TECHNISCHE UNIVERSITÄT
KAISERSLAUTERN

Analysis of a Tableau-based Decision Procedure for CTL*

Bachelor Thesis

of

Geri Gokaj

April 13, 2021

Technische Universität Kaiserslautern,
Department of Computer Science,
67653 Kaiserslautern,
Germany

Examiner: Prof. Dr. Klaus Schneider
M.Sc. Martin Köhler

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die von mir vorgelegte Arbeit mit dem Thema "Analysis of a Tableau-based Decision Procedure for CTL*" selbstständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Kaiserslautern, den 13.4.2021

Geri Gokaj

Abstract

The problem of deciding whether a CTL* formula is satisfiable has been shown to be in 2-EXPTIME [YS85]. Tableau oriented approaches have been used to decide many logics including the temporal logic CTL*. This thesis will analyze a particular tableau oriented decision procedure. It will formally prove the correctness of the tableau rules and using an equivalent approach with alternating tree automata, upper and lower bounds for the runtime of the procedure will be shown. Concluding, a family of worst case examples are stated and the corresponding worst case runtime is shown to be correct.

Zusammenfassung

Es wurde gezeigt, dass das Problem zu entscheiden, ob eine gegebene CTL* Formel erfüllbar ist, in 2-EXPTIME liegt[YS85]. Tableau-basierte Verfahren werden zum Entscheiden vieler Logiken, inklusive CTL* benutzt. Diese Arbeit wird ein bestimmtes Tableauverfahren analysieren. Es wird ein formaler Beweis für die Korrektheit erbracht und anhand eines äquivalenten Verfahrens, das alternierende Baumautomaten nutzt, werden untere und obere Schranken für die Laufzeit gezeigt. Anschließend wird eine Familie von worst-Case Beispielen erläutert und die entsprechende worst-case Laufzeit bewiesen.

Contents

List of Figures

1 Introduction

1.1 Problem statement

The logic CTL* was developed by Emerson and Halpern in 1983 as a natural extension of CTL(Computational Tree Logic) and LTL(Linear Time Logic). It is a temporal logic, expressing logical propositions with respect to a given timeframe. In classical propositional logic, a formula was said to be satisfiable if and only if there exists a boolean assignment of the variabes that fulfills the formula. The assignments in the logic CTL* will not be boolean assignments but rather a graph like structure called **Kripke Structure** usually denoted by $\mathcal{K}$. See below an example of a Kripke structure. Formally $\mathcal{K} \models \mathsf{EG}a$ holds as there exists an infinite path starting at the initial state (doubly circled state) where in each state (globally) the property a holds. We formally define the following decision problems and give analogies to classical propositional logic.

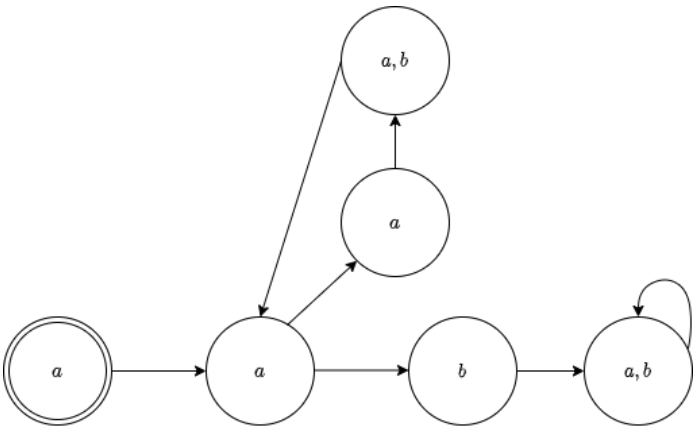

Figure 1.1: *A Kripke Structure $\mathcal{K}$ satsifying* $\mathsf{EG}a$.

Model Checking Problem of CTL*

Input: A CTL* formula φ and a Kripke structure $\mathcal{K}$.

Question: Does $\mathcal{K}$ fulfill φ, or formally $\mathcal{K} \models \varphi$?

In classical propositional logic, the equivalent model checking problem was easy to solve. We would just plug in the assignment in the formula and evaluate. In CTL* this problem is more complex. We mention this problem,

as it is needed to understand the main problem of this thesis, captured by the following decision problem. It will not be relevant for the rest of this thesis.

Satisfiability Problem of CTL*

Input: A CTL* formula φ.

Question: Does a Kripke Structure $\mathcal{K}$ with $\mathcal{K} \models \varphi$ exist?

Firstly, observe the analogy to the satisfiability problem of propositional logic. To solve the satisfiability problem for propositional logic, tableau-based decision procedures have been used. Tableau procedures are based on a set of rules. These rules are applied exhaustively on the formulas and new generated subformulas until no rule is left applicable. Tableau procedures are intuitive and by design easy to implement as it will just basically be an application of rules. The tableau procedure we analyze, will also gives us a fulfilling Kripke Structure, in case one actually exists. In this thesis we will have a look at a tableau procedure for the logic CTL* and in particular analyze its worst case behavior.

1.2 Related Work

There are many approaches for CTL* decision procedures. [FL10] states 3 main decision procedures which are state of the art, which we will discuss in the following.

- Emerson et. al
- Reynolds Tableaux
- Infinite Tableaux

Emerson et. al. [ES84b] [EJ99]

Emerson et. al. use an approach based on automata. It is similar to the approach discussed in this thesis. They convert the given CTL* formula into a special normal form. Now they build a tree automaton which will essentially accept all Kripke Structures satisfying the given formula. To check whether one exists, one finally makes an emptiness check on this tree automaton. This is usually done by a reduction to parity games.

Reynolds Tableaux [Rey09]

Reynolds approach is somewhat different. The nodes of the tableau, represent a set of sets in which the inner sets are called hues. A set of hues is then called a color. A hue is a set describing a closure of subformulas. This closure will describe a path in a model Kripke Structure. In order to be consistent, it is

required that a successor nodes hues are a continuation of the parent nodes hue. Some techniques are used to keep the tableau finite. At the end, a given formula φ is satisfiable if and only if we can construct a consistent tableau.

Infinite tableaux [FLL10]

[FLL10] uses another more sophisticated method that is a tableau based procedure with inference rules, after the construction of the tableau automata are used to perform a check whether the tableau is correctly formed. Most of this procedure is discussed in this thesis although in an altered fashion.

1.3 Contribution

Given the tableau procedure, the aim of this thesis was to find worst case examples. Along the way towards the main theorem, this thesis is supplemented by smaller results such as the formal proofs of the tableau procedure and normal forms. Lower and upper-bounds are stated for most functions and proven. Finally the worst case example is stated and a formal proof is given.

1.4 Outline

In Chapter 2 we will firstly have a look at the syntax of CTL* and basic results about the negation normal form and disjunctive normal form in the propositional logic. After that we introduce the tableau procedure which we will analyze, the rules are proven to be correct. We continue by introducing a useful tool the Unroll function. In Chapter 3 we look at an alternate view of this tableau procedure, as a type of equation systems. As this procedure will be more elegant and clear, this will let us state candidates of worst case examples. Concluding the thesis, we see a worst case example and a formal proof.

2 Preliminaries

We start off this thesis by having a look at the basics of temporal logic. We will later on use these definitions as building blocks and tools for new definitions and algorithms.

2.1 Temporal logics basics

Definition 1 (Syntax of CTL^*) *For a given set of variables $\mathcal{V}$, the set of CTL^* formulas is recursively defined as follows:*

1. *every variable $x \in \mathcal{V}$ is a formula*
2. $\neg\varphi \in \mathsf{CTL}^*$ *provided that* $\varphi \in \mathsf{CTL}^*$
3. $\varphi \wedge \psi \in \mathsf{CTL}^*$ *provided that* $\varphi \in \mathsf{CTL}^*$ *and* $\psi \in \mathsf{CTL}^*$
4. $\varphi \vee \psi \in \mathsf{CTL}^*$ *provided that* $\varphi \in \mathsf{CTL}^*$ *and* $\psi \in \mathsf{CTL}^*$
5. $\mathsf{X}\varphi \in \mathsf{CTL}^*$ *provided that* $\varphi \in \mathsf{CTL}^*$
6. $[\varphi \,\underline{\mathsf{U}}\, \psi] \in \mathsf{CTL}^*$ *provided that* $\varphi \in \mathsf{CTL}^*$ *and* $\psi \in \mathsf{CTL}^*$
7. $[\varphi \,\mathsf{B}\, \psi] \in \mathsf{CTL}^*$ *provided that* $\varphi \in \mathsf{CTL}^*$ *and* $\psi \in \mathsf{CTL}^*$
8. $\mathsf{E}\varphi \in \mathsf{CTL}^*$ *provided that* $\varphi \in \mathsf{CTL}^*$
9. $\mathsf{A}\varphi \in \mathsf{CTL}^*$ *provided that* $\varphi \in \mathsf{CTL}^*$

The sublogic defined by the rules $(1-4)$ will for our purposes, be propositional logic. It is a well known fact that $\vee$, $\wedge$ and $\neg$ operators suffice to simulate any other propositional logic operator.

For our purposes, a formula in LTL is a CTL^* formula of the kind $\mathsf{A}\Phi$, with Φ constructed by the rules $(1-7)$.
Whereas a formula in CTL is a CTL^* formula, with the restriction that every occurrence of $\mathsf{X}\varphi, [\varphi \,\underline{\mathsf{U}}\, \psi]$ and $[\varphi \,\mathsf{B}\, \psi]$ is preceded by an E or A. In order to make a proper analysis, we should firstly define how to capture the meaning of the size of a formula.

Definition 2 (Size of CTL^* Formulas) *The size $|\varphi|$ of a CTL^* formula φ is recursively defined as follows:*

- $|x| = 1$ *for any* $x \in \mathcal{V}$
- $|\neg\varphi| = |\varphi| + 1$
- $|\varphi \wedge \psi| = |\varphi| + |\psi| + 1$
- $|\varphi \vee \psi| = |\varphi| + |\psi| + 1$
- $|\mathsf{X}\varphi| = |\varphi| + 1$
- $|[\varphi \,\underline{\mathsf{U}}\, \psi]| = |\varphi| + |\psi| + 1$

- $|[\varphi\ \mathsf{B}\ \psi]| = |\varphi| + |\psi| + 1$
- $|\mathsf{E}\varphi| = |\varphi| + 1$
- $|\mathsf{A}\varphi| = |\varphi| + 1$

Definition 3 (Negation Normal Form of CTL* Formulas) *For every* CTL* *formula* φ*, we define the following formulas* $\mathsf{NNF}(b, \varphi)$*:*

- $\mathsf{NNF}(\mathsf{false}, x) = x$
- $\mathsf{NNF}(\mathsf{false}, \neg\varphi) = \mathsf{NNF}(\mathsf{true}, \varphi)$
- $\mathsf{NNF}(\mathsf{false}, \varphi \wedge \psi) = \mathsf{NNF}(\mathsf{false}, \varphi) \wedge \mathsf{NNF}(\mathsf{false}, \psi)$
- $\mathsf{NNF}(\mathsf{false}, \varphi \vee \psi) = \mathsf{NNF}(\mathsf{false}, \varphi) \vee \mathsf{NNF}(\mathsf{false}, \psi)$
- $\mathsf{NNF}(\mathsf{false}, \mathsf{X}\varphi) = \mathsf{X}\ \mathsf{NNF}(\mathsf{false}, \varphi)$
- $\mathsf{NNF}(\mathsf{false}, [\varphi\ \underline{\mathsf{U}}\ \psi]) = [\mathsf{NNF}(\mathsf{false}, \varphi)\ \underline{\mathsf{U}}\ \mathsf{NNF}(\mathsf{false}, \psi)]$
- $\mathsf{NNF}(\mathsf{false}, [\varphi\ \mathsf{B}\ \psi]) = [\mathsf{NNF}(\mathsf{false}, \varphi)\ \mathsf{B}\ \mathsf{NNF}(\mathsf{false}, \psi)]$
- $\mathsf{NNF}(\mathsf{false}, \mathsf{E}\varphi) = \mathsf{E}\ \mathsf{NNF}(\mathsf{false}, \varphi)$
- $\mathsf{NNF}(\mathsf{false}, \mathsf{A}\varphi) = \mathsf{A}\ \mathsf{NNF}(\mathsf{false}, \varphi)$

In the previous cases the NNF was only propagated. Now the more interesting cases arise.

- $\mathsf{NNF}(\mathsf{true}, x) = \neg x$
- $\mathsf{NNF}(\mathsf{true}, \neg\varphi) = \varphi$
- $\mathsf{NNF}(\mathsf{true}, \varphi \wedge \psi) = \mathsf{NNF}(\mathsf{true}, \varphi) \vee \mathsf{NNF}(\mathsf{true}, \psi)$
- $\mathsf{NNF}(\mathsf{true}, \varphi \vee \psi) = \mathsf{NNF}(\mathsf{true}, \varphi) \wedge \mathsf{NNF}(\mathsf{true}, \psi)$
- $\mathsf{NNF}(\mathsf{true}, \mathsf{X}\varphi) = \mathsf{X}\ \mathsf{NNF}(\mathsf{true}, \varphi)$
- $\mathsf{NNF}(\mathsf{true}, [\varphi\ \underline{\mathsf{U}}\ \psi]) = [\mathsf{NNF}(\mathsf{true}, \varphi)\ \mathsf{B}\ \mathsf{NNF}(\mathsf{false}, \psi)]$
- $\mathsf{NNF}(\mathsf{true}, [\varphi\ \mathsf{B}\ \psi]) = [\mathsf{NNF}(\mathsf{true}, \varphi)\ \underline{\mathsf{U}}\ \mathsf{NNF}(\mathsf{false}, \psi)]$
- $\mathsf{NNF}(\mathsf{true}, \mathsf{E}\varphi) = \mathsf{A}\ \mathsf{NNF}(\mathsf{true}, \varphi)$
- $\mathsf{NNF}(\mathsf{true}, \mathsf{A}\varphi) = \mathsf{E}\ \mathsf{NNF}(\mathsf{true}, \varphi)$

We need the negation normal form in order to omit some cases later on in the definition of the tableau rules. The following lemma, will show that using the negation normal form does not change the semantics. As noted above only the second half of the cases are interesting.

Lemma 1 *For every* CTL* *formula* φ*, we have* $\varphi \Leftrightarrow \mathsf{NNF}(\mathsf{false}, \varphi)$ *and* $\neg\varphi \Leftrightarrow \mathsf{NNF}(\mathsf{true}, \varphi)$*. Moreover, we have* $|\mathsf{NNF}(b, \varphi)| \leqslant 2\,|\varphi|$ *for any boolean value* b *and any* CTL* *formula* φ*.*

Proof: We first prove the equivalences:

- $\mathsf{NNF}(\mathsf{false}, x) = x$
- $\mathsf{NNF}(\mathsf{false}, \neg\varphi) = \mathsf{NNF}(\mathsf{true}, \varphi) = \neg\varphi$
- $\mathsf{NNF}(\mathsf{false}, \varphi \wedge \psi) = \mathsf{NNF}(\mathsf{false}, \varphi) \wedge \mathsf{NNF}(\mathsf{false}, \psi) = \varphi \wedge \psi$
- $\mathsf{NNF}(\mathsf{false}, \varphi \vee \psi) = \mathsf{NNF}(\mathsf{false}, \varphi) \vee \mathsf{NNF}(\mathsf{false}, \psi) = \varphi \vee \psi$
- $\mathsf{NNF}(\mathsf{false}, \mathsf{X}\varphi) = \mathsf{X}\ \mathsf{NNF}(\mathsf{false}, \varphi) = \mathsf{X}\varphi$

- $\mathsf{NNF}(\mathsf{false}, [\varphi\ \underline{\mathsf{U}}\ \psi]) = [\mathsf{NNF}(\mathsf{false}, \varphi)\ \underline{\mathsf{U}}\ \mathsf{NNF}(\mathsf{false}, \psi)] = [\varphi\ \underline{\mathsf{U}}\ \psi]$
- $\mathsf{NNF}(\mathsf{false}, [\varphi\ \mathsf{B}\ \psi]) = [\mathsf{NNF}(\mathsf{false}, \varphi)\ \mathsf{B}\ \mathsf{NNF}(\mathsf{false}, \psi)] = [\varphi\ \mathsf{B}\ \psi]$
- $\mathsf{NNF}(\mathsf{false}, \mathsf{E}\varphi) = \mathsf{E}\ \mathsf{NNF}(\mathsf{false}, \varphi) = \mathsf{E}\varphi$
- $\mathsf{NNF}(\mathsf{false}, \mathsf{A}\varphi) = \mathsf{A}\ \mathsf{NNF}(\mathsf{false}, \varphi) = \mathsf{A}\varphi$

- $\mathsf{NNF}(\mathsf{true}, x) = \neg x$
- $\mathsf{NNF}(\mathsf{true}, \neg\varphi) = \varphi = \neg\neg\varphi$
- $\mathsf{NNF}(\mathsf{true}, \varphi \wedge \psi) = \mathsf{NNF}(\mathsf{true}, \varphi) \vee \mathsf{NNF}(\mathsf{true}, \psi) = \neg\varphi \vee \neg\psi = \neg(\varphi \wedge \psi)$
- $\mathsf{NNF}(\mathsf{true}, \varphi \vee \psi) = \mathsf{NNF}(\mathsf{true}, \varphi) \wedge \mathsf{NNF}(\mathsf{true}, \psi) = \neg\varphi \wedge \neg\psi = \neg(\varphi \vee \psi)$
- $\mathsf{NNF}(\mathsf{true}, \mathsf{X}\varphi) = \mathsf{X}\mathsf{NNF}(\mathsf{true}, \varphi) = \mathsf{X}\neg\varphi = \neg\mathsf{X}\varphi$
- $\mathsf{NNF}(\mathsf{true}, [\varphi\ \underline{\mathsf{U}}\ \psi]) = [\mathsf{NNF}(\mathsf{true}, \varphi)\ \mathsf{B}\ \mathsf{NNF}(\mathsf{false}, \psi)] = [\neg\varphi\ \mathsf{B}\ \psi] = \neg\,[\varphi\ \underline{\mathsf{U}}\ \psi]$
- $\mathsf{NNF}(\mathsf{true}, [\varphi\ \mathsf{B}\ \psi]) = [\mathsf{NNF}(\mathsf{true}, \varphi)\ \underline{\mathsf{U}}\ \mathsf{NNF}(\mathsf{false}, \psi)] = [\neg\varphi\ \underline{\mathsf{U}}\ \psi] = \neg\,[\varphi\ \mathsf{B}\ \psi]$
- $\mathsf{NNF}(\mathsf{true}, \mathsf{E}\varphi) = \mathsf{A}\ \mathsf{NNF}(\mathsf{true}, \varphi) = \mathsf{A}\neg\varphi = \neg\mathsf{E}\varphi$
- $\mathsf{NNF}(\mathsf{true}, \mathsf{A}\varphi) = \mathsf{E}\ \mathsf{NNF}(\mathsf{true}, \varphi) = \mathsf{E}\neg\varphi = \neg\mathsf{A}\varphi$

Next, we prove the upper bound of the size of the negation normal form. We proceed by a structural induction. The $\vee$ and $\wedge$ cases, E and A cases and the $[\psi\ \mathsf{U}\ \varphi]$ and $[\psi\ \mathsf{B}\ \varphi]$ are similar to each other. It suffices to look at one of them, but for completeness both are proven.

- $|\mathsf{NNF}(\mathsf{false}, x)| = |x| = 1 \leqslant 2 = 2\,|x|$
- $|\mathsf{NNF}(\mathsf{false}, \neg\varphi)| = |\mathsf{NNF}(\mathsf{true}, \varphi)| \leqslant 2\,|\varphi| \leqslant 2(|\varphi| + 1) = 2\,|\neg\varphi|$
- $|\mathsf{NNF}(\mathsf{false}, \varphi \wedge \psi)| = |\mathsf{NNF}(\mathsf{false}, \varphi) \wedge \mathsf{NNF}(\mathsf{false}, \psi)| = |\mathsf{NNF}(\mathsf{false}, \varphi)| + |\mathsf{NNF}(\mathsf{false}, \psi)| + 1 \leqslant 2\,|\varphi| + 2\,|\psi| + 1 \leqslant 2\,|\varphi \wedge \psi|$
- $|\mathsf{NNF}(\mathsf{false}, \varphi \vee \psi)| = |\mathsf{NNF}(\mathsf{false}, \varphi) \vee \mathsf{NNF}(\mathsf{false}, \psi)| = |\mathsf{NNF}(\mathsf{false}, \varphi)| + |\mathsf{NNF}(\mathsf{false}, \psi)| + 1 \leqslant 2\,|\varphi| + 2\,|\psi| + 1 \leqslant 2\,|\varphi \vee \psi|$
- $|\mathsf{NNF}(\mathsf{false}, \mathsf{X}\varphi)| = |\mathsf{X}\ \mathsf{NNF}(\mathsf{false}, \varphi)| \leqslant 2\,|\varphi| + 1 \leqslant 2\,|\mathsf{X}\varphi|$
- $|\mathsf{NNF}(\mathsf{false}, [\varphi\ \underline{\mathsf{U}}\ \psi])| = |[\mathsf{NNF}(\mathsf{false}, \varphi)\ \underline{\mathsf{U}}\ \mathsf{NNF}(\mathsf{false}, \psi)]| = |\mathsf{NNF}(\mathsf{false}, \varphi)| + |\mathsf{NNF}(\mathsf{false}, \psi)| + 1 \leqslant 2\,|\varphi| + 2\,|\psi| + 1 \leqslant 2\,|[\varphi\ \underline{\mathsf{U}}\ \psi]|$
- $|\mathsf{NNF}(\mathsf{false}, [\varphi\ \mathsf{B}\ \psi])| = |[\mathsf{NNF}(\mathsf{false}, \varphi)\ \mathsf{U}\ \mathsf{NNF}(\mathsf{false}, \psi)]| = |\mathsf{NNF}(\mathsf{false}, \varphi)| + |\mathsf{NNF}(\mathsf{false}, \psi)| + 1 \leqslant 2\,|\varphi| + 2\,|\psi| + 1 \leqslant 2\,|[\varphi\ \mathsf{B}\ \psi]|$
- $|\mathsf{NNF}(\mathsf{false}, \mathsf{E}\varphi)| = |\mathsf{E}\ \mathsf{NNF}(\mathsf{false}, \varphi)| = |\mathsf{NNF}(\mathsf{false}, \varphi)| + 1 \leqslant 2\,|\varphi| + 1 \leqslant 2\,|\mathsf{E}\varphi|$
- $|\mathsf{NNF}(\mathsf{false}, \mathsf{A}\varphi)| = |\mathsf{A}\ \mathsf{NNF}(\mathsf{false}, \varphi)| = |\mathsf{NNF}(\mathsf{false}, \varphi)| + 1 \leqslant 2\,|\varphi| + 1 \leqslant 2\,|\mathsf{A}\varphi|$
- $|\mathsf{NNF}(\mathsf{true}, x)| = |\neg x| = |x| + 1 = 2 \leqslant 2\,|x|$
- $|\mathsf{NNF}(\mathsf{true}, \neg\varphi)| = |\varphi| \leqslant 2\,|\varphi| \leqslant 2\,|\neg\varphi|$
- $|\mathsf{NNF}(\mathsf{true}, \varphi \wedge \psi)| = |\mathsf{NNF}(\mathsf{true}, \varphi) \vee \mathsf{NNF}(\mathsf{true}, \psi)| = |\mathsf{NNF}(\mathsf{true}, \varphi)| + |\mathsf{NNF}(\mathsf{true}, \varphi)| + 1 \leqslant 2(|\varphi| + |\psi| + 1) = 2\,|\varphi \wedge \psi|$
- $|\mathsf{NNF}(\mathsf{true}, \varphi \vee \psi)| = |\mathsf{NNF}(\mathsf{true}, \varphi) \wedge \mathsf{NNF}(\mathsf{true}, \psi)| = |\mathsf{NNF}(\mathsf{true}, \varphi)| + |\mathsf{NNF}(\mathsf{true}, \varphi)| + 1 \leqslant 2(|\varphi| + |\psi| + 1) = 2\,|\varphi \vee \psi|$
- $|\mathsf{NNF}(\mathsf{true}, \mathsf{X}\varphi)| = |\mathsf{X}\ \mathsf{NNF}(\mathsf{true}, \varphi)| = 1 + |\mathsf{NNF}(\mathsf{true}, \varphi)| \leqslant 1 + 2\,|\varphi| \leqslant 2(1 + |\varphi|) = 2\,|\mathsf{X}\varphi|$
- $|\mathsf{NNF}(\mathsf{true}, [\varphi\ \underline{\mathsf{U}}\ \psi])| = |[\mathsf{NNF}(\mathsf{true}, \varphi)\ \mathsf{B}\ \mathsf{NNF}(\mathsf{false}, \psi)]| = 1 + |\mathsf{NNF}(\mathsf{true}, \varphi)| + |\mathsf{NNF}(\mathsf{false}, \psi)| \leqslant 2(|\varphi| + 1 + |\psi|) = 2\,|[\psi\ \underline{\mathsf{U}}\ \varphi]|$

- $|\mathsf{NNF}(\mathsf{true}, [\varphi\ \mathsf{B}\ \psi])| = |[\mathsf{NNF}(\mathsf{true}, \varphi)\ \underline{\mathsf{U}}\ \mathsf{NNF}(\mathsf{false}, \psi)]| = 1+|\mathsf{NNF}(\mathsf{true}, \varphi)|+ |\mathsf{NNF}(\mathsf{false}, \psi)| \leqslant 2(|\varphi| + 1 + |\psi|) = 2\,|[\psi\ \mathsf{B}\ \varphi]|$
- $|\mathsf{NNF}(\mathsf{true}, \mathsf{E}\varphi)| = |\mathsf{A}\ \mathsf{NNF}(\mathsf{true}, \varphi)| = 1 + |\mathsf{NNF}(\mathsf{true}, \varphi)| \leqslant 1 + 2\,|\varphi| \leqslant 2(1 + |\varphi|) = 2\,|\mathsf{E}\varphi|$
- $|\mathsf{NNF}(\mathsf{true}, \mathsf{A}\varphi)| = |\mathsf{E}\ \mathsf{NNF}(\mathsf{true}, \varphi)| = 1 + |\mathsf{NNF}(\mathsf{true}, \varphi)| \leqslant 1 + 2\,|\varphi| \leqslant 2(1 + |\varphi|) = 2\,|\mathsf{A}\varphi|$

As seen above, turning a CTL^* formula into NNF is efficient and constrains the appearance of CTL^* formulas. For instance, observe that a formula in NNF can not begin with a negation sign.

As propositional logic is a sublogic of CTL^*, we can consider the procedure on propositional logic only. Based on this, we can now formulate the DNF (disjunctive normal form).

Definition 4 (Cube)
A cube $\mathcal{C}$ is a conjunction of literals (possibly negated variables):

$$\mathcal{C} \equiv \bigwedge_{i=1}^{n} \ell_i$$

Definition 5 (Disjunctive Normal Form)
Let $C_1, \dots C_n$ be cubes.
A propositional logic formula φ is in disjunctive normal form (DNF) *if syntactically it is a disjunction of cubes*

$$\varphi \equiv \bigvee_{i=1}^{n} C_i$$

Denote by $\mathcal{N}(\varphi) := n$ the number of cubes

Definition 6 (Multiplying cubes)
Let $\varphi \equiv \bigvee_{i=1}^{n} C_{\varphi i}$ and $\psi \equiv \bigvee_{j=1}^{m} C_{\psi j}$ be propositional logic formulas in DNF*. Define*

$$\varphi \times \psi := \bigvee_{i,j} C_{\varphi i} \wedge C_{\psi j}$$

For $\varphi_1, \varphi_2, \dots \varphi_n$ in DNF*, we write*

$$\bigtimes_{i=1}^{n} \varphi_i := \varphi_1 \times \varphi_2 \cdots \times \varphi_n$$

We can also denote the DNF *by a set of sets.*
For example $p \wedge \neg q \wedge j \vee \neg p \wedge \neg j$ as $\{\{p, \neg q, j\}, \{\neg p, \neg j\}\}$

We make the crucial observation that $\varphi \times \psi$ is still in DNF. The DNF will play an important role later on in Chapte 3.

Definition 7 (DNF) *Consider a formula φ in negation normal form. The following recursively defined function will turn a formula into* DNF*.*

- $DNF(\ell) := \ell$ *for every literal* ℓ
- $DNF(\varphi \vee \psi) := DNF(\varphi) \vee DNF(\psi)$
- $DNF(\varphi \wedge \psi) := DNF(\varphi) \times DNF(\psi)$

Now that we know how to construct a DNF, let us analyze the runtime in the following:

Lemma 2 (Complexity of DNF) *When turning a formula into* DNF *by our above construction we get that*

$$\mathcal{N}(DNF(\varphi)) \leqslant 2^{|\varphi|}$$

Proof:

Induction Base: $\varphi \equiv \ell$**:**
$\mathcal{N}(DNF(\varphi)) = \mathcal{N}(DNF(\ell)) = 1 \leqslant 2^{|\ell|} = 2^{|\varphi|}$.

Induction step:

- $\mathcal{N}(DNF(\varphi \vee \psi)) = \mathcal{N}(DNF(\varphi) \vee DNF(\psi)) = \mathcal{N}(DNF(\varphi)) + \mathcal{N}(DNF(\psi)) \leqslant 2^{|\varphi|} + 2^{|\psi|} \leqslant 2^{|\varphi|+|\psi|} \leqslant 2^{|\varphi \vee \psi|}$.
- $\mathcal{N}(DNF(\varphi \wedge \psi)) = \mathcal{N}(DNF(\varphi) \times DNF(\psi)) = \mathcal{N}(DNF(\varphi)) \cdot \mathcal{N}(DNF(\psi)) \leqslant 2^{|\varphi|} \cdot 2^{|\psi|} = 2^{|\varphi|+|\psi|} \leqslant 2^{|\varphi \wedge \psi|}$.

Corollary 1 (Exponential length of DNF) *Observe that in the* DNF *construction, the length of a cube is bounded by* $|\varphi|$*, as at most each literal of* φ *can occur in a cube. Thus we get the following result:*

$$|DNF(\varphi))| \leqslant |\varphi|\, 2^{|\varphi|}$$

2.2 The Tableau Procedure

Having the basics out of the way, we can now explain how the tableau procedure works. The high level idea will be the following.
The decompose function unrolls the top-level temporal logic operators inside a CTL* formula, and then computes a type of disjunctive normal form where additionally the path quantifiers are driven as much as possible inwards. This way, the function computes for a given CTL* formula φ *a set of goals* $\mathfrak{G} = \{\mathcal{G}_0, \ldots, \mathcal{G}_{g-1}\}$ where each goal $\mathcal{G}_k$ is a set of *blocks* as defined below: The normal form and the rules are based on [FLL10]. One could also choose an alternative normal form where the A block also consists of conjunctions. Of course the decompose rules would need to change aswell.

Definition 8 (Blocks and Goals of CTL* Formulas) *A block is one of the following three types:*

- *every* CTL* *formula* φ *is a (atomic) block*
- $\mathsf{E}\{\varphi_0, \ldots, \varphi_{n-1}\}$ *with* CTL* *formulas* $\varphi_0, \ldots, \varphi_{n-1}$
- $\mathsf{A}\{\varphi_0, \ldots, \varphi_{n-1}\}$ *with* CTL* *formulas* $\varphi_0, \ldots, \varphi_{n-1}$

A goal $\mathcal{G}_k$ *is a set of blocks* $\mathcal{G}_k = \{\ell_0, \ldots, \ell_{p-1}, \mathsf{E}\Pi_0, \ldots, \mathsf{E}\Pi_{q-1}, \mathsf{A}\Sigma_0, \ldots, \mathsf{A}\Sigma_{r-1}\}$ *with* CTL* *formulas* ℓ_i*, and* E*- and* A*-blocks* $\mathsf{E}\Pi_i$ *and* $\mathsf{A}\Sigma_i$*, respectively, with the following meaning:*

$$[\![\mathcal{G}_k]\!] = \left(\bigwedge_{i=0}^{p-1} \ell_i\right) \wedge \left(\bigwedge_{i=0}^{q-1} \mathsf{E} \bigwedge_{\pi \in \Pi_i} \pi\right) \wedge \left(\bigwedge_{i=0}^{q-1} \mathsf{A} \bigvee_{\sigma \in \Sigma_i} \sigma\right)$$

Moreover, a set of goals $\mathfrak{G} = \{\mathcal{G}_0, \ldots, \mathcal{G}_{g-1}\}$ *has the following meaning:*

$$[\![\mathfrak{G}]\!] = [\![\{\mathcal{G}_0, \ldots, \mathcal{G}_{g-1}\}]\!] = \bigvee_{i=0}^{g-1} [\![\mathcal{G}_i]\!]$$

Before we discuss the computation of the set of goals for a given CTL* formula, let us first estimate an upper bound for the possible goals that can be derived for a given CTL* formula φ. Clearly, φ has no more than $|\varphi|$ many variables, and since each one of them may occur as a positive literal, as a negative literal or not at all in the literal part, we see that there are no more than $3^{|\varphi|}$ possibilities for the literal part of a goal $\mathcal{G}$. Concerning the number of blocks, we first note that the decomposition rules that we define in Definition 9 below will unroll temporal operators so that they are all guarded by an X-operator, and the rules moreover eliminate all top-level operators other than X-operators. Since we moreover consider a negation normal form, the number of blocks is limited by $2^{|\varphi|}$ since each subformula starting with an X-operator may or may not occur in a block and there are at most $2^{|\varphi|}$ many of such subformulas in φ. Hence, there are also $2^{|\varphi|}$ many E- and A-blocks, so that any subset of them can be chosen to form a goal. Hence, we finally obtain the following upper bound:

Lemma 3 (Upper Bound for Number of Decomposed Goals) *For any* CTL* *formula* φ*, the number of goals that can be constructed by its literals and* X*-subformulas is no more than*

$$3^{|\varphi|} \cdot 2^{2^{|\varphi|}} \cdot 2^{2^{|\varphi|}} = 2^{\log(3) \cdot 2^{|\varphi|+1}} \in O\left(2^{2^{|\varphi|}}\right)$$

Hence, in a potential worst-case, we may generate a doubly exponential number of goals for a given CTL* formula φ, but this can only happen if the formula will generate an exponential number of blocks, and this in turn can only happen if the formula contains at least $O(|\varphi|)$ temporal operators which will multiply up by nesting of conjunction and disjunction to $2^{|\varphi|}$ many blocks. We believe this bound is too pessimistic. To start the procedure, we would start with the set of goals $\{\{\varphi\}\}$ if φ starts with a path quantifier and otherwise, we start with $\{\{\mathsf{A}\varphi\}\}$:

Now that we understood what goals are, we can decompose them into other

goals. The rules and the names of them are based on [FLL10]. They are not entirely the same, the paper used typical inference rules of the type

$$\frac{\text{premise}}{\text{conclusion}}$$

whereas we used a function which maps our goals to more goals. Notice that as an underlying data structure we use a set of sets.

Definition 9 (Decomposition of Goals) *The following recursively defined function decomposes a goal $\mathcal{G}$ into an equivalent set of goals $\mathfrak{G} := \mathfrak{D}(\mathcal{G})$:*

- $(\mathsf{E}\ell) : \mathfrak{D}(\{\mathsf{E}(\{\ell\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\ell, \mathsf{E}\mathcal{B}\} \cup \mathcal{G})$ *for any literal* $\ell \in \mathcal{L}$
- $(\mathsf{E}\wedge) : \mathfrak{D}(\{\mathsf{E}(\{\varphi \wedge \psi\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{E}(\{\varphi, \psi\} \cup \mathcal{B})\} \cup \mathcal{G})$
- $(\mathsf{E}\vee) : \mathfrak{D}(\{\mathsf{E}(\{\varphi \vee \psi\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{E}(\{\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G})$
- $(\mathsf{E}[\ \underline{\mathsf{U}}\ ]) : \mathfrak{D}(\{\mathsf{E}(\{[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{E}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\varphi, \mathsf{X}[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})$
- $(\mathsf{E}[\ \mathsf{B}\ ]) : \mathfrak{D}(\{\mathsf{E}(\{[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{E}(\{\varphi, \neg\psi\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\neg\psi, \mathsf{X}[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})$
- $(\mathsf{EE}) : \mathfrak{D}(\{\mathsf{E}(\{\mathsf{E}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{E}\varphi, \mathsf{E}\mathcal{B}\} \cup \mathcal{G})$
- $(\mathsf{EA}) : \mathfrak{D}(\{\mathsf{E}(\{\mathsf{A}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{A}\varphi, \mathsf{E}\mathcal{B}\} \cup \mathcal{G})$
- $(\mathsf{A}\ell) : \mathfrak{D}(\{\mathsf{A}(\{\ell\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\{\ell\} \cup \mathcal{G}\}) \cup \mathfrak{D}(\{\mathsf{A}\mathcal{B}\} \cup \mathcal{G})$ *for any literal* $\ell \in \mathcal{L}$
- $(\mathsf{A}\wedge) : \mathfrak{D}(\{\mathsf{A}(\{\varphi \wedge \psi\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{A}(\{\varphi\} \cup \mathcal{B}), \mathsf{A}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G})$
- $(\mathsf{A}\vee) : \mathfrak{D}(\{\mathsf{A}(\{\varphi \vee \psi\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{A}(\{\varphi, \psi\} \cup \mathcal{B})\} \cup \mathcal{G})$
- $(\mathsf{A}[\ \underline{\mathsf{U}}\ ]) : \mathfrak{D}(\{\mathsf{A}(\{[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{A}(\{\varphi, \psi\} \cup \mathcal{B}), \mathsf{A}(\{\psi, \mathsf{X}[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})$
- $(\mathsf{A}[\ \mathsf{B}\ ]) : \mathfrak{D}(\{\mathsf{A}(\{[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{A}(\{\neg\psi\} \cup \mathcal{B}), \mathsf{A}(\{\varphi, \mathsf{X}[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})$
- $(\mathsf{AE}) : \mathfrak{D}(\{\mathsf{A}(\{\mathsf{E}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\{\mathsf{E}\varphi\}) \cup \mathcal{G}\} \cup \mathfrak{D}(\{\mathsf{A}\mathcal{B}\} \cup \mathcal{G})$
- $(\mathsf{AA}) : \mathfrak{D}(\{\mathsf{A}(\{\mathsf{A}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\{\mathsf{A}\varphi\}) \cup \mathcal{G}\} \cup \mathfrak{D}(\{\mathsf{A}\mathcal{B}\} \cup \mathcal{G})$

Observe that there are no rules for decomposing formulas starting with negations which is justified by first computing a negation normal form of the given CTL^* formulas. We also have to recompute the negation normal form for in the rule of the before operator since its unrolling reintroduces unfortunately a negation. We may use the release operator instead to avoid this, or the weak until operator to replace the before operator before computing negation normal form.

Example To familiarize ourselves with the rules, let us try decompose the example

$$\mathsf{E}(\mathsf{F}a \wedge \mathsf{F}b) = \mathsf{E}([1\ \underline{\mathsf{U}}\ a] \wedge [1\ \underline{\mathsf{U}}\ b])$$

So we start with

$$\{\mathsf{E}(\{[1\ \underline{\mathsf{U}}\ a] \wedge [1\ \underline{\mathsf{U}}\ b]\})\}$$

which apparently is an E block and calculate

$$\mathfrak{D}(\{\mathsf{E}(\{[1 \underline{\mathsf{U}}\, a] \wedge [1 \underline{\mathsf{U}}\, b]\})\})$$

The ($\mathsf{E}\wedge$) rule seems applicable leading to the addition of a set in the E block. We get the equivalent:

$$\mathfrak{D}(\{\mathsf{E}(\{[1 \underline{\mathsf{U}}\, a], [1 \underline{\mathsf{U}}\, b]\})\})$$

Observe now that we have two choices, we can either continue with $[1 \underline{\mathsf{U}}\, a]$ first or with $[1 \underline{\mathsf{U}}\, b]$. This makes the nondeterminism of this procedure clear. We continue with decomposing $[1 \underline{\mathsf{U}}\, a]$ first. So using the ($\mathsf{E}[\, \underline{\mathsf{U}}\,]$) rule we get the equivalent.

$$\mathfrak{D}(\{\mathsf{E}(\{a, [1 \underline{\mathsf{U}}\, b]\})\}) \cup \mathfrak{D}(\{\mathsf{E}(\{1, [1 \underline{\mathsf{U}}\, b], \mathsf{X}[1 \underline{\mathsf{U}}\, a]\})\})$$

Let us firstly decompose the left side of the above, by the ($\mathsf{E}[\, \underline{\mathsf{U}}\,]$) rule we get

$$\begin{aligned}\mathfrak{D}(\{\mathsf{E}(\{a, [1 \underline{\mathsf{U}}\, b]\})\}) &= \mathfrak{D}(\{\mathsf{E}(\{a, b\})\}) \cup \mathfrak{D}(\{\mathsf{E}(\{1, \mathsf{X}[1 \underline{\mathsf{U}}\, b], a\})\}) \\ &= \mathfrak{D}(\{a, b\}) \cup \mathfrak{D}(\{1, a\}) \cup \mathsf{E}(\mathsf{X}[1 \underline{\mathsf{U}}\, b]) \\ &= \{\{a, b\}\} \cup \ \{\{1, a, \mathsf{E}(\mathsf{X}[1 \underline{\mathsf{U}}\, b])\}\} \\ &= \{\{a, b\}, \{a, \mathsf{E}(\mathsf{XF}b)\}\}\end{aligned}$$

Now continue with the right side by first applying the ($\mathsf{E}[\, \underline{\mathsf{U}}\,]$) rule:

$$\begin{aligned}&\mathfrak{D}(\{\mathsf{E}(\{1, [1 \underline{\mathsf{U}}\, b], \mathsf{X}[1 \underline{\mathsf{U}}\, a]\})\}) \\ &\quad = \mathfrak{D}(\{\mathsf{E}(b, 1, \mathsf{X}[1 \underline{\mathsf{U}}\, a])\}) \cup \mathfrak{D}(\{\mathsf{E}(1, 1, \mathsf{X}[1 \underline{\mathsf{U}}\, a], \mathsf{X}[1 \underline{\mathsf{U}}\, b])\}) \\ &\quad = \{\{b, \mathsf{E}(\mathsf{XF}a)\}\} \cup \{\{\mathsf{E}(\mathsf{XF}b), \mathsf{E}(\mathsf{XF}a)\}\} \\ &\quad = \{\{\mathsf{E}(\mathsf{XF}b), \mathsf{E}(\mathsf{XF}a)\}, \{b, \mathsf{E}(\mathsf{XF}a)\}\}\end{aligned}$$

Concluding we get the following goal decomposition

$$\mathfrak{D}(\{\mathsf{E}(\mathsf{F}a \wedge \mathsf{F}b)\}) = \{\{\mathsf{E}(\mathsf{XF}b), \mathsf{E}(\mathsf{XF}a)\}, \{b, \mathsf{E}(\mathsf{XF}a)\}, \{a, b\}, \{a, \mathsf{E}(\mathsf{XF}b)\}\}$$

Apart from negations, only X-operators are missing in the rules to decompose an $\mathsf{E}-$ or A-block in a goal. Hence, the above decomposition rules will reduce any given goal $\mathcal{G}$ into a form where the E- and A-blocks only contain formulas that start with a X-operator. For this reason, the number of possible goals that can be generated in the worst case for a CTL^* formula is limited by Lemma 3. We consider worst-case examples later, and first prove the correctness of the above decomposition rules by showing the semantic of the goals does not change by application of decompose.

Lemma 4 (Correctness of Decomposition) *We have* $[\![\mathfrak{D}(\mathcal{G})]\!] = [\![\mathcal{G}]\!]$ *for any goal* $\mathcal{G}$.

Proof: We prove $[\![\mathfrak{D}(\mathcal{G})]\!] = [\![\mathcal{G}]\!]$ for any goal $\mathcal{G}$ by structural induction on $\mathcal{G}$. The single proof steps given below are quite technical, and are finally based on the following equivalences:

- $(\varphi \vee \beta) \wedge (\psi \vee \beta) \Leftrightarrow \varphi \wedge \psi \vee \beta$
- $\varphi \vee \mathsf{A}\psi \Leftrightarrow \mathsf{A}(\varphi \vee \psi)$ for any state formula φ
- $\varphi \wedge \mathsf{E}\psi \Leftrightarrow \mathsf{E}(\varphi \wedge \psi)$ for any state formula φ
- $\mathsf{A}\varphi \wedge \mathsf{A}\psi \Leftrightarrow \mathsf{A}(\varphi \wedge \psi)$
- $\mathsf{E}\varphi \vee \mathsf{E}\psi \Leftrightarrow \mathsf{E}(\varphi \vee \psi)$
- $[\varphi \,\underline{\mathsf{U}}\, \psi] \Leftrightarrow (\psi \vee \varphi) \wedge (\psi \vee \mathsf{X}\,[\varphi \,\underline{\mathsf{U}}\, \psi])$
- $[\varphi \,\underline{\mathsf{U}}\, \psi] \Leftrightarrow \psi \vee \varphi \wedge \mathsf{X}\,[\varphi \,\underline{\mathsf{U}}\, \psi]$
- $[\varphi \,\mathsf{B}\, \psi] \Leftrightarrow \neg\psi \wedge (\varphi \vee \mathsf{X}\,[\varphi \,\mathsf{B}\, \psi])$
- $[\varphi \,\mathsf{B}\, \psi] \Leftrightarrow \neg\psi \wedge \varphi \vee \neg\psi \wedge \mathsf{X}\,[\varphi \,\mathsf{B}\, \psi])$

Using these equivalences, the structural induction is now done as follows where equivalences marked with IH are based on the induction hypothesis, and equivalences marked with $(*)$ make use of one of the above equivalences:

- $$\begin{aligned}
&[\![\mathfrak{D}(\{\mathsf{E}(\{\ell\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \\
&\quad = [\![\mathfrak{D}(\{\ell, \mathsf{E}\mathcal{B}\} \cup \mathcal{G})]\!] \\
&\quad \overset{(IH)}{=} [\![\{\ell, \mathsf{E}\mathcal{B}\} \cup \mathcal{G}]\!] \\
&\quad = \ell \wedge \mathsf{E}\textstyle\bigwedge \mathcal{B} \wedge \bigwedge \mathcal{G} \\
&\quad \overset{(*)}{=} \mathsf{E}(\ell \wedge \textstyle\bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G} \\
&\quad = [\![\{\mathsf{E}(\{\ell\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]
\end{aligned}$$

- One may think that the $\vee$ and $\wedge$ cases could be similar due to duality. This is not the case here as the duality is not transferred onto the decompose function.

- $$\begin{aligned}
&[\![\mathfrak{D}(\{\mathsf{E}(\{\varphi \wedge \psi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \\
&\quad = [\![\mathfrak{D}(\{\mathsf{E}(\{\varphi, \psi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \\
&\quad \overset{(IH)}{=} [\![\{\mathsf{E}(\{\varphi, \psi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!] \\
&\quad = \mathsf{E}(\varphi \wedge \psi \wedge \textstyle\bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G} \\
&\quad = [\![\{\mathsf{E}(\{\varphi \wedge \psi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]
\end{aligned}$$

- $$\begin{aligned}
&[\![\mathfrak{D}(\{\mathsf{E}(\{\varphi \vee \psi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \\
&\quad = [\![\mathfrak{D}(\{\mathsf{E}(\{\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \\
&\quad = [\![\mathfrak{D}(\{\mathsf{E}(\{\varphi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \;\vee\; [\![\mathfrak{D}(\{\mathsf{E}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \\
&\quad \overset{(IH)}{=} [\![\{\mathsf{E}(\{\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!] \;\vee\; [\![\{\mathsf{E}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!] \\
&\quad = \mathsf{E}(\varphi \wedge \textstyle\bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G} \vee \mathsf{E}(\psi \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G} \\
&\quad \overset{(*)}{=} (\mathsf{E}(\varphi \wedge \textstyle\bigwedge \mathcal{B}) \vee \mathsf{E}(\psi \wedge \bigwedge \mathcal{B})) \wedge \bigwedge \mathcal{G} \\
&\quad \overset{(*)}{=} \mathsf{E}(\varphi \wedge \textstyle\bigwedge \mathcal{B} \vee \psi \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G} \\
&\quad \overset{(*)}{=} \mathsf{E}((\varphi \vee \psi) \wedge \textstyle\bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G} \\
&\quad = [\![\{\mathsf{E}(\{\varphi \vee \psi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]
\end{aligned}$$

- $[\![\mathfrak{D}(\{\mathsf{E}(\{[\varphi \,\underline{\mathsf{U}}\, \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})]\!]$
  $= [\![\mathfrak{D}(\{\mathsf{E}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\varphi, \mathsf{X}\,[\varphi \,\underline{\mathsf{U}}\, \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})]\!]$
  $= [\![\mathfrak{D}(\{\mathsf{E}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \vee [\![\mathfrak{D}(\{\mathsf{E}(\{\varphi, \mathsf{X}\,[\varphi \,\underline{\mathsf{U}}\, \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})]\!]$
  $\overset{(IH)}{=} [\![\{\mathsf{E}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!] \vee [\![\{\mathsf{E}(\{\varphi, \mathsf{X}\,[\varphi \,\underline{\mathsf{U}}\, \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]$
  $= \mathsf{E}(\psi \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G} \vee \mathsf{E}(\varphi \wedge \mathsf{X}\,[\varphi \,\underline{\mathsf{U}}\, \psi] \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G}$
  $\overset{(*)}{=} (\mathsf{E}(\psi \wedge \bigwedge \mathcal{B}) \vee \mathsf{E}(\varphi \wedge \mathsf{X}\,[\varphi \,\underline{\mathsf{U}}\, \psi] \wedge \bigwedge \mathcal{B})) \wedge \bigwedge \mathcal{G}$
  $\overset{(*)}{=} \mathsf{E}(\psi \wedge \bigwedge \mathcal{B} \vee \varphi \wedge \mathsf{X}\,[\varphi \,\underline{\mathsf{U}}\, \psi] \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G}$
  $\overset{(*)}{=} \mathsf{E}((\psi \vee \varphi \wedge \mathsf{X}\,[\varphi \,\underline{\mathsf{U}}\, \psi]) \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G}$
  $\overset{(*)}{=} \mathsf{E}([\varphi \,\underline{\mathsf{U}}\, \psi] \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G}$
  $= [\![\{\mathsf{E}(\{[\varphi \,\underline{\mathsf{U}}\, \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]$

- $[\![\mathfrak{D}(\{\mathsf{E}(\{[\varphi \,\mathsf{B}\, \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})]\!]$
  $= [\![\mathfrak{D}(\{\mathsf{E}(\{\neg\psi, \varphi\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\neg\psi, \mathsf{X}\,[\varphi \,\mathsf{B}\, \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})]\!]$
  $= [\![\mathfrak{D}(\{\mathsf{E}(\{\neg\psi, \varphi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \vee [\![\mathfrak{D}(\{\mathsf{E}(\{\neg\psi, \mathsf{X}\,[\varphi \,\mathsf{B}\, \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})]\!]$
  $\overset{(IH)}{=} [\![\{\mathsf{E}(\{\neg\psi, \varphi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!] \vee [\![\{\mathsf{E}(\{\neg\psi, \mathsf{X}\,[\varphi \,\mathsf{B}\, \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]$
  $= \mathsf{E}(\neg\psi \wedge \varphi \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G} \vee \mathsf{E}(\neg\psi \wedge \mathsf{X}\,[\varphi \,\mathsf{B}\, \psi] \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G}$
  $\overset{(*)}{=} (\mathsf{E}(\neg\psi \wedge \varphi \wedge \bigwedge \mathcal{B}) \vee \mathsf{E}(\neg\psi \wedge \mathsf{X}\,[\varphi \,\mathsf{B}\, \psi] \wedge \bigwedge \mathcal{B})) \wedge \bigwedge \mathcal{G}$
  $\overset{(*)}{=} \mathsf{E}(\neg\psi \wedge \varphi \wedge \bigwedge \mathcal{B} \vee \neg\psi \wedge \mathsf{X}\,[\varphi \,\mathsf{B}\, \psi] \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G}$
  $\overset{(*)}{=} \mathsf{E}((\neg\psi \wedge \varphi \vee \neg\psi \wedge \mathsf{X}\,[\varphi \,\mathsf{B}\, \psi]) \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G}$
  $\overset{(*)}{=} \mathsf{E}([\varphi \,\mathsf{B}\, \psi] \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G}$
  $= [\![\{\mathsf{E}(\{[\varphi \,\mathsf{B}\, \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]$

  The above cases were the most technical to prove. Finally, the last two cases are quite easy and similar.

- $[\![\mathfrak{D}(\{\mathsf{E}(\{\mathsf{E}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!]$
  $= [\![\mathfrak{D}(\{\mathsf{E}\varphi, \mathsf{E}\mathcal{B}\} \cup \mathcal{G})]\!]$
  $\overset{(IH)}{=} [\![\{\mathsf{E}\varphi, \mathsf{E}\mathcal{B}\} \cup \mathcal{G}]\!]$
  $= \mathsf{E}\varphi \wedge \mathsf{E}\bigwedge \mathcal{B} \wedge \bigwedge \mathcal{G}$
  $\overset{(*)}{=} \mathsf{E}(\mathsf{E}\varphi \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G}$
  $= [\![\{\mathsf{E}(\{\mathsf{E}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]$

- $[\![\mathfrak{D}(\{\mathsf{E}(\{\mathsf{A}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!]$
  $= [\![\mathfrak{D}(\{\mathsf{A}\varphi, \mathsf{E}\mathcal{B}\} \cup \mathcal{G})]\!]$
  $\overset{(IH)}{=} [\![\{\mathsf{A}\varphi, \mathsf{E}\mathcal{B}\} \cup \mathcal{G}]\!]$
  $= \mathsf{A}\varphi \wedge \mathsf{E}\bigwedge \mathcal{B} \wedge \bigwedge \mathcal{G}$
  $\overset{(*)}{=} \mathsf{E}(\mathsf{A}\varphi \wedge \bigwedge \mathcal{B}) \wedge \bigwedge \mathcal{G}$
  $= [\![\{\mathsf{E}(\{\mathsf{A}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]$

- For any state formula Φ which is either a literal or of the form $\Phi \equiv \mathsf{E}\varphi$ or $\Phi \equiv \mathsf{A}\varphi$, we have:

$$
\begin{aligned}
&[\![\mathfrak{D}(\{\mathsf{A}(\{\Phi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \\
&\qquad = [\![\mathfrak{D}(\{\{\Phi\} \cup \mathcal{G}\}) \cup \mathfrak{D}(\{\mathsf{A}(\mathcal{B})\} \cup \mathcal{G})]\!] \\
&\qquad = [\![\mathfrak{D}(\{\{\Phi\} \cup \mathcal{G}\})]\!] \ \vee [\![\mathfrak{D}(\{\mathsf{A}(\mathcal{B})\} \cup \mathcal{G})]\!] \\
&\qquad \overset{(IH)}{=} [\![\{\{\Phi\} \cup \mathcal{G}\}]\!] \ \vee [\![\{\mathsf{A}(\mathcal{B})\} \cup \mathcal{G}]\!] \\
&\qquad = \Phi \wedge (\bigwedge \mathcal{G}) \vee \mathsf{A}(\bigvee \mathcal{B}) \wedge (\bigwedge \mathcal{G}) \\
&\qquad = (\Phi \vee \mathsf{A}(\bigvee \mathcal{B})) \wedge (\bigwedge \mathcal{G}) \\
&\qquad = [\![\{\Phi \vee \mathsf{A}(\bigvee \mathcal{B})\} \cup \mathcal{G}]\!] \\
&\qquad \overset{(*)}{=} [\![\{\mathsf{A}(\Phi \vee \bigvee \mathcal{B})\} \cup \mathcal{G}]\!] \\
&\qquad = [\![\{\mathsf{A}(\{\Phi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]
\end{aligned}
$$

and hence, we have for the particular state formulas ℓ, $\mathsf{E}\varphi$, and $\mathsf{A}\varphi$

- $[\![\mathfrak{D}(\{\mathsf{A}(\{\ell\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \quad = [\![\{\mathsf{A}(\{\ell\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]$
- $[\![\mathfrak{D}(\{\mathsf{A}(\{\mathsf{E}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] = [\![\{\mathsf{A}(\{\mathsf{E}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]$
- $[\![\mathfrak{D}(\{\mathsf{A}(\{\mathsf{A}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] = [\![\{\mathsf{A}(\{\mathsf{A}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]$

As the decompose function behaves quite differently for E and A cases, we can not say that it suffices to only look at the E cases. However they are shown with the same tricks and they do not add any new insight.

- $$
\begin{aligned}
&[\![\mathfrak{D}(\{\mathsf{A}(\{\varphi \wedge \psi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \\
&\qquad = [\![\mathfrak{D}(\{\mathsf{A}(\{\varphi\} \cup \mathcal{B}), \mathsf{A}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \\
&\qquad \overset{(IH)}{=} [\![\{\mathsf{A}(\{\varphi\} \cup \mathcal{B}), \mathsf{A}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!] \\
&\qquad = \mathsf{A}(\varphi \vee \bigvee \mathcal{B}) \wedge \mathsf{A}(\psi \vee \bigvee \mathcal{B}) \wedge \bigwedge \mathcal{G} \\
&\qquad \overset{(*)}{=} \mathsf{A}((\varphi \vee \bigvee \mathcal{B}) \wedge (\psi \vee \bigvee \mathcal{B})) \wedge \bigwedge \mathcal{G} \\
&\qquad \overset{(*)}{=} \mathsf{A}(\varphi \wedge \psi \vee \bigvee \mathcal{B}) \wedge \bigwedge \mathcal{G} \\
&\qquad = [\![\{\mathsf{A}(\{\varphi \wedge \psi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]
\end{aligned}
$$

- $$
\begin{aligned}
&[\![\mathfrak{D}(\{\mathsf{A}(\{\varphi \vee \psi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \\
&\qquad = [\![\mathfrak{D}(\{\mathsf{A}(\{\varphi, \psi\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \\
&\qquad \overset{(IH)}{=} [\![\{\mathsf{A}(\{\varphi, \psi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!] \\
&\qquad = [\![\{\mathsf{A}(\varphi \vee \psi \vee \bigvee \mathcal{B})\} \cup \mathcal{G}]\!] \\
&\qquad = [\![\{\mathsf{A}(\{\varphi \vee \psi\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]
\end{aligned}
$$

- $$
\begin{aligned}
&[\![\mathfrak{D}(\{\mathsf{A}(\{[\varphi \ \underline{\mathsf{U}} \ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \\
&\qquad = [\![\mathfrak{D}(\{\mathsf{A}(\{\varphi, \psi\} \cup \mathcal{B}), \mathsf{A}(\{\psi, \mathsf{X}\,[\varphi \ \underline{\mathsf{U}} \ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})]\!] \\
&\qquad \overset{(IH)}{=} [\![\{\mathsf{A}(\{\varphi, \psi\} \cup \mathcal{B}), \mathsf{A}(\{\psi, \mathsf{X}\,[\varphi \ \underline{\mathsf{U}} \ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}]\!] \\
&\qquad = \mathsf{A}(\varphi \vee \psi \vee \bigvee \mathcal{B}) \wedge \mathsf{A}(\psi \vee \mathsf{X}\,[\varphi \ \underline{\mathsf{U}} \ \psi] \vee \bigvee \mathcal{B}) \wedge \bigwedge \mathcal{G} \\
&\qquad \overset{(*)}{=} \mathsf{A}((\varphi \vee \psi \vee \bigvee \mathcal{B}) \wedge (\psi \vee \mathsf{X}\,[\varphi \ \underline{\mathsf{U}} \ \psi] \vee \bigvee \mathcal{B})) \wedge \bigwedge \mathcal{G} \\
&\qquad \overset{(*)}{=} \mathsf{A}((\varphi \vee \psi) \wedge (\psi \vee \mathsf{X}\,[\varphi \ \underline{\mathsf{U}} \ \psi]) \vee \bigvee \mathcal{B}) \wedge \bigwedge \mathcal{G} \\
&\qquad \overset{(*)}{=} \mathsf{A}([\varphi \ \underline{\mathsf{U}} \ \psi] \vee \bigvee \mathcal{B}) \wedge \bigwedge \mathcal{G} \\
&\qquad = [\![\{\mathsf{A}(\{[\varphi \ \underline{\mathsf{U}} \ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]
\end{aligned}
$$

- $[\![\mathfrak{D}(\{\mathsf{A}(\{[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})]\!]$
$= [\![\mathfrak{D}(\{\mathsf{A}(\{\neg\psi\} \cup \mathcal{B}), \mathsf{A}(\{\varphi, \mathsf{X}\,[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G})]\!]$
$\overset{(IH)}{=} [\![\{\mathsf{A}(\{\neg\psi\} \cup \mathcal{B}), \mathsf{A}(\{\varphi, \mathsf{X}\,[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]$
$= \mathsf{A}(\neg\psi \vee \bigvee \mathcal{B}) \wedge \mathsf{A}(\varphi \vee \mathsf{X}\,[\varphi\ \mathsf{B}\ \psi] \vee \bigvee \mathcal{B}) \wedge \bigwedge \mathcal{G}$
$\overset{(*)}{=} \mathsf{A}((\neg\psi \vee \bigvee \mathcal{B}) \wedge (\varphi \vee \mathsf{X}\,[\varphi\ \mathsf{B}\ \psi] \vee \bigvee \mathcal{B})) \wedge \bigwedge \mathcal{G}$
$\overset{(*)}{=} \mathsf{A}(\neg\psi \wedge (\varphi \vee \mathsf{X}\,[\varphi\ \mathsf{B}\ \psi]) \vee \bigvee \mathcal{B}) \wedge \bigwedge \mathcal{G}$
$\overset{(*)}{=} \mathsf{A}([\varphi\ \mathsf{B}\ \psi] \vee \bigvee \mathcal{B}) \wedge \bigwedge \mathcal{G}$
$= [\![\{\mathsf{A}(\{[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}]\!]$

If applied to the special goal $\mathcal{G} = \{\varphi\}$, we obtain a set of goals $\mathfrak{G} := \mathfrak{D}(\{\varphi\})$ which correspond in essence to a disjunctive normal form of the CTL^* formula φ. This disjunctive normal form generates all possible states that may be the root of a tree model that could satisfy the given CTL^* formula φ; no other state could do that since all possibilities are listed in the cubes of a disjunctive normal form. Hence, it yields the initial states of a state transition system that may satisfy the CTL^* formula φ.

Hence, if we are searching a model for a given CTL^* formula φ, we first compute $\mathfrak{D}(\{\varphi\}) = \{\mathcal{G}_0, \ldots, \mathcal{G}_{g-1}\}$. Recall that the E- and A-blocks of each goal $\mathcal{G}_i$ only contain formulas that start with a X-operator. These E- and A-blocks will now define transitions of a potential tree model by the following rules:

Definition 10 (Transitions of Goals) *Given a decomposed goal* $\mathcal{G} = \{\ell_0, \ldots, \ell_{p-1}, \mathsf{E}\Pi_0, \ldots, \mathsf{E}\Pi_{q-1}, \mathsf{A}\Sigma_0, \ldots, \mathsf{A}\Sigma_{r-1}\}$ *where the* E- *and* A-*blocks only contain formulas that start with an* X-*operator. Let* $\Pi_i = \{\mathsf{X}\pi_{i,0}, \ldots, \mathsf{X}\pi_{i,s_i}\}$ *and* $\Sigma_i = \{\mathsf{X}\sigma_{i,0}, \ldots, \mathsf{X}\sigma_{i,t_i}\}$. *We then define* $\Pi'_i = \{\pi_{i,0}, \ldots, \pi_{i,s_i}\}$ *and* $\Sigma'_i = \{\sigma_{i,0}, \ldots, \sigma_{i,t_i}\}$

$$\mathfrak{X}(\mathcal{G}) = \begin{cases} \{\{\mathsf{E}\Pi'_i, \mathsf{A}\Sigma'_0, \ldots, \mathsf{A}\Sigma'_{r-1}\} \mid i = 0, \ldots, q-1\} & : \text{for } q > 0 \\ \{\{\mathsf{A}\Sigma'_0, \ldots, \mathsf{A}\Sigma'_{r-1}\}\} & : \text{for } q = 0 \end{cases}$$

We now focus on the complexity of the above model generation procedure. At first, let us estimate an upper bound for the set of goals that may be derived from a CTL^* formula φ with n variables. Clearly, a goal may have no more than n literals, however, the number of E- and A-blocks is more difficult to estimate. To this end, we consider the following definition of a CTL^* formula $\mathfrak{U}(\varphi)$ which corresponds with the decompositon rules for the temporal logic operators:

Definition 11 (Unrolling CTL^* Formulas) *For every* CTL^* *formula* φ, *we define the following* CTL^* *formula* $\mathfrak{U}(\varphi)$*:*

- $\mathfrak{U}(x) = x$
- $\mathfrak{U}(\neg\varphi) = \neg\mathfrak{U}(\varphi)$
- $\mathfrak{U}(\varphi \wedge \psi) = \mathfrak{U}(\varphi) \wedge \mathfrak{U}(\psi)$
- $\mathfrak{U}(\varphi \vee \psi) = \mathfrak{U}(\varphi) \vee \mathfrak{U}(\psi)$
- $\mathfrak{U}(\mathsf{X}\varphi) = \mathsf{X}\varphi$

- $\mathfrak{U}([\varphi \underline{\mathsf{U}} \psi]) = \mathfrak{U}(\psi) \vee \mathfrak{U}(\varphi) \wedge \mathsf{X}[\varphi \underline{\mathsf{U}} \psi]$
- $\mathfrak{U}([\varphi \mathsf{B} \psi]) = \neg\mathfrak{U}(\psi) \wedge (\mathfrak{U}(\varphi) \vee [\varphi \mathsf{B} \psi])$
- $\mathfrak{U}(\mathsf{E}\varphi) = \mathsf{E}\,\mathfrak{U}(\varphi)$
- $\mathfrak{U}(\mathsf{A}\varphi) = \mathsf{A}\,\mathfrak{U}(\varphi)$

We naturally extend this definition to the blocks for $\Theta \in \{\mathsf{E}, \mathsf{A}\}$:

$$\mathfrak{U}(\Theta\mathcal{B})\{\gamma_1, \gamma_2, \ldots \gamma_n\} := \Theta\mathcal{B}\{\mathfrak{U}(\gamma_1), \mathfrak{U}(\gamma_2), \ldots, \mathfrak{U}(\gamma_n)\}$$

Lemma 5 (Complexity of Unroll) *For any* CTL^* *formula* φ*, we have* $|\mathfrak{U}(\varphi)| \in O(|\varphi|^2)$.

Proof: We proceed by structural induction. Show that $|\mathfrak{U}(\varphi)| \leqslant 10\,|\varphi|^2 + 4$. As usual, the $\vee$ and $\wedge$ cases, the E and A and finally the $[\psi \mathsf{B} \varphi], [\psi \mathsf{U} \varphi]$ are similar.

- $|\mathfrak{U}(x)| = |x| = 1 \leqslant 14 = 10\,|x|^2 + 4$
- $|\mathfrak{U}(\neg\varphi)| = |\neg\mathfrak{U}(\varphi)| = 1 + |\mathfrak{U}(\varphi)| \leqslant 1 + 10\,|\varphi|^2 + 4 \leqslant 10(1 + |\varphi|)^2 + 4 = 10(|\neg\varphi|)^2 + 4$
- $|\mathfrak{U}(\varphi \wedge \psi)| = |\mathfrak{U}(\varphi) \wedge \mathfrak{U}(\psi)| = 1 + |\mathfrak{U}(\varphi)| + |\mathfrak{U}(\psi)| \leqslant 1 + 10\,|\varphi|^2 + 4 + 10\,|\psi|^2 + 4 \leqslant 10(1 + |\varphi| + |\psi|)^2 + 4 = 10\,|\varphi \wedge \psi|^2 + 4$
- $|\mathfrak{U}(\varphi \vee \psi)| = |\mathfrak{U}(\varphi) \vee \mathfrak{U}(\psi)| = 1 + |\mathfrak{U}(\varphi)| + |\mathfrak{U}(\psi)| \leqslant 1 + 10\,|\varphi|^2 + 4 + 10\,|\psi|^2 + 4 \leqslant 10(1 + |\varphi| + |\psi|)^2 + 4 = 10\,|\varphi \vee \psi|^2 + 4$
- $|\mathfrak{U}(\mathsf{X}\varphi)| = |\mathsf{X}\varphi| = 1 + |\varphi| \leqslant 10(1 + |\varphi|)^2 + 4 = 10(|\mathsf{X}\varphi|)^2 + 4$
- $|\mathfrak{U}([\varphi \underline{\mathsf{U}} \psi])| = |\mathfrak{U}(\psi) \vee \mathfrak{U}(\varphi) \wedge \mathsf{X}[\varphi \underline{\mathsf{U}} \psi]| = 1 + |\mathfrak{U}(\psi)| + |\mathfrak{U}(\phi)| + 1 + 4 \leqslant 1 + 10\,|\varphi|^2 + 4 + 10\,|\psi|^2 + 4 + 5 \leqslant 10(1 + |\varphi| + |\psi|)^2 + 4 = 10\,|[\psi \underline{\mathsf{U}} \varphi]|^2 + 4$
- $|\mathfrak{U}([\varphi \mathsf{B} \psi])| = |\neg\mathfrak{U}(\psi) \wedge (\mathfrak{U}(\varphi) \vee [\varphi \mathsf{B} \psi])| = 1 + |\mathfrak{U}(\psi)| + 1 + |\mathfrak{U}(\varphi)| + 1 + 3 \leqslant 1 + 10\,|\psi|^2 + 4 + 1 + 10\,|\varphi|^2 + 4 + 1 + 3 \leqslant 10(1 + |\psi| + |\varphi|)^2 + 4 = 10\,|[\varphi \mathsf{B} \psi]|^2 + 4$
- $|\mathfrak{U}(\mathsf{E}\varphi)| = |\mathsf{E}\,\mathfrak{U}(\varphi)| = 1 + |\mathfrak{U}(\varphi)| \leqslant 1 + 10(|\varphi|)^2 + 4 \leqslant 10(1 + |\varphi|)^2 + 4 = 10(|\mathsf{E}\varphi|)^2 + 4$
- $|\mathfrak{U}(\mathsf{A}\varphi)| = |\mathsf{A}\,\mathfrak{U}(\varphi)| = 1 + |\mathfrak{U}(\varphi)| \leqslant 1 + 10(|\varphi|)^2 + 4 \leqslant 10(1 + |\varphi|)^2 + 4 = 10(|\mathsf{A}\varphi|)^2 + 4$

As a worst case example, where the quadratic blow-up is observed, consider the formulas Φ_n which are recursively defined as follows

- $\Phi_0 := [p_0 \underline{\mathsf{U}} q_0]$
- $\Phi_{n+1} := [p_{n+1} \underline{\mathsf{U}} (q_{n+1} \wedge \Phi_n)]$

We can directly see that $|\Phi_0| = 3$ and $|\Phi_{n+1}| = |\Phi_n| + 4$ so that we have $|\Phi_n| = 4n + 3$ which can be easily proved by induction. Considering the definition of the unrolling function, we have

- $\mathfrak{U}(\Phi_0) := p_0 \vee q_0 \wedge \mathsf{X}[p_0 \underline{\mathsf{U}} q_0]$
- $\mathfrak{U}(\Phi_{n+1}) := p_{n+1} \vee q_{n+1} \wedge \mathfrak{U}(\Phi_n) \wedge \mathsf{X}\Phi_{n+1}$

Hence, we have $|\mathfrak{U}(\Phi_0)| = 8$ and $|\mathfrak{U}(\Phi_{n+1})| = |\mathfrak{U}(\Phi_n)| + |\Phi_{n+1}| + 6 = |\mathfrak{U}(\Phi_n)| + 4n + 13$. It is straighforward to prove by induction that we therefore have $|\mathfrak{U}(\Phi_n)| = 2n^2 + 11n + 8$:

- $|\mathfrak{U}(\Phi_0)| = 8 = 2 \cdot 0^2 + 11 \cdot 0 + 8$
- $|\mathfrak{U}(\Phi_{n+1})| = |\mathfrak{U}(\Phi_n)| + 4n + 13 = 2n^2 + 11n + 8 + 4n + 13 = 2(n+1)^2 + 11(n+1) + 8$

Hence, for our example Φ_n, we have $|\Phi_n| = 4n + 3$ and $|\mathfrak{U}(\Phi_n)| = 2n^2 + 11n + 8$ which proves that the quadratic blow-up can really occur and the above estimation is tight.

Having unrolled a CTL^* formula φ, we next consider the contained variables and subformulas of $\mathfrak{U}(\varphi)$ that start with an X-operator.

First of all, we prove a lower bound for the number of goals that can be derived by single goal $\mathcal{G}$ by computing $\mathfrak{D}(\mathcal{G})$. To this end, we need to define the size $|\mathcal{G}|$ of a goal $\mathcal{G}$ which is simply the sum of the sizes of the contained blocks where the size of a block is the sum of the sizes of the contained formulas.

Lemma 6 (Exponential Complexity of Decomposition) *There exists a goal $\mathcal{G}$ with $|\mathfrak{D}(\mathcal{G})| \in 2^{O(|\mathcal{G}|)}$.*

Proof: Consider the goal:

$$\mathcal{G} := \{\mathsf{E}\{p_1 \vee q_1, p_2 \vee q_2, \ldots, p_n \vee q_n\}\}$$

with the following semantics

$$[\![\mathcal{G}]\!] = \mathsf{E}\left(\bigwedge_{i=1}^{n}(p_i \vee q_i)\right)$$

Notice that $|\mathcal{G}| \in \Theta(n)$. By exhaustively using the decomposition rule:

$$\mathfrak{D}(\{\mathsf{E}(\{\varphi \vee \psi\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{E}(\{\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G})$$

Consider a graphical representation as follows: The starting node is our starting goal, the twofolded branching of a node represents the usage of a decomposition rule. The nodes at the end can not be decomposed any further. As we get a full binary tree, the number of nodes at the end is in $2^{O(n)}$. See Figure 2.1 for further details.

Now that we have an intuition about decompose, we apply the introduced Unroll function to it. Firstly prove by induction that the semantics does not change.

Lemma 7 (Unrolling once) *For any block $\mathsf{E}\{\pi_0, \ldots, \pi_{m-1}\}$ and any block $\mathsf{A}\{\sigma_0, \ldots, \sigma_{n-1}\}$, we have the following*

- $\mathfrak{D}(\{\mathsf{E}\{\pi_0, \ldots, \pi_{m-1}\} \cup \mathcal{G}\}) = \mathfrak{D}(\{\mathsf{E}\{\mathfrak{U}(\pi_0), \ldots, \pi_{m-1}\} \cup \mathcal{G}\})$
- $\mathfrak{D}(\{\mathsf{A}\{\sigma_0, \ldots, \sigma_{n-1}\} \cup \mathcal{G}\}) = \mathfrak{D}(\{\mathsf{A}\{\mathfrak{U}(\sigma_0), \ldots, \sigma_{n-1}\} \cup \mathcal{G}\})$

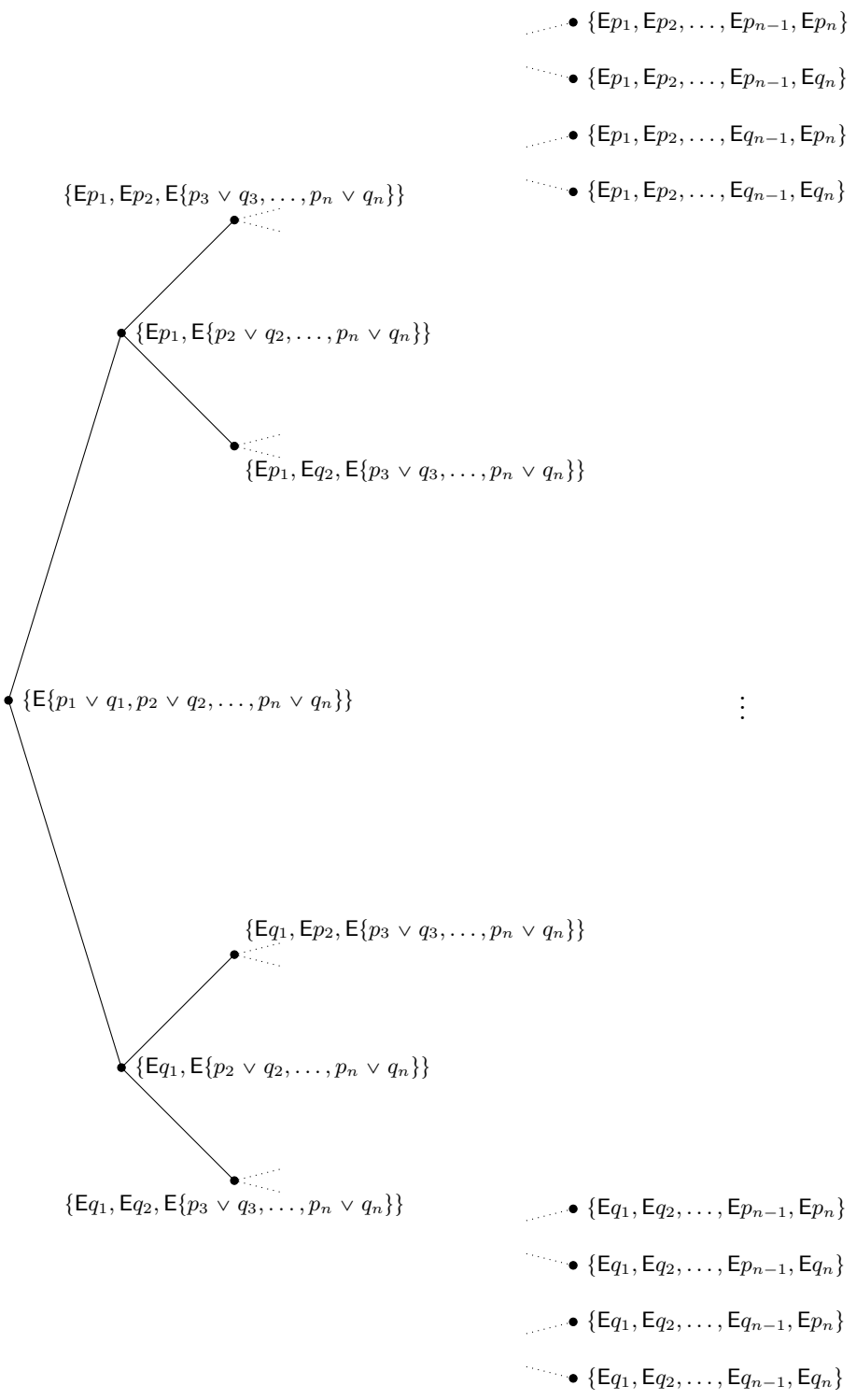

Figure 2.1: *The decompose tableau of* $\{\mathsf{E}\{p_1 \vee q_1, p_2 \vee q_2, \ldots, p_n \vee q_n\}\}$

Proof:

Induction Base:

$$\mathfrak{D}(\{\mathsf{E}(\{x\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(x)\} \cup \mathcal{B})\} \cup \mathcal{G})$$

$$\mathfrak{D}(\{\mathsf{A}(\{x\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(x)\} \cup \mathcal{B})\} \cup \mathcal{G})$$

Induction Step:

$$(\mathsf{E}\ell) : \mathfrak{D}(\{\mathsf{E}(\{\ell\} \cup \mathcal{B})\} \cup \mathcal{G}) = \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\ell)\} \cup \mathcal{B})\} \cup \mathcal{G})$$

Observe that the $(\mathsf{E}\vee)$ and $(\mathsf{E}\wedge)$ wedge cases are quite different. This is due to the fact that the decompose function behaves differently in both those cases.

$$\begin{aligned}(\mathsf{E}\wedge) : \mathfrak{D}(\{\mathsf{E}(\{\varphi \wedge \psi\} \cup \mathcal{B})\} \cup \mathcal{G}) &= \mathfrak{D}(\{\mathsf{E}(\{\varphi, \psi\} \cup \mathcal{B})\} \cup \mathcal{G}) \\ &= \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\varphi), \mathfrak{U}(\psi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \\ &= \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\varphi \wedge \psi\}) \cup \mathcal{B})\} \cup \mathcal{G})\end{aligned}$$

$$\begin{aligned}(\mathsf{E}\vee) : \mathfrak{D}(\{\mathsf{E}(\{\varphi \vee \psi\} \cup \mathcal{B})\} \cup \mathcal{G}) &= \mathfrak{D}(\{\mathsf{E}(\{\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G}) \\ &= \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\varphi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\psi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \\ &= \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\varphi) \vee \mathfrak{U}(\psi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \\ &= \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\varphi \vee \psi)\} \cup \mathcal{B})\} \cup \mathcal{G})\end{aligned}$$

Moving on to the before and until cases.

$$\begin{aligned}(\mathsf{E}[\,\underline{\mathsf{U}}\,]) :&\mathfrak{D}(\{\mathsf{E}(\{[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\ &= \mathfrak{D}(\{\mathsf{E}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\varphi, \mathsf{X}[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\ &= \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\psi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\varphi), \mathsf{X}[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\ &= \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\psi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\varphi) \wedge \mathsf{X}[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\ &= \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\psi) \vee \mathfrak{U}(\varphi) \wedge \mathsf{X}[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\ &= \mathfrak{D}(\{\mathsf{E}\{\mathfrak{U}([\varphi\ \underline{\mathsf{U}}\ \psi])\} \cup \mathcal{B})\} \cup \mathcal{G})\end{aligned}$$

$$
\begin{aligned}
&(\mathsf{E}[\ \mathsf{B}\ ]) : \mathfrak{D}(\{\mathsf{E}(\{[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{E}(\{\varphi, \neg\psi\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\neg\psi, \mathsf{X}[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\varphi), \mathfrak{U}(\neg\psi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\neg\psi), \mathsf{X}[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\varphi) \wedge \mathfrak{U}(\neg\psi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \cup \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\neg\psi) \wedge \mathsf{X}[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{E}(\mathfrak{U}(\varphi) \wedge \mathfrak{U}(\neg\psi) \vee \mathfrak{U}(\neg\psi) \wedge \mathsf{X}[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{E}\{\mathfrak{U}([\varphi\ \mathsf{B}\ \psi])\} \cup \mathcal{B})\} \cup \mathcal{G})
\end{aligned}
$$

Finally the quite similar (EE) and (EA) cases.

$$
\begin{aligned}
&(\mathsf{EE}) : \mathfrak{D}(\{\mathsf{E}(\{\mathsf{E}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{E}\varphi, \mathsf{E}\mathcal{B}\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{E}\mathfrak{U}(\varphi), \mathsf{E}\mathcal{B}\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{E}(\{\mathsf{E}\mathfrak{U}(\varphi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\mathsf{E}\varphi)\} \cup \mathcal{B})\} \cup \mathcal{G})
\end{aligned}
$$

$$
\begin{aligned}
&(\mathsf{EA}) : \mathfrak{D}(\{\mathsf{E}(\{\mathsf{A}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{A}\varphi, \mathsf{E}\mathcal{B}\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{A}\mathfrak{U}(\varphi), \mathsf{E}\mathcal{B}\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{E}(\{\mathsf{A}\mathfrak{U}(\varphi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{E}(\{\mathfrak{U}(\mathsf{A}\varphi)\} \cup \mathcal{B})\} \cup \mathcal{G})
\end{aligned}
$$

The A cases are very similar to the E cases up to a few exceptions. It actually suffices to only look at the E cases, but for completeness we give the full proof.

$$
\begin{aligned}
&(\mathsf{A}\ell) : \mathfrak{D}(\{\mathsf{A}(\{\ell\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\ell)\} \cup \mathcal{B})\} \cup \mathcal{G})
\end{aligned}
$$

$$
\begin{aligned}
&(\mathsf{A}\vee) : \mathfrak{D}(\{\mathsf{A}(\{\varphi \vee \psi\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{A}(\{\varphi, \psi\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\varphi), \psi\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{A}(\{\psi\} \cup \mathcal{B} \cup \mathfrak{U}(\varphi))\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\psi)\} \cup \mathcal{B} \cup \{\mathfrak{U}(\varphi)\})\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\varphi), \mathfrak{U}(\psi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{A}(\mathfrak{U}(\varphi) \vee \mathfrak{U}(\psi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&\qquad = \mathfrak{D}(\{\mathsf{A}(\mathfrak{U}(\varphi \vee \psi)\} \cup \mathcal{B})\} \cup \mathcal{G})
\end{aligned}
$$

$$
\begin{aligned}
(\mathsf{A}\wedge) : \mathfrak{D}(\{\mathsf{A}(\{\varphi \wedge \psi\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\varphi\} \cup \mathcal{B}), \mathsf{A}(\{\psi\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\varphi)\} \cup \mathcal{B}), \mathsf{A}(\{\mathfrak{U}(\psi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\varphi) \wedge \mathfrak{U}(\psi)\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\varphi \wedge \psi)\} \cup \mathcal{B})\} \cup \mathcal{G})
\end{aligned}
$$

$$
\begin{aligned}
(\mathsf{A}[\,\underline{\mathsf{U}}\,]) : &\mathfrak{D}(\{\mathsf{A}(\{[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\varphi, \psi\} \cup \mathcal{B}), \mathsf{A}(\{\psi, \mathsf{X}[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\varphi), \mathfrak{U}(\psi)\} \cup \mathcal{B}), \mathsf{A}(\{\mathfrak{U}(\psi), \mathsf{X}[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\varphi) \vee \mathfrak{U}(\psi)\} \cup \mathcal{B}), \mathsf{A}(\{\mathfrak{U}(\psi) \vee \mathsf{X}[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\psi) \vee \mathfrak{U}(\varphi) \wedge \mathsf{X}[\varphi\ \underline{\mathsf{U}}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}([\varphi\ \underline{\mathsf{U}}\ \psi])\} \cup \mathcal{B})\} \cup \mathcal{G})
\end{aligned}
$$

$$
\begin{aligned}
(\mathsf{A}[\,\mathsf{B}\,]) : &\mathfrak{D}(\{\mathsf{A}(\{[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\neg\psi\} \cup \mathcal{B}), \mathsf{A}(\{\varphi, \mathsf{X}[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\neg\psi)\} \cup \mathcal{B}), \mathsf{A}(\{\mathfrak{U}(\varphi), \mathsf{X}[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\neg\psi)\} \cup \mathcal{B}), \mathsf{A}(\{\mathfrak{U}(\varphi) \vee \mathsf{X}[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\neg\psi) \wedge (\mathfrak{U}(\varphi) \vee \mathsf{X}[\varphi\ \mathsf{B}\ \psi]\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}([\varphi\ \mathsf{B}\ \psi])\} \cup \mathcal{B})\} \cup \mathcal{G})
\end{aligned}
$$

And now the final cases, which are again very similar.

$$
\begin{aligned}
(\mathsf{AE}) : \mathfrak{D}(\{\mathsf{A}(\{\mathsf{E}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\{\mathsf{E}\varphi\}) \cup \mathcal{G}\} \cup \mathfrak{D}(\{\mathsf{A}\mathcal{B}\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\{\mathsf{E}\mathfrak{U}(\varphi)\}) \cup \mathcal{G}\} \cup \mathfrak{D}(\{\mathsf{A}\mathcal{B}\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\{\mathsf{E}\mathsf{E}\mathfrak{U}(\varphi)\}) \cup \mathcal{G}\} \cup \mathfrak{D}(\{\mathsf{A}\mathcal{B}\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\{\mathsf{E}\mathfrak{U}(\mathsf{E}\varphi)\}) \cup \mathcal{G}\} \cup \mathfrak{D}(\{\mathsf{A}\mathcal{B}\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\mathsf{E}\varphi)\} \cup \mathcal{B})\} \cup \mathcal{G})
\end{aligned}
$$

$$
\begin{aligned}
(\mathsf{AA}) : \mathfrak{D}(\{\mathsf{A}(\{\mathsf{A}\varphi\} \cup \mathcal{B})\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\{\mathsf{A}\varphi\}) \cup \mathcal{G}\} \cup \mathfrak{D}(\{\mathsf{A}\mathcal{B}\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\{\mathsf{A}\mathfrak{U}(\varphi)\}) \cup \mathcal{G}\} \cup \mathfrak{D}(\{\mathsf{A}\mathcal{B}\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\{\mathsf{A}\mathsf{A}\mathfrak{U}(\varphi)\}) \cup \mathcal{G}\} \cup \mathfrak{D}(\{\mathsf{A}\mathcal{B}\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\{\mathsf{A}\mathfrak{U}(\mathsf{A}\varphi)\}) \cup \mathcal{G}\} \cup \mathfrak{D}(\{\mathsf{A}\mathcal{B}\} \cup \mathcal{G}) \\
&= \mathfrak{D}(\{\mathsf{A}(\{\mathfrak{U}(\mathsf{A}\varphi)\} \cup \mathcal{B})\} \cup \mathcal{G})
\end{aligned}
$$

By inductively applying Lemma 7 for each element in a block we get the following:

Lemma 8 (Unrolling temporal operators) *For any block* $\mathsf{E}\{\pi_0, \ldots, \pi_{m-1}\}$ *and any block* $\mathsf{A}\{\sigma_0, \ldots, \sigma_{n-1}\}$*, we have the following*

- $\mathfrak{D}(\{\mathsf{E}\{\pi_0, \ldots, \pi_{m-1}\} \cup \mathcal{G}\}) = \mathfrak{D}(\{\mathsf{E}\{\mathfrak{U}(\pi_0), \ldots, \mathfrak{U}(\pi_{m-1})\} \cup \mathcal{G}\})$
- $\mathfrak{D}(\{\mathsf{A}\{\sigma_0, \ldots, \sigma_{n-1}\} \cup \mathcal{G}\}) = \mathfrak{D}(\{\mathsf{A}\{\mathfrak{U}(\sigma_0), \ldots, \mathfrak{U}(\sigma_{n-1})\} \cup \mathcal{G}\})$

By unrolling the formula prior decomposition, all decomposition cases of a temporal operator can be omitted. Furthermore, the length of the formula will stay polynomial. This will give us an insight, as to why the procedure explained in the following chapter is isomorphic to this one. A formal isomorphism is beyond the scope of this thesis.

Concluding, we also state our strong belief that the upper bound of the Decompose function is exponential, because it strongly resembles Algorithm "Construction of the automaton" in the following Chapter.

3 Alternating Tree Automata

3.1 Motivation

The tableau procedure of the previous section can also be described in a symbolic way which then leads to alternating tree automata which are closely related to the vector μ-calculus. We describe this procedure first with examples and give formal definitions later.

As a first example, consider the CTL formula $\mathsf{EGEF}a$. We first abbreviate the subformulas starting with a path quantifier and obtain this way the definitions $x_1 := \mathsf{EF}a$ and $x_2 := \mathsf{EG}x_1$ for the remaining formula x_1. In a second step, we unroll the abbreviations and obtain

$$\left\{\begin{array}{l} x_1 = \mathsf{EF}a \\ \hline x_2 = \mathsf{EG}x_1 \\ \hline x_2 \end{array}\right\} \Leftrightarrow \left\{\begin{array}{l} x_1 = \mathsf{E}(a \vee \mathsf{XF}a) \\ \hline x_2 = \mathsf{E}(x_1 \wedge \mathsf{XG}x_1) \\ \hline x_2 \end{array}\right\} \Leftrightarrow \left\{\begin{array}{l} x_1 = a \vee \mathsf{EXEF}a \\ \hline x_2 = x_1 \wedge \mathsf{EXEG}x_1 \\ \hline x_2 \end{array}\right\}$$

As can be seen, we obtain on the right hand side the previous right hand side as subformulas so that we obtain recursive definitions for the variables x_i:

$$\left\{\begin{array}{l} x_1 = a \vee \mathsf{EX}x_1 \\ \hline x_2 = x_1 \wedge \mathsf{EX}x_2 \\ \hline x_2 \end{array}\right\}$$

The above equation system describes an alternating tree automaton which can also be used to create a model: The initial states are those satisfying the remaining formula x_2, i.e., $(a \vee \mathsf{EX}x_1) \wedge \mathsf{EX}x_2$. Hence, our initial states need to satisfy $\psi_1 := \{a, \mathsf{EX}x_2\}$ and $\psi_2 := \{\mathsf{EX}x_1, \mathsf{EX}x_2\}$. By the labels of these states, it follows that the states satisfying ψ_1, which we will call $s_0 := \mathsf{SAT}(\psi_1)$, must have a successor state where x_2 holds and that the states satisfying ψ_2, which we will call $s_1 := \mathsf{SAT}(\psi_2)$, must have a successor state where x_1 holds and another successor state where x_2 holds. We already expanded x_2 into the states s_0 and s_1 and therefore now have the transitions $\{(i,j) \,|\, i \in s_1, j \in s_0\}$ and $\{(i,j) \,|\, i \in s_1, j \in s_1\}$. The expansion of x_1 yields $a \vee \mathsf{EX}x_1$, i.e., the set of states satisfying $s_2 := \mathsf{SAT}(\{a\})$ and $s_3 := \mathsf{SAT}(\{\mathsf{EX}x_1\})$. The above explanation leads to a structure with deadend states which is not precisely correct for a model of the formula. It is however exactly the same as the procedure described in [FLL10] computes.

This is because we translate $\mathsf{EF}a$ to $\mu x.(a \vee \mathsf{EX}x)$, but the precise translation is $\mu x.(a \wedge \Phi_\infty \vee \Diamond x)$ where $\Phi_\infty := \nu y.\Diamond y$ holds in all states having an infinite outgoing path.

If we repeat the above calculations this way, we obtain

$$\left\{\begin{array}{l} x_0 \stackrel{\nu}{=} \Diamond x_0 \\ x_1 = a \wedge x_0 \vee \Diamond x_1 \\ x_2 = x_1 \wedge \Diamond x_2 \\ \hline x_2 \end{array}\right\} \Leftrightarrow \left\{\begin{array}{l} x_0 \stackrel{\nu}{=} \Diamond x_0 \\ x_1 \stackrel{\mu}{=} a \wedge \Diamond x_0 \vee \Diamond x_1 \\ x_2 \stackrel{\nu}{=} a \wedge \Diamond x_0 \wedge \Diamond x_2 \vee \Diamond x_1 \wedge \Diamond x_2 \\ \hline x_2 \end{array}\right\}$$

Let us consider a second example with the formula $\mathsf{E}(\mathsf{F}a \wedge \mathsf{F}b)$. It is not difficult to see that we compute the following alternating tree automaton:

$$\left\{\begin{array}{l} x_0 \stackrel{\nu}{=} \Diamond x_0 \\ x_1 \stackrel{\mu}{=} a \wedge \Diamond x_0 \vee \Diamond x_1 \\ x_2 \stackrel{\mu}{=} b \wedge \Diamond x_0 \vee \Diamond x_2 \\ x_3 \stackrel{\nu}{=} a \wedge b \wedge \Diamond x_0 \vee a \wedge \Diamond x_2 \vee b \wedge \Diamond x_1 \vee \Diamond x_3 \\ \hline x_3 \end{array}\right\}$$

3.2 Construction of a model Kripke structure

In order to define an alternating tree automaton we firstly need the following definition:

Definition 12 (Modal logic form) *A formula in modal logic form is a propositional logic formula over the variables $\mathcal{V}$ also allowing literals of the kind $\Diamond v$ and $\Box v$ for $v \in \mathcal{V}$. Denote by $MF(\mathcal{V})$ the set of modal logic form formulas over the variables $\mathcal{V}$.*

Definition 13 (Alternating Tree Automaton) *An Alternating Tree Automaton is a 4 tuple $(\mathcal{X}, G, \alpha, \mathcal{V})$.*

- *$\mathcal{X}$ is a finite set of abbreviated variables $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$*
- *G is a map, $\mathcal{X} \to MF(\mathcal{V} \cup \mathcal{X})$*
- *α is a propositional logic formula over the variables $\mathcal{X}$ describing the initial states*
- *$\mathcal{V}$ is a set of variables.*

G will describe the system of equations, as it maps each variable to a modal logic form variable. α will be our residue formula, the one that will describe the initial states. Now let us describe the algorithm that turns a given CTL^* formula into an alternating tree automaton.

Algorithm Construction of the automaton

Input: A CTL* formula φ
Output: An alternating tree automaton $(\mathcal{X}, G, \alpha, \mathcal{V})$

1: $\mathcal{V} := \mathcal{V}(\varphi)$
2: $\mathcal{G} := \{\}$
3: $\mathcal{X} := \{\}$
4: $\alpha := \mathsf{NNF}(\varphi)$
5: In a top-down manner, recursively abbreviate each new non literal state formula ψ by a new variable x_i in α.
6: $\mathcal{X} := \mathcal{X} \cup \{x_i\}$
7: $\mathcal{G} := \mathcal{G} \cup \{x_i := \psi\}$
8: Unroll each variable x_i and make a DNF.
9: Now that we have a goal normal form, for the E and A blocks abbreviate $\mathsf{EX}\mathcal{B}$ and $\mathsf{AX}\mathcal{B}$ by $\mathsf{EX}x_j$ and $\mathsf{AX}x_j$ respectively for a fresh x_j.
10: If not yet unrolled variables exist GOTO 5.
11: RETURN $(\mathcal{X}, \mathcal{G}, \alpha, \mathcal{V})$;

Algorithm Construction of the Kripke structure

Input: An alternating tree automaton $(\mathcal{X}, G, \alpha, \mathcal{V})$ for φ
Output: A Kripke structure $\mathcal{K} = (\mathcal{S}, \mathcal{I}, \mathcal{R}, \mathcal{L})$

1: $\mathcal{I} := \mathsf{SAT}(\alpha)$
2: $\mathcal{Q} = \mathcal{I}$
3: $\mathcal{S} = \mathcal{Q}$
4: while $\mathcal{Q} \neq \{\}$ do :
5: Let $q \in \mathcal{Q}$ be a not yet explored state.
6: For each $\Diamond x$ in the state q do :
7: Let ψ be the definition of x.
8: $\mathcal{Q} := \mathcal{Q} \cup \mathsf{SAT}(\psi)$
9: $\mathcal{R} := \mathcal{R} \cup (\{q\} \times \mathsf{SAT}(\psi))$
10: end for each
11: For each $\Box x$ in the state q and successor q' do :
12: Let ψ be the definition of x.
13: $\mathcal{L}(q') := \mathcal{L}(q') \cup \{\psi\}$
14: end for each
15: Remove all variables in $\mathcal{X}$ from q.
16: $\mathcal{Q} := \mathcal{Q} \backslash \{q\}$
17: end while
18: Remove all inconsistent states and transitions leading to one in $\mathcal{K}$.
19: if $\mathcal{K} \models \varphi$
20: RETURN $\mathcal{K}$;
21: else
22: RETURN UNSATISFIABLE;

Let us underline the fact, that different DNF will lead to different alternating

tree automata as made clear by the two equivalent formulas

$$\mathsf{EF}(a \vee b) = \mathsf{EF}\left((a \wedge b) \vee (a \wedge \neg b) \vee (\neg a \wedge b)\right).$$

Lemma 9 *For a given* CTL *formula φ, we obtain at most $|\varphi|$ many equations and the sizes of the right hand sides are in $O(1)$.*

A state of the Kripke structure consists of subset of variables $\mathcal{V}$, a subset of diamond variables, and a further subset of box variables. Since all these sets are bound by $|\varphi|$, it can be clearly seen that we have at most $2^{O(|\varphi|)}$ many states.

Lemma 10 *For a given* CTL *formula φ, the Kripke structure has at most $2^{O(|\varphi|)}$ many states and each state has at most $O(|\varphi|)$ many successor states.*

Lemma 11 *There are* CTL* *formulas Φ_n of size $O(n)$ which yield $O(2^n)$ many abbrevations/equations.*

Table 3.1: *An overview of this procedure for different types of logic*

Logic	**Equations**	$\lvert\alpha\rvert$	**Decision Runtime**
$\varphi \in$ CTL	$O(\lvert\varphi\rvert)$	$O(\lvert\varphi\rvert)$	$O(2^{\lvert\varphi\rvert})$
$\varphi \in$ LTL	$O(2^{\lvert\varphi\rvert})$	$O(1)$	$O(2^{2^{\lvert\varphi\rvert}})$
$\varphi \in$ CTL*	$O(2^{\lvert\varphi\rvert})$	$O(2^{\lvert\varphi\rvert})$	$O(2^{2^{\lvert\varphi\rvert}})$
$\varphi \in \mu$-calculus	$O(\lvert\varphi\rvert)$	$O(2^{\lvert\varphi\rvert})$	$O(2^{2^{\lvert\varphi\rvert}})$

Although the decision runtime for LTL is actually single exponential our decision procedure will need doubly exponential time for it. [SC85]

3.3 A Worst-Case Example

Theorem 1 (Worst-Case Example) *The formula below requires an exponential number of equations and yields a double exponentially sized Kripke structure.*

$$\Phi_n := \mathsf{E} \bigwedge_{i=0}^{n-1} \mathsf{F} p_i$$

In particular, the above formula Φ_n generates $2^n - 1$ abbreviations and $O(2^{2^n})$ many initial states of the Kripke structure.

Proof:
We first prove that we get exactly $2^n - 1$ many abbreviations for Φ_n. To this end, we first consider the following unrolling where $\mathcal{P}$ is the set of all subsets of $\{0, \ldots, n-1\}$ and for any $\vartheta \subseteq \mathcal{P}$, we define $\overline{\vartheta} := \{0, \ldots, n-1\} \backslash \vartheta$:

$$\Phi_n = \mathsf{E} \bigwedge_{i \in \{0,\ldots,n-1\}} \mathsf{F} p_i = \mathsf{E} \bigwedge_{i \in \{0,\ldots,n-1\}} (p_i \vee \mathsf{XF} p_i) = \bigvee_{\vartheta \subseteq \mathcal{P}} \mathsf{E} \left(\bigwedge_{j \in \vartheta} p_j \wedge \bigwedge_{j \in \overline{\vartheta}} \mathsf{XF} p_j \right)$$

We can now shift the path quantifier inwards and get

$$\Phi_n = \bigvee_{\vartheta \subseteq \mathcal{P}} \left(\bigwedge_{j \in \vartheta} p_j \wedge \Diamond \underbrace{\mathsf{E} \bigwedge_{j \in \overline{\vartheta}} \mathsf{F} p_j}_{=: x_\vartheta} \right)$$

Hence, for any nonempty set $\vartheta \subseteq \mathcal{P}$, we get one abbreviation. Hence, we have the following $2^n - 1$ many abbreviations

$$x_\vartheta := \mathsf{E} \bigwedge_{j \in \overline{\vartheta}} \mathsf{F} p_j$$

Each one of these abbreviation can be determined as a fixpoint (since the definitions mutually depend on each other):

$$x_\vartheta = \bigvee_{\kappa \subseteq \vartheta} \left(\bigwedge_{j \in \kappa} p_j \wedge \Diamond x_\kappa \right)$$

The above variables x_ϑ are the state variables of the alternating tree automaton, and their definitions determine the state transitions. Moreover, we obtain the following formula for the initial states:

$$\Psi_n = \bigvee_{\vartheta \subseteq \mathcal{P}} \left(\bigwedge_{j \in \vartheta} p_j \wedge \Diamond x_\vartheta \right)$$

The above formula for the initial states is in disjunctive normal form and it has obviously 2^n many cubes which will form goals in the related tableau.

If we consider a related state transition system whose states are determined by variables $p_0, \ldots, p_{n-1}$ and the $2^n - 1$ many variables x_ϑ in the version $\Diamond x_\vartheta$, we can have up to 2^{n+2^n-1} variable assignments, and thus states.

We can determine the above numbers as follows: The number of initial states is the number of satisfying assignments of the formula Ψ_n. As already, discussed, this formula is in disjunctive normal form and has exactly 2^n cubes of the form $\bigwedge_{j\in\vartheta} p_j \wedge \Diamond x_\vartheta$. Such a cube determines a unique truth value for the $|\vartheta|$ many variables p_j with index $j \in \vartheta$ and also for one of the abbreviation variables. The truth values of all other variables is not determined, so that they may have any value.

Hence, the cube $\bigwedge_{j\in\vartheta} p_j \wedge \Diamond x_\vartheta$ determines the truth values of $|\vartheta| + 1$ of the $n + 2^n - 1$ many variables and therefore represents the following number of satisfying assignments (and thus states):

$$2^{n+2^n-1-|\vartheta|-1} = 2^{n+2^n-|\vartheta|-2}$$

Observe that the above is also a lowerbound on the number of initial states.

Altogether, Ψ_n has as a trivial upper bound the following number of initial states

$$\sum_{\vartheta\subseteq\mathcal{P}} 2^{n+2^{n+1}-|\vartheta|-3}$$

In particular, the above formula Φ_n generates $2^n - 1$ abbreviations and $O(2^{2^n})$ many initial states of the Kripke structure.

4 Conclusion & Future Work

This thesis was originally motivated by the following observation in the paper of [FLL10] about the number of goals. It was stated that the number of all possible goals is a finite set of at most doubly exponential size.

We proved this statement to be correct for the number of all possible goals that can exist. However, a more interesting question is whether the tableau algorithm really can create a doubly exponential sized number of goals. This was firstly our original question, but then the main focus quickly deviated into actually finding a worst case example.

Although this was not the main focus, after the results on this thesis, it is almost sure this can not happen. As said before, to conclude the final proof of this, we would need a formal isomorphism between the algorithms in Chapter 2 and Chapter 3. We believe that this can be done with a structural induction over the structure of goals. This could be potential future work.

The worst case example is stated in Theorem 1. Along the way we learned a lot about the structure of these worst case examples. At the beginning we believed it would be very rare to find such a formula, but it turns out that there are a lot of formulas which induce this worst case runtime. We believe the only real requirement is that the formula is not in CTL and not CTL like, as the constructed alternating tree automaton will only have a linear number of equations. We took the example which was most friendly to analyze analytically and made a formal proof of the runtime.

Also a notable achievement of this thesis were the smaller technical proofs, which will clear the way for further stronger statements in the future. It seems intuitive that the unroll function alone could potentially lead to another decision procedure, of course we would also need another function to continue unrolling after we did this once.

The decision procedures for CTL* use heavy machinery of theoretical computer science ranging from tree automata to parity games. It would also be of didactic importance to find decision procedures more welcoming to students, where the runtime might not be as good.

Bibliography

[Dam94] M. Dam. "CTL* and ECTL* as fragments of the modal μ-calculus". In: *Theoretical Computer Science (TCS)* 126.1 (1994), pp. 77–96.

[EJ88] E.A. Emerson and C.S. Jutla. "The Complexity of Tree Automata and Logics of Programs". In: *Foundations of Computer Science (FOCS)*. White Plains, New York, USA, 1988, pp. 328–337.

[EJ99] E Allen Emerson and Charanjit S Jutla. "The complexity of tree automata and logics of programs". In: *SIAM Journal on Computing* 29.1 (1999), pp. 132–158.

[EL85] E.A. Emerson and C.-L. Lei. "Modalities for Model Checking: Branching Time Strikes Back". In: *Principles of Programming Languages (POPL)*. New Orleans, Louisiana, USA: ACM, 1985, pp. 84–96.

[ES84a] E. Allen Emerson and A. Prasad Sistla. "Deciding Branching Time Logic". In: *Proceedings of the 16th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1984, Washington, DC, USA*. Ed. by Richard A. DeMillo. ACM, 1984, pp. 14–24.

[ES84b] E.A. Emerson and A.P. Sistla. "Deciding Full Branching Time Logic". In: *Information and Control* 61.3 (1984), pp. 175–201.

[FL10] Oliver Friedmann and Markus Latte. "Decision Procedures for CTL". In: *Proceedings of the 2010 International Workshop on Comparing Logical Decision Methods, CLoDeM'2010*. Edinburgh, United Kingdom, 2010.

[FLL10] O. Friedmann, M. Latte, and M. Lange. "A Decision Procedure for CTL* Based on Tableaux and Automata". In: *International Joint Conference on Automated Reasoning (IJCAR)*. Ed. by J. Giesl and R. Hähnle. Vol. 6173. LNCS. Edinburgh, Scotland, UK: Springer, 2010, pp. 331–345.

[GTW02] E. Grädel, W. Thomas, and T. Wilke. *Automata, Logics, and Infinite Games*. Ed. by E. Grädel, W. Thomas, and T. Wilke. Vol. 2500. LNCS. Dagstuhl, Germany: Springer, 2002.

[KPV12] O. Kupferman, A. Pnueli, and M.Y. Vardi. "Once and for all". In: *Journal of Computer and System Sciences* 78 (2012), pp. 981–996.

[Rey09] Mark Reynolds. "A tableau for ctl". In: *International Symposium on Formal Methods*. Springer. 2009, pp. 403–418.

[SC85] A Prasad Sistla and Edmund M Clarke. "The complexity of propositional linear temporal logics". In: *Journal of the ACM (JACM)* 32.3 (1985), pp. 733–749.

[YS85] M. Y Vardi and L. Stockmeyer. “Improved upper and lower bounds for modal logics of programs”. In: *Symposium on Theory of Computing (STOC)*. ACM, 1985, pp. 240–251.