On the Complexity of Modal Logic Variants and their Fragments

Von der Fakultät für
Elektrotechnik und Informatik der
Gottfried Wilhelm Leibniz Universität Hannover

zur Erlangung des Grades
Doktor der Naturwissenschaften
Dr. rer. nat.

genehmigte Dissertation

von
M. Sc. Arne Meier
geboren am 06.05.1982 in Hannover

2011

Referent: Heribert Vollmer, Gottfried Wilhelm Leibniz Universität Hannover
Korreferent: Martin Mundhenk, Friedrich-Schiller-Universität Jena

Tag der Promotion: 14.11.2011

Für Julia

Mathematical reasoning may be regarded rather schematically as the exercise of a combination of two facilities, which we may call intuition and ingenuity.

(Alan Turing)

Danksagung

Zu allererst möchte ich meinem Doktorvater Heribert Vollmer für die großartige Unterstützung beim Verfassen dieser Arbeit danken. Ein weiterer Dank gebührt ihm dafür, dass ohne ihn ich wohl niemals die theoretische Informatik und das wissenschaftliche Arbeiten an sich lieben gelernt hätte. Des Weiteren möchte ich mich bei Martin Mundhenk für die vielen Besuche in Jena bedanken, die besonders die Teile zur temporalen Logik in dieser Arbeit stark geprägt haben.

Weiteren Dank schulde ich meinem Kollegen Michael Thomas, mit dem ich drei Jahre ein Büro geteilt habe, und viele Gespräche mit ihm mir sehr geholfen haben. Außerdem danke ich ihm für die gemeinsame Forschung, die mir persönlich viel gebracht hat. Daniel Zaum und Peter Lohmann möchte ich sehr für das Lesen meiner Arbeit und den damit verbundenen Anregungen danken, sowie Daniel für die Hilfe bei der Umschlaggestaltung. Vielen Dank auch an Johannes Ebbing für die Gespräche über die Logik ECTL.

Außerdem möchte ich Thomas Schneider für seine Ratschläge in Bezug auf das wissenschaftliche Arbeiten und insbesondere für die gemeinsame Forschung an den Beschreibungslogiken danken.

An besonderer Stelle danke ich von tiefstem Herzen meinen Eltern Inge und Heinz dafür, dass sie mir nicht nur ein Studium in der Informatik ermöglicht haben, sondern auch dafür, dass sie immer zu mir gestanden und mich unterstütz haben. Ohne diese Hilfe wäre es mir niemals möglich gewesen in diesem Fach auch zu promovieren und überhaupt so weit zu kommen!

Natürlich danke ich besonders meiner Frau, Julia, welche mir immer Verständnis entgegen gebracht hat, nicht nur wenn ich wieder an der Arbeit geschrieben habe, sondern auch jedes mal, wenn ich beruflich unterwegs war. (Vielen)$^{\omega}$ Dank für Deine Liebe, die mir sehr hilft und mich antreibt.

Acknowledgements

First and foremost I want to thank my supervisor Heribert Vollmer for his great assistance at writing this thesis. Without him I probably would not have started to contract a passion to theoretical computer science or to science itself. Also I would like to thank Martin Mundhenk for my visits in Jena which influenced especially the parts about temporal logic in this thesis.

Further thanks I owe to my colleague Michael Thomas with whom I shared an office for three years. Many talks and answers to my questions helped a lot. Also the research together have been enjoyed by myself very much. Additionally I want to thank Daniel Zaum and Peter Lohman for reading my thesis and the therewith connected suggestions; in particular I want to thank Daniel for designing the cover. Thank you, Johannes Ebbing, for the talks about the logic ECTL.

Besides I wish to thank Thomas Schneider for his advice about how to write proofs and also about the joint work on the description logics.

At this point, I want to thank my parents Inge and Heinz from the bottom of my heart for making studies in computer science possible for me and also for being always supportive. Without that much help I would have never been able to take a doctoral degree in computer science.

Of course I thank my wife, Julia, who always was patient and sympathetic where I needed it when I worked on this thesis or whenever I was underway from work. Thank you (very)$^{\omega}$ much for your love which helps very much and is my impulse.

Zusammenfassung

Die komplett automatische Verifikation von Computer Programmen ist ein sehr entscheidender Schritt im Rahmen der Entwicklung von Software. In diesem Kontext wurden Temporale Logiken erfunden, die eine Erweiterung der Modallogik darstellen, welche selbst eine Erweiterung der gewöhnlichen Aussagenlogik ist. Aus diesem Grund kann man sie *Modallogik-Varianten* nennen.

Der erste Teil dieser Arbeit wird die beiden Temporalen Logiken CTL und CTL* mit Bezug auf ihr Model Checking- und Erfüllbarkeits-Problem untersuchen. Wir werden die Komplexität von Fragmenten dieser Probleme im Sinne von Einschränkungen bezüglich erlaubter Operatoren und Boole'scher Funktionen analysieren. Hierbei werden wir für das Erfüllbarkeitsproblem sehen, inwiefern die Operator-Fragmente eine Trichotomie bilden, und die Boole'schen Fragmente in vier verschiedene Komplexitätsgrade zerfallen. Das Model Checking-Problem für CTL teilen wir in ein monotones, eins nur mit atomarer Negation, und ein positives Fragment. Überraschenderweise werden wir sehen, dass diese Fragmente sich äquivalent bezüglich ihrer Komplexität verhalten. Darüberhinaus werden wir die Fragmente im obigen Sinne mit Bezug auf die obigen Probleme einiger sehr bekannter Erweiterungen von CTL klassifizieren.

Im zweiten Teil werden wir uns mit sogenannten Beschreibungslogiken beschäftigen. Diese Modallogik-Erweiterungen spielen eine wichtige Rolle im Bereich des Semantic Web, der Datenbanksysteme, und in der Künstlichen Intelligenz. Diese Logiken werden unter anderem dazu verwendet, um große Datenmengen zu beschreiben und auf ihnen zu operieren. Neben den typischen Erfüllbarkeits-Problemen werden wir mit einem speziell an diesen Typ von Logiken angepassten Implikations-Problem arbeiten, welches Subsumption genannt wird. Wir werden außerdem sehen, dass alle diese Logiken zwei sehr mächtige Boole'sche Konzepte innehaben, nämlich Implikation und Konjunktion, welche maßgeblich die Komplexität der Probleme beeinflussen. Hierdurch wird das Verbieten von großen Mengen Boole'scher Funktionen die eigentliche Komplexität dieser Probleme nicht erheblich vermindern.

Schlagworte: Beschreibungslogik, Komplexität, Modale Logik, Post'scher Verband, Temporale Logik.

Abstract

The automatic verification of computer programs is an important step in software engineering. In this regard temporal logics have been invented as an extension of modal logic which itself is an extension of propositional logic. Therefore, one may call them *modal logic variants*.

The first part of this thesis will investigate the two temporal logics CTL and CTL* with respect to their model checking and satisfiability problem. We will analyze the complexity of fragments of these problems by means of operator and Boolean function restrictions. There we will see for the satisfiability problem, how the operator fragments form a trichotomy and the Boolean fragments form a quartering. The model checking problem for CTL is divided into three types: monotone, atomic negation, and positive fragments. Surprisingly, we will see that these three fragments are computationally equivalent. Furthermore, several prominent extensions of CTL will be visited and classified with respect to their Boolean and operator fragments.

In the second part we will concentrate on description logics which are modal logic extensions settled in the area of semantic web, databases, and artificial intelligence. These types of logics are used to express, and work on, large sets of data. Besides the usual satisfiability problems, we will work with some special kind of implication problem, which is called subsumption. We will see that these logics combine two very strong Boolean concepts, namely implication and conjunction, such that restricting large sets of Boolean functions do not reduce the complexity of the problems significantly.

Keywords: computational complexity, description logic, modal logic, Post's lattice, temporal logic.

Contents

1 Introduction **1**
- 1.1 Modal Logic . . . 2
  - 1.1.1 Temporal Logic . . . 3
  - 1.1.2 Description Logic . . . 3
  - 1.1.3 Post's Lattice . . . 5
- 1.2 Results . . . 5
- 1.3 Publications . . . 6

2 Preliminaries **9**
- 2.1 Complexity Theory . . . 10
- 2.2 Boolean Clones . . . 13
- 2.3 Modal Logic . . . 17
  - 2.3.1 Temporal Logic . . . 18
  - 2.3.2 Description Logic . . . 23
- 2.4 Complete Problems . . . 26

3 Temporal Logic **29**
- 3.1 Satisfiability in CTL and CTL* . . . 29
  - 3.1.1 Restricting the Boolean connectives . . . 29
  - 3.1.2 Restricting the CTL-operators . . . 32
  - 3.1.3 Satisfiability for fragments of CTL* . . . 41
  - 3.1.4 About the Affine Cases . . . 43
  - 3.1.5 Fragments of Extensions of CTL: Fairness, Succinctness, and LTL$^+$ 45
  - 3.1.6 Conclusion . . . 49
- 3.2 Model Checking in CTL and CTL* . . . 50
  - 3.2.1 Model Checking CTL and CTL$_{\text{pos}}$ . . . 53
  - 3.2.2 Model Checking Extensions of CTL . . . 61
  - 3.2.3 Model Checking CTL* . . . 68
  - 3.2.4 Conclusion . . . 72

4 Description Logic **75**
- 4.1 TBox and Ontology Satisfiability . . . 75
  - 4.1.1 Both quantifiers . . . 79
  - 4.1.2 Restricted quantifiers . . . 82
  - 4.1.3 Conclusion . . . 93
- 4.2 Subsumption . . . 93
  - 4.2.1 Conclusion . . . 103

5 Concluding Remarks **107**

Bibliography **111**

Index **121**

Lebenslauf **125**

List of Figures

2.1 Complexity class inclusion diagram 12
2.2 Post's lattice of all Boolean clones. 16
2.3 The lattice induced by all CTL- and all CTL*-operators 21

3.1 General Kripke Structure for $\varphi \in \text{CTL-SAT}(\{\text{AF}\}, \text{BF})$ 33
3.2 Part of a nested tree-like structure used in the proof of Theorem 3.4. 34
3.3 Quasi-model for $\varphi = \text{EG}(p \vee \neg q) \wedge \text{AF}(\text{EG}(q))$ 35
3.4 The relevant part of the lattice induced by all ECTL-operators. 47
3.5 Example showing $\text{AFAG}p \not\equiv \overset{\infty}{\text{AG}}p$ 49
3.6 Overview of the complexity of CTL-SAT(T, BF), CTL*-SAT(T, BF), and CTL$^+$-SAT(T, BF), i.e., without any restrictions to the Boolean functions. . . 50
3.7 The complexity of CTL-SAT, CTL$^+$-SAT, and CTL*-SAT, without restrictions on the temporal operators. 51
3.8 Kripke structure constructed for $\text{CTL}_{\text{mon}}\text{-MC}(\{\text{EG}\})$ 57
3.9 Kripke structure for $\text{CTL}^+{}_{\text{pos}}\text{-MC}(\{\text{E}, \text{G}\})$ 65
3.10 Post's lattice restricted to clones with both constants. 68
3.11 Complexity of $\text{CTL}_{\text{pos}}\text{-MC}(T), \text{CTL}_{\text{mon}}\text{-MC}(T), \text{CTL}_{\text{a.n.}}\text{-MC}(T)$ for all sets T of CTL-operators 73

4.1 Post's lattice showing the complexity of $\text{SUBS}_{\mathcal{Q}}(B)$ for all non-empty sets $\emptyset \subsetneq \mathcal{Q} \subseteq \{\exists, \forall\}$ and all Boolean clones $[B]$. 105
4.2 Post's lattice showing the complexity for $\text{SUBS}_{\emptyset}(B)$ and all Boolean clones $[B]$. 106

List of Tables

2.1 A list of Boolean clones with definitions and bases. 15

3.1 Complexity overview for CTL*-MC(T, B) for all T with $|T| \leq 2$. 69

4.1 Complexity overview for $\text{TSAT}_{\mathcal{Q}}(B)$ and $\star\text{SAT}^{\text{ind}}_{\mathcal{Q}}(B)$ 94

Chapter 1

Introduction

Computational complexity is an area of theoretical computer science in which one aims to classify a given problem with respect to its worst case complexity measured in required computation time and space. Here a completeness result is the most satisfying answer as, informally, it states that the problem's complexity is fully classified. Now such a result may prove that the problem will always stay intractable and therefore forbids the existence of a polynomial time algorithm. One of the possible approaches to overcome this fact is the restriction of the problem with the hope of getting a faster algorithm for the fragment of this problem.

Now suppose you want to visit r of your relatives in one big journey. As you are free from work for only one week you are interested whether there exists an efficient route in at most one week duration and how it would look like. Thus we consider two different kind of questions. On the one hand there is a decision problem to which we can answer with simply *yes* or *no*. On the other hand we want to compute one optimal solution. Without doubt, knowing an optimal solution implies answering the decision problem. Vice versa, it is not clear if this is possible. Visualizing our situation in a graph of nodes (one for each relative and one for you) having edges between every pair of nodes, and edge labels with a distance or travel duration. The naïve approach computes all routes and selects one of the best. The computational effort of this algorithm measured in runtime is limited by the factorial of r, that is, in $r^{O(r)}$ many steps. Having about twenty relatives and computing one billion routes in one second would still need approximately $2 \cdot 10^8$ years to finish the computation. Further, more computation power would not lower the waiting time significantly as the problem exhibits exponential runtime. Thus we either need to find a better algorithm which uses some intelligent approach in deducing one of the desired routes or we simplify the problem by making several restrictions[1]. This could be forcing the triangle inequality to hold, disallowing asymmetric paths, or, e.g., forbidding several connections between some nodes (possibly one cannot directly travel from city x to city y). These approaches may involve understanding which parts of the problem make it inherently hard to solve.

Another promising approach is a transfer to propositional logic which enlarges the field of possible applicable algorithms. The most prominent open problem in theoretical computer science is the P-NP-problem which essentially is the question whether there exists an algorithm running in polynomial time solving the question from above, or

[1] Other approaches that will not be discussed in detail are approximation algorithms (see, for instance, [ACG+99]), or randomized algorithms which have an error property connected to wrong answers (see [MR95] for more information about this topic).

equivalently, deciding the satisfiability of a propositional formula. At present such an algorithm is not known to exist. But propositional logic has been proven itself to be a very powerful tool for encoding several difficult problems into the satisfiability problem SAT. By this property many different algorithms have been exhibited. Another very interesting property of the problem SAT is that one can efficiently verify solutions of instances, that is, given assignments to the variables one can check in polynomial time if this is indeed a correct solution. This fact is the main property of problems in the class NP (which stands for nondeterministic polynomial time). Further, SAT has received great attention because a polynomial time algorithm for SAT implies polynomial time algorithms for every problem of the class NP [Coo71b, Lev73]. Therefore several restrictions of SAT have been investigated where k-SAT comprises one surprising characteristic. If we restrict propositional formulae to conjunctive normal form, i.e., any formula can be written as conjunctions of disjunctive clauses containing only k literals (which are variables or their negations), then for $k = 2$ the problem becomes tractable whereas for $k = 3$ it is intractable unless P equals NP.

Furthermore, an approach used by H. Lewis in 1979 is the origin of an auspicious technique for understanding the hardness of a problem involving propositional logic [Lew79]: H. Lewis used a tool investigated by E. Post 1941 [Pos41], which is a lattice of all Boolean functions wherefore it is also called *Post's lattice*. The main application of this tool is to fragment any problem which inherently uses propositional connectives into all parts by means of any possible set of Boolean functions. Thereby H. Lewis was able to connect the intractability (unless $\mathsf{P} = \mathsf{NP}$) of SAT to some specific Boolean function, i.e., the negation of the implication function $\nrightarrow$. Thus whenever a formula is composed of Boolean functions that are in some way able to express $\nrightarrow$, one works with an instance of the intractable version of SAT. Consequently if we would be able to write a propositional formula for the travel problem from above avoiding $\nrightarrow$ (and functions that can express $\nrightarrow$ as well) then we would have a polynomial time algorithm for our recent case (unless the constructed formula is of super-polynomial size). Unfortunately it is not known whether such a formula exists as this would answer the P-NP-question as well.

The motivation of this thesis strictly encompasses this question. Which functions make a decision problem hard to solve? Why does the tractability of some problem depend on the availability of some Boolean function or operator? Here we will investigate several powerful extensions of propositional logic which are closely connected to modal logic.

1.1 Modal Logic

The connection of propositional logic to computer programs requires an adequate model where Kripke structures have been proven of great use. These structures are essentially directed node-labeled graphs simulating the behavior of a computer program in the means of different program states. Informally, modal logic is the extension of propositional logic by a new operator $\Diamond$ enabling formulae to express transitions between program states. 1918, modal logic has been firstly introduced by C. I. Lewis [Lew18] and has become very popular since the 1960s [Kri63, HC68] and until now [Gol06, BvW06]. Furthermore a

complete study with respect to the Boolean fragments of modal logic has been done by Hemaspaandra et al. [HSS08] recently.

1.1.1 Temporal Logic

Enriching modal logic with concepts to interact more densely with computer programs leads to the field of temporal logics which have been introduced by A. N. Prior in 1957 who has been called "the founding father of temporal logic" by the Danish Centre for Philosophy and Science Studies [Pri57, Pri67, Aal11]. From 1971 to 1986 significant effort by Pnueli, Emerson, Halpern, and Clarke resulted in the definition of the linear time logic LTL and the computation tree logics CTL* and CTL [Pnu77, CE81, QS82, EH86]. These logics have been invented to be of great benefit in the process of software engineering for verifying non-terminating programs. Describing specifications through formulae results in an evaluation of the written programs which are modeled by the Kripke structures as explained above. In the course of time, temporal logics emerged as being useful with major relevance for practical experience [VS85a]. In this context the *model checking problem* and the *satisfiability problem* of these logics are of great interest. For the model checking problem one asks if a given formula is satisfied in a given world of a given Kripke structure. Thus essentially the question whether a computer program fulfills its specification. For the satisfiability problem the question is whether a Kripke structure (containing a world) exists which satisfies a given formula. Hence we occupy with the question if there exists a computer program fulfilling this specification. In other words we ask some kind of consistency question with respect to a specification modeled by a temporal logic formula.

These two problems with respect to the three logics have been completely classified with respect to their complexity and without making any restrictions to the problems in [FL79, VS85a, CES86, Eme90, EJ00]. A comparison of these results bare a tremendous gap between model checking and satisfiability of CTL: a polynomial time model checking algorithm (and also hardness for P) is accompanied by an exponential time algorithm for satisfiability with the proof that there can be no better one. For the other two temporal logics the gap between the complexity of satisfiability and model checking is similar huge but both problems are intractable unless $\mathsf{P} = \mathsf{PSPACE}$. Model checking in both logics is complete for polynomial space whereas satisfiability remains PSPACE-complete for LTL and jumps up to double exponential time for CTL*. Whilst for LTL the classification of all Boolean and modal operator fragments has been achieved by Bauland et al. [BMS$^+$11, BSS$^+$09], the fragments of the computation tree logics are still open and will be investigated in Chapter 3.

1.1.2 Description Logic

The concept of databases influenced the development of description logics significantly. The origin of research has been considered to have started with the work of Brachman and Levesque in 1984 [BL84], whilst some principles of these logics go back to semantic networks and the KL-ONE system [BS85]. An extensive introduction to this field of logics has been written by Baader et al. [BCM$^+$03]. Description logics (DLs) are usually

defined as extension of the logic $\mathcal{AL}$ however some smaller fragments of $\mathcal{AL}$ recently received attention in the research community, namely the $\mathcal{FL}$- and $\mathcal{EL}$-family [Baa03, Bra04a, BBL05a, BBL08]. DLs are widely used in the semantic web in terms of the web ontology language OWL 2 [MPSP09]. Several terms from the web ontology language are synonyms of terms in the family of description logics and connect these two areas very closely. Further, DLs are used in the codification of medical knowledge by ontologies whose definition is explained below.

Regarding the connection to databases the two main formalisms in DLs are terminology and assertional boxes which are abbreviated by the terms *TBox* and *ABox*. The union of both is referred to as an *ontology*. An ABox is essentially a relational database with its pairs whereas the TBox expresses constraints for the database in form of rules (*axioms*, or without restrictions, *general concept inclusions* GCI). These rules are pairs of formulae which are composed of the functions *and* $\sqcap$, *or* $\sqcup$, and *not* $\neg$ as well as the *role quantifiers* which can be *existential* $\exists R$ or *universal* $\forall R$ for some role R. The different kinds of used symbols for expressing the Boolean functions base on the origin of the logics which was disjoined from modal logic as described above. However, the connection to first order logic is immediate but DLs have more efficient decision problems. With respect to the ability to express arbitrary Boolean functions $\mathcal{ALC}$ can be considered as best suitable for the use with Post's lattice due to the availability of $\wedge$, $\vee$, and $\neg$ in this logic. The remarkable part for the decision problem with respect to TBoxes is the following. A TBox $\mathcal{T}$ is said to be consistent for the corresponding model if and only if every axiom in $\mathcal{T}$ is consistent with every world in the model. By virtue of this definition this problem is already complete for exponential time [BBL05a, Hof05] and thus prohibits the existence of a polynomial time algorithm.

More formally the decision problems of interest for DLs are

- the satisfiability problem of TBoxes,
- the concept satisfiability problem with respect to a given TBox,
- the satisfiability problem of an ontology, and
- the subsumption problem with respect to a given TBox.

The latter problem is a special kind of the *implication problem* in the sense of description logics. Further, a method similar to logical deduction which is called *structural comparison* has been deployed but not proven itself to always state correct results. Lacking the completeness it has been shown to be weaker than logical subsumption recently [NB03]. Thus subsumption can be seen as one of the central problems in the area of DLs.

The unrestricted versions of the aforementioned decision problems have been classified previously by their correspondence to propositional dynamic logics [Pra78, VW86, DM00]. To the best of the author's knowledge a complete classification of these problems with respect to all possible Boolean functions has not been done yet and will be the topic of Chapter 4. Especially the study of less commonly used operators as the negation of implication $\not\rightarrow$ or the binary exclusive-or $\oplus$ will give an insight to the influence of Boolean functions on tractability.

The classification of all Boolean function and operator fragments of the concept satisfiability problem for the description logic $\mathcal{ALC}$ immediately follows from the work of Hemaspaandra et al. [HSS08] due to the equivalence to modal logic. They obtained a trichotomy which comprises of complexity degrees from contained in P, through coNP-complete to PSPACE-complete fragments.

1.1.3 Post's Lattice

As motivated above, our approach is to follow Lewis' technique for getting the most fine granulated and complete classification with respect to all possible Boolean functions and operators for each of the decision problems which have been mentioned above. Previously this approach has been followed extensively in the areas of constraint satisfaction [Bau07, Sch07, Sch08], nonmonotonic logics [BMTV09a, Tho09, CMTV10, Tho10], modal and propositional logics [Rei01, HSS08, BMTV09b], abduction, and argumentation [CST10, CSTW10]. These studies have one goal in common. They want to understand which Boolean functions play the role of $\rightarrow\!\!\!\!\!\rightarrow$ in the therein studied extended propositional logics. This is the main goal in this thesis as well.

More formally let B be a finite set of Boolean functions. Then we define the *clone* $[B]$ of B as the set of all Boolean functions which can be constructed by arbitrary composition and projection of functions from B. B is called a *base* of $[B]$ in this context. Post constructed the infinite lattice comprising of all possible clones and proved the existence of a finite base for each of these clones. Usually one aims to achieve a complete classification with respect to Post's lattice. Therefore one needs to overcome the infinity within the lattice by stating matching upper and lower bounds ranging from both ends of the infinite chains in the lattice (see Figure 2.2 on page 16). By definition of the lattice those results state completeness results for any decision problem fragment with respect to each clone within the infinite chain.

1.2 Results

In the first part of Chapter 3 we visit the satisfiability problem of CTL and classify the temporal operator and Boolean fragments. There we show how they form a trichotomy ranging through NP-, PSPACE-, and EXP-complete cases (see Figure 3.6 on page 50) whereas the Boolean fragments, without respect to the temporal operators, lead to TC^0-, NC^1-, and EXP-complete cases (see Figure 3.7 on page 51). Section 3.1.4 aims to describe the problems occurring when working with affine cases which resisted getting fully classified for this decision problem in temporal logic. Furthermore, we will visit extensions of the temporal logics CTL and CTL*, particularly, *(i)* CTL$^+$ which behaves similarly as CTL-SAT (and we also classify in parallel the fragment LTL$^+$) and *(ii)* the fairness extension ECTL where all relevant operator fragments are either PSPACE- or EXP-complete.

The second part of this chapter covers the research on the model checking problems of CTL, CTL*, and the same extensions as above. As the model checking problem for CTL is tractable, and in fact P-complete, we will follow an approach by Sistla and Clarke: we investigate three different kinds of fragments in terms of allowed negation symbols,

starting with monotone, atomic negations only, and positive fragments (for an explicit definition see page 52). The latter one are fragments where operators (not Boolean functions) may not occur in the scope of a negation. Surprisingly, we will show that these three problems are actually computationally equivalent (see Theorem 3.24), and are NC^1-complete if no temporal operator is available, LOGCFL-complete if either we have a non-empty subset of $\{\mathrm{EX}, \mathrm{EF}\}$ or $\{\mathrm{AX}, \mathrm{AG}\}$, and P-complete otherwise. Hence, most fragments of the CTL model checking problem are inherently sequential (see Figure 3.11 on page 73). Thus there is no way to develop parallel algorithms for these cases. While ECTL behaves analogously to CTL, the other extensions exhibit different properties, and their classifications range through six different complexity classes (see Theorem 3.27 and Corollary 3.28). As a starting point for further research, we will achieve a classification for all operator/quantifier fragments of cardinality at most two. We will show how fragments which are easy for this problem, use some CTL-algorithms, and how intractable cases depict parallels to the model checking problem of LTL, in Theorem 3.29.

Finally in Chapter 4 we turn towards the area of description logics, an extension which is widely used by the semantic web community. There we visit all fragments with respect to the possible subsets of the quantifiers $\exists$ and $\forall$, and all Boolean clones. Given a single terminology $\mathscr{T}$ using both quantifiers, we will see how the connected satisfiability problem is either EXP-complete or trivial, i.e., always having satisfiable terminologies. The latter holds if and only if only c-reproducing functions for $c \in \{\top, \bot\}$ are used in $\mathscr{T}$. Allowing only one quantifier turns the fragments which use conjunctions or disjunctions tractable, i.e., P-complete. Without any quantifiers we reach NLOGSPACE-completeness for the fragments using only unary functions. The classification for the decision problems asking about the satisfiability of a concept with respect to a terminology behaves similarly but with two exceptions. First, the $\bot$-reproducing cases are not trivial any longer. Secondly, the lower complexity bounds can be improved to hold without using the constant $\top$. An overview of the results is depicted in Table 4.1 on page 94.

Lastly, in Section 4.2 we classify the implication problem adjusted to description logics, which is the subsumption problem with respect to all quantifier sets and Boolean function sets. There we will show that whenever we are able to express one of the constants besides having access to all quantifiers, the complexity of the fragment remains EXP-complete. By the use of only one quantifier the problem becomes tractable (P-complete) if either conjunctions or disjunctions are allowed—depending on which quantifier is existent. Disallowing quantifiers in general leads to a similar classification as previously has been achieved by Beyersdorff et al. for the propositional implication problem [BMTV09b] with a slight exception for the affine cases involving the function exclusive-or $\oplus$. The complete arrangement in Post's lattice is visualized in Figures 4.1 and 4.2 on pages 105 and 106.

1.3 Publications

Sections 3.1.1 to 3.1.3 have been previously published in [MMTV09] but the proof of Theorem 3.4 (1.) is new. Sections 3.1.4 and 3.1.5 contain unpublished results about the

affine cases and extensions. Section 3.2 contains published results from [BMM+11] but Section 3.2.3 contains unpublished results about fragments of the model checking problem for CTL*. Section 4.1 has been published in [MS11a, MS11b]. Section 4.2 contains new and unpublished results.

Chapter 2

Preliminaries

We assume that the reader is familiar with the standard mathematical notions of functions, the Landau notation (also known as *Big Oh* notation), sets, and propositional logic. For introductory literature we refer the reader to the standard works [End01, Sip05, HMU00]. Whenever we define some expression α as β in this thesis we will write $\alpha \underset{\text{def}}{=} \beta$ in order to denote this fact.

In this work we define 0 as a natural number and denote the set of natural numbers with $\mathbb{N}$, i.e., $\mathbb{N} = \{0, 1, \ldots\}$. A *lattice* $(L, \leq)$ is a partially ordered set in which for any two elements $a, b \in L$ there exists a unique supremum denoted by its *join* $a \cup b$ and an unique infimum denoted by its *meet* $a \cap b$. Such lattices can be visualized via Hasse diagrams as in Figures 2.2 and 2.3.

A Boolean function f is defined over the set $\{\top, \bot\}$, where $\top$ and $\bot$ are abbreviations for the truth values *true* and *false*. Thus an n-ary Boolean function is defined as $f : \{\top, \bot\}^n \to \{\top, \bot\}$ for $n \in \mathbb{N}$. For $n = 0$ we have 0-ary Boolean functions, i.e., the constant functions $\top$ and $\bot$. Further we will make use of the functions *not* $\neg$, *and* $\wedge$, *or* $\vee$, *implication* $\to$, *equivalence* $\leftrightarrow$, and exclusive-or $\oplus$; we will write $x \nrightarrow y$ for the negation of the implication which is defined as $x \nrightarrow y \underset{\text{def}}{=} \neg(x \to y) = x \wedge \neg y$.

Let PL denote the set of all propositional formulae which are defined inductively beginning with the constants $\top, \bot$, variables, and finally through arbitrary compositions of these concepts with Boolean functions. For the lack of space we sometimes will write $\overline{x}$ instead of $\neg x$ for a propositional variable x. For any propositional variable x define the *literals* (of x) as $\neg x$ and x. Let $\mathscr{F}$ be some class of formulae. If $\phi \in \mathscr{F}$ is an $\mathscr{F}$-formula then we denote with $\phi|_{[\alpha_1/\beta_1, \ldots, \alpha_n/\beta_n]}$ the formula that is generated by substituting all α_i in ϕ by β_i for $n \in \mathbb{N}$ where α_i, β_i are some strings for $1 \leq i \leq n$. Additionally we say φ is in *negation normal form* iff negation symbols $\neg$ occur only in front of variables. Further, given a formula $\varphi \in \text{PL}$ let $\mathbf{Vars}(\varphi)$ denote the set of variables contained in φ. Also, we use the notion $\varphi(x_1, \ldots, x_n)$ to denote that $\mathbf{Vars}(\varphi) = \{x_1, \ldots, x_n\}$. Furthermore, given a formula $\varphi \in \text{PL}$ we denote with $\mathbf{SF}(\varphi)$ the set of all sub-formulae of φ including φ itself. Given a propositional formula $\varphi(x_1, \ldots, x_n)$, for $n \in \mathbb{N}$, then an *assignment* is a total function $\theta : \{x_1, \ldots, x_n\} \to \{\top, \bot\}$ that assigns truth values to each variable in φ. Using the signs $\hat{\theta}$ we naturally extend this type of function to work with complex propositional formulae instead of only variables. Therefore we define the function $\hat{\theta} : \text{PL} \to \{\top, \bot\}$

inductively as follows where $\varphi, \varphi_1, \ldots, \varphi_n \in \mathrm{PL}$:

$$\hat{\theta}(x) \underset{\text{def}}{=} \theta(x) \quad \text{for all } x \in \mathbf{Vars}(\varphi),$$
$$\hat{\theta}(f(\varphi_1, \ldots, \varphi_n)) \underset{\text{def}}{=} f(\hat{\theta}(\varphi_1), \ldots, \hat{\theta}(\varphi_n)),$$

for $n \in \mathbb{N}$ and some n-ary Boolean function f.

2.1 Complexity Theory

A language L is a subset of Σ^* for a finite alphabet Σ, and as it suffices to use a binary alphabet for every language L we can conclude $L \subseteq \mathbb{N}$.

In order to define the complexy classes occuring in Chapters 3 and 4, we will make use of the standard model for computation, i.e., Turing machines which we will abbreviate with TM (cf. [Sip05, HMU00, Pap94] for a more elaborative introduction to complexity theory). Let $f : \mathbb{N} \to \mathbb{N}$ be a function. Then we denote with $\mathsf{DTIME}(f)$ (resp., $\mathsf{DSPACE}(f)$) the set of all decision problems (or languages) that can be solved by a *deterministic* Turing machine in time $O(f(n))$, resp., space $O(f(n))$. In the same way we use for the *nondeterministic* case $\mathsf{NTIME}(f)$ and $\mathsf{NSPACE}(f)$. Hence we can define the usual complexity classes

$$\mathsf{LOGSPACE} \underset{\text{def}}{=} \mathsf{DSPACE}(\log(n)),$$
$$\mathsf{P} \underset{\text{def}}{=} \mathsf{DTIME}(n^{O(1)}),$$
$$\mathsf{NP} \underset{\text{def}}{=} \mathsf{NTIME}(n^{O(1)}), \text{ and}$$
$$\mathsf{PSPACE} = \mathsf{NPSPACE} \underset{\text{def}}{=} \mathsf{NSPACE}(n^{O(1)}),$$

where the latter was proven by Savitch in [Sav70]. Counting the number of accepting paths in a computation tree is the quintessence for the class $\oplus\mathsf{LOGSPACE}$ which is defined as the class of decision problems solvable by an $\mathsf{NLOGSPACE}$ machine s.t. the answer is *yes* iff the number of accepting paths is odd. Furthermore we need to define classes above PH and PSPACE (unless $\mathsf{LOGSPACE} = \mathsf{P}$). For $k \in \mathbb{N}$ let $\mathbf{exp}_k(n)$ denote the kth iteration of the exponential function. The complexity class EXP is then defined as $\mathsf{DTIME}(2^{n^{O(1)}}) = \mathsf{DTIME}(\mathbf{exp}_1(n^{O(1)}))$, and one exponential jump farther we define $\mathsf{EEXP} \underset{\text{def}}{=} \mathsf{DTIME}(\mathbf{exp}_2(n^{O(1)}))$.

An *oracle* Turing Machine M is a nondeterministic Turing Machine with three special states $z_+, z_-, z_?$, and an additional oracle band to interact with the predefined oracle B, for a language B. During the computation M may write a word w on the oracle band and change to state $z_?$. *One step* later M enters the state z_+ iff $w \in B$, and z_- iff $w \notin B$, and deletes the oracle band afterwards. If $\mathscr{C}$ is a complexity class and B an oracle, then we denote with $\mathscr{C}^B$ the set of all oracle Turing machines that operate in the class $\mathscr{C}$ and have

access to the oracle B. In this manner we define for two complexity classes $\mathscr{C}, \mathscr{D}$

$$\mathscr{C}^{\mathscr{D}} \underset{\text{def}}{=} \bigcup_{L \in \mathscr{D}} \mathscr{C}^{L}.$$

In the following definition we make use of these Turing Machines to define the polynomial hierarchy PH.

Definition 2.1 (Polynomial time hierarchy).
Let $k \in \mathbb{N}$. Then

$$\Sigma_0^{\mathsf{P}} = \Pi_0^{\mathsf{P}} = \Delta_0^{\mathsf{P}} \underset{\text{def}}{=} \mathsf{P},$$
$$\Sigma_{k+1}^{\mathsf{P}} \underset{\text{def}}{=} \mathsf{NP}^{\Sigma_k^{\mathsf{P}}}, \quad \Delta_{k+1}^{\mathsf{P}} \underset{\text{def}}{=} \mathsf{P}^{\Sigma_k^{\mathsf{P}}}, \quad \Pi_{k+1}^{\mathsf{P}} \underset{\text{def}}{=} \left\{ \mathrm{co}A \mid A \in \Sigma_{k+1}^{\mathsf{P}} \right\},$$
$$\mathsf{PH} \underset{\text{def}}{=} \bigcup_{k \in \mathbb{N}} \Sigma_k^{\mathsf{P}} \cup \Pi_k^{\mathsf{P}} \cup \Delta_k^{\mathsf{P}},$$

where $\mathrm{co}A \underset{\text{def}}{=} \Gamma^* \setminus A$, *for* $A \subseteq \Gamma^*$.

Through this definition one can show that $\Sigma_k^{\mathsf{P}} \cup \Pi_k^{\mathsf{P}} \subseteq \Delta_{k+1}^{\mathsf{P}} \subseteq \Sigma_{k+1}^{\mathsf{P}} \cup \Pi_{k+1}^{\mathsf{P}}$ holds which is visualized in Figure 2.1.

For some results in Chapter 3 that are connected to *promise*[1] problems, or model checking we achieved complexity results for classes deep inside of P. Let LOGCFL denote the class of decision problems that are logspace-reducible[2] to context-free languages. There is also another characterization by a circuit complexity class called SAC^1, i.e., AC^1 with either bounded and-, or or-gates (cf. [Joh90]; for a formal definition of SAC^1 , see [Vol99, Chapter 4.3]).

Circuits. In the following we will define requiried notions from circuit complexity. See also [Vol99] for basic definitions we assume the reader to be familiar with. The relevant classes of this thesis will make use of the bounded base $\mathscr{B}_0 = \{\wedge^2, \vee^2, \neg\}$, and the unbounded base $\mathscr{B}_1 = \{\neg, (\wedge^n)_{n\in\mathbb{N}}, (\vee^n)_{n\in\mathbb{N}}\}$:

$$\mathsf{NC}^i \underset{\text{def}}{=} \mathsf{SIZE\text{-}DEPTH}_{\mathscr{B}_0}(n^{O(1)}, (\log n)^i),$$
$$\mathsf{AC}^i \underset{\text{def}}{=} \mathsf{SIZE\text{-}DEPTH}_{\mathscr{B}_1}(n^{O(1)}, (\log n)^i),$$
$$\mathsf{TC}^i \underset{\text{def}}{=} \mathsf{SIZE\text{-}DEPTH}_{\mathscr{B}_1 \cup \{\mathrm{MAJ}\}}(n^{O(1)}, (\log n)^i),$$

where $\mathsf{SIZE\text{-}DEPTH}_{\mathscr{B}}(s(n), d(n))$ is the class of all sets $A \subseteq \{0,1\}^*$ for which there is a circuit family $\mathscr{C}$ over basis $\mathscr{B}$ of size $O(s(n))$, depth $O(d(n))$ that accepts A, and MAJ

[1] A *promise problem* P is a decision problem with the promise that the input x is syntactically correct, i.e., the membership of x concerning P does not depend on the syntactical structure of x.

[2] A logspace reduction, $\leq_{\mathrm{m}}^{\log}$ in symbols, is a usual reduction that can be computed by a deterministic Turing machine in LOGSPACE.

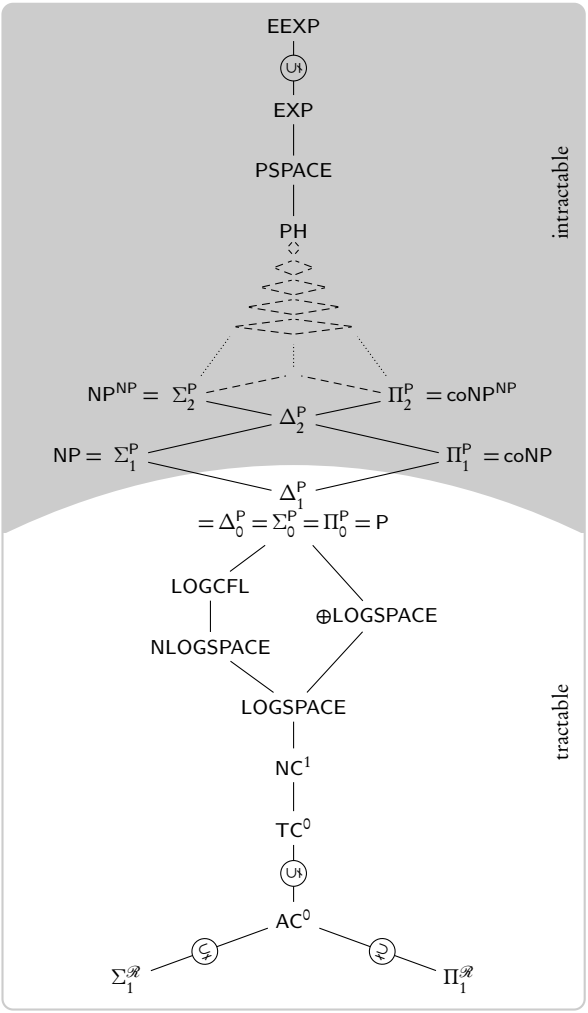


Figure 2.1: Complexity class inclusion diagram. Known strict inclusion are denoted via ⊊.

are gates for the language $\mathrm{MAJ} \underset{\text{def}}{=} \{w \in \{0,1\}^* \mid |w|_1 \geq |w|_0\}$ which is TC^0-complete under $\leq_{\mathrm{cd}}$.

Unless otherwise stated, we are usually working with $\leq_{\mathrm{cd}}$ reductions in this thesis.

Definition 2.2 (Constant depth reductions).
A language A is constant-depth reducible *to B, written $A \leq_{\mathrm{cd}} B$, if there is a logtime-uniform AC^0-circuit family with oracle gates for B that decides membership in A.*

Here, *logtime-uniform* means there is a deterministic TM that can check the structure of the circuit familiy $\mathscr{C}$ in time $O(\log n)$ where n is the size of $\mathscr{C}$.

Classes inside AC^0. The structure of words for some language is the only relevant part for their membership behavior. Consider, e.g., the set of all words which contain at least one 1, hence, the language $\{w \subseteq \{0,1\}^* \mid |w|_1 \geq 1\}$. Thus it suffices to check for a given input $x = x_1 x_2 \ldots x_n$ with $x_i \in \{0,1\}$ whether there is an $1 \leq i \leq n$ s.t. $x_i = 1$. Hence in order to obtain such an i it is sufficient to guess nondeterministically such a position i in x, and accept iff $x_i = 1$.

In this regard a logarithmic time hierarchy can be established wherefore we will now define the complexity class $\Sigma_1^{\mathscr{R}}$ according to proviso ($\mathscr{R}$) in [RV97] as a nondeterministic Turing machine M with a special *index tape* for giving 'random access' to the input word x, where everytime M enters the query state $q_?$ for accessing bit i it is charged 1 time unit for this query and this query may only used once w.l.o.g. at the end of the computation—analogously $\Pi_1^{\mathscr{R}}$ is defined as the 'co-class' of $\Sigma_1^{\mathscr{R}}$.

Theorem 3.3 on page 31 addresses complexity issues in $\Sigma_1^{\mathscr{R}}$, resp., $\Pi_1^{\mathscr{R}}$, where $\leq_{\mathrm{cd}}$-reducibility is of no use since AC^0 forms their $\mathsf{0}$-degree, and $\Sigma_1^{\mathscr{R}} \cup \Pi_1^{\mathscr{R}} \subseteq \mathsf{AC}^0$. Instead, we will make use of the *dlt-projection reducibility* ($A \leq_{\mathrm{proj}}^{\mathrm{dlt}} B$) as introduced in [RV97]. We note that TC^0 and NC^1 are closed under $\leq_{\mathrm{cd}}$, and $\Sigma_1^{\mathscr{R}}$ and $\Pi_1^{\mathscr{R}}$ are closed under $\leq_{\mathrm{proj}}^{\mathrm{dlt}}$.

2.2 Boolean Clones

Since there are infinitely many finite sets of Boolean functions, we introduce some algebraic tools to classify the complexity of the infinitely many arising satisfiability problems. A set B of Boolean functions is called a *clone* if it is closed under superposition, which means that B contains all projections and is closed under arbitrary composition [Pip97]. For a set B of Boolean functions we denote with $[B]$ the smallest clone containing B and call B a *base* for $[B]$. In [Pos41], Post classified the lattice of all clones and found a finite base for each clone (see Figure 2.2). In order to introduce clones, we define the following properties of Boolean functions, where f is an n-ary Boolean function, and $c \in \{\top, \bot\}$.

- f is *c-reproducing* if $f(c, \ldots, c) = c$.
- f is *monotone* if $a_1 \leq b_1, \ldots, a_n \leq b_n$ implies $f(a_1, \ldots, a_n) \leq f(b_1, \ldots, b_n)$.

- f is *c-separating* if there exists an $i \in \{1,\dots,n\}$ such that $f(a_1,\dots,a_n) = 1$ implies $a_i = c$.
- f is *c-separating of degree n* if all $A \subseteq f^{-1}(c)$ with $|A| = n$ are c-separating.
- f is *self-dual* if $f \equiv \mathbf{dual}(f)$, where $\mathbf{dual}(f)(x_1,\dots,x_n) = \neg f(\neg x_1,\dots,\neg x_n)$.
- f is *linear* (or *affine*) if $f(x_1,\dots,x_n) \equiv x_1 \oplus \cdots \oplus x_n \oplus c$.

The list of all clones are shown in Table 2.1, where **id** is the identity function (i.e., $\mathbf{id}(x) = x$ for all x of the respective domain), and $T_n^{n+1} \underset{\text{def}}{=} \bigvee_{i=0}^{n}(x_0 \wedge \cdots \wedge x_{i-1} \wedge x_{i+1} \wedge \cdots \wedge x_n)$ is a threshold function requiring n bits out of $n+1$ set to $\top$.

In this context we extend the definition of PL to PL(B) for a set of Boolean functions B s.t. PL(B) is the set of all propositional formulae with connectives from $[B]$ only. Similarly we define a B-formula φ to contain connectives from $[B]$ only.

Whilst working with Post's lattice in the context of, e.g., a decision problem Γ for propositional logic, or more general, a problem where formulae with Boolean connectives appear one must be careful because of possible blow ups that occur if one want to express a given Boolean function f via some other functions $g_1,\dots,g_k, k \in \mathbb{N}$. In this respect such a problem can occur whenever one wants to show a reduction from $\Gamma(B)$ to $\Gamma(B')$ for two sets of Boolean functions B and B' sucht that $B \subseteq [B']$, and there exist no *short-representation* for a function $f \in B$ with functions in $[B']$.

Example 2.3. *Consider two sets of Boolean functions, $B = \{\oplus\}$ and $B' = \{\wedge,\neg\}$. For writing the function $\oplus$ in B with connectives from B' we obtain $x \oplus y \equiv \neg(\neg(x \wedge \neg y) \wedge \neg(\neg x \wedge y))$. Now let $\phi \in \mathrm{PL}(B)$ with $\phi = x_1 \oplus (x_2 \oplus (\cdots \oplus x_n)\cdots)$ be a formula that has to be written with connectives of B'. Then we obtain a formula $\phi' \in \mathrm{PL}(B')$ with $|\phi'| \in O(2^{|\phi|})$.*

For this reason we establish a lemma for the most used clones in the lattice which utilizes important properties from [Lew79] and [Sch10].

Lemma 2.4 ([Sch10, Lemma 4]).
Let B be a finite set of Boolean functions sucht that $\bot, \top \in B$.

(1.) If $\mathsf{V} \subseteq [B] \subseteq \mathsf{M}$ ($\mathsf{E} \subseteq [B] \subseteq \mathsf{M}$, resp.), then B efficiently implements $\vee$ (resp. $\wedge$), i.e., there exists a B-formula $f(x,y)$ such that f represents $x \wedge y$ ($x \vee y$, resp.) and each of the variables x and y occurs exactly once in $f(x,y)$.

(2.) If $[B] = \mathsf{L}$, then B efficiently implements $\oplus$.

(3.) If $\mathsf{N} \subseteq [B]$, then B efficiently implements $\neg$ via some formula f. If $[B] \subseteq \mathsf{L}$, then f can be chosen in such a way that the variable x occurs in f as the last symbol.

(4.) If $[B] = \mathsf{BF}$, then B efficiently implements $\vee$ and $\wedge$.

Class	Definition	Base
BF	All Boolean functions	$\{x \wedge y, \neg x\}$
R_0	$\{f \mid f \text{ is } \bot\text{-reproducing}\}$	$\{x \wedge y, x \oplus y\}$
R_1	$\{f \mid f \text{ is } \top\text{-reproducing}\}$	$\{x \vee y, x \leftrightarrow y\}$
R_2	$\mathsf{R}_0 \cap \mathsf{R}_1$	$\{\vee, x \wedge (y \leftrightarrow z)\}$
M	$\{f \mid f \text{ is monotone}\}$	$\{x \vee y, x \wedge y, \bot, \top\}$
M_0	$\mathsf{M} \cap \mathsf{R}_0$	$\{x \vee y, x \wedge y, \bot\}$
M_1	$\mathsf{M} \cap \mathsf{R}_1$	$\{x \vee y, x \wedge y, \top\}$
M_2	$\mathsf{M} \cap \mathsf{R}_2$	$\{x \vee y, x \wedge y\}$
S_0	$\{f \mid f \text{ is } \bot\text{-separating}\}$	$\{x \rightarrow y\}$
S_1	$\{f \mid f \text{ is } \top\text{-separating}\}$	$\{x \nrightarrow y\}$
S_0^n	$\{f \mid f \text{ is } \bot\text{-separating of degree } n\}$	$\{x \rightarrow y, \mathbf{dual}(T_n^{n+1})\}$
S_1^n	$\{f \mid f \text{ is } \top\text{-separating of degree } n\}$	$\{x \nrightarrow y, T_n^{n+1}\}$
S_{00}	$\mathsf{S}_0 \cap \mathsf{R}_2 \cap \mathsf{M}$	$\{x \vee (y \wedge z)\}$
S_{00}^n	$\mathsf{S}_0^n \cap \mathsf{R}_2 \cap \mathsf{M}$	$\{x \vee (y \wedge z), \mathbf{dual}(T_n^{n+1})\}$
S_{01}	$\mathsf{S}_0 \cap \mathsf{M}$	$\{x \vee (y \wedge z), \top\}$
S_{01}^n	$\mathsf{S}_0^n \cap \mathsf{M}$	$\{\mathbf{dual}(T_n^{n+1}), \top\}$
S_{02}	$\mathsf{S}_0 \cap \mathsf{R}_2$	$\{x \vee (y \nrightarrow z)\}$
S_{02}^n	$\mathsf{S}_0^n \cap \mathsf{R}_2$	$\{x \vee (y \nrightarrow z), \mathbf{dual}(T_n^{n+1})\}$
S_{10}	$\mathsf{S}_1 \cap \mathsf{R}_2 \cap \mathsf{M}$	$\{x \wedge (y \vee z)\}$
S_{10}^n	$\mathsf{S}_1^n \cap \mathsf{R}_2 \cap \mathsf{M}$	$\{x \wedge (y \vee z), T_n^{n+1}\}$
S_{11}	$\mathsf{S}_1 \cap \mathsf{M}$	$\{x \wedge (y \vee z), \bot\}$
S_{11}^n	$\mathsf{S}_1^n \cap \mathsf{M}$	$\{T_n^{n+1}, \bot\}$
S_{12}	$\mathsf{S}_1 \cap \mathsf{R}_2$	$\{x \wedge (y \rightarrow z)\}$
S_{12}^n	$\mathsf{S}_1^n \cap \mathsf{R}_2$	$\{x \wedge (y \rightarrow z), T_n^{n+1}\}$
D	$\{f \mid f \text{ is self-dual}\}$	$\{(x \nrightarrow y) \vee (x \nrightarrow z) \vee (\overline{y} \nrightarrow z)\}$
D_1	$\mathsf{D} \cap \mathsf{R}_2$	$\{(x \nrightarrow \overline{y}) \vee (x \nrightarrow z) \vee (y \nrightarrow z)\}$
D_2	$\mathsf{D} \cap \mathsf{M}$	$\{(x \nrightarrow \overline{y}) \vee (x \nrightarrow \overline{z}) \vee (y \nrightarrow \overline{z})\}$
L	$\{f \mid f \text{ is linear}\}$	$\{x \oplus y, \top\}$
L_0	$\mathsf{L} \cap \mathsf{R}_0$	$\{x \oplus y\}$
L_1	$\mathsf{L} \cap \mathsf{R}_1$	$\{x \leftrightarrow y\}$
L_2	$\mathsf{L} \cap \mathsf{R}_2$	$\{x \oplus y \oplus z\}$
L_3	$\mathsf{L} \cap \mathsf{D}$	$\{x \oplus y \oplus z \oplus \top\}$
V	$\{f \mid f \text{ is a disjunction or constant}\}$	$\{x \vee y, \bot, \top\}$
V_0	$\mathsf{M}_0 \cap \mathsf{V}$	$\{x \vee y, \bot\}$
V_1	$\mathsf{M}_1 \cap \mathsf{V}$	$\{x \vee y, \top\}$
V_2	$\mathsf{M}_2 \cap \mathsf{V}$	$\{x \vee y\}$
E	$\{f \mid f \text{ is a conjunction or constant}\}$	$\{x \wedge y, \bot, \top\}$
E_0	$\mathsf{M}_0 \cap \mathsf{E}$	$\{x \wedge y, \bot\}$
E_1	$\mathsf{M}_1 \cap \mathsf{E}$	$\{x \wedge y, \top\}$
E_2	$\mathsf{M}_2 \cap \mathsf{E}$	$\{x \wedge y\}$
N	$\{f \mid f \text{ depends on at most one variable}\}$	$\{\neg x, \bot, \top\}$
N_2	$\mathsf{L}_3 \cap \mathsf{N}$	$\{\neg x\}$
I	$\{f \mid f \text{ is a projection or a constant}\}$	$\{\mathbf{id}, \bot, \top\}$
I_0	$\mathsf{R}_0 \cap \mathsf{I}$	$\{\mathbf{id}, \bot\}$
I_1	$\mathsf{R}_1 \cap \mathsf{I}$	$\{\mathbf{id}, \top\}$
I_2	$\mathsf{R}_2 \cap \mathsf{I}$	$\{\mathbf{id}\}$

Table 2.1: A list of Boolean clones with definitions and bases.

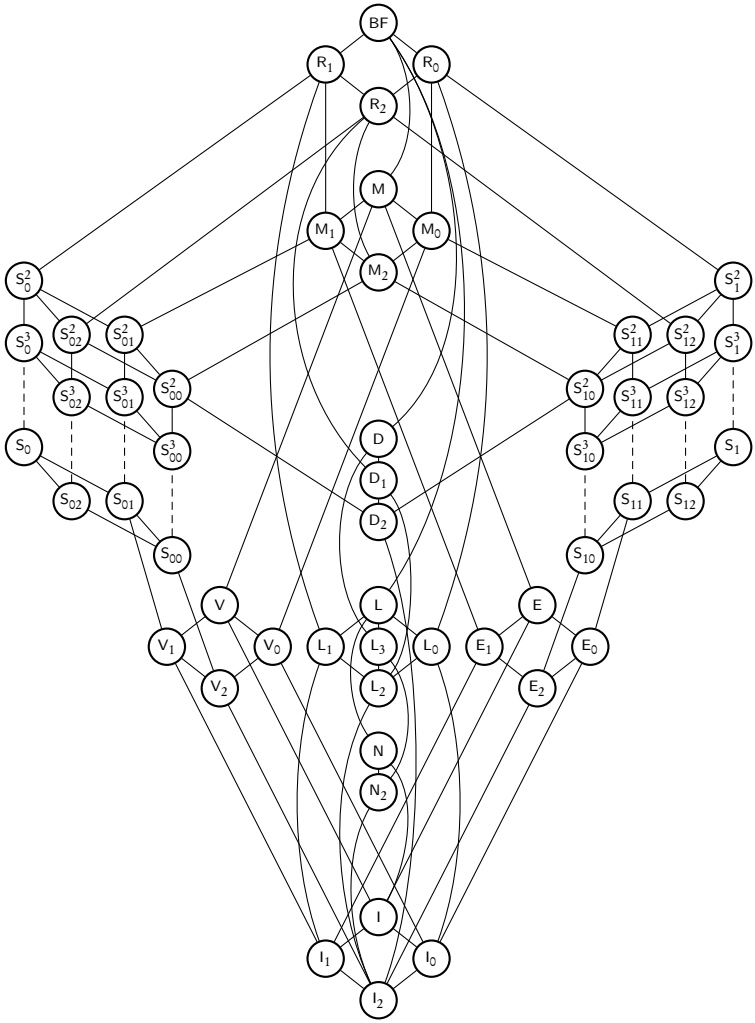


Figure 2.2: Post's lattice of all Boolean clones.

2.3 Modal Logic

Modal logic can be dated back to very early formal approaches of Aristoteles. If propositional logic is the logic that talks about one state (world, individual), then the extension of propositional logic, which is called *modal logic*, overcomes this lack of expressivity. Therefore one new operator $\Diamond$ is introduced which models this fact. Thus the syntax of modal logic is defined by the following grammar:

$$\varphi ::= \top \mid x \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \Diamond\varphi,$$

where x is an *atomic proposition* which is an element of the set PROP that are denoted with uncapitalized letters as x, y, z, p, q. As usual it holds that $\bot \underset{\text{def}}{=} \neg\top$, and $\Box\varphi$ is interpreted as $\neg\Diamond\neg\varphi$. The set of all modal formulae is denoted with ML. Further let ML(B) be the set of all modal formulae that only use connectives from the clone $[B]$ of set of Boolean functions B, and ML($B, \mathscr{Q}$) be the set of all ML(B)-formulae that only use modalities from $\mathscr{Q} \subseteq \{\Box, \Diamond\}$. Before being able to define their semantics, we need to define Kripke structures which are the transition system of choice.

Definition 2.5 (Kripke structure).
A Kripke structure *is a quadruple*

$$\mathfrak{K} = (W, \mathscr{R}, \eta, \mathsf{PROP}),$$

where W is a set of worlds, $\mathscr{R} = \{R_1, R_2, \ldots, R_n\}$ is a set of transition relations, and $\eta\colon \mathsf{PROP} \to \mathfrak{P}(W)$ is a labeling function that associates atomic propositions to sets of worlds to denote in which worlds they are labeled.

Remark 2.6. *Definition 2.5 extends the grammar from above. Whenever we write $\Diamond_i$, resp., $\Box_i$ we refer to the transition relation $R_i \in \mathscr{R}$ for a given Kripke structure $\mathfrak{K}$. Additionally, whenever* PROP *is either not relevant or if the situation makes a definition obvious we usually omit an explicit statement.*

Finally the semantics of modal formulae are defined with respect to Kripke structures as follows.

Definition 2.7 (Semantics).
Let $\varphi, \psi \in$ ML be some modal formulae, $\mathfrak{K} = (W, \mathscr{R}, \eta, \mathsf{PROP})$ be a Kripke structure, and $w, w' \in W$. Then

$\mathfrak{K}, w \models \top$	*always holds,*	
$\mathfrak{K}, w \models x$	*iff*	$x \in \mathsf{PROP}$ *and* $w \in \eta(x)$,
$\mathfrak{K}, w \models \neg\varphi$	*iff*	$\mathfrak{K}, w \not\models \varphi$,
$\mathfrak{K}, w \models (\varphi \wedge \psi)$	*iff*	$\mathfrak{K}, w \models \varphi$ *and* $\mathfrak{K}, w \models \psi$, *and*
$\mathfrak{K}, w \models \Diamond_i\varphi$	*iff*	$\mathfrak{K}, w' \models \varphi$ *and* $(w, w') \in R_i$ *for* $R_i \in \mathscr{R}$.

We say a ML($B, \mathscr{Q}$) formula φ is *satisfiable* iff there is a Kripke structure $\mathfrak{K} = (W, \mathscr{R}, \eta,$ PROP) s.t. $\mathfrak{K}, w \models \varphi$ for some $w \in W$.

A *frame* is a class of structures with a certain property. The two relevant frames for this thesis with their properties and their respective names are

$\mathfrak{F}_{\text{all}}$ the set of all frames.

$\mathfrak{F}_{\text{total}}$ the set of all total frames, i.e., for $\mathfrak{K} = (W, \mathscr{R}, \eta)$, and all $w \in W$ and all $R_i \in \mathscr{R}$ there is a w' s.t. $(w, w') \in R_i$ (each world has for each transition relation a successor).

Of course there are other frames which are investigated in the literature, e.g., *transitive*, or *total transitive* frames. In this thesis we employ only to the two defined above because they simulate the behavior of computing systems the best. Now we will take a closer look at an example that illustrates the expressiveness of a very specific frame class which combines two very fundamental properties of frames.

Example 2.8. *For the frame class of all transitive*[3] *and irreflexive*[4] *structures one can easily construct a formula whose satisfying structures require infinite many states. Therefore consider the formula* $\varphi = \Diamond\top \wedge \Box\Diamond\top$ *which says, informally, that every world has a successor. Both conjuncts are important to express this property because without* $\Diamond\top$ *we could construct a world without any successor which also fulfills the semantics of the* $\Box$*-preceded formula. Thus for any Kripke structure* $\mathfrak{K} = (W, \mathscr{R}, \eta, \mathsf{PROP})$ *with* $\mathfrak{K} \models \varphi$ *it holds that* $|W| = \infty$.

The most prominent decision problem for this logic is of course the satisfiability problem

Problem (ML-SAT$_{\mathscr{Q}}(B)$)
Input: an ML$(B, \mathscr{Q})$-formula φ.
Question: is ϕ satisfiable?

which has been proven to be PSPACE-complete in 1977 by Ladner:

Theorem 2.9 ([Lad77]).
ML-SAT$_{\{\Box,\Diamond\}}(\mathsf{BF})$ *is* PSPACE*-complete.*

For ease of notion we will omit the braces from now on and will write ML-SAT$_{\Box\Diamond}(B)$ instead of ML-SAT$_{\{\Box,\Diamond\}}(B)$. For more information about modal logics and their properties we refer the reader to [BdV01].

2.3.1 Temporal Logic

Temporal logic has a huge influence to computer science, in particular in the areas of artificial intelligence and program verification, but also in a broad field of other sciences such as physics, ethics and philosophy [Pri67, Pnu77, Krö87, Gal87, ØH95].

There are several different extensions to temporal logics available starting with hybridizations [KWLS09, Web09a] or probabilistic versions [HJ94, RKNP04]. In this thesis we concentrate on the temporal logic that is the most close to modal logic, which is the computation tree logic CTL*. In this area of modal logic one can model several different kinds of computational behavior, e.g., properties that must hold on some computation path invariantly, sometimes, or eventually. Now let PROP be a finite set of atomic propositions.

[3] if $(u, v), (v, w) \in E$ then also $(u, w) \in E$
[4] $(v, v) \notin E$ for all $v \in V$

The symbols used are the atomic propositions in PROP, the constant symbols $\top$ and $\bot$, the Boolean connectives $\wedge$ and $\neg$, and the temporal operator symbols X, U, and A. A is also called a *path quantifier*, temporal operators aside from path quantifiers are also called *pure temporal operators*.

The atomic propositions, $\top$ and $\bot$ are called *atomic formulae*. There exist two other kinds of formulae, *state formulae* and *path formulae*. Each atomic formula is a state formula. Let φ, ψ be state formulae and χ, π be path formulae. Then $\neg\varphi$, $(\varphi \wedge \psi)$, $\mathrm{A}\chi$ are state formulae, and φ, $\neg\chi$, $(\chi \wedge \pi)$, $\mathrm{X}\chi$, and $[\chi \mathrm{U} \pi]$ are path formulae. The set of CTL*-formulae consists of all path formulae χ, and is equivalently defined via the grammar

$$\varphi ::= \top \mid \bot \mid p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \mathrm{A}\chi,$$
$$\chi ::= \varphi \mid \neg\chi \mid (\chi \wedge \chi) \mid \mathrm{X}\chi \mid [\chi \mathrm{U} \chi],$$

where p is an atomic proposition. In this definition φ describe state formulae and χ path formulae. We define $\mathrm{CTL}^\star(T, B)$ to be the set of CTL*-formulae using Boolean connectives in B and temporal operators in T only. For ease of notation, we will also write $\mathrm{CTL}^\star(T, B)$ if B is a clone and identify B with an arbitrary finite base for B. If unary or associative functions occur then the braces '(' and ')' may be omitted. Further, observe that for complexity classes above and including PSPACE, the choice of this base is irrelevant as for any Boolean function f appearing, one can substitute the use of f recursively by its respective clause of the truth-table constructed disjunctive normal form representation.

In temporal logic we often refer to *models* instead of Kripke structures (which is literally the same), and usually write $M = (S, \{R\}, l)$ for such a model, where l is the labeling function which corresponds to η in the modal world for Kripke structures $\mathfrak{K} = (W, R, \eta)$. Often, l is defined as a function $l : S \to \mathfrak{P}(\mathrm{PROP})$ and therefore maps states to sets of propositions contrary to η mapping propositions to sets of states. The models in temporal logic are always defined over the frame $\mathfrak{F}_{\mathrm{total}}$. Hence, as such models generally consist of only one transition relation R, we commonly just write $M = (S, R, l)$ and omit the $\{\}$ braces. A *path* x is an infinite sequence $x = (x_0, x_1, \ldots) \in S^\omega$ such that $(x_i, x_{i+1}) \in R$ for all $i \geq 0$. For a path $x = (x_0, x_1, \ldots)$ we denote by x^i the path $(x_i, x_{i+1}, \ldots)$. Let $M = (S, R, l)$ be a model, $s \in S$ be a state, and $x = (x_0, x_1, \ldots) \in S^\omega$ be a path. The truth of a CTL*-formula w.r.t. M is inductively defined using the following semantics.

Definition 2.10 (Semantics of CTL*).
Let $\varphi, \psi, \chi, \pi \in \mathrm{CTL}^\star$ for state formulae φ, ψ, path formulae χ, π, and an infinite path $x = (x_1, x_2, \ldots)$.

$M, s \models \top$		*always holds,*
$M, s \models \bot$		*never holds,*
$M, s \models p$	*iff*	$p \in \mathrm{PROP}$ *and* $p \in l(s)$,
$M, s \models \neg\varphi$	*iff*	$M, s \not\models \varphi$,
$M, s \models (\varphi \wedge \psi)$	*iff*	$M, s \models \varphi$ *and* $M, s \models \psi$,
$M, s \models \mathrm{A}\chi$	*iff*	*for all paths* $x = (s, x_2, x_3, \ldots)$ *holds* $M, x \models \chi$,
$M, x \models \varphi$	*iff*	$M, x_1 \models \varphi$,

$$
\begin{array}{lll}
M,x \models \neg\chi & \textit{iff} & M,x \not\models \chi, \\
M,x \models (\chi \wedge \pi) & \textit{iff} & M,x \models \chi \textit{ and } M,x \models \pi, \\
M,x \models \mathrm{X}\chi & \textit{iff} & M,x_2 \models \chi, \\
M,x \models [\chi \mathrm{U} \pi] & \textit{iff} & M,x^k \models \pi \textit{ for some } k \in \mathbb{N}, \textit{ and} \\
 & & M,x^i \models \chi \textit{ for all } 1 \leq i < k.
\end{array}
$$

All remaining Boolean functions f can be defined in terms of the connectives $\neg$ and $\wedge$. The other temporal operators are defined as usual: $\mathrm{E}\varphi \equiv \neg\mathrm{A}\neg\varphi, \mathrm{F}\varphi \equiv \top\mathrm{U}\varphi, \mathrm{G}\varphi \equiv \neg\mathrm{F}\neg\varphi$, where E is again also called a path quantifier. A formula φ is hence said to be *satisfied by model M* if there exists an $x \in S^\omega$ such that $M,x \models \varphi$ (written as $M \models \varphi$). Further, φ is said to be *satisfiable* if there exists a model M that satisfies φ.

Problem (CTL*-SAT(T,B))
Input: a CTL$^*(T,B)$-formula φ.
Question: is there a model $M=(S,R,l)$ s.t. there exists a state $s \in S$ with $M,s \models \varphi$?

Theorem 2.11 ([VS85a, VS85b, EJ00]).
CTL*-SAT(ALL, BF) *is* EXPTIME*-complete under* $\leq_{\mathrm{cd}}$*-reductions.*

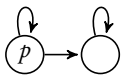
Talking about the existence of a satisfying model for a given formula is joined with asking whether a given model satisfies a given formula, i.e., the model checking problem.

Problem (CTL*-MC(T,B))
Input: a CTL$^*(T,B)$-formula φ, a model $M=(S,R,l)$.
Question: is there a state $s \in S$ s.t. $M,s \models \varphi$?

Theorem 2.12 ([CES86]).
CTL*-MC(ALL, BF) *is* PSPACE*-complete under* $\leq_{\mathrm{cd}}$*-reductions.*

A CTL*-formula* is a CTL*-formula in which each path quantifier is followed by exactly one pure temporal operator and each pure temporal operator is preceded by exactly one path quantifier. Here, we define a CTL*-operator* as every combination of a path quantifier and a pure temporal operator. The set of CTL-formulae forms a strict subset of the set of CTL*-formulae which is illustrated through the following example.

Example 2.13. $\mathrm{E}(\mathrm{G}p \wedge \mathrm{X}\neg p)$ *is a* CTL**-formula which is not satisfiable. Further it is not expressible with a* CTL*-formula. Also, this formula shows that the path quantifier* E *is not distributive, as the formula* $\mathrm{EG}p \wedge \mathrm{EX}\neg p$ *is satisfied via the model*



as in an infinite path looping the first node satisfies $\mathrm{EG}p$ *and the succeeding state fulfills* $\neg p$.

Remark 2.14. *Let B be a finite set of Boolean functions and let $\varphi \in \mathrm{ML}(B)$ be some modal formula. Then there exists a connection to temporal logic in the following way. It holds that $\varphi \in$ ML-SAT(B) if and only if $\varphi' \in$ CTL-SAT$(\{\mathrm{AX},\mathrm{EX}\},B)$, where $\varphi' \underset{\mathrm{def}}{=} \varphi|_{[\Box/\mathrm{AX},\Diamond/\mathrm{EX}]}$.*

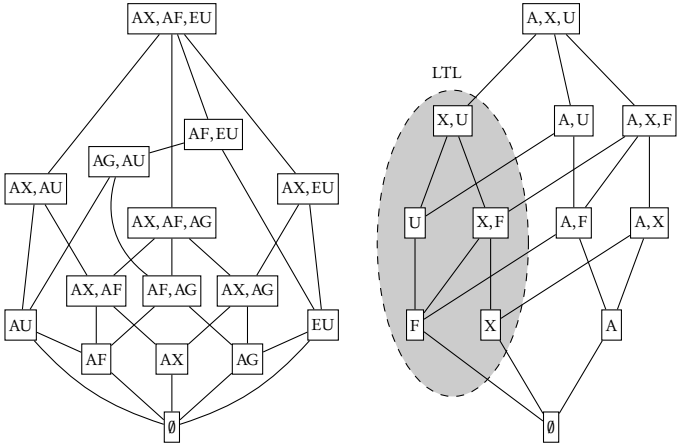


Figure 2.3: The lattice induced by all CTL- (left) and all CTL*-operators (right). Each node is labeled with a minimal set of operators without any restrictions on the Boolean connectives.

Pairs of path quantifiers and pure temporal operators are also referred to as CTL-*operators*.

Remark 2.15. *The following equivalences among* CTL*-operators hold:*

$$\mathrm{EX}\varphi \equiv \neg\mathrm{AX}\neg\varphi, \qquad \mathrm{EF}\varphi \equiv \mathrm{E}[\top \mathrm{U} \varphi],$$
$$\mathrm{AF}\varphi \equiv \mathrm{A}[\top \mathrm{U} \varphi], \qquad \mathrm{AG}\varphi \equiv \neg\mathrm{EF}\neg\varphi,$$
$$\mathrm{EG}\varphi \equiv \neg\mathrm{AF}\neg\varphi, \qquad \textit{and } \mathrm{A}[\psi \mathrm{U} \chi] \equiv \mathrm{AF}\chi \wedge \neg\mathrm{E}[\neg\chi \mathrm{U} (\neg\psi \wedge \neg\chi)].$$

Hence, in presence of all Boolean connectives, $\{\mathrm{AX}, \mathrm{AF}, \mathrm{EU}\}$ *is a minimal set of* CTL*-operators for* CTL*, whereas* $\{\mathrm{AX}, \mathrm{AG}, \mathrm{AU}\}$ *is not [Lar95].*

In Figure 2.3 one can see how the CTL- and CTL*-operators form a lattice.

Alike CTL*-SAT, we define $\mathrm{CTL}(T, B)$ to be the set of all CTL-formulae using Boolean connectives in B and CTL-operators in T, and define CTL-SAT(T, B) to be the problem of deciding whether a given $\mathrm{CTL}(T, B)$-formula is satisfiable. The corresponding decision problems for this strictly more restricted logic are

Problem (CTL-SAT(T, B))
Input: a $\mathrm{CTL}(T, B)$ formula φ.
Question: Is there a model $M = (S, R, l)$ s.t. there exists a state $s \in S$ with $M, s \models \varphi$?

and

Problem (CTL-MC(T, B))
Input: a $\mathrm{CTL}(T, B)$ formula φ, a model $M = (S, R, l)$.

Question: Is there a state $s \in S$ s.t. $M, s \models \varphi$?

Theorem 2.16 ([FL79, Pra80]).
CTL-SAT(ALL, BF) *is* EXP*-complete under* $\leq_m^{\log}$.

Theorem 2.17 ([CES86, Sch02]).
CTL-MC(ALL, BF) *is* P*-complete under* $\leq_m^{\log}$.

The disability of expressing fairness properties is a lack of CTL which has been overcome by introducing ECTL in [EH86]. Therefore a new temporal operator $\overset{\infty}{\mathrm{F}}$ is introduced for a model $M = (S, R, l)$, a path $x = (x_1, x_2, \ldots)$, and a path formula χ as

$$M, x \models \overset{\infty}{\mathrm{F}}\chi \quad \text{iff} \quad M, x \models \mathrm{GF}\chi,$$

where $\overset{\infty}{\mathrm{G}}\psi \equiv \mathrm{FG}\psi$ denotes the analogously defined dual operator. Another fragment of CTL* which has been defined in [EH86] is the logic CTL$^+$ which is an extension of CTL that allows Boolean combinations (without nesting) of temporal operators in path formulae under the scope of a path quantifier. From the point of expressiveness the logics are the same whereas a translation between these logics leads to an exponential blow up in the size of the given formula [EH85, Wil99, AI03]. One can easily observe that CTL* with fairness constraints, i.e., the logic ECTL* equals CTL* due to the equivalence of $\overset{\infty}{\mathrm{F}}$ with GF. In the same way as before we define their respective decision problems.

Theorem 2.18 (follows from Theorem 2.16 and [Eme90]).
ECTL-SAT(ALL, BF) *is* EXP*-complete under* $\leq_m^{\log}$.

Theorem 2.19 ([Sch02]).
ECTL-MC(ALL, BF) *is* P*-complete under* $\leq_m^{\log}$.

Theorem 2.20 ([VS85a, EJ00]).
ECTL$^+$-SAT(ALL, BF) *is* EEXP*-complete under* $\leq_m^{\log}$.

Theorem 2.21 ([LMS01]).
ECTL$^+$-MC(ALL, BF) *is* Δ_2^{P}*-complete under* $\leq_m^{\log}$.

Theorem 2.22 ([JL03]).
CTL$^+$-SAT(ALL, BF) *is* EEXP*-complete under* $\leq_m^{\log}$.

Theorem 2.23 ([LMS01]).
CTL$^+$-MC(ALL, BF) *is* Δ_2^{P}*-complete under* $\leq_m^{\log}$.

Example 2.24. *In the fragment of* CTL *with only* AX *and* AF *as allowed operators besides all Boolean functions one can construct a binary counter that enforces paths of exponential*

length in every satisfying model. Consider the set $\Phi \underset{\text{def}}{=} \{q_i \mid 1 \leq i \leq n\}$ *as atomic propositions in the given formula* φ_{cnt} *which is defined as*

$$\varphi_{cnt} \underset{\text{def}}{=} \bigwedge_{i=1}^{n} \neg q_i \wedge \mathrm{EG}\Bigg(\bigwedge_{i=1}^{n}\Bigg[\Big(\bigwedge_{k=1}^{i} q_k \rightarrow \mathrm{AX}\bigwedge_{j=1}^{i} \neg q_j\Big)\Bigg) \wedge \\ \wedge\Big(\bigwedge_{k=i+1}^{n} (q_k \rightarrow \mathrm{AX}q_k) \wedge (\neg q_k \rightarrow \mathrm{AX}\neg q_k)\Big)\Bigg] \wedge \\ \wedge\Bigg(\Big(\bigwedge_{i=1}^{n} q_i\Big) \rightarrow \mathrm{AX}\Big(\bigwedge_{i=1}^{n} \neg q_i\Big)\Bigg)\Bigg),$$

where empty conjunctions are assumed to be $\top$*. Starting in a state where none proposition is true the formula enforces in the first big conjunction to flip all bits until and including the* i*-th bit whereas all more significant bits remain their value. This clearly models the behavior of a counter whence every satisfying model comprehends of an exponential long path rippling through all binary numbers. Substituting the conclusion in the last implication with a contradiction, e.g., saying* $\neg q_1$ *holds in this state, then constructing a model in the naïve way bears this conflict only after exponential many steps. Observe that this kind of construction with solely* AF *instead of* AX *is not possible.*

2.3.2 Description Logic

There are many different types of description logics which are applied in the areas of the semantic web, object-oriented representations, but also at type systems and medical ontologies [BCM+03]. Thus any such logical derivative is trimmed for its recent purpose. Some of these logics exhibit tractable algorithms for several decision problems and therefore find practical application, e.g., $\mathcal{EL}^{++}$ which admits sound and complete reasoning in polynomial time [BBL05a, BBL05b, BBL08].

In the area of description logics there are several different names for concepts that are also available in modal logic, i.e., *individuals* for worlds/nominals, *concepts* for unary relations, *roles* for binary relations/modalities, *concept descriptions* for formulae. For that reason we will keep these naturalized notions and names whenever we are in the scope of the description logics. In fact, this correspondence to the multimodal logic over the frame class $\mathfrak{F}_{\text{all}}$(cf. [HM92]) has been pointed out by Schild already [Sch91]. As these equivalence of some description logics to modal logic has been discovered several years after the first definitions definitions in the 1970s the use of $\sqcap$ and $\sqcup$ have been become the de facto standard symbols corresponding to conjunction and disjunction in the sense of set operations. For this reason we will stick to these symbols in order to avoid misunderstandings. For all remaining Boolean connectives, e.g., $\oplus$, $\rightarrow$, $\nrightarrow$, we will use the same symbols.

Our approach in finding a description logic which interacts closest with Post's lattice leads to the description logic $\mathcal{ALC}$ whose operators comply to the Boolean standard

base of BF. Usually the definition of a description logic starts with the set $\mathsf{N_C}$ of atomic concepts, the universal concept $\top$, the empty concept $\bot$, and the set $\mathsf{N_R}$ of all roles. The syntax for all concept descriptions in $\mathcal{ALC}$ are usually inductively defined as

$$C ::= A \mid \top \mid \bot \mid \neg C \mid C \sqcap C \mid \exists R.C,$$

where $A \in \mathsf{N_C}$, and $R \in \mathsf{N_R}$. In order to work with Boolean clones we adjust this definition to

$$C ::= A \mid \top \mid \bot \mid \circ_f(C,\ldots,C) \mid \exists R.C,$$

where A and R are as from above, but $\circ_f$ is the operator which corresponds to the Boolean function f. In the following we will define the semantics of this logic in the same notation.

Definition 2.25 ($\mathcal{ALC}$ semantics).
An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *is a finite, not empty set* $\Delta^{\mathcal{I}}$ *of individuals and a mapping* $\cdot^{\mathcal{I}}$ *which assigns*

- *atomic concepts from* $\mathsf{N_C}$ *and individuals from* $\mathsf{N_I}$ *to* $\mathfrak{P}(\Delta^{\mathcal{I}})$, *and*
- *roles from* $\mathsf{N_R}$ *to* $\mathfrak{P}(\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}})$.

This interpretation $\mathcal{I}$ *is extended to arbitrary concepts via*

$$\circ_f(C_1,\ldots,C_m)^{\mathcal{I}} \underset{\text{def}}{=} \left\{ x \in \Delta^{\mathcal{I}} \;\middle|\; f(\|x \in C_1^{\mathcal{I}}\|,\ldots,\|x \in C_n^{\mathcal{I}}\|) = 1 \right\},$$

where $\|x \in C_j^{\mathcal{I}}\| = 1$ *if* $x \in C_j^{\mathcal{I}}$ *and* $\|x \in C_j^{\mathcal{I}}\| = 0$ *if* $x \notin C_j^{\mathcal{I}}$, *and*

$$(\exists R.C)^{\mathcal{I}} \underset{\text{def}}{=} \left\{ x \in \Delta^{\mathcal{I}} \;\middle|\; \left\{ y \in C^{\mathcal{I}} \;\middle|\; (x,y) \in R^{\mathcal{I}} \right\} \neq \emptyset \right\}.$$

Further, let B be a finite set of Boolean functions and $\mathcal{Q} \subseteq \{\exists, \forall\}$ be a set of quantifiers. Then, define $\mathsf{Con}_{\mathcal{Q}}(B)$ as the set of concepts which only use quantifiers from $\mathcal{Q}$ and operators that correspond to Boolean functions from $[B]$. The satisfiability problem for concept expressions is defined as follows:

Problem ($\mathrm{CSAT}_{\mathcal{Q}}(B)$)
Input: a concept description $C \in \mathsf{Con}_{\mathcal{Q}}(B)$.
Question: is there an interpretation $\mathcal{I}$ s.t. $C^{\mathcal{I}} \neq \emptyset$?

One can easily observe that the connection to modal logic is immediate because $\mathrm{CSAT}_{\mathcal{Q}}(B) \equiv^{\log}_{\mathrm{m}} \text{ML-SAT}_{\mathcal{Q}'}(B)$ holds via f where $\mathcal{Q} \subseteq \{\forall, \exists\}, \mathcal{Q}' = \{\Box, \Diamond\}$, and the reduction function $f : \mathrm{DL} \to \mathrm{ML}$ is defined inductively as follows

$$f(C) \underset{\text{def}}{=} x_C, \qquad f(\top) \underset{\text{def}}{=} \top, \qquad f(\bot) \underset{\text{def}}{=} \bot,$$
$$f(\circ_g(C_1,\ldots,C_n)) \underset{\text{def}}{=} g(f(C_1),\ldots,f(C_n)),$$
$$f(\forall R.C) \underset{\text{def}}{=} \Box_R f(C), \qquad f(\exists R.C) \underset{\text{def}}{=} \Diamond_R f(C)$$

for any atomic concept C, and any operator $\circ_g$ which corresponds to the Boolean function g.

Thus the complexity classification for all quantifier- and function-fragments immediately arises out of the work of Hemaspaandra et al.:

Theorem 2.26 ([HSS08]).
Let B be a finite set of Boolean operators.

(1.) If $\mathsf{S}_{11} \subseteq [B]$, then $\mathrm{CSAT}_{\exists\forall}(B)$ is PSPACE*-complete.*

(2.) If $[B] \in \{\mathsf{E}, \mathsf{E}_0\}$, then $\mathrm{CSAT}_{\exists\forall}(B)$ is coNP*-complete.*

(3.) If $[B] \subseteq \mathsf{R}_1$, then $\mathrm{CSAT}_{\exists\forall}(B)$ is trivial.

(4.) Otherwise $\mathrm{CSAT}_{\exists\forall}(B) \in \mathsf{P}$.

(5.) If $\mathsf{S}_1 \subseteq [B]$, then $\mathrm{CSAT}_{\exists}(B)$ and $\mathrm{CSAT}_{\forall}(B)$ are PSPACE*-complete.*

(6.) If $[B] \subseteq \mathsf{R}_1$, then $\mathrm{CSAT}_{\exists}(B)$ and $\mathrm{CSAT}_{\forall}(B)$ are trivial.

(7.) Otherwise $\mathrm{CSAT}_{\exists}(B) \in \mathsf{P}$ and $\mathrm{CSAT}_{\forall}(B) \in \mathsf{P}$.

Definition 2.27 (GCI, TBox, ABox, Ontology).
Let C, D be concepts and R be a role. Then we define a general concept inclusion (GCI) *as an axiom of the form $C \sqsubseteq D$. Further we will write $C \equiv D$ abbreviating $C \sqsubseteq D$ and $D \sqsubseteq C$. A* TBox *is a finite set of GCIs without restrictions. An* ABox *is a finite set of axioms of the form $C(x)$ or $R(x, y)$. Lastly, the union of a TBox and an ABox is called an* ontology.

Let B be a finite set of Boolean functions and $\mathcal{Q}$ be a set of quantifiers. With $\mathfrak{T}_{\mathcal{Q}}(B)$ and $\mathfrak{O}_{\mathcal{Q}}(B)$ we denote the set of all TBoxes and ontologies using operators corresponding to functions in $[B]$ and quantifiers from $\mathcal{Q}$.

For two given concepts C, D we say an interpretation $\mathcal{I}$ *satisfies* an axiom $C \sqsubseteq D$, in symbols $\mathcal{I} \models C \sqsubseteq D$, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. Furthermore, given a concept C and a role R, we say $\mathcal{I}$ satisfies $C(x)$ or $R(x, y)$ if $x^{\mathcal{I}} \in C^{\mathcal{I}}$ or $(x^{\mathcal{I}}, y^{\mathcal{I}}) \in R^{\mathcal{I}}$. Finally, an interpretation $\mathcal{I}$ satisfies a TBox (ABox, ontology) if it satisfies every axiom therein. It is then also called a *model* of this set of axioms. Now we are ready to define the appropriate decision problems:

Problem ($\mathrm{TSAT}_{\mathcal{Q}}(B)$)
Input: a TBox $\mathcal{T} \in \mathfrak{T}_{\mathcal{Q}}(B)$.
Question: is there an interpretation $\mathcal{I}$ which is a model for $\mathcal{T}$?

Problem ($\mathrm{TCSAT}_{\mathcal{Q}}(B)$)
Input: a TBox $\mathcal{T} \in \mathfrak{T}_{\mathcal{Q}}(B)$ and a concept $C \in \mathsf{Con}_{\mathcal{Q}}(B)$.
Question: is there an interpretation $\mathcal{I}$ which is a model for $\mathcal{T}$ and $C^{\mathcal{I}} \neq \emptyset$?

Problem ($\mathrm{OSAT}_{\mathcal{Q}}(B)$)
Input: an ontology $\mathcal{O} \in \mathfrak{O}_{\mathcal{Q}}(B)$.

Question: is there an interpretation $\mathcal{I}$ which is a model for $\mathcal{O}$?

Problem (OCSAT$_{\mathcal{Q}}(B)$)
Input: an ontology $\mathcal{O} \in \mathfrak{O}_{\mathcal{Q}}(B)$ and a concept $C \in \mathsf{Con}_{\mathcal{Q}}(B)$.
Question: is there an interpretation $\mathcal{I}$ which is a model for $\mathcal{O}$ and $C^{\mathcal{I}} \neq \emptyset$?

By abusing the notation we will always write $\mathrm{TSAT}_{\exists\forall}(B)$ instead of $\mathrm{TSAT}_{\{\exists,\forall\}}(B)$ and similarly use this shortcut for all quantifier subsets of $\{\exists, \forall\}$.

Theorem 2.28 ([Pra78, VW86, DM00, FL79, Gia95]).
$\mathrm{OCSAT}_{\exists\forall}(\mathsf{BF})$ *is* EXP*-complete.*

Now it is easy to see, that $\mathrm{OCSAT}_{\mathcal{Q}}(B) \equiv_{\mathrm{cd}} \mathrm{OSAT}_{\mathcal{Q}}(B)$ as a concept C is satisfiable iff the ontology $\{C(a)\}$ is satisfiable for a fresh chosen individual a. This leads to an interreducibility independent from B and $\mathcal{Q}$ in the following way:

$$\mathrm{TSAT}_{\mathcal{Q}}(B) \leq_{\mathrm{cd}} \mathrm{TCSAT}_{\mathcal{Q}}(B) \leq_{\mathrm{cd}} \mathrm{OCSAT}_{\mathcal{Q}}(B) \equiv_{\mathrm{cd}} \mathrm{OSAT}_{\mathcal{Q}}(B).$$

Besides these more general satisfiability problems a counterpart to the propositional implication problem plays an import role in the area of description logics, namely, subsumption. Therefore one says a concept C *is subsumed* by another concept D, i.e., $C \sqsubseteq D$ if and only if for all interpretations $\mathcal{I}$ it holds that $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. Usually we investigate this property with respect to a given terminology $\mathcal{T}$. If $\mathcal{T}$ is a terminology and C, D are concepts, then $C \sqsubseteq_{\mathcal{T}} D$ holds iff for all interpretations $\mathcal{I}$ it holds that $\mathcal{I} \models \mathcal{T}$ implies $C^{\mathcal{I}} \sqsubseteq D^{\mathcal{I}}$.

Problem (Subsumption SUBS$_{\mathcal{Q}}(B)$)
Input: Given $C, D \in \mathsf{N}_{\mathsf{C}\,\mathcal{Q}}(B)$ and $\mathcal{T} \subseteq \mathfrak{T}_{\mathcal{Q}}(B)$.
Question: does $C \sqsubseteq_{\mathcal{T}} D$ hold?

Due to being interreducible to $\mathrm{TCSAT}_{\exists\forall}(\mathsf{BF})$ (see Lemma 4.39 on page 95) the following theorem can be achieved.

Theorem 2.29.
$\mathrm{SUBS}_{\exists\forall}(\mathsf{BF})$ *is* EXP*-complete under* $\leq_{\mathrm{cd}}$.

2.4 Complete Problems

For several results in this thesis other decision problems play an important role in order to state respecting upper or lower complexity bounds. In this paragraph these problems will be defined.

Problem (GAP)
Input: a directed graph $G = (V, E)$, two vertices $s, t \in V$.
Question: is there a path from s to t in G?

Theorem 2.30 ([Sav70]).
GAP *is* NLOGSPACE*-complete under* $\leq^{\log}_{\mathrm{m}}$.

Yet the problem GAP becomes LOGSPACE-complete if we have symmetric edges or undirected edges wherefore it is denoted by UGAP.

Problem (UGAP)
Input: an undirected graph $G = (V, E)$, two vertices $s, t \in V$.
Question: is there a path from s to t in G?

Theorem 2.31 ([LP82, Rei05]).
UGAP *is* LOGSPACE*-complete under* $\leq_{\mathrm{m}}^{\log}$.

The *majority problem* not only plays an important role through the definition of the circuit complexity class TC^0 introduced on page 11, it also provides itself useful as a problem to state reductions shown in Theorem 3.2 on page 30.

Problem (MAJ)
Input: a string $w \in \{0,1\}^*$.
Question: does w contain more than or equal many ones as zeros, i.e., does $|w|_1 \geq |w|_0$ hold?

Theorem 2.32 ([CSV84]).
MAJ *is* TC^0*-complete under* $\leq_{\mathrm{cd}}$.

The very easy question whether a given string of ones and zeros contains an even number of ones characterizes $\mathsf{AC}^0[2]$ by definition of adding MOD_2-gates to AC^0 and therefore is a strict super class of AC^0. The problem is used in the context of promise problems in Theorem 3.3 on page 31.

Problem (PARITY)
Input: a string $w \in \{0,1\}^*$.
Question: does w contain an even number of ones, i.e., does $|w|_1 \equiv 0 \mod 2$ hold?

Theorem 2.33 ([Smo87]).
PARITY *is* $\mathsf{AC}^0[2]$*-complete under* $\leq_{\mathrm{cd}}$.

Quantified Boolean formulae are an extension of propositional logic by two quantifiers $\exists$ and $\forall$ in the following way. If $\varphi \in \mathrm{PL}$ is a propositional formula, then it is also a quantified Boolean formula. Let denote with QBF the set of all quantified Boolean formulae. If $\varphi, \psi \in \mathrm{QBF}$ then $\exists x\varphi, \forall x\psi \in \mathrm{QBF}$, where $\exists x\varphi \equiv \varphi|_{[x/\top]} \vee \varphi|_{[x/\bot]}$ and $\forall x\varphi \equiv \varphi|_{[x/\top]} \wedge \varphi|_{[x/\bot]}$. Let φ be a quantified Boolean formula. Then we say that φ is a *closed* quantified Boolean formula if all variables in $\mathbf{Vars}(\varphi)$ are quantified.

Problem (QBF-VAL)
Input: a closed quantified Boolean formula φ.
Question: does $\varphi \equiv \top$ hold?

Problem (QBF-3VAL)
Input: a closed quantified Boolean formula $\varphi \equiv \exists x_1 \forall x_2 \ldots Q_n x_n F$, where F is in 3CNF, and $Q_n \equiv \exists$ if n is odd and otherwise $Q_n \equiv \forall$.

Question: does $\varphi \equiv \top$ hold?

Theorem 2.34 ([Sto77]).
QBF-VAL *and* QBF-3VAL *are* PSPACE*-complete under* $\leq_{\mathrm{cd}}$.

Chapter 3

Temporal Logic

3.1 Satisfiability in CTL and CTL*

In this section we consider the complexity of the satisfiability problem for arbitrary fragments CTL(T,B) of CTL. Surprisingly, if B cannot express the negation of implication, then the complexity of CTL-SAT(T,B) is independent of T, and it drops down to and in some cases even below NC^1. If B suffices to express the negation of implication, then the complexity of CTL-SAT(T,B) depends only on T which is shown in Section 3.1.1. In Section 3.1.2 we consider the fragments with complexity dependent on T and show completeness of satisfiability for NP, PSPACE, and EXP.

3.1.1 Restricting the Boolean connectives

We separate the fragments CTL(T,B) into two groups: one for which the complexity of satisfiability only depends on B, and one for which it only depends on T.

Theorem 3.1.
Let T denote a set of CTL*-operators and let B be a finite set of Boolean functions such that $[B] \notin \{\mathsf{L}, \mathsf{L}_0\}$. Then* CTL-SAT$(T,B)$ *is*

(1.) equivalent to CTL-SAT(T, BF) *if* $\mathsf{S}_1 \subseteq [B]$,

(2.) in NC^1 *otherwise.*

Proof. For (1.), note that $\mathsf{BF} = [\mathsf{S}_1 \cup \{\top\}] = [B \cup \{\top\}]$ proves this equivalence if we are able to simulate $\top$ in all sets of Boolean functions B satisfying $[B] \supseteq \mathsf{S}_1$. A method allowing for such an expression has been presented in [Lew79] and is often referred to as 'Lewis knack'. In this technique one substitutes any constant $\top$ by a fresh variable t and finally adds to each subformula the conjunct $\wedge t$. From $\mathsf{E}_0 \subseteq \mathsf{S}_1$ we know that $\wedge \in [B]$ is available for $\mathsf{S}_1 \subseteq [B]$. Therefore t is treated similarly to $\top$ in any model. Observe that Lemma 2.4 ensures that we have access to a short representation of $\wedge$ circumventing possible blow-ups.

For (2.), we have to distinguish four cases. We start with $\mathsf{S}_{11} \subseteq [B] \subseteq \mathsf{M}$ (case (2a)). Since $[B]$ does not contain negation, $\varphi \in$ CTL-SAT(T,B) iff the model $M = (\{s\}, \{(s,s)\}, l)$ with $l(s) = \mathsf{PROP}$ satisfies φ (note that CTL models are required to have total transition relations). Evaluating φ under M can be simulated by substituting each atomic proposition

in φ with $\top$, replacing each CTL-operator $O(\cdot)$ with $\mathbf{id}(\cdot)$, and evaluating this proposition-free formula like a propositional formula. As evaluation of propositional S_{11}-formulae is NC^1-complete [Sch10], the claim follows.

The following cases locate the satisfiability problem even in TC^0. First, consider the cases $[B] \subseteq \mathsf{R}_1$ and $[B] \subseteq \mathsf{D}$ (case (2b)). An induction on the formula structure shows that all formulae are trivially satisfiable by the model $M = (\{s\}, \{(s,s)\}, l)$ with either $l(s) = \mathsf{PROP}$ or $l(s) = \emptyset$. For R_1-formulae we use the first labeling, and for D-formulae it depends on the occurring self-dual functions. Second (case (2c)), consider $[B] \subseteq \mathsf{N}$. After moving the negation symbols inside, we can w.l.o.g. assume that

$$\varphi \equiv \mathcal{O}_{11} \cdots \mathcal{O}_{1k_1} \mathcal{P}_1 \Big[\psi \mathrm{U} \mathcal{O}_{21} \cdots \mathcal{O}_{2k_2} \mathcal{P}_2 \big[\cdots \mathrm{U} \mathcal{O}_{\ell 1} \cdots \mathcal{O}_{\ell k_\ell} \mathcal{P}_\ell [\cdots \mathrm{U} \psi'] \cdots \big] \Big],$$

where $\psi \in \mathrm{CTL}(T,B)$, $\psi' \in \mathrm{CTL}(T \setminus \{\mathrm{AU}, \mathrm{EU}\}, B)$, $\mathcal{O}_{ij} \in T \setminus \{\mathrm{AU}, \mathrm{EU}\}$ for $\ell, k \in \mathbb{N}$, $1 \leq j \leq k$, $1 \leq i \leq \ell$ and $\mathcal{P}_1, \ldots, \mathcal{P}_\ell \in \{\mathrm{A}, \mathrm{E}\}$. Observe that ψ' is equal to a literal after counting the preceding negations. Hence we only need to count the number of preceding negations of ψ' modulo 2 in order to construct a 'looping' model which immediately satisfies ψ' (or return false if $\psi' \equiv \bot$). For the remaining clones (case (2d)), either $[B] \subseteq \mathsf{V}$ or $[B] \subseteq \mathsf{E}$. Hence, we can substitute the propositions with $\top$ and only need to guess nondeterministically the position of a $\top$ (case $[B] \subseteq \mathsf{V}$), or ensure absence of $\bot$ (case $[B] \subseteq \mathsf{E}$) leading to $\Sigma_1^{\mathscr{R}}$ and $\Pi_1^{\mathscr{R}}$ which are both subsets of NC^1. □

An analysis of the following proof yields completeness results for NC^1 and below.

Theorem 3.2.
Let T denote a set of CTL*-operators and let B be a finite set of Boolean functions such that $[B] \notin \{\mathsf{L}, \mathsf{L}_0\}$ and $\mathsf{S}_1 \not\subseteq [B]$. Then* CTL-SAT$(T,B)$ *is*

(1.) NC^1-complete under $\leq_{\mathrm{cd}}$-reductions if $\mathsf{S}_{11} \subseteq [B] \subseteq \mathsf{M}$, and

(2.) TC^0-complete under $\leq_{\mathrm{cd}}$-reductions in all other cases.

Proof. The proof of NC^1-completeness, respectively, membership in TC^0 is already contained in case (2a) resp. the cases (2b)–(2d) in the proof of Theorem 3.1.

Now we turn towards (2.). Syntactical correctness of the input depends on the encoding and proper nesting of the parentheses. As any sensible encoding can be verified in AC^0, it remains to check that the number of opening parentheses is greater or equal to the number of closing parentheses for any (decoded) prefix of the input and that the number of opening and closing parentheses matches. This can clearly be done in TC^0. As for the TC^0-hardness, we consider the majority problem $\mathrm{MAJ} \underset{\text{def}}{=} \{w \in \{0,1\}^* \mid |w|_1 \geq |w|_0\}$, which is complete for TC^0 under $\leq_{\mathrm{cd}}$-reductions. Given $w \in \{0,1\}^n$, it holds that $|w|_1 \geq |w|_0$ iff there is a k s.t. $0 \leq k \leq n : |w|_1 = |w0^k|_0$. Hence, $w \in \mathrm{MAJ}$ iff $\bigvee_{0 \leq k \leq n} |1^n w 0^{n+k}|_1 = |1^n w 0^{n+k}|_0$ is satisfiable. Moreover, if $w \in \mathrm{MAJ}$ it holds that $|u|_1 \geq |u|_0$ for every prefix u of $1^n w 0^{n+k}$, where $k = |w|_1 - |w|_0$. For ℓ satisfying $|1^n w 0^{n+\ell}|_1 = |1^n w 0^{n+\ell}|_0$, $1^n w 0^{n+\ell}$ can thus

be interpreted as a balanced string of parentheses. Let $p \in \mathsf{PROP}$, and $\otimes$ be a binary projection function $x_1 \otimes x_2 \underset{\text{def}}{=} x_1$. Regardless how B is defined we always have access to such a function. Now it is not hard to construct a homomorphism h mapping $\{0,1\}^*$ to $\{(,),p,\otimes\}^*$ such that $1^n w 0^{n+\ell} \in \mathrm{MAJ}$ iff $h(1^n w 0^{n+\ell})$ is a syntactically correct $\mathrm{CTL}(T,B)$ formula. Therefore define

$$h(w_i) \underset{\text{def}}{=} \begin{cases} (, & \text{if } w_i = w_{i+1} = 1 \text{ or } w_i = 1, i = n \\ (p, & \text{if } 1 = w_i \neq w_{i+1} = 0 \\ ), & \text{if } w_i = w_{i+1} = 0 \text{ or } w_i = 0, i = n \\ )\otimes, & \text{if } 0 = w_i \neq w_{i+1} = 1, \end{cases}$$

for $w = w_1 w_2 \dots w_n$ and $1 \leq i \leq n$. Then $w \in \mathrm{MAJ}$ iff $\bigvee_{0 \leq k \leq n} h(1^n w 0^{n+k}) \in \mathrm{CTL}(T,B)$. From this it is clear how to construct an AC^0 circuit with oracle gates for $\mathrm{CTL}(T,B)$ to decide MAJ, and hence $\mathrm{MAJ} \leq_{\mathrm{cd}} \mathrm{CTL}(T,B)$ follows. □

Thus, the hard part for checking satisfiability for these specific type of formulae is strictly connected to the syntactical correctness of the input. In order to classify the complexity of $\mathrm{CTL\text{-}SAT}(T,B)$ beyond this point, we restrict our attention to, now *promised*, syntactically correct formulae: Let $\mathrm{CTL\text{-}SAT}_{\mathscr{P}}(T,B)$ denote the *promise problem* of deciding whether a given syntactically correct $\mathrm{CTL}(T,B)$-formula is satisfiable. This can be used to refine Theorem 3.2 for subclasses of TC^0.

Theorem 3.3.
Let T denote a set of CTL*-operators and let $B \notin \{\mathsf{L}, \mathsf{L}_0\}$ and $\mathsf{S}_1 \not\subseteq [B]$ be a finite set of Boolean functions such that* $\mathrm{CTL\text{-}SAT}(T,B)$ *is TC^0-complete.*

Then $\mathrm{CTL\text{-}SAT}_{\mathscr{P}}(T,B)$ *is*

(1.) in TC^0 if $T \cap \{\mathrm{AU}, \mathrm{EU}\} \neq \emptyset$ and $[B] \in \{\mathsf{V}, \mathsf{V}_0, \mathsf{E}, \mathsf{E}_0, \mathsf{N}\}$,

(2.) $\Sigma_1^{\mathscr{R}}$-complete if $T \cap \{\mathrm{AU}, \mathrm{EU}\} = \emptyset$ and $[B] \in \{\mathsf{V}, \mathsf{V}_0\}$,

(3.) $\Pi_1^{\mathscr{R}}$-complete if $T \cap \{\mathrm{AU}, \mathrm{EU}\} = \emptyset$ and $[B] \in \{\mathsf{E}, \mathsf{E}_0\}$,

(4.) $\mathsf{AC}^0[2]$-complete if $T \cap \{\mathrm{AU}, \mathrm{EU}\} = \emptyset$ and $[B] = \mathsf{N}$, and

(5.) trivial in all other cases,

with respect to $\leq^{\mathrm{dlt}}_{\mathrm{proj}}$-reductions.

Proof. For (1.), one has to determine the relevant parts of the formula first. This requires counting the parentheses, therefore the problem remains in TC^0.

The cases (2.) and (3.) can be solved analogously to [Sch10, Lemma 9], that is, by guessing the position of a satisfying $\top$ (or a falsifying $\bot$, resp.) after substituting all propositions with $\top$. Hardness is obtained via a reduction from the language $\{0,1\}^* 1 \{0,1\}^*$ (or $\{0\}^*$, resp.).

For (4.), syntactically correct formulae in $\mathrm{CTL}(T,\mathsf{N})$ can be checked for satisfiability by just counting the preceding negations modulo 2. Hardness for this case arises from a reduction from $\mathrm{PARITY} = \{w \in \{0,1\}^* \mid |w|_1 \equiv 1 \mod 2\}$.

Lastly, in any other case, all $\mathrm{CTL}(T,B)$-formulae are trivially satisfiable. □

3.1.2 Restricting the CTL-operators

We continue to determine the complexity of CTL-SAT(T, BF) for $\mathsf{S}_1 \subseteq [B]$ and an arbitrary set T of CTL-operators (case (1.) in Theorem 3.1). It turns out that each fragment's satisfiability problem is complete for NP, PSPACE, or EXP.

Theorem 3.4.
Let T be a set of CTL*-operators. Then* CTL-SAT(T, BF) *is*

(1.) NP*-complete under $\leq_{\text{cd}}$-reductions if $T \subseteq \{\text{AF}\}$,*

(2.) PSPACE*-complete under $\leq_{\text{cd}}$-reductions if $\{\text{AG}\} \subseteq T \subseteq \{\text{AG}, \text{AF}\}$ or $\{\text{AX}\} \subseteq T \subseteq \{\text{AX}, \text{AF}\}$, and*

(3.) EXP*-complete under $\leq_{\text{cd}}$-reductions in all other cases.*

Proof. We will prove (1.) to (3.) in the following three lemmata. □

Lemma 3.5.
Let $T \subseteq \{\text{AF}\}$ be a set of CTL*-operators. Then* CTL-SAT(T, BF) *is* NP*-complete under $\leq_{\text{cd}}$-reductions.*

Proof. The NP-hardness of CTL-SAT($\emptyset$, BF) is immediate from the NP-hardness of SAT under $\leq_{\text{cd}}$-reductions [Coo71b]. Considering the upper bound, i.e., the membership of CTL-SAT($\{\text{AF}\}$, BF) in NP follows from a small model property:

Claim. *$\varphi \in \text{CTL}(\{\text{AF}\}, \text{BF})$ is satisfiable iff φ is satisfiable by a polynomial-sized model.*

Proof of Claim. The following proof is a modification that strongly builds on the insights in a proof in [Eme90, Theorem 6.14, Small Model Theorem for CTL, pp. 1034]. There, Emerson shows that an arbitrary satisfiable CTL formula can be satisfied in a canonical model of at most *exponential* size in the length of the formula. The construction consists of unfurling a usual model into an infinite computation tree model which can be transformed into a pseudo-Hintikka structure (a structure which contains subformulae as labels satisfying the eventualities imposed by formulae containing F's or U's) and uses dag-like[1] structures. This pseudo-Hintikka structure can on his part be transformed into a tableau-like matrix structure containing for each Hintikka-set a dag-structure on the horizontal level and vertically for each eventuality one dag. As there are exponential many Hintikka-sets for given φ and linear many eventualities in φ this leads finally to a exponential sized model for φ.

For our case where only AF and EG operators are allowed we will now describe how to shrink the exponential sized model to the desired polynomial size. For every path in the structure depicted in Figure 3.1 any AF-preceded subformula of φ is already satisfied. Therefore deleting dags in the structure retroactively until the last branching does not change the satisfiability of these formulae. Now observe that for any of the possible linear many different EG-preceded subformulae EGψ of φ exactly m different dags in

[1] 'dag' is the abbreviation of *directed acyclic graph*.

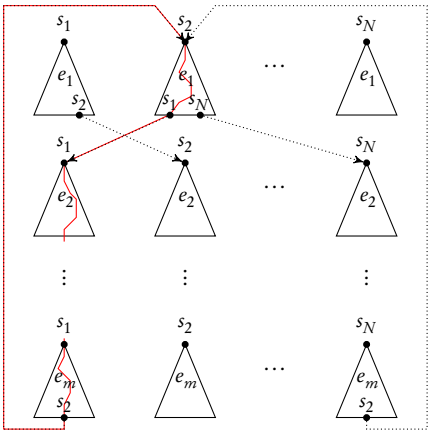


Figure 3.1: Constructed structure from [Eme90, Thm. 6.14, p.1036]. Dotted arrows indicate node replacement. The red line denotes the path of an EG-prefixed formula.

the infinite path through the structure are traversed, as the matrix-like structure has size $N \times m$, where m is the number of eventualities in φ hence bounded by $|\mathbf{SF}(\varphi)|$. Thus at most $O(|\varphi| \cdot m)$ different dags are needed in order to satisfy φ and the remaining dags can be deleted. In our case the size each dag is in fact polynomial in $|\varphi|$ (and not exponential as in the original proof) because the q-rank (which denotes the length of the fulfilling path) of each subformulae is 0 and only EF- or EU-formulae have a q-rank > 0. Altogether that means we have a model of size $O(|\varphi|^2 \cdot m)$ which is clearly polynomial. ⊣

Thus it holds that $\varphi \in \mathrm{CTL}(\{\mathrm{AF}\}, \mathsf{BF})$ is satisfiable iff φ is satisfiable in a model of size $\leq |\varphi|^{O(1)}$. As the model checking problem CTL-MC is in P (see Theorem 2.17), guessing such a model M and checking whether $M \models \varphi$ can be performed nondeterministically in polynomial time. □

For the second claim of Theorem 3.4 we will show how to express quantified Boolean formulae and how to modify Ladner's algorithm for deciding satisfiability of modal formulae.

Lemma 3.6.
Let T be a set of CTL*-operators s.t.* $\{\mathrm{AG}\} \subseteq T \subseteq \{\mathrm{AG}, \mathrm{AF}\}$ *or* $\{\mathrm{AX}\} \subseteq T \subseteq \{\mathrm{AX}, \mathrm{AF}\}$. *Then* $\mathrm{CTL\text{-}SAT}(T, \mathsf{BF})$ *is* PSPACE*-complete under* $\leq_{\mathrm{cd}}$*-reductions.*

Proof. Now it suffices to show PSPACE-hardness for $T = \{\mathrm{AG}\}, \{\mathrm{AX}\}$, and membership in PSPACE for $T = \{\mathrm{AF}, \mathrm{AG}\}, \{\mathrm{AX}, \mathrm{AF}\}$. The hardness result for $T = \{\mathrm{AX}\}$ is shown by Ladner in [Lad77, Theorem 3.1] as $\mathrm{CTL}(\{\mathrm{AX}\}, \mathsf{BF})$ is nothing but plain modal logic.

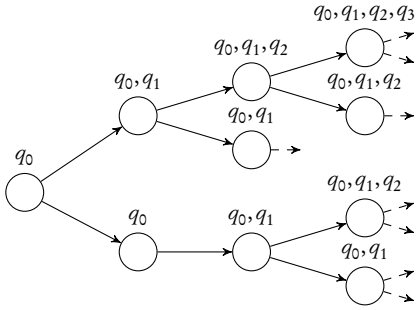


Figure 3.2: Part of a nested tree-like structure used in the proof of Theorem 3.4.

Now we show the PSPACE-hardness for $T = \{\mathrm{AG}\}$ by giving straightforward modification of the Ladner-reduction from the validity problem for quantified Boolean formulae QBF-VAL. Let λ be a quantified Boolean formula that is w.l.o.g. of the form $\lambda = \exists x_1 \forall x_2 \cdots \mathfrak{Q} x_n (C_1 \wedge \cdots \wedge C_m)$, where the C_is are disjunctions of literals, and $\mathfrak{Q} = \exists$ if n is odd, and $\mathfrak{Q} = \forall$ otherwise. We define the reduction f by $\lambda \mapsto \varphi_{\text{tree}} \wedge \varphi_{\text{cla}} \wedge \varphi_\lambda$, where φ_{tree}, φ_{cla} and φ_λ are defined as follows: φ_{tree} enforces the existence of properly nested *tree-like*[2] structures $(T_s)_{s\in\{0,1\}^*}$ such that for all $s \in \{0,1\}^*$, $s = s_1 \cdots s_n$,

(a) the root of T_s satisfies $\neg q_{n+1} \wedge \bigwedge_{1\le i\le n} \mathrm{AG} q_i$,

(b) T_{s0} and T_{s1} are disjoint, properly nested tree-like structures of T_s achieved by the formula

$$\left(\neg q_{n+1} \wedge \bigwedge_{1\le i\le n} q_i\right) \rightarrow \left(\mathrm{EFAG}(q_{n+1} \wedge x_{n+1}) \wedge \mathrm{EFAG}(q_{n+1} \wedge \neg x_{n+1})\right), \text{ and}$$

(c) in T_s, $s_i = 1$ ($s_i = 0$) implies that proposition x_i (resp. $\overline{x_i}$) holds globally.

An example for a nested tree-like structure is depicted in Figure 3.2. Next, φ_{cla} enforces the labelling of the clauses of λ into $(T_s)_{s\in\{0,1\}^*}$ by requiring that globally $x_i \rightarrow C_j$ if $x_i \in C_j$ and $\overline{x}_i \rightarrow C_j$ if $\neg x_i \in C_j$, for all T_s with $|s| \ge i$. Also the existence of a label of at least one literal per clause in a state is required by a labeled clause proposition C_j. Finally, we define φ_λ to ensure the existence of a tree-like substructure of T_ε, whose leaves fulfill all clauses and that is build according to the quantification in λ. That is, $\varphi_\lambda = \mathrm{EF}(q_1 \wedge \mathrm{AG}(q_2 \wedge \mathrm{EF} \cdots \exists (q_n \wedge (C_1 \wedge \cdots \wedge C_m)) \cdots))$, where $\exists = \mathrm{EF}$ if n is odd, and $\exists = \mathrm{AG}$ otherwise. It is easy to verify that f is polynomial-time computable and that λ is satisfiable iff $f(\lambda) = \varphi_{\text{tree}} \wedge \varphi_{\text{cla}} \wedge \varphi_\lambda$ is satisfiable.

[2] We say a model $M = (S, R, l)$ is *tree-like* iff for the graph (S, R) for each $s \in S$ there is at most one predecessor.

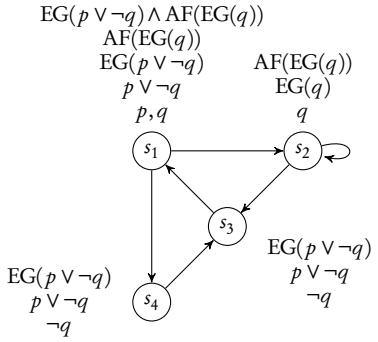


Figure 3.3: Quasi-model for $\varphi = \mathrm{EG}(p \vee \neg q) \wedge \mathrm{AF}(\mathrm{EG}(q))$

Now consider $T = \{\mathrm{AF}, \mathrm{AG}\}$. To show membership in PSPACE, we present a method inspired by the algorithm showing that provability in the modal logic K is in PSPACE [Lad77]. The algorithm is based on the notion of *quasi models*:

Let $\varphi \in \mathrm{CTL}(\{\mathrm{AF}, \mathrm{AG}, \mathrm{EF}, \mathrm{EG}\}, \mathsf{BF})$ be in negation normal form, a *quasi model* for φ is defined as a model $M = (S, R, l)$ with labels $l : S \to \mathfrak{P}(\mathrm{CTL}(\{\mathrm{AF}, \mathrm{AG}\}, \mathsf{BF}))$ such that

- for all $s \in S$, $l(s)$ is a *minimal* set satisfying
  (a) $\psi \wedge \chi \in l(s)$ implies $\psi \in l(s)$ and $\chi \in l(s)$, and
  (b) $\psi \vee \chi \in l(s)$ implies $\psi \in l(s)$ or $\chi \in l(s)$,
- $\varphi \in l(s)$ for some $s \in S$,
- for all $s \in S$, $\mathcal{O} \in \{\mathrm{AF}, \mathrm{EF}, \mathrm{AG}, \mathrm{EG}\}$ and each $\mathcal{O}\psi \in l(s)$, M satisfies the constraints imposed by $\mathcal{O}\psi$ starting in s, i.e., ψ is in $l(x_i)$ for all/some paths $x = (x_1, x_2, x_3, \ldots)$, $x_1 = s$, and all/some $1 \leq i \in \mathbb{N}$, and
- there is no $s \in S$, $\varphi \in \mathrm{CTL}$ s.t. $\varphi, \neg\varphi$ are labeled in s.

Note that the labels of quasi models bear resemblance to Hintikka sets (cf. [BdV01, Definition 6.24]) but differ in that they are allowed to contain $\bot$. We say a quasi model is *consistent* iff none of the properties from above are violated. It clearly holds that φ is satisfiable iff there exists a consistent quasi model for φ. To reduce the search space, we introduce the notion of minimality for quasi models. Say that a quasi model $M = (S, R, l)$ for some formula φ is *minimal* if no states or transitions can be deleted such that the resulting structure is still a quasi model for φ. We point out that the minimal quasi models for φ may differ in the number of states and transitions. In Figure 3.3 a quasi-model for $\varphi = \mathrm{EG}(p \vee \neg q) \wedge \mathrm{AF}(\mathrm{EG}(\neg q))$ is depicted. Observe that this model is not minimal as one can delete the edge (s_2, s_3) and still have a quasi-model for φ.

The algorithm is based on the following observation:

Claim. *$\varphi \in \mathrm{CTL}(\{\mathrm{AF}, \mathrm{AG}\}, \mathsf{BF})$ is unsatisfiable iff, for all minimal quasi models for φ, there is a path holding an inconsistent labeling in a state on a prefix of linear length.*

Proof of Claim. Using contraposition, the direction from right to left is obvious: it suffices to show that there exists a minimal quasi model for φ with consistent labels if φ is satisfiable. This follows from supplementing the labels of the models satisfying φ with the locally satisfied subformulae.

For the direction from left to right, let $\varphi \in \mathrm{CTL}(\{\mathrm{AF}, \mathrm{AG}\}, \mathsf{BF})$ be an unsatisfiable formula, w.l.o.g. in negation normal form. Observe that φ may contain EG- as well as EF-operators now. Define $\#_{\mathrm{T}}(\psi)$ as the number of pure temporal operators in the formula ψ. We prove, by induction on $\#_{\mathrm{T}}(\varphi)$, that for all minimal quasi models $M = (S, R, l)$ there is a path $x = (x_1, x_2, \ldots)$ with $\varphi \in l(x_1)$ such that $l(x_k)$ is inconsistent for some $1 \leq k \leq \#_{\mathrm{T}}(\varphi) + 1$:

For a formula ψ, a quasi model $M = (S, R, l)$ for ψ and a path $x \in S^\omega$, let $L(x) = \bigcup_{i>0} l(x_i)$ be the set of all quasi labels on x, and let $c_\psi(M, x) = |L(x)|$ be the number of distinct quasi labels on x and let $c_\psi = \max\{c_\psi(M, x) \mid x \in S^\omega$ and M is a minimal quasi model for $\psi\}$. c_ψ is the maximum number of distinct quasi labels on all paths over all minimal quasi models for ψ. We will thus show that $c_\varphi \leq \#_{\mathrm{T}}(\varphi) + 1$. This implies that among the inconsistent quasi-models there is one containing a contradiction on a path prefix of linear length.

If $\#_{\mathrm{T}}(\psi) = 0$ then ψ is unsatisfiable iff it is unsatisfiable in all models with a single state already. Hence, $\psi \in l(s)$ implies the inconsistency of $l(s)$ for every quasi model $M = (S, R, l)$.

For the inductive step, let $\psi, \chi \in \mathrm{CTL}(\{\mathrm{AF}, \mathrm{AG}\}, \mathsf{BF})$ and assume that $c_\psi \leq \#_{\mathrm{T}}(\psi) + 1, c_\chi \leq \#_{\mathrm{T}}(\chi) + 1$. Further say that for models $M = (S, R, l)$ and $M' = (S', R', l')$, M' *embeds* M if there exists a function $h\colon S \to S'$ such that $l(s) \subseteq l'(h(s))$ and $(s_1, s_2) \in R$ implies $(h(s_1), h(s_2)) \in (R')^*$, where $(R')^*$ denotes the reflexive and transitive closure of R'. The following cases have to be distinguished:

$\psi \vee \chi$: Each minimal quasi model M_q for $\psi \vee \chi$ is the extension of a minimal quasi model for either ψ or χ with $\psi \vee \chi$ added to the labels of any state containing ψ (or χ, respectively). Hence, $c_{\psi \vee \chi} = \min\{c_\psi, c_\chi\} = \min\{\#_{\mathrm{T}}(\psi) + 1, \#_{\mathrm{T}}(\chi) + 1\} \leq \#_{\mathrm{T}}(\psi \vee \chi) + 1$.

$\psi \wedge \chi$: Let $M = (S, R, l)$ be a minimal quasi model for $\psi \wedge \chi$, let $s_0 \in S$ be such that $\psi \wedge \chi \in l(s_0)$. Therefore $\psi, \chi \in l(s_0)$ and thus there are minimal quasi models M_ψ, M_χ such that M_ψ and M_χ are embedded into M (any non-minimal quasi model for ψ or χ itself embeds a minimal model by definition). As M_ψ and M_χ are minimal quasi models, any path x in either of the models satisfies $c_\psi(M_\psi, x) \leq c_\psi$ (resp. $c_\chi(M_\chi, x) \leq c_\chi$). Hence $c_{\psi \wedge \chi} \leq c_\psi + c_\chi - 1$, since any path $x = (x_1, x_2, \ldots)$ longer than $\#_{\mathrm{T}}(\psi) + \#_{\mathrm{T}}(\chi) + 1$ would include a quasi-label from $l(x_k)$ (for some $k \in \mathbb{N}$) such that neither M_ψ nor M_χ include $l(x_k)$; a contradiction to the minimality of M. Hence, $c_{\psi \wedge \chi} \leq c_\psi + c_\chi + 1 \leq \#_{\mathrm{T}}(\psi \wedge \chi) + 1$.

AGψ, EGψ: Let $M = (S, R, l)$ be a minimal quasi model for AGψ (resp. EGψ). There further exists a minimal quasi model $M_\psi = (S_\psi, R_\psi, l_\psi)$ embedded into M via h. Let Γ denote the smallest quasi label containing AGψ (resp. EGψ). For a path $x = (x_1, x_2, \ldots) \in S_\psi^\omega$, let $h(x) \underset{\text{def}}{=} (h(x_1), h(x_2), \ldots) \in S^\omega$. Then, for any path $x = (x_1, x_2, \ldots) \in S_\psi^\omega$, the set of quasi labels $L(h(x))$ in M is a superset of $\bigcup_{i>0} l_\psi(x_i) \cup \Gamma$. If $L(h(x))$ is a strict superset of $\bigcup_{i>0} l_\psi(x_i) \cup \Gamma$, then there is a $k \in \mathbb{N}$ such that $l(h(x_k)) = l_\psi(x_k) \cup \Gamma \cup \Delta$ for some $\Delta \neq \emptyset$. Still M_ψ is a quasi model for ψ and thus, for all subformulae χ of ψ, $\chi \in l(h(x_k))$ implies $\chi \in l_\psi(x_k)$. Hence, all labels $l_\psi(x_k) \cup \Gamma \cup \Delta$ can be replaced by $l_\psi(x_k) \cup \Gamma$ and we obtain that $L(h(x)) = \bigcup_{i>0} l_\psi(h(x_i)) \cup \Gamma$. Therefore, $c_{\text{AG}\psi}(M, h(x)) = |\{l_\psi(x) \cup \Gamma \mid x \in S_\psi^\omega\}| = c_\psi(M_\psi, x) + 1 \leq c_\psi$.

Due to the minimality of M, each path $x \in S^\omega$ starting in a state s with AG$\psi \in l(s)$ (resp. EG$\psi \in l(s)$) is the image of a path in an embedded minimal quasi-model. Hence, $c_{\text{AG}\psi} = \max\{c_{\text{AG}\psi}(M, x) \mid x \in S^\omega$ and M is a minimal quasi model for AG$\psi\} = \max\{c_\psi(M_\psi, x) \mid x \in S_\psi^\omega$ and M_ψ is a minimal quasi model for $\psi\} \leq c_\psi \leq \#_{\text{T}}(\text{AG}\psi) + 1$ (resp. $c_{\text{EG}\psi} \leq c_\psi$).

AFψ, EFψ: Again, each minimal quasi model M for AFψ (resp. EFψ) is the extension of a minimal quasi model for ψ, with AFψ (resp. EFψ) added to the quasi label of all states containing ψ. Hence, $c_{\text{AF}\psi} = c_\psi = \#_{\text{T}}(\psi) + 1 \leq \#_{\text{T}}(\text{AF}\psi) + 1$ (resp. $c_{\text{EF}\psi} \leq \#_{\text{T}}(\text{EF}\psi) + 1$).

Hence, every minimal quasi model for φ includes an inconsistent quasi label on a path of length $\leq \#_{\text{T}}(\varphi) + 1$. ⊣

The method given in Algorithms 3.1 and 3.2 performs a nondeterministic depth-first search for contradictions on the set of minimal quasi models for φ. The first parameter d stems from the proof of the claim from above (temporal depth of the input formula plus one) and is therefore linear in $|\varphi|$. The remaining six parameters are all subsets of formulae in $\mathbf{SF}(\varphi)$, that should be either globally true on all paths ($\mathscr{T}_{\text{AG}}$), eventually false on a path ($\mathscr{F}_{\text{AG}}$), eventually true on all paths ($\mathscr{T}_{\text{AF}}$), globally false on a path ($\mathscr{F}_{\text{AF}}$), true or false in the current state ($\mathscr{T}$ resp. $\mathscr{F}$). The space bound derives from the linear length of path prefixes to be investigated.

For $T = \{\text{AX}, \text{AF}\}$, we will present a proof sketch. A straightforward modification of the former algorithm is not possible, since the X operator allows for the construction of "counters" such that contradictions may firstly occur in exponential depth. This fact is shown in Example 2.24 on page 22. Yet, CTL($\{\text{AX}, \text{AF}\}, \mathsf{BF}$)-formulae may impose at most linearly many temporal constraints. Using the fixpoint-characterisation EG$\varphi \equiv \varphi \wedge \text{EXEG}\varphi$, we derive an algorithm for formulae $\varphi \in \text{CTL}(\{\text{AX}, \text{AF}\}, \mathsf{BF})$ in a two-step approach: first verify that φ with all EG operators ignored is satisfiable, then test each of the EG-prefixed subformulae for satisfiability separately. The first step is completely analogous to the above; the second step follows from the fact that an EG-prefixed subformula is satisfiable iff it is satisfied on an ultimately periodic path which is a path containing an infinitely many often occurring suffix. Therefore we need to

Algorithm 3.1: The function determining satisfiability of a CTL($\{\mathsf{AF},\mathsf{AG}\},\mathsf{BF}$)-formula.

satisfiable : $d,\mathscr{T},\mathscr{F},\mathscr{T}_{\mathsf{AG}},\mathscr{F}_{\mathsf{AG}},\mathscr{T}_{\mathsf{AF}},\mathscr{F}_{\mathsf{AF}}$

1 **if** $\mathscr{T}\cup\mathscr{F}\nsubseteq \mathsf{PROP}\cup\{\bot,\top\}$ **then**
2 decomposeBF($d,\mathscr{T},\mathscr{F},\mathscr{T}_{\mathsf{AG}},\mathscr{F}_{\mathsf{AG}},\mathscr{T}_{\mathsf{AF}},\mathscr{F}_{\mathsf{AF}}$);
3 **else** /* $\mathscr{T}\cup\mathscr{F}\subseteq\mathsf{PROP}\cup\{\bot,\top\}$ */
4 **if** $\mathscr{T}\cap\mathscr{F}\neq\emptyset$ **or** $\top\in\mathscr{F}$ **or** $\bot\in\mathscr{T}$ **then return** false;
5 **else if** $d=0$ **then return** true;
6 **else**
7 $t\leftarrow$ true;
8 **foreach** $\alpha\in\mathscr{F}_{\mathsf{AG}}$ **do** /* check whether α can eventually be falsified (on some path) */
9 nondeterministically guess $\emptyset\subsetneq\beta\subseteq\mathscr{T}_{\mathsf{AF}}\cup\{\alpha\}$;
10 $t\leftarrow t\wedge$ satisfiable$(d-1,\beta\cup\mathscr{T}_{\mathsf{AG}},\emptyset,\mathscr{T}_{\mathsf{AG}},\{\alpha\}\setminus\beta,\mathscr{T}_{\mathsf{AF}}\setminus\beta,\emptyset)$;
11 **foreach** $\alpha\in\mathscr{F}_{\mathsf{AF}}$ **do** /* check whether α is invariantly falsified (on some path) */
12 nondeterministically guess $\emptyset\subsetneq\beta\subseteq\mathscr{T}_{\mathsf{AF}}$;
13 $t\leftarrow t\wedge$ satisfiable$(d-1,\{\alpha\}\cup\beta\cup\mathscr{T}_{\mathsf{AG}},\emptyset,\mathscr{T}_{\mathsf{AG}},\emptyset,\mathscr{T}_{\mathsf{AF}}\setminus\beta,\{\alpha\})$;
14 **return** t;

check whether for a subformula $\mathrm{EG}\psi\in\mathbf{SF}(\varphi)$ the respecting eventualities imposed by AX- and AF-subformulae can be satisfied concurrently. Mapping each $\mathrm{EG}\psi$ to a separate infinite path whose prefix is loop-free and afterwards satisfying ψ for consistent quasi labels enables us to iteratively verify the consistency of the given φ without needing several recursive calls leading to exponential space. Despite the possibility of exponentially large paths using dynamic programming to construct the respecting sets in the tableau-like algorithm merges into an algorithm using polynomial space. □

Finally we will construct a generic reduction from a special kind of Turing machines in order to state the desired lower bound for the class EXP.

Lemma 3.7.
Let T be a set of CTL*-operators s.t.* $\{\mathrm{AX},\mathrm{AG}\}\subseteq T$, $\{\mathrm{EU}\}\subseteq T$, *or* $\{\mathrm{AU}\}\subseteq T$. *Then* CTL-SAT$(T,\mathsf{BF})$ *is* EXP*-complete under* $\leq_{\mathsf{cd}}$*-reductions.*

Proof. The membership of CTL-SAT($\{\mathrm{AX},\mathrm{AU},\mathrm{EU}\},\mathsf{BF}$) in EXP is due to Theorem 2.16. Hardness for EXP is obtained from reducing the word problem for polynomial-space alternating[3] Turing machines to CTL-SAT(T,BF) for $T=\{\mathrm{AX},\mathrm{AG}\}$, $T=\{\mathrm{EU}\}$, and $T=\{\mathrm{AU}\}$. The hardness of the remaining fragments follows (cf. Figure 2.3).

First consider the case $T=\{\mathrm{AX},\mathrm{AG}\}$. Let Σ be some fix alphabet, $w=w_0\cdots w_{n-1}\in\Sigma^\star$ be the input of length $|w|=n$ and let $M=(Q,\Sigma,\Gamma,\delta,q_0,\Box)$ be an alternating Turing machine (ATM) working in space n^k, $k\in\mathbb{N}$. Further denote by $Q=Q_\exists\uplus Q_\forall\uplus Q_{\mathrm{acc}}\uplus Q_{\mathrm{rej}}$ the sets of existential, universal, accepting and rejecting states of M. We assume w.l.o.g.

[3] Observe that alternating polynomial-space is equal to deterministic exponential time by [CKS81].

Algorithm 3.2: The decomposition subfunction determining satisfiability of a CTL({AF, AG}, BF)-formula.

decomposeBF : $d, \mathcal{T}, \mathcal{F}, \mathcal{T}_{AG}, \mathcal{F}_{AG}, \mathcal{T}_{AF}, \mathcal{F}_{AF}$
randomly choose an $\alpha \in (\mathcal{T} \cup \mathcal{F}) \setminus \mathsf{PROP}$;
if $\alpha = \neg\beta$ **and** $\alpha \in \mathcal{T}$ **then**
  return satisfiable$(d, \mathcal{T} \setminus \{\alpha\}, \mathcal{F} \cup \{\beta\}, \mathcal{T}_{AG}, \mathcal{F}_{AG}, \mathcal{T}_{AF}, \mathcal{F}_{AF})$;
else if $\alpha = \neg\beta$ **and** $\alpha \in \mathcal{F}$ **then**
  return satisfiable$(d, \mathcal{T} \cup \{\beta\}, \mathcal{F} \setminus \{\alpha\}, \mathcal{T}_{AG}, \mathcal{F}_{AG}, \mathcal{T}_{AF}, \mathcal{F}_{AF})$;
else if $\alpha = \beta \wedge \gamma$ **and** $\alpha \in \mathcal{T}$ **then**
  return satisfiable$(d, (\mathcal{T} \cup \{\beta, \gamma\}) \setminus \{\alpha\}, \mathcal{F}, \mathcal{T}_{AG}, \mathcal{F}_{AG}, \mathcal{T}_{AF}, \mathcal{F}_{AF})$;
else if $\alpha = \beta \wedge \gamma$ **and** $\alpha \in \mathcal{F}$ **then**
  nondeterministically guess $\delta \in \{\beta, \gamma\}$;
  return satisfiable$(d, \mathcal{T}, (\mathcal{F} \cup \{\delta\}) \setminus \{\alpha\}, \mathcal{T}_{AG}, \mathcal{F}_{AG}, \mathcal{T}_{AF}, \mathcal{F}_{AF})$;
else if $\alpha = \mathrm{AG}\beta$ **and** $\alpha \in \mathcal{T}$ **then**
  return satisfiable$(d, (\mathcal{T} \cup \{\beta\}) \setminus \{\alpha\}, \mathcal{F}, \mathcal{T}_{AG} \cup \{\beta\}, \mathcal{F}_{AG}, \mathcal{T}_{AF}, \mathcal{F}_{AF})$;
else if $\alpha = \mathrm{AG}\beta$ **and** $\alpha \in \mathcal{F}$ **then**
  $b \leftarrow$ nondeterministically guess whether β is false in this state;
  if b **then return** satisfiable$(d, \mathcal{T}, (\mathcal{F} \cup \{\beta\}) \setminus \{\alpha\}, \mathcal{T}_{AG}, \mathcal{F}_{AG}, \mathcal{T}_{AF}, \mathcal{F}_{AF})$;
  else return satisfiable$(d, \mathcal{T}, \mathcal{F} \setminus \{\alpha\}, \mathcal{T}_{AG}, \mathcal{F}_{AG} \cup \{\beta\}, \mathcal{T}_{AF}, \mathcal{F}_{AF})$;
else if $\alpha = \mathrm{AF}\beta$ **and** $\alpha \in \mathcal{T}$ **then**
  $b \leftarrow$ nondeterministically guess whether β is true in this state;
  if b **then return** satisfiable$(d, (\mathcal{T} \cup \{\beta\}) \setminus \{\alpha\}, \mathcal{F}, \mathcal{T}_{AG}, \mathcal{F}_{AG}, \mathcal{T}_{AF}, \mathcal{F}_{AF})$;
  else return satisfiable$(d, \mathcal{T} \setminus \{\alpha\}, \mathcal{F}, \mathcal{T}_{AG}, \mathcal{F}_{AG}, \mathcal{T}_{AF} \cup \{\beta\}, \mathcal{F}_{AF})$;
else if $\alpha = \mathrm{AF}\beta$ **and** $\alpha \in \mathcal{F}$ **then**
  return satisfiable$(d, \mathcal{T}, (\mathcal{F} \cup \{\beta\}) \setminus \{\alpha\}, \mathcal{T}_{AG}, \mathcal{F}_{AG}, \mathcal{T}_{AF}, \mathcal{F}_{AF} \cup \{\beta\})$;

that all ATMs halt after a finite number of steps and that $\delta(q,c) \neq \emptyset$ iff $q \in Q_\forall \cup Q_\exists$, for all $c \in \Gamma$. As M is space-bounded by n^k, M may only visit $2n^k+1$ different tape cells. We can hence describe each configuration of M as a string of length $2n^k+1$ that is constructed by padding with $\square$ to the left and to the right, such that all reachable tape cells are included. Such a padded configuration will be called *situation*. We define the reduction function f to map $M, w \mapsto \varphi \in \mathrm{CTL}(\{\mathrm{AX}, \mathrm{AG}\}, \mathsf{BF})$ such that φ forces a satisfying model to encode a proof tree of situations of an accepting computation of M on input w.

We determine the set of atomic propositions PROP for φ and then construct the formula φ as $\varphi \underset{\text{def}}{=} \varphi_{\text{init}} \wedge \bigwedge_{i=1}^{3} \mathrm{AG}\varphi_i \wedge \mathrm{AG}\varphi_\delta$ to be defined below. Let $\mathsf{PROP} = \mathsf{PROP}_S \cup \mathsf{PROP}_T \cup \mathsf{PROP}_P$ be a set of atomic propositions, where $\mathsf{PROP}_S = \{s_q \mid q \in Q\}$ are called *state propositions*, $\mathsf{PROP}_T = \{t_{i,a} \mid -n^k \leq i \leq n^k, a \in \Sigma\}$ are called *tape propositions*, and $\mathsf{PROP}_P = \{p_i \mid -n^k \leq i \leq n^k\}$ are called *position propositions*.

In the following the formulae φ_1 and φ_2 ensure that in each state of a model for φ exactly one state proposition s_q, exactly one position proposition p_i and exactly one tape proposition $t_{i,a}$ per $i \in [-n^k, n^k]$ holds. From now on we use I to denote the intervall $[-n^k, n^k]$. The formula φ_3 states that, for $i \in I$, the tape propositions (i.e., the contents of the tape) must not change between connected states in M unless the position proposition p_i denotes it. These formulae are defined as

$$\varphi_1 \underset{\text{def}}{=} \bigwedge_{q \in Q} \Big(s_q \rightarrow \bigwedge_{\substack{q' \in Q, \\ q' \neq q}} \neg s_{q'} \Big) \wedge \bigwedge_{\substack{i \in I, \\ a \in \Sigma}} \Big(t_{i,a} \rightarrow \bigwedge_{\substack{a' \in \Sigma, \\ a' \neq a}} \neg t_{i,a'} \Big) \wedge \bigwedge_{i \in I} \Big(p_i \rightarrow \bigwedge_{\substack{i' \in I, \\ i' \neq i}} \neg p_{i'} \Big),$$

$$\varphi_2 \underset{\text{def}}{=} \bigvee_{q \in Q} s_q \wedge \bigvee_{i \in I} p_i \wedge \bigwedge_{i \in I} \bigvee_{a \in \Sigma} t_{i,a}, \text{ and}$$

$$\varphi_3 \underset{\text{def}}{=} \bigwedge_{i \in I} \Big(p_i \rightarrow \bigwedge_{\substack{i' \in I, \\ i' \neq i, \\ a \in \Sigma}} \big((t_{i',a} \rightarrow \mathrm{AX} t_{i',a}) \wedge (\neg t_{i',a} \rightarrow \mathrm{AX} \neg t_{i',a}) \big) \Big).$$

Next, the formula φ_{init} states that $N, s \models \varphi_{\text{init}}$, for a model $N = (S, R, l)$ and a state $s \in S$, only if s encodes the starting situation of M:

$$\varphi_{\text{init}} \underset{\text{def}}{=} s_{q_0} \wedge p_0 \wedge \bigwedge_{i \in [-n^k, -1]} t_{i,\square} \wedge \bigwedge_{i \in [0, n-1]} t_{i,w_i} \wedge \bigwedge_{i \in [n, n^k]} t_{i,\square}.$$

Finally we need to encode the transition function δ into a formula. Therefore we just need to take care of the tape proposition which corresponds to the position proposition

and change them accordingly to the δ-steps. The formula is defined as

$$\varphi_\delta \underset{\text{def}}{=} \bigwedge_{\substack{q\in Q_\exists,\\ a\in\Sigma,\\ i\in I}} \Big(s_q \wedge p_i \wedge t_{i,a} \rightarrow \bigvee_{(q',a',X)\in\delta(q,a)} \mathrm{EX}\big(s_{q'} \wedge p_{i+x} \wedge t_{i,a'}\big)\Big) \wedge$$
$$\bigwedge_{\substack{q\in Q_\forall,\\ a\in\Sigma,\\ i\in I}} \Big(s_q \wedge p_i \wedge t_{i,a} \rightarrow \bigwedge_{(q',a',X)\in\delta(q,a)} \mathrm{EX}\big(s_{q'} \wedge p_{i+x} \wedge t_{i,a'}\big)\Big) \wedge \bigwedge_{q\in Q_{\text{rej}}} \big(s_q \rightarrow \bot\big),$$

with $x = 1$ if $X = R$, $x = 0$ if $X = N$, and $x = -1$ if $X = L$. That is, φ_δ encodes the transition relation δ of M. Note that the part $\mathrm{AG}\bigwedge_{q\in Q_{\text{rej}}}(s_q \rightarrow \bot)$ eventually refutes rejecting computation trees. It is straightforward to check that M accepts w iff φ is satisfiable. Since $\mathrm{EX}\psi \equiv \neg\mathrm{AX}\neg\psi$, it follows that φ can be expressed in CTL($\{\mathrm{AX}, \mathrm{AG}\}$, BF), and therefore CTL-SAT($\{\mathrm{AX}, \mathrm{AG}\}$, BF) is EXP-hard.

For the cases $T = \{\mathrm{EU}\}$, we modify the above reduction as follows. We replace the expressions $\mathrm{EX}\big(s_{q'} \wedge p_{i+x} \wedge t_{i,a'}\big)$ in φ_δ with $\mathrm{E}\big[(s_q \wedge p_i \wedge t_{i,a})\mathrm{U}(s_{q'} \wedge p_{i+x} \wedge t_{i,a'})\big]$. Analogously, the AX-operators in φ_3 are replaced using AU-expression and then rewritten using the fact that $\mathrm{A}[\psi\mathrm{U}\chi]$ and $\neg\mathrm{E}[\neg\chi\mathrm{U}(\neg\psi \wedge \neg\chi)]$ are equivalent if χ will eventually occur. Lastly, $\mathrm{AG}\psi$ is replaced with $\neg\mathrm{E}[\top\mathrm{U}\neg\psi]$.

For the case $T = \{\mathrm{AU}\}$, we require an additional proposition h denoting the end of the computation of the ATM M on every path. We thus introduce a corresponding formula $\varphi_4 \underset{\text{def}}{=} \mathrm{AF}\big(\pi_{\text{term}} \wedge \neg h \wedge \mathrm{AF}h\big) \wedge \mathrm{A}[\neg h\mathrm{U}\pi_{\text{term}}]$ with $\pi_{\text{term}} \underset{\text{def}}{=} \bigvee_{q\in Q_{\text{acc}}\cup Q_{\text{rej}}} s_q$ that enforces our intuition of h in a satisfying model of the resulting formulae. Consequently, replace $\mathrm{AG}\psi$ with $\mathrm{A}[\psi\mathrm{U}h]$. In particular, $\mathrm{EX}\psi$ can now be replaced with $\mathrm{AF}h \wedge \mathrm{EF}\psi \equiv \mathrm{AF}h \wedge \neg\mathrm{A}[\neg\psi\mathrm{U}h]$. Finally, φ_3 states that, on *all* paths, the contents of all tape cells remains unchanged *until* either the head moves onto the cell or π_{term} holds. □

Now we have seen how the complexity of satisfiability for CTL is trichotomous ranging from NP-, over PSPACE-, to EXP-complete cases. The main ideas have been a small model property for the first cases, an extended modal satisfiability test and the ability to express QBF-VAL formulae for the PSPACE-complete cases, and thirdly a reduction from the word problem for alternating polynomial space Turing machines.

3.1.3 Satisfiability for fragments of CTL*

Turning towards the logic CTL* which allows arbitrary nesting of path quantifiers and temporal operators we can therefore observe an increasing of the complexity in general. One can use several proof ideas from the previous sections to classify fragments of this strictly more expressive logic, that is, we can use Theorems 3.1 and 3.2 in order to classify the Boolean fragments again independently from the temporal operators and path quantifiers for the logic CTL*.

Theorem 3.8.
Let T denote a set of temporal operators and let B be a finite set of Boolean functions such that $[B] \notin \{\mathsf{L}, \mathsf{L}_0\}$. Then CTL*-SAT$(T, B)$ *is*

(1.) equivalent to CTL*-SAT(T, BF) *if* $\mathsf{S}_1 \subseteq [B]$,

(2.) NC^1*-complete under* $\leq_{\text{cd}}$*-reductions if* $\mathsf{S}_{11} \subseteq [B] \subseteq \mathsf{M}$, *and*

(3.) TC^0*-complete for all other cases.*

It remains to consider CTL*-SAT(T, BF) for arbitrary temporal operators. As for CTL, there are three cases. Other than for CTL, the hardest cases are EEXP-complete.

Theorem 3.9.
Let T denote a set of temporal operators. Then CTL*-SAT(T, BF) *is*

(1.) NP*-complete under* $\leq_{\text{cd}}$ *if* $T = \emptyset, \{\mathrm{A}\}, \{\mathrm{F}\}, \{\mathrm{X}\}$,

(2.) PSPACE*-complete under* $\leq_{\text{cd}}$ *if* $T = \{\mathrm{U}\}, \{\mathrm{X}, \mathrm{F}\}, \{\mathrm{X}, \mathrm{U}\}, \{\mathrm{A}, \mathrm{X}\}, \{\mathrm{A}, \mathrm{F}\}$,

(3.) EEXP*-complete under* $\leq_{\text{cd}}$ *in all other cases.*

Proof. For (1.), NP-hardness follows directly by the NP-completeness of SAT [Coo71b, Lev73]. Any formula $\varphi \in \text{CTL}^*(\{\mathrm{A}\}, \mathsf{BF})$ is satisfiable if and only if the formula obtained from φ by deleting all appearances of A is satisfiable (by definition of semantics). For the remaining two cases observe that the equivalence of CTL*-SAT$(\{\mathrm{X}\}, \mathsf{BF})$ and LTL-SAT$(\{\mathrm{X}\}, \mathsf{BF})$, resp., CTL*-SAT$(\{\mathrm{F}\}, \mathsf{BF})$ and LTL-SAT$(\{\mathrm{F}\}, \mathsf{BF})$ is directly achieved by the definition of LTL. As both of the LTL-SAT-problems are NP-complete [BSS+09] the theorem applies.

As for (2.), we have that CTL*-SAT(T, BF) is equivalent to LTL-SAT(T, BF) for the sets $T = \{\mathrm{U}\}, \{\mathrm{X}, \mathrm{F}\}, \{\mathrm{X}, \mathrm{U}\}$ which are PSPACE-complete [BSS+09]. The remaining two cases $\{\mathrm{A}, \mathrm{X}\}$ and $\{\mathrm{A}, \mathrm{F}\}$ can be proven similarly as for the CTL-cases.

Finally, for (3.), we modify the proof given by Vardi showing that CTL*-SAT restricted to $\{\mathrm{A}, \mathrm{X}, \mathrm{U}\}$ and BF is EEXP-hard [VS85b]. In his proof he reduces the word problem for exponential-space alternating Turing machines to the problem CTL*-SAT$(\{\mathrm{A}, \mathrm{X}, \mathrm{U}\}, \mathsf{BF})$ whereas stating hardness under logarithmic-space reductions only, his proof actually yields a $\leq_{\text{cd}}$-reduction. There, a construction consisting of a formula $r \wedge \mathrm{A}\mathrm{G}g$ is used, r describing properties of the root in the model and g characterizing some invariant that needs to hold at every state. At this, a formula simulating a counter visualizes every step of the computation and within the root Vardi makes use of the U operator in a subformula called *init* only. Yet in that context, we may delete the subformulae $\mathrm{X}(S = \Box \mathrm{U} C = 0)$ and $I \wedge \mathrm{X}(I \mathrm{U} C = 0)$ and add the conjunct $\mathrm{G}I \wedge \mathrm{G}\big(S = \Box \rightarrow \mathrm{X}(S = \Box \vee C = 0)\big)$ to the formula *init*. The modified reduction remains correct, for existence of a path encoding the initial configuration of an ATM is assured. This proves hardness for $T = \{\mathrm{A}, \mathrm{X}, \mathrm{F}\}$.

For $T = \{\mathrm{A}, \mathrm{U}\}$, elemination of X essentially leads to the relaxation of the one-to-one correspondence of tape cells and states in a model of the resulting formula. First, we need to construct a counter without the X operator: Let the propositions $c_0, \ldots, c_n$ encode the bits of the counter $C = \sum_{i=0}^{n} c_i 2^i$ and let $C = x$, $x \in \mathbb{N}$, abbreviate the subformula $\bigwedge_{i \in J} c_i \wedge \bigwedge_{i \notin J} \neg c_i$, where $J = \{i \mid \text{bit } i \text{ in } x \text{ is on}\}$. Then the following formula ensures

that the counter C is monotonically increasing:

$$\bigwedge_{i=0}^{n} \neg c_i \wedge \mathrm{A}\Bigg(\bigwedge_{i=0}^{n} \Bigg(\Big(\neg c_i \wedge \bigwedge_{k=0}^{i-1} c_k \rightarrow \Big[\neg c_i \wedge \bigwedge_{k=0}^{i-1} c_k \mathrm{U} c_i \wedge \bigwedge_{k=0}^{i-1} \neg c_k \Big] \Big) \wedge \bigwedge_{k=i+1}^{n} \Big(c_k \rightarrow \Big[c_k \mathrm{U} \bigwedge_{l=0}^{k} \neg c_l \Big] \Big) \wedge \bigwedge_{k=i+1}^{n} \Big(\neg c_k \rightarrow \Big[\neg c_k \mathrm{U} \bigwedge_{l=0}^{k} c_l \Big] \Big) \Bigg) \Bigg).$$

Next, we associate with each tape cell the set of states having the same counter value C. Thus, the *init* formula will be translated to

$$\mathrm{E}\Big((C = 0 \rightarrow S = (q_0, y_1) \wedge I) \wedge \bigwedge_{i=1}^{n-1} (C = i \rightarrow S = y_{i+1} \wedge I) \wedge \wedge \big(C \geq n \rightarrow [S = \Box \wedge I \mathrm{U} C = 0]\big) \Big).$$

Observe that in the formula from above the usage of '=' is just a shortcut and abbreviates some corresponding formulae in Vardis proof. Finally, the formula $\mathrm{A}(\neg \mathit{badpath} \rightarrow \mathit{check})$ that requires a model of φ_f to encode correct transitions between configurations is translated analogously. The EEXP-hardness of the remaining fragments follows. The membership result for EEXP follows from Theorem 2.11. □

3.1.4 About the Affine Cases

The use of Post's lattice as a tool for classifying fragments of various decision problems for arbitrary extensions of propositional logic has been proven adjuvant. Though it seems that the affine functions which consist of a standard base containing the exclusive-or function $\oplus$, are somewhat harder to classify for completeness results than any of the other clones. In [BMS+11, CSTW10, CST10, Tho09, BSS+09, MMS+09, HSS08, Rei01] several fragments corresponding to the affine functions are either left open or lack matching lower bounds for membership results (whereas the last one got matching bounds for several L-cases in some decision problems, too), and in [Tho10] four affine cases are left open for translations between some fragments of nonmonotonic logics. Apart from that there are only few papers that fully classify (read, with matching upper and lower bounds) these affine cases, namely [BMTV09a, BMTV09b]—whereas only the latter explicitly states reductions using affine functions within its technical parts. Thus there is quite some kind of black box surrounding these clones wherefore wondering about their hardness seems evident. Sifting through the landscape of complexity theory for other problems involving or connecting to affine functions which do not emerge from a fragmental analysis by Post's lattice bears the following candidates for stating reductions in order to classify such affine clone fragments:

- PARITY $\underset{\text{def}}{=} \{w \in \{0,1\}^* \mid |w|_1 \equiv 0 \mod 2\}$, see Theorem 2.33 on pg. 27,

- MAJ $\underset{\text{def}}{=} \{w \in \{0,1\}^* \mid |w|_1 \geq |w|_0\}$, see Theorem 2.32 on pg. 27,
- solving equations over the field $\mathbb{Z}_2$ is $\oplus$LOGSPACE-complete under $\leq_m^{\mathsf{AC}^0}$-reductions [BDHM92],
- GOAP $\underset{\text{def}}{=} \{G, s, t \mid G = (V, E)$ is a dag whose vertices have outdegree 0 or 2, $s, t \in V$, the number of all paths from s to t is odd$\}$, which is $\oplus$LOGSPACE-complete under $\leq_{\text{cd}}$-reductions [Rei01],
- 1-in-3-SAT $\underset{\text{def}}{=} \left\{ \bigwedge_{i=1}^{n} \bigvee_{j=1}^{3} l_{ij} \in 3\text{CNF} \mid l_{ij} \text{ are literals}, \exists \theta \text{ s.t.} \theta \models \varphi \text{ and } \forall 1 \leq i \leq n \; \exists! 1 \leq j \leq 3 : \theta(l_{ij}) = \top \right\}$, which is NP-complete w.r.t. $\leq_{\text{cd}}$-reductions [GJ79, p. 221], and
- NAE-SAT $\underset{\text{def}}{=} \left\{ \bigwedge_{i=1}^{n} \bigvee_{j=1}^{3} l_{ij} \in 3\text{CNF} \mid l_{ij} \text{ are literals}, \exists \theta \text{ s.t.} \theta \models \varphi \text{ and } \forall 1 \leq i \leq n \; \exists 1 \leq j \neq k \leq 3 : \theta(l_{ij}) \neq \theta(l_{ik}) \right\}$, which is NP -complete w.r.t. $\leq_{\text{cd}}$-reductions (see [Pap94]).

Concerning the aforementioned publications the decision problems from above are the usual suspects used for classifying the respective fragments in some logic. Usually one main inconvenience is the lack of conjunction in the clone in order to encode the needed properties. Another difficult circumstance is a more informal aspect which is hard to assess, i.e., $\oplus$ being counterintuitive to the usual logical reasoning. Typically in order to satisfy a formula satisfying subparts (i.e., subformulae) cannot lead to false for the recent evaluation which can be in particular the case for affine functions. Therefore it is very hard to estimate the outcome when constructing some satisfying model due to the interaction between any subformulae. This is also a reason attempting to solve this somehow semantical problem by working through syntactical arguments becomes a valid and promising approach [HSS08, Theorem 3.19]. Using this technique permits us to state better upper bounds for some fragments for CTL- and CTL*-formulae which are shown in Theorems 3.10 and 3.11.

Theorem 3.10.
Let B be a finite set of Boolean functions with $[B] \subseteq \mathsf{L}$. Then

(1.) CTL-SAT$(\{\text{AX}\}, B)$ *is in* P,

(2.) CTL-SAT$(\{\text{AG}, \text{AX}\}, B)$ *is in* NP.

Proof. (1.) At first rewrite an input formula $\varphi \in \text{CTL}(\{\text{AX}\}, B)$ s.t. φ consists only of EX-operators and no AX-operator by substituting $\text{AX}\psi$ with $(\text{EX}(\psi \oplus \top)) \oplus \top$ for each occurrence in φ. Observe that this is not problematic for $\top \notin [B]$ as we aim to construct a circuit in the following. Now let the substitution of the AXs be denoted by φ'. In the next step construct from φ' the corresponding modal formula $\varphi_{\text{ml}} \underset{\text{def}}{=} \varphi'|_{[\text{EX}/\Diamond]}$, and then the corresponding modal Boolean circuit $C_{\varphi_{\text{ml}}}$ for the frame class $\mathscr{F} = \text{KD}$. Finally run

the algorithm $\oplus$-SAT with input $C_{\varphi_{\mathrm{ml}}}$ from [HSS08]. Each of those steps clearly runs in polynomial time.

(2.) Similar to (1.) replace all AGψ's with $(\mathrm{EF}(\psi \oplus \top)) \oplus \top$ and let denote this change by φ'. In the next step for each $\mathrm{EF}\psi \in \mathbf{SF}(\varphi')$ guess nondeterministically an $0 \leq i \leq |\varphi'|$ and replace $\mathrm{EF}\psi$ by $\mathrm{EX}^i\psi$ where $\mathrm{EX}^0\psi \underset{\mathrm{def}}{=} \psi$ and otherwise with the i-times concatenation of EX. A straightforward inductive argument proves that guessing until the linear depth reached suffices. Then proceed as in (1.) which leads to an NP-algorithm. Finally the case {AG, AX} Follows from a combination of (1.) and the construction for {AG}. □

Theorem 3.11.
Let B be a finite set of Boolean functions with $[B] = \mathsf{L}$. Then

(1.) CTL*-SAT($\{\mathrm{A}\}, B$) *is* $\oplus$LOGSPACE*-complete w.r.t.* $\leq_{\mathrm{cd}}$*-reductions. This also holds for* $[B] = \mathsf{L}_0$.

(2.) CTL*-SAT($\{\mathrm{X}\}, B$) *is in* P.

Proof. (1.) This problem is equivalent to SAT(L) (resp., SAT(L_0)) and therefore also $\oplus$LOGSPACE-complete w.r.t. $\leq_{\mathrm{cd}}$-reductions [Rei01].

(2.) A straightforward modification of the algorithm $\oplus$-SAT from [HSS08]. □

3.1.5 Fragments of Extensions of CTL: Fairness, Succinctness, and LTL$^+$

As introduced in Chapter 2 two popular extensions of CTL are, on the one hand, ECTL with the ability to express fairness constraints through the operators $\overset{\infty}{\mathrm{F}}$ and $\overset{\infty}{\mathrm{G}}$ and, on the other hand CTL$^+$, permitting Boolean combinations of temporal operators which is as expressive as CTL but the former being much more succinct.

The classification of almost all fragments of CTL-SAT(T, B) and CTL*-SAT(T, B) for any set of Boolean functions B and any set of temporal operators and path quantifiers T entails the question whether any of the results transfer to these extensions at all. As $\mathrm{CTL}(T, B) \subset \mathrm{CTL}^+(T, B), \mathrm{ECTL}(T, B) \subset \mathrm{CTL}^*(T, B)$ the respecting lower and upper bounds apply to CTL$^+$-SAT(T, B) and ECTL-SAT(T, B).

Corollary 3.12.
Let B be a finite set of Boolean functions s.t. $[B] \notin \{\mathsf{L}, \mathsf{L}_0\}$. Then CTL$^+$-SAT($T, B$) *is*

(1.) equivalent to CTL$^+$-SAT(T, BF) *if* $\mathsf{S}_1 \subseteq [B]$,

(2.) NC^1*-complete under* $\leq_{\mathrm{cd}}$*-reductions if* $\mathsf{S}_{11} \subseteq [B] \subseteq \mathsf{M}$, *and*

(3.) TC^0*-complete for all other cases.*

This result is immediately obtained by applying Theorems 3.2, 3.3 and 3.8. We continue with the fragments induced by sets of temporal operators and path quantifiers.

Theorem 3.13.
Let T be a set of temporal operators and path quantifiers. Then CTL$^+$-SAT(T, BF) *is*

(1.) NP*-complete if* $T = \emptyset, \{A\}, \{X\}, \{F\}, \{X,F\}, \{U\}, \{X,U\}$,

(2.) PSPACE*-complete if* $T = \{A,F\}, \{A,X\}$, *and*

(3.) EEXP*-complete otherwise.*

Proof. All non-LTL cases in (1.) and (2.), and the cases containing U in (3.) follow directly from Theorems 3.4 and 3.9 where for $\{A,U\}$ the modifications in the proof already consist of CTL$^+$-formulae.

Observe that the LTL upper bounds immediately transfer to our respecting cases due to CTL$^+$ being a more restricted LTL-case. Hence for $T = \{X\}, \{F\}$ both lead to NP-completeness due to the trivial lower bound by SAT. Thereby only the upper bounds for $\{X,F\}, \{U\}, \{X,U\}$ remain to be classified. Satisfiability for CTL$^+(\{X,F\}, \mathsf{BF})$ is easier than for CTL* (i.e., in NP). Let $\psi \in \mathrm{CTL}^+$, and let the function h be inductively defined as

$$\begin{aligned}
h(p) &\underset{\text{def}}{=} p_1, \quad \text{for } p \in \mathsf{PROP},\\
h(p_i) &\underset{\text{def}}{=} p_{i+1} \quad \text{for } p \in \mathsf{PROP}, i \in \mathbb{N}\\
h(\mathrm{O}\psi) &\underset{\text{def}}{=} \mathrm{O}h(\psi) \quad \text{for } \psi \in \mathrm{CTL}^+, \mathrm{O} \in \{X,F,G,U\},\\
h(f(\psi)) &\underset{\text{def}}{=} f(h(\psi)) \quad \text{for } \psi \in \mathrm{CTL}^+, f \in \mathsf{BF}.
\end{aligned}$$

Further let $h^i(\alpha)$ denote the i-times concatenation of h, i.e., $h^i(\alpha) = \overbrace{h(h(\cdots h(}^{i \text{ times}}\alpha)\cdots))$. Now let $\varphi \in \mathrm{CTL}^+(\{X,F\}, \mathsf{BF})$ and inductively construct the mapped formula φ' by the following rules:

- for each $X\psi \in \mathbf{SF}(\varphi)$: replace $X\psi$ with $h(\psi)$,
- for each $G\psi \in \mathbf{SF}(\varphi)$: replace $G\psi$ with $\psi \wedge \bigwedge_{i=1}^{\#_T(\varphi)} h^i(\psi)$,
- for each $F\psi \in \mathbf{SF}(\varphi)$: replace $F\psi$ with $\left(\psi \vee \bigvee_{i=1}^{\#_T(\varphi)} h^i(\psi)\right)$ to φ.

Now one can easily show that $\varphi \in \mathrm{CTL}^+$-SAT$(\{X,F\}, \mathsf{BF})$ iff $\varphi' \in$ SAT by observing the lack of temporal operators in each of the ψs from above. The sufficient linear depth stems from the same argumentation as for the CTL-cases.

If one proceeds and allows U as temporal operator one can extend this reduction to SAT with the following rule

- for each $\psi U \chi \in \mathbf{SF}(\varphi)$: replace $\psi U \chi$ with

$$\left(\chi \vee \bigvee_{i=1}^{\#_T(\varphi)} \left(\psi \wedge h^i(\chi) \wedge \bigwedge_{j=1}^{i-1} h^j(\psi)\right)\right).$$

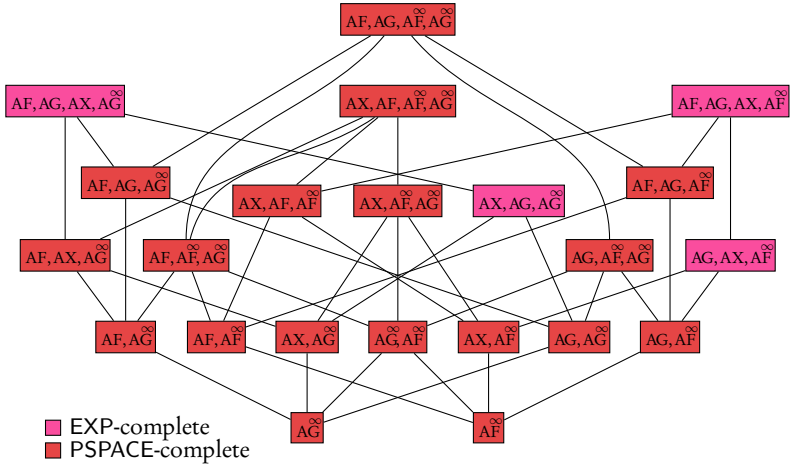


Figure 3.4: The relevant part of the lattice induced by all ECTL-operators. Each node is labelled with a minimal set of operators without any restrictions on the Boolean connectives.

Thus differently to CTL* the satisfiability problem for the logic LTL$^+$ which is defined as LTL without nesting of temporal operators (analogously to CTL$^+$) always stays NP-complete regardless of which temporal operators are allowed beyond all Boolean functions.

For the remaining case $T = \{\mathrm{A}, \mathrm{X}, \mathrm{F}\}$ in (3.) we need to remove the nesting of the temporal operators in the formulae *init* and *badpath* in order to get the valid EEXP lower bound. We achieve this validity for CTL$^+$ by setting the formula *init* to

$$\mathrm{E}\Big((C = 0 \to S = (q_0, y_1) \wedge I) \wedge \bigwedge_{i=1}^{n-1} (C = i \to S = y_{i+1} \wedge I) \wedge \Big(C \geq n \to [S = \Box \wedge \mathrm{G}\big(C \neq 0 \to (I \vee C = 0)\big) \wedge \mathrm{F}(C = 0)]\Big)\Big),$$

and finally the *badpath* rules can be adjusted similar to the construction of the counter with an CTL($\{\mathrm{AX}, \mathrm{EG}\}$, BF)-formula as in Example 2.24. □

An overview how the results with respect to the temporal operators arrange in a lattice for CTL$^+$ is depicted in Figure 3.6.

Now we consider fragments containing the fairness operators $\overset{\infty}{\mathrm{F}}$ and $\overset{\infty}{\mathrm{G}}$. The fragments that need to be classified are depicted in Figure 3.4. The complexity degrees for the remaining fragments directly follow from the lower and upper bounds of the fragments shown in this figure and the ones shown in Figure 3.6 for CTL.

Theorem 3.14.
Let T be a set of ECTL *operators. Then* ECTL-SAT(T, BF) *is*

(1.) PSPACE*-complete if* $\{\mathrm{AG}\} \subseteq T \subseteq \{\mathrm{AF}, \mathrm{AG}, \mathrm{A}\overset{\infty}{\mathrm{F}}, \mathrm{A}\overset{\infty}{\mathrm{G}}\}$.

(2.) PSPACE*-complete if* $\{\mathrm{AX}\} \subseteq T \subseteq \{\mathrm{AX}, \mathrm{AF}, \mathrm{A}\overset{\infty}{\mathrm{F}}, \mathrm{A}\overset{\infty}{\mathrm{G}}\}$.

(3.) PSPACE*-complete if* $\{\mathrm{A}\overset{\infty}{\mathrm{G}}\} \subseteq T \subseteq \{\mathrm{AF}, \mathrm{A}\overset{\infty}{\mathrm{F}}, \mathrm{A}\overset{\infty}{\mathrm{G}}\}$.

(4.) PSPACE*-complete if* $\{\mathrm{A}\overset{\infty}{\mathrm{F}}\} \subseteq T \subseteq \{\mathrm{AF}, \mathrm{A}\overset{\infty}{\mathrm{F}}\}$.

Otherwise for any remaining case if $\{\mathrm{A}\overset{\infty}{\mathrm{G}}\} \subseteq T$ *or* $\{\mathrm{A}\overset{\infty}{\mathrm{F}}\} \subseteq T$, *then* ECTL-SAT$(T, \mathsf{BF})$ *is* EXP*-complete.*

Proof. For (1.) observe that CTL*-SAT$(\{\mathrm{A}, \mathrm{F}\}, \mathsf{BF}) \in$ PSPACE by Theorem 3.9. This lets us use the equivalences $\mathrm{A}\overset{\infty}{\mathrm{F}}\varphi = \mathrm{AGF}\varphi$, $\mathrm{A}\overset{\infty}{\mathrm{G}}\varphi = \mathrm{AFG}\varphi$, $\mathrm{E}\overset{\infty}{\mathrm{F}}\varphi = \mathrm{EGF}\varphi$, and $\mathrm{E}\overset{\infty}{\mathrm{G}}\varphi = \mathrm{EFG}\varphi$ to run the CTL*-SAT-algorithm instead. The lower bound is shown in Theorem 3.4 (2.).

The case (2.) needs a little bit more work. Observe that $\mathrm{E}\overset{\infty}{\mathrm{G}}\varphi \equiv \mathrm{EFG}\varphi \equiv \mathrm{EFEG}\varphi$ and therefore also $\mathrm{A}\overset{\infty}{\mathrm{F}}\varphi \equiv \mathrm{AGF}\varphi \equiv \mathrm{AGAF}\varphi$ holds. Thus we can just replace any occurrence of these $\mathrm{A}\overset{\infty}{\mathrm{F}}$- and $\mathrm{E}\overset{\infty}{\mathrm{G}}$-preceded subformulae with the respective CTL-operator representation and proceed with Algorithms 3.1 and 3.2. Hence we only need to extend the algorithms for the cases $\mathrm{A}\overset{\infty}{\mathrm{G}}$ and $\mathrm{E}\overset{\infty}{\mathrm{F}}$, because here, the equivalence for the opposite cases cannot be applied as visualized in Figure 3.5 which also follows from [Eme90, Fig.3]. To satisfy subformulae of the kind $\mathrm{E}\overset{\infty}{\mathrm{F}}\varphi$ in some state s of some Kripke structure $\mathfrak{K}$ requires an infinite path $\pi = \pi_1\pi_2\ldots$ such that (i) $\pi_1 = s$, (ii) there exists an index $k \geq 1$ such that π_k stands for the point where the loop starts, and (iii) there exists at least one state $k \leq j$ with $\varphi \in l(\pi_j)$. Thus the algorithm only needs to guess these two meaningful states for $\mathrm{E}\overset{\infty}{\mathrm{F}}\varphi$ and verifies between leaving π_k and reentering π_j that there was a state where φ is satisfied. For the opposite kind namely the subformulae $\mathrm{A}\overset{\infty}{\mathrm{G}}\varphi$ one can substitute these formulae with $\neg\mathrm{E}\overset{\infty}{\mathrm{F}}\neg\varphi$ and just use the same algorithm as before in order to verify unsatisfiability. As PSPACE is closed under complement we achieve the desired upper bound result. Now we need to combine these insights with the fixpoint characterization technique from Lemma 3.6 on page 33 (for the case $T = \{\mathrm{AX}, \mathrm{AF}\}$) and achieve a PSPACE-algorithm stating the desired upper bound.

For (3.) we need to modify the reduction from Theorem 3.4 (2.). Therefore we replace each EF-operator with $\mathrm{E}\overset{\infty}{\mathrm{F}}$ and each AG-operator with $\mathrm{A}\overset{\infty}{\mathrm{G}}$ on page 34. Observe that these substitutions only slightly change the resulting Kripke structure for the satisfying case in a way where all worlds containing q_n become reflexive. This change is triggered by the $\mathrm{E}\overset{\infty}{\mathrm{F}}$-operators that require a global path.

For (4.) we proceed as for (3.) and substitute each AG operator with $\mathrm{A}\overset{\infty}{\mathrm{F}}$, and EF operator with $\mathrm{E}\overset{\infty}{\mathrm{G}}$ changing the satisfying tree-like structure only in the leaf-states to be reflexive. □

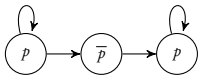


Figure 3.5: Example showing that AFAGp and $\overset{\infty}{\mathrm{AG}}p$ are not equivalent.

Finally we deduce a similar corollary expressing the results for the Boolean fragments of ECTL which again uses the Lewis knack and the argumentation as in Theorems 3.1 and 3.2.

Corollary 3.15.
Let B be a finite set of Boolean functions s.t. $[B] \notin \{\mathsf{L}, \mathsf{L}_0\}$. Then ECTL-SAT$(T, B)$ *is*

(1.) equivalent to ECTL-SAT(T, BF) *if* $\mathsf{S}_1 \subseteq [B]$,

(2.) NC^1*-complete under* $\leq_{\mathrm{cd}}$*-reductions if* $\mathsf{S}_{11} \subseteq [B] \subseteq \mathsf{M}$, *and*

(3.) TC^0*-complete for all other cases.*

3.1.6 Conclusion

Considering all possible combinations of temporal operators and Boolean functions we classified almost all fragments emerging from this perspective—only the affine cases resisted getting fully classified again (cf. Section 3.1.4). The decision problem CTL-SAT(T, BF) is, on the one hand, a trichotomy for all its temporal operator fragments ranging from NP-, to PSPACE-, and to EXP-completeness; where AX and AG alone lead to PSPACE-complete cases and both together to an EXP-complete fragment as shown in Figure 3.6. On the other hand, studying the Boolean fragments for this decision problem we classified the complexity as a trichotomy from EXP-complete cases from S_1 up to BF (cf. Figure 3.7), NC^1-complete fragments for the monotone and 1-separating cases between S_{11} and M, and TC^0-completeness for all other clones in the lattice, whereby each of the three latter cases is independent from the allowed temporal operators.

In 1983, Ben-Ari, Pnueli, and Manna introduced the first concept of branching time logics with their logic UB and also proposed the logics CTL$^+$ and UB$^+$ which are defined by Boolean combination of paths (cf. [BAPM83]). Weber remarks in [Web09b] that the precise complexity of the satisfiability problem for the logic UB$^+$ is still unknown. Through our classification we give the answer to this question: UB$^+$-SAT is equal to CTL$^+$-SAT$(\{\mathrm{A}, \mathrm{F}, \mathrm{X}\}, \mathsf{BF})$ which is EEXP-complete by Theorem 3.13.

Nevertheless, the classification for the more expressive logic CTL* and its satisfiability problem processed almost similar to the one for CTL whereas the therewith connected highest complexity degree is EEXP-completeness. One part of the result encompasses the classification of the linear time temporal logic LTL which had been classified previously by Bauland et al. (cf. [BSS$^+$09]). We accentuate again that the results for the Boolean fragments are also independent from the allowed temporal operators and path quantifiers.

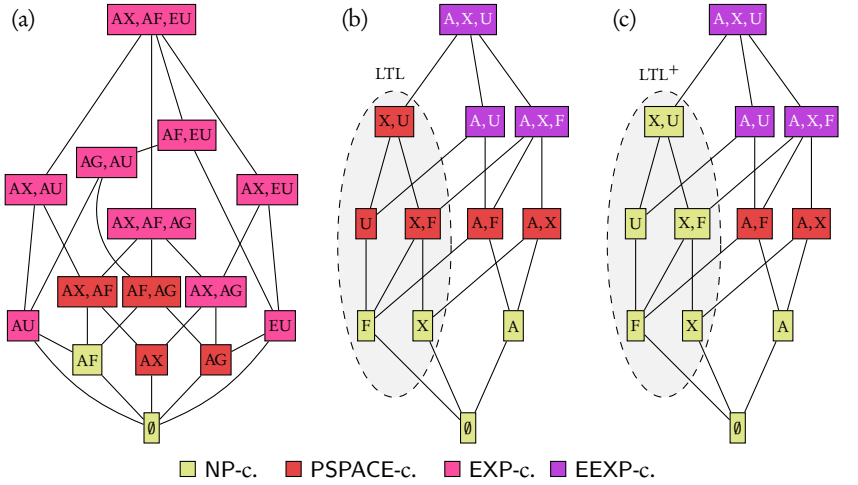


Figure 3.6: The complexity of (a) CTL-SAT(T, BF), (b) CTL*-SAT(T, BF), and (c) CTL$^+$-SAT(T, BF), i.e., without any restrictions to the Boolean functions.

Moreover in Section 3.1.5 we investigated the extension CTL$^+$ and achieved a trichotomy for the temporal fragments whereas for the Boolean fragments the logic behaves as CTL*. Within we gave results for all operator fragments of the emerging logic LTL$^+$ which stayed always NP-complete due to its lack of nested temporal operators.

Furthermore we classified the extension ECTL into PSPACE- and EXP-complete fragments where the rule for determining the complexity for ECTL-SAT(T, B) is the following: if $\mathsf{S}_1 \subseteq [B]$, then ECTL-SAT($T, B$) is PSPACE-complete unless T contains a combination of CTL-operators for which CTL-SAT(T, B) is already EXP-complete (for this rule we assume that T contains at least one real ECTL-operator), otherwise the problem is EXP-complete as well. For the remaining cases we are always inside NC^1 if $[B] \notin \{\mathsf{L}_0, \mathsf{L}\}$.

Thus the next step would be closing the gaps for the affine functions as motivated in Section 3.1.4 comprising the possible combinations of L, L_0 together with all temporal operators. Moreover a universal study of optimization possibilities for algorithms influenced by this classification into temporal SAT-solving would be of great interest.

Another further step includes the cases for the operator R which is the duality to U.

3.2 Model Checking in CTL and CTL*

Fragments of the model checking problem has been studied previously in the context of linear temporal logic, LTL, by Sistla and Clarke [SC85] and Markey [Mar04]. They introduced the restricted use of negation, interaction of future and past operators, and separated tractable from intractable cases. The *path model checking problem* in LTL, i.e.,

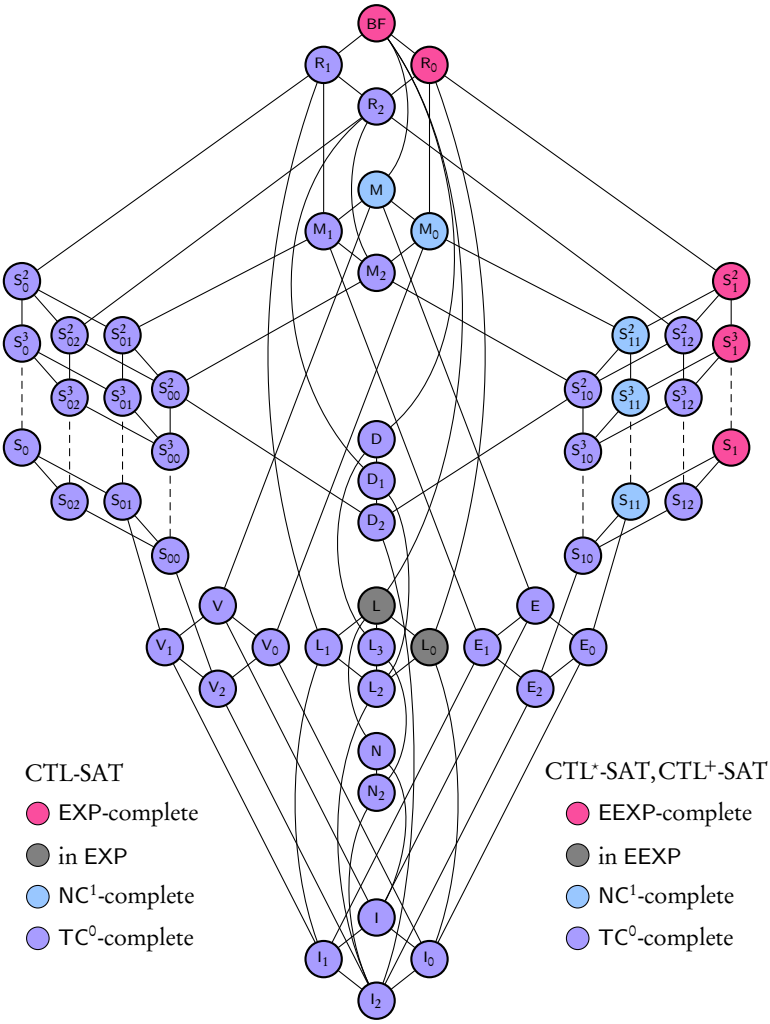


Figure 3.7: The complexity of CTL-SAT, CTL$^+$-SAT, and CTL*-SAT, without restrictions on the temporal operators.

the problem given a nonempty finite path π in a Kripke structure and a formula φ asking whether $\pi \models \varphi$ does hold is investigated in [KF09]. There, Kuhtz and Finkbeiner exhibit an $\mathsf{AC}^1(\mathsf{LOGDCFL})$ algorithm for this problem.

From Theorem 2.17 we know that the model checking problem for the logic CTL is complete for the complexity class P. One of our main motivations was to identify the border between tractability and intractability by means of clones within Post's lattice. Now as the model checking problem CTL-MC is tractable, it hence does not fit into this approach. A question connected to tractable complexity classes involves the search of parallel algorithms. Completeness for the class P prohibits such fast parallel algorithms which are settled inside the class NC [Vol99]. Further the typical model checking algorithms involve the basis $\{\wedge, \vee, \neg\}$ of BF as described in [CGP99]. Therefore we will not follow the classification approach of Post's lattice for the model checking problem and stick to the one of Sistla et al. which seems in first place more viable in the context of the search of parallel algorithms. We define the following fragments of $\mathrm{CTL}(T)$ and $\mathrm{CTL}^+(T)$:

- $\mathrm{CTL}_{\mathrm{pos}}(T)$ and $\mathrm{CTL}^+_{\mathrm{pos}}(T)$ (positive)
  CTL- or temporal operators may not occur in the scope of a negation.
- $\mathrm{CTL}_{\mathrm{a.n.}}(T)$ and $\mathrm{CTL}^+_{\mathrm{a.n.}}(T)$ (atomic negation)
  Negations appear only directly in front of atomic propositions.
- $\mathrm{CTL}_{\mathrm{mon}}(T)$ and $\mathrm{CTL}^+_{\mathrm{mon}}(T)$ (monotone)
  No negations allowed.

In the original notation of Sistla and Clarke they used $\widetilde{\mathrm{L}}(T)$ for $\mathrm{CTL}_{\mathrm{a.n.}}(T)$ and $\mathrm{L}^+(T)$ for $\mathrm{CTL}_{\mathrm{pos}}(T)$. At first we will consider the fragments of CTL and later in Section 3.2.2 we will classify the extensions of CTL with respect to the model checking problem.

Studying the interreducibility of these fragments, we obtain the following relations between model checking for fragments of CTL with restricted negation:

Lemma 3.16.
For every set T of CTL*-operators, it holds that*

$$\mathrm{CTL}_{\mathrm{mon}}\text{-MC}(T) \leq_{\mathbf{cd}} \mathrm{CTL}_{\mathrm{a.n.}}\text{-MC}(T) \leq_{\mathbf{cd}} \mathrm{CTL}_{\mathrm{pos}}\text{-MC}(T).$$

Furthermore, atomic negation can be eluded for model checking, that is, it holds that

$$\mathrm{CTL}_{\mathrm{a.n.}}\text{-MC}(T) \leq_{\mathbf{cd}} \mathrm{CTL}_{\mathrm{mon}}\text{-MC}(T).$$

Proof. The first two reductions are immediate by using the identity function as reduction function. For the second proposition, let $\mathfrak{K} = (W, R, \eta)$ be a Kripke structure and let φ be a $\mathrm{CTL}_{\mathrm{a.n.}}(T)$-formula using the propositions $p_1, \ldots, p_n$. Every negation in φ appears directly in front of an atomic proposition. We obtain φ' by replacing every negative literal $\neg p_i$ with a fresh atomic proposition q_i. Further the labeling function needs to be modified. Therefore we define $\mathfrak{K}' = (W, R, \eta')$, where $\eta'(w) = \eta(w) \cup \{q_i \mid p_i \notin \eta(w), 1 \leq i \leq n\}$. Observe that this construction correctly interacts with contradictory subformulae like $p_1 \wedge \neg p_1$. It is easy to see that it holds that $\mathfrak{K}, w \models \varphi$ iff $\mathfrak{K}', w \models \varphi'$ for all $w \in W$. The mapping $(\mathfrak{K}, w, \varphi) \mapsto (\mathfrak{K}', w, \varphi')$ can be performed by an AC^0-circuit. □

For giving the full picture of the classification it is vital to express the positive fragments by only monotone fragments. This reducibility, i.e., $\mathrm{CTL}_{\mathrm{pos}}\text{-MC}(T) \leq_{\mathrm{cd}} \mathrm{CTL}_{\mathrm{mon}}\text{-MC}(T)$ will be proven in the Section *The Power of Negation* on page 60.

3.2.1 Model Checking CTL and $\mathrm{CTL}_{\mathrm{pos}}$

In this section we will present our main results on the complexity of model checking for CTL and $\mathrm{CTL}_{\mathrm{pos}}$. Model checking for CTL in general was proven to be P-complete (see Theorem 2.17). By showing that only one temporal operator is sufficient to obtain P-hardness, we improve the lower bound of this result.

Theorem 3.17.
For each nonempty set T of CTL*-operators,* $\text{CTL-MC}(T)$ *is* P*-complete. If $T = \emptyset$, then* $\text{CTL-MC}(T)$ *is* NC^1*-complete.*

Now, considering only formulae from $\mathrm{CTL}_{\mathrm{pos}}$ the situation changes and the complexity of model checking exhibits a dichotomous behavior.

Theorem 3.18.
Let T be any set of CTL*-operators. Then* $\mathrm{CTL}_{\mathrm{pos}}\text{-MC}(T)$ *is*

(1.) NC^1*-complete if $T = \emptyset$,*

(2.) LOGCFL*-complete if $\emptyset \subsetneq T \subseteq \{\mathrm{EX}, \mathrm{EF}\}$ or $\emptyset \subsetneq T \subseteq \{\mathrm{AX}, \mathrm{AG}\}$, and*

(3.) P*-complete otherwise.*

We split the proofs of Theorems 3.17 and 3.18 into the upper and lower bounds in the following two subsections.

Upper Bounds

As previously described model checking for CTL is known to be solvable in P. While this upper bound also applies to $\mathrm{CTL}_{\mathrm{pos}}\text{-MC}(T)$ (for every T), we improve it for positive CTL-formulae using only EX and EF, or only AX and AG.

Proposition 3.19.
Let T be a set of CTL*-operators such that $T \subseteq \{\mathrm{EX}, \mathrm{EF}\}$ or $T \subseteq \{\mathrm{AX}, \mathrm{AG}\}$. Then the problem* $\mathrm{CTL}_{\mathrm{pos}}\text{-MC}(T)$ *is in* LOGCFL.

Proof. First consider the case $T \subseteq \{\mathrm{EX}, \mathrm{EF}\}$. We claim that Algorithm 3.3 runs on a LOGCFL machine and recursively decides whether the Kripke structure $\mathfrak{K} = (W, R, \eta)$ satisfies the $\mathrm{CTL}_{\mathrm{pos}}(T)$-formula φ in state $w_0 \in W$. The algorithm uses a stack S which stores pairs $(\varphi, w) \in \mathrm{CTL}_{\mathrm{pos}}(T) \times W$ and R^* denotes the transitive closure of R (which can be computed by a LOGCFL-machine through a binary counter).

Algorithm 3.3 always terminates because each subformula of φ is pushed to the stack S at most once. For correctness, an induction on the structure of formulae shows that

Algorithm 3.3: LOGCFL-algorithm for Proposition 3.19.

Input : a Kripke structure $\mathfrak{K} = (W, R, \eta)$, $w_0 \in W$, $\varphi \in \mathrm{CTL}_{\mathrm{pos}}(T)$
Output : true if and only if $\mathfrak{K}, w_0 \models \varphi$

```
1  push(S, (φ, w_0));
2  while S is not empty do
3      (φ, w) ← pop(S);
4      if φ is a propositional formula then
5          if φ evaluates to false in w under η then return false;
6      else if φ = α ∧ β then push(S, (β, w)) and push(S, (α, w));
7      else if φ = α ∨ β then nondeteterministically push(S, (α, w)) or push(S, (β, w));
8      else if φ = EXα then
9          nondeteterministically choose w' ∈ {w' | (w, w') ∈ R};
10         push(S, (α, w'));
11     else if φ = EFα then
12         nondeteterministically choose w' ∈ {w' | (w, w') ∈ R*};
13         push(S, (α, w'));
14 return true;
```

Algorithm 3.3 returns **false** if and only if for the most recently removed pair (ψ, w) from S, we have $\mathfrak{K}, w \not\models \psi$. Thence, in particular, Algorithm 3.3 returns **true** iff $\mathfrak{K}, w \models \varphi$.

Algorithm 3.3 can be implemented on a nondeterministic polynomial-time Turing machine that besides its (unbounded) stack uses only logarithmic memory for the local variables. Thus $\mathrm{CTL}_{\mathrm{pos}}$-MC($T$) is in LOGCFL.

The proof for the case $T \subseteq \{\mathrm{AX}, \mathrm{AG}\}$ is analogous and follows from closure of LOGCFL under complementation. □

Finally, for the trivial case where no CTL-operators are present, model checking CTL($\emptyset$)-formulae is equivalent to the problem of evaluating a propositional formula. This problem is known to be solvable in NC^1 [Bus87].

Lower Bounds

The P-hardness of model checking for CTL was stated in [Sch02] first. We improve this lower bound and concentrate on the smallest fragments w.r.t. CTL-operators of monotone CTL with P-hard model checking. For the monotone fragment we need to take care of the dual operator R of U, which is therefore defined through $[\chi \mathrm{R} \pi] \equiv \neg[\neg\chi \mathrm{U} \neg\pi]$.

Proposition 3.20.
Let T denote a set of CTL*-operators. Then* $\mathrm{CTL}_{\mathrm{mon}}$-MC($T$) *is* P*-hard if T contains an existential and a universal* CTL*-operator.*

Proof. First, assume that $T = \{\mathrm{AX}, \mathrm{EX}\}$. We give a generic reduction from the word problem for alternating Turing machines working in logarithmic space, which follows a similar approach as the classical proof idea (see [Sch02, Theorem 3.8]), and which we

will modify in order to be useful for other combinations of CTL-operators. Starting with an arbitrary language $A \in \mathsf{ALOGSPACE}$, let M be an alternating logspace Turing machine s.t. $L(M) = A$, and let x be an input to M. We may assume w.l.o.g. that each transition of M leads from an existential to a universal configuration and vice versa. Further we may assume that each computation of M ends after the same number $p(n)$ of steps, where p is a polynomial and n is the length of M's input x. Furthermore we may assume that there exists a polynomial q such that $q(n)$ is the number of configurations of M on any input of length n.

Let $c_1, \ldots, c_{q(n)}$ be an enumeration of all possible configurations of M on input x, starting with the initial configuration c_1. We construct a Kripke structure $\mathfrak{K} = (W, R, \eta)$ by defining the set $W \underset{\text{def}}{=} \{c_i^j \mid 1 \le i \le q(n), 0 \le j \le p(n)\}$ and the relation $R \subseteq W \times W$ as

$$\begin{aligned} R \underset{\text{def}}{=} & \left\{(c_i^j, c_k^{j+1}) \;\middle|\; M \text{ reaches configuration } c_k \text{ from } c_i \text{ in one step}, 0 \le j < p(n)\right\} \\ & \cup \left\{(c_i^j, c_i^j) \;\middle|\; c_i^j \text{ has no successor}, 1 \le i \le q(n), 0 \le j < p(n)\right\} \\ & \cup \left\{(c_i^{p(n)}, c_i^{p(n)}) \;\middle|\; 1 \le i \le q(n)\right\}. \end{aligned}$$

The labelling function η is defined for all $c_i^j \in W$ as

$$\eta(c_i^j) \underset{\text{def}}{=} \begin{cases} \{t\}, & \text{if } c_i \text{ is an accepting configuration and } j = p(n) \\ \emptyset, & \text{otherwise} \end{cases}$$

where t is the only atom used by this labelling. It then holds that

$$M \text{ accepts } x \quad \text{iff} \quad \mathfrak{K}, c_1^0 \models \psi_1\Big(\psi_2(\cdots \psi_{p(n)}(t)) \cdots\Big),$$

where $\psi_i(x) \underset{\text{def}}{=} \mathrm{AX}(x)$ if M's configurations *before* the ith step are universal, and $\psi_i(x) \underset{\text{def}}{=} \mathrm{EX}(x)$ otherwise. Notice that the constructed CTL-formula does not contain any Boolean connective. Since $p(n)$ and $q(n)$ are polynomials, the size of $\mathfrak{K}$ and φ is polynomial in the size of (M, x). Moreover, $\mathfrak{K}$ and φ can be constructed from M and x using AC^0-circuits. Thus, $A \le_{\text{cd}} \mathrm{CTL}_{\text{mon}}\text{-MC}(\{\mathrm{AX}, \mathrm{EX}\})$ for all $A \in \mathsf{ALOGSPACE} = \mathsf{P}$.

For $T = \{\mathrm{AF}, \mathrm{EG}\}$ we modify the above reduction by defining the labelling function η and the formula ψ_i as follows:

$$\begin{aligned} \eta(c_i^j) &\underset{\text{def}}{=} \begin{cases} \{d_j, t\}, & \text{if } c_i \text{ is an accepting configuration and } j = p(n) \\ \{d_j\}, & \text{otherwise} \end{cases} \\ \psi_i(x) &\underset{\text{def}}{=} \begin{cases} \mathrm{AF}(d_i \wedge x), & \text{if } M\text{'s configurations } \textit{before} \text{ step } i \text{ are universal,} \\ \mathrm{EG}(D_i \vee x), & \text{otherwise,} \end{cases} \end{aligned} \tag{3.1}$$

where d_j are atomic propositions encoding the 'execution time point' of the respective configurations and $D_i = \bigvee_{i \neq j \in \{0,\ldots,p(n)\}} d_j$.

For the combinations of T being one of $\{\mathrm{AF}, \mathrm{EF}\}$, $\{\mathrm{AF}, \mathrm{EX}\}$, $\{\mathrm{AG}, \mathrm{EG}\}$, $\{\mathrm{AG}, \mathrm{EX}\}$, $\{\mathrm{AX}, \mathrm{EF}\}$, and $\{\mathrm{AX}, \mathrm{EG}\}$, the P-hardness of $\mathrm{CTL}_{\text{mon}}\text{-MC}(T)$ is obtained using analogous modifications to η and the ψ_i's.

For the remaining combinations involving the until operator, observe that w.r.t. the Kripke structure $\mathfrak{K}$ as defined in (3.1), $\mathrm{AF}(d_i \wedge x)$ and $\mathrm{EG}(D_i \vee x)$ are equivalent to $\mathrm{A}[d_{i-1}\mathrm{U}x]$ and $\mathrm{E}[d_{i-1}\mathrm{U}x]$., and that R and U are duals. □

In the presence of arbitrary negation, universal operators are definable by existential operators and vice versa. Hence, from Proposition 3.20 we obtain the following corollary.

Corollary 3.21.
The model checking problem CTL-MC(T) *is* P*-hard for each nonempty set* T *of* CTL*-operators.*

We will now see that for monotone CTL, in most cases even one operator suffices to make model checking P-hard:

Proposition 3.22.
Let T *denote a set of* CTL*-operators. Then* $\mathrm{CTL}_{\mathrm{mon}}$-MC($T$) *is* P*-hard if* T *contains at least one of the operators* EG, EU, ER, AF, AU, *or* AR.

Proof. We modify the proof of Proposition 3.20 to work with EG only. The remaining fragments follow from the closure of P under complementation and $\mathrm{F}\chi \equiv \neg\mathrm{G}\neg\chi \equiv [\top\mathrm{U}\chi]$, $[\chi\mathrm{U}\pi] \equiv \neg[\neg\chi\mathrm{R}\neg\pi]$.

Let the Turing machine M, the word x, the polynomials p, q, and $\mathfrak{K}$ be as above. Further assume w.l.o.g. that M branches only binary in each step. In the following we will construct a Kripke structure $\mathfrak{K}'$ which is visualized in Figure 3.8. Denote by $W_\exists$ (resp. $W_\forall$) the set of states corresponding to existential (resp. universal) configurations. Let the Kripke structure $\mathfrak{K}'$ be defined as $\mathfrak{K}' \underset{\text{def}}{=} (W', R, \eta)$ consisting of $q(n)+1$ layers and a 'trap' as follows: let $W' \underset{\text{def}}{=} W \times \{1,\ldots,q(n)+1\} \cup \{z\}$. The transition relation $R \subseteq W' \times W'$ is defined as

$$
\begin{aligned}
R \underset{\text{def}}{=} & \left\{ \left((c_i^j,\ell),(c_k^{j+1},\ell)\right) \;\middle|\; \begin{array}{l} c_i^j \in W_\exists, M \text{ reaches } c_k \text{ from } c_i \text{ in one step,} \\ 1 \le \ell \le q(n)+1, 0 \le j < p(n) \end{array} \right\} \\
\cup & \left\{ \begin{array}{l} \left((c_i^j,\ell),(c_k^{j+1},i)\right), \\ \left((c_k^{j+1},i),(c_{k'}^{j+1},q(n)+1)\right), \\ \left((c_{k'}^{j+1},q(n)+1),z\right) \end{array} \;\middle|\; \begin{array}{l} c_i^j \in W_\forall, M \text{ reaches } c_k \text{ and } c_{k'} \text{ from } c_i \text{ in one step,} \\ c_k \le c_{k'}, 0 \le j < p(n), 1 \le \ell \le q(n)+1 \end{array} \right\} \\
\cup & \left\{ \left((c_i^{p(n)},\ell),(c_i^{p(n)},\ell)\right) \mid 1 \le i \le q(n), 1 \le \ell \le q(n)+1 \right\} \\
\cup & \left\{ (z,z) \right\}
\end{aligned}
$$

That is, the arcs leaving an existential configuration c_i lead to the successor configurations of c_i inside each layer; while any universal configuration c_i has exactly one outgoing arc pointing to its (lexicographically) first successor configuration in the layer i, from where another arc leads to the second successor of c_i in layer $q(n)+1$, which in turn has an outgoing arc to the state z. Observe that the layers are used to ensure the uniqueness of the successors of universal configurations which is essential for the function $\psi_i(x)$ constructed below. The labelling function η is defined as $\eta(z) \underset{\text{def}}{=} \{z\}$, $\eta((c_i^j,\ell)) \underset{\text{def}}{=} \{\ell, d_j, t\}$ if c_i is an

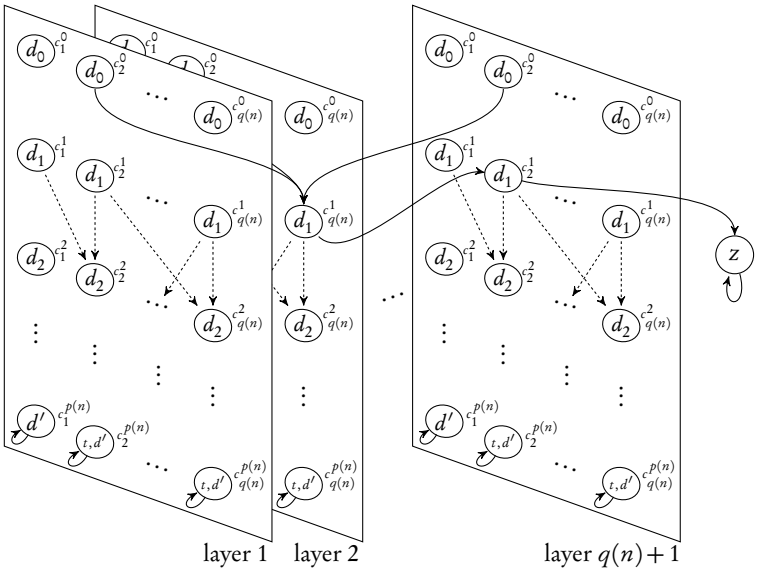


Figure 3.8: The Kripke structure $\mathfrak{K}'$; dashed (resp. solid) arrows correspond to transitions leaving existential (resp. universal) configurations. d' abbreviates $d_{p(n)}$.

accepting configuration, and otherwise $\eta((c_i^j,\ell)) \underset{\text{def}}{=} \{\ell, d_j\}$ for $(1 \le \ell \le q(n)+1)$. Define

$$\psi_i(x) \underset{\text{def}}{=} \begin{cases} \mathrm{EG}(d_{i-1} \vee (d_i \wedge x) \vee z), \text{if } M\text{'s configurations in step } i \text{ are universal,} \\ \mathrm{EG}(D_i \vee x), \text{if } M\text{'s configurations in step } i \text{ are existential,} \end{cases}$$

and $D_i = \bigvee_{i \neq j \in \{0,\ldots,p(n)\}} d_j$. Through the construction of the Kripke structure observe that—for the universal case—the EG-path induced by the corresponding $\psi_i(x)$ visits each of the series-connected successor states and *must* finish in the trap. As in each successor state both d_{i-1} and z are not satisfied the subformula $d_i \wedge x$ must be satisfied which enforces two new EG-paths defined through the next ψ_{i+1}. By this one can verify that it holds that $\mathfrak{K}, w \models \mathrm{AX}(d_i \wedge x)$ if and only if $\mathfrak{K}', (w,\ell) \models \mathrm{EG}\big(d_{i-1} \vee (d_i \wedge x) \vee z)\big)$, for all $w \in W_\forall$, $1 \le \ell \le q(n)+1$ and $1 \le i \le p(n)$. From this the following equivalence follows

$$M \text{ accepts } x \quad \text{iff} \quad \mathfrak{K}', (c_1^0, 1) \models \psi_1\Big(\psi_2\big(\cdots \psi_{p(n)}(t)\big)\cdots\Big).$$

As the size of the set of states in $\mathfrak{K}$ has only grown by factor $q(n)+1$ and R can be constructed from all triples of states in W', $\mathfrak{K}'$ remains AC^0 constructible. Concluding $A \le_{\mathbf{cd}} \mathrm{CTL}_{\mathrm{mon}}\text{-MC}(\{\mathrm{EG}\})$ for all $A \in \mathsf{P}$. □

By Lemma 3.16, $\mathrm{CTL}_{\mathrm{mon}}\text{-MC}(T) \le_{\mathbf{cd}} \mathrm{CTL}_{\mathrm{pos}}\text{-MC}(T)$ and hence the above results directly translate to model checking for $\mathrm{CTL}_{\mathrm{pos}}$: for any set T of temporal operators, $\mathrm{CTL}_{\mathrm{pos}}\text{-MC}(T)$ is P-hard if $T \not\subseteq \{\mathrm{EX}, \mathrm{EF}\}$ and $T \not\subseteq \{\mathrm{AX}, \mathrm{AG}\}$. These results cannot be improved w.r.t. T, as for $T \subseteq \{\mathrm{EX}, \mathrm{EF}\}$ and $T \subseteq \{\mathrm{AX}, \mathrm{AG}\}$ we obtain a LOGCFL upper bound for model checking from Proposition 3.19. In the following proposition we prove the matching LOGCFL lower bound.

Proposition 3.23.
Let T be a nonempty set of CTL*-operators. Then the monotone model checking problem* $\mathrm{CTL}_{\mathrm{mon}}\text{-MC}(T)$ *is* LOGCFL*-hard.*

Proof. As explained in Chapter 2, LOGCFL can be characterized as the set of languages recognizable by logtime-uniform SAC^1 circuits. For every single CTL-operator O, we will show that $\mathrm{CTL}_{\mathrm{mon}}\text{-MC}(T)$ is LOGCFL-hard for all $T \supseteq \{O\}$ by giving a generic $\le_{\mathbf{cd}}$-reduction f from the word problem for SAC^1 circuits to $\mathrm{CTL}_{\mathrm{mon}}\text{-MC}(T)$.

We start by considering the case $\mathrm{EX} \in T$. Let C be a logtime-uniform SAC^1 circuit of depth ℓ with n inputs and let $x = x_1 \ldots x_n \in \{0,1\}^n$. Assume w.l.o.g. that C is connected, layered into alternating layers of $\wedge$-gates and $\vee$-gates, and that the output gate of C is an $\vee$-gate. We number the layers bottom-up, that is, the layer containing (only) the output gate has level 0, whereas the input-gates and negations of the input-gates are situated in layer ℓ. Denote the graph of C by $G = (V, E)$, where $V \underset{\text{def}}{=} V_{\text{in}} \uplus V_\wedge \uplus V_\vee$ is partitioned into the sets corresponding to the (possibly negated) input-gates, the $\wedge$-gates, and the $\vee$-gates, respectively. G is acyclic and directed with paths leading from the input to the output gates. From (V, E) we construct a Kripke structure that allows to distinguish the

two predecessors of an $\wedge$-gate from each other. This will be required to model proof trees using $\mathrm{CTL}_{\mathrm{mon}}(\{\mathrm{EX}\})$-formulae.

For $i \in \{1,2\}$, let $V^i_{\mathrm{in}} \underset{\mathrm{def}}{=} \{v^i \mid v \in V_{\mathrm{in}}\}$, $V^i_{\vee} \underset{\mathrm{def}}{=} \{v^i \mid v \in V_{\vee}\}$ and define $V^i_{\mathrm{in},\vee} \underset{\mathrm{def}}{=} V^i_{\mathrm{in}} \cup V^i_{\vee}$. Further define

$$E' \underset{\mathrm{def}}{=} \big\{(v,u^i) \in V_{\wedge} \times V^i_{\mathrm{in},\vee} \mid (u,v) \in E \text{ and } u \text{ is the } i\text{th predecessor of } v\big\}$$
$$\cup \big\{(v,v) \mid v \in V^1_{\mathrm{in}} \cup V^2_{\mathrm{in}}\big\} \cup \bigcup_{i\in\{1,2\}} \big\{(v^i,u) \in V^i_{\mathrm{in},\vee} \times V_{\wedge} \mid (u,v) \in E\big\},$$

where the ordering of the predecessors is implicitly given in the encoding of C. We now define a Kripke structure $\mathfrak{K} \underset{\mathrm{def}}{=} (V',E',\eta)$ with states $V' \underset{\mathrm{def}}{=} V^1_{\mathrm{in},\vee} \cup V^2_{\mathrm{in},\vee} \cup V_{\wedge}$, transition relation E', and labelling function $\eta : V' \to \mathfrak{P}(\{1,2,t\})$ defined as

$$\eta(v) \underset{\mathrm{def}}{=} \begin{cases} \{i,t\}, & \text{if } (v = v_{\mathrm{in}_j} \in V^i_{\mathrm{in}} \text{ and } x_j = 1) \text{ or } (v = \overline{v}_{\mathrm{in}_j} \in V^i_{\mathrm{in}} \text{ and } x_j = 0), \\ \{i\}, & \text{if } (v = v_{\mathrm{in}_j} \in V^i_{\mathrm{in}} \text{ and } x_j = 0) \text{ or } (v = \overline{v}_{\mathrm{in}_j} \in V^i_{\mathrm{in}} \text{ and } x_j = 1) \\ & \quad \text{or } v \in V^i_{\vee}, \\ \emptyset, & \text{otherwise,} \end{cases}$$

where $i = 1,2$, $j = 1,\ldots,n$ and $v_{\mathrm{in}_1},\ldots,v_{\mathrm{in}_n}, \overline{v}_{\mathrm{in}_1},\ldots,\overline{v}_{\mathrm{in}_n}$ enumerate the input gates and their negations. The formula φ that is to be evaluated on $\mathfrak{K}$ consists of atomic propositions 1, 2 and t, Boolean connectives $\wedge$ and $\vee$, and the CTL-operator EX. To construct φ we recursively define formulae $(\varphi_i)_{0\le i\le \ell}$ by

$$\varphi_i \underset{\mathrm{def}}{=} \begin{cases} t, & \text{if } i = \ell, \\ \mathrm{EX}\varphi_{i+1}, & \text{if } i \text{ is even } (\vee\text{-layers}), \\ \bigwedge_{j=1,2} \mathrm{EX}(j \wedge \varphi_{i+1}), & \text{if } i \text{ is odd } (\wedge\text{-layers}). \end{cases}$$

We define the reduction function f as the mapping $(C,x) \mapsto (\mathfrak{K}, v_0, \varphi_0)$, where v_0 is the node corresponding to the output gate of C. It is to be accentuated that the size of φ is polynomial, for the depth of C is logarithmic only. Clearly, each minimal accepting subtree (cf. [Ruz80] or [Vol99, Definition 4.15]) of C on input x translates into a substructure $\mathfrak{K}'$ of $\mathfrak{K}$ such that $\mathfrak{K}', v_0 \models \varphi_0$, where

(a) $\mathfrak{K}'$ includes v_0,

(b) $\mathfrak{K}'$ includes one successor for every node corresponding to an $\vee$-gate, and

(c) $\mathfrak{K}'$ includes the two successors of every node corresponding to an $\wedge$-gate.

As $C(x) = 1$ iff there exists a minimal accepting subtree of C on x, the LOGCFL-hardness of $\mathrm{CTL}_{\mathrm{mon}}$-MC($T$) for $\mathrm{EX} \in T$ follows.

Next, consider $\mathrm{EF} \in T$. We have to extend our Kripke structure to contain information about the depth of the corresponding gate. We may assume w.l.o.g. that C is encoded

such that each gate contains an additional counter holding the distance to the output gate (which is equal to the number of the layer it is contained in, cf. [Vol99]). We extend η to encode this distance i, $1 \leq i \leq \ell$, into the "depth-propositions" d_i as in the proof of Proposition 3.20. Denote this modified Kripke structure by $\mathfrak{K}'$. Further, we define $(\varphi'_i)_{0 \leq i \leq \ell}$ as

$$\varphi'_i \underset{\text{def}}{=} \begin{cases} t, & \text{if } i = \ell, \\ \mathrm{EF}(d_{i+1} \wedge \varphi'_{i+1}), & \text{if } i \text{ is even}, \\ \bigwedge_{j=1,2} \mathrm{EF}(d_{i+1} \wedge j \wedge \varphi'_{i+1}), & \text{if } i \text{ is odd}. \end{cases}$$

Redefining the reduction f as $(C, x) \mapsto (\mathfrak{K}', v_0, \varphi'_0)$ hence yields the LOGCFL-hardness of $\mathrm{CTL}_{\mathrm{mon}}$-MC($T$) for $\mathrm{EF} \in T$.

Finally consider $\mathrm{AX} \in T$. Observe the following for the reduction in case 1 for monotone $\mathrm{CTL}_{\mathrm{mon}}(\{\mathrm{EX}\})$-formulae, where $f(C, x) = (\mathfrak{K}, v_0, \varphi)$ is the value computed by the reduction function. It holds that $C(x) = 1$ iff $\mathfrak{K}, v_0 \models \varphi$, and equivalently $C(x) = 0$ iff $\mathfrak{K}, v_0 \models \neg\varphi$. Let φ' be the formula obtained from $\neg\varphi$ by multiplying the negation into the formula. Then φ' is a $\mathrm{CTL}_{\mathrm{a.n.}}(\{\mathrm{AX}\})$-formula. Since LOGCFL is closed under complement, it follows that $\mathrm{CTL}_{\mathrm{a.n.}}$-MC($\{\mathrm{AX}\}$) is LOGCFL-hard. Using Lemma 3.16, we obtain that $\mathrm{CTL}_{\mathrm{mon}}$-MC($\{\mathrm{AX}\}$) is LOGCFL-hard, too. An analogous argument works for the case $\mathrm{AG} \in T$. The remaining fragments are even P-complete by Proposition 3.22. □

Using Lemma 3.16 we obtain LOGCFL-hardness of $\mathrm{CTL}_{\mathrm{pos}}$-MC($T$) for all nonempty sets T of CTL-operators.

In the absence of CTL-operators, the lower bound for the model checking problem again follows from the lower bound for evaluating monotone propositional formulae. This problem is known to be hard for NC^1 [Bus87, Sch10].

This concludes the proof of Theorem 3.17 and Theorem 3.18.

The Power of Negation

We will now show that model checking for the fragments $\mathrm{CTL}_{\mathrm{a.n.}}$ and $\mathrm{CTL}_{\mathrm{pos}}$ is computationally equivalent to model checking for $\mathrm{CTL}_{\mathrm{mon}}$, for any set T of CTL-operators. Since we consider $\leq_{\mathrm{cd}}$-reductions, this is not immediate.

From Lemma 3.16 it follows that the hardness results for $\mathrm{CTL}_{\mathrm{mon}}$-MC($T$) also hold for $\mathrm{CTL}_{\mathrm{a.n.}}$-MC($T$) and $\mathrm{CTL}_{\mathrm{pos}}$-MC($T$). Moreover, the lemma implies that upper bounds of $\mathrm{CTL}_{\mathrm{pos}}$-MC($T$) hold for the other two problems, i.e., the algorithms for $\mathrm{CTL}_{\mathrm{pos}}$-MC($T$) also work for $\mathrm{CTL}_{\mathrm{mon}}$-MC($T$) and $\mathrm{CTL}_{\mathrm{a.n.}}$-MC($T$). Both observations together yield the same completeness results for all CTL-fragments with restricted negations.

Theorem 3.24.
Let T be any set of CTL*-operators, and $\mathscr{C} \in \{\mathrm{CTL}_{\mathrm{mon}}, \mathrm{CTL}_{\mathrm{a.n.}}, \mathrm{CTL}_{\mathrm{pos}}\}$. Then $\mathscr{C}$-MC(T) is*

(1.) NC^1*-complete if T is empty,*

(2.) LOGCFL*-complete if $\emptyset \subsetneq T \subseteq \{\mathrm{EX}, \mathrm{EF}\}$ or $\emptyset \subsetneq T \subseteq \{\mathrm{AX}, \mathrm{AG}\}$,*

(3.) P*-complete otherwise.*

Further, the problems $\mathrm{CTL}_{\mathrm{mon}}$-MC($T$), $\mathrm{CTL}_{\mathrm{a.n.}}$-MC($T$), *and* $\mathrm{CTL}_{\mathrm{pos}}$-MC($T$) *are equivalent w.r.t.* $\leq_{\mathbf{cd}}$*-reductions.*

This equivalence extends Lemma 3.16. This equivalence is not straightforward by simply applying de Morgan's laws to transform one problem into another because this requires counting the number of negations on top of ∧- and ∨-connectives. This counting cannot be achieved by an AC^0-circuit and does not lead to the aspired reduction. The theorem obtains equivalence of the problems as a consequence of our generic hardness proofs in the Section *Lower Bounds* on page 54.

3.2.2 Model Checking Extensions of CTL

As for CTL, model checking for ECTL is known to be tractable [Sch02]. Moreover, our next result shows that even for all fragments, model checking for ECTL is not harder than for CTL.

Theorem 3.25.
Let T *be a set of* ECTL*-operators. Then*

$$\text{ECTL-MC}(T) \equiv_{\mathbf{cd}} \text{CTL-MC}(T') \text{ and } \mathrm{ECTL}_{\mathrm{pos}}\text{-MC}(T) \equiv_{\mathbf{cd}} \mathrm{CTL}_{\mathrm{pos}}\text{-MC}(T'),$$

where T' *is obtained from* T *by substituting* $\overset{\infty}{\mathrm{F}}$ *with* F *and* $\overset{\infty}{\mathrm{G}}$ *with* G.

Proof. For the upper bounds, notice that for the full fragment we have a membership result for P. Thus it holds that ECTL-MC(ALL ∪ $\{\overset{\infty}{\mathrm{EF}}, \overset{\infty}{\mathrm{AF}}\}$) ∈ P. It thus remains to show that $\mathrm{ECTL}_{\mathrm{pos}}$-MC($T$) ∈ LOGCFL for $T \subseteq \{\mathrm{EX}, \mathrm{EF}, \overset{\infty}{\mathrm{EF}}\}$ and $T \subseteq \{\mathrm{AX}, \mathrm{AG}, \overset{\infty}{\mathrm{AG}}\}$ First, consider the case that $T \subseteq \{\mathrm{EX}, \mathrm{EF}, \overset{\infty}{\mathrm{EF}}\}$. We modify Algorithm 3.3 to handle $\overset{\infty}{\mathrm{EF}}$ by extending the case distinction in lines 4–13 with the code fragment given in Algorithm 3.4. The algorithm for $T \subseteq \{\mathrm{AX}, \mathrm{AG}, \overset{\infty}{\mathrm{AG}}\}$ is analogous and membership in LOGCFL follows from its closure under complementation.

Algorithm 3.4: Case distinction for $\overset{\infty}{\mathrm{EF}}$

```
1 else if φ = EF^∞ α then
2 |   nondet. choose k ≤ |W| and a path (w_i)_{1≤i≤k} s.t. (w, w_1) ∈ R*, (w_k, w_1) ∈ R;
3 |   nondet. choose some 1 ≤ i ≤ k and push(S, (α, w_i));
```

For proving the lower bounds, we extend the proofs of Propositions 3.20, 3.22 and 3.23 to handle sets T involving also the operators $\overset{\infty}{\mathrm{AF}}$, $\overset{\infty}{\mathrm{AG}}$, $\overset{\infty}{\mathrm{EF}}$, and $\overset{\infty}{\mathrm{EG}}$. Therefore, we only need modify the accessibility relation R of respective Kripke structure $\mathfrak{K}$ to be reflexive. The hardness results follow by replacing F with $\overset{\infty}{\mathrm{F}}$ and G with $\overset{\infty}{\mathrm{G}}$ in the respective reductions.

First consider the case that T contains an existential and a universal operator, say $T = \{\mathrm{A}\overset{\infty}{\mathrm{F}}, \mathrm{E}\overset{\infty}{\mathrm{G}}\}$. Let M, x, and p be defined as in the proof of Proposition 3.20. We map (M, x) to $(\tilde{\mathfrak{K}}, c_1^0, \psi_1)$, where for $\tilde{\mathfrak{K}} = (W, \tilde{R}, \eta)$ the accessibility relation $\tilde{R}$ is the reflexive closure of R in the Kripke structure $\mathfrak{K}$ defined for the P-hardness of CTL-MC($\{\mathrm{AF}, \mathrm{EG}\}$), $c_1^0 \in W$, and $\psi \underset{\text{def}}{=} \psi_1\Big(\psi_2\big(\cdots \psi_{p(n)}(t)\big)\cdots\Big)$, where

$$\psi_i(x) \underset{\text{def}}{=} \begin{cases} \mathrm{A}\overset{\infty}{\mathrm{F}}(d_i \wedge x), & \text{if } M\text{'s configurations in step } i \text{ are universal, and} \\ \mathrm{E}\overset{\infty}{\mathrm{G}}(D_i \vee x), & \text{otherwise.} \end{cases}$$

In $\tilde{\mathfrak{K}}$ it now holds that $d_i \in \eta(w)$ and $(w, w') \in R$ together imply that either $w = w'$ or $d_i \notin \eta(w')$. Hence, for all $w \in W$ and $1 \le i \le p(|x|)$, $\tilde{\mathfrak{K}}, w \models \mathrm{A}\overset{\infty}{\mathrm{F}}(d_i \wedge x)$ iff $\mathfrak{K}, w \models \mathrm{AF}(d_i \wedge x)$, and $\tilde{\mathfrak{K}}, w \models \mathrm{E}\overset{\infty}{\mathrm{G}}(\bigvee_{i \neq j \in \{0,\ldots,p(n)\}} d_j \vee x)$ iff $\mathfrak{K}, w \models \mathrm{EG}(\bigvee_{i \neq j \in \{0,\ldots,p(n)\}} d_j \vee x)$. From this, correctness of the reduction follows. The P-hardness of CTL-MC(T) for the remaining fragments can be proven analogously.

As for $T \subseteq \{\mathrm{EX}, \mathrm{EF}, \mathrm{E}\overset{\infty}{\mathrm{F}}\}$, we will show that $\mathrm{ECTL}_{\mathrm{mon}}$-MC($T$) is LOGCFL-hard under $\leq_{\mathrm{cd}}$-reductions for $T = \{\mathrm{E}\overset{\infty}{\mathrm{F}}\}$. Let C, x, and ℓ be as in the proof of Proposition 3.23. We map the pair (C, x) to the triple $(\tilde{\mathfrak{K}}', v_0, \varphi_0)$, where $\tilde{\mathfrak{K}}' = (V', E', \eta)$ is the reflexive closure of the Kripke structure $\mathfrak{K}'$ defined for the LOGCFL-hardness of CTL-MC($\{\mathrm{EF}\}$), $v_0 \in V'$, and φ_0 is recursively defined via $(\varphi_i')_{0 \le i \le \ell}$ as

$$\varphi_i \underset{\text{def}}{=} \begin{cases} t, & \text{if } i = \ell, \\ \mathrm{E}\overset{\infty}{\mathrm{F}}(d_{i+1} \wedge \varphi_{i+1}), & \text{if } i \text{ is even,} \\ \bigwedge_{j=1,2} \mathrm{E}\overset{\infty}{\mathrm{F}}(d_{i+1} \wedge j \wedge \varphi_{i+1}), & \text{if } i \text{ is odd.} \end{cases}$$

Again, we conclude that in $\tilde{\mathfrak{K}}'$, $d_i \in \eta(v)$ and $(v, v') \in E'$ together imply that either $v = v'$ or $d_i \notin \eta(v')$. It hence follows $\tilde{\mathfrak{K}}', v \models \mathrm{E}\overset{\infty}{\mathrm{F}}(d_i \wedge \varphi_i)$ iff $\mathfrak{K}', v \models \mathrm{EF}(d_i \wedge \varphi_i)$, for all $v \in V'$ and $1 \le i \le \ell$. We conclude that $\mathrm{ECTL}_{\mathrm{mon}}$-MC($\{\mathrm{E}\overset{\infty}{\mathrm{F}}\}$) is LOGCFL-hard. The hardness for case $T = \{\mathrm{A}\overset{\infty}{\mathrm{G}}\}$ results from the complement argument. □

We will now consider CTL$^+$, the extension of CTL by Boolean combinations of path formulae.

In contrast to CTL, model checking for CTL$^+$ is not tractable, but complete for Δ_2^{P} w.r.t. $\leq_{\mathrm{cd}}$-reductions [LMS01]. Below we classify the complexity of model checking for both the full and the positive fragments of CTL$^+$.

Theorem 3.26.
Let T be a set of temporal operators containing at least one path quantifier. Then CTL$^+$-MC(T) *is*

(1.) NC^1*-complete if* $T \subseteq \{\mathrm{A}, \mathrm{E}\}$,

(2.) P*-complete if* $\{\mathrm{X}\} \subsetneq T \subseteq \{\mathrm{A}, \mathrm{E}, \mathrm{X}\}$, *and*

(3.) Δ_2^{P}*-complete otherwise.*

Proof. If $T \subseteq \{\mathrm{A}, \mathrm{E}\}$ then deciding CTL$^+$-MC(T) is equivalent to the problem of evaluating a propositional formula, which is known to be NC^1-complete [Bus87, Sch10].

If $\{\mathrm{X}\} \subsetneq T \subseteq \{\mathrm{A}, \mathrm{E}, \mathrm{X}\}$, then CTL$^+$-MC($T$) can be solved using a labelling algorithm: Let $\mathfrak{K} = (W, R, \eta)$ be a Kripke structure, and φ be a CTL$^+$($\{\mathrm{A}, \mathrm{E}, \mathrm{X}\}$)-formula. Assume w.l.o.g. that φ starts with an E and that it does not contain any A's (substitute $\mathrm{A}\phi$ with $\neg\mathrm{E}\neg\phi$). Compute all states $w \in W$ s.t. $\mathfrak{K}, w \models \psi$ for all subformulae $\mathrm{E}\psi$ of φ such that ψ is free of path quantifiers, and replace $\mathrm{E}\psi$ in φ with a new proposition p_ψ while extending the labelling function η such that $p_\psi \in \eta(w)$ iff $\mathfrak{K}, w \models \psi$. Repeat this step until φ is free of path quantifiers and denote the resulting (propositional) formula by φ'. To decide whether $\mathfrak{K}, w \models \varphi$ for some $w \in W$, it now suffices to check whether φ' is satisfied by the assignment implied by $\eta(w)$. As for all of the above subformulae $\mathrm{E}\psi$ of φ, $\psi \in$ CTL$^+$($\{\mathrm{X}\}$), it follows that $\mathfrak{K}, w \models \psi$ can be determined in polynomial time in the size of $\mathfrak{K}$ and ψ. Considering that the number of labelling steps is at most $O(|\varphi| \cdot |W|)$ it follows that CTL$^+$-MC(T) is in P. The P-hardness follows from CTL-MC($\{\mathrm{EX}\}$) $\leq_{\mathrm{cd}}$ CTL$^+$-MC($\{\mathrm{E}, \mathrm{X}\}$) resp. CTL-MC($\{\mathrm{AX}\}$) $\leq_{\mathrm{cd}}$ CTL$^+$-MC($\{\mathrm{A}, \mathrm{X}\}$).

For all other possible sets T, we have $T \cap \{\mathrm{E}, \mathrm{A}\} \neq \emptyset$ and $T \cap \{\mathrm{F}, \mathrm{G}, \mathrm{U}\} \neq \emptyset$. Consequently, each of the temporal operators A, E, F, and G can be expressed in CTL$^+$(T). The claim now follows from [LMS01] proving the respecting Δ_2^{P} lower bounds. □

For the positive fragments of CTL$^+$ we obtain a more complex classification which comprises six different completeness degrees:

Theorem 3.27.
Let T be a set of temporal operators containing at least one path quantifier. Then the problem CTL$^+_{\mathrm{pos}}$-MC(T) *is*

(1.) NC^1*-complete if* $T \subseteq \{\mathrm{A}, \mathrm{E}\}$,

(2.) LOGCFL*-complete if* $T = \{\mathrm{A}, \mathrm{X}\}$ *or* $T = \{\mathrm{E}, \mathrm{X}\}$,

(3.) P*-complete if* $T = \{\mathrm{A}, \mathrm{E}, \mathrm{X}\}$,

(4.) NP*-complete if* $\mathrm{E} \in T$, $\mathrm{A} \notin T$ *and T contains exactly one pure temporal operator aside from* X,

(5.) coNP*-complete if* $\mathrm{A} \in T$, $\mathrm{E} \notin T$ *and T contains exactly one pure temporal operator aside from* X, *and*

(6.) Δ_2^{P}*-complete otherwise.*

Proof. The first and third claim follow from Theorem 3.26 and from the monotone formula value problem being NC^1-complete [Sch10].

For the second claim, consider the case $T = \{\mathrm{E}, \mathrm{X}\}$. It is straightforward to adopt Algorithm 3.3 to guess a successor w' of the current state once for every path quantifier

E that has been read and decompose the formula w.r.t. w'. For $T = \{\mathrm{A}, \mathrm{X}\}$ analogous arguments hold. The lower bounds apply due to Proposition 3.23.

The fourth claim can be solved with a labelling algorithm analogously to the algorithm for CTL$^+$-MC($\{\mathrm{A}, \mathrm{E}, \mathrm{X}\}$). In this case, however, whole paths need to be guessed in the Kripke structures. Hence, we obtain a polynomial time algorithm deciding CTL$^+_{\text{pos}}$-MC(T) using an oracle $B \in \mathsf{NP}$ (resp. $B \in \mathsf{coNP}$) . Furthermore this algorithm is a monotone $\leq^{\mathrm{P}}_{\mathrm{T}}$-reduction from CTL$^+_{\text{pos}}$-MC($T$) to B, in the sense that for any deterministic oracle Turing machine M that executes the algorithm,

$$A \subseteq B \implies L(M, A) \subseteq L(M, B),$$

where $L(M, X)$ is the language recognized by M with oracle X. Both NP and coNP are closed under monotone $\leq^{\mathrm{P}}_{\mathrm{T}}$-reductions [Sel82]. Thus we can conclude that the desired membership result holds, i.e., CTL$^+_{\text{pos}}$-MC(T) $\in \mathsf{NP}$ (resp., CTL$^+_{\text{pos}}$-MC(T) $\in \mathsf{coNP}$).

As for the NP-hardness of CTL$^+_{\text{pos}}$-MC(T), note that the reduction from 3SAT to the linear temporal logic model checking problem LTL-MC($\{\mathrm{F}\}$), using the F-operator only, given by Sistla and Clarke in [SC85] is a reduction to CTL$^+_{\text{pos}}$-MC($\{\mathrm{E}, \mathrm{F}\}$) indeed. The NP-hardness of CTL$^+_{\text{pos}}$-MC($\{\mathrm{E}, \mathrm{G}\}$) is obtained by a similar reduction: let φ be a propositional formula in 3CNF, *i.e.*, $\varphi = \bigwedge_{i=1}^{n} C_i$ with $C_i = \ell_{i1} \vee \ell_{i2} \vee \ell_{i3}$ and $\ell_{ij} = x_k$ or $\ell_{ij} = \neg x_k$ for all $1 \leq i \leq n$, all $1 \leq j \leq 3$, and some $1 \leq k \leq m$. Recall that for a set A, $\bigvee A$ denotes the disjunction $\bigvee_{a \in A} a$. We map φ to the triple $(\mathfrak{K}, y_0, \psi)$, where $\mathfrak{K} = (W, R, \eta)$ is the Kripke structure defined as

$$\begin{aligned} W &\underset{\text{def}}{=} \{y_0\} \cup \{x_i, \overline{x}_i, y_i \mid 1 \leq i \leq m\}, \\ R &\underset{\text{def}}{=} \{(y_{i-1}, x_i), (x_i, y_i), (y_{i-1}, \overline{x}_i), (\overline{x}_i, y_i) \mid 1 \leq i \leq m\} \cup \{(y_m, y_m)\}, \\ \eta(w) &\underset{\text{def}}{=} \{w\} \text{ for all } w \in W. \end{aligned}$$

and $\psi \underset{\text{def}}{=} \mathrm{E} \bigwedge_{i=1}^{n} \bigvee_{j=1}^{3} \mathrm{G} \bigvee(\Phi \setminus \{\sim \ell_{ij}\})$ with $\Phi \underset{\text{def}}{=} \{y_0, y_i, x_i, \overline{x}_i \mid 1 \leq i \leq m\}$ and $\sim \ell_{ij}$ denoting the complementary literal of ℓ_{ij}. Note that the above reductions prove hardness for CTL$^+_{\text{mon}}$-MC(T) already. The coNP-hardness of the two cases CTL$^+_{\text{pos}}$-MC($\{\mathrm{A}, \mathrm{G}\}$) and CTL$^+_{\text{pos}}$-MC($\{\mathrm{A}, \mathrm{F}\}$) follows from the same reductions.

As for the the last claim, note that the Δ^{P}_2-hardness of CTL$^+$-MC($\{\mathrm{A}, \mathrm{E}, \mathrm{F}, \mathrm{G}\}$) carries over to CTL$^+_{\text{mon}}$-MC($\{\mathrm{A}, \mathrm{E}, \mathrm{F}, \mathrm{G}\}$), because any CTL$^+$($\{\mathrm{A}, \mathrm{E}, \mathrm{F}, \mathrm{G}\}$)-formula can be transformed into a CTL$^+_{\text{a.n.}}$($\{\mathrm{A}, \mathrm{E}, \mathrm{F}, \mathrm{G}\}$)-formula, in which all negated atoms $\neg p$ may be replaced by fresh propositions $\overline{p}$ that are mapped into all states of the Kripke structure whose label does not contain p. It thus remains to prove the Δ^{P}_2-hardness of CTL$^+_{\text{pos}}$-MC($\{\mathrm{A}, \mathrm{E}, \mathrm{F}\}$) and CTL$^+_{\text{pos}}$-MC($\{\mathrm{A}, \mathrm{E}, \mathrm{G}\}$). Consider CTL$^+_{\text{pos}}$-MC($\{\mathrm{A}, \mathrm{E}, \mathrm{G}\}$). Laroussinie et al. reduce from SNSAT (sequentially nested satisfiability). This is the problem to decide, given disjoint sets $Z_1, \ldots, Z_n$ of propositional variables from $\{z_1, \ldots, z_p\}$ and a list $\varphi_1(Z_1), \varphi_2(x_1, Z_2), \ldots, \varphi_n(x_1, \ldots, x_n, Z_n)$ of formulae in conjunctive normal form,

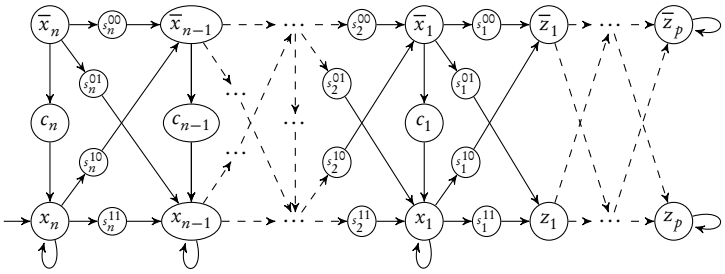


Figure 3.9: Extended version of the Kripke structure constructed in [LMS01, Figure 3].

whether x_n holds in the unique evaluation σ. The valuation function $\sigma : \mathbf{Vars}(\varphi) \to \{\top, \bot\}$ is defined by

$$\sigma(x_i) = \top \quad \text{iff} \quad \varphi_i(x_1, \ldots, x_{i-1}, Z_i) \text{ is satisfiable.} \tag{3.2}$$

An instance I of SNSAT is transformed to the Kripke structure $\mathfrak{K}$ depicted in Figure 3.9 and the formula ψ_{2n-1} that is recursively defined as

$$\psi_k \underset{\text{def}}{=} \mathrm{E}\Bigg[\underbrace{\mathrm{G}\Big(\bigvee_{i=1}^{n} \overline{x}_i \to \mathrm{E}\big(\neg \mathrm{F} \bigvee_{i=1}^{n} (s_i^{00} \vee s_i^{01} \vee s_i^{10} \vee s_i^{11}) \wedge \mathrm{F}\big(\bigvee_{i=1}^{n} x_i \not\to \psi_{k-1}\big)\big)\Big)}_{(A)}$$

$$\wedge \underbrace{\mathrm{G}\Big(\bigwedge_{i=1}^{n} \neg c_i\Big)}_{(B)} \wedge \underbrace{\bigwedge_{i=1}^{n} \Big(\mathrm{F} x_i \to \bigwedge_{j} \bigvee_{m} \mathrm{F} \ell_{i,j,m}\Big)}_{(C)}\Bigg],$$

for $1 \le k \le n$, $\psi_0 \underset{\text{def}}{=} \top$, and $\varphi_i = \bigwedge_j \bigvee_m \ell_{i,j,m}$, where the $\ell_{i,j,m}$'s are literals over $\{x_1, \ldots, x_n\} \cup Z_i$. Note that the structure $\mathfrak{K}$ from Figure 3.9 differs from the Kripke structure constructed in [LMS01] in that we introduce different labels c_i and s_i^j for $1 \le i \le n$ and $j \in \{00, 01, 10, 11\}$, as we need to distinguish the states later on. The intuitive interpretation of (B) is that the existentially quantified path does actually encode an assignment of $\{x_1, \ldots, x_n\}$ to $\{\bot, \top\}$, while (C) states that this assignment coincides with σ on all propositions that are set to $\top$. Lastly (A) expresses the recursion inherent in the definition of SNSAT. It holds that $I \in \text{SNSAT}$ iff $\mathfrak{K}, x_n \models \psi_{2n-1}$ (see [LMS01] for the correctness of this argument).

We modify the given reduction to hold without using F. At first note that ψ_{k-1} occurs negatively in ψ_k. We will therefore consider the formulae $\psi_{2n-1}, \psi_{2n-3}, \ldots, \psi_1$ and $\neg\psi_{2n-2}, \neg\psi_{2n-4}, \ldots, \neg\psi_2$ separately. In $\psi_{2n-1}, \psi_{2n-3}, \ldots, \psi_1$ replace

- (A) with $\mathrm{G}\Big(\bigvee_{i=1}^{n} \overline{x}_i \to \mathrm{E}\big(\mathrm{G} \bigwedge_{i=1}^{n} (\neg s_i^{00} \wedge \neg s_i^{01} \wedge \neg s_i^{10} \wedge \neg s_i^{11}) \wedge \mathrm{G}\big(\bigvee_{i=1}^{n} \overline{x}_i \vee \bigvee_{i=1}^{n} c_i \vee \neg\psi_{k-1}\big)\big)\Big)$,

- (C) with $\bigwedge_{i=1}^{n}\Big(\mathrm{G}\neg x_i \vee \bigwedge_{j}\bigvee_{m}\mathrm{G}\bigvee(\Phi\setminus\{\sim\ell_{i,j,m}\})\Big)$;

and in $\neg\psi_{2n-2},\neg\psi_{2n-4},\ldots,\neg\psi_2$ replace

- (A) with $\bigvee_{1\le i\le n}\mathrm{G}\Big(\bigvee(\Phi\setminus\{\overline{x}_i\})\vee\mathrm{A}\big(\mathrm{G}(\bigvee\Phi\setminus\{c_i\})\vee\mathrm{G}(c_i\vee\psi_{k-1})\big)\Big)$,

- (B) with $\bigvee_{i=2}^{n}\mathrm{G}\bigvee(\Phi\setminus\{s_i^{00},s_i^{01},s_{i-1}^{01},s_{i-1}^{11}\})$, and

- (C) with $\bigvee_{i=1}^{n}\Big(\mathrm{G}\bigvee(\Phi\setminus\{\overline{x}_i\})\wedge\bigvee_{j}\bigwedge_{m}\mathrm{G}\neg\ell_{i,j,m}\Big)$,

where $\Phi \underset{\text{def}}{=} \{x_i,\overline{x}_i,c_i,s_i^{00},s_i^{01},s_i^{10},s_i^{11} \mid 1\le i\le n\}\cup\{z_i,\overline{z}_i \mid 1\le i\le p\}$ is the set of all propositions used in $\mathfrak{K}$. Denote the resulting formulae by ψ'_k, $k\ge 0$. In ψ'_k, all negations are atomic and only the temporal operators E, A and G are used.

To verify that $\mathfrak{K},x_k\models\psi_k$ iff $\mathfrak{K},x_k\models\psi'_k$ holds for all $0\le k<2n$, consider ψ_k with k odd first. Suppose $\mathfrak{K},x_k\models\psi_k$. Then, by (A), there exists a path π in $\mathfrak{K}$ such that whenever some $\overline{x}_i$ is labelled in the current state π_p, then there exists a path π' starting in π_p that never visits any state labelled with s_i^j, $1\le i\le n$, $j\in\{00,01,10,11\}$, and eventually falsifies ψ_{k-1} because it reaches a state where neither $\overline{x}_i$ nor c_i holds for all $1\le i\le n$. Hence, by construction of $\mathfrak{K}$, π' has to visit the states labelled with c_i and x_i for i such that $\overline{x}_i\in\eta(\pi_p)$. This is equivalent to the existence of a path π' starting in π_p which never visits any state labelled with s_i^j, $1\le i\le n$, $j\in\{00,01,10,11\}$, and that falsifies ψ_{k-1} if the current state is not labelled with c_i or $\overline{x}_i$ for all $1\le i\le n$. Hence the substitution performed on (A) does not alter the set of states in $\mathfrak{K}$ on which the formula is satisfied.

The formula (C), on the other hand, states that whenever the path π quantified by the outmost E in ψ_k visits the state labelled x_i, then for every clause j in the ith formula φ_i of given SNSAT instance at least one literal $\ell_{i,j,m}$ occurs in the labels on π (*i.e.*, φ_i is satisfied by the assignment induced by π). The path π is guaranteed to visit either a state labelled x_i or a state labelled $\overline{x}_i$ but not both, by virtue of the subformula (B). Therefore, the eventual satisfaction of x_i is equivalent to globally satisfying $\neg x_i$, whereas the satisfaction of φ_i can be asserted by requiring that for any clause some literal is globally absent from the labels on π. Thus the substitution performed on (C) does not alter the set of states on which the formula is satisfied either. Concluding, $\mathfrak{K},x_k\models\psi_k$ iff $\mathfrak{K},x_k\models\psi'_k$ for all odd $0\le k<2n$.

Now, if k is even, then

$$\neg\psi_k \equiv \mathrm{A}\Big[\underbrace{\mathrm{F}\Big(\bigvee_{i=1}^{n}\overline{x}_i \wedge \mathrm{A}\big(\mathrm{F}\bigvee_{i=1}^{n}(s_i^{00}\vee s_i^{01}\vee s_i^{10}\vee s_i^{11})\vee \mathrm{G}(\bigvee_{i=1}^{n}x_i \to \psi_{k-1})\big)\Big)}_{(A)}$$
$$\vee \underbrace{\mathrm{F}\Big(\bigvee_{i=1}^{n}c_i\Big)}_{(B)} \vee \underbrace{\bigvee_{i=1}^{n}\Big(\mathrm{F}x_i \wedge \bigvee_{j}\bigwedge_{m}\mathrm{G}\neg\ell_{i,j,m}\Big)}_{(C)}\Big].$$

Here, (A) asserts that on all paths π there is a state π_p such that $\overline{x}_i \in \eta(\pi_p)$ for some $1 \le i \le n$ and all paths π' starting in π_p eventually visit a state labelled with s_i^j, $1 \le i \le n$, $j \in \{00,01,10,11\}$, or satisfy ψ_{k-1} whenever $x_i \in \eta(\pi_p)$ for some $1 \le i \le n$. By construction of $\mathfrak{K}$, this is equivalent to stating that all paths π' either pass the state labelled c_i and globally satisfy $c_i \vee \psi_{k-1}$ or do not pass the state labelled c_i. As for the states in $\mathfrak{K}$ the formula $\mathrm{F}(\bigvee_{i=1}^{n}\overline{x}_i \wedge \chi) \equiv \bigvee_{i=1}^{n}\mathrm{F}(\overline{x}_i \wedge \chi)$ is satisfied iff $\bigvee_{i=1}^{n}\mathrm{G}(\bigvee(\Phi \setminus \{\overline{x}_i\}) \vee \chi)$ is satisfied, the set of states in $\mathfrak{K}$ on which the ψ_k is satisfied remains unaltered when substituting (A) with $\bigvee_{1\le i\le n}\mathrm{G}(\bigvee(\Phi \setminus \{\overline{x}_i\}) \vee \mathrm{A}(\mathrm{G}(\bigvee\Phi \setminus \{c_i\}) \vee \mathrm{G}(c_i \vee \psi_{k-1}))$.

Similarly, the set of states in $\mathfrak{K}$ on which the ψ_k is satisfied remains unaltered when substituting (B) with $\bigvee_{i=2}^{n}\mathrm{G}\bigvee(\Phi \setminus \{s_i^{00}, s_i^{01}, s_{i-1}^{01}, s_{i-1}^{11}\})$, as any path in $\mathfrak{K}$ that visits a state labelled with some c_i cannot pass via states labelled with s_i^{00}, s_i^{01}, s_{i-1}^{01}, or s_{i-1}^{11}.

Finally, the equivalence of ψ_k with $\bigvee_{i=1}^{n}(\mathrm{G}\bigvee(\Phi \setminus \{\overline{x}_i\}) \wedge \bigvee_j \bigwedge_m \mathrm{G}\neg\ell_{i,j,m})$ follows from arguments similar to those for the (C) part in the case that k is odd. We conclude that $\mathfrak{K}, x_k \models \psi_k$ iff $\mathfrak{K}, x_k \models \psi'_k$ for all $0 \le k < 2n$. Hence, $\mathrm{CTL}^+_{\mathrm{pos}}$-MC$(\{\mathrm{A},\mathrm{E},\mathrm{G}\})$ is Δ_2^{P}-hard.

For $T = \{\mathrm{A},\mathrm{E},\mathrm{F}\}$ similar modifications show that $\mathrm{CTL}^+_{\mathrm{pos}}$-MC$(T)$ is Δ_2^{P}-hard, too. This concludes the proof of Theorem 3.27. □

Lastly consider ECTL$^+$, the combination of ECTL and CTL$^+$. One can adapt the above hardness and membership proofs to hold for $\overset{\infty}{\mathrm{F}}$ and $\overset{\infty}{\mathrm{G}}$ instead of F and G: For example, to establish the Δ_2^{P}-hardness of $\mathrm{ECTL}^+_{\mathrm{pos}}$-MC$(T)$ in case $T = \{\mathrm{A},\mathrm{E},\overset{\infty}{\mathrm{G}}\}$ we modify $\mathfrak{K}$ such that the states labelled x_n and $\overline{x}_n$ are reachable from z_p and $\overline{z}_p$ and assert that (a) the path quantified by the outmost path quantifier in ψ_k, $1 \le i < 2n$, additionally satisfies $\bigwedge_{i=1}^{n}(\overset{\infty}{\mathrm{G}}\neg x_i \vee \overset{\infty}{\mathrm{G}}\neg\overline{x}_i)$ and (b) whenever $\overline{x}_i$ is labelled, then there exists a path that all but a finite number of times satisfies x_i. The changes if $\overset{\infty}{\mathrm{F}}$ is available instead of $\overset{\infty}{\mathrm{G}}$ follow by the duality principle of these operators. For its model checking problem we hence obtain:

Corollary 3.28.
Let T be a set of temporal operators containing at least one path quantifier and let T' by obtained from T by substituting $\overset{\infty}{\mathrm{F}}$ with F *and $\overset{\infty}{\mathrm{G}}$ with* G. *Then* ECTL$^+$-MC$(T) \equiv_{\mathbf{cd}}$ CTL$^+$-MC(T') *and* $\mathrm{ECTL}^+_{\mathrm{pos}}$-MC$(T) \equiv_{\mathbf{cd}} \mathrm{CTL}^+_{\mathrm{pos}}$-MC$(T)$.

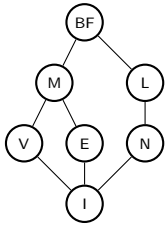


Figure 3.10: Post's lattice restricted to clones with both constants.

3.2.3 Model Checking CTL*

In 1986, CTL*-MC has been proven to be PSPACE-complete by Clarke, Emerson and Sistla (see Theorem 2.12). For the model checking problem put into context of Post's lattice it is obvious that one always has access to both constants by strictly encoding them with fresh propositions into the Kripke structure and formula. This leaves us with the clones I, N, E, V, M, L, and BF which are depicted in Figure 3.10.

Classifying all possible fragments emerging by operator, path quantifier, and Boolean function restrictions requires to study $2^6 \cdot 7 = 448$ different fragments or at least 384 different ones by using some duality principle which allows us to classify either the clone V or the clone E. Adding the dual operator R of U to the classification leads to 896, resp., 768 fragments. Thus, as a first approach to a full study we will state a classification of all fragments CTL*-MC(T,B) for all sets of temporal operators and path quantifiers for which $|T| \leq 2$ and $\mathrm{R} \notin T$ holds. There we will categorize the fragments into tractable and intractable cases only (therefore some cases will lack completeness results).

For the following theorem let FMC(B) denote the (propositional) formula model checking problem as defined in [Sch10] as

Problem (FMC(B))
Input: a propositional formula $\varphi \in \mathrm{PL}(B)$, and an assignment θ: **Vars**$(\varphi) \to \{\top, \bot\}$.
Question: does it hold that $\theta \models \varphi$?

Table 3.1 depicts the cases from Theorem 3.29 and shows in which part the complexity of which fragment is proven.

Theorem 3.29.
Let B be a finite set of Boolean functions and T be a set of path quantifiers and/or pure temporal operators such that $|T| \leq 2$. Then CTL*-MC(T,B) *is*

(1.) NP*-complete if* $[B] = \mathsf{M}$ *and* $T = \{\mathrm{X}\}, \{\mathrm{F}\}, \{\mathrm{E},\mathrm{X}\}, \{\mathrm{E},\mathrm{F}\}$,

(2.) coNP*-complete if* $[B] = \mathsf{M}$ *and* $T = \{\mathrm{A},\mathrm{X}\}, \{\mathrm{A},\mathrm{G}\}$,

(3.) NP*-hard if* $\mathsf{E} \subseteq [B]$ *and* $\{\mathrm{F}\} \subseteq T \subseteq \{\mathrm{E},\mathrm{F}\}$, *or*
if $\mathsf{V} \subseteq [B]$ *and* $\{\mathrm{G},\mathrm{X}\} = T$, *or*
if $\mathrm{U} \in T$,

CTL*-MC(T,B)	I	E	V	M	N	L	BF
E/A	$\equiv_{cd}$ FMC$^{(9.)}$						
G/U	$\equiv_{cd}$ LTL-MC$^{(8.)}$						
X/F	$\equiv_{cd}$ LTL-MC$^{(8.)}$			**NP**$^{(1.)}$	$\equiv_{cd}$ LTL-MC$^{(8.)}$		
E,A	$\equiv_{cd}$ FMC$^{(9.)}$						
E,G	**NL**$^{(7.)}$		**P**$^{(5.)}$	NP$^{(3.)}$	open		
E,X	**NL**$^{(7.)}$	**LCFL**$^{(6.)}$	**NL**$^{(7.)}$	**NP**$^{(1.)}$	open		
E,F	**NL**$^{(7.)}$	NP$^{(3.)}$	**NL**$^{(7.)}$	**NP**$^{(1.)}$	open		
A,X	**NL**$^{(7.)}$		**LCFL**$^{(6.)}$	**coNP**$^{(2.)}$	open		
A,F	**NL**$^{(7.)}$	**P**$^{(5.)}$	**NL**$^{(7.)}$	coNP$^{(4.)}$	open		
A,G	**NL**$^{(7.)}$		coNP$^{(4.)}$	**coNP**$^{(2.)}$	open		
G,X/G,F/X,F	$\equiv_{cd}$ LTL-MC$^{(8.)}$						
U,$\star$	all intractable (8.)						

Table 3.1: This table maps which fragment is proven in which case of Theorem 3.29. All open cases encompass sets T with $|T| > 2$ due to the availability of negation. Bold-face type fonts denote completeness results whereas all others entries denote hardness results.

(4.) coNP-*hard if* $\mathsf{V} \subseteq [B]$ *and* $T = \{\mathrm{A},\mathrm{G}\}$, *or if* $\mathsf{M} \subseteq [B]$ *and* $T = \{\mathrm{A},\mathrm{F}\}$,

(5.) P-*complete if* $[B] = \mathsf{V}$ *and* $T = \{\mathrm{E},\mathrm{G}\}$, *or* $[B] = \mathsf{E}$ *and* $T = \{\mathrm{A},\mathrm{F}\}$,

(6.) LOGCFL-*complete if* $[B] = \mathsf{E}$ *and* $T = \{\mathrm{E},\mathrm{X}\}$, *or* $[B] = \mathsf{V}$ *and* $T = \{\mathrm{A},\mathrm{X}\}$,

(7.) NLOGSPACE-*complete if* $\mathsf{I} \subseteq [B] \subseteq \mathsf{E}$ *and* $T = \{\mathrm{E},\mathrm{G}\},\{\mathrm{A},\mathrm{G}\},\{\mathrm{A},\mathrm{X}\}$, *or if* $\mathsf{I} \subseteq [B] \subseteq \mathsf{V}$ *and* $T = \{\mathrm{E},\mathrm{X}\},\{\mathrm{E},\mathrm{F}\},\{\mathrm{A},\mathrm{F}\}$

(8.) *equivalent to* LTL-MC(T,B) *if* $T \subseteq \{\mathrm{X},\mathrm{F},\mathrm{G},\mathrm{U}\}$,

(9.) *equivalent to* FMC(B) *if* $T \subseteq \{\mathrm{A},\mathrm{E}\}$.

Proof. Proving (1.) only involves showing a membership result for NP as the lower bound applies due to LTL-MC. Thus for the upper bound the algorithm simply guesses paths of length at most n in the given structure for the $\{\mathrm{E},\mathrm{X}\}$-case whenever a new E-operator is reached. Also the algorithm must take care of the scope of the path quantifier and needs to test depending on the path on which state the subformulae must be satisfied. Again as described in Section 3.1.2 the test until linear depth reached suffices in this case. For the F-case each occurring F-operator is separately substituted by X^k for nondeterministic guessed values $0 \leq k \leq n$ where n is the size of the input.

(2.) follows by complementation and negation normal form from (1.). We want to accentuate that this is not a contradiction to the LTL-results, e.g., for the case NP-hard case LTL-MC($\{X\}, M$). LTL-formulae are defined as path formulae and ask for the existence of a path in some Kripke structure. Thus the result is bound to this existential quantified problem. If we now allow the existence of the universal A operator, then we cannot carry over the results from LTL any longer.
Case (3.) results from LTL-MC(T,B) for the mentioned sets T and clones $[B]$ where the NP lower bounds were proven in [BMS+11].

The coNP-hardness of CTL*-MC(T,B) for $T = \{A,G\}$ and $[B] = V$ in (4.) is entailed by CTL*-MC(T',B') being NP-hard for $T' = \{E,F\}$ and $[B'] = E$. The other case follows by CTL*-MC$(\{E,G\},B)$ and $M = [B]$.

Turning now to the tractable fragments, we start with case (5.). The hardness carries over from the proof for $\mathrm{CTL}_{\mathrm{mon}}$-MC$(\{EG\})$ in [BMS+11] which uses only V as connective. For the membership in P observe that the following simplifications of formulae with E, G and V are possible:

$$GG\varphi \equiv G\varphi, \quad E(\psi_{prop} \vee \varphi) \equiv \psi_{prop} \vee E(\varphi), \quad E(\varphi \vee \psi) \equiv E\varphi \vee E\psi,$$

where $\varphi, \psi \in \mathrm{CTL}^*(T,B)$ and ψ_{prop} is a propositional formula. This leads to formulae where only EG-operators are present besides V, which is indeed model checking for CTL and this is in P. The opposite case $\{A,F\}$ and E is due to complementation.

Case (6.) follows from the following observation. The LOGCFL-hardness for $\{E,X\}$ and E follows from the proof for $\mathrm{CTL}_{\mathrm{mon}}$-MC$(\{EX\})$ which uses only the $\wedge$-operator as Boolean connective in [BMS+11]. The LOGCFL-algorithm is shown in Algorithm 3.5. The main idea of the algorithm is to use the stack to memorize the last state whenever a new path is introduced by an E-operator. The algorithm saves the recent state on the stack and then proceeds with the new subformula until this is finished. The overall relevant states is bounded by the nesting depth of temporal operators $\#_T(\varphi)$ and therefore involves only polynomial stack size and overall runtime. The other remaining case follows by LOGCFL being closed under complementation.

The main idea for case (7.) is a normal form which will be described now. Therefore we enumerate all required upper bounds:

- $\{E,G\}, E$: Using the equivalences

$$\begin{aligned} GG\varphi &\equiv G\varphi, & E(G\varphi \wedge G\psi) &\equiv EG(\varphi \wedge \psi), \\ E(\psi_{prop} \wedge \varphi) &\equiv \psi_{prop} \wedge E\varphi, & EG(\varphi \wedge EG\psi) &\equiv EG(\varphi \wedge \psi), \end{aligned}$$

  where $\varphi, \psi \in \mathrm{CTL}^*(T,B)$ and ψ_{prop} is a propositional formula, we can simplify the given formula to the form $EG(\varphi) \wedge \varphi_{prop}$ which can be then verified by the NLOGSPACE-algorithm for LTL-MC$(\{G\},B)$. The equivalences from above can be computed in logarithmic space, see [BMS+11].

- $\{A,G\}, E$: follows by closure under complement of NLOGSPACE and the case $\{E,F\}, V$.

Algorithm 3.5: LOGCFL-machine deciding CTL*-MC($\{\mathrm{E},\mathrm{X}\},\mathsf{E}$)

Input : Kripke structure $\mathfrak{K}=(W,R,\eta)$, formula $\varphi\in\mathrm{CTL}^*(\{\mathrm{E},\mathrm{X}\},\mathsf{E})$, state $s_0\in W$.
Output : true if and only if $\mathfrak{K},s_0\models\varphi$

```
1  stack S ← ∅; push(S,(φ,s_0));
2  while S is not empty do
3  |  (φ,w) ← pop(S);
4  |  if φ is a propositional formula then
5  |  |  if φ evaluates to false in w under η then return false;
6  |  else if φ = α ∧ β then
7  |  |  push(S,(α,w)); push(S,(β,w));
8  |  else if φ = Eα then
9  |  |  S ← makePath(𝔎,S,(α,w));
10 return true;
```

makePath :
Input : a Kripke structure $\mathfrak{K}=(W,R,\eta)$, a stack S, a formula α and a state w
Output : a stack S which is increased by at most $\#_{\mathrm{X}}(\alpha)\cdot|\mathbf{SF}(\alpha)|$ tuples denoting the tests for each state in a combined path

```
11 w_last ← w; i ← 0;
12 while i ≤ #_X(α) do
13 |  if i > 0 then guess an R-successor w of w_last; w_last ← w;
14 |  foreach ψ being propositional or starting with an E in α that is preceded by exactly i-many X
   |  operators without Es in between do
15 |  |  push(S,(ψ,w_last));
16 |  i ← i + 1;
17 return S;
```

- $\{A,X\},E$: follows by closure under complement of NLOGSPACE and the case $\{E,X\},V$.
- $\{E,X\},V$: guess which subformula is satisfied on which path.
- $\{E,F\},V$: guess which proposition is reachable.
- $\{A,F\},V$: follows by closure under complement of NLOGSPACE and the case $\{E,G\},E$.

The NLOGSPACE-hardness follows from the respective LTL-MC-cases in [BMS+11].

Case (8.) holds by the definition of the logics and [Sch02, Remark 2.6] and [BMS+11].

For (9.) observe that the path quantifiers can be deleted and this substitution transforms the input formula into a propositional formula. The complexity depends only on $[B]$ and has been classified in [Sch10]. □

3.2.4 Conclusion

We have shown (Theorem 3.18) that model checking for $\mathrm{CTL}_{\mathrm{pos}}(T)$ is already P-complete for most fragments of CTL. Only for some weak fragments, model checking becomes easier: for nonempty sets $T \subseteq \{\mathrm{EX},\mathrm{EF}\}$ or $T \subseteq \{\mathrm{AX},\mathrm{AG}\}$, the problem $\mathrm{CTL}_{\mathrm{pos}}$-MC$(T)$ is LOGCFL-complete. In the case that no CTL-operators are used, NC^1-completeness of evaluating propositional formulae applies. As a direct consequence (Theorem 3.17), model checking for CTL(T) is P-complete for every nonempty T. This shows that for the majority of interesting fragments, model checking CTL(T) is inherently sequential and cannot be sped up using parallelism.

While all the results above can be transferred to ECTL (Theorem 3.25), CTL$^+$ and ECTL$^+$ exhibit different properties. For both logics, the general model checking problem was shown to be complete for Δ_2^{P} in [LMS01]. Here we proved that model checking fragments of CTL$^+(T)$ and ECTL$^+(T)$ for $T \subseteq \{\mathrm{A},\mathrm{E},\mathrm{X}\}$ remains tractable, while the existential and the universal fragments of $\mathrm{CTL}^+_{\mathrm{pos}}(T)$ and $\mathrm{ECTL}^+_{\mathrm{pos}}(T)$ containing temporal operators other than X are complete for NP and coNP, respectively.

Instead of restricting only the use of negation as done in this thesis, one might go one step further and restrict the allowed Boolean connectives in an arbitrary way. For example, restricting the Boolean connectives to only one of the functions AND or OR leads to many NLOGSPACE-complete fragments in the presence of certain sets of temporal operators. However a full classification is still open.

Regarding the classification of CTL*-MC for arbitrary sets of temporal operators and Boolean operators we restricted the study to sets T of temporal operators and/or path quantifiers whose size is bounded by two. There, we investigated at least five different complexity degrees ranging through the classes NLOGSPACE, P, LOGCFL, NP, coNP, and PSPACE. The latter class is hidden in the equivalence of CTL*-MC(T,B) to LTL-MC(T,B) for $T \subseteq \{\mathrm{X},\mathrm{F},\mathrm{G},\mathrm{U}\}$. For the tractable cases of CTL*-MC(T,B) we showed how the proof applies several techniques known from the classification of CTL-MC but also from

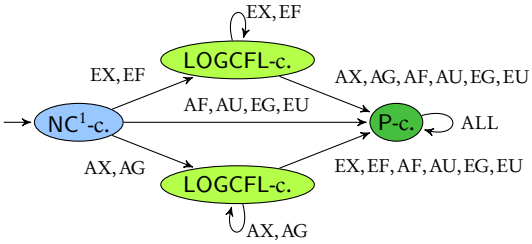


Figure 3.11: Complexity of $\mathrm{CTL}_{\mathrm{pos}}$-MC($T$) for all sets T of CTL-operators (depicted as a "finite automaton" where states indicate completeness results and arrows indicate an increase of the set of CTL-operators).

LTL-MC. Thus, completing the started study of classification of all operator and function fragments of CTL* would be of great interest.

All in all the CTL variants considered in this thesis but over *arbitrary* sets of Boolean operators would be one way to generalize our results. In the case of CTL+, where model checking is intractable [EL87, Sch02, LMS01], such a more fine-grained complexity analysis could help draw a tighter border between fragments with tractable and intractable model checking problems.

In Section 3.2 we made the assumption that the formula *and* the Kripke structure are part of the input and therefore can vary in size. The case where the complexity is measured in terms of the size of the formula (or the Kripke structure), and the other component is assumed to be fixed, is usually referred to as *specification complexity* (or *system complexity*). The approach which has been pursued in this thesis measures the joint complexity. In applications, where usually the structure is significantly bigger than the specification, an analysis of the system complexity becomes interesting. For system complexity, model checking for CTL and CTL* is already known to be NLOGSPACE-complete [BVW94, KVW00]. Nevertheless, the hope for a significant drop of system complexity is a justification of a systematic analysis of fragments of these logics.

Chapter 4

Description Logic

4.1 TBox and Ontology Satisfiability

In this section we will investigate the four satisfiability problems which are connected to terminology boxes and ontologies. As the results can be separated into two parts we will use $\star\mathrm{SAT}_{\mathscr{Q}}(B)$ to refer to any of the four problems $\mathrm{TSAT}_{\mathscr{Q}}(B), \mathrm{TCSAT}_{\mathscr{Q}}(B), \mathrm{OSAT}_{\mathscr{Q}}(B)$ and $\mathrm{OCSAT}_{\mathscr{Q}}(B)$ whereas the three problems which may refer to a single individual are denoted with $\star\mathrm{SAT}^{\mathrm{ind}}_{\mathscr{Q}}(B) = \star\mathrm{SAT}_{\mathscr{Q}} \setminus \{\mathrm{TSAT}_{\mathscr{Q}}(B)\}$ by abusing our notion for the problems $\star\mathrm{SAT}_{\mathscr{Q}}(B)$ without $\mathrm{TSAT}_{\mathscr{Q}}(B)$.

The first part of this section will be used to state some technical lemmata to help restrict the length of concepts in some of our reductions. The first lemma will show that for certain operator sets B, there are always short concepts representing the operators $\sqcap$, $\sqcup$, $\oplus$, or $\neg$, respectively. It directly follows from Lemma 2.4.

Lemma 4.1.
Let B be a finite set of Boolean operators.

(1.) If $\mathsf{V} \subseteq [B] \subseteq \mathsf{M}$ ($\mathsf{E} \subseteq [B] \subseteq \mathsf{M}$, resp.), then there exists a B-concept C such that C is equivalent to $A_1 \sqcup A_2$ ($A_1 \sqcap A_2$, resp.) and each of the atomic concepts A_1, A_2 occurs exactly once in C.

(2.) If $[B] = \mathsf{L}$, then there exists a B-concept C such that C is equivalent to $A_1 \oplus A_2$ and each of the atomic concepts A_1, A_2 occurs exactly once.

(3.) If $\mathsf{N} \subseteq [B]$, then there is a B-concept C such that C is equivalent to $\neg A$ and the atomic concept A occurs in C only once.

(4.) If $[B] = \mathsf{BF}$, then there are B-concepts C and D such that C is equivalent to $A_1 \sqcup A_2$, D is equivalent to $A_1 \sqcap A_2$, and each of the atomic concepts A_1, A_2 occurs in C and D exactly once.

Lemma 4.2.
Let B be a finite set of Boolean operators s.t. $\mathsf{N}_2 \subseteq [B]$ and $\mathscr{Q} \subseteq \{\exists, \forall\}$. Then it holds that $\star\mathrm{SAT}_{\mathscr{Q}}(B) \equiv^{\log}_{\mathrm{m}} \star\mathrm{SAT}_{\mathscr{Q}}(B \cup \{\top, \bot\})$.

Proof. It is easy to observe that the concepts $\top$ and $\bot$ can be simulated by fresh atomic concepts T and B, using the axioms $\neg T \sqsubseteq T$ and $B \sqsubseteq \neg B$. □

Lemma 4.3.
Let B be a finite set of Boolean operators and $\mathcal{Q} \subseteq \{\exists, \forall\}$. Then it holds that $\mathrm{TCSAT}_{\mathcal{Q}}(B) \leq_{\mathrm{m}}^{\log} \mathrm{TSAT}_{\mathcal{Q}\cup\{\exists\}}(B \cup \{\top\})$.

Proof. It can be easily shown that $(C, \mathcal{T}) \in \mathrm{TCSAT}_{\mathcal{Q}}(B)$ iff $(\mathcal{T} \cup \{\top \sqsubseteq \exists R.C\}) \in \mathrm{TSAT}_{\mathcal{Q}}(B \cup \{\top\})$, where R is a fresh role. For "$\Rightarrow$" observe that for the satisfying interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ there must be an individual w' where C holds and then from every individual $w \in \Delta^{\mathcal{I}}$ there has to be an R-edge from w to w' to satisfy $\mathcal{T} \cup \{\top \sqsubseteq \exists R.C\}$. For "$\Leftarrow$" note that for a satisfying interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ all axioms in $\mathcal{T} \cup \{\top \sqsubseteq \exists R.C\}$ are satisfied. In particular the axiom $\top \sqsubseteq \exists R.C$. Hence there must be at least one individual w' s.t. $w' \models C$. Thus $\mathcal{I} \models \mathcal{T}$ and $C^{\mathcal{I}} \supseteq \{w'\} \neq \emptyset$. □

Moreover, for a set B of Boolean operators that allows us to access both constants $\top, \bot \in [B]$, we are able to simulate the negation of an atomic concept in the following way: adding the axioms $A \equiv \exists R_A.\top$ and $A' \equiv \forall R_A.\bot$ to the terminology enforces each model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ interpreting A' as the complement of A, i.e., $(A')^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus A^{\mathcal{I}}$.

Base independence is a very helpful property which allows us to restrict the argumentation to a standard base and lets us generalize the complexity results from $\star\mathrm{SAT}_{\mathcal{Q}}(B_1)$ to $\star\mathrm{SAT}_{\mathcal{Q}}(B_2)$ for *arbitrary* bases B_2 of $[B_1]$. Therefore we will utilize a result from [HSS08] which can be used for any clones in our case due to the formalisms of TBoxes. Hemaspaandra et al. were only able to prove it for clones having access to implication and conjunction.

Lemma 4.4.
Let B_1, B_2 be two finite sets of Boolean operators s.t. $[B_1] = [B_2]$, and let $\mathcal{Q} \subseteq \{\exists, \forall\}$. Then $\star\mathrm{SAT}_{\mathcal{Q}} \leq_{\mathrm{m}}^{\log} \star\mathrm{SAT}_{\mathcal{Q}}(B_2)$.

Proof. According to [HSS08, Theorem 3.6], we translate for any given instance each concept (hence each side of an axiom) into a Boolean circuit over the basis B_1. This circuit can be easily transformed into a circuit over the basis B_2. This new circuit will be expressed by several new axioms that are constructed in the style of the formulae in [HSS08]:

- For input gates g, we add the axiom $g \equiv x_i$.
- If g is a gate computing the Boolean operator $\circ$ and $h_1, \ldots, h_n$ are the respective predecessor gates in this circuit, we add the axiom $g \equiv \circ(h_1, \ldots, h_n)$.
- For $\exists R$-gates g, we add the axiom $g \equiv \exists R.h$.
- Analogously for $\forall R$-gates.

For each axiom $A \sqsubseteq B$, let g_{out}^A and g_{out}^B be the output gates of the appropriate circuits. Then we need to add one new axiom $g_{out}^A \sqsubseteq g_{out}^B$ to ensure the axiomatic property of $A \sqsubseteq B$. For a concept C in the input (relevant for the problems $\mathrm{TCSAT}_{\mathcal{Q}}, \mathrm{OCSAT}_{\mathcal{Q}}$), its translation is mapped to the respective out-gate g_{out}^C.

This reduction is computable in logarithmic space and its correctness can be shown in the same way as in the Proof of [HSS08, Theorem 3.6]. □

The idea for the following lemma goes back to Lewis [Lew79].

Lemma 4.5 (Lewis Knack).
Let B be a finite set of Boolean operators and $\mathcal{Q} \subseteq \{\forall, \exists\}$. Then it holds that $\mathrm{TSAT}_{\mathcal{Q}}(B \cup \{\top\}) \leq_{\mathrm{m}}^{\log} \mathrm{TCSAT}_{\mathcal{Q}}(B)$.

Proof. Similarly as to $\mathbf{SF}(\cdot)$ let $\mathbf{SC}(\mathcal{T})$ be the set of all (sub-)concepts occurring in $\mathcal{T}$. For every $C \in \mathbf{SC}(\mathcal{T})$, we use C_T to denote C with all occurrences of $\top$ replaced by T. Furthermore, we write $\mathcal{T}_T$ for $\{C_T \sqsubseteq D_T \mid C \sqsubseteq D \in \mathcal{T}\}$.

We claim that $\mathcal{T} \in \mathrm{TSAT}_{\mathcal{Q}}(B)$ iff $(\mathcal{T}', T) \in \mathrm{TCSAT}_{\mathcal{Q}}(B)$, where

$$\mathcal{T}' = \mathcal{T}_T \cup \{\, C_T \sqsubseteq T \mid C \in \mathbf{SC}(\mathcal{T})\,\}.$$

For the direction "$\Rightarrow$" observe that for any interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with $\mathcal{I} \models \mathcal{T}$, we can set $T^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and then have $\mathcal{I} \models \mathcal{T}'$ and obviously $T^{\mathcal{I}} \neq \emptyset$.

Now consider the opposite direction "$\Leftarrow$". Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation s.t. $\mathcal{I} \models \mathcal{T}'$ and $T^{\mathcal{I}} \neq \emptyset$. We construct $\mathcal{J}$ from $\mathcal{I}$ via restriction to $T^{\mathcal{I}}$, i.e., $\Delta^{\mathcal{J}} = T^{\mathcal{I}}$, $A^{\mathcal{J}} = A^{\mathcal{I}} \cap T^{\mathcal{I}}$ for atomic concepts A, and $R^{\mathcal{J}} = R^{\mathcal{I}} \cap (T^{\mathcal{I}} \times T^{\mathcal{I}})$ for roles R. We claim the following:

Claim. For every individual $x \in T^{\mathcal{I}}$ and every (sub-)concept C occurring in $\mathcal{T}$, it holds that $x \in C_T^{\mathcal{I}}$ if and only if $x \in C^{\mathcal{J}}$.

This claim implies that $\mathcal{J} \models \mathcal{T}$: for any $x \in \Delta^{\mathcal{J}} = T^{\mathcal{I}}$ and any axiom $D \sqsubseteq E \in \mathcal{T}$, we have that $x \in D^{\mathcal{J}}$ implies $x \in D_T^{\mathcal{I}}$ due to the claim, which implies $x \in E_T^{\mathcal{I}}$ because $\mathcal{I} \models \mathcal{T}'$, which implies $x \in E^{\mathcal{J}}$ due to the claim.

Proof of Claim. We proceed by induction on the structure of C. The base case includes atomic C as well as $\top$ and $\bot$, and follows from the construction of $\mathcal{J}$.

For the induction step, we consider the following cases.

- In case $C = \circ_f(C^1, \ldots, C^n)$, where $\circ_f$ is an arbitrary n-ary boolean operator corresponding to an n-ary Boolean function f, and the C^i are smaller subconcepts of C, the following holds.

$$\begin{aligned} x \in C_T^{\mathcal{I}} \quad &\text{iff} \quad f(\|x \in (C_T^1)^{\mathcal{I}}\|, \ldots, \|x \in (C_T^n)^{\mathcal{I}}\|) = 1 && \text{def. of satisfaction} \\ &\text{iff} \quad f(\|x \in (C^1)^{\mathcal{J}}\|, \ldots, \|x \in (C^n)^{\mathcal{J}}\|) = 1 && \text{induction hypothesis} \\ &\text{iff} \quad x \in C^{\mathcal{J}} && \text{def. of satisfaction} \end{aligned}$$

- In case $C = \exists R.D$, the following holds.

$$\begin{aligned} x \in C_T^{\mathcal{I}} \quad &\text{iff} \quad \text{for some } y \in \Delta^{\mathcal{I}} : (x,y) \in R^{\mathcal{I}} \text{ and } y \in D_T^{\mathcal{I}} \\ &\text{iff} \quad \text{for some } y \in T^{\mathcal{I}} : (x,y) \in R^{\mathcal{I}} \text{ and } y \in D_T^{\mathcal{I}} \\ &\text{iff} \quad \text{for some } y \in T^{\mathcal{I}} : (x,y) \in R^{\mathcal{J}} \text{ and } y \in D^{\mathcal{J}} \\ &\text{iff} \quad x \in C^{\mathcal{J}} \end{aligned}$$

The first equivalence is due to the definition of satisfaction. The second's "$\Rightarrow$" direction is due to the additional axiom $D_T \sqsubseteq T$ in $\mathcal{T}'$, while the "$\Leftarrow$" direction is obvious. The third equivalence is again due to the definition of satisfaction and the construction $\Delta^{\mathcal{I}} = T^{\mathcal{I}}$.

- In case $C = \forall R.D$, we rewrite to $C = \neg\exists R.\neg D$, apply the previous two cases, and rewrite back. □

The following lemma uses the duality of Boolean operators, and quantifiers for stating an equivalence of these dual fragments. Therefore we extend our notion of the operator $\mathbf{dual}(\cdot)$ to quantifiers by defining $\mathbf{dual}(\exists) \underset{\text{def}}{=} \forall$ and $\mathbf{dual}(\forall) \underset{\text{def}}{=} \exists$.

Lemma 4.6 (Contraposition).
Let B be a finite set of Boolean operators and $\mathcal{Q} \subseteq \{\exists, \forall\}$. Then

(1.) $\mathrm{TSAT}_{\mathcal{Q}}(B) \leq^{\log}_{\mathrm{m}} \mathrm{TSAT}_{\mathbf{dual}(\mathcal{Q})}(\mathbf{dual}(B))$, *and*

(2.) $\mathrm{TCSAT}_{\mathcal{Q}}(B) \leq^{\log}_{\mathrm{m}} \mathrm{TCSAT}_{\mathbf{dual}(\mathcal{Q})}(\mathbf{dual}(B) \cup \{\bot, \sqcap\})$,

where $\mathbf{dual}(B) \underset{\text{def}}{=} \{\mathbf{dual}(f) \mid f \in B\}$ *and* $\mathbf{dual}(\mathcal{Q}) = \{\mathbf{dual}(q) \mid q \in \mathcal{Q}\}$.

Proof. Let B be a finite set of Boolean operators and $\mathcal{Q} \subseteq \{\exists, \forall\}$. Let $\mathcal{T} \in \mathfrak{T}_{\mathcal{Q}}(B)$ be a terminology.

(1.) Then it holds that $\mathcal{T} \in \mathrm{TSAT}_{\mathcal{Q}}(B)$ iff $\mathcal{T}^{\mathrm{con}} \in \mathrm{TSAT}_{\mathbf{dual}(\mathcal{Q})}(\mathbf{dual}(B))$, where

$$\mathcal{T}^{\mathrm{con}} \underset{\text{def}}{=} \{D^{\neg} \sqsubseteq C^{\neg} \mid (C \sqsubseteq D) \in \mathcal{T}\},$$

and $C^{\neg}$ is C in negation normal form (all negations are moved inside s.t. they are in front of atomic concepts) and the negated atomic concepts $\neg A$ are replaced with fresh atomic concepts A'. Because of the negation normal form all functions are mapped to their dual and the quantifiers are expressed via their dual one. Therefore note that $C \sqsubseteq D$ iff $\neg D \sqsubseteq \neg C$.

(2.) Here we need the operators $\bot$ and $\sqcap$ to ensure that the input concept C is not instantiated by the same individual as C'. Now it is easy to see that it holds that $(C, \mathcal{T}) \in \mathrm{TCSAT}_{\mathcal{Q}}(B)$ iff

$$(C, \mathcal{T}^{\mathrm{con}} \cup \{C \sqcap C' \sqsubseteq \bot\}) \in \mathrm{TCSAT}_{\mathbf{dual}(\mathcal{Q})}(\mathbf{dual}(B)),$$

where $\mathcal{T}^{\mathrm{con}}$ is as in (1.). □

4.1.1 Both quantifiers

Now we will consider all Boolean function and quantifier fragments in the upcoming sections. Therefore, we start with the fragments that contain both quantifiers $\forall, \exists$.

Due to the interreducibilities stated in Section 2.3.2 on page 26, it suffices to show lower bounds for TSAT and upper bounds for OCSAT. Moreover Lemma 4.4 enables us to restrict the proofs to the standard basis of each clone for stating general results. From Theorem 2.28 we know that $\mathrm{OCSAT}_{\exists\forall}(\mathsf{BF})$ is EXP-complete.

Theorem 4.7.
Let B be a finite set of Boolean operators.

(1.) If $\mathsf{I} \subseteq [B]$ or $\mathsf{N}_2 \subseteq [B]$, then $\mathrm{TSAT}_{\exists\forall}(B)$ is EXP-complete under $\leq_{\mathbf{m}}^{\log}$.

(2.) If $\mathsf{I}_0 \subseteq [B]$ or $\mathsf{N}_2 \subseteq [B]$, then $\star\mathrm{SAT}^{\mathrm{ind}}_{\exists\forall}(B)$ is EXP-complete under $\leq_{\mathbf{m}}^{\log}$.

(3.) If $[B] \subseteq \mathsf{R}_0$, then $\mathrm{TSAT}_{\exists\forall}(B)$ is trivial.

(4.) If $[B] \subseteq \mathsf{R}_1$, then $\star\mathrm{SAT}_{\exists\forall}(B)$ is trivial.

Proof. Parts 1.–4. are formulated as Lemmas 4.8 to 4.12, and are proven below. □

The first two parts describe the high expressivity of terminologies. As they contain limited forms of implication and conjunction the restriction of Boolean operators to only constants (or only the constant $\bot$ for the case $\star\mathrm{SAT}^{\mathrm{ind}}_{\exists\forall}$) does not change the overall complexity of the fragment and remains highly intractable. Further the results of this classification differ from similar analyses of sub-Boolean modal logics by having such hard cases near the bottom of Post's lattice.

Furthermore, the second part of the theorem is a generalization of subsumption for $\mathcal{FL}_0$ and $\mathcal{AL}$ with respect to GCIs [GMWK02, Don03, BBL05a, Hof05]. The contrast to the tractability of subsumption with respect to GCIs in $\mathcal{EL}$, which uses only existential quantifiers, undermines the observation that, for negation-free fragments, the choice of the quantifier affects tractability and not the choice between conjunction and disjunction. The logics DL-Lite and $\mathcal{ALU}$ cannot be put into this context because they use unqualified restrictions, that are, expressions of the form $\exists R.\top$ which only allow to enforce the existence of an R-role but nothing more.

The last two parts reflect the fact that TSAT is less expressive than the other three decision problems: a terminology alone cannot *speak* about one single individual, therefore it cannot simulate the constant $\top$ as contrast to the instances of $\star\mathrm{SAT}^{\mathrm{ind}}$.

Lemma 4.8.
Let B be a finite set of Boolean operators s.t. B contains only $\top$-reproducing operators. Then $\mathrm{OCSAT}_{\exists\forall}(B)$ is trivial.

Proof. According to Post's lattice, every B that does not fall under Theorem 4.7 (1.)–(3.) contains only $\top$-reproducing operators. Hence the following interpretation satisfies any instance $(\mathcal{O}, C)$: $\mathcal{I} = (\{w\}, \cdot^{\mathcal{I}})$ s.t. $A^{\mathcal{I}} = \{w\}$ for each atomic concept A, $r^{\mathcal{I}} = \{(w,w)\}$ for each role r, and $a^{\mathcal{I}} = w$ for each individual a. It then holds trivially that $\mathcal{I} \models \mathcal{O}$ and $C^{\mathcal{I}} = \{w\} \neq \emptyset$. □

Lemma 4.9.
Let B be a finite set of Boolean operators s.t. B contains only $\bot$-reproducing operators. Then $\mathrm{TSAT}_{\exists\forall}(B)$ *is trivial.*

Proof. The intepretation $\mathscr{I} = (\{w\}, \cdot^{\mathscr{I}})$ with $A^{\mathscr{I}} = \emptyset$ for each atomic concept A, and $r^{\mathscr{I}} = \{(w,w)\}$ for each role r satisfies any instance $\mathscr{T}$ for $\mathrm{TSAT}_{\exists\forall}(B)$, where B contains only $\bot$-reproducing operators. This follows from the observation that for each axiom $A \sqsubseteq B$ in $\mathscr{T}$ both sides are always falsified by $\mathscr{I}$ (because every atomic concept is falsified, and we only have $\bot$-reproducing operators as connectives). This can be shown by an easy induction on the concept structure. Please note that we need to construct a looping node concerning the transition relations due to the fact that we need to falsify axioms with $\forall r.\bot$ on the left side for some role r. If we set $r^{\mathscr{I}} = \emptyset$ then this expression would be satisfied and would contradict our argumentation for the axiom $\forall r.\bot \sqsubseteq \bot$. Moreover this construction cannot fulfill wrongly the left side of an axiom because of the absence of $\top$ and as no atomic concept has instances with w. □

As an intermediate step we will show that for $B \in \{\mathsf{V}, \mathsf{E}\}$ the problem $\mathrm{TCSAT}_{\exists\forall}(B)$ is EXP-hard. Finally we will prove how to remove the conjunction operator of concepts reaching the hardness for I.

Lemma 4.10.
Let B be a finite set of Boolean operators with $\{\bot, \sqcap\} \subseteq [B]$, or $\{\bot, \sqcup\} \subseteq [B]$. Then $\star\mathrm{SAT}^{\mathrm{ind}}_{\exists\forall}(B)$ *is* EXP*-complete w.r.t.* $\leq^{\log}_{\mathrm{m}}$. *If* $\mathsf{D} \subseteq [B]$, *then* $\mathrm{TSAT}_{\exists\forall}(B)$ *is* EXP*-complete w.r.t.* $\leq^{\log}_{\mathrm{m}}$.

Proof. The membership in EXP for $\mathrm{OCSAT}_{\exists\forall}(B)$ follows from Theorem 2.28 on page 26 in combination with Lemma 4.4.

For EXP-hardness, we first consider the case $\sqcap \in [B]$ and reduce from the positive entailment problem for Tarskian set constraints in [GMWK02]: thus we start from the question if $\mathscr{T} \models C \sqsubseteq D$, for concepts C, D and a terminology $\mathscr{T}$ that uses the quantifiers $\forall$ and $\exists$, and $\sqcap$ as the only Boolean connective. Hence $\mathscr{T} \models C \sqsubseteq D$ if and only if $\mathscr{T}' \notin \mathrm{TSAT}_{\exists\forall}(\{\sqcap, \top, \bot\})$, for $\mathscr{T}' \underset{\text{def}}{=} \mathscr{T} \cup \{\top \sqsubseteq \exists R.(C \sqcap D'),\ D' \equiv \exists R_D.\top,\ D \equiv \forall R_D.\bot\}$, where D' is a new atomic concept and R, R_B are new roles. This holds as C does not imply D iff there is an instance of C which is not an instance of D. As D and D' are declared disjoint, the claim applies. Now for $\mathrm{TCSAT}_{\exists\forall}(\{\bot, \sqcap\})$, we transform $\mathscr{T}'$ into $\mathscr{T}''$ by substituting the two introduced occurrences of $\top$ with a fresh concept name C and put C into the instance of $\mathrm{TCSAT}_{\exists\forall}(\{\bot, \sqcap\})$ we are reducing to. Then, $\mathscr{T} \models C \sqsubseteq D$ iff $(\mathscr{T}'', C) \notin \mathrm{TCSAT}_{\exists\forall}(\{\bot, \sqcap\})$.

For $\mathrm{TCSAT}_{\exists\forall}(\{\bot, \sqcup\})$, we modify the above definition of $\mathscr{T}''$ to dispose of the introduced conjunction: using a fresh atomic concept E, we set $\mathscr{T}'' \underset{\text{def}}{=} \mathscr{T} \cup \{E \sqsubseteq C,\ E \sqsubseteq D',\ C \sqsubseteq \exists R.E,\ D' \equiv \exists R_D.C,\ D \equiv \forall R_D.\bot\}$. Observe that $\mathscr{T}$ still consists of concept expressions over $\bot$ *and* $\sqcup$.

The remaining case for the self-dual operators follows from Lemmas 4.1 and 4.2, as all self-dual functions in combination with the constants $\top, \bot$ (to which we have access as $\neg$ is self-dual) can express any arbitrary Boolean function. □

Lemma 4.11.
Let B, B' be finite sets of Boolean operators s.t. $\mathsf{I}_0 \subseteq [B]$ and $\mathsf{I} \subseteq [B']$. Then $\star\mathrm{SAT}^{\mathrm{ind}}_{\exists\forall}(B)$ and $\mathrm{TSAT}_{\exists\forall}(B')$ are EXP*-complete w.r.t. $\leq^{\log}_{\mathrm{m}}$.*

Proof. For the upper bound we apply Theorem 2.28 and Lemma 4.4. For hardness, we reduce from $\mathrm{TSAT}_{\exists\forall}(\{\sqcap, \bot, \top\})$ to $\mathrm{TSAT}_{\exists\forall}(\{\bot, \top\})$—the former shown to be EXP-complete in the proof of Lemma 4.10. The main idea is an extension of the normalization rules in [Bra04b]. The following normalization rules have been stated and proven to be correct in [Bra04b]:

$$
\begin{array}{lllll}
(\mathbf{NF1}) & \hat{C} \sqcap D \boxdot E & \rightsquigarrow & \{A \equiv \hat{C}, A \sqcap D \boxdot E\} \\
(\mathbf{NF2}) & C \boxdot D \sqcap \hat{E} & \rightsquigarrow & \{C \boxdot D \sqcap A, A \equiv \hat{E}\} \\
(\mathbf{NF3}) & \exists r.\hat{C} \boxdot D & \rightsquigarrow & \{A \equiv \hat{C}, \exists r.A \boxdot D\} \\
(\mathbf{NF4}) & C \boxdot \exists r.\hat{D} & \rightsquigarrow & \{C \boxdot \exists r.A, A \equiv \hat{D}\} \\
(\mathbf{NF5}) & C \sqsubseteq D \sqcap E & \rightsquigarrow & \{C \sqsubseteq D, C \sqsubseteq E\} \\
(\mathbf{NF6}) & C \equiv D & \rightsquigarrow & \{C \sqsubseteq D, D \sqsubseteq C\}
\end{array}
$$

where $\boxdot \in \{\sqsubseteq, \equiv\}$, $\hat{C}$ states that the concept description C is no concept name, and A is a new concept name.

Now we want to extend these rules for conjunctions on the left side of GCIs and for $\forall$-quantification:

$$
\begin{array}{lllll}
(\mathbf{NF3b}) & \forall r.\hat{C} \boxdot D & \rightsquigarrow & \{A \equiv \hat{C}, \forall r.A \boxdot D\} \\
(\mathbf{NF4b}) & C \boxdot \forall r.\hat{D} & \rightsquigarrow & \{A \equiv \hat{D}, C \boxdot \forall r.A\} \\
(\mathbf{NF7}) & A \sqcap B \sqsubseteq C & \rightsquigarrow & \{A \sqsubseteq \exists R_A.\top, B \sqsubseteq \forall R_A.A', \exists R_A.A' \sqsubseteq C\}
\end{array}
$$

where R_A is a fresh role, and A' is a fresh concept name. For (**NF7**) we will prove its correctness.

Assume $A \sqcap B \sqsubseteq C$ holds in the interpretation $\mathscr{I} = (\Delta^{\mathscr{I}}, \cdot^{\mathscr{I}})$. Thus for each individual $w \in \Delta^{\mathscr{I}}$ with $w^{\mathscr{I}} \supseteq \{A, B\}$ it holds $C \in w^{\mathscr{I}}$ as assumed.

In the following we will construct a modified interpretation $\mathscr{I}'$ from $\mathscr{I}$ that satisfies the axioms constructed by (**NF7**), i.e., the axioms in $\{A \sqsubseteq \exists R_A.\top, B \sqsubseteq \forall R_A.A', \exists R_A.A' \sqsubseteq C\}$. As $A \in w^{\mathscr{I}'}$, we add one R_A-edge to the same individual w, and due to $B \sqsubseteq \forall R_A.A'$ we must add A' to $w^{\mathscr{I}'}$. Finally the last GCI is satisfied as we have $C \in w^{\mathscr{I}'}$.

For the opposite direction assume $A \sqcap B \sqsubseteq C$ cannot be satisfied, i.e., in every interpretation there is an individual which is an instance of A and B but not of C. Hence we take an arbitrary interpretation $\mathscr{I}$ such that it satisfies the first two axioms $A \sqsubseteq \exists R_A.\top$ and $B \sqsubseteq \forall R_A.A'$. Due to our assumption every individual w is in instance of A and B, and hence we have an R_A-edge to an individual where A' must hold. Therefore the left side of the third axiom is fulfilled but C does not hold for the individual w. Hence this axiom is not satisfied and we have the desired contradiction.

As this normalization procedure runs in polynomial time and eliminates every conjunction of concepts, we have a reduction from $\mathrm{TCSAT}_{\exists\forall}(\{\sqcap, \bot\})$ to $\mathrm{TCSAT}_{\exists\forall}(\{\bot\})$, and also from $\mathrm{TSAT}_{\exists\forall}(\{\sqcap, \bot, \top\})$ to $\mathrm{TSAT}_{\exists\forall}(\{\top, \bot\})$. Hence the Lemma applies. □

Lemma 4.12.
$\star\mathrm{SAT}_{\exists\forall}(\mathsf{N}_2)$ *is* EXP*-complete w.r.t.* $\leq_{\mathrm{m}}^{\log}$.

Proof. The upper bound follows from Theorem 2.28 and Lemma 4.4. For the lower bound use Lemma 4.2 to simulate $\top$ and $\bot$ with fresh atomic concepts. Then the argumentation follows similarly to Lemmas 4.10 and 4.11. □

4.1.2 Restricted quantifiers

In this section we investigate the complexity of $\mathrm{OCSAT}_{\mathcal{Q}}$, $\mathrm{OSAT}_{\mathcal{Q}}$, $\mathrm{TCSAT}_{\mathcal{Q}}$, and $\mathrm{TSAT}_{\mathcal{Q}}$, where $\mathcal{Q}$ contains at most one of the quantifiers $\exists$ or $\forall$. Even the case $\mathcal{Q} = \emptyset$ is nontrivial: for example, $\mathrm{TSAT}_{\emptyset}(B)$ does not reduce to propositional satisfiability for B because the (restricted) use of implication and conjunction is implicit in sets of axioms.

Theorem 4.13 (Results for Terminology Satisfiability without Quantifiers).
Let B be a finite set of Boolean operators.

(1.) If $\mathsf{L}_3 \subseteq [B]$ *or* $\mathsf{M} \subseteq [B]$, *then* $\mathrm{TSAT}_{\emptyset}(B)$ *is* NP*-complete w.r.t.* $\leq_{\mathrm{m}}^{\log}$.

(2.) If $\mathsf{E} = [B]$ *or* $\mathsf{V} = [B]$, *then* $\mathrm{TSAT}_{\emptyset}(B)$ *is* P*-complete w.r.t.* $\leq_{\mathrm{m}}^{\log}$.

(3.) If $[B] \in \{\mathsf{I}, \mathsf{N}_2, \mathsf{N}\}$, *then* $\mathrm{TSAT}_{\emptyset}(B)$ *is* NLOGSPACE*-complete w.r.t.* $\leq_{\mathrm{m}}^{\log}$.

(4.) Otherwise (if $[B] \subseteq \mathsf{R}_1$ *or* $[B] \subseteq \mathsf{R}_{\mathcal{O}}$*), then* $\mathrm{TSAT}_{\emptyset}(B)$ *is trivial.*

Proof. NP-completeness for (1.) results for the upper bound from $\mathrm{OCSAT}_{\exists}(\{\sqcap, \neg, \top, \bot\})$ whose membership in NP is proven in Lemma 4.27 and the lower bounds which are proven in Lemmas 4.14 and 4.15. Both lower bounds of (2.) will be proven through Lemmas 4.16 and 4.17. The upper bound is due to $\mathrm{OCSAT}_{\exists}(\{\sqcap, \top, \bot\})$ which is shown to be in P in Lemma 4.33. The membership of the third item results from $\mathrm{TCSAT}_{\emptyset}(\{\neg, \top\})$ which is proven to be in NLOGSPACE in Lemma 4.28 and the hardness result is proven in Lemma 4.18. Item (4.) follows through Lemmas 4.8 and 4.9. □

Lemma 4.14.
Let B be a finite set of Boolean operators s.t. $\mathsf{D} \subseteq [B]$ *or* $\mathsf{M} \subseteq [B]$. *Then* $\mathrm{TSAT}_{\emptyset}(B)$ *is* NP*-hard.*

Proof. We start with the implication problem for the self-dual (resp. monotone) fragment of propositional logic IMP(D) (resp. IMP(M)), which is shown to be coNP-complete in [BMTV09b]. To establish NP-hardness of $\mathrm{TSAT}_{\emptyset}(\mathsf{M})$, we reduce from the complement of IMP(M) in the following way. Let φ, ψ be two propositional formulae with monotone operators only. Then

$$(\varphi, \psi) \notin \mathrm{IMP}(\mathsf{M}) \quad \text{iff} \quad \varphi \not\models \psi \quad \text{iff} \quad \exists \theta : \theta \models \varphi \wedge \neg\psi$$
$$\text{iff} \quad \{C_\psi \sqsubseteq \bot, \top \sqsubseteq C_\varphi\} \in \mathrm{TSAT}_{\emptyset}(\mathsf{M}),$$

where C_φ and C_ψ are concepts corresponding to φ, ψ in the usual way.

For $\mathrm{TSAT}_{\emptyset}(\mathsf{D})$, we use the same reduction, but need to replace the introduced operators $\top, \bot$ as in Lemma 4.2. □

Lemma 4.15.
Let B be a finite set of Boolean operators s.t. $\mathsf{L}_3 \subseteq [B]$, then $\mathrm{TSAT}_\emptyset(B)$ is NP*-hard.*

Proof. Here we will provide a reduction from the NP-complete problem 1-in-3-SAT. In the following we can use the binary exclusive-or as we have access to negation because $x \oplus x \oplus z \oplus \top \equiv \neg z$, and we have access to both constants $\top$ and $\bot$ due to Lemma 4.2. Thus we are able to use the binary exclusive-or operator because $x \oplus y \oplus \top \oplus \top \equiv x \oplus y$.

The main idea of the reduction is to use for each clause $(x \vee y \vee z) \in \varphi$ an axiom $\top \sqsubseteq x \oplus y \oplus z$ is used to enforce that only one literal is satisfied. As for this axiom it is possible to have all literals satisfied we need some additional axioms to circumvent this problem.

Let φ defined as above, then the reduction is defined as $\varphi \mapsto \mathcal{T}$, where

$$\begin{aligned}
\mathcal{T} \underset{\text{def}}{=} &\left\{\top \sqsubseteq f(l_{i1}) \oplus f(l_{i2}) \oplus f(l_{i3}) \oplus s^i \oplus \top \mid 1 \le i \le n\right\} \cup \\
&\cup \{\top \sqsubseteq f(l_{i1}) \oplus f(l_{i2}) \oplus f(l_{i3}) \mid 1 \le i \le n\} \cup \\
&\cup \left\{s_1^i \sqsubseteq f(l_{i1}) \oplus f(l_{i2}) \mid 1 \le i \le n\right\} \cup \\
&\cup \left\{s_2^i \sqsubseteq f(l_{i1}) \oplus f(l_{i3}) \mid 1 \le i \le n\right\} \cup \\
&\cup \left\{s_3^i \sqsubseteq f(l_{i2}) \oplus f(l_{i3}) \mid 1 \le i \le n\right\} \cup \\
&\cup \left\{s^i \sqsubseteq s_1^i \oplus s_2^i \oplus s_3^i \mid 1 \le i \le n\right\} \cup \\
&\cup \left\{\top \sqsubseteq A_x \oplus A_{x'} \mid x \text{ variable in } \varphi\right\},
\end{aligned}$$

where $f(x) = A_x$ and $f(\bar{x}) = A_{x'}$. Now we claim that $\varphi \in$ 1-in-3-SAT iff $\mathcal{T} \in \mathrm{TSAT}_\emptyset(\mathsf{L}_0)$.

Consider an arbitrary clause $c = x \vee y \vee z$ from φ with x, y, z literals. Then the following axioms which differ for convenience slightly from the notion above are part of $\mathcal{T}$

$$\top \sqsubseteq x \oplus y \oplus z \oplus s \oplus \top \quad (4.1)$$
$$\top \sqsubseteq x \oplus y \oplus z \quad (4.2)$$
$$s_1 \sqsubseteq x \oplus y$$
$$s_2 \sqsubseteq x \oplus z$$
$$s_3 \sqsubseteq y \oplus z$$
$$s \sqsubseteq s_1 \oplus s_2 \oplus s_3.$$

x	y	z	s_1	s_2	s_3	s	(4.1)	(4.2)
0	0	0						n
0	0	1	$\underline{0}$	1	0	$\underline{1}$	y	y
0	1	0	1	$\underline{0}$	0	$\underline{1}$	y	y
0	1	1						n
1	0	0	1	0	$\underline{0}$	$\underline{1}$	y	y
1	0	1						n
1	1	0						n
1	1	1	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	n	y

The table on the upper right shows each possible assignment for x, y, z and suitable assignments for the s_is and the validity of the axioms (4.1) and (4.2). Underlined numbers denote mandatory truth values which are enforced by the axioms, whereas blank cells denote arbitrary choices. If at least one of (4.1) and (4.2) are contradicted then there exists no interpretation for $\mathcal{T}$. At first we start with an interpretation that assigns the individuals x, y, z to the recent world in some way. Then we immediately observe if axiom (4.2) is

contradicted or not. If not contradicted then we have to look at the remaining s_i axioms in order to find an extension of this interpretation which assigns the s_is and s in a way such that (4.2) is not violated whenever we have an interpretation which corresponds to a valid 1-in-3-SAT assignment. Otherwise we have to show that there exists no possible extension that falsely satisfies axiom (4.2).

Thus the table shows that for every eligible assignment we always have a fulfilling interpretation, and for every improper assignment it is not possible to construct a fulfilling one.

Lemma 4.16.
Let B be a finite set of Boolean operators s.t. $\mathsf{E} \subseteq [B]$, then $\mathrm{TSAT}_\emptyset(B)$ is P*-hard.*

Proof. In the following we will state a $\leq_{\mathrm{cd}}$-reduction from the complement of the P-complete problem HGAP, which is the accessibility problem for directed hypergraphs. In a given hypergraph $H = (V, E)$, a hyperedge $e \in E$ is a set of source nodes $src(e) \subseteq V$ and one destination node $dest(e) \in V$. Instances of HGAP consist of a directed hypergraph $H = (V, E)$, a set $S \subseteq V$ of source nodes, and a target node $t \in V$. Now the question is whether there exists a hyperpath from the set S to the node t, i.e., whether there are hyperedges $e_1, e_2, \ldots, e_k$ s.t. for each e_i there are $e_{i_1}, \ldots, e_{i_\nu}$ with $1 \leq i_1, \ldots, i_\nu < i$ and $\bigcup_{j \in \{i_1, \ldots, i_\nu\}} dest(e_j) \cup src(e_j) \supseteq src(e_i)$, and $src(e_1) = S$ and $dest(e_k) = t$.

HGAP remains P-complete even if we restrict the hyperedges to contain at most two source nodes [SI90]. W.l.o.g. assume that if there is a path from S to t, then the last edge of that path is a usual edge with only one source node.

Let $G = (V, E)$ be a directed hypergraph, $\{s_1, \ldots, s_k\} = S \subseteq V$ with $s_1, \ldots, s_k \in V$ be the set of source nodes, and $t \in V$ be the target node. For each node $v \in V$, we use a new atomic concept v. In addition let t, t' be fresh atomic concepts. Now define

$$\mathscr{T} \underset{\text{def}}{=} \{u_1 \sqcap \ldots \sqcap u_k \sqsubseteq v \mid (u_1, \ldots, u_k; v) \in E\} \cup \{\top \sqsubseteq s_1 \sqcap \ldots \sqcap s_k \sqcap t', t \sqcap t' \sqsubseteq \bot\}.$$

Then $(G, S, t) \in$ HGAP iff $\mathscr{T} \notin \mathrm{TSAT}_\emptyset(\{\sqcap, \top, \bot\})$.

"$\Rightarrow$": Assume there is a hyperpath from S to t as above. Thus in every interpretation $\mathscr{I} = (\Delta^{\mathscr{I}}, \cdot^{\mathscr{I}})$ it holds for all $w \in \Delta^{\mathscr{I}}$ that $s_1, \ldots, s_k, t' \in w^{\mathscr{I}}$. As the before mentioned hyperpath exists, t must also be in $w^{\mathscr{I}}$ through the chain of axioms that correspond to the hyperedges in the path. This violates the axiom $t \sqcap t' \sqsubseteq \bot$.

"$\Leftarrow$": Assume there is no hyperpath from S to t in $G = (V, E)$. Hence there is no chain of axioms that enforce t to be true in every state. Therefore we are able to construct a satisfying interpretation in the following way: $\mathscr{I} = (\{w\}, \cdot^{\mathscr{I}})$ and

$$w^{\mathscr{I}} \underset{\text{def}}{=} \{v \mid (s_1, \ldots, s_k; v) \in E^*\} \cup \{t'\},$$

where E is the transitive closure of E. Please note that $(s_1, \ldots, s_k; t) \notin E^*$ and thus $t \notin w^{\mathscr{I}}$. Therefore, all axioms are satisfied and $\mathscr{T} \in \mathrm{TSAT}_\emptyset(\{\sqcap, \top, \bot\})$. □

Lemma 4.17.
Let B be a finite set of Boolean operators s.t. $\mathsf{V} \subseteq [B]$, then $\mathrm{TSAT}_\emptyset(B)$ is P*-hard.*

Proof. To realize the desired lower bound, we use Lemma 4.6 to state a reduction from $\mathrm{TSAT}_\emptyset(\mathsf{E})$ to $\mathrm{TSAT}_\emptyset(\mathsf{V})$. □

Lemma 4.18.
Let B be a finite set of Boolean operators s.t. $\mathsf{I} \subseteq [B]$, then $\mathrm{TSAT}_\emptyset(B)$ is NLOGSPACE*-hard.*

Proof. For proving NLOGSPACE-hardness we will reduce from the complement of the graph accessibility problem GAP which is NLOGSPACE-complete. Consider a given directed graph $G = (V, E)$ and two nodes $s, t \in V$ as the recent instance for GAP asking for a path from s to t in G. We introduce a concept name A_v per node $v \in V$ and define

$$\mathcal{T} \underset{\text{def}}{=} \{A_u \sqsubseteq A_v \mid (u, v) \in E\} \cup \{\top \sqsubseteq A_s, A_t \sqsubseteq \bot\}.$$

We will now prove that $(G, s, t) \notin \mathrm{GAP}$ iff $\mathcal{T} \in \mathrm{TSAT}_\emptyset(B)$.

"$(G, s, t) \notin \mathrm{GAP} \Rightarrow \mathcal{T} \in \mathrm{TSAT}_\emptyset(B)$": Assume there is no path from s to t. Take the interpretation $\mathcal{I} \underset{\text{def}}{=} (\{x\}, \cdot^{\mathcal{I}})$ with

$$A_v^{\mathcal{I}} \underset{\text{def}}{=} \begin{cases} \{x\} & \text{if } v \text{ is reachable from } s, \\ \emptyset & \text{otherwise,} \end{cases}$$

for each $v \in V$. Then $A_t^{\mathcal{I}} = \emptyset$ and with that all axioms are satisfied. Thus it holds that $\mathcal{I} \models \mathcal{T}$.

"$(G, s, t) \in \mathrm{GAP} \Rightarrow \mathcal{T} \notin \mathrm{TSAT}_\emptyset(B)$": Now assume we have a path $\pi = v_1, \ldots, v_k$ in G with $k \in \mathbb{N}$, $(v_i, v_{i+1}) \in E$, $v_i \in V$ for $1 \leq i \leq k$, $v_1 = s$, and $v_k = t$ from s to t. Now any interpretation needs to include an individual x instantiating A_s (else $\top \sqsubseteq A_s$ would be contradicted) and also $A_{v_2}, \ldots, A_{v_k} = A_t$. But with $A_t \in x^{\mathcal{I}}$ we contradict the axiom $A_t \sqsubseteq \bot$. Thus $\mathcal{I} \not\models \mathcal{T}$, and with that $\mathcal{T} \notin \mathrm{TSAT}_\emptyset(B)$. □

Lemma 4.19.
Let B be a finite set of Boolean operators s.t. $[B] \subseteq \mathsf{I}$, then $\mathrm{TSAT}_\emptyset(B)$ is in NLOGSPACE*.*

Proof. The main idea is to do a path search in a concept dependence graph—a reduction to the complement of GAP. A given $\mathcal{T}$ is mapped to $G = (V, E)$ where

$$V \underset{\text{def}}{=} \{v_A, v_B \mid A \sqsubseteq B \in \mathcal{T}\} \cup \{v_\top, v_\bot\} \text{ and}$$
$$E \underset{\text{def}}{=} \{(v_A, v_B) \mid A \sqsubseteq B\}.$$

Now it holds $\mathcal{T} \in \mathrm{TSAT}_\emptyset(B)$ iff $(G, v_\top, v_\bot) \notin \mathrm{GAP}$. Please note that we need to add $v_\top, v_\bot$ to V in order to keep consistency if at least one of $\top$ and $\bot$ is not part of an axiom side. If $\mathcal{T}$ is not satisfiable, then in every interpretation there is at least one axiom contradicted. w.l.o.g. the contradicted axiom is of the form $C \sqsubseteq \bot$ and C is instantiated by some individual x. Thus there must be a chain of axioms that enforce C to be true and it can be easily shown that this chain starts at some axiom $\top \sqsubseteq C'$. Hence we have a path starting at $v_\top$ in the Graph G which leads to a node $v_\bot$. For the opposite direction the argumentation is analogue. □

Theorem 4.20 (Results for Terminology Satisfiability with One Quantifier).
Let B be a finite set of Boolean operators and $\mathcal{Q} \in \{\forall, \exists\}$ be a quantifier.

(1.) If $\mathsf{M} \subseteq [B]$ or $\mathsf{N}_2 \subseteq [B]$, then $\mathrm{TSAT}_{\mathcal{Q}}(B)$ is EXP-complete w.r.t. $\leq_{\mathrm{m}}^{\log}$.

(2.) If $\mathsf{E} = [B]$, $\mathsf{V} = [B]$, or $\mathsf{I} = [B]$, then $\mathrm{TSAT}_{\mathcal{Q}}(B)$ is P-complete w.r.t. $\leq_{\mathrm{m}}^{\log}$.

(3.) Otherwise (if $[B] \subseteq \mathsf{R}_1$ or $[B] \subseteq \mathsf{R}_0$), then $\mathrm{TSAT}_{\mathcal{Q}}(B)$ is trivial.

Proof. For the monotone case in (1.) consider Lemmas 4.21 and 4.22. The proof for N_2 can be found in Lemma 4.23. The respective upper bounds for (1.) result from Theorem 2.28 in combination with Lemma 4.4. The needed lower bound for the P-hardness results in (2.) is shown for $\mathrm{TSAT}_{\exists}(\mathsf{I})$ in Lemma 4.25 (case $\forall$ is due to Lemma 4.6). The membership in P for the cases in (3.) result on the one hand from $\mathrm{OCSAT}_{\exists}(\sqcap, \top, \bot)$ which is shown to be in P in Lemma 4.33 and on the other hand from $\mathrm{TSAT}_{\forall}(\sqcap, \top, \bot)$ is proven in Lemma 4.24. The two remaining upper bounds for $[B] = \mathsf{V}$ follow from the complementary problem through Lemma 4.6.

Item (3.) follows through Lemmas 4.8 and 4.9. □

Part (3.) generalizes the fact that every $\mathcal{EL}$- and $\mathcal{FL}_0$-TBox is satisfiable, and the whole theorem shows that separating either conjunction and disjunction, or the constants is the only way to achieve tractability for TSAT.

Lemma 4.21.
Let B be a finite set of Boolean operators s.t. $\mathsf{M} \subseteq [B]$, then $\mathrm{TSAT}_{\exists}(B)$ is EXP-hard.

Proof. For EXP-hardness, we will reduce from the complement of the subsumption problem w.r.t. TBoxes for the logic $\mathcal{ELU}$, which has been investigated in [BBL05a, Thm. 7]. $\mathcal{ELU}$ is $\mathcal{ALC}$ restricted to the operators $\top, \sqcap, \sqcup, \exists$. Now it holds that

$$
\begin{aligned}
&(\mathcal{T}, A, B) \in \mathcal{ELU}\text{-SUBS}\\
&\quad \text{iff} \quad \mathcal{T} \models A \sqsubseteq B\\
&\quad \text{iff} \quad \text{for all } \mathcal{I} : \mathcal{I} \models \mathcal{T} \text{ implies } \mathcal{I} \models A \sqsubseteq B\\
&\quad \text{iff} \quad \text{there is no } \mathcal{I} : \mathcal{I} \models \mathcal{T} \text{ and } \mathcal{I} \models A \not\sqsubseteq B\\
&\quad \text{iff} \quad \text{there is no } \mathcal{I} : \mathcal{I} \models \mathcal{T} \text{ and } \mathcal{I} \models \top \sqsubseteq \exists R.A \sqcap \neg B\\
&\quad \text{iff} \quad \text{there is no } \mathcal{I} : \mathcal{I} \models \underbrace{\mathcal{T} \cup \{\top \sqsubseteq \exists R.(A \sqcap B'), \top \sqsubseteq B \sqcup B', B \sqcap B' \sqsubseteq \bot\}}_{\mathcal{T}'}\\
&\quad \text{iff} \quad \mathcal{T}' \notin \mathrm{TSAT}_{\exists}(\mathsf{M}),
\end{aligned}
$$

for a fresh role R and a fresh concept B'. □

Lemma 4.22.
Let B be a finite set of Boolean operators s.t. $\mathsf{M} \subseteq [B]$, then $\mathrm{TSAT}_{\forall}(B)$ is EXP-hard.

Proof. As in the proof of Lemma 4.17, we can reduce from the dual problem $\mathrm{TSAT}_{\exists}(B)$ through Lemma 4.6. □

Lemma 4.23.
Let B be a finite set of Boolean operators s.t. $\mathsf{N}_2 \subseteq [B]$ *and* $\mathcal{Q} \in \{\forall, \exists\}$, *then* $\mathrm{TSAT}_{\mathcal{Q}}(B)$ *is* EXP*-hard.*

Proof. We reduce from $\mathrm{TSAT}_{\exists\forall}(\mathsf{I})$ whose EXP-hardness follows from Lemma 4.12. As known from Lemma 4.2, we can simulate the constants using new concept names and negation. Additionally observe that, although $\mathcal{Q}$ contains only one quantifier, the other quantifier can be expressed using $\neg$. □

Lemma 4.24.
Let B be a finite set of Boolean operators s.t. $[B] \subseteq \mathsf{E}$, *then* $\mathrm{TSAT}_{\forall}(B)$ *is in* P.

Proof. Here we will specify an algorithm for satisfiability similar to the one in [Bra04a] that constructs iteratively the transitive closure of atomic concepts that imply each other. Thus, informal speaking, starting by the empty set $S_0 \underset{\text{def}}{=} \emptyset$, for each S_i we look at each axiom $C \sqsubseteq D$ and add D to S_{i+1} iff $C \in S_i$. The construction of these sets is defined inductively as follows, where $\mathcal{T}$ is a TBox that is in normalform (i.e., $\mathcal{T}$ contains only expressions of the form $C \sqsubseteq D$, $C_1 \sqcap C_2 \sqsubseteq D$, $\forall r.C \sqsubseteq D$, or $C \sqsubseteq \forall r.D$, where C and D are atomic concepts and r is a role–please note that for each S_i it holds $S_i \subseteq (\mathsf{N_C} \cup \{\top, \bot\})^*$):

(IS1) If $C_1 \in S_i(C)$ and $C_1 \sqsubseteq D \in \mathcal{T}$, then $S_{i+1}(C) \underset{\text{def}}{=} S_i(C) \cup \{D\}$.

(IS2) If $C_1, C_2 \in S_i(C)$ and $C_1 \sqcap C_2 \sqsubseteq D \in \mathcal{T}$, then $S_{i+1}(C) \underset{\text{def}}{=} S_i(C) \cup \{D\}$.

(IS3) If $C_1 \in S_i(C)$ and $C_1 \sqsubseteq \forall r.D \in \mathcal{T}$ and $D_1 \in S_i(D)$ and $\forall r.D_1 \sqsubseteq C \in \mathcal{T}$, then $S_{i+1}(C) \underset{\text{def}}{=} S_i(C) \cup \{D\}$.

The construction for each of those sets S_i takes time at most $\mathcal{O}(|\mathcal{T}|)$ and eventually stops for an atomic concept C if $S_i(C) = S_{i+1}(C)$ for some $i \in \mathbb{N}$.

We now claim that $\mathcal{T} \in \mathrm{TSAT}_{\forall}(B)$ iff $\bot \notin S_*^{\mathcal{T}}(\top)$, where $S_*^{\mathcal{T}}(\top)$ denotes the transitive closure of S_i for $\top$ w.r.t. $\mathcal{T}$.

"⇒": Let $\mathcal{T} \in \mathrm{TSAT}_{\forall}(B)$ via the interpretation $\mathcal{I}$. Hence $\mathcal{I} \models \mathcal{T}$ and in particular for each $C \sqsubseteq D \in \mathcal{T}$ it holds that $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. As **(IS1)** to **(IS3)** hold, we have $\bot \notin S_*^{\mathcal{T}}(\top)$, otherwise there exist $C_1 \sqsubseteq D_1, \ldots, C_\ell \sqsubseteq D_\ell \in \mathcal{T}$ s.t. $C_1 = \top$ and $D_\ell = \bot$, and C_1 implies D_ℓ through these axioms. We show this by induction on n, where n is the index of the first S_i with $\bot \notin S_i^{\mathcal{T}}(\top)$.

Let $n = 1$, then $C_1 = \top$ and $D_1 = \bot$; hence we apply **(IS1)** for $\top \sqsubseteq \bot \in \mathcal{T}$ and $\bot \in S_1^{\mathcal{T}}(\top)$.

$n \to n+1$: Let $1 \leq i, j \leq n$,

(1.) $C_{n+1} = D_j$, and $D_j \in S_n(\top)$, then $D_{n+1} \in S_{n+1}(\top)$.

(2.) $C_{n+1} = D_i \sqcap D_j$, and $D_i, D_j \in S_n(\top)$, then $D_{n+1} \in S_{n+1}(\top)$.

(3.) $C_k = D_j$, $1 \leq k \neq j < n$, $D_k = \forall r.C_s$, $k \leq s \leq n$, and $C_i \in S_n(\top)$, and $\forall r.C_i \sqsubseteq D_n \in \mathcal{T}$, then $D_{n+1} \in S_{n+1}(\top)$.

Hence, if $D_{n+1} = \bot$, then $\bot \in S^{\mathcal{T}}_{n+1}(\top)$.

The argumentation for the opposite direction is analogue to [Bra04c]. □

Lemma 4.25.
Let B be a finite set of Boolean operators s.t. $\mathsf{I} \subseteq [B]$, then $\mathrm{TSAT}_{\exists}(B)$ is P-hard.

Proof. A reduction from $\mathrm{SUBS}_{\exists}(\mathsf{I}_2)$ to $\overline{\mathrm{TSAT}_{\exists}(\mathsf{I})}$, justified by $(\mathcal{T}, A, B) \in \mathrm{SUBS}_{\exists}(\mathsf{I}_2)$ iff $(\mathcal{T} \cup \{\top \equiv A, B \equiv \bot\}) \notin \mathrm{TSAT}_{\exists}(\mathsf{I})$, provides P-hardness of $\mathrm{TSAT}_{\exists}(\mathsf{I})$ as $\mathrm{SUBS}_{\exists}(D)$ with $\mathsf{I}_2 \subseteq [D]$ is P-hard. The proof for the P-hardness can be found in Lemma 4.43. □

Theorem 4.26 (Results for $\star\mathrm{SAT}^{\mathrm{ind}}_{\emptyset}(B)$ without Quantifiers).
Let B be a finite set of Boolean operators.

(1.) If $\mathsf{S}_{11} \subseteq [B]$, $\mathsf{L}_3 \subseteq [B]$, or $\mathsf{L}_0 \subseteq [B]$, then $\star\mathrm{SAT}^{\mathrm{ind}}_{\emptyset}(B)$ is NP-complete w.r.t. $\leq^{\log}_{\mathrm{m}}$.

(2.) If $[B] \in \{\mathsf{E}_0, \mathsf{E}, \mathsf{V}_0, \mathsf{V}\}$, then $\star\mathrm{SAT}^{\mathrm{ind}}_{\emptyset}(B)$ is P-complete w.r.t. $\leq^{\log}_{\mathrm{m}}$.

(3.) If $[B] \in \{\mathsf{I}_0, \mathsf{I}, \mathsf{N}_2, \mathsf{N}\}$, then $\star\mathrm{SAT}^{\mathrm{ind}}_{\emptyset}(B)$ is $\mathsf{NLOGSPACE}$-complete w.r.t. $\leq^{\log}_{\mathrm{m}}$.

(4.) Otherwise (if $[B] \subseteq \mathsf{R}_1$), then $\star\mathrm{SAT}^{\mathrm{ind}}_{\emptyset}(B)$ is trivial.

Proof. NP-hardness for (1.) follows from the respective $\mathrm{TSAT}_{\emptyset}(B)$ results in Lemmas 4.14 and 4.15 in combination with Lemma 4.5 for the lower bound. The membership in NP is shown in Lemma 4.27.

The lower bounds for (2.) result from $\mathrm{TSAT}_{\emptyset}(\sqcap, \top, \bot)$ and $\mathrm{TSAT}_{\emptyset}(\sqcup, \top, \bot)$ shown in Lemmas 4.16 and 4.17 in combination with Lemma 4.5 while the upper bound applies due to $\mathrm{OCSAT}_{\exists}(\sqcap, \top, \bot)$ which is proven to be in P in Lemma 4.33.

The lower bound of (3.) is proven in Lemma 4.29. The upper bound follows from Lemmas 4.28 and 4.30.

(4.) is due to Lemma 4.8. □

Lemma 4.27.
Let B be a finite set of Boolean operators s.t. $[B] \subseteq \mathsf{BF}$. Then $\mathrm{OCSAT}_{\emptyset}(B)$ is in NP.

Proof. We will reduce $\mathrm{OCSAT}_{\emptyset}(B)$ to SAT, the satisfiability problem for propositional formulae. Due to Lemma 4.4, we can assume that $B = \{\sqcap, \neg\}$. Let $\big((\mathcal{T}, \mathcal{A}), C\big)$ be an instance of $\mathrm{OCSAT}_{\emptyset}(B)$. Since $\mathcal{ALC}_{\emptyset}(B)$ does not have quantifiers, $\mathcal{T}$ only makes propositional statements about all individuals and cannot enforce more individuals than those in $\mathcal{A}$. Let $D_j \sqsubseteq E_j$, $j = 1, \ldots, n$, be the axioms in $\mathcal{T}$ and $a_1, \ldots, a_m$ the individuals occurring in $\mathcal{A}$. We introduce a fresh atomic proposition p^i_A for each $i = 0, \ldots, m$ and each atomic concept A occurring in $(\mathcal{T}, \mathcal{A})$. Every p^i_A expresses that A has as instance either the individual a_i (if $i \geq 1$) or is an instance of C (if $i = 0$). Although C may have several instances, the absence of quantifiers allows us to identify them with a single individual.

For $i = 0, \ldots, m$, we define a function f^i that maps from arbitrary concepts occurring in $((\mathcal{T}, \mathcal{A}), C)$ to propositional formulae as follows:

$$f^i(A) = p_A^i \quad \text{for atomic concepts } A, \qquad f^i(\top) = 1, \qquad f^i(\bot) = 0,$$
$$f^i(\neg A) = \overline{f^i(A)}, \qquad f^i(A_1 \sqcap A_2) = f^i(A_1) \wedge f^i(A_2).$$

We express the instance $((\mathcal{T}, \mathcal{A}), C)$ using the following propositional formulae:

$$\varphi_{\mathcal{T}} = \bigwedge_{i=0}^{m} \bigwedge_{j=1}^{n} (f^i(D_j) \rightarrow f^i(E_j)) \qquad \varphi_{\mathcal{A}} = \bigwedge_{i=1}^{m} \bigwedge_{D(a_i) \in \mathcal{A}} f^i(D),$$
$$\varphi_C = f^0(C), \qquad \varphi_{\mathcal{T},\mathcal{A},C} = \varphi_{\mathcal{T}} \wedge \varphi_{\mathcal{A}} \wedge \varphi_C.$$

We will now show that $((\mathcal{T}, \mathcal{A}), C) \in \mathrm{OCSAT}_{\emptyset}(B)$ if and only if $\varphi_{\mathcal{T},\mathcal{A},C} \in \mathrm{SAT}$.

For "$\Rightarrow$", assume that $((\mathcal{T}, \mathcal{A}), C) \in \mathrm{OCSAT}_{\emptyset}(B)$. Then there is an interpretation $\mathcal{I}$ such that $\mathcal{I} \models (\mathcal{T}, \mathcal{A})$ and $C^{\mathcal{I}} \neq \emptyset$. Fix individuals $x_0, \ldots, x_m \in \Delta^{\mathcal{I}}$ such that $x_0 \in C^{\mathcal{I}}$ and $x_i = a_i^{\mathcal{I}}$ for $i = 1, \ldots, m$. Now construct a propositional assignment β such that $\beta(p_A^i) = 1$ if and only if $x_i \in A^{\mathcal{I}}$. It is straightforward to show by induction on X that for every, possibly complex, concept X occurring in $((\mathcal{T}, \mathcal{A}), C)$ and each $i = 0, \ldots, m$, it holds that $\beta(f^i(X)) = 1$ if and only if $x_i \in X^{\mathcal{I}}$. Using this equivalence, we show that $\beta(\varphi_{\mathcal{T},\mathcal{A},C}) = 1$.

- $\beta(\varphi_{\mathcal{T}}) = 1$ because, for every i, j, the axiom $D_j \sqsubseteq E_j$ in $\mathcal{T}$ ensures that $x_i \in D_j^{\mathcal{I}}$ implies $x_i \in E_j^{\mathcal{I}}$.
- $\beta(\varphi_{\mathcal{A}}) = 1$ because every $D(a_i)$ in $\mathcal{A}$ means that $x_i \in D^{\mathcal{I}}$.
- $\beta(\varphi_C) = 1$ because $x_0 \in C^{\mathcal{I}}$.

For "$\Leftarrow$", assume that $\varphi_{\mathcal{T},\mathcal{A},C} \in \mathrm{SAT}$. Then there is an assignment β under which all three conjuncts $\varphi_{\mathcal{T}}, \varphi_{\mathcal{A}}, \varphi_C$ evaluate to 1. We construct an interpretation $\mathcal{I}$ from β as follows. $\Delta^{\mathcal{I}} = \{x_0, \ldots, x_m\}$; for every $i = 0, \ldots, m$, every individual a in $\mathcal{A}$ and every atomic concept A in $((\mathcal{T}, \mathcal{A}), C)$: $a_i^{\mathcal{I}} = x_i$ and $x_i \in A^{\mathcal{I}}$ if and only if $\beta(p_A^i) = 1$. As above, it is straightforward to show that $\beta(f^i(X)) = 1$ if and only if $x_i \in X^{\mathcal{I}}$, for every X in $((\mathcal{T}, \mathcal{A}), C)$ and every $i = 0, \ldots, m$. Using this equivalence, we show that $\mathcal{I} \models (\mathcal{T}, \mathcal{A})$ and $C^{\mathcal{I}} \neq \emptyset$.

- $\mathcal{I} \models D_j \sqsubseteq E_j$, $j = 1, \ldots, n$ because, for every $i = 0, \ldots, m$, the conjuncts in $\varphi_{\mathcal{T}}$ ensure that $\beta(f^i(D_j)) = 1$ implies that $\beta(f^i(E_j)) = 1$, and therefore $x_i \in D_j^{\mathcal{I}}$ implies $x_i \in E_j^{\mathcal{I}}$.
- $\mathcal{I} \models D(a_i), D(a_i) \in \mathcal{A}$, because the conjuncts in $\varphi_{\mathcal{A}}$ ensure that $x_i \in D^{\mathcal{I}}$.
- $C^{\mathcal{I}} \neq \emptyset$ because φ_C ensures that $x_0 \in C^{\mathcal{I}}$. □

Lemma 4.28.
Let B be a finite set of Boolean operators s.t. $[B] \subseteq \mathsf{N}$, *then* $\mathrm{TCSAT}_\emptyset(B)$ *is in* NLOGSPACE.

Proof. Here we will provide a nondeterministic algorithm for $\mathrm{TSAT}_\emptyset(B)$ that runs in logarithmic space, which can be generalized to also work with $\mathrm{TCSAT}_\emptyset(B)$ instances $(\mathcal{T}, C)$ by adding an axiom $\top \sqsubseteq C$ to the input terminology (in our case this maintains satisfiability because we can only talk about one individual). The algorithm consists of a search for cycles with contradictory atomic concepts in the (directed) implication graph $G_\mathcal{T}$ which is induced by $\mathcal{T}$.

W.l.o.g. assume $\mathcal{T}$ to be normalized in a way that all blocks of leading negations $\neg$ in front of concepts are replaced by one negation if the number was odd, and completely removed otherwise. Thus $\mathcal{T}$ consists only of axioms $C \sqsubseteq D$, where C, D are atomic concepts, constants, or its negations. The before mentioned implication graph $G_\mathcal{T} = (V, E)$ is constructed from $\mathcal{T}$ as follows:

$$V \underset{\text{def}}{=} \{v_A, v_{\neg A} \mid A \text{ is an atomic concept in } \mathcal{T}\} \cup \{v_\top, v_\bot\},$$
$$E \underset{\text{def}}{=} \{(v_C, v_D) \mid C \sqsubseteq D \in \mathcal{T}\} \cup$$
$$\cup \{(v_\bot, v_A), (v_A, v_\top) \mid A \text{ is an atomic concept in } \mathcal{T}\} \cup \{(v_\bot, v_\top)\}.$$

Now we claim that $\mathcal{T} \in \mathrm{TSAT}_\emptyset(B)$ iff $G_\mathcal{T}$ does not contain a cycle that contains both nodes $v_A, v_{\neg A}$ for some $A \in \mathsf{N_C} \cup \{\top, \bot\}$.

"$\Rightarrow$": Let $\mathcal{T} \in \mathrm{TSAT}_\emptyset(B)$ witnessed by the interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$. W.l.o.g. assume $\Delta^\mathcal{I} = \{x\}$ by the same argumentation as in Lemma 4.27. Then it holds that $\mathcal{I} \models \mathcal{T}$. Hence each axiom is satisfied, and with that there is no axiom $C \sqsubseteq D$ s.t. $x \in C^\mathcal{I}$ but $x \notin D^\mathcal{I}$. Now assume that we have a cyclic path π containing the nodes v_A and $v_{\neg A}$. If $x \in A^\mathcal{I}$ then for all successor nodes $v_{A_1}, v_{A_2}, \ldots$ of v_A on π it must hold that $x \in A_i^\mathcal{I}$ for $i = 1, 2, \ldots$, which is a contradiction to $\neg A$ for which $v_{\neg A}$ is a successor of v_A. If $x \notin A^\mathcal{I}$ then $x \in (\neg A)^\mathcal{I}$. Thus for all axioms $A_1, A_2, \ldots$ with $v_{A_1}, v_{A_2}, \ldots$ being successor nodes of $v_{\neg A}$ it must hold that $x \in (A_i)^\mathcal{I}$. In particular this must hold for v_A which is a contradiction to $x \notin A^\mathcal{I}$.

"$\Leftarrow$": Assume that for each atomic concept A (including $\top$ and $\bot$) there is no cyclic path containing v_A and $v_{\neg A}$. In the following we will construct an interpretation $\mathcal{I} = (\{x\}, \cdot^\mathcal{I})$ that satisfies $\mathcal{T}$. For each concept $A \in \mathsf{Con}(\{\top, \bot, \neg\})$ s.t. $\top \sqsubseteq^* A$, add x to $A^\mathcal{I}$. As we have $(v_A, v_{\sim A}) \notin E^*$ (where E^* is the transitive closure of E, and $\sim A = \neg B$ if $A = B$ and $\sim A = B$ if $A = \neg B$) it must hold that also $A \not\sqsubseteq^* \sim A$ and thus $\mathcal{I} \models \mathcal{T}$, as all remaining concepts are not enforced to be true. This completes the proof of the claim.

The NLOGSPACE-algorithm just checks for each concept A that there is no cycle from v_A containing $v_{\sim A}$. □

Lemma 4.29.
Let B be a finite set of Boolean operators s.t. $\mathsf{I}_0 \subseteq [B]$, *then* $\mathrm{TCSAT}_\emptyset(B)$ *is* NLOGSPACE*-hard.*

Proof. This result follows from Lemma 4.18 in combination with Lemma 4.5. □

Lemma 4.30.
Let B be a finite set of Boolean operators s.t. $[B] \subseteq \mathsf{N}$, then $\mathrm{OCSAT}_\emptyset(B)$ is in NLOGSPACE.

Proof. Let B be a finite set of Boolean operators s.t. $\mathsf{N} = [B]$. The algorithm first checks whether the given TBox is solely satisfiable. Afterwards we need to ensure the given ABox is consistent together with the TBox. Therefore observe for an ABox $\mathcal{A}$ the following property holds: $(\mathcal{A}, \mathcal{T}, C) \in \mathrm{OCSAT}_\emptyset(B)$ iff $(\mathcal{A} \cup \{R(a,b)\}, \mathcal{T}, C) \in \mathrm{OCSAT}_\emptyset(B)$ for new individuals a, b and a role R, as role assertions cannot affect the satisfiability of an instance if quantifiers are not allowed. The algorithm now tests consecutively for each individual $a \in \mathcal{A}$ if $(\mathcal{T}^a, C) \in \mathrm{TCSAT}_\emptyset(B)$, where $\mathcal{T}^a = \mathcal{T} \cup \{\top \sqsubseteq D \mid D(a) \in \mathcal{A}\}$.

Now it holds that $(\mathcal{A}, \mathcal{T}, C) \in \mathrm{OCSAT}_\emptyset(B)$ iff $(\mathcal{T}^a, C) \in \mathrm{TCSAT}_\emptyset(B)$ for all individuals $a \in \mathcal{A}$ and $(\mathcal{T}, C) \in \mathrm{TCSAT}_\emptyset(B)$.

If $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ is an interpretation with $\mathcal{I} \models (\mathcal{T}, \mathcal{A})$ and $C^\mathcal{I} \neq \emptyset$, then for the terminologies $\mathcal{T}^a$ for each individual $a \in \mathcal{A}$ it holds that $\mathcal{I}|_a \models \mathcal{T}^a$, where $\mathcal{I}|_a$ is the restriction of $\mathcal{I}$ to the individual a. For the opposite direction to be considered, we have interpretations $\mathcal{I}^a = (\Delta^{\mathcal{I}^a}, \cdot^{\mathcal{I}^a})$ s.t. $\mathcal{I}^a \models \mathcal{T}^a$ and $C^{\mathcal{I}^a} \neq \emptyset$. W.l.o.g. assume $\Delta^{\mathcal{I}^a} = \{a\}$, then an easy inductive argument proves that $\mathcal{I} \models (\mathcal{T}, \mathcal{A})$ and $C^\mathcal{I} \neq \emptyset$ for $\mathcal{I} = (\bigcup_{a \in \mathcal{A}} \Delta^{\mathcal{I}^a}, \cdot^{\bigcup_{a \in \mathcal{A}} \mathcal{I}^a})$.

This connection between $\mathrm{OCSAT}_\emptyset(B)$ and $\mathrm{TCSAT}_\emptyset(B)$ is possible as we can assume different individuals to be distinct. As besides of that point we cannot speak about more than one individual for a given TBox which is restricted to a single individual a, and therefore we may assume the concept D to hold (and consider also the axiom $\top \sqsubseteq D$) if $D(a) \in \mathcal{A}$ for $\mathcal{T}^a$. □

Theorem 4.31 (Results for $\star\mathrm{SAT}^{\mathrm{ind}}_\mathcal{Q}(B)$ with One Quantifier).
Let B be a finite set of Boolean operators, and $\mathcal{Q} \in \{\forall, \exists\}$ be a quantifier.

(1.) If $\mathsf{S}_{11} \subseteq [B]$, $\mathsf{N}_2 \subseteq [B]$, or $\mathsf{L}_0 \subseteq [B]$ then $\star\mathrm{SAT}^{\mathrm{ind}}_\mathcal{Q}(B)$ is EXP*-complete w.r.t. $\leq^{\log}_{\mathrm{m}}$.*

(2.) If $\mathsf{I}_0 \subseteq [B] \subseteq \mathsf{V}$, then $\mathrm{TCSAT}_\exists(B)$ and $\star\mathrm{SAT}^{\mathrm{ind}}_\forall(B)$ are P*-complete[1] w.r.t. $\leq^{\log}_{\mathrm{m}}$.*

(3.) If $[B] \in \{\mathsf{E}_0, \mathsf{E}\}$, then $\star\mathrm{SAT}^{\mathrm{ind}}_\forall(B)$ is EXP*-complete w.r.t. $\leq^{\log}_{\mathrm{m}}$, and $\star\mathrm{SAT}^{\mathrm{ind}}_\exists(B)$ is* P*-complete w.r.t. $\leq^{\log}_{\mathrm{m}}$.*

(4.) If $[B] \subseteq \mathsf{R}_1$, then $\star\mathrm{SAT}^{\mathrm{ind}}_\mathcal{Q}(B)$ is trivial.

Proof. For (1.) combine the EXP-completeness of $\mathrm{TSAT}_\mathcal{Q}(\mathsf{M})$ from Lemma 4.21 with the usual $\top$-knack known from Lemma 4.5.

The lower bound for N_2 is due to Lemma 4.23 to state a reduction from $\mathrm{TSAT}_\mathcal{Q}(\mathsf{L})$ with Lemma 4.5 to $\mathrm{TCSAT}_\mathcal{Q}(\mathsf{L}_0)$ for $\mathcal{Q} \in \{\exists, \forall\}$.

The EXP-completeness in case (3.) follows from Lemma 4.32. For the P-complete cases in (2.) and (3.) the results are organized as follows:

- the P-hardness of these cases results from $\mathrm{TSAT}_\mathcal{Q}(\{\top, \bot\})$ in Lemma 4.25 in combination with Lemma 4.5,

[1] $\mathrm{OSAT}_\exists(B)$ and $\mathrm{OCSAT}_\exists(B)$ are P-hard w.r.t. $\leq^{\log}_{\mathrm{m}}$ for $[B] \in \{\mathsf{V}_0, \mathsf{V}\}$ and in EXP.

- the membership in P of $\mathrm{TCSAT}_\forall(\{\sqcup,\top,\bot\})$ follows by $\mathrm{OCSAT}_\forall(\{\sqcup,\top,\bot\}) \in \mathsf{P}$ proven in Lemma 4.34,
- the membership in P of $\mathrm{TCSAT}_\exists(\{\sqcup,\top,\bot\})$ follows by $\mathrm{TSAT}_\exists(\{\sqcup,\top,\bot\})$ in combination with Lemma 4.3,
- membership in P of $\mathrm{TCSAT}_\exists(\{\sqcap,\top,\bot\})$ follows by $\mathrm{OCSAT}_\exists(\{\sqcap,\top,\bot\})$ and its P-membership shown in Lemma 4.33.

(4.) is due to Lemma 4.8. □

Theorem 4.31 shows one reason why the logics in the $\mathcal{EL}$ family have been much more successful as "small" logics with efficient reasoning methods than the $\mathcal{FL}$ family: the combination of the $\forall$ with conjunction is intractable, while $\exists$ and conjunction are still in polynomial time. Again, separating either conjunction and disjunction, or the constants is crucial for tractability.

Lemma 4.32.
Let B be a finite set of Boolean operators s.t. $\mathsf{E}_0 \subseteq [B]$, *then* $\mathrm{TCSAT}_\forall(B)$ *is* EXP*-hard.*

Proof. As a result from [BBL05a, Hof05] the subsumption problem w.r.t. a TBox for the logic $\mathcal{FL}_0$ (the description logic with $\forall$ and $\sqcap$ as allowed operators) is EXP-complete. For this lemma we will reduce from this problem in $\mathcal{FL}_0$. Observe that the following holds

$$\begin{aligned}
&(\mathcal{T}, C, D) \in \mathcal{FL}_0\text{-SUBS}\\
&\quad\text{iff}\quad \forall \mathcal{I} : \mathcal{I} \models \mathcal{T} \text{ it holds } \mathcal{I} \models C \sqsubseteq D\\
&\quad\text{iff}\quad \text{not}(\exists \mathcal{I} : \mathcal{I} \models \mathcal{T} \text{ and } (C \sqcap \neg D)^{\mathcal{I}} \neq \emptyset)\\
&\quad\text{iff}\quad \text{not}(\exists \mathcal{I} : \mathcal{I} \models \mathcal{T} \cup \{D \sqcap D' \sqsubseteq \bot\} \text{ and } (C \sqcap D')^{\mathcal{I}} \neq \emptyset)\\
&\quad\text{iff}\quad (\mathcal{T} \cup \{D \sqcap D' \sqsubseteq \bot\}, C \sqcap D') \notin \mathrm{TCSAT}_\forall(B)
\end{aligned}$$

Lemma 4.33.
Let B be a finite set of Boolean operators s.t. $[B] \subseteq \mathsf{E}$, *then* $\mathrm{OCSAT}_\exists(B)$ *is in* P.

Proof. To provide an algorithm running in polynomial time, we will reduce the given problem to the complement of the subsumption problem for the logic $\mathcal{EL}^{++}$, which is known to be P-complete by [BBL08].

The reduction works as follows:

$$\begin{aligned}
((\mathcal{T}, \mathcal{A}), C) \in \mathrm{OCSAT}_\exists(B) \quad\text{iff}\quad & \exists \mathcal{I} : \mathcal{I} \models \mathcal{T} \text{ and } \mathcal{C}_{\mathcal{A}}^{\mathcal{I}} \neq \emptyset \text{ and } C^{\mathcal{I}} \neq \emptyset\\
\text{iff}\quad & \exists \mathcal{I}' : \mathcal{I}' \models \mathcal{T} \cup \{\top \sqsubseteq \exists R.\mathcal{C}_{\mathcal{A}}\} \text{ and } C^{\mathcal{I}'} \neq \emptyset\\
\text{iff}\quad & \mathcal{T} \cup \{\top \sqsubseteq \exists R.\mathcal{C}_{\mathcal{A}}\} \not\models C \sqsubseteq \bot\\
\text{iff}\quad & (\mathcal{T} \cup \{\top \sqsubseteq \exists R.\mathcal{C}_{\mathcal{A}}\}, C, \bot) \notin \mathcal{EL}^{++}\text{-SUBS},
\end{aligned}$$

where $\mathscr{T}$ is a TBox, $\mathscr{A}$ is an ABox, R is a fresh role, and

$$\mathscr{C}_{\mathscr{A}} \underset{\text{def}}{=} \bigsqcap_{C(a)\in\mathscr{A}} \exists u.(\{a\}\sqcap C)\sqcap \bigsqcap_{r(a,b)\in\mathscr{A}} \exists u.(\{a\}\sqcap\exists r.\{b\})$$

is the concept constructed as in [BBL05b] from the ABox $\mathscr{A}$, where u is a fresh role name, and $\{a\}$ and $\{b\}$ denote nominals corresponding to the ABox individuals a and b.□

Lemma 4.34.
Let B be a finite set of Boolean operators s.t. $[B]\subseteq \mathsf{V}$, then $\mathrm{OCSAT}_{\forall}(B)$ is in P.

Proof. Here, a reduction to the dual problem $\mathrm{OCSAT}_{\exists}(\mathsf{E})$ suffices (see Lemma 4.33 for its membership in P). Consider an ontology $(\mathscr{T},\mathscr{A})$ where $\mathscr{T}$ is a TBox and $\mathscr{A}$ an ABox, and a concept C as the given instance of $\mathrm{OCSAT}_{\forall}(B)$. w.l.o.g. assume C to be atomic. Now first construct the new terminology $\mathscr{T}'$ similarly to Lemma 4.17. Then add for each $A\in\mathsf{N_C}$ and hence each A' the GCIs $A\sqcap A'\sqsubseteq\bot$ to ensure they are disjoint. Denote this change by the terminology $\mathscr{T}''$. Then it holds that $((\mathscr{T},\mathscr{A}),C)\in\mathrm{OCSAT}_{\forall}(B)$ iff $((\mathscr{T}'',\mathscr{A}),C')\in\mathrm{OCSAT}_{\exists}(\mathsf{E})$. □

4.1.3 Conclusion

Having almost completely classified all operator and quantifier fragments, Table 4.1 shows the dichotomy for $^{\star}\mathrm{SAT}_{\exists\forall}(B)$, the trichotomy for $^{\star}\mathrm{SAT}_{\mathscr{Q}}(B)$ and $|\mathscr{Q}|=1$, and the quartering for $^{\star}\mathrm{SAT}_{\emptyset}(B)$.

We have shown that the problem $\mathrm{TSAT}_{\mathscr{Q}}(B)$ is strictly less expressive than the problems $^{\star}\mathrm{SAT}^{\mathrm{ind}}_{\mathscr{Q}}(B)$ as a single terminology cannot enforce single individuals and therefore is not able to simulate the constant $\top$ in contrast to $^{\star}\mathrm{SAT}^{\mathrm{ind}}_{\mathscr{Q}}(B)$. Interestingly, the problems become tractable if either disjunction or conjunction is present and only one quantifier is allowed. Without any quantifiers the problems also become tractable for all unary operators, namely the ones in the clone N.

Further research in this area would encompass closing the gaps for $\mathrm{OCSAT}_{\exists}(\mathsf{V})$ and $\mathrm{OSAT}_{\exists}(\mathsf{V})$, and also classifying fragments using unqualified restrictions of the form $\exists R.\top$. These kind of fragments are only able to speak about the existence of a role-edge and not about properties that hold there. Furthermore, a classification of the parameterized complexity of the intractable cases would be of great interest.

4.2 Subsumption

Given two concepts and a terminology, asking the question whether one concept *subsumes* the other with respect to the terminology, is the pendant to the propositional implication problem and therewith a very important problem in the area of description logics. Nardi and Brachman describe the problem as the "key inference" [NB03].

In Figures 4.1 and 4.2 on page 105 and 105 it is depicted how the following results arrange in Post's lattice.

$\mathrm{TSAT}_{\mathscr{Q}}(B)$	I	V	E	N/N_2	M	L_3 to BF	else
$\mathscr{Q}=\emptyset$	NL	P		NL	NP		trivial
$\lvert\mathscr{Q}\rvert=1$	P			EXP			trivial
$\mathscr{Q}=\{\exists,\forall\}$	EXP						trivial

$\star\mathrm{SAT}^{\mathrm{ind}}_{\mathscr{Q}}(B)$	I/I_0	V/V_0	E/E_0	N/N_2	S_{11} to M	$\mathsf{L}_3/\mathsf{L}_0$ to BF	else
$\mathscr{Q}=\emptyset$	NL	P		NL	NP		trivial
$\mathscr{Q}=\{\exists\}$	P	$\mathsf{P}^{\S}$	P	EXP			trivial
$\mathscr{Q}=\{\forall\}$	P		EXP				trivial
$\mathscr{Q}=\{\exists,\forall\}$	EXP						trivial

Table 4.1: Complexity overview for all Boolean function and quantifier fragments. All results are completeness results for the given complexity class, except for the case marked §: here, OCSAT and OSAT are in EXP and P-hard. NL abbreviates NLOGSPACE.

In [BMTV09b] Beyersdorff et al. classify the propositional implication problem with respect to all fragments parameterized by all Boolean clones. As the subsumption problem is closely related to the implication problem this theorem will be helpful by stating upper and lower bounds.

Theorem 4.35 ([BMTV09b]).
Let B be a finite set of Boolean operators.

(1.) If $C \subseteq [B]$ for $C \in \{\mathsf{S}_{00}, \mathsf{D}_2, \mathsf{S}_{10}\}$, then $\mathrm{IMP}(B)$ *is* coNP*-complete w.r.t.* $\leq_m^{\mathsf{AC}^0}$[2].

(2.) If $\mathsf{L}_2 \subseteq [B] \subseteq \mathsf{L}$, then $\mathrm{IMP}(B)$ *is* $\oplus$LOGSPACE*-complete w.r.t.* $\leq_m^{\mathsf{AC}^0}$.

(3.) If $\mathsf{N}_2 \subseteq [B] \subseteq \mathsf{N}$, then $\mathrm{IMP}(B)$ *is in* $\mathsf{AC}^0[2]$.

(4.) Otherwise $\mathrm{IMP}(B) \in \mathsf{AC}^0$.

The following two lemmata translate Lemma 4.6 and Lemma 4.2 to the subsumption problem and follows the same proof technique.

Lemma 4.36.
Let B be a finite set of Boolean operators and $\mathscr{Q} \subseteq \{\forall,\exists\}$. Then

$$\mathrm{SUBS}_{\mathscr{Q}}(B) \leq_{\mathbf{m}}^{\mathbf{log}} \mathrm{SUBS}_{\mathbf{dual}(\mathscr{Q})}(\mathbf{dual}(B)).$$

[2] A language A is AC^0 many-one reducible to a language B ($A \leq_m^{\mathsf{AC}^0} B$) if there exists a function f computable by a logtime-uniform AC^0-circuit familiy such that $x \in A$ iff $f(x) \in B$.

Lemma 4.37.
Let B be a finite set of Boolean operators s.t. $\mathsf{N}_2 \subseteq [B]$ and $\mathcal{Q} \subseteq \{\exists, \forall\}$. Then it holds that $\mathrm{SUBS}_{\mathcal{Q}}(B) \equiv_{\mathrm{m}}^{\log} \mathrm{SUBS}_{\mathcal{Q}}(B \cup \{\top, \bot\})$.

Using Lemma 4.2 in [BMTV09b] we can easily obtain the ability to express the constant $\top$ whenever we have access to conjunctions, and the constant $\bot$ whenever we are able to use disjunctions.

Lemma 4.38.
Let B be a finite set of Boolean operators and $\mathcal{Q} \subseteq \{\forall, \exists\}$.

(1.) If $\mathsf{E}_0 \subseteq [B]$, then $\mathrm{SUBS}_{\mathcal{Q}}(B) \equiv_{\mathrm{m}}^{\log} \mathrm{SUBS}_{\mathcal{Q}}(B \cup \{\top\})$.

(2.) If $\mathsf{V}_0 \subseteq [B]$, then $\mathrm{SUBS}_{\mathcal{Q}}(B) \equiv_{\mathrm{m}}^{\log} \mathrm{SUBS}_{\mathcal{Q}}(B \cup \{\bot\})$.

The connection of subsumption to terminology satisfiability and propositional implication is crucial for stating upper and lower bound results. The next lemma connects the problem to the mentioned problems TCSAT from Section 4.1 and also to the propositional implication problem.

Lemma 4.39.
Let B be a finite set of Boolean operators and $\mathcal{Q} \subseteq \{\forall, \exists\}$ be a set of quantifiers. Then

(1.) $\mathrm{IMP}(B) \leq_{\mathrm{m}}^{\log} \mathrm{SUBS}_{\emptyset}(B)$.

(2.) $\mathrm{SUBS}_{\mathcal{Q}}(B) \leq_{\mathrm{m}}^{\log} \overline{\mathrm{TCSAT}_{\mathcal{Q}}(B \cup \{\nrightarrow\})}$.

(3.) $\overline{\mathrm{TCSAT}_{\mathcal{Q}}(B)} \leq_{\mathrm{m}}^{\log} \mathrm{SUBS}_{\mathcal{Q}}(B \cup \{\bot\})$.

Proof. (1.) Holds due to $(\varphi, \psi) \in \mathrm{IMP}(B)$ iff $(C_\varphi, C_\psi, \emptyset) \in \mathrm{SUBS}_{\emptyset}(B)$, for concept descriptions $C_\varphi = f(\varphi), C_\psi = f(\psi)$ with f mapping propositional formulae to concept descriptions via

$$f(\top) = \top, \qquad f(\bot) = \bot,$$
$$f(x) = C_x, \text{ for variable } x,$$
$$f(g(C_1, \ldots, C_n)) = \circ_g(f(C_1), \ldots, f(C_n))$$

where g is an n-ary Boolean function and $\circ_g$ is the corresponding operator.

(2.) $(C, D, \mathcal{T}) \in \mathrm{SUBS}_{\mathcal{Q}}(B)$ iff $(\mathcal{T}, C \nrightarrow D) \in \overline{\mathrm{TCSAT}_{\mathcal{Q}}(B \cup \{\nrightarrow\})}$. [BCM+03].

(3.) $(\mathcal{T}, C) \in \overline{\mathrm{TCSAT}_{\mathcal{Q}}(B)}$ iff $(C, \bot, \mathcal{T}) \in \mathrm{SUBS}_{\mathcal{Q}}(B \cup \{\bot\})$. [BCM+03]. □

We will start with the subsumption problem using no quantifiers and will show that the problem either is coNP-, P-, NLOGSPACE-complete, or is $\oplus$LOGSPACE-hard.

Theorem 4.40 ($\mathcal{Q} = \emptyset$).
Let B be a finite set of Boolean operators.

(1.) If $X \subseteq [B]$ *for* $X \in \{\mathsf{L}_0, \mathsf{L}_1, \mathsf{L}_3, \mathsf{S}_{10}, \mathsf{S}_{00}, \mathsf{D}_2\}$, *then* $\mathrm{SUBS}_\emptyset(B)$ *is* coNP*-complete.*

(2.) If $\mathsf{E}_2 \subseteq [B] \subseteq \mathsf{E}$ *or* $\mathsf{V}_2 \subseteq [B] \subseteq \mathsf{V}$, *then* $\mathrm{SUBS}_\emptyset(B)$ *is* P*-complete.*

(3.) If $[B] = \mathsf{L}_2$, *then* $\mathrm{SUBS}_\emptyset(B)$ *is* $\oplus$LOGSPACE*-hard.*

(4.) If $\mathsf{I}_2 \subseteq [B] \subseteq \mathsf{N}$, *then* $\mathrm{SUBS}_\emptyset(B)$ *is* NLOGSPACE*-complete.*

All hardness results hold w.r.t. $\leq_{\mathrm{m}}^{\log}$ *reductions.*

Proof. (1.) The reduction from the implication problem IMP(B) in Lemma 4.39 (1.) in combination with Theorem 4.35 proves the coNP lower bounds of $\mathsf{S}_{10}, \mathsf{S}_{00}, \mathsf{D}_2$. The lower bounds for $\mathsf{L}_0 \subseteq [B]$ and $\mathsf{L}_3 \subseteq [B]$ follow from Lemma 4.39 (3.) with $\overline{\mathrm{TCSAT}_\emptyset(B)}$ being coNP-complete which follows from the NP-completeness result of $\mathrm{TCSAT}_\emptyset(B)$ shown in Theorem 4.26 on page 88. Further the lower bound for $\mathsf{L}_1 \subseteq [B]$ follows from the duality of '$\oplus$' and '$\equiv$' and Lemma 4.36 with respect to the case $\mathsf{L}_0 \subseteq [B]$ allowing us to state the reduction

$$\mathrm{SUBS}_\emptyset(\mathsf{L}_0) \leq_{\mathrm{m}}^{\log} \mathrm{SUBS}_{\mathbf{dual}(\emptyset)}(\mathbf{dual}(\mathsf{L}_0)) = \mathrm{SUBS}_\emptyset(\mathsf{L}_1).$$

The upper bound follows from a reduction to $\overline{\mathrm{TCSAT}_\emptyset(\mathsf{BF})}$ by Lemma 4.39 (2.) and $\mathrm{TCSAT}_\emptyset(\mathsf{BF}) \in \mathsf{NP}$ by Theorem 4.26 on page 88.

(2.) The upper bound follows from the memberships in P for $\mathrm{SUBS}_\exists(\mathsf{E})$ and $\mathrm{SUBS}_\forall(\mathsf{V})$ proven in Theorems 4.41 and 4.42.

The lower bound for $[B] = \mathsf{E}_2$ follows from a reduction from HGAP: set $\mathscr{T} = \{u_1 \sqcap \ldots \sqcap u_k \sqsubseteq v \mid (u_1, \ldots, u_k; v) \in E\}$, assume w.l.o.g. the set of source nodes as $S = \{s\}$, then $(G, S, t) \in \mathrm{HGAP}$ iff $(\mathscr{T}, s, t) \in \mathrm{SUBS}_\emptyset(\mathsf{E}_2)$. For the lower bound of V_2 apply Lemma 4.36.

(3.) Follows directly from the reduction from IMP(L_2) by virtue of Theorem 4.35 and Lemma 4.39 (1.).

(4.) For the lower bound we show a reduction from GAP to $\mathrm{SUBS}_\emptyset(\mathsf{I}_2)$. Let $G = (V, E)$ be a undirected graph and $s, t \in V$ be the vertices for the input. Then for $\mathscr{T} := \{(A_u \sqsubseteq A_v) \mid (u, v) \in E\}$ it holds that $(G, s, t) \in \mathrm{GAP}$ iff $(\mathscr{T}, A_s, A_t) \in \mathrm{SUBS}_\emptyset(\mathsf{I}_2)$.

For the upper bound we follow the idea from Lemma 4.28 on page 90. Given the input instance $(\mathscr{T}, C, D)$ we can similarly assume that for each $E \sqsubseteq F \in \mathscr{T}$ it holds that E, F are atomic concepts, or their negations, or constants. Now $(\mathscr{T}, C, D) \in \mathrm{SUBS}_\emptyset(\mathsf{N})$ holds iff for every interpretation $\mathscr{I} = (\Delta^\mathscr{I}, \cdot^\mathscr{I})$ and $x \in \Delta^\mathscr{I}$ it holds that if $x \in C^\mathscr{I}$ then $x \in D^\mathscr{I}$ holds iff for the implication graph $G_\mathscr{T}$ (constructed as in Lemma 4.28) there exists a path from v_C to v_D.

Informally if there is no path from v_C to v_D then D is not implied by C, i.e., it is possible to construct an interpretation for which there exists an individual which is a member of $C^\mathscr{I}$ but not of $D^\mathscr{I}$.

Thus we have provided an coNLOGSPACE-algorithm which first checks accordingly to the algorithm in Lemma 4.28 if there are not any cycles containing contradictory axioms. Then we verify that there is no path from v_C to v_D implying that C is not subsumed by D. □

Using some results from the previous theorem we are now able to classify most fragments of the subsumption problem using only the $\forall$ or $\exists$ quantifier with respect to all possible Boolean clones in the following two theorems.

Theorem 4.41 ($\mathscr{Q} = \{\forall\}$).
Let B be a finite set of Boolean operators.

(1.) If $\mathsf{E}_2 \subseteq [B]$, then $\mathrm{SUBS}_\forall(B)$ is EXP-complete.

(2.) If $\mathsf{N}_2 \subseteq [B]$ or $\mathsf{L}_0 \subseteq [B]$, then $\mathrm{SUBS}_\forall(B)$ is EXP-complete.

(3.) If $\mathsf{L}_1 \subseteq [B]$, then $\mathrm{SUBS}_\forall(B)$ is EXP-complete.

(4.) If $\mathsf{S}_{00} \subseteq [B]$, then $\mathrm{SUBS}_\forall(B)$ is EXP-complete.

(5.) If $\mathsf{D}_2 \subseteq [B] \subseteq \mathsf{D}_1$, then $\mathrm{SUBS}_\forall(B)$ is coNP-hard and in EXP.

(6.) If $[B] \subseteq \mathsf{V}$, then $\mathrm{SUBS}_\forall(B)$ is P-complete.

(7.) If $[B] = \mathsf{L}_2$, then $\mathrm{SUBS}_\forall(B)$ is P-hard and in EXP.

All hardness results hold w.r.t. $\leq_{\mathrm{m}}^{\log}$ reductions.

Proof. (1.) Follows from EXP-hardness of $\mathscr{FL}_0$-SUBS which has been shown in [Hof05, Thm 7.6].

(2.) The lower bound for $\mathsf{N}_2 \subseteq [B]$ is achieved through the reductions

$$\overline{\mathrm{TCSAT}_\forall(\mathsf{N}_2)} \leq_{\mathrm{m}}^{\log} \mathrm{SUBS}_\forall(\mathsf{N}) \equiv_{\mathrm{m}}^{\log} \mathrm{SUBS}_\forall(\mathsf{N}_2),$$

where the first reduction is due to Lemma 4.39 (3.) and the second equivalence holds through Lemma 4.37 which enables us to use always both constants for $\mathsf{N}_2 \subseteq [B]$. The EXP-hardness now follows from $\mathrm{TCSAT}_\forall(\mathsf{N}_2)$ being EXP-complete proven in Theorem 4.31 on page 91.

The EXP-hardness for $\mathsf{L}_0 \subseteq [B]$ follows from Lemma 4.39 (3.) which states the reduction $\overline{\mathrm{TCSAT}_\forall(\mathsf{L}_0)} \leq_{\mathrm{m}}^{\log} \mathrm{SUBS}_\forall(\mathsf{L}_0 \cup \{\bot\})$ where $[\mathsf{L}_0 \cup \{\bot\}] = \mathsf{L}_0$. From Theorem 4.31 on page 91 we know that $\mathrm{TCSAT}_\forall(\mathsf{L}_0)$ is EXP-complete.

(3.) The EXP-hardness follows from the following reduction:

$$\overline{\mathrm{TCSAT}_\exists(\mathsf{L}_0)} \overset{(a)}{\leq_{\mathrm{m}}^{\log}} \mathrm{SUBS}_\exists(\mathsf{L}_0) \overset{(b)}{\leq_{\mathrm{m}}^{\log}} \mathrm{SUBS}_{\mathbf{dual}(\exists)}(\mathbf{dual}(\mathsf{L}_0)) = \mathrm{SUBS}_\forall(\mathsf{L}_1),$$

by virtue of $\mathrm{TCSAT}_\exists(\mathsf{L}_0)$ being EXP-complete (Theorem 4.31 on page 91) for (a), and Lemma 4.36 for (b).

(4.) Follows from Lemma 4.38 and the EXP-hardness of $\mathrm{SUBS}_\forall(\mathsf{M}_0)$ overlaid by Theorem 4.41 (1.), as $\mathsf{M}_0 = [\mathsf{S}_{00} \cup \{\bot\}]$.

(5.) The coNP-hardness follows from $\mathrm{SUBS}_\emptyset(\mathsf{D}_2)$ being coNP-hard shown in Theorem 4.40.

(6.) For the upper bound Lemma 4.36 lets us state the reduction $\mathrm{SUBS}_\forall(\mathsf{V}) \leq_{\mathrm{m}}^{\log} \mathrm{SUBS}_\exists(\mathsf{E})$, where the latter is in P by virtue of Theorem 4.42 (6.).

The lower bound follows again from Lemma 4.36, and the P-hardness of $\mathrm{SUBS}_\exists(\mathsf{I}_2)$ which is proven in Lemma 4.43.

(7.) The P-hardness follows from (6.). □

Theorem 4.42 ($\mathscr{Q} = \{\exists\}$).
Let B be a finite set of Boolean operators and $\mathscr{Q} = \{\exists\}$.

(1.) If $\mathsf{V}_2 \subseteq [B]$, then $\mathrm{SUBS}_{\mathscr{Q}}(B)$ is EXP*-complete.*

(2.) If $\mathsf{N}_2 \subseteq [B]$ or $\mathsf{L}_0 \subseteq [B]$, then $\mathrm{SUBS}_{\mathscr{Q}}(B)$ is EXP*-complete.*

(3.) If $\mathsf{L}_1 \subseteq [B]$, then $\mathrm{SUBS}_{\mathscr{Q}}(B)$ is EXP*-complete.*

(4.) If $\mathsf{S}_{10} \subseteq [B]$, then $\mathrm{SUBS}_{\mathscr{Q}}(B)$ is EXP*-complete.*

(5.) If $\mathsf{D}_2 \subseteq [B] \subseteq \mathsf{D}_1$, then $\mathrm{SUBS}_{\mathscr{Q}}(B)$ is coNP*-hard and in* EXP.

(6.) If $[B] \subseteq \mathsf{E}$, then $\mathrm{SUBS}_{\mathscr{Q}}(B)$ is P*-complete.*

(7.) If $[B] = \mathsf{L}_2$, then $\mathrm{SUBS}_{\mathscr{Q}}(B)$ is P*-hard and in* EXP.

All hardness results hold w.r.t. $\leq_{\mathrm{m}}^{\log}$ reductions.

Proof. (1.)-(3.): For the following reductions showing the needed lower bounds for $\mathsf{V}_2, \mathsf{N}_2, \mathsf{L}_1$, and L_0 we use Theorem 4.41 in combination with the contraposition argument in Lemma 4.36:

$$\mathrm{SUBS}_\forall(\mathsf{E}_2) \leq_{\mathrm{m}}^{\log} \mathrm{SUBS}_\exists(\mathsf{V}_2), \qquad \mathrm{SUBS}_\forall(\mathsf{N}_2) \leq_{\mathrm{m}}^{\log} \mathrm{SUBS}_\exists(\mathsf{N}_2),$$
$$\mathrm{SUBS}_\forall(\mathsf{L}_0) \leq_{\mathrm{m}}^{\log} \mathrm{SUBS}_\exists(\mathsf{L}_1), \text{ and} \qquad \mathrm{SUBS}_\forall(\mathsf{L}_1) \leq_{\mathrm{m}}^{\log} \mathrm{SUBS}_\exists(\mathsf{L}_0).$$

(4.) The needed lower bound follows from Lemma 4.38 whereas the EXP-hardness of $\mathrm{SUBS}_\exists(\mathsf{M}_1)$ overlaid by Theorem 4.42 (1.) as $\mathsf{M}_1 = [\mathsf{S}_{10} \cup \{\top\}]$.
(5.) The coNP lower bound follows from $\mathrm{SUBS}_\emptyset(B)$ shown in Theorem 4.40.
(6.) The upper bound follows from the membership of subsumption for the logic $\mathcal{ELH}$ in P, [Bra04c, Thm. 9]. The lower bound is proven below in Lemma 4.43.
(7.) The lower bound follows from (6.). □

Lemma 4.43.
Let B be a finite set of Boolean operators s.t. $\mathsf{I}_2 \subseteq [B]$. Then $\mathrm{SUBS}_\exists(B)$ is P*-hard.*

Proof. We will reduce the word problem for the Turing machine model that characterizes LOGCFL to $\mathrm{SUBS}_{\exists}(B)$. As in the proof the runtime of the Turing machine is not relevant we achieve instead a P-hardness result (because an NLOGSPACE-Turing machine using a stack with arbitrary runtime leads to the class P [Coo71a]).

Let M be a nondeterministic Turing machine, which has access to a read-only input tape, a read-write work tape and a stack, and whose runtime is bounded by a polynomial in the size of the input. Let M be the 6-tuple $(\Sigma, \Psi, \Gamma, Q, f, q_0)$, where

- Σ is the input alphabet;
- Ψ is the work alphabet containing the empty-cell symbol #;
- Γ is the stack alphabet containing the bottom-of-stack symbol $\square$;
- Q is the set of states;
- $f : Q \times \Sigma \times \Psi \times \Gamma \to Q \times \Psi \times \{-,+\}^2 \times (\Gamma \setminus \{\square\})^*$ is the state transition function which describes a transition where the machine is in a state, reads an input symbol, reads a work symbol and takes a symbol from the stack, and goes into another state, writes a symbol to the work tape, makes a step on each tape (left or right) and possibly adds a sequence of symbols to the stack;
- $q_0 \in Q$ is the initial state.

We assume that each computation of M starts in q_0 with the heads at the left-most position of each tape and with exactly the symbol $\square$ on the stack. W.l.o.g., the machine accepts whenever the stack is empty, regardless of its current state.

Let $x = x_1 \dots x_n$ be an input of M. We consider the configurations that can occur during any computation of $M(x)$ in two versions. A *shallow configuration* of $M(x)$ is a sequence $(p\delta_1 \dots \delta_{k-1} q \delta_k \dots \delta_\ell)$, where

- $p \in \{1, \dots, n\}$ is the current position on the input tape, represented in binary;
- $\ell \in O(\log n)$ is the maximal number of positions on the work tape of M relevant for the computations of $M(x)$;
- $\delta_1, \dots, \delta_\ell$ is the current content of the work tape;
- k is the current position on the work tape;
- q is the current state of M.

The initial shallow configuration $(0q_0\# \dots \#)$ is denoted by S_0. Let $\mathscr{SC}_{M,x}$ be the set of all possible shallow configurations that can occur during any computation of $M(x)$. The cardinality of this set is bounded by a polynomial in n because the number of work-tape cells used is logarithmic in n and the binary counter for the position on the input tape is logarithmic in n.

A *deep configuration* of $M(x)$ is a sequence $(R_1 \dots R_m p \delta_1 \dots \delta_{k-1} q \delta_k \dots \delta_\ell)$, where the R_i are the symbols currently on the stack and the remaining components are as

above. Let $\mathscr{DC}_{M,x}$ be the set of all possible deep configurations that can occur during any computation of $M(x)$. The cardinality of this set can be exponential as soon as Γ has more than two elements besides $\square$. This is not a problem for our reduction, which will only touch shallow configurations.

We now construct an instance of $\mathrm{SUBS}_\exists(\mathsf{I}_2)$ from M and x. We use each shallow configuration $S \in \mathscr{SC}_{M,x}$ as a concept name and each stack symbol as a role name. The TBox $\mathscr{T}_{M,x}$ describes all possible computations of $M(x)$ by containing an axiom for every two deep configurations that the machine can take on before and after some computation step. A deep configuration D is represented by the concept corresponding to D's shallow part, preceded by the sequence of existentially quantified stack symbols corresponding to the stack content in D. The TBox $\mathscr{T}_{M,x}$ is constructed from a set of axioms per entry in f. (We will omit the subscript from now on.) For the instruction

$$(q, \sigma, \delta, R) \mapsto (q', \delta', -, -, R_1 \dots R_k)$$

of f, we add the axioms

$$\begin{aligned} &\exists R.(\mathrm{bin}(p)\delta_0 \dots \delta_{i-1} q \delta \delta_{i+1} \dots \delta_\ell) \sqsubseteq \\ &\exists R_1 \dots \exists R_k.(\mathrm{bin}(p \dot{-} 1)\delta_0 \dots \delta_{i-2} q' \delta_{i-1} \delta' \delta_{i+1} \dots \delta_\ell) \end{aligned} \tag{4.3}$$

for every p with $x_p = \sigma$, every $i = 1, \dots, \ell$, and all $\delta_0, \dots, \delta_{i-1}, \delta_{i+1}, \dots, \delta_\ell$. The expression $p \dot{-} 1$ stands for $p - 1$ if $p \geqslant 2$ and for 1 otherwise, reflecting the assumption that the machine does not move on the input tape on a "go left" instruction if it is already on the left-most input symbol. This behaviour can always be assumed w.l.o.g. In case $k = 0$, the quantifier prefix on the right-hand side is empty. For instructions of f requiring "+" steps on any of the tapes, the construction is analogue. The number of axioms generated by each instruction is bounded by the number of shallow configurations; therefore the overall number of axioms is bounded by a polynomial in $n \cdot |f|$.

Furthermore, we use a fresh concept name B and add an axiom $\mathscr{S} \sqsubseteq B$ for each shallow configuration $\mathscr{S}$. Also we add a single axiom $S \sqsubseteq \exists \square . S_0$ to $\mathscr{T}$. The instance of $\mathrm{SUBS}_\exists(\emptyset)$ is constructed as $(\mathscr{T}, S, B)$. $\mathscr{T}$ can be constructed in logarithmic space. It remains to prove the following claim.

Claim. $M(x)$ has an accepting computation if and only if $S \sqsubseteq_{\mathscr{T}} B$.

Proof of Claim. For the "⇒" direction, we observe that, for each step in the accepting computation, the (arbitrary) concept associated with the pre-configuration is subsumed by the concept associated with the post-configuration. More precisely, if $M(x)$ makes a step

$$(q, \sigma, \delta, R) \mapsto (q', \delta', -, -, R_1 \dots R_k),$$

then its deep configuration before that step has to be

$$S_1 \dots S_j R p \delta_0 \dots \delta_{i-1} q \delta \delta_{i+1} \dots \delta_\ell,$$

for some $S_1, \dots, S_j \in \Gamma$, $\delta_0, \dots, \delta_{i-1}, \delta_{i+1}, \dots, \delta_\ell \in \Psi$ and $p \in \mathbb{N}$., and the deep configuration after that step is

$$S_1 \dots S_j R_1 \dots R_k (p \dot{-} 1) \delta_0 \dots \delta_{i-2} q' \delta_{i-1} \delta' \delta_{i+1} \dots \delta_\ell.$$

The set of axioms constructed in 4.3 ensures that there is an axiom that implies

$$\exists S_1 \dots \exists S_j . \exists R . (\mathrm{bin}(p) \delta_0 \dots \delta_{i-1} q \delta \delta_{i+1} \dots \delta_\ell) \sqsubseteq_{\mathscr{T}}$$
$$\exists S_1 \dots \exists S_j . \exists R_1 \dots R_k . (\mathrm{bin}(p \dot{-} 1) \delta_0 \dots \delta_{i-2} q' \delta_{i-1} \delta' \delta_{i+1} \dots \delta_\ell).$$

Since some computation of $M(x)$ reaches a configuration with an empty stack, we can conclude that some atomic concept corresponding to a shallow configuration $\mathscr{S}$, and therefore also B, subsumes $\exists \square . S_0$ which subsumes S (per definition).

For the "$\Leftarrow$" direction, we assume that $M(x)$ has no accepting computation. This means that, during every computation of $M(x)$, the stack does never become empty. From the set of all computations of $M(x)$, we will show that there exists an interpretation $\mathscr{I}$ that satisfies $\mathscr{T}$, but not $S \sqsubseteq B$; hereby we can conclude $(\mathscr{T}, S, B) \notin \mathrm{SUBS}_{\exists}(\emptyset)$.

Observe that any atomic concept besides S and B in $\mathscr{T}$ correspond to a specific shallow configuration of $M(x)$. Let $T_{M(x)} := (V, E)$ denote the computation tree of $M(x)$. Thus every node $v \in V$ represents a deep configuration of $M(x)$ which will be denoted via C_v. Then for two nodes $u, v \in V$ with $(u, v) \in E$ it holds that $C_u \vdash_M C_v$. In the following we will describe how to construct an interpretation $\mathscr{I}$ from $T_{M(x)}$ which has a witness for $S^{\mathscr{I}} \not\subseteq B^{\mathscr{I}}$. Further on we will denote individuals $\mathbf{x}$ in bold font to differ them from the input x for M. For ease of notion we will write for some shallow configuration $\mu \in \mathscr{SC}_{M,x}$ in the following also μ for the respecting concept in $\mathscr{T}$.

The root of $T_{M(x)}$ is the initial configuration $\square 0 q_0 \underbrace{\# \dots \#}_{\ell}$. Now we will define

$$\mathscr{I}(S) := \bigcup_{i \geq 0} \mathscr{I}_i(S)$$

starting with $\Delta^{\mathscr{I}_0(S)} := \{\mathbf{x}\}$ and

- $S^{\mathscr{I}_0(S)} := \{\mathbf{x}\}$, and
- $y \in (S_0)^{\mathscr{I}_0(S)}$ with $(\mathbf{x}, \mathbf{y}) \in \square^{\mathscr{I}_0(S)}$ (i.e., $(\exists \square . S_0)^{\mathscr{I}_0(S)} = \{\mathbf{x}\}$)

inductively as follows. (**1**) For every node $v \in V$ s.t. $C_v = S_1 \dots S_j R \mu$ with $\mu \in \mathrm{bin}(\mathbb{N}) \times \Psi^h \cdot Q \cdot \Psi^k$ and $h + k = \ell - 1$ is the corresponding configuration in $M(x)$ and let $\mathbf{x}_1, \dots, \mathbf{x}_j, \mathbf{x}_r, \mathbf{x}_\mu \in \Delta^{\mathscr{I}_i(S)}$ be individuals such that $(\mathbf{x}_1, \mathbf{x}_2) \in (S_1)^{\mathscr{I}_i(S)}, (\mathbf{x}_2, \mathbf{x}_3) \in (S_2)^{\mathscr{I}_i(S)}, \dots, (\mathbf{x}_j, \mathbf{x}_r) \in (S_j)^{\mathscr{I}_i(S)}, (\mathbf{x}_r, \mathbf{x}_\mu) \in R^{\mathscr{I}_i(S)}$ and $\mathbf{x}_\mu \in \mu^{\mathscr{I}_i(S)}$:

if $u \in V$ with $(v, u) \in E$ is a post configuration $C_u = S_1 \dots S_j R_1 \dots R_k \lambda$ for $\lambda \in \mathrm{bin}(\mathbb{N}) \times \Psi^h \cdot Q \cdot \Psi^k$ and $h + k = \ell - 1$ of the configuration C_v in the computation of $M(x)$, i.e., $C_v \vdash_M C_u$, then

- add $\mathbf{x}_r$ to $\lambda^{\mathscr{I}_{i+1}(S)}$ for $k = 0$, and otherwise
- if there do not exist $\mathbf{y}_1, \dots, \mathbf{y}_k \in \Delta^{\mathscr{I}_i(S)}$ with

$$(\mathbf{x}_r, \mathbf{y}_1) \in (R_1)^{\mathscr{I}_i(S)}, (\mathbf{y}_1, \mathbf{y}_2) \in (R_2)^{\mathscr{I}_i(S)}, \dots, (\mathbf{y}_{k-1}, \mathbf{y}_k) \in (R_k)^{\mathscr{I}_i(S)}$$

and $\mathbf{y}_k \in \lambda^{\mathcal{I}_i(S)}$, then introduce new individuals $\mathbf{y}_1, \ldots, \mathbf{y}_k$ to $\Delta^{\mathcal{I}_{i+1}(S)}$ and add $(\mathbf{x}_\mu, \mathbf{y}_1)$ to $(R_1)^{\mathcal{I}_{i+1}(S)}$, $(\mathbf{y}_1, \mathbf{y}_2)$ to $(R_2)^{\mathcal{I}_{i+1}(S)}$, …, $(\mathbf{y}_{k-1}, \mathbf{y}_k)$ to $(R_k)^{\mathcal{I}_{i+1}(S)}$ and include $\mathbf{y}_k$ into $\lambda^{\mathcal{I}_{i+1}(S)}$.

(**2**) For every individual $\mathbf{x} \in \Delta^{\mathcal{I}_i(S)}$ and deep configuration χ that is also a shallow configuration with $\mathbf{x} \in \chi^{\mathcal{I}_i(S)}$ include $\mathbf{x}$ into $B^{\mathcal{I}_{i+1}(S)}$.

In the following we will show that $\mathcal{I}(S)$ is indeed a valid interpretation for $\mathcal{T}$ but $S \not\sqsubseteq_{\mathcal{T}} B$. As there is no axiom in $\mathcal{T}$ with S on the right side it holds that $|S^{\mathcal{I}(S)}| = 1$. Assume there is some GCI $G = A_G \sqsubseteq B_G \in \mathcal{T}$ which is violated in $\mathcal{I}(S)$, i.e., we have some individual $\mathbf{x}' \in \Delta^{\mathcal{I}(S)}$ s.t. $\mathbf{x}' \in (A_G)^{\mathcal{I}(S)}$ but $\mathbf{x}' \notin (B_G)^{\mathcal{I}(S)}$. As in $\mathcal{T}$ there are two different kinds of axioms we have to distinguish these cases (because the axiom with S on the left side cannot be such a violated axiom):

(1.) If $G = \alpha \sqsubseteq \beta \in \mathcal{T}$ for α and β being atomic (this is the case for axioms with concepts representing shallow configurations on the left side and B on the right side), then $\mathbf{x}' \in \alpha^{\mathcal{I}(S)}$ but $\mathbf{x}' \notin \alpha^{\mathcal{I}(S)}$. Now consider the least index n s.t. $\mathbf{x}' \in \alpha^{\mathcal{I}_n(S)}$. As α represents clearly a shallow configuration and $\beta = B$ then $\mathbf{x}'$ is added to $\beta^{\mathcal{I}_{n+1}(S)} \subseteq \beta^{\mathcal{I}(S)}$ by (**2**), which contradicts the assumption.

(2.) If $G = \exists R.\mu \sqsubseteq \exists R_1.\ldots\exists R_k.\lambda \in \mathcal{T}$ wherefore exist some entry in f from M s.t. $(S_1 \ldots S_j R \mu) \vdash_M (S_1 \ldots S_j R_1 \ldots R_k \lambda)$ for some stack symbols $S_1, \ldots, S_j$, then it holds that $\mathbf{x}' \in (\exists R.\mu)^{\mathcal{I}(S)}$ but $\mathbf{x}' \notin (\exists R_1.\ldots\exists R_k.\lambda)^{\mathcal{I}(S)}$. Now let n denote the least index s.t. $\mathbf{y}$ is added to $(\mu)^{\mathcal{I}_n(S)}$ and there must be some $m < n$ s.t. $(\mathbf{x}', \mathbf{y})$ is added to $R^{\mathcal{I}_m(S)}$. Then in step (**1**) there are $\mathbf{y}_1, \ldots, \mathbf{y}_k$ added to $\Delta^{\mathcal{I}_{n+1}(S)}$, the corresponding R_i-edges are added to their respective $(R_i)^{\mathcal{I}_{n+1}(S)}$-set and $\mathbf{y}_k$ is added to $\lambda^{\mathcal{I}_{n+1}(S)}$ obeying $\mathbf{x} \in (\exists R_1.\ldots\exists R_k.\lambda)^{\mathcal{I}_{n+1}(S)} \subseteq (\exists R_1.\ldots\exists R_k.\lambda)^{\mathcal{I}(S)}$. This contradicts our assumption again.

Consequently $\mathcal{I}(S)$ is a model of $\mathcal{T}$. Now assume that $S^{\mathcal{I}(S)} \subseteq B^{\mathcal{I}(S)}$. Thus for the starting point $\mathbf{x}$ which is added to $S^{\mathcal{I}(S)}$ at the initial construction step of $\mathcal{I}(S)$, it holds in particular that $\mathbf{x} \in B^{\mathcal{I}(S)}$. As $\mathbf{x}$ is added to $B^{\mathcal{I}(S)}$ if and only if $\mathbf{x}$ is added to $\mu^{\mathcal{I}(S)}$ for some shallow configuration μ, we can conclude that an accepting configuration must be reachable in $T_{M(x)}$ which contradicts our assumption (of the absence of such a computation sequence). Thus an inductive argument proves that $\mu \in \mathbf{x}^{\mathcal{I}_n(S)}$ for $\{\mathbf{x}\} = S^{\mathcal{I}(S)}$ implies that M reaches an accepting configuration on x in $T_{M(x)}$.

Claim. Let $C = (R_1 \ldots R_k \mu)$ be a configuration. It holds for all $n \in \mathbb{N}$ that if $\mathbf{x} \in (\exists R_1.\ldots\exists R_k.\mu)^{\mathcal{I}_n(S)}$ and $\{\mathbf{x}\} = S^{\mathcal{I}(S)}$ then M reaches C in the computation on x in its computation tree $T_{M(x)}$.

Induction basis. Let $n = 1$ and $C = (R_1 \ldots R_k.\mu)$ for $\mu \in \mathcal{SC}_{M,s}$ be some configuration with $\mathbf{x} \in (\exists R_1.\ldots\exists R_k.\mu)^{\mathcal{I}_1(S)}$ and $\{\mathbf{x}\} = S^{\mathcal{I}(S)}$. Thus the individual $\mathbf{x}$ is added to $(\exists R_1.\ldots\exists R_k.\mu)^{\mathcal{I}_1(S)}$ because we have some axiom s.t. $\exists\square.(\mathrm{bin}(0)\#\ldots\#) \sqsubseteq \exists R_1.\ldots\exists R_k.\mu \in \mathcal{T}$ as we only have one step in this case. Hence C can be reached from the initial configuration $\square 0 q_0 \#\ldots\#$ in one step via the transition that corresponds to the before mentioned axiom, i.e., $\square 0 q_0 \#\ldots\# \vdash_M R_1.\ldots R_k \mu$.

Induction step. Let $n > 1$ and assume the claim holds for all $m < n$. Now we have some configuration $C = (S_1 \dots S_j R_1 \dots R_k \mu)$ for $\mu \in \mathscr{SC}_{M,x}$ such that it holds that $\mathbf{x} \in (\exists S_1 \dots . \exists S_j . \exists R_1 \dots . \exists R_k . \mu)^{\mathscr{I}_n(S)}$ and $\{\mathbf{x}\} = S^{\mathscr{I}(S)}$. By induction hypothesis we have some other configuration $C' = (S_1 \dots S_j R \lambda)$ with $\lambda \in \mathscr{SC}_{M,x}$ from which C occurs in one step, i.e., $C' \vdash_M C$, and C is reachable on the computation of $M(x)$ and $\mathbf{x} \in (\exists S_1 \dots . \exists S_j . \exists R . \lambda)^{\mathscr{I}_{n-1}(S)}$. Thus we also have some axiom that adds $\mathbf{x}$ to the set $(\exists S_1 \dots . \exists S_j . \exists R_1 \dots . \exists R_k . \mu)^{\mathscr{I}_n(S)}$ in (**1**). This axiom is of the form

$$\exists R . \lambda \sqsubseteq \exists R_1 \dots . \exists R_k . \mu \in \mathscr{T}.$$

As M reaches C' by induction hypothesis and C can be reached via one step from C' and $\mathbf{x}$ is an instance of $\exists S_1 \dots . \exists S_j . \exists R_1 \dots . \exists R_k . \mu$, M can also reach C within the computation on x.

Hence this contradicts our assumption that M does not accept x and completes our proof. □

Finally the classification of the full quantifier fragments naturally emerges from the previous cases to EXP-complete, coNP-, and P-hard cases.

Theorem 4.44 ($\mathscr{Q} = \{\forall, \exists\}$).
Let B be a finite set of Boolean operators and $\mathscr{Q} = \{\forall, \exists\}$.

(1.) Let $X \in \{\mathsf{N}_2, \mathsf{V}_2, \mathsf{E}_2\}$. If $X \subseteq [B]$, then $\mathrm{SUBS}_{\mathscr{Q}}(B)$ is EXP-complete.

(2.) If $\mathsf{I}_0 \subseteq [B]$ or $\mathsf{I}_1 \subseteq [B]$, then $\mathrm{SUBS}_{\mathscr{Q}}(B)$ is EXP-complete.

(3.) If $\mathsf{D}_2 \subseteq [B] \subseteq \mathsf{D}_1$, then $\mathrm{SUBS}_{\mathscr{Q}}(B)$ is coNP-hard and in EXP.

(4.) If $[B] \in \{\mathsf{I}_2, \mathsf{L}_2\}$, then $\mathrm{SUBS}_{\mathscr{Q}}(B)$ is P-hard and in EXP.

All hardness results hold w.r.t. $\leq_{\mathbf{m}}^{\log}$ reductions.

Proof. (1.) Follows from the respective lower bounds of $\mathrm{SUBS}_{\exists}(B)$, resp., $\mathrm{SUBS}_{\forall}(B)$ shown in Theorems 4.41 and 4.42.

(2.) The needed lower bound follows from Lemma 4.39 (3.) enabling a reduction from $\overline{\mathrm{TCSAT}_{\exists\forall}(\mathsf{I}_0)}$ which is EXP-complete by Theorem 4.7 on page 79. The case $\mathrm{SUBS}_{\exists\forall}(B)$ with $\mathsf{I}_1 \subseteq [B]$ follows from the contraposition argument in Lemma 4.36.

(3.)+(4.) The lower bounds carry over from $\mathrm{SUBS}_{\emptyset}(B)$ for the respective sets B (see Theorem 4.40). □

4.2.1 Conclusion

In Figures 4.1 and 4.2 it is depicted how the results arrange in Post's lattice. The classification has shown that the subsumption problem with both quantifiers is much harder than the previously visited terminology problems in Section 4.1. Having access to at least one constant already raises the complexity of the problem to EXP-completeness. For the fragments having access to only one of the quantifiers the clones which are able to

express either disjunction (for the universal quantifier) or conjunction (for the existential case) become tractable (plus both constants). Without any quantifier allowed the problem almost behaves as the propositional implication problem with respect to tractability. The only exception of this rule are the L-cases that can express negation or at least one constant. They become coNP-complete and therewith intractable.

Finally a similar systematic study of the subsumption problem for concepts (without respect to a terminology) would be of great interest because of the close relation to the implication problem of modal formulae. To the best of the author's knowledge such a study has not been done yet and would enrich the overall picture of the complexity situation in this area of research. Further, a classification for other description logics from the $\mathscr{FL}$- and $\mathscr{EL}$-family would be very interesting. Furthermore, closing the gaps between upper and lower bounds of the clones $\mathsf{I}_2, \mathsf{L}_2, \mathsf{D}_2$ and D_1 would finish our study.

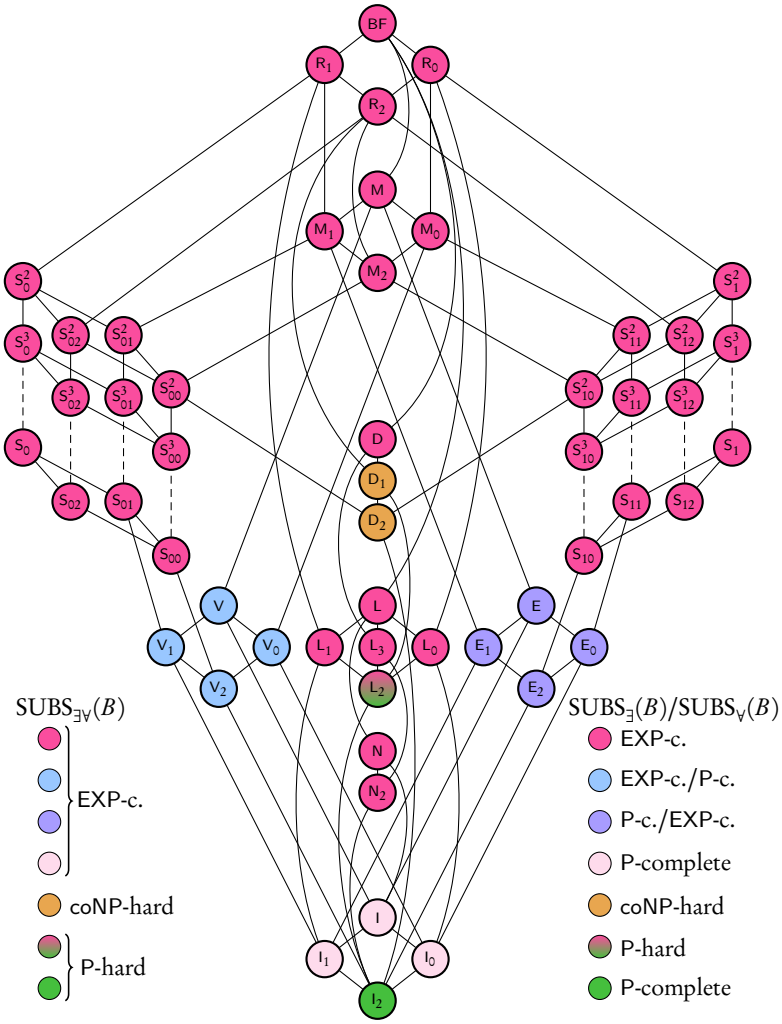


Figure 4.1: Post's lattice showing the complexity of $\mathrm{SUBS}_{\mathscr{Q}}(B)$ for all non-empty sets $\emptyset \subsetneq \mathscr{Q} \subseteq \{\exists, \forall\}$ and all Boolean clones $[B]$.

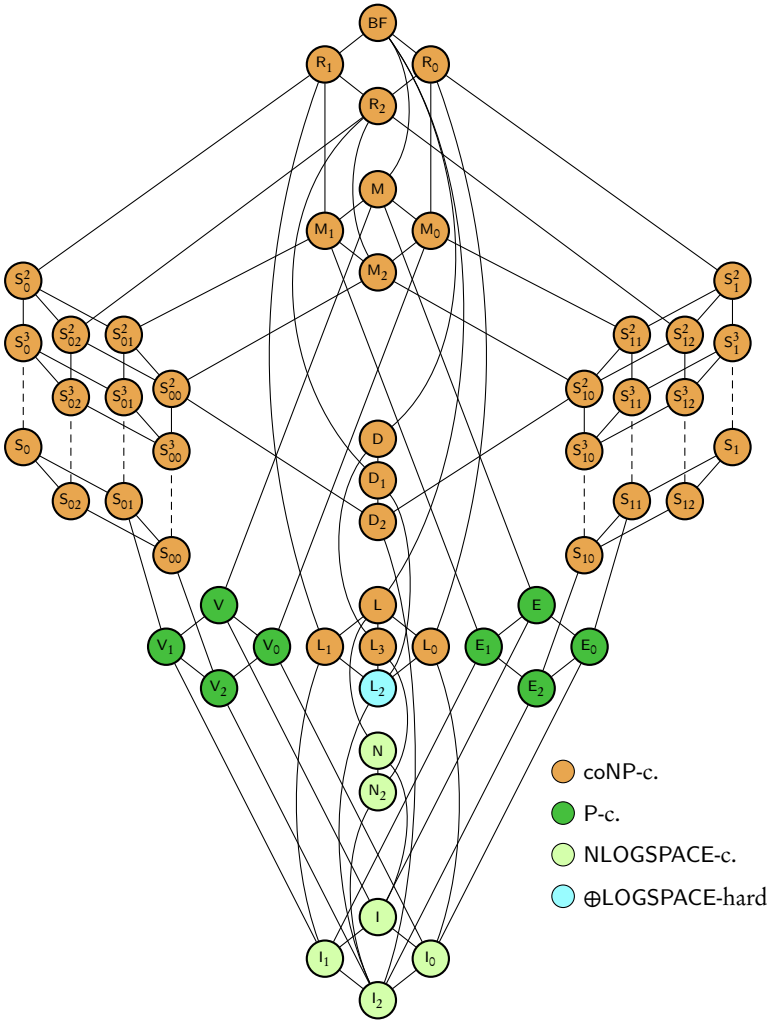


Figure 4.2: Post's lattice showing the complexity for $\mathrm{SUBS}_{\emptyset}(B)$ and all Boolean clones $[B]$.

Chapter 5

Concluding Remarks

We almost completely classified the satisfiability as well as the model checking problem for the temporal computation tree logics CTL and CTL* with respect to their operators and path quantifiers as well as their Boolean fragments. Furthermore we have seen how several ideas and techniques used in these proofs can be generalized to work with extensions of the logic CTL. We thereby visited and classified the logic CTL$^+$ which is, informally, the logic CTL that allows Boolean combinations of path formulae. Adding the ability to express fairness constraints to CTL extends the logic to ECTL which has been studied with respect to the aforementioned fragments in the next step. In the context of temporal logic we always worked with the two most central problems. On the one hand this is the satisfiability problem and on the other hand the model checking problem. For the latter we followed an approach of Sistla et al. and divided the problems into three kinds of fragments: monotone, atomic negation, and positive. Interestingly we were able to show that these three types of instances are computationally equivalent. Thus negation seems rather irrelevant in the context of model checking.

Further research in this area of temporal logics should aim to close the gaps between upper and lower bounds for the affine cases (which involve the exclusive-or function) with respect to the satisfiability problem: for a temporal logic $\mathscr{L} \in \{\mathrm{CTL}, \mathrm{CTL}^\star, \mathrm{CTL}^+, \mathrm{ECTL}\}$, B a set of Boolean functions such that $\mathsf{L}_0 \subseteq [B] \subseteq \mathsf{L}$, and all sets of temporal operators/path quantifiers T the problems $\mathscr{L}$-SAT(T,B) lack matching upper and lower bounds. Nevertheless we presented some intuition about these cases in Section 3.1.4. Additionally we were able to state some improvements of the trivial upper bounds for four fragments.

After leaving the temporal extensions of modal logic we moved forward to the field of description logics which comprises of several rather diverse extensions. As many such concepts emerge in as many various types of description logics we restricted ourself to the logic $\mathscr{ALC}$ which is closest to modal logic. Hereby we could analyze which concepts in this logic would interact with the intractability of their decision problems. These concepts turned out to be implication and conjunction which are heavily included in the definition of terminologies. Thus the behavior of such fragments were mostly independent of the allowed Boolean functions or quantifiers which are allowed to be used in the concept formulae within the axioms. The decision problems which have been classified for their operator and function fragments are terminology satisfiability (with and without respect to a given concept), ontology satisfiability (with and without respect to a given concept), and subsumption with respect to a given terminology.

Whilst the problems are all interreducible the reductions do not hold without any respect to the allowed Boolean functions. Nevertheless we achieved (with two small open

cases) a computational equivalence between the three problems which can talk about a single individual, that are, concept satisfiability with respect to a terminology TCSAT, ontology satisfiability OSAT, and concept satisfiability with respect to an ontology OCSAT. The subsumption problem SUBS with respect to a terminology turned out to be the hardest of the five investigated decision problems as, strictly speaking, only the existence of one single constant (no matter which one) makes the problem EXP-complete if both quantifiers are present.

For further research, one should prove completeness results for the two ontology connected decision problems in the following way. Let $\mathcal{O}\text{-SAT} \in \{\text{OCSAT}, \text{OSAT}\}$ and B be a finite set of Boolean operators such that $[B] \in \{\mathsf{V}, \mathsf{V}_0\}$. Then $\mathcal{O}\text{-SAT}_\exists(B)$ is P-hard and in EXP. We conjecture that both problems can be solved in P but were not able to achieve a membership result yet. One of the most promising approaches was the construction of an equivalent terminology by expressing concept assertions $C(a)$ through axioms $C_a \sqsubseteq C$ and role assertions $S(a, b)$ through axioms $C_a \sqsubseteq \exists S.C_b$. All available individuals a in the given ABox will be then enforced to exist by axioms $\top \sqsubseteq \exists R.C_a$ for each individual a. This technique was inspired by [Hol96]. At the moment it is not clear how to use some minimality condition defined on the size of the interpretation within the proof. Also the construction of a nondeterministic polynomial-time algorithm guessing which part of the disjunctions should be satisfied for the GCIs did not succeed. Thus the author suggests that the first approach should be proving a small model property to get an NP membership result as an intermediate step to the polynomial time algorithm.

Furthermore the next steps of the study of the subsumption problem should involve finding an optimal complexity classification for

- $\text{SUBS}_{\exists\forall}(B)$ for $[B] \in \{\mathsf{I}_2, \mathsf{L}_2, \mathsf{D}_2, \mathsf{D}_1\}$,
- $\text{SUBS}_{\mathcal{Q}}(B)$ for $\mathcal{Q} \in \{\exists, \forall\}$ and $[B] \in \{\mathsf{L}_2, \mathsf{D}_2, \mathsf{D}_1\}$, and
- $\text{SUBS}_{\emptyset}(B)$ for $[B] = \mathsf{L}_2$.

Also, a classification of the subsumption problem *without* respect to a terminology would be of great interest because the problem is equivalent to the modal implication problem, i.e., the problem given two modal formulae $\varphi, \psi \in \text{ML}$ asking whether it holds that $\varphi \models \psi$. To the best of the author's knowledge this problem has not been classified or visited yet.

As mentioned in Section 2.3.2, Schild has proven the correspondence to Propositional Dynamic Logic in [Sch91]. This connection was useful in achieving and improving several results for DL. Still it would be very interesting to obtain more general equivalences of concepts in other logics to the ABoxes in description logics.

Additionally, a study on the application of the complexity results for the tractable cases may be very interesting. How do they influence actual algorithms and how may they enhance the runtime or the space requirements? Especially in the case of model checking for the temporal logics and also for subsumption in the area of description logics these questions are of great relevance and could lead to a significant improvement of current algorithms.

Recently, the field of parameterized complexity [DF99] has gotten more attention in the research community. In this sense, the study of the parameterized versions of a decision problem leads to a very deep understanding of which parts of the problem are inherently difficult to solve. Hence, the study of different kinds of parameterizations may lead to FPT-algorithms stating that the hard parts of the problem only belong to the parameter. Achilleos et al. turned towards parameterized modal satisfiability in [ALM10]. There they inspect several new kinds of parameterizations of modal formulae, e.g., modality depth, diamond dimension, and modal width. There the *diamond dimension* $d_\Diamond(\varphi)$ of a modal formula φ in negation normal form is inductively defined as

$$\begin{aligned} d_\Diamond(p) = d_\Diamond(\neg p) &= 0, \text{ if } p \in \mathsf{PROP}, \\ d_\Diamond(\varphi \wedge \psi) &= d_\Diamond(\varphi) + d_\Diamond(\psi), \\ d_\Diamond(\varphi \vee \psi) &= \max\{d_\Diamond(\varphi), d_\Diamond(\psi)\}, \\ d_\Diamond(\Box\varphi) &= d_\Diamond(\varphi), \text{ and } d_\Diamond(\Diamond\varphi) = 1 + d_\Diamond(\varphi) \end{aligned}$$

and models the intuition, that diamond preceded formulae require some kind of branching whereas boxes do not enforce this mandatorily. The modal width of a formula φ describes, informally speaking, the greatest number of subformulae which appear at some modal depth in φ. An investigation of the interplay of these parameterizations with the modal logic variants inspected in this thesis would be very interesting.

In particular, several kinds of *space* complexity classes have been introduced to the area of parameterized complexity by Stockhusen [Sto11] lately. Thus an application of these classes with respect to the modal logic variants is also of great interest and may improve the overall understanding of intractability notably.

Bibliography

[Aal11] Aalborg University, *A.N. Prior - The Founding Father of Temporal Logic*, 2011, `http://www.prior.aau.dk/`.

[ACG+99] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi, *Complexity and approximation*, 2nd ed., Springer, 1999.

[AI03] M. Adler and N. Immerman, *An n! lower bound on formula size*, ACM Trans. Comput. Logic **4** (2003), no. 3, 296–314.

[ALM10] A. Achilleos, M. Lampis, and V. Mitsou, *Parameterized modal satisfiability*, Proceedings of the 37th international colloquium conference on Automata, languages and programming: Part II (Berlin, Heidelberg), ICALP'10, Springer-Verlag, 2010, pp. 369–380.

[Baa03] F. Baader, *Terminological cycles in a description logic with existential restrictions*, Proc. IJCAI, 2003, pp. 325–330.

[BAPM83] M. Ben-Ari, A. Pnueli, and Z. Manna, *The temporal logic of branching time*, Acta Informatica **20** (1983), 207–226.

[Bau07] M. Bauland, *Complexity results for boolean constraint satisfaction problems*, Ph.D. thesis, Gottfried Wilhelm Leibniz Universität Hannover, 2007.

[BBL05a] F. Baader, S. Brandt, and C. Lutz, *Pushing the $\mathcal{EL}$ envelope*, Proc. IJCAI, 2005, pp. 364–369.

[BBL05b] _____, *Pushing the $\mathcal{EL}$ envelope*, LTCS–Report, vol. 05-01, 2005.

[BBL08] _____, *Pushing the $\mathcal{EL}$ envelope further*, Proc. OWLED DC, 2008.

[BCM+03] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider (eds.), *The description logic handbook: Theory, implementation, and applications*, 2nd ed., vol. 1, Cambridge University Press, 2003.

[BDHM92] G. Buntrock., C. Damm, U. Hertrampf, and C. Meinel, *Structure and importance of logspace-mod class*, Mathematical Systems Theory **25** (1992), no. 3, 223–237.

[BdV01] P. Blackburn, M. de Rijke, and Y. Venema, *Modal logic*, Cambridge University Press, New York, NY, USA, 2001.

[BL84] R. J. Brachman and H. J. Levesque, *The tractability of subsumption in frame-based description languages*, AAAI, 1984, pp. 34–37.

[BMM+11] O. Beyersdorff, A. Meier, M. Mundhenk, T. Schneider, M. Thomas, and H. Vollmer, *Model checking* CTL *is almost always inherently sequential*, Logical Methods in Computer Science (2011).

[BMS+11] M. Bauland, M. Mundhenk, T. Schneider, H. Schnoor, I. Schnoor, and H. Vollmer, *The tractability of model checking for LTL: the good, the bad, and the ugly fragments*, ACM Transactions on Computational Logic (TOCL) **12** (2011), no. 2, 13:1–13:28.

[BMTV09a] O. Beyersdorff, A. Meier, M. Thomas, and H. Vollmer, *The complexity of reasoning for fragments of default logic*, Theory and Applications of Satisfiability Testing - SAT 2009, Lecture Notes in Computer Science, vol. 5584, Springer Berlin / Heidelberg, 2009, pp. 51–64.

[BMTV09b] ———, *The Complexity of Propositional Implication*, Information Processing Letters **109** (2009), no. 18, 1071–1077.

[Bra04a] S. Brandt, *Polynomial time reasoning in a description logic with existential restrictions, GCI axioms, and—what else?*, Proc. ECAI, 2004, pp. 298–302.

[Bra04b] S. Brandt, *Reasoning in $\mathcal{EL}\mathcal{H}$ w.r.t. general concept inclusion axioms*, LTCS-Report LTCS-04-03, Dresden University of Technology, Germany, 2004.

[Bra04c] ———, *Subsumption and instance problem in $\mathcal{EL}\mathcal{H}$ w.r.t. general tboxes*, LTCS-Report LTCS-04-04, Chair for Automata Theory, Institute for Theoretical Computer Science, Dresden University of Technology, Germany, 2004, See http://lat.inf.tu-dresden.de/research/reports.html.

[BS85] R. J. Brachman and J. G. Schmolze, *An overview of the* KL-ONE *knowledge representation system*, Cognitive Science **9** (1985), no. 2, 171–216.

[BSS+09] M. Bauland, T. Schneider, H. Schnoor, I. Schnoor, and H. Vollmer, *The complexity of generalized satisfiability for Linear Temporal Logic*, LMCS **5** (2009), no. 1, 1–21.

[Bus87] S. R. Buss, *The Boolean formula value problem is in* ALOGTIME, Proceedings 19th Symposium on Theory of Computing, ACM Press, 1987, pp. 123–131.

[BVW94] O. Bernholtz, M. Vardi, and P. Wolper, *An automata-theoretic approach to branching-time model checking (extended abstract)*, Proc. 6th International Conference on Computer Aided Verification, Lecture Notes in Computer Science, vol. 818, Springer, 1994, pp. 142–155.

[BvW06] P. Blackburn, J. van Bentham, and F. Wolter, *Handbook of modal logic*, Nort Holland, Amsterdam, 2006.

[CE81] E. M. Clarke and E. A. Emerson, *Desing and synthesis of synchronisation skeletons using branching time temporal logic*, Logic of Programs, Lecture Notes in Computer Science, vol. 131, Springer Verlag, 1981, pp. 52–71.

[CES86] E. Clarke, E. Allen Emerson, and A. Sistla, *Automatic verification of finite-state concurrent systems using temporal logic specifications*, ACM Transactions on Programming Languages and Systems **8** (1986), no. 2, 244–263.

[CGP99] E. Clarke, O. Grumberg, and D. Peled, *Model checking*, The MIT Press, 1999.

[CKS81] A. K. Chandra, D. Kozen, and L. J. Stockmeyer, *Alternation*, Journal of the Association for Computing Machinery **28** (1981), 114–133.

[CMTV10] N. Creignou, A. Meier, M. Thomas, and H. Vollmer, *The complexity of reasoning for fragments of autoepistemic logic*, Circuits, Logic, and Games (Dagstuhl, Germany) (Benjamin Rossman, Thomas Schwentick, Denis Thérien, and Heribert Vollmer, eds.), Dagstuhl Seminar Proceedings, no. 10061, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2010.

[Coo71a] S. A. Cook, *Characterizations of pushdown machines in terms of time-bounded computers*, Journal of the ACM **18** (1971), no. 1, 4–18.

[Coo71b] ———, *The complexity of theorem proving procedures*, Proceedings 3rd Symposium on Theory of Computing, ACM Press, 1971, pp. 151–158.

[CST10] N. Creignou, J. Schmidt, and M. Thomas, *Complexity of propositional abduction for restricted sets of boolean functions*, Proc. 12th International Conference on the Principles of Knowledge Representation and Reasoning, AAAI Press, 2010, pp. 8–16.

[CSTW10] N. Creignou, J. Schmidt, M. Thomas, and S. Woltran, *Sets of boolean connectives that make argumentation easier*, Proc. 12th European Conference on Logics in Artificial Intelligence, Lecture Notes in Computer Science, vol. 6341, Springer, 2010, pp. 117–129.

[CSV84] A. K. Chandra, L. Stockmeyer, and U. Vishkin, *Constant depth reducibility*, SIAM Journal of Computing **13** (1984), no. 2, 423–439.

[DF99] R. G. Downey and M. R. Fellows, *Parameterized complexity*, Springer, 1999.

[DM00] F. M. Donini and F. Massacci, EXPTIME *tableaux for $\mathcal{ALC}$* , AI **124** (2000), no. 1, 87–138.

[Don03] F. M. Donini, *Complexity of reasoning*, in Baader et al. [BCM+03], pp. 96–136.

[EH85] E. Allen Emerson and J. Y. Halpern, *Decision procedures and expressiveness in the temporal logic of branching time*, Journal of Computer and System Sciences **30** (1985), no. 1, 1–24.

[EH86] ———, *"sometimes" and "not never" revisited: On branching versus linear time*, Journal of the ACM **33** (1986), no. 1, 151–178.

[EJ00] E. Allen Emerson and C. S. Jutla, *The complexity of tree automata and logics of programs.*, SIAM Journal of Computing **29** (2000), no. 1, 132–158.

[EL87] E. Allen Emerson and C.-L. Lei, *Modalities for model checking: Branching time logic strikes back*, Science of Computer Programming **8** (1987), no. 3, 275–306.

[Eme90] E. Allen Emerson, *Temporal and modal logic*, Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics, ch. 16, pp. 995–1072, Elsevier and MIT Press, 1990.

[End01] H. B. Enderton, *A mathematical introduction to logic*, second ed., Hartcourt/Academic Press, 2001.

[FL79] M. J. Fischer and R. E. Ladner, *Propositional modal logic of programs*, Journal of Computer and Systems Sciences **18** (1979), 194–211.

[Gal87] A. Galton, *Temporal logics and their applications*, Academic Press, Inc., San Diego, CA, USA, 1987.

[Gia95] G. De Giacomo, *Decidability of class-based knowledge representation formalisms*, Ph.D. thesis, Università degli Studi di Roma "La Sapienza", 1995.

[GJ79] M. R. Garey and D. S. Johnson, *Computers and intractability, a guide to the theory of np-completeness*, Freeman, New York, 1979.

[GMWK02] R. Givan, D. McAllester, C. Wittny, and D. Kozen, *Tarskian set constraints*, Information and Computation **174** (2002), 105–131.

[Gol06] R. Goldblatt, *Mathematical modal logic: A view of its evolution*, Logic and the Modalities in the Twentieth Century (D. Gabbay and J. Woods, eds.), Handbook of the History of Logic, vol. 7, North-Holland, Amsterdam, 2006, pp. 1–98.

[HC68] G. Hughes and M. Cresswell, *An introduction to modal logic*, Methuen, London, UK, 1968.

[HJ94] H. Hansson and B. Jonsson, *A logic for reasoning about time and reliability*, Formal Aspects of Computing **6** (1994), 512–535, 10.1007/BF01211866.

[HM92] J. Halpern and Y. Moses, *A guide to completeness and complexity for modal logics of knowledge and belief.*, Artificial Intelligence **54** (1992), no. 2, 319–379.

[HMU00] J. E. Hopcroft, R. Motwani, and J. D. Ullman, *Introduction to automata theory, languages, and computation*, 2nd ed., Addison Wesley, November 2000.

[Hof05] M. Hofmann, *Proof-theoretic approach to description-logic*, Proc. LICS, 2005, pp. 229–237.

[Hol96] B. Hollunder, *Consitency checking reduced to satisfiability of concepts in terminological systems*, Annals of Mathematics and Artificial Intelligence **18** (1996), 133–157.

[HSS08] E. Hemaspaandra, H. Schnoor, and I. Schnoor, *Generalized modal satisfiability*, CoRR **abs/0804.2729** (2008), 1–32.

[JL03] J. Johannsen and M. Lange, CTL$^+$ *is complete for double exponential time*, Proc. 30th Int. Coll. on Automata, Logics and Programming, ICALP'03, vol. 2719 of LNCS, Springer, 2003, pp. 767–775.

[Joh90] D. S. Johnson, *A catalogue of complexity classes*, Handbook of Theoretical Computer Science (J. van Leeuwen, ed.), vol. A, Elsevier, 1990, pp. 67–161.

[KF09] L. Kuhtz and B. Finkbeiner, *LTL Path Checking is Efficiently Parallelizable*, Proceedings of the 36th International Colloquium on Automata, Languages and Programming: Part II, Lecture Notes in Computer Science, vol. 5556, 2009, pp. 235–246.

[Kri63] S. Kripke, *Semantical considerations on modal logic*, Acta Philosophica Fennica, vol. 16, 1963, pp. 84–94.

[Krö87] F. Kröger, *Temporal logic of programs*, Springer-Verlag New York, Inc., New York, NY, USA, 1987.

[KVW00] O. Kupferman, M. Y. Vardi, and P. Wolper, *An automata-theoretic approach to branching-time model checking*, Journal of the ACM **47** (2000), no. 2, 312–360.

[KWLS09] A. Kara, V. Weber, M. Lange, and T. Schwentick, *On the hybrid extension of* CTL *and* CTL$^+$, MFCS '09: Proceedings of the 34th International Symposium on Mathematical Foundations of Computer Science 2009 (Berlin, Heidelberg), Springer-Verlag, 2009, pp. 427–438.

[Lad77] R. Ladner, *The computational complexity of provability in systems of modal propositional logic*, SIAM J. Comput. **6** (1977), no. 3, 467–480.

[Lar95] F. Laroussinie, *About the expressive power of* CTL *combinators*, Information Processing Letters **54** (1995), no. 6, 343–345.

[Lev73] L. A. Levin, *Universal sorting problems*, Problems of Information Transmission **9** (1973), 265–266.

[Lew18] C. I. Lewis, *A survey of symbolic logic*, University of California Press, Berkley, 1918.

[Lew79] H. Lewis, *Satisfiability problems for propositional calculi*, Math. Sys. Theory **13** (1979), 45–53.

[LMS01] F. Laroussinie, N. Markey, and P. Schnoebelen, *Model Checking* CTL$^+$ *and FCTL is Hard*, Proc. 4th Foundations of Software Science and Computation Structure, Lecture Notes in Computer Science, vol. 2030, Springer Verlag, 2001, pp. 318–331.

[LP82] H. R. Lewis and C. H. Papadimitriou, *Symmetric space-bounded computation*, Theoretical Computer Science **19** (1982), no. 2, 161 – 187.

[Mar04] N. Markey, *Past is for free: on the complexity of verifying linear temporal properties with past*, Acta Informatica **40** (2004), no. 6-7, 431–458.

[MMS+09] A. Meier, M. Mundhenk, T. Schneider, M. Thomas, V. Weber, and F. Weiss, *The complexity of satisfiability for fragments of hybrid logic — Part I*, Proc. MFCS, LNCS, vol. 5734, 2009, pp. 587–599.

[MMTV09] A. Meier, M. Mundhenk, M. Thomas, and H. Vollmer, *The complexity of satisfiability for fragments of* CTL *and* CTL*, International Journal of Foundations of Computer Science **20** (2009), no. 05, 901–918.

[MPSP09] B. Motik, P. F. Patel-Schneider, and B. Parsia, *Owl 2 web ontology language: Structural specification and functional-style syntax*, 2009, `http://www.w3.org/TR/2009/REC-owl2-syntax-20091027/`.

[MR95] R. Motwani and P. Raghavan, *Randomized algorithms*, Cambridge University Press, 1995.

[MS11a] A. Meier and T. Schneider, *Generalized satisfiability for the description logic $\mathcal{ALC}$*, Proceedings of the 8th Annual Conference on Theory and Application of Models of Computation, Lecture Notes in Computer Science, vol. 6648, Springer Verlag, 2011, pp. 552–562.

[MS11b] ———, *Generalized satisfiability for the description logic $\mathcal{ALC}$*, CoRR **abs/1103.0853** (2011), 1–37.

[NB03] D. Nardi and R. J. Brachman, *An introduction to description logics*, 2nd ed., ch. 1, vol. 1 of Baader et al. [BCM+03], 2003.

[ØH95] P. Øhrstrøm and P. Hasle, *Temporal logic: From ancient ideas to artificial intelligence*, Kluwer Academic Publishers, Dordrecht, Boston and London, 1995.

[Pap94] C. H. Papadimitriou, *Computational complexity*, Addison-Wesley, 1994.

[Pip97] N. Pippenger, *Theories of computability*, Cambridge University Press, 1997.

[Pnu77] A. Pnueli, *The temporal logic of programs*, Proc. 18th Symposium on Foundations of Computer Science, IEEE Computer Society Press, 1977, pp. 46–57.

[Pos41] E. Post, *The two-valued iterative systems of mathematical logic*, Annals of Mathematical Studies **5** (1941), 1–122.

[Pra78] V. R. Pratt, *A practical decision method for propositional dynamic logic: Preliminary report*, STOC, ACM, 1978, pp. 326–337.

[Pra80] ———, *A near-optimal method for reasoning about action*, Journal of Computer and System Sciences **20** (1980), no. 2, 231–254.

[Pri57] A. N. Prior, *Time and modality*, Clarendon Press, Oxford, 1957.

[Pri67] ———, *Past, present, and future*, Clarendon Press, Oxford, 1967.

[QS82] J.-P. Queille and J. Sifakis, *Specification and verification of concurrent systems in cesar*, Proceedings 5th International Symposium on Programming, Lecture Notes in Computer Science, vol. 137, Springer Verlag, 1982, pp. 337–351.

[Rei01] S. Reith, *Generalized satisfiability problems*, Ph.D. thesis, Fachbereich Mathematik und Informatik, Universität Würzburg, 2001.

[Rei05] O. Reingold, *Undirected st-connectivity in log-space*, STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing (New York, NY, USA), ACM, 2005, pp. 376–385.

[RKNP04] J. Rutten, M. Kwiatkowska, G. Norman, and D. Parker, *Mathematical techniques for analyzing concurrent and probabilistic systems,* p. panangaden and f. van breugel (eds.), CRM Monograph Series, vol. 23, American Mathematical Society, 2004.

[Ruz80] W. L. Ruzzo, *Tree-size bounded alternation*, Journal of Computer and System Sciences **21** (1980), 218–235.

[RV97] K. Regan and H. Vollmer, *Gap-languages and log-time complexity classes*, Theoretical Computer Science **188** (1997), 101–116.

[Sav70] W. J. Savitch, *Relationships between nondeterministic and deterministic tape complexities*, Journal of Computer and System Sciences **4** (1970), no. 2, 177 – 192.

[SC85] A. Sistla and E. Clarke, *The complexity of propositional linear temporal logics*, Journal of the ACM **32** (1985), no. 3, 733–749.

[Sch91] K. Schild, *A correspondence theory for terminological logics: Preliminary report*, In Proc. of IJCAI-91, 1991, pp. 466–471.

[Sch02] P. Schnoebelen, *The complexity of temporal logic model checking*, Advances in Modal Logic, vol. 4, 2002, pp. 393–436.

[Sch07] H. Schnoor, *Algebraic techniques for satisfiability problems*, Ph.D. thesis, Gottfried Wilhelm Leibniz Universität Hannover, 2007.

[Sch08] I. Schnoor, *The weak base method for constraint satisfaction*, Ph.D. thesis, Gottfried Wilhelm Leibniz Universität Hannover, 2008.

[Sch10] H. Schnoor, *The complexity of model checking for Boolean formulas*, Int. J. Found. Comput. Sci. **21** (2010), no. 3, 289–309.

[Sel82] A. L. Selman, *Reductions on* NP *and* P*-selective sets*, Theoretical Computer Science **19** (1982), 287–304.

[SI90] R. Sridhar and S. Iyengar, *Efficient parallel algorithms for functional dependency manipulations*, Proc. ICPADS, ACM, 1990, pp. 126–137.

[Sip05] M. Sipser, *Introduction to the theory of computation*, 2nd ed., Course Technology, February 2005.

[Smo87] R. Smolensky, *Algebraic methods in the theory of lower bounds for Boolean circuit complexity*, Proceedings of the 19th annual ACM symposium on Theory of computing (New York, NY, USA), STOC '87, ACM, 1987, pp. 77–82.

[Sto77] L. J. Stockmeyer, *The polynomial-time hierarchy*, Theoretical Computer Science **3** (1977), 1–22.

[Sto11] C. Stockhusen, *Anwendung monadischer Logik zweiter Stufe auf Probleme beschränkter Baumweite und deren Platzkomplexität*, Diplomarbeit, Universität zu Lübeck, 2011.

[Tho09] M. Thomas, *The complexity of circumscriptive inference in post's lattice*, Proc. 10th International Conference on Logic Programming and Nonmonotonic Reasoning, Lecture Notes in Computer Science, vol. 5753, Springer, 2009, pp. 209–302.

[Tho10] ———, *On the complexity of fragments of nonmonotonic logics*, Ph.D. thesis, Leibniz University of Hannover, 2010.

[Vol99] H. Vollmer, *Introduction to circuit complexity*, Springer, 1999.

[VS85a] M. Y. Vardi and L. Stockmeyer, *Improved upper and lower bounds for modal logics of programs: Preliminary report*, STOC '85: Proceedings of the seventeenth annual ACM Symposium on Theory of computing, Lecture Notes in Computer Science, 1985, pp. 240–251.

[VS85b] ———, *Lower bound in full (*EEXP*-hardness for* CTL**-SAT)*, Online, available at `http://www.cs.rice.edu/~vardi/papers/ctl_star_lower_bound.pdf`, 1985.

[VW86] M. Y. Vardi and P. Wolper, *Automata-theoretic techniques for modal logics of programs*, JCSS **32** (1986), no. 2, 183–221.

[Web09a] V. Weber, *Branching-time logics repeatedly referring to states*, Journal of Logic, Language and Information **18** (2009), 593–624, 10.1007/s10849-009-9093-x.

[Web09b] ———, *On the complexity of branching-time logics*, Computer Science Logic (Erich Grädel and Reinhard Kahle, eds.), Lecture Notes in Computer Science, vol. 5771, Springer Berlin / Heidelberg, 2009, 10.1007/978-3-642-04027-6_38, pp. 530–545.

[Wil99] T. Wilke, CTL$^+$ *is exponentially more succinct than* CTL, Proceedings of the 19th Conference on Foundations of Software Technology and Theoretical Computer Science (London, UK), Springer-Verlag, 1999, pp. 110–121.

Index

Symbols

$\#_{\mathrm{T}}(\psi)$ **36**, 46
$\Box$ **17**
$\Box_i$ **17**
Δ_k^{P} **11**
$\Diamond$ **17**
$\Diamond_i$ **17**
$\mathfrak{K}$ **17**
$\mathfrak{O}_{\mathscr{Q}}(B)$ **25**
Π_k^{P} **11**
$\Pi_1^{\mathscr{R}}$ **13**
R **54**
⋆SAT **75**
Σ_k^{P} **11**
$\Sigma_1^{\mathscr{R}}$ **13**
$\mathfrak{T}_{\mathscr{Q}}(B)$ **25**
⋆SAT$^{\mathrm{ind}}$ **75**
$\leq_{\mathbf{m}}^{\mathbf{log}}$ . . . . . *see* reduction, logspace
η **17**
$\bot$ **9**, 17
$\leq_{\mathbf{cd}}$ . . . . . *see* reduction, constant depth
$\leq_{\mathbf{proj}}^{\mathbf{dlt}}$ . . . . . *see* reduction, dlt-proj
⊕LOGSPACE **10**, 44, 94, 106
$\top$ **9**, 17
[B] **13**
1-in-3-SAT **44**, 83, 84

A

ABox **25**
AC^1(LOGDCFL) 52
AC^i **11**
affine **14**
$\mathscr{AL}$ 79
$\mathscr{ALC}$ **23**, 86, 88
    semantics **24**
$\mathscr{ALU}$ 79
atomic
    concept . . . . . *see* concept, atomic
    formulae **19**
    propositions 17
axiom **25**

B

base **13**
    $\mathscr{B}_0$ **11**
    $\mathscr{B}_1$ **11**
Boolean function **9**
    affine **14**, 43
    c-reproducing **13**, 79
    c-separating **14**
        of degree *n* **14**
    identical **14**
    linear **14**
    monotone **13**, 82
    self-dual **14**, 80, 94
    T_n^{n+1} **14**

C

c-reproducing **13**, 79
c-separating **14**
    of degree *n* **14**
clone **13**
    base **13**
    list of all 15
concept **23**
    atomic **24**

set of all **24**
description **24**
empty **24**
subconcept **77**
universal **24**
concept description **23**
coNLOGTIME 12
$\mathsf{Con}_{\mathscr{Q}}(B)$ **24**
constant depth reduction . *see* reduction
CSAT **24**, 25
CTL **20**
CTL^{+} **22**
$\mathrm{CTL}_{\mathrm{a.n.}}$ **52**
$\mathrm{CTL}_{\mathrm{mon}}$ **52**
CTL^{+} 62
CTL^{+}-SAT 45, 49, 50
$\mathrm{CTL}_{\mathrm{pos}}$ **52**
CTL-SAT **21**, 32, 50
$\mathrm{CTL}^{\star}$ **19**, 68
$\mathrm{CTL}^{\star}$-MC **20**
$\mathrm{CTL}^{\star}$-SAT **20**, 41, 50
$\mathrm{CTL}^{\star}$ **19**
ECTL . . . . . *see* ECTL
formula 20
operators 21, 32
CTL-operator **20**

D

dual(·) **14**, **78**, 80, 94

E

ECTL **22**, 47
EEXP **10**, 12
$\mathscr{EL}$ 79, 86, 92, 104
$\mathscr{ELH}$ 98
$\mathscr{EL}^{++}$ 23, 92
$\mathscr{ELU}$ 86
empty concept . . . . . *see* concept
$\mathbf{exp}_k(n)$ **10**
EXP **10**, 12

F

fairness 22
$\mathfrak{F}_{\mathrm{all}}$ **18**
$\overset{\infty}{\mathrm{F}}$ **22**, 61
fixpoint 37
$\mathscr{FL}$ 92, 104
$\mathscr{FL}_0$ 79, 86, 92, 97
FMC **68**
formula
propositional **9**
SF(·) . **9**, *see* formula, subformulae, 77
subformulae **9**
frame **17**
$\mathfrak{F}_{\mathrm{total}}$ **18**, 19

G

GAP **26**, 85
GCI . . . . . *see* general concept inclusion
general concept inclusion **25**
$\overset{\infty}{\mathrm{G}}$ **22**, 61
GOAP **44**

H

HGAP **84**
Hintikka 32

I

id **14**
IMP **94**
implication 94
individual **23**

K

Kripke structure **17**

L

Lewis knack 29, 77
linear **14**

literal **9**
LOGCFL **11**, 99
LOGSPACE **10**
logtime-uniform **13**, 94
LTL^+ **47**, 50

M

MAJ **27**
MAJ **44**
ML **17**
ML-SAT 25
model **25**
models 19
quasi . . . . . *see* quasi model
small model theorem 32
monotone **13**, 82
ML-SAT **18**

N

NAE-SAT **44**
N_C . . . . . *see* concept, set of all
NC^1 12
NC^i **11**
negation normal form . **9**, 35, 36, 70, 78
NLOGTIME 12
normalization 81
NP **10**
N_R . . . . . *see* role, set of all roles

O

OCSAT **26**, 88
ontology **25**
oracle Turing machine . . . . . *see* Turing machine
OSAT **26**, 88

P

P **10**, 11, 99
PARITY **27**, **43**
⊕LOGSPACE **10**, 44, 94, 106
path **19**
formulae **19**
model checking **50**
quantifier **19**
PH **11**, 12
PL **9**
PL(B) **14**
Polynomial time hierarchy . . . . . *see* PH
Post's lattice 16
both constants 68
promise problem 11, **31**
PSPACE **10**, 12
pure temporal operators **19**, 36

Q

QBF-VAL **27**
QBF **27**
quasi model **35**

R

reduction
constant depth **13**, 13
dlt-proj **13**
logspace **11**
release operator **54**
role **23**
set of all roles **24**

S

SAC^1 11
satisfiable **17**
satisfy **25**
SC(·) . . . . . *see* concept, subconcept
self-dual **14**, 80, 94
SF(·) . . . . . *see* formula, subformula
short-representation 14
SIZE-DEPTH$_{\mathcal{B}}(s(n), d(n))$ **11**
state formulae **19**
SUBS **26**
subsume **26**, 93
subsumption **26**
superposition **13**

T

TBox **25**
TC^0 12
TC^i **11**
TCSAT **25**, 88
TM . . . . . *see* Turing machine
T_n^{n+1} **14**
tree-like **34**
TSAT **25**, 82
Turing machine
    alternating 38, 42
    deterministic 10
    nondeterministic 10
    oracle 10

U

UGAP **27**
universal concept . . . . . *see* concept

V

Vars(·) **9**

Lebenslauf

Persönliches

Name	Arne Meier
Geburt	06. Mai 1982 in Hannover

Schulische Laufbahn

1988–1992	Grundschule Kestnerstraße, Hannover
1992–1994	Orientierungsstufe Schulzentrum Lüerstraße, Hannover
1994–2001	Gymnasium Sophienschule, Hannover

Universitäre Laufbahn

2002-2005	Bachelor of Science in Informatik, Leibniz Universität Hannover
2005-2007	Master of Science in Informatik, Leibniz Universität Hannover
2003-2006	Vertreter im Fakultätsfachschaftsrat für Elektrotechnik und Informatik
2007	Ehrung für besondere Leistungen in der studentischen Selbstverwaltung
seit Januar 2008	Wissenschaftlicher Mitarbeiter am Institut für Theoretische Informatik an der Leibniz Universität Hannover

Außerberufliche Laufbahn

2001-2002	Zivildienst an der Sophienklinik in Hannover
25. April 2009	Heirat mit Julia Meier, geb. Chamis