi \in \Pi(w_s) \exists k \geq 1 : \mathcal{M}'(H), \pi[k] \models 1 \text{ and} \\
&\quad \forall i < k : \mathcal{M}'(H), \pi[i] \models ((\top \mathsf{AU}\ t) \mathsf{AU}\ n) \cdots \mathsf{AU}\ 2 && \text{(definition AU)} \\
&\Rightarrow \forall \pi \in \Pi(w_s) \exists k \geq 1 : \mathcal{M}'(H), \pi[k] \models 1 \text{ and} \\
&\quad \mathcal{M}'(H), \pi[k-1] \models ((\top \mathsf{AU}\ t) \mathsf{AU}\ n) \cdots \mathsf{AU}\ 2 && \text{(take } i = k-1\text{)} \\
&\Rightarrow \forall \pi \in \Pi(w_s) \exists k \geq 1 : \mathcal{M}'(H), \pi[k] \models 1 \text{ and } \forall \pi' \in \Pi(k-1) \exists j \geq 1 : \mathcal{M}'(H), \pi'[j] \models 2 \text{ and} \\
&\quad \forall i < j : \mathcal{M}'(H), \pi'[i] \models ((\top \mathsf{AU}\ t) \mathsf{AU}\ n) \cdots \mathsf{AU}\ 3 && \text{(definition AU)} \\
&\Rightarrow \forall \pi \in \Pi(w_s) \exists k \geq 1 : \mathcal{M}'(H), \pi[k] \models 1 \text{ and } \exists k' \geq k-1 : \mathcal{M}'(H), \pi[k'] \models 2 \text{ and} \\
&\quad \forall i < j : \mathcal{M}'(H), \pi[i] \models ((\top \mathsf{AU}\ t) \mathsf{AU}\ n) \cdots \mathsf{AU}\ 3 && \text{(all } \pi' \text{ in } \Pi(R) \text{ with some prefix)} \\
&\Rightarrow \forall \pi \in \Pi(w_s) \exists k \geq 1 : \mathcal{M}'(H), \pi[k] \models 1 \text{ and } \exists k' \geq k-1 : \mathcal{M}'(H), \pi[k'] \models 2 \text{ and} \\
&\quad \mathcal{M}'(H), \pi[k'-1] \models ((\top \mathsf{AU}\ t) \mathsf{AU}\ n) \cdots \mathsf{AU}\ 3 && \text{(take } i = k'-1\text{)}
\end{aligned}$$

Repeating this process leads to

$$\begin{aligned}
&\forall \pi \in \Pi(w_s) \exists k \geq 1 : \mathcal{M}'(H), \pi[k] \models 1 \\
&\text{and } \exists k' \geq k-1 : \mathcal{M}'(H), \pi[k'] \models 2 \\
&\text{and } \ldots \\
&\text{and } \exists k^{(n-1)} \geq k^{(n-2)} - 1 : \mathcal{M}'(H), \pi[k^{(n-1)}] \models n.
\end{aligned}$$

Notice that in the construction of $\mathcal{M}(H)$ the predecessor world of worlds labeled with a number has no label themselves. Also notice that the world label with a number has no other label. Therefore $k^{(i)} > k^{(i-1)}$ instead of $k^{(i)} \geq k^{(i-1)} - 1$ must hold for $0 \leq i \leq n-1$.

From this we can conclude that all paths of $\mathcal{M}'(H)$ have to contain $n$ worlds labeled from 1 to $n$. Since labeled words can only be reached from worlds $w_v$ for $v \in V$, each $w_v$ can only be on a path once due to the acyclicity of $\mathcal{M}'(H)$, and there are $n$ worlds $w_v$ in total, this means that all $w_v$ have to be on all paths of $\mathcal{M}'(H)$ exactly once.

This shows that $H \in$ HAMPATH, iff $\exists \mathcal{M}'(H) \subseteq \mathcal{M}(H) : \mathcal{M}'(H) \models \varphi$. $\mathcal{M}(H)$ and $\varphi$ can be computed in polynomial time. So HAMPATH $\leq_m^{\mathsf{P}}$ ExistSubmodel(AU) and with Corollary 17 it follows that E-CTL-Submodel(AU) is DelNP-complete. □

**Theorem 40.** E-CTL-Submodel(AR) *is* DelNP-*complete.*

*Proof.* The upper bound follows from Theorem 22 again.

For the lower bound we will reduce from HAMPATH, but this time we need to define a new Kripke model $\mathcal{M}_{\mathsf{AR}}(H)$. Let $G = (V, E)$ be a graph with $n = |V|$ and $H := \langle G, s, t \rangle$ an instance of HAMPATH. Further let $\mathcal{M}_{\mathsf{AR}}(H) := (W, R, \eta, w_s)$ be as follows (see Figure 3.8 for an example):

$$
\begin{aligned}
W &:= \{w_v, \tilde{w}_v, \hat{w}_v, w_{v,i} \mid v \in V, 1 \le i \le n\} \\
R &:= \{(w_v, w_{v,i}), (w_{v,i}, \tilde{w}_v), (\tilde{w}_v, \hat{w}_v) \mid v \in V, 1 \le i \le n\} \\
&\quad \cup \{(\hat{w}_u, w_v) \mid (u,v) \in E \text{ and } u \ne t\} && (t \text{ has no outgoing relation}) \\
&\quad \cup \{(\hat{w}_t, \hat{w}_t)\} && (\text{except to itself}) \\
\eta(w_{v,i}) &:= \{i\} \text{ for } 1 \le i \le n \\
\eta(w_v) &:= \{x, 1, 2, \dots, n\} \\
\eta(\tilde{w}_v) &:= \{y, 1, 2, \dots, n\} \\
\eta(\hat{w}_v) &:= \{x, y\}
\end{aligned}
$$

Further let $\varphi$ be the following $\{\mathsf{AR}\}$-formula

$$\varphi := (((((((y \mathbin{\mathsf{AR}} n) \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y) \mathbin{\mathsf{AR}} n-1) \cdots \mathbin{\mathsf{AR}} 2) \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y) \mathbin{\mathsf{AR}} 1.$$

We will now show that $\langle G, s, t\rangle \mapsto \langle \mathcal{M}_{\mathsf{AR}}(H), \varphi\rangle$ is a valid reduction function. Suppose $\langle G, s, t\rangle$ is an instance of HAMPATH. Further let $\mathcal{M}'_{\mathsf{AR}}(H)$ be a satisfying submodel.

The following claim shows that $\varphi$ and $\mathcal{M}_{\mathsf{AR}}(H)$ force submodels to only contain paths of a certain order.

*Claim* 41. For any path $\pi \in \Pi(\mathcal{M}'_{\mathsf{AR}}(H))$ we have that for all $1 \le i \le n$ exists a $v \in V$ such that

$$\pi[4i-3] = w_v, \pi[4i-2] = w_{v,i}, \pi[4i-1] = \tilde{w}_v \text{ and } \pi[4i] = \hat{w}_v$$

and

$$
\begin{aligned}
M'_{\mathsf{AR}}(H), \pi[4i-3] &\models ((((y \mathbin{\mathsf{AR}} n) \cdots \mathbin{\mathsf{AR}} i+1) \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y) \mathbin{\mathsf{AR}} i \\
M'_{\mathsf{AR}}(H), \pi[4i-1] &\models ((((y \mathbin{\mathsf{AR}} n) \cdots \mathbin{\mathsf{AR}} y) \mathbin{\mathsf{AR}} i+1) \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y \\
M'_{\mathsf{AR}}(H), \pi[4i] &\models ((((y \mathbin{\mathsf{AR}} n) \cdots \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y) \mathbin{\mathsf{AR}} i+1) \mathbin{\mathsf{AR}} x.
\end{aligned}
$$

*Proof.* We proceed by induction on $i$. For the base case $i = 1$, it is clear that $\pi[1] = w_s$, considering that all paths start at the root. Notice that from

$$\mathcal{M}'_{\mathsf{AR}}(H), w_s \models ((((y \mathbin{\mathsf{AR}} n) \cdots \mathbin{\mathsf{AR}} 2) \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y) \mathbin{\mathsf{AR}} 1$$

and

$$\mathcal{M}'_{\mathsf{AR}}(H), w_s \not\models ((((y \mathbin{\mathsf{AR}} n) \cdots \mathbin{\mathsf{AR}} 2) \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y)$$

follows that $\mathcal{M}'_{\mathsf{AR}}(H), \pi[2] \models 1$. By the construction of $\mathcal{M}_{\mathsf{AR}}(H)$ the only world directly after $w_s$ labeled with 1 is $w_{s,1}$, thus we have $\pi[2] = w_{s,1}$. It is then obvious that $\pi[3] = \tilde{w}_s$ and

$\pi[4] = \hat{w}_s$, because $\tilde{w}_s$ is the only successor of $w_{s,1}$ and has only $\hat{w}_s$ as successor.

Now consider how this unravels $\varphi$. First observe that $\mathcal{M}_{\mathsf{AR}}(H), \pi[4] \not\models 1$, it follows that there exists an $i < 4$ such that

$$\mathcal{M}'_{\mathsf{AR}}(H), \pi[i] \models (((y \mathbin{\mathsf{AR}} n) \cdots \mathbin{\mathsf{AR}} 2) \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y.$$

We have already established that $i \neq 1$. Also $i \neq 2$, because $y \notin \eta(\pi[2])(= \eta(w_{s,1}))$, leaving only $i = 3$. So

$$\mathcal{M}'_{\mathsf{AR}}(H), \pi[3] \models (((y \mathbin{\mathsf{AR}} n) \cdots \mathbin{\mathsf{AR}} 2) \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y.$$

must hold. Similarly, since all successors of $\hat{w}_s$ in $\mathcal{M}_{\mathsf{AR}}(H)$ (and thus in all its submodels) are not labeled with $y$ and $\tilde{w}_s$ is not labeled with $x$, it follows that

$$\mathcal{M}'_{\mathsf{AR}}(H), \pi[4] \models (((y \mathbin{\mathsf{AR}} n) \cdots \mathbin{\mathsf{AR}} y) \mathbin{\mathsf{AR}} 2) \mathbin{\mathsf{AR}} x.$$

For the induction step, we have

$$M'_{\mathsf{AR}}(H), \pi[4i] \models ((((y \mathbin{\mathsf{AR}} n) \cdots \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y) \mathbin{\mathsf{AR}} i+1) \mathbin{\mathsf{AR}} x$$

and $\pi[4i] = \hat{w}_v$ as induction hypothesis. Notice that all successors $w_u$ of $\hat{w}_v$ are labeled with $x$, while their successors are not. Thus

$$M'_{\mathsf{AR}}(H), \pi[4i+1] \models ((((y \mathbin{\mathsf{AR}} n) \cdots \mathbin{\mathsf{AR}} i+2) \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y) \mathbin{\mathsf{AR}} i+1$$

with $\pi[4i+1] = w_u$ for some $u \in V$. Note again that this formula can only be true at $\pi[4i+1]$ and not $\pi[4i]$, because $i+1 \notin \eta(\pi[4i])$. Similar to the base case, the only successor of $w_u$ labeled with $i+1$ is $w_{u,i+1}$, therefore $\pi[4i+2] = w_{u,i+1}$. $\pi[4i+3] = \tilde{w}_u$ and $\pi[4i+4] = \hat{w}_u$ follow immediately.

We can again observe the unraveling of the formula.

$$M'_{\mathsf{AR}}(H), \pi[4i+3] \models ((((y \mathbin{\mathsf{AR}} n) \cdots \mathbin{\mathsf{AR}} y) \mathbin{\mathsf{AR}} i+2) \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y$$

must be true because, $y \notin \eta(\pi[4i+2])$ and $i+1 \notin \eta(\pi[4i+4])$. Also

$$M'_{\mathsf{AR}}(H), \pi[4i+4] \models ((((y \mathbin{\mathsf{AR}} n) \cdots \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y) \mathbin{\mathsf{AR}} i+2) \mathbin{\mathsf{AR}} x$$

because $x \notin \eta(\pi[4i+3])$ and no successor of $\hat{w}_u$ has label $y$. □

It follows from Claim 41 that all paths of a submodel satisfying $\varphi$ visit $\geq n$ worlds $w_{v,i}$. Claim 41 also shows that

$$M'_{\mathsf{AR}}(H), w_v \models ((((y \mathbin{\mathsf{AR}} n) \cdots \mathbin{\mathsf{AR}} i+1) \mathbin{\mathsf{AR}} x) \mathbin{\mathsf{AR}} y) \mathbin{\mathsf{AR}} i$$

which forces all successors of $w_v$ to be labeled with $i$. For this to be true $w_{v,i}$ has to be the only successor of $w_v$. From this we can conclude that all $w_{v,i}$ must have different $v$. With $\geq n$ worlds $w_{v,i}$ on any path and $|V| = n$, it follows that all paths visit all worlds $w_v$ once. Notice that by our construction of the Kripke model, world $\hat{w}_t$ is a dead end and therefore must be visited last.

This shows that all satisfying submodels of $\mathcal{M}_{\mathsf{AR}}(H)$ must describe a Hamiltonian path of $G$. The reduction function is computable in polynomial time, since both the model $\mathcal{M}_{\mathsf{AR}}(H)$ and the formula $\varphi$ can be constructed done in polynomial time with respect to the graph $G$. □

**Example 42.** Take the Kripke model shown in Figure 3.8 and the following formula:

$$\varphi = (\cdots (y \mathsf{\ AR\ } 4) \mathsf{\ AR\ } x) \mathsf{\ AR\ } y) \mathsf{\ AR\ } 3) \mathsf{\ AR\ } x) \mathsf{\ AR\ } y) \mathsf{\ AR\ } 2) \mathsf{\ AR\ } x) \mathsf{\ AR\ } y) \mathsf{\ AR\ } 1$$

Figure 3.9 depicts a submodel satisfying $\varphi$.

**Theorem 43.** E-CTL-Submodel(AX) *is* DelNP*-complete.*

*Proof.* The upper bound follows as usual from Theorem 20. For the lower bound again define a new, but this time simpler, Kripke model for a graph $G = (V, E)$ and nodes $s, t \in V$. Let $\mathcal{M}_{\mathsf{AX}}(H) = (W, R, \eta, w_s)$ be a Kripke model, for $H := \langle G, s, t\rangle$, with

$$\begin{aligned} W &:= \{w_v \mid v \in V\} \cup \{w_{end}\} \\ R &:= \{(w_u, w_v) \mid (u, v) \in E \text{ and } u \neq t\} && w_t \text{ has no outgoing relations} \\ &\quad\cup \{(w_t, w_{end}), (w_{end}, w_{end})\} && \text{other than to } w_{end} \\ \eta(w_t) &:= \{t\} \end{aligned}$$

The underlying graph of this model is almost $G$ itself, except that a new world $w_{end}$ is added, which became the only successor of $w_t$ and has only one relation to itself. Figure 3.10 depicts such a model.

Suppose $H$ is an instance of HAMPATH and $n = |V|$. Then let $\mathcal{M}_{\mathsf{AX}}(H)$ be the Kripke model as described above and let

$$\varphi := \mathsf{AX}^{n-1}\, t$$

be an {AX}-formula, where $\mathsf{AX}^k$ donates concatenating the AX operator $k$ times. Further let $\mathcal{M}'_{\mathsf{AX}}(G) \subseteq \mathcal{M}_{\mathsf{AX}}(H)$ be a satisfying submodel.

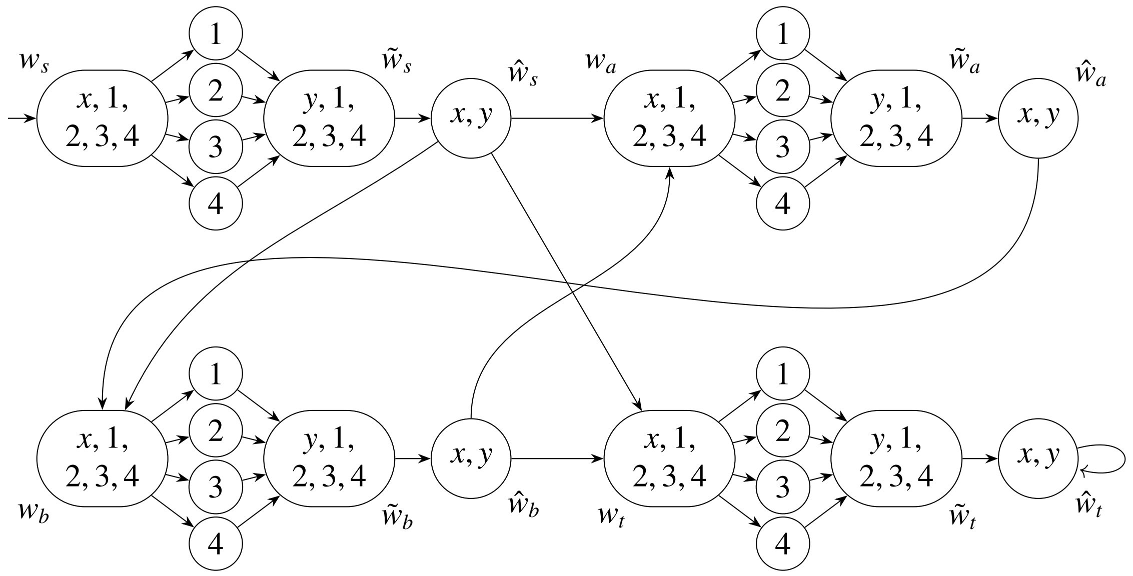


Figure 3.8: Kripke model $\mathcal{M}_{\mathsf{AR}}(H)$ of the graph in Figure 3.5.

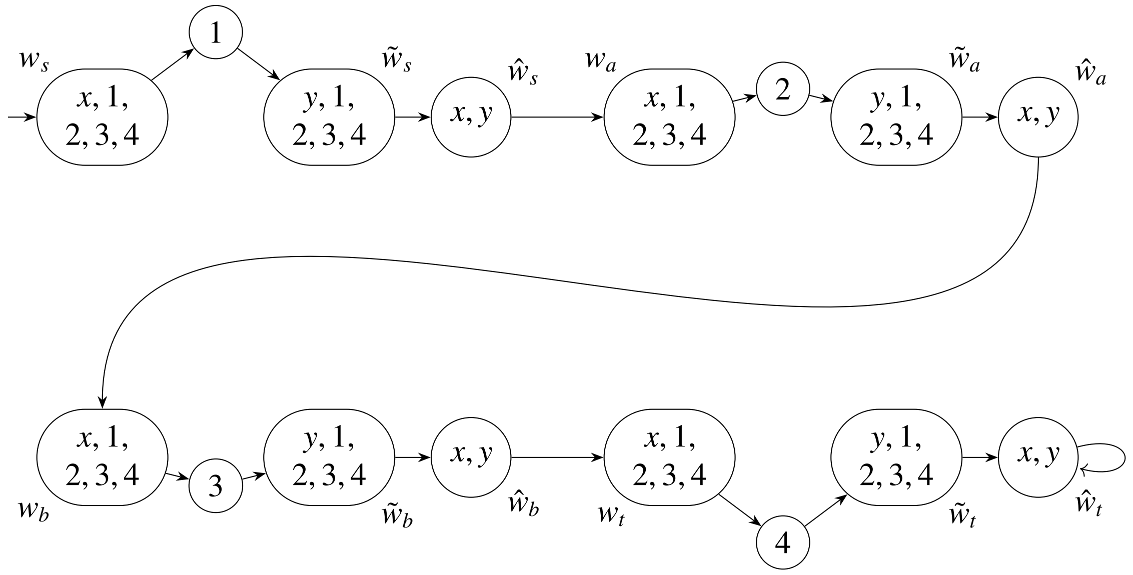


Figure 3.9: Submodel of the Kripke model in Figure 3.8

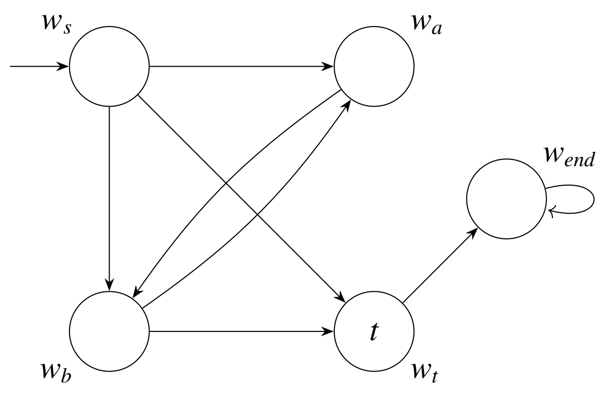


Figure 3.10: Kripke model $\mathcal{M}_{\mathsf{AX}}(H)$ of the graph depicted in Figure 3.5.

We begin by showing, that all $\pi[n] = w_t$ for all paths $\pi \in \Pi(M'_{\mathsf{AX}}(G))$.

$$\begin{aligned}
&M_{\mathsf{AX}}(H), w_s \models \mathsf{AX}^{n-1}\, t \\
&\Leftrightarrow \forall \pi \in \Pi(w_s) : \mathcal{M}_{\mathsf{AX}}(H), \pi[2] \models \mathsf{AX}^{n-2}\, t && \text{(definition AX)} \\
&\Leftrightarrow \forall \pi \in \Pi(w_s) \forall \sigma \in \Pi(\pi[2]) : \mathcal{M}_{\mathsf{AX}}(H), \sigma[2] \models \mathsf{AX}^{n-3}\, t && \text{(definition AX)} \\
&\Leftrightarrow \forall \pi \in \Pi(w_s) : \mathcal{M}_{\mathsf{AX}}(H), \pi[3] \models \mathsf{AX}^{n-3}\, t && \text{(all } \sigma \text{ contained in } \Pi \text{ with a prefix)} \\
&\Leftrightarrow \forall \pi \in \Pi(w_s) : \mathcal{M}_{\mathsf{AX}}(H), \pi[n-1] \models \mathsf{AX}\, t && \text{(repeat this process)} \\
&\Leftrightarrow \forall \pi \in \Pi(w_s) : \mathcal{M}_{\mathsf{AX}}(H), \pi[n] \models t
\end{aligned}$$

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

Now note that $w_t$ cannot be on any path before that. Otherwise the path would continue to $w_{end}$ and get stuck there. Also submodels again cannot have cycles, or else there would be a path that never reaches $w_t$. So we can conclude that on all paths in $\mathcal{M}'_{\mathsf{AX}}(H)$ the first $n$ elements must be different. With $n$ worlds other than $w_end$, this leads to satisfying submodels which are Hamiltonian paths from $w_s$ to $w_t$, showing the correctness of the reduction function as desired.

Also $\langle G, s, t\rangle \mapsto \langle \mathcal{M}_{\mathsf{AX}}(H), \varphi\rangle$ can obviously be computed in polynomial time. □

We conclude this section on clones without Boolean operators by proving that the fragment $\{\mathsf{AF}, \mathsf{AG}\}$ is tractable.

First, we adapted the following Lemma 44 from a result presented by Krebs et al. [13, Lemma 10], showing that every $\{\mathsf{AF}, \mathsf{AG}\}$-formula can be reduced to contain only two temporal operators.

**Lemma 44.** *The following equivalences hold:*

*(1)* $\mathsf{AF}\,\mathsf{AF}\, x \equiv \mathsf{AF}\, x$

*(2)* $\mathsf{AG}\,\mathsf{AG}\, x \equiv \mathsf{AG}\, x$

*(3)* $\mathsf{AG}\,\mathsf{AF}\,\mathsf{AG}\,x \equiv \mathsf{AF}\,\mathsf{AG}\,x$

*(4)* $\mathsf{AF}\,\mathsf{AG}\,\mathsf{AF}\,x \equiv \mathsf{AG}\,\mathsf{AF}\,x$

*Proof.* (1)

$$\begin{aligned}
&\mathcal{M}, w \models \mathsf{AF}\,\mathsf{AF}\,x\\
&\Leftrightarrow \forall \pi \in \Pi(w) \exists k \geq 1 \forall \sigma \in \Pi(\pi[k]) \exists j \geq 1 : \mathcal{M}, \sigma[j] \models x\\
&\Leftrightarrow \forall \pi \in \Pi(w) \exists k \geq 1 \exists j \geq k : \mathcal{M}, \pi[j] \models x\\
&\Leftrightarrow \forall \pi \in \Pi(w) \exists k \geq 1 : \mathcal{M}, \pi[k] \models x\\
&\Leftrightarrow \mathcal{M}, w \models \mathsf{AF}\,x
\end{aligned}$$

(2) analogously.

(3) $\mathcal{M}, w \models \mathsf{AG}\,\mathsf{AF}\,\mathsf{AG}\,x \Rightarrow \mathcal{M}, w \models \mathsf{AF}\,\mathsf{AG}\,x$ is trivial. For the other direction, assume $\mathcal{M}, w \models \mathsf{AF}\,\mathsf{AG}\,x$. Let $\pi \in \Pi(w)$ be an arbitrary path. $\mathcal{M}, \pi[k] \models \mathsf{AG}\,x$ holds for some $k$ with $\mathcal{M}, \pi[i] \not\models \mathsf{AG}\,x$ for all $i < k$. With $\mathcal{M}, \pi[1] \models \mathsf{AF}\,\mathsf{AG}\,x$, it follows that $\mathcal{M}, \pi[i] \models \mathsf{AF}\,\mathsf{AG}\,x$ for all $i < k$. Further take some $\sigma \in \Pi(\pi[k])$. From $\mathcal{M}, \pi[k] \models \mathsf{AG}\,x$ it follows that $\mathcal{M}, \sigma[j] \models \mathsf{AG}\,x$ which leads to $\mathcal{M}, \sigma[j] \models \mathsf{AF}\,\mathsf{AG}\,x$ for all $j \geq 1$. Therefore, we have an infinite path $\rho = \pi[1], \pi[2], \ldots \pi[k-1], \sigma[1](= \pi[k]), \sigma[2]$ with $\mathcal{M}, \rho[i] \models \mathsf{AF}\,\mathsf{AG}\,x$ for all $i \geq 1$. Since $\pi$ and $\sigma$ are arbitrary, this holds for all $\rho \in \Pi(w)$, so $\mathcal{M}, w \models \mathsf{AG}\,\mathsf{AF}\,\mathsf{AG}\,x$.

(4) $\mathcal{M}, w \models \mathsf{AG}\,\mathsf{AF}\,x \Rightarrow \mathcal{M}, w \models \mathsf{AF}\,\mathsf{AG}\,\mathsf{AF}\,x$ is trivial. For the other direction, assume $\mathcal{M}, w \models \mathsf{AF}\,\mathsf{AG}\,\mathsf{AF}\,x$. Now suppose $\mathcal{M}, w \not\models \mathsf{AG}\,\mathsf{AF}\,x$. By the duality of $\mathsf{AG}$ and $\mathsf{AF}$ it follows that $\mathcal{M}, w \models \mathsf{EF}\,\mathsf{EG}\,\neg x$, but this cannot be true without contradicting our assumption. On a path $\pi \in \Pi(w)$ witnessing this there would be a $k \geq 1$ such that for all $i \geq k : \mathcal{M}, \pi[i] \models \neg x$. But this contradicts our assumption that on all path, there would be an $k \geq 1$ such that for all $i \geq k$ there is an $h \geq i : \mathcal{M}, \pi[h] \models x$. We can therefore conclude that $\mathcal{M}, w \models \mathsf{AG}\,\mathsf{AF}\,x$. □

**Theorem 45.** E-CTL-Submodel($\mathsf{AF}, \mathsf{AG}$) *is in* $\mathsf{DelP}$.

*Proof.* We give an algorithm deciding ExtendSubmodel($\mathsf{AF}, \mathsf{AG}$) in polynomial time.

The algorithm gets $\langle \mathcal{M}, \varphi, D \rangle$ as input with, $\mathcal{M} = (W, R, \eta, r)$ a Kripke model, $\varphi$ a $\{\mathsf{AF}, \mathsf{AG}\}$-formula and $D$ a set of deletions. Let $\mathcal{M}' = (W', R', \eta, r) := \mathcal{M} - D$ be current submodel and $\varphi'$ be the shortened formula obtained from $\varphi$ using Lemma 44. Notice that $\varphi'$ can only have one of four forms.

Now, the algorithm has the following behaviour, depending on $\varphi'$:

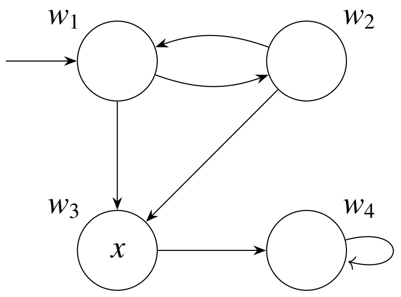
1. If $\varphi' = \mathsf{AF}\,x$, then accept if $\mathcal{M}' \models \mathsf{EF}\,x$, otherwise reject.
2. If $\varphi' = \mathsf{AG}\,x$, then accept if $\mathcal{M}' \models \mathsf{EG}\,x$, otherwise reject.
3. If $\varphi' = \mathsf{AF}\,\mathsf{AG}\,x$, then accept if $\mathcal{M}' \models \mathsf{EF}\,\mathsf{EG}\,x$, otherwise reject.

4. If $\varphi' = \mathsf{AG}\,\mathsf{AF}\,x$, then let $\hat{\mathcal{M}} = (W', R', \hat{\eta}, r)$ be the submodel $\mathcal{M}'$ but with a new labeling function $\hat{\eta}$ defined as
$$\hat{\eta}(w') := \{x_{w'}\}$$
for all $w' \in W'$ with $x \in \eta(w')$.

   Accept if $\hat{\mathcal{M}} \models \bigvee_{w' \in W'} \mathsf{EF}(x_{w'} \wedge \mathsf{EX}\,\mathsf{EF}\,x_{w'})$, otherwise reject.

Correctness of the first two cases is trivial. A path witnessing $\mathsf{EF}\,x$ or $\mathsf{EG}\,x$ induces a submodel, where $\mathsf{AF}\,x$ or $\mathsf{AG}\,x$ holds, respectively. The third case is also quite obvious. If $\mathcal{M}' \models \mathsf{EF}\,\mathsf{EG}\,x$, then there is a path $\pi$ and a $k$ such that $\mathcal{M}', \pi[k] \models \mathsf{EG}\,x$. Let $\sigma$ be the path witnessing $\mathcal{M}', \pi[k] \models \mathsf{EG}\,x$, we than have a path $\rho = \pi[1], \ldots, \pi[k-1], \rho[1](= \pi[k]), \rho[2], \ldots$ which induces a submodel satisfying $\varphi' = \mathsf{AF}\,\mathsf{AG}\,x$.

For $\mathsf{AG}\,\mathsf{AF}\,x$ this approach does not work. Take a look at the following model $\mathcal{M}_0$:



While $\mathcal{M}_0 \models \mathsf{EG}\,\mathsf{EF}\,x$ holds, with $\pi = w_1, w_2, w_1, w_2 \ldots$ as witness, no submodel can satisfy $\mathsf{AG}\,\mathsf{AF}\,x$, because all submodel $\mathcal{M}_0' \subseteq \mathcal{M}_0$ contain $w_4$ and $\mathcal{M}_0', w_4 \not\models \mathsf{AF}\,x$, which means $\mathcal{M}_0' \not\models \mathsf{AG}\,\mathsf{AF}\,x$.

Observer that $\mathsf{AG}\,\mathsf{AF}\,x$ implies that *all* path contain infinitely many worlds where $x$ holds. Since our models are finite it follows that at least on such world must occur on the path infinitely often.

We mimic this property in terms of model checking by first constructing another model $\hat{\mathcal{M}}$, where each world $w' \in W'$ labeled with $x$ gets a new and unique label $x_{w'}$. Secondly, we construct a formula as a disjunction of $\mathsf{EF}(x_{w'} \wedge \mathsf{EX}\,\mathsf{EF}\,x_{w'})$. Notice that this disjunction is true, if and only if the model has at least one cycle containing a world labeled with $x_{w'}$ and thereby a path which contains this worlds infinitely often.

$$\begin{aligned}
&\hat{\mathcal{M}} \models \mathsf{EF}(x_{w'} \wedge \mathsf{EX}\,\mathsf{EF}\,x_{w'})\\
&\Leftrightarrow \exists \pi \in \Pi(\hat{\mathcal{M}}) \exists k \geq 1 : \hat{\mathcal{M}}, \pi[k] \models x_{w'} \text{ and } \hat{\mathcal{M}}, \pi[k] \models \mathsf{EX}\,\mathsf{EF}\,x_{w'}\\
&\Leftrightarrow \exists \pi \in \Pi(\hat{\mathcal{M}}) \exists k \geq 1 : \hat{\mathcal{M}}, \pi[k] \models x_{w'} \text{ and } \hat{\mathcal{M}}, \pi[k+1] \models \mathsf{EF}\,x_{w'}\\
&\Leftrightarrow \exists \pi \in \Pi(\hat{\mathcal{M}}) \exists k \geq 1 : \hat{\mathcal{M}}, \pi[k] \models x_{w'} \text{ and } \exists j \geq k+1 : \hat{\mathcal{M}}, \pi[j] \models x_{w'}\\
&\Leftrightarrow \exists \pi \in \Pi(\hat{\mathcal{M}}) \exists k \geq 1 : \pi[k] = w' \text{ and } \exists j > k : \pi[j] = w'
\end{aligned}$$

It then follows that the path

$$\rho = \pi[1], \dots \pi[k-1], \pi[k], \pi[k+1], \dots, \pi[j-1], \pi[k](= \pi[j]), \pi[k+1], \dots$$

of $\hat{\mathcal{M}}$ induces a satisfying submodel of $\mathcal{M}'$.

Constructing $\varphi'$ and model checking the first three cases can clearly be done in polynomial time. For the fourth case we additionally need to construct a new submodel. But since the size of the new model is identical to the old one, this means it can also be done in polynomial time. The size of the disjunction is linear in the number of worlds of the submodel. Its construction and the model checking can therefore be done in polynomial time. □

Let us illustrate the behaviour of the algorithm with an example.

**Example 46.** Let $\mathcal{M}$ be the Kripke model depicted in Figure 3.11. Further let

$$\varphi := \mathsf{AF}\,\mathsf{AG}\,\mathsf{AG}\,\mathsf{AF}\,x.$$

We now call the algorithm from Theorem 45 on the input $\langle \mathcal{M}, \varphi, \emptyset \rangle$.

The first step is to trim $\varphi$. Note that with (1) from Lemma 44 $\varphi \equiv \mathsf{AF}\,\mathsf{AG}\,\mathsf{AF}\,x$ and with (2) $\mathsf{AF}\,\mathsf{AG}\,\mathsf{AF}\,x \equiv \mathsf{AG}\,\mathsf{AF}\,x =: \varphi'$. So we proceed as follows.

First we construct the model $\hat{\mathcal{M}}$ (see Figure 3.12) and the formula

$$\psi := \mathsf{EF}(x_{w_2} \wedge \mathsf{EX}\,\mathsf{EF}\,x_{w_2}) \vee \mathsf{EF}(x_{w_3} \wedge \mathsf{EX}\,\mathsf{EF}\,x_{w_3}).$$

The algorithm then uses a model checking algorithm to test whether $\hat{\mathcal{M}} \models \psi$. The model checking algorithm will return true, so our algorithm accepts.

The model induced by a path witnessing $\psi$ can be seen in Figure 3.13, also notice that this model obviously satisfies $\varphi'$ and thereby $\varphi$.

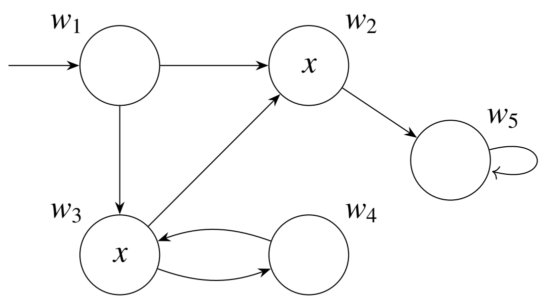


Figure 3.11: Kripke model $\mathcal{M}$ used in Example 46

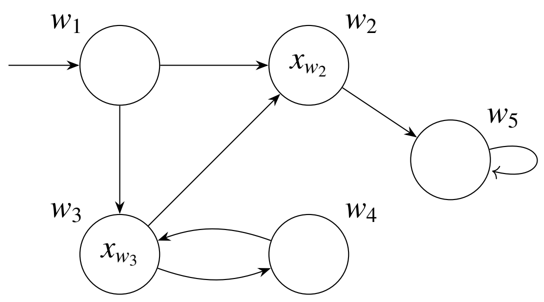


Figure 3.12: Kripke model $\hat{\mathcal{M}}$. Identical to fig. 3.11, but with different labels.

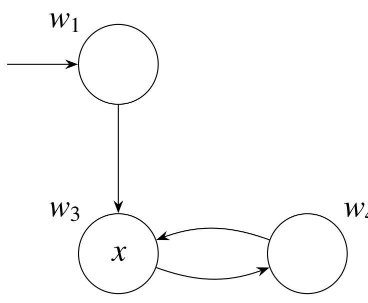


Figure 3.13: Induced Kripke model of a path witnessing $\mathsf{EF}(x_{w_3} \wedge \mathsf{EX}\,\mathsf{EF}\, x_{w_3})$ in Figure 3.12.

## 3.6 Results for formulas with only negation

In this last section we will transfer the previous results onto the clone with only negation. To achieve this we make use of the equivalences between $\mathsf{CTL}$-operators shown in Observation 4.

The fragments $\{\mathsf{AU}, \neg\}$, $\{\mathsf{AR}, \neg\}$ and $\{\mathsf{AX}, \neg\}$ are obviously $\mathsf{DelNP}$-complete. But the addition of negation allows us to show the hardness of further fragments.

The following corollary, extends the results of Theorem 39, 40 and 43.

**Corollary 47.** *The enumeration problems*

*(1)* E-CTL-Submodel($\mathsf{ER}, \neg$),

*(2)* E-CTL-Submodel($\mathsf{EU}, \neg$) *and*

*(3)* E-CTL-Submodel($\mathsf{EX}, \neg$)

*are* $\mathsf{DelNP}$*-complete.*

*Proof.* (1) We have that $\varphi \mathsf{AU} \psi \equiv \neg(\neg\varphi \mathsf{EU} \neg\psi)$ by Observation 4. Thus any $\{\mathsf{AU}\}$-formula can be expressed using $\mathsf{ER}$ and $\neg$, giving rise to a simple reduction from E-CTL-Submodel($\mathsf{AR}$) to E-CTL-Submodel($\mathsf{ER}, \neg$).
(2) Similar to (1), we have that $\varphi\ \mathsf{AR}\ \psi \equiv \neg(\neg\varphi\ \mathsf{EU}\ \neg\psi)$.
(3) Analogously with $\mathsf{AX}\,\varphi = \neg\,\mathsf{EX}\,\neg\varphi$. □

# 4 Conclusion

## 4.1 Summary

In this thesis we introduced a new enumeration problem E-CTL-Submodel, which is the problem of enumerating all submodels of a Kripke model that satisfy a CTL formula. We started by showing its intractability, or more precisely DelNP-completeness, in Corollary 23, with a straightforward reduction from SAT, the satisfiability problem of propositional logic. This unfortunate result motivated us to consider restrictions on the CTL formulas used, in the hope of finding tractable fragments.

To do this, we started by simply disallowing the negation operator, which led to the monotone Boolean clone. Here we proved DelP membership for E-CTL-Submodel(EX, EF, EG, EU, ER, $\wedge$, $\vee$), our first tractability result, in Theorem 26. Unfortunately, Theorem 29 and Corollary 30 show that all monotone formulas containing AX, AF, AG, AU or AR are intractable. We were again able to encode SAT, this time using only A quantified temporal operator and without negation.

So we further restricted the Boolean operators. Theorem 35 shows DelNP-completeness for E-CTL-Submodel(AF, $\wedge$). In this proof we turned to HAMPATH, another NP-complete problem, to reduce from. By constructing special Kripke models and {AF, $\wedge$}-formulas, we were able to relate the existence of submodels to the existence of Hamiltonian paths in a given graph.

Afterwards we used the same approach to show intractability for E-CTL-Submodel(AU) (see Theorem 39), E-CTL-Submodel(AR) (see Theorem 39) and E-CTL-Submodel(AX) (see Theorem 39). These results show that the enumeration problem E-CTL-Submodel is inherently hard for A quantified temporal operators and thus for most fragments of CTL. This gets even worse for the Boolean clone with negation. Corollary 47 proves DelNP-completeness for the fragments E-CTL-Submodel(ER, $\neg$), E-CTL-Submodel(EU, $\neg$) and E-CTL-Submodel(EX, $\neg$), wich makes almost all fragments with negation being intractable.

Table 4.1 summarizes the complexity results for some of the more interesting fragments of CTL.

| Operators | *id* | ¬ | ∧ | ∨ | ∨, ∧ | ⊕ | ∧, ⊕ |
|---|---|---|---|---|---|---|---|
| EG | DelP | | DelP | DelP | DelP | | DelNP-c |
| EF | DelP | | DelP | DelP | DelP | | DelNP-c |
| EX | DelP | DelNP-c | DelP | DelP | DelP | DelNP-c | DelNP-c |
| EU | DelP | DelNP-c | DelP | DelP | DelP | DelNP-c | DelNP-c |
| ER | DelP | DelNP-c | DelP | DelP | DelP | DelNP-c | DelNP-c |
| EX, EU, ER | DelP | DelNP-c | DelP | DelP | DelP | DelNP-c | DelNP-c |
| AG | DelP | | | | DelNP-c | | DelNP-c |
| AF | DelP | | DelNP-c | | DelNP-c | | DelNP-c |
| AG, AF | DelP | | DelNP-c | | DelNP-c | | DelNP-c |
| AX | DelNP-c | DelNP-c | DelNP-c | DelNP-c | DelNP-c | DelNP-c | DelNP-c |
| AU | DelNP-c | DelNP-c | DelNP-c | DelNP-c | DelNP-c | DelNP-c | DelNP-c |
| AR | DelNP-c | DelNP-c | DelNP-c | DelNP-c | DelNP-c | DelNP-c | DelNP-c |

Table 4.1: Enumeration complexity results for fragments of E-CTL-Submodel. Here, DelP means membership in DelP and DelNP-c means DelNP-completeness. If an entry is empty, it means that we currently have no concrete result, besides membership in DelNP which is true for all fragments.

## 4.2 Outlook

With the hardness results for the fragments E-CTL-Submodel(AU), E-CTL-Submodel(AR) and E-CTL-Submodel(AX) propagating upwards in the lattice of Boolean clones, we have classified the enumeration problem of enumerating satisfying submodels for most fragments of CTL as intractable. But some gaps still remain to be filled. Some notable fragments with no current results are:

- E-CTL-Submodel(AG, ∧) and E-CTL-Submodel(AG, ∨),
- E-CTL-Submodel(AF, ∨),
- E-CTL-Submodel(AF, AG, EF, EG, ¬),
- E-CTL-Submodel(AG, EX, EU, ER) and
- E-CTL-Submodel(AF, EX, EU, ER)

It seems reasonable to assume that most will have intractability results, given the apparent hardness of submodel enumeration in general. However, e.g. AG in of itself does not allow much leeway on satisfying submodels, which may make it easier to find them, and may give rise to a DelP algorithm.

Also, while we have proven both, membership in DelNP and DelNP-hardness, i.e. DelNP-completeness for many fragments, the same is not true for our DelP results, where we have only shown membership. CTL operators have different complexity results when considering model checking [13], making it plausible to find better upper bounds for fragments of submodel enumeration without access to all Boolean functions. Enumeration classes defined by circuits [5] may be useful in this context.

Another approach to a better understanding of submodel enumeration might be parameterised enumeration [8, 7, 15], with possible parameters like the depth of the formula or the treewidth of the Kripke model [9, Cha. 7].

# Bibliography

[1] Elmar Bohler, Nadia Creignou, Steen Reith, and Heribert Vollmer. Playing with boolean blocks, part i: Post's lattice with applications to complexity theory1. *SIGACT News*, 34, 01 2003.

[2] Edmund M. Clarke. The birth of model checking. In *25 Years of Model Checking*, volume 5000 of *Lecture Notes in Computer Science*, pages 1–26. Springer, 2008.

[3] Edmund M. Clarke, Orna Grumberg, Daniel Kroening, Doron A. Peled, and Helmut Veith. *Model checking, 2nd Edition*. MIT Press, 2018.

[4] Stephen A. Cook. The complexity of theorem-proving procedures. In Michael A. Harrison, Ranan B. Banerji, and Jeffrey D. Ullman, editors, *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing, May 3-5, 1971, Shaker Heights, Ohio, USA*, pages 151–158. ACM, 1971.

[5] Nadia Creignou, Arnaud Durand, and Heribert Vollmer. Enumeration classes defined by circuits. In Stefan Szeider, Robert Ganian, and Alexandra Silva, editors, *47th International Symposium on Mathematical Foundations of Computer Science, MFCS 2022, August 22-26, 2022, Vienna, Austria*, volume 241 of *LIPIcs*, pages 38:1–38:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[6] Nadia Creignou, Markus Kröll, Reinhard Pichler, Sebastian Skritek, and Heribert Vollmer. A complexity theory for hard enumeration problems. *Discret. Appl. Math.*, 268:191–209, 2019.

[7] Nadia Creignou, Raïda Ktari, Arne Meier, Julian-Steffen Müller, Frédéric Olive, and Heribert Vollmer. Parameterised enumeration for modification problems. *Algorithms*, 12(9):189, 2019.

[8] Nadia Creignou, Arne Meier, Julian-Steffen Müller, Johannes Schmidt, and Heribert Vollmer. Paradigms for parameterized enumeration. *Theory Comput. Syst.*, 60(4):737–758, 2017.

[9] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

[10] Michael J. Fischer and Richard E. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18(2):194–211, 1979.

[11] Nicolas Fröhlich and Arne Meier. Submodel enumeration of kripke structures in modal logic. In David Fernández-Duque, Alessandra Palmigiano, and Sophie Pinchinat, editors, *Advances in Modal Logic, AiML 2022, Rennes, France, August 22-25, 2022*, pages 391–406. College Publications, 2022.

[12] Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972.

[13] Andreas Krebs, Arne Meier, and Martin Mundhenk. The model checking fingerprints of CTL operators. *Acta Informatica*, 56(6):487–519, 2019.

[14] Saul Kripke. Semantical considerations on modal logic. *Acta Philosophica Fennica*, 16:83–94, 1963.

[15] Arne Meier. *Parametrised enumeration*. Habilitation thesis, Leibniz Universität Hannover, 2020.

[16] Amir Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977*, pages 46–57. IEEE Computer Society, 1977.

[17] Amir Pnueli. The temporal semantics of concurrent programs. *Theor. Comput. Sci.*, 13:45–60, 1981.

[18] Philippe Schnoebelen. The complexity of temporal logic model checking. In Philippe Balbiani, Nobu-Yuki Suzuki, Frank Wolter, and Michael Zakharyaschev, editors, *Advances in Modal Logic 4, papers from the fourth conference on "Advances in Modal logic," held in Toulouse, France, 30 September - 2 October 2002*, pages 393–436. King's College Publications, 2002.

[19] A. Prasad Sistla and Edmund M. Clarke. The complexity of propositional linear temporal logics. *J. ACM*, 32(3):733–749, 1985.

[20] Moshe Y. Vardi and Larry J. Stockmeyer. Improved upper and lower bounds for modal logics of programs: Preliminary report. In Robert Sedgewick, editor, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 240–251. ACM, 1985.