Taming Complex Modal and Description Logics

Zusammenfassung der wissenschaftlichen Arbeiten

vorgelegt dem Rat des
Fachbereichs 3 – Mathematik und Informatik
der Universität Bremen

anstelle einer Habilitationsschrift

von Dr. rer. nat. Thomas Schneider
geboren am 15. Juni 1976 in Leipzig

11. Oktober 2015

Abstract

Modal logics and syntactic variants thereof are widely used in knowledge representation and verification. The availability of modal logics with a suitable trade-off between expressive power and computational complexity is paramount for applications. This thesis reports on our work on "taming" expressive temporal, description, and hybrid logics by identifying computationally well-behaved fragments and studying modularity. Our work is threefold: it consists of systematic complexity studies of sub-Boolean fragments of expressive modal(-like) logics, theoretical and practical studies of module extraction and decomposition of description logic ontologies, and a complexity study of branching-time temporal description logics.

Contents

Chapter 1

Introduction

Modal logic (ML) has its roots in philosophy. It was devised as an extension of classical propositional logic that overcomes some of the paradoxes of classical implication, and it was designed to study the concepts of necessity and possibility. The most influential work connected to modern ML is C. I. Lewis's work on symbolic logic from the 1910s [Lew18, LL32]. Lewis introduced the diamond operator $\Diamond$ and five logical systems based on axiomatizations of varying strength. His work led to the study of what we consider modern modal logics: extensions of classical propositional logic with modalities for speaking and reasoning about concepts such as obligation, belief, knowledge, and temporal successorship. The foundations for the modern model theory of ML were laid by the seminal work of Jónsson and Tarski [JT52a, JT52b], Kripke and Hintikka [Kri59, Kri63a, Kri63b, Hin62], and Lemmon and Scott [LS77].

In addition to their rich philosophical background, modal logics have been found to be very useful in computer science: for the past few decades, they have been playing an important rôle in artificial intelligence. Numerous variants of MLs are in use to represent knowledge and draw inferences; the two most successful groups are certainly the following.

- **Temporal logics** originate from Prior's philosophical work on tense logic [Pri57, Pri67, Pri68]. In the past decades, tense logic was developed into formalisms for specifying and verifying properties of interactive systems, which is a vital component in the design of reliable hardware and software, allowing to systematically check relevant system properties such as correctness, reachability, safety, liveness, fairness. Classical temporal logics include linear temporal logic LTL [Pnu77], computation tree logic CTL [EC82] and CTL* [EH86], and propositional dynamic logic PDL [Pra76]. More recently, real-time and probabilistic variants such as TCTL and PCTL [ACD90, SL94] have been studied. Based on these logics, a wide range of techniques for model checking (the verification of interactive systems) has been developed [CGP99, BK08]. Modern model-checking systems such as SPIN, (Nu)SMV, Uppaal, Kronos, and PRISM [Hol97, Hol04, CCGR00, BDL04, Yov97, KNP04] can deal with large-scale industrial systems of impressive sizes.

- More recently, the large family of **description logics** [BCM$^+$03] has become a successful formalism for representing domain knowledge in logical theories *(ontologies)* and for performing automated reasoning over this knowledge, with or without instance data. Description logics have been developed for this purpose and implemented in early knowledge representation systems [BS85, Neb90a, MDW91, Pel91, Mac91]. Nowadays, the W3C Web Ontology Language OWL[1] [HPSv03], based on the powerful description logic $\mathcal{SROIQ}$ [HKS06a], is a widely accepted standard, and OWL ontologies are used in diverse application areas such as knowledge representation and management, semantic databases,

[1] `http://www.w3.org/TR/owl2-overview`

the semantic web, biomedical informatics, the life sciences, linguistics, the geosciences. Modern ontologies contain up to several hundreds of thousands of logical statements, and highly optimized reasoning systems such as Racer, FaCT++, CEL, Pellet, and HermiT [HM01, TH06, BLS06, SPC+07, MSH09] are able to infer implicit knowledge with impressive efficiency. Initially, the development of description logic was independent of modal logic, but soon it was observed that description logics are notational variants of modal logics [Sch91, DL94, Sch94]. From now on, we will use the term *modal-like logics* for referring to modal and description logics alike.

Though incomparable in success with the above two, **hybrid logics** have gained attention because of their distinguishing ability to refer to states in structures directly. Going back to Prior's and Bull's philosophical work [Pri58, Pri67, Pri68, Bul70], hybrid logics are extensions of modal-like logics with the ability to name single states in structures and to describe specific substructures. These powerful logics could certainly have become established representation and specification languages, were it not for their bad computational properties [ABM99, ABM00, FdS03]. Despite this problem, reasoning with hybrid logics has been implemented in the systems HTab and Spartacus [HA09, GKS10].

Applications usually impose two rivaling requirements on logics used for representation and reasoning: on the one hand, a logic should provide *high expressive power*, allowing to make statements relevant for the respective application domain in a comfortable and natural way. On the other hand, they should allow for *efficient reasoning* – that is, the relevant decision problems (e.g., satisfiability, entailment) should admit efficient procedures that lend themselves easily to implementation. Unsurprisingly, there is a trade-off between these two requirements: with increasing expressivity of a formalism, the computational complexity of its associated reasoning tasks usually increases. A classical example for this trade-off is propositional logic versus first-order logic: while the former has an NP-complete satisfiability problem, satisfiability for the latter is undecidable.

The trade-off between expressivity and computational complexity can be observed particularly well when comparing the wide range of existing description logics: at the "bottom" of this range, there are lightweight DLs such as those from the $\mathcal{EL}$ and DL-Lite families [BBL05, BBL08, CDL+05, ACKZ09a], which do not even contain full propositional logic and admit standard reasoning tasks in polynomial time. At the "top", there are very expressive DLs such as $\mathcal{SROIQ}$ [HKS06a], whose combination of features has been carefully tailored such that satisfiability is still decidable, although with a rather high complexity (N2ExpTime-complete [Kaz08]). $\mathcal{SROIQ}$ as well as $\mathcal{EL}$ and DL-Lite are considered important modeling languages; they form the foundation of OWL and its profiles EL and QL, and many modern ontologies are written in OWL: prominent examples include SNOMED CT, the "Systematized Nomenclature of Medicine, Clinical Terms"[2] [Spa00], which falls within $\mathcal{EL}$, and the NCI Thesaurus [GFH+03]. The NCBO BioPortal ontology repository[3] contains almost 400 biomedical ontologies of varying expressivity and size.

The work reported in this cumulative habilitation thesis approaches the described trade-off from the direction of hard modal-like logics and studies two specific ways to alleviate reasoning: (a) by systematically studying fragments obtained by restricting the Boolean part of the logic, and (b) by advancing logic-based module extraction and modularization techniques. More precisely,

[2] http://www.ihtsdo.org/snomed-ct
[3] http://bioportal.bioontology.org

we start from expressive modal-like logics that have computationally hard or even undecidable standard reasoning problems (satisfiability, model checking, subsumption), and we contribute

in Chapter 2 a systematic study of the computational complexity for syntactic fragments of temporal, description, and hybrid logics obtained by restricting the Boolean operators, which identifies decidable and tractable fragments, and delineates a fine-grained decidability and/or tractability border;

in Chapter 3 a comprehensive study of theoretical and practical aspects of logic-based module extraction and modularization of description logic ontologies written in OWL, fostering the replacement of a large ontology with one or several logically indistinguishable subsets (not only) in order to perform reasoning more efficiently;

in Chapter 4 a pioneering study of new lightweight variants of *temporal description logics*, which allow for expressing temporal knowledge in ontologies, and which notoriously become undecidable when including features such as time-invariant binary relations.

General preliminaries

In Chapters 2 and 4, we will use the standard notions of complexity theory and circuit complexity as defined, e.g., in [Pap94, AB09]. In particular, we will make use of the following standard complexity classes and the known inclusions between them.

$$\mathsf{L} \subseteq \mathsf{NL} \subseteq \mathsf{P} \begin{array}{c} \subseteq \\ \subseteq \end{array} \begin{array}{c} \mathsf{NP} \\ \mathsf{coNP} \end{array} \begin{array}{c} \subseteq \\ \subseteq \end{array} \mathsf{PSPACE} \subseteq \mathsf{EXPTIME} \subseteq \mathsf{NEXPTIME} \subseteq \mathsf{2EXPTIME} \subseteq \mathsf{N2EXPTIME} \subset \mathsf{coRE}$$

Problems that are in no $k\mathsf{EXPTIME}$ are called *inherently nonelementary*. Unless stated otherwise, our hardness and completeness results will be based on logarithmic-space and polynomial-time reductions $\leq_{\mathsf{m}}^{\mathsf{log}}$ and $\leq_{\mathsf{m}}^{\mathsf{p}}$, and their corresponding equivalences $\equiv_{\mathsf{m}}^{\mathsf{log}}$ and $\equiv_{\mathsf{m}}^{\mathsf{p}}$.

Chapter 2

Complexity of Sub-Boolean Fragments

2.1 Introduction

As announced in Chapter 1, this chapter will report on a systematic study of the computational complexity for syntactic fragments of temporal, description, and hybrid logics obtained by restricting the Boolean operators, which will identify decidable and tractable fragments, and delineate a fine-grained decidability and/or tractability border.

One obvious choice for obtaining syntactic fragments of expressive modal-like logics would be to restrict the available *modal-like* features (modal operators; DL features such as cardinality restrictions, inverse roles, nominals, complex role hierarchies; hybrid binders) or their interaction. The literature contains an extensive account of the effects of allowing certain subsets of modal-like features on the decidability and complexity of reasoning for temporal, description, and hybrid logic [SC85, HB91, CDLN01, Tob01, BCM$^+$03, HKS06a, HKS06b, Kaz08, ABM99, ABM00]. Consequently, there is generally a clear understanding of *good* and *bad* modal-like features or combinations thereof – those whose presence tends to have mild effects on the complexity of reasoning, and those which notoriously lead to undecidability or intractability. Some of these results seem to imply that, in order to obtain computationally easy fragments, one would have to completely avoid certain bad operators.

However, if we want to keep the bad modal-like operators in order to benefit from their expressive power, we can try to restrict other features of the logic, such as the *Boolean operators* present. This way, we can reduce the expressive power stemming from *interactions* between modal-like and Boolean operators without completely forgoing the bad operators. In this chapter, we thus systematically study the effects of allowing arbitrary sets of *Boolean operators* in temporal, hybrid, and description logics on the computational behavior of these logics. The tractable lightweight description logic $\mathcal{EL}$ already mentioned in Chapter 1 is a prominent example for a computationally well-behaved sub-Boolean fragment of a reasonably expressive modal-like logic (in this case, the description logic $\mathcal{ALC}$). $\mathcal{EL}$ allows conjunction and the constant 1 as the only Boolean operators and the existential quantifier as the only modal-like feature; a number of its extensions with further modal-like features remain tractable [Bra04, BBL05, BBL08]) and are thus computationally well-behaved fragments of expressive description logics in the sense of this chapter.

Rather than considering specific combinations of Boolean operators separately, we study *all* combinations (with certain closure properties). This will result in a classification of the decidability and complexity of the respective decision problem for an infinite family of fragments for each logic. Our study will not only identify all cases in this framework where the respective decision problem is decidable or even tractable; it will also provide a better insight into the sources of hardness by identifying the combinations of modal-like and Boolean operators that lead to computationally hard or even undecidable fragments.

More precisely, we classify the decidability and computational complexity of the standard decision problems for the following logics.

1. Linear Temporal Logic (LTL) and its satisfiability and model-checking problem, which are PSPACE-complete when all Boolean and temporal operators are allowed [SC85] (Section 2.3);
2. The basic DL $\mathcal{ALC}$ and several variants of its standard consistency and concept satisfiability problems,[1] all of which are EXPTIME-complete when all Boolean operators are allowed [Sch94, DM00] (Section 2.4);
3. Hybrid logic (HL) with the downarrow binder $\downarrow$ and its standard satisfiability problem over a number of classes of Kripke structures, which is undecidable over all Kripke structures when all Boolean operators are allowed [BS95, ABM99] (Section 2.5).

In case 2, our results can be seen as a systematic underpinning of the folklore knowledge that the $\mathcal{EL}$ and DL-Lite families are the only reasonably useful $\mathcal{ALC}$ fragments whose standard reasoning tasks relative to unrestricted TBoxes are tractable.

Related work. The effect of Boolean restrictions on the complexity of a logic was first considered in this systematic way for the case of satisfiability for propositional logic by H. Lewis in [Lew79]. He established a dichotomy: depending on the set of Boolean operators, satisfiability is either NP-complete or decidable in polynomial time. Lewis's classification is complete in terms of restrictions on the Boolean operators. It follows the structure of Post's lattice of all closed sets of Boolean functions [Pos41], which captures the multitude of all sets of Boolean operators in a strong sense, as we will explain in Section 2.2.

Since Lewis's seminal study, Post's lattice has been used for systematically studying the complexity of various decision problems for sub-Boolean fragments of classical and non-classical logics and constraint satisfaction problems, among them the problems of counting solutions [RW00], learnability [Dal00], deciding equivalence [Rei01], finding minimal solutions [RV03], circumscription [Nor05, Tho12], the formula value problem [Sch07], and abduction [CST12] for propositional logic. For modal logic, Bauland et al. established a trichotomy for the satisfiability problem over several standard classes of Kripke structures: depending on the allowed Boolean operators, the problem is PSPACE-complete, CONP-complete, or in P [BHSS06, HSS10]. A systematic study of this kind has also been applied to various decision and enumeration problems from default logic [CHS07, BMTV12], autoepistemic logic [CMVT12] and for constraint satisfaction problems [BCC$^+$04, SS07, SS08, ABI$^+$09, BH09, BBC$^+$10].

We analyze the same systematic Boolean restrictions for the logics and decision problems listed above combined with restrictions to the modal-like operators. We study decidability and computational complexity of these infinitely many problems.

Bibliographic notes. The results on LTL in Section 2.3 are from [BSS$^+$09] and [BMS$^+$11]; those on $\mathcal{ALC}$ in Section 2.4 appeared in [MS13]; Section 2.5 on hybrid logic was published in [MMS$^+$10] and [GMM$^+$12].

[1] Consistency and satisfiability are no longer equivalent when certain Boolean operators are omitted.

2.2 Boolean Operators and Post's Lattice

A *Boolean function* is a function $f : \{0,1\}^n \to \{0,1\}$. We identify an n-ary Boolean operator c with the n-ary Boolean function f as follows: $f(a_1,\dots,a_n) = 1$ if and only if the formula $c(x_1,\dots,x_n)$ becomes true when assigning a_i to x_i for all $1 \le i \le n$. Given a modal-like logic $\mathcal{L}$ that contains propositional logic, we consider arbitrary sub-Boolean fragments $\mathcal{L}^-$ of $\mathcal{L}$, each of which allows only a certain set of Boolean operators, which can be nested arbitrarily. The set of operators expressible in each such $\mathcal{L}^-$ thus corresponds to a set B of Boolean functions with the following properties.

1. B is *closed under superposition*, that is, if B contains an n-ary function f m-ary functions $g_1,\dots,g_n$, then it also contains the m-ary function h defined by
$$h(a_1,\dots,a_m) = f(g_1(a_1,\dots,a_m),\ \dots,\ g_n(a_1,\dots,a_m)).$$
2. B contains the *projections* (identities) id^n_k, where $1 \le k \le n$, defined by
$$\mathsf{id}^n_k(a_1,\dots,a_n) = a_k.$$

These two properties are necessary to represent nesting of Boolean operators corresponding to the functions in B: for example, if we allow binary disjunction $\wedge_2$ as an operator, then we implicitly allow all disjunctions $\wedge_n$ of arbitrary arity. The Boolean functions and_n corresponding to $\wedge_n$ are obtained from the binary function and_2 via superposition, involving projections:

$$and_3(a_1,a_2,a_3) = and_2\Big(\mathsf{id}^3_1(a_1,a_2,a_3),\ and_2(\mathsf{id}^3_2(a_1,a_2,a_3),\ \mathsf{id}^3_3(a_1,a_2,a_3))\Big) \quad \text{etc.}$$

We call a set of Boolean functions that satisfies Properties 1 and 2 a *clone* [Pip97]. Given a set B of Boolean functions, we denote with $[B]$ the smallest clone containing B and call B a *base* for $[B]$. It is clear that the set of Boolean operators expressible in a fragment $\mathcal{L}^-$ as given above corresponds to a clone. For example, if we allow binary conjunction as the only explicit Boolean operator in $\mathcal{L}^-$, then the corresponding clone is $\mathsf{E_2} = [\{and_2\}]$. There are infinitely many clones, and Emil Post [Pos41] established their lattice and found a finite base for each clone.

In order to present Post's lattice, we first need to define some specific Boolean functions and properties of Boolean functions. From now on, we deliberately use operator symbols for the corresponding functions, for example $\neg$ for the unary negation function, $\wedge, \vee$ for the binary conjunction and disjunction, and $\oplus$ for the binary exclusive-or function, i.e., $a_1 \oplus a_2 = 1$ if and only if $a_1 \neq a_2$. We denote with c^n_a the n-ary constant function defined by $c^n_a(a_1,\dots,a_n) = a$. For $c^1_1(a)$ and $c^1_0(a)$ we simply write 1 and 0.

Definition 1. Let f be an n-ary Boolean function; let $a \in \{0,1\}$ and $m \ge 2$.

- f is *a-reproducing* if $f(a,\dots,a) = a$.
- f is *monotone*[2] if $a_1 \le b_1,\dots,a_n \le b_n$ implies $f(a_1,\dots,a_n) \le f(b_1,\dots,b_n)$.
- f is *a-separating* if there is some $i \in \{1,\dots,n\}$ such that $f(a_1,\dots,a_n) = a$ implies $a_i = 1$.
- f is *a-separating of degree m* if, for all $U \subseteq \{0,1\}^n$ with $|U| = m$, the following hold: if $f(a_1,\dots,a_n) = a$ for all $(a_1,\dots,a_n) \in U$, then there is some $i \in \{1,\dots,n\}$ such that $a_i = a$ for all $(a_1,\dots,a_n) \in U$.
- f is *self-dual* if $f \equiv \mathsf{dual}(f)$, where $\mathsf{dual}(f)(a_1,\dots,a_n) = \neg f(\neg a_1,\dots,\neg a_n)$.

- f is *linear* if $f \equiv a_1 \oplus \cdots \oplus a_n \oplus c$ for a constant $c \in \{0, 1\}$ and variables $a_1, \ldots, a_n$.

In Table 1 we define all clones and give Post's bases [Pos41] for them. Post's lattice is given in Figure 1. Now Lewis's dichotomy can be described using Post's lattice as follows, denoting with $\mathsf{SAT}(B)$ the satisfiability problem of the fragment of propositional logic that allows only Boolean operators corresponding to a given set B of Boolean functions.

Theorem 2. [Lew79] $\mathsf{SAT}(B)$ *is* NP*-complete if* $\mathsf{S}_1 \subseteq [B]$ *and solvable in polynomial time otherwise.*

In a similar fashion, Bauland et al.'s trichotomy for the basic modal logic K can be described as follows. For a nonempty subset $M \subseteq \{\Diamond, \Box\}$ of the two standard modal operators $\Diamond$ ("possibly") and $\Box$ ("necessarily"), and a set B of Boolean functions, let $\mathsf{K}_M(B)$-SAT be the satisfiability problem for the fragment of K that allows only the modal operators in M and the Boolean operators corresponding to B.

Theorem 3. [BHSS06, HSS10] $\mathsf{K}_{\Diamond,\Box}(B)$-SAT *is*

- PSPACE*-complete if* $\mathsf{S}_{11} \subseteq [B]$*,*
- coNP*-complete if* $\mathsf{E}_0 \subseteq [B] \subseteq \mathsf{E}$*, and*
- *solvable in polynomial time otherwise.*

$\mathsf{K}_\Diamond(B)$-SAT *and* $\mathsf{K}_\Box(B)$-SAT *are* PSPACE*-complete if* $\mathsf{S}_1 \subseteq [B]$ *and solvable in polynomial time otherwise.*

Bauland et al.'s result is more general and captures multiple modalities, formulas given by circuits, and different modal logics such as KD, T, S4, S5, i.e., the modal logics of serial, reflexive, transitive, and complete frames, respectively.

2.3 Linear Temporal Logic

Linear Temporal Logic (LTL) was introduced by Pnueli in [Pnu77] as a formalism for reasoning about the properties and the behaviors of parallel programs and concurrent systems, and has widely been used for these purposes. The standard reasoning tasks include satisfiability and model checking. The former asks whether, given an LTL formula φ, there is a linear path at whose starting point φ is true; the latter's existential version additionally receives as input a Kripke structure $\mathcal{M}$ and a state s and asks whether φ is true at *some* path starting at s in $\mathcal{M}$. Validity and universal model checking are additional standard reasoning tasks; however, are not in the scope of our work.

Sistla and Clarke were the first to study the computational complexity of satisfiability and (existential) model checking. For full LTL with the operators F (eventually), G (invariantly), X (next-time), U (until), and S (since), they showed that both problems are PSPACE-complete; for some restrictions to the temporal operators – allowing only {X} or at most {F, G} –, they established NP-completeness [SC85]. They also showed that the restriction to atomic negation

[2] Monotone operators and the corresponding fragments of logics are often called *positive*, partly for historic reasons, partly in order to avoid false associations with the distinction between logics with monotonic and non-monotonic *inference*. All logics studied in this chapter have monotonic inference, and some of their fragments contain only monotone (positive) Boolean operators. We have decided to prefer "monotone" to "positive" because of the established clone name M.

Name	Definition	Base(s)
BF	All Boolean functions	$\{\vee, \wedge, \neg\}$
R_0	$\{f \in BF \mid f$ is 0-reproducing $\}$	$\{\wedge, \oplus\}$
R_1	$\{f \in BF \mid f$ is 1-reproducing $\}$	$\{\vee, \leftrightarrow\}$
R_2	$R_1 \cap R_0$	$\{\vee, x \wedge (y \leftrightarrow z)\}$
M	$\{f \in BF \mid f$ is monotone $\}$	$\{\vee, \wedge, 0, 1\}$
M_0	$M \cap R_0$	$\{\vee, \wedge, 0\}$
M_1	$M \cap R_1$	$\{\vee, \wedge, 1\}$
M_2	$M \cap R_2$	$\{\vee, \wedge\}$
S_0^n	$\{f \in BF \mid f$ is 0-separating of degree $n\}$	$\{\rightarrow, h_n\}$
S_0	$\{f \in BF \mid f$ is 0-separating$\}$	$\{\rightarrow\}$
S_1^n	$\{f \in BF \mid f$ is 1-separating of degree $n\}$	$\{x \wedge \overline{y}, \mathsf{dual}(h_n)\}$
S_1	$\{f \in BF \mid f$ is 1-separating$\}$	$\{x \wedge \overline{y}\}$
S_{02}^n	$S_0^n \cap R_2$	$\{x \vee (y \wedge \overline{z}), h_n\}$
S_{02}	$S_0 \cap R_2$	$\{x \vee (y \wedge \overline{z})\}$
S_{01}^n	$S_0^n \cap M$	$\{h_n, 1\}$
S_{01}	$S_0 \cap M$	$\{x \vee (y \wedge z), 1\}$
S_{00}^n	$S_0^n \cap R_2 \cap M$	$\{x \vee (y \wedge z), h_n\}$
S_{00}	$S_0 \cap R_2 \cap M$	$\{x \vee (y \wedge z)\}$
S_{12}^n	$S_1^n \cap R_2$	$\{x \wedge (y \vee \overline{z}), \mathsf{dual}(h_n)\}$
S_{12}	$S_1 \cap R_2$	$\{x \wedge (y \vee \overline{z})\}$
S_{11}^n	$S_1^n \cap M$	$\{\mathsf{dual}(h_n), 0\}$
S_{11}	$S_1 \cap M$	$\{x \wedge (y \vee z), 0\}$
S_{10}^n	$S_1^n \cap R_2 \cap M$	$\{x \wedge (y \vee z), \mathsf{dual}(h_n)\}$
S_{10}	$S_1 \cap R_2 \cap M$	$\{x \wedge (y \vee z)\}$
D	$\{f \mid f$ is self-dual$\}$	$\{x\overline{y} \vee x\overline{z} \vee (\overline{y} \wedge \overline{z})\}$
D_1	$D \cap R_2$	$\{xy \vee x\overline{z} \vee y\overline{z}\}$
D_2	$D \cap M$	$\{xy \vee yz \vee xz\}$
L	$\{f \mid f$ is linear$\}$	$\{\oplus, 1\}$
L_0	$[\{\oplus\}]$	$\{\oplus\}$
L_1	$L \cap R_1$	$\{\leftrightarrow\}$
L_2	$L \cap R_2$	$\{x \oplus y \oplus z\}$
L_3	$L \cap D$	$\{x \oplus y \oplus z \oplus 1\}$
V	$\{f \mid f$ is an n-ary *or*-function or a constant function$\}$	$\{\wedge, 0, 1\}$
V_0	$[\{\vee\}] \cup \{0\}$	$\{\vee, 0\}$
V_1	$[\{\vee\}] \cup \{1\}$	$\{\vee, 1\}$
V_2	$[\{\vee\}]$	$\{\vee\}$
E	$\{f \mid f$ is an n-ary *and*-function or a constant function$\}$	$\{\wedge, 0, 1\}$
E_0	$[\{\wedge\}] \cup \{0\}$	$\{\wedge, 0\}$
E_1	$[\{\wedge\}] \cup \{1\}$	$\{\wedge, 1\}$
E_2	$[\{\wedge\}]$	$\{\wedge\}$
N	$[\{\neg\}] \cup \{0\} \cup \{1\}$	$\{\neg, 0\},\ \{\neg, 1\}$
N_2	$[\{\neg\}]$	$\{\neg\}$
I	$\{f \mid f$ is a projection$\} \cup \{0, 1\}$	$\{0, 1\}$
I_0	$\{f \mid f$ is a projection$\} \cup \{0\}$	$\{0\}$
I_1	$\{f \mid f$ is a projection$\} \cup \{1\}$	$\{1\}$
I_2	$\{f \mid f$ is a projection$\}$	$\emptyset$

The auxiliary function h_n is defined by $h_n(x_1, \ldots, x_{n+1}) = \bigwedge_{i=1}^{n+1} x_1 \vee x_2 \vee \cdots \vee x_{i-1} \vee x_{i+1} \vee \cdots \vee x_{n+1}$.

Table 1: List of all closed classes of Boolean functions with bases, taken from [BCRV03]

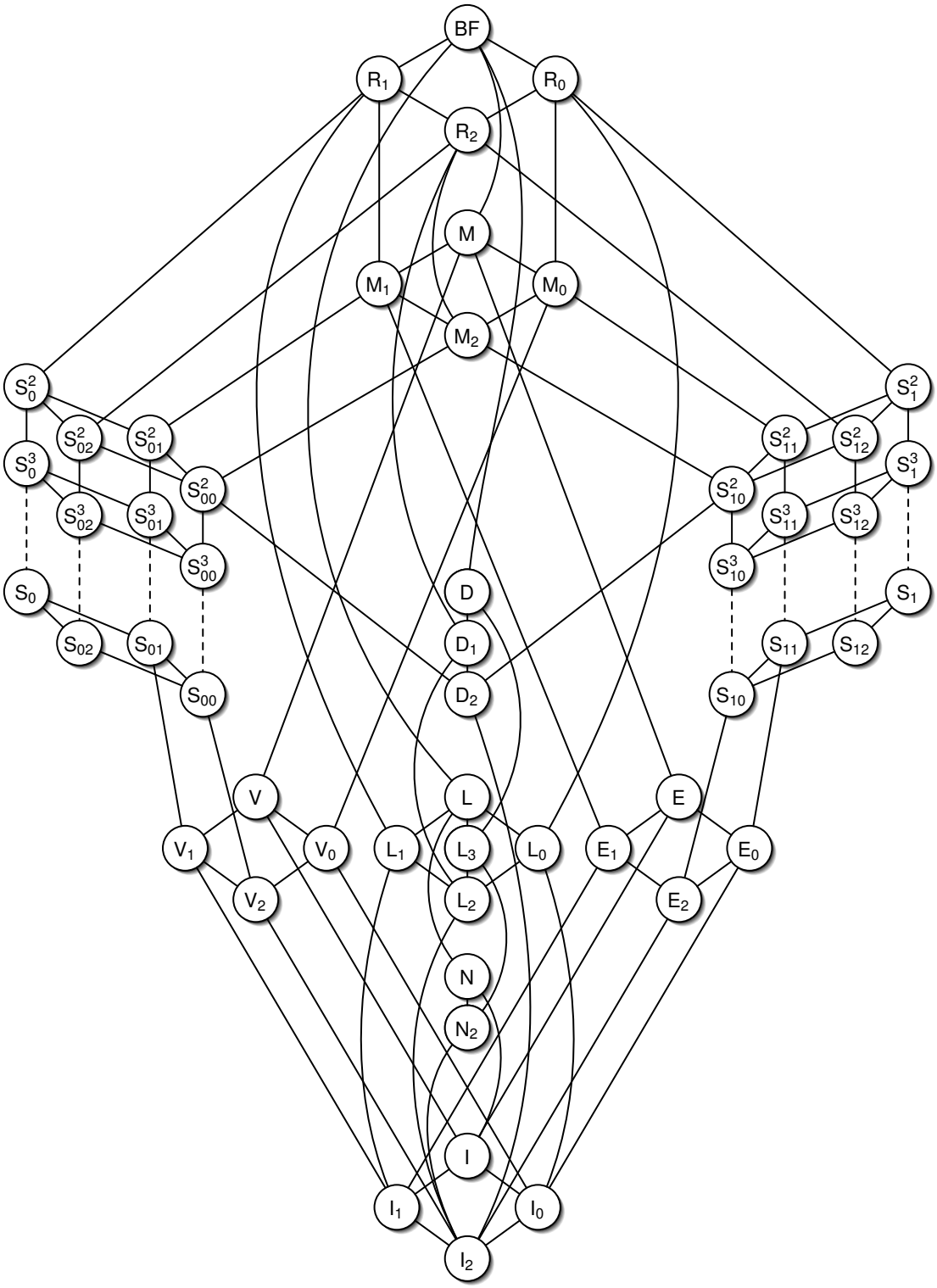


Figure 1: Post's lattice

leads to NP-completeness in the case of {F, X}. These results imply that, under reasonable complexity-theoretic assumptions, reasoning with LTL is not tractable.

Subsequently, the effect of restricting the set of temporal operators on the complexity of the satisfiability and model-checking for LTL was studied more systematically in the literature, together with *single* restrictions of the Boolean operators: Demri and Schnoebelen [DS02] investigated satisfiability and existential model checking for restrictions on the set of temporal operators, their nesting, and the number of atomic propositions. Markey [Mar04] studied the complexity of all four standard decision problems for LTL fragments with various subsets of temporal operators (including the past versions of F and X), together with further restrictions on the interaction between future and past operators, and between temporal operators and negation. He showed that these fragments are (CO)NP- or PSPACE-complete, respectively. Muscholl and Walukiewicz [MW05] showed that satisfiability for the LTL variant that allows {F, G} together with a version of X "guarded" by atomic propositions is NP-complete – in contrast to PSPACE-completeness for the fragment {F, X} [SC85]. Since only Demri and Schnoebelen exhibit tractable fragments at all, it can be concluded that a multitude of LTL fragments have an intractable satisfiability and/or model-checking problem. In fact, not even the restriction to Horn formulas leads to a decrease in complexity of satisfiability for LTL, as shown earlier by Chen and Lin [CL93], and Dixon et al. [DFR00].

Fragments of related temporal logics have been investigated too: Emerson et al. [EES90] studied three fragments of computation tree logic (CTL) with temporal and Boolean restrictions and showed tractability or NP-completeness. Hemaspaandra [Hem01] showed that satisfiability for modal logic over linear frames drops from NP-complete to tractable if Boolean operators are restricted to conjunction and atomic negation. Finally, one of the first to systematically study the complexity of fragments of modal-like logics was Halpern [Hal95]. He investigated satisfiability for multimodal logics, bounding the depth of modal operators and the number of atomic propositions. Only the combination of both restrictions led to tractable fragments.

2.3.1 Basic Notions

Let PROP be a countable set of propositional variables, B be a finite set of Boolean functions, and T be a set of temporal operators. The set of *temporal B-formulas over T* is defined inductively by the grammar

$$\varphi ::= p \mid \circ_f(\varphi, \ldots, \varphi) \mid \tau(\varphi, \ldots, \varphi),$$

where $p \in \mathsf{PROP}$, $\circ_f$ is a Boolean operator (of the appropriate arity) corresponding to a function $f \in B$, and τ is a temporal operator (of the appropriate arity) from T. We consider the unary temporal operators X (next-time), F (eventually), G (invariantly) and the binary temporal operators U (until), R (release) and S (since). The set of all temporal B-formulas over T is denoted by $\mathsf{LTL}_T(B)$.

LTL-formulas are interpreted over infinite paths of states, which intuitively can be seen as different points of time, with propositional assignments. It is common to consider potentially infinite sets of paths, represented by a class of finite Kripke structures, called transition systems. A *transition system* is a triple $K = (W, R, \eta)$, where W is a finite set of states, $R \subseteq W \times W$ is a total binary relation (i.e., for each $a \in W$, there is some $b \in W$ such that aRb), and $\eta : W \to 2^{\mathsf{PROP}_K}$ for a finite set $\mathsf{PROP}_K \subseteq \mathsf{PROP}$ of variables. A *path* π in K is an infinite sequence denoted as $(\pi_0, \pi_1, \ldots)$, where, for all $i \geq 0$, $\pi_i \in W$ and $\pi_i R \pi_{i+1}$.

For a transition system $K = (W, R, \eta)$, a path π in K, and a temporal B-formula over the

temporal operators $\{\mathsf{F}, \mathsf{G}, \mathsf{X}, \mathsf{U}, \mathsf{R}, \mathsf{S}\}$ with variables from PROP_K, we define in Figure 2 what it means that π *satisfies* φ *in* π_i, denoted by $\pi, i \models \varphi$.

$\pi, i \models x$	iff	$x \in \eta(\pi_i), \qquad x \in \mathsf{PROP}_K$
$\pi, i \models \circ_f(\varphi_1, \ldots, \varphi_n)$	iff	$f(t_1, \ldots, t_n) = 1$, where $t_j = 1$ if $\pi, i \models \varphi_j$, and $t_i = 0$ otherwise
$\pi, i \models \mathsf{X}\varphi_1$	iff	$\pi, i+1 \models \varphi_1$
$\pi, i \models \mathsf{F}\varphi_1$	iff	$\pi, j \models \varphi_1$ for some $j \geq i$
$\pi, i \models \mathsf{G}\varphi_1$	iff	$\pi, j \models \varphi_1$ for all $j \geq i$
$\pi, i \models \varphi_1 \mathsf{U} \varphi_2$	iff	there is an $\ell \geq i$ with $\pi, \ell \models \varphi_2$ and, for every $i \leq j < \ell$, $\pi, j \models \varphi_1$
$\pi, i \models \varphi_1 \mathsf{R} \varphi_2$	iff	for all $\ell \geq i$ with $\pi, \ell \models \varphi_2$, there is some $i \leq j < \ell$ with $\pi, j \models \varphi_1$
$\pi, i \models \varphi_1 \mathsf{S} \varphi_2$	iff	there is an $\ell \leq i$ with $\pi, \ell \models \varphi_2$ and, for every $\ell < j \leq i$, $\pi, j \models \varphi_1$

Figure 2: Satisfaction relation for LTL

Some temporal operators are often regarded as abbreviations of others because, for example, $\mathsf{F}\varphi = \mathit{true}\,\mathsf{U}\varphi$, $\mathsf{G}\varphi = \neg\mathsf{F}\neg\varphi$, and $\varphi\mathsf{R}\psi = \neg((\neg\varphi)\mathsf{U}(\neg\psi))$. However, these abbreviations rely on certain Boolean operators, which may not always be present in B.

A path π is said to *satisfy* a given formula φ if $\pi, i \models \varphi$ for some position i on φ. A formula φ is called *satisfiable* if some path π satisfies φ, and φ is *satisfiable in a state* $w \in W$ *of a transition system* $K = (W, R, \eta)$ if, for some path π in K starting at w, we have $\pi, 0 \models \varphi$. We consider the following decision problems, for every finite set B of Boolean functions and every set T of temporal operators.

Satisfiability, $\mathsf{LTL}_T(B)$-SAT. Given a temporal B-formula φ over T, is φ satisfiable?

Model Checking, $\mathsf{LTL}_T(B)$-MC. Given a temporal B-formula φ over T, a transition system $K = (W, R, \eta)$ and a state $w \in W$, is φ satisfiable in w?

In the literature, a variant of the satisfiability problem is sometimes considered, where we ask if a formula can be satisfied at the *initial* state of a path. For all fragments without S, this variant is computationally equivalent to ours.

Our version of the model-checking problem is also called *existential*. The *universal* one, in contrast, asks whether *all* paths starting at w in K satisfy φ [Mar04]. In the following, we will omit this distinction because we will only consider the existential version.

For studying the complexity of the two decision problems, we need to make a few commitments. First, we assume that the input formula is represented as a string, not as a circuit or DAG, which would allow for more succinctness and would certainly affect some of the complexity bounds proven. Second, the complexity of $\mathsf{LTL}_T(B)$-MC is measured in the sum of the sizes of all three inputs φ, K, w. Third, we consider simple representations of structures which contain an entry for each state and one for each edge (pair of states), as opposed to more condensed representations.

In our notation, Sistla and Clarke's fundamental result can be formulated as follows.

Theorem 4. [SC85] $\mathsf{LTL}_T(\wedge, \neg)$-SAT *and* $\mathsf{LTL}_T(\wedge, \neg)$-MC *are*

- PSPACE*-complete if* $\{\mathsf{U}\} \subseteq T$ *or* $\{\mathsf{F}, \mathsf{X}\} \subseteq T$*, and*
- NP*-complete if* $T \subseteq \{\mathsf{F}, \mathsf{G}\}$ *or* $T = \{\mathsf{X}\}$.

2.3.2 Satisfiability

Using Post's lattice, we examine the satisfiability problem for every possible fragment of LTL determined by an arbitrary set of Boolean operators *and* any subset of the five temporal operators $\{\mathsf{F}, \mathsf{G}, \mathsf{X}, \mathsf{U}, \mathsf{S}\}$ studied by Sistla and Clarke. We determine the computational complexity of these problems, showing that all cases – except for two sets of Boolean operators – are either PSPACE-complete, NP-complete, or in P.

Among our results, we exhibit cases with nontrivial tractability as well as the smallest possible sets of Boolean and temporal operators that already lead to NP-completeness or PSPACE-completeness, respectively. Examples for the first group are cases in which only the unary *not* function, or only monotone functions are allowed, but there is no restriction on the temporal operators. As for the second group, if only the binary function f with $f(x, y) = (x \wedge \overline{y})$ is permitted, then satisfiability is NP-complete already in the case of propositional logic [Lew79]. Our results show that the presence of the same function f separates the tractable languages from the NP-complete and PSPACE-complete ones, depending on the set of temporal operators used. According to this, minimal sets of temporal operators leading to PSPACE-completeness together with f are, for example, $\{\mathsf{U}\}$ and $\{\mathsf{F}, \mathsf{X}\}$.

Our results are formulated in Theorem 5 and depicted in Figure 10 (Appendix C).

Theorem 5. $\mathsf{LTL}_T(B)$-SAT *is*

- NP-*complete if* $\mathsf{S}_1 \subseteq [B]$ *and* $T \subseteq \{\mathsf{F}, \mathsf{G}\}$ *or* $T = \{\mathsf{X}\}$,
- PSPACE-*complete if* $\mathsf{S}_1 \subseteq [B]$ *and* T *is not covered by the previous case, and*
- *solvable in polynomial time in all other cases except for* $[B] = \mathsf{L}_0$ *or* $[B] = \mathsf{L}$.

The technically most involved proof is that of PSPACE-hardness in the case $T = \{\mathsf{S}\}$. The difficulty lies in simulating the quantifier tree of a Quantified Boolean Formula (QBF) in a linear structure. We do this in three steps: first, we partition a finite prefix of a path π into 2^n subsequent intervals, each of which satisfies a distinct combination of truth values of the n universally quantified variables occurring in the given QBF. This requires an LTL-subformula using $\wedge$, $\neg$, and a constant number of S-operators. Second, we construct another LTL-subformulas using $\wedge$, $\vee$, $\neg$, and S, which sets the values for the existentially quantified variables in the previously delineated intervals. Finally, we rewrite all occurrences of $\wedge$, $\vee$, $\neg$ using polynomially-sized formulas over the base $f(x, y) = (x \wedge \overline{y})$ of S_1, using a standard technique that goes back to a result of Lewis's [Lew79].

The two missing cases $[B] = \mathsf{L}_0$ or $[B] = \mathsf{L}$ are based on the binary *xor* function and have already defied classification in [BHSS06, HSS10] for modal logics of reflexive frames classes. Due to reflexivity, which is implicit in the semantics of LTL, a formula $\mathsf{F}\varphi$ is satisfied at some state whenever φ is. This prohibits attempts to treat the propositional part of a formula separately from the "remainder" when trying to find decision procedures. On the other hand, the restricted expressivity of *xor* makes it difficult to prove lower bounds.

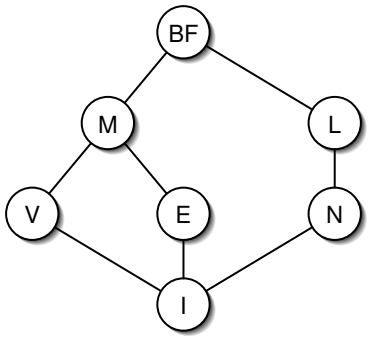
The results in Theorem 5 establish a homogeneous complexity landscape and reveal a clear borderline between tractable and intractable fragments: the intractable[3] cases are characterized by the ability to express the Boolean function $f(x, y) = x \wedge \overline{y}$, independently of the set of temporal operators allowed. The further separation between NP- and PSPACE-completeness is determined solely by the temporal operators allowed.

[3] Throughout this thesis, we assume $\mathsf{P} \neq \mathsf{NP}$.

2.3.3 Model Checking

Using Post's lattice, we examine the existential model-checking problem for every possible fragment of LTL determined by an arbitrary set of Boolean operators *and* any subset of the five temporal future operators {F, G, X, U, R}. We separate the model-checking problem for almost all of these fragments into tractable (here: polynomial-time solvable) and intractable (here: NP-hard or PSPACE-hard) cases. In contrast to earlier work discussed above, we exhibit many tractable fragments, and they are even in NL (nondeterministic logarithmic space). As for satisfiability, we had to leave open the case of the binary *xor*-operator.

For the model-checking problem, the Boolean constants 0, 1 are irrelevant because they can easily be simulated using fresh propositional variables that are interpreted as false or true in every state of the input transition system. Hence the following holds.



Lemma 6. *Let B be a finite set of Boolean functions and T be a set of temporal operators. Then* $\mathsf{LTL}_T(B \cup \{0,1\})\text{-MC} \equiv_m^{\log} \mathsf{LTL}_T(B)\text{-MC}$.

Consequently, it suffices to formulate results only for clones with both constants, and they will carry over to the corresponding clones with at most one constant. The figure on the right shows all clones with both constants and their inclusion structure.

Our results will exhibit a complexity landscape that is much less homogeneous than in the satisfiability case, inducing a more diffuse tractability borderline. In particular, unlike in the satisfiability case, there are sets of Boolean operators that lead to both tractable and intractable model-checking problems in the presence of different sets of temporal operators. Due to this effect, we have had to establish more, and less uniform, complexity results, i.e., most of those apply to only a few combinations of operators. For this reason, we have put a much stronger focus on establishing tractability versus intractability, abstaining from proving upper bounds for the intractable cases.

We present the results in Table 2. The top row refers to the clones from above. Entries "NL-c" denote completeness for NL under logspace many-one reductions; all other entries denote hardness results for NP and PSPACE. The column "BF" is due to [SC85]; the remaining entries are our own results.

Our most surprising intractability result is the NP-hardness of the fragment that only allows the temporal operator U (respectively only its dual R) and no propositional operator at all. That is, propositional satisfiability can be encoded by a suitable combination of a transition system and an LTL formula using only U (or R) and no propositional operators at all.

Our technically most complex result is the NL-completeness for the combination of F, G with $\vee$. It relies on being able to verify two properties simultaneously: whether a formula is true at the initial state of a path, and whether it is true in *all* states. The variant "in all states" is required to recursively treat subformulas starting with G. The corresponding nondeterministic algorithm has a special recursive nature that allows it to be implemented in logarithmic space. This result is our most surprising tractability result because the combination of F, G with $\vee$ features universal and existential operators at the same time. Given that the combination of F with $\wedge$ is already NP-hard (a consequence of the results in [SC85]), we would have expected the same lower bound already for the dual combination of G with $\vee$. The expected duality eventually occurs in the presence of

B / T	I	N	E	V	M	L	BF
X	**NL-c**	**NL-c**	**NL-c**	**NL-c**	NP-h	**NL-c**	NP-h
G	**NL-c**	**NL-c**	**NL-c**	**NL-c**	NP-h	?	NP-h
F	**NL-c**	**NL-c**	NP-h	**NL-c**	NP-h	?	NP-h
FG	**NL-c**	**NL-c**	NP-h	**NL-c**	NP-h	?	NP-h
FX	**NL-c**	**NL-c**	NP-h	**NL-c**	NP-h	?	PSPACE-h
GX	**NL-c**	**NL-c**	**NL-c**	NP-h	PSPACE-h	?	PSPACE-h
FGX	**NL-c**	**NL-c**	NP-h	NP-h	PSPACE-h	?	PSPACE-h
all other combinations (i.e., with U or R)	NP-h	NP-h	NP-h	NP-h	PSPACE-h	NP-h	PSPACE-h

Table 2: Results for $\mathsf{LTL}_T(B)$-MC. Hardness is indicated by "h", completeness by "**c**".

the X-operator: the cases $T = \{\mathsf{F}, \mathsf{X}\}$ and $B = \{\wedge\}$, as well as $T = \{\mathsf{G}, \mathsf{X}\}$ and $B = \{\vee\}$, are both NP-hard.

Corresponding NP and PSPACE upper bounds for the intractable cases are not obvious: Sistla and Clarke's upper bounds have been established only for $B = \{\wedge, \vee, \neg\}$ and do not carry over automatically to arbitrary sets of Boolean operators. More precisely, it is not clear whether $\mathsf{LTL}_T(B)$-MC can be (logspace- or polytime-) reduced to $\mathsf{LTL}_T(\wedge, \vee, \neg)$-MC even though all operators in B can be expressed using $\wedge, \vee, \neg$, the reason being that, for some operators such as the binary *xor* ($\oplus$), the transformation of an $\mathsf{LTL}_T(\oplus)$-formula into an equivalent $\mathsf{LTL}_T(\wedge, \vee, \neg)$-formula may incur an exponential blowup. However, from our experiences with similar studies, we do conjecture completeness.

As indicated above, the borderline between tractable and intractable fragments is rather diffuse. However, it can be observed that intractability largely correlates with the presence of both universal and existential operators:

- All fragments with U are NP- or PSPACE-hard (and the semantics of U already nests a universal into an existential quantifier).
- All fragments that allow all monotone Boolean operators (i.e., $\wedge$ and $\vee$ at the same time) are NP- or PSPACE-hard.
- For the remaining fragments to be intractable, it is necessary (but not sufficient) to combine either $\wedge$ and F, or $\vee$ and G.

2.3.4 Discussion

To our knowledge, our work is the first to systematically study LTL fragments obtained by restricting temporal and Boolean operators simultaneously. In comparison to previous work, we have identified a higher proportion of tractable fragments, making it easier to locate the tractability border. Our results are complete for *all* possible sets of Boolean operators and for all sets of temporal operators that are subsets of either F, G, X, U, S in the case of satisfiability or F, G, X, U, R in the case of model checking.

Our results show that simultaneous restrictions of the temporal and Boolean operators induce more tractable than intractable fragments overall. Of course, many of the tractable (and trivial) fragments are clearly too inexpressive to be taken seriously as specification languages. However,

some tractable fragments are quite expressive, for example those with monotone Boolean operators (in the case of satisfiability) or with only conjunction (in the case of satisfiability and model checking). There is clearly a parallel to lightweight description logics such as $\mathcal{EL}$ (see Section 2.1). It is possible that monotone fragments of LTL are sufficient to formulate specifications in certain applications of verification. These applications could rely on the efficient decision procedures emanating from our results.

We chose the temporal operators in the case of satisfiability because they were considered by Sistla and Clarke [SC85], and in the case of model checking because they are the standard *future* operators. We provide more justifications for this choice in our original work [BMS+11].

For future work, it would be tempting to initiate an even more systematic study that is complete with respect to all possible sets of temporal operators, including past operators (which were studied previously [SC85, Mar04]), but also operators defined by arbitrary LTL formulas, e.g. the ternary operator $\mathsf{O}(\alpha, \beta, \gamma) = \mathsf{F}\alpha \vee (\beta \mathsf{U} \gamma)$, or even operators based on automata [Wol83]. Unfortunately, it is not very realistic to achieve this kind of completeness, because of the combinatorial explosion incurred:

First, even if one adds only the five past counterparts of the above $\mathsf{F}, \mathsf{G}, \mathsf{X}, \mathsf{U}, \mathsf{R}$, the number of fragments to consider will blow up significantly: $2^{10} - 1 = 1023$ instead of $2^5 - 1 = 31$ sets of temporal operators, each combined with a large number of sets of Boolean operators. Given the large number of NL-complete fragments in Table 2 and the fact that many of the proofs for their upper bounds rely on the absence of past operators, we expect that a huge number of additional single theorems would have to be proven. So far, we can at least say that almost all fragments containing the S (since) operator are as hard as the corresponding fragments with U, for the same reasons. However, there are exceptions which are due to the asymmetry that paths have a first, but no last, state [BMS+09]. Still, it would of course be interesting to find out whether the conclusion "past is for free" drawn in [Mar04] extends to sub-Boolean fragments of LTL.

Second, a truly systematic account of all possible temporal operators would have to start with cataloging all definitions of temporal operators in the formalisms mentioned above, analogously to Post's lattice of all Boolean functions. Since these formalisms are much more expressive than propositional logic, it is unclear whether such a research program would be feasible at all (if one considers that Post's lattice already took its author several years to establish).

A different, more feasible, direction for future work is to apply our systematic study to branching-time temporal logics, such as CTL(*) (but satisfiability has already been classified [MMTV09]), to the μ-calculus, which extends both LTL and CTL(*), or even to the hybrid μ-calculus [SV01].

2.4 Description Logic

We already know from Chapter 1 that description logics (DLs) [BCM+03] are a successful family of knowledge representation languages. They are decidable fragments of first-order logic and underlie the W3C Web Ontology Language OWL. The main feature of DLs is the ability to describe concepts (e.g., "a patient who has a history of high blood pressure"), to define new concepts in terms of others (e.g., "HighRiskPatient" in the context of heart diseases), and to express background knowledge (e.g., "a patient who has a history of high blood pressure is someone who has an increased risk of suffering a stroke"). Definitions and background knowledge are specified in terminologies, also called TBoxes, via axioms that relate concept descriptions with each other, such as concept inclusions. The standard reasoning tasks of satisfiability and

subsumption ask whether a given concept description has a model or whether a given concept description implies another (in both cases possibly with respect to a given TBox). They have been studied extensively for various DLs of different expressivity. Their complexity ranges between trivial for fragments of the basic DL $\mathcal{ALC}$ and N2ExpTime for the OWL 2 standard $\mathcal{SROIQ}$. Another factor that determines the complexity is the distinction whether terminological knowledge, general background knowledge, or no TBoxes at all are allowed.

For $\mathcal{ALC}$ with general TBoxes, satisfiability and subsumption are interreducible and ExpTime-complete: the upper bound is due to the correspondence with propositional dynamic logic [Pra78, VW86, DM00], and the lower bound was proved by Schild [Sch94]. To obtain tractable DLs with tractable reasoning problems, specific fragments of $\mathcal{ALC}$ based on restrictions to the allowed Boolean operators and quantifiers has been studied. Notable examples include members of the prominent $\mathcal{EL}$ and DL-Lite families (see Section 2.1). The following is a brief survey of the complexity landscape for such specific sub-Boolean $\mathcal{ALC}$ fragments (it must be observed that, unlike for $\mathcal{ALC}$, the standard reasoning problems are no longer interreducible in the absence of certain Boolean operators).

- $\mathcal{EL}$ allows only conjunctions and existential restrictions [Baa03], and thus satisfiability for $\mathcal{EL}$ is uninteresting because every $\mathcal{EL}$-concept and -TBox is satisfiable. Concept subsumption with or without TBoxes is tractable in $\mathcal{EL}$ [Baa03, Bra04], and it remains tractable under a variety of extensions such as nominals, concrete domains, role chain inclusions, and domain and range restrictions [BBL05, BBL08].
- In contrast, the presence of universal quantifiers usually breaks tractability: subsumption in $\mathcal{FL}_0$, which allows only conjunction and universal restrictions, is coNP-complete [Neb90b]. Relative to TBoxes, the complexity increases to PSpace-complete for cyclic TBoxes [Baa96, Kd03] and ExpTime-complete for general TBoxes [BBL05, Hof05]. In [DLN+92, DLNN97], concept satisfiability and subsumption for several logics below and above $\mathcal{ALC}$ that extend $\mathcal{FL}_0$ with disjunction, negation and existential restrictions and other features, is shown to be tractable, NP-complete, coNP-complete or PSpace-complete.
- ExpTime-hardness of subsumption relative to general TBoxes can be observed already in fragments of $\mathcal{ALC}$ containing either conjunction or disjunction and both existential and universal restrictions [GMWK02], or only conjunction, universal restrictions and unqualified existential restrictions [Don03].
- In the historically first member of the large DL-Lite family, where unqualified existential restrictions, atomic negation on the right-hand side of concept inclusions, as well as inverse and functional roles are allowed, satisfiability is tractable [CDL+05]. Several extensions of DL-Lite are shown to have tractable, NP-complete, or ExpTime-complete satisfiability problems in [ACKZ07, ACKZ09a, ACKZ09b].

DLs from the above families with tractable reasoning problems are called *lightweight DLs*. They are based on specific sets of allowed Boolean operators and quantifiers (and, in some cases, rely on further limitations, such as unqualified existential restrictions). Despite the success of the EL and DL-Lite families, the principled question remains whether it is possible to design lightweight DLs based on different restrictions. By giving a systematic account of the complexity of DLs with restricted Boolean operators and quantifiers, we aim at answering this question. We pursue the same approach as in the previous section and classify satisfiability with respect to TBoxes for $\mathcal{ALC}$ fragments obtained by arbitrary sets of Boolean operators and quantifiers.

2.4.1 Basic Notions

We use the standard syntax and semantics of $\mathcal{ALC}$ [BCM+03], with the Boolean operators $\sqcap$, $\sqcup$, $\neg$, $\top$, $\bot$ replaced by arbitrary operators $\circ_f$ corresponding to Boolean functions $f : \{0,1\}^n \to \{0,1\}$ of arity n. Let $\mathsf{N_C}$, $\mathsf{N_R}$ and $\mathsf{N_I}$ be sets of atomic concepts, roles and individuals, and let B be a set of Boolean functions and $Q \subseteq \{\exists, \forall\}$ a set of quantifiers. Then the set of $\mathcal{ALC}$-concept descriptions using only operators from B and Q, for short (B,Q)-*concepts*, is defined by

$$C ::= A \mid \circ_f(C, \ldots, C) \mid q\, r.C,$$

where $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$, $\circ_f \in B$ is a Boolean operator corresponding to a function $f \in B$, and $q \in Q$ is a quantifier. A *general* (B,Q)*-concept inclusion (*(B,Q)*-GCI)* is an axiom of the form $C \sqsubseteq D$ where C, D are B-concepts. A (B,Q)*-TBox* is a finite set of (B,Q)-GCIs. A (B,Q)*-ABox* is a finite set of axioms of the form $C(a)$ or $r(a,b)$, where C is a (B,Q)-concept, $r \in \mathsf{N_R}$ and $a, b \in \mathsf{N_I}$. A (B,Q)*-ontology* is the union of a (B,Q)-TBox and (B,Q)-ABox (this simplified view suffices for our purposes).

An *interpretation* is a pair $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta^\mathcal{I}$ is a nonempty set and the *interpretation function* $\cdot^\mathcal{I}$ maps every atomic concept to a subset of $\Delta^\mathcal{I}$, every role to a binary relation over $\Delta^\mathcal{I}$, and every individual to an element of $\Delta^\mathcal{I}$. The interpretation function is extended to arbitrary (B,Q)-concepts as follows.

$$\begin{aligned}
\circ_f(C_1, \ldots, C_n)^\mathcal{I} &= \{x \in \Delta^\mathcal{I} \mid f(t_1, \ldots, t_n) = 1\}, \text{ where } t_i = 1 \text{ if } x \in C_i^\mathcal{I}, \text{ and } t_i = 0 \text{ otherwise} \\
\exists R.C^\mathcal{I} &= \{x \in \Delta^\mathcal{I} \mid \{y \in C^\mathcal{I} \mid (x,y) \in R^\mathcal{I}\} \neq \emptyset\} \\
\forall R.C^\mathcal{I} &= \{x \in \Delta^\mathcal{I} \mid \{y \in C^\mathcal{I} \mid (x,y) \notin R^\mathcal{I}\} = \emptyset\}
\end{aligned}$$

An interpretation $\mathcal{I}$ *satisfies* the GCI $C \sqsubseteq D$, written $\mathcal{I} \models C \sqsubseteq D$, if $C^\mathcal{I} \subseteq D^\mathcal{I}$. Furthermore, $\mathcal{I}$ satisfies $C(a)$ or $R(a,b)$ if $a^\mathcal{I} \in C^\mathcal{I}$ or $(a^\mathcal{I}, b^\mathcal{I}) \in R^\mathcal{I}$. An interpretation $\mathcal{I}$ satisfies a TBox (ABox, ontology) if it satisfies every axiom therein. It is then called a *model* of this set of axioms.

The following decision problems are of interest for this section.

Concept satisfiability $\mathsf{CSAT}_Q(B)$:
Given a (B,Q)-concept C, is there an interpretation $\mathcal{I}$ s.t. $C^\mathcal{I} \neq \emptyset$?

TBox satisfiability $\mathsf{TSAT}_Q(B)$:
Given a (B,Q)-TBox $\mathcal{T}$, is there an interpretation $\mathcal{I}$ s.t. $\mathcal{I} \models \mathcal{T}$?

TBox-concept satisfiability $\mathsf{TCSAT}_Q(B)$:
Given a (B,Q)-TBox $\mathcal{T}$ and a (B,Q)-concept C, is there an $\mathcal{I}$ s.t. $\mathcal{I} \models \mathcal{T}$ and $C^\mathcal{I} \neq \emptyset$?

Ontology satisfiability $\mathsf{OSAT}_Q(B)$:
Given a (B,Q)-ontology $\mathcal{O}$, is there an interpretation $\mathcal{I}$ s.t. $\mathcal{I} \models \mathcal{O}$?

Ontology-concept satisfiability $\mathsf{OCSAT}_Q(B)$:
Given a (B,Q)-ontology $\mathcal{O}$ and a (B,Q)-concept C, is there an $\mathcal{I}$ s.t. $\mathcal{I} \models \mathcal{O}$ and $C^\mathcal{I} \neq \emptyset$?

We are interested in the complexity of these problems. The first, concept satisfiability without axioms, is already covered by Bauland et al.'s study of the satisfiability problem for modal logic (Theorem 3) because $\mathcal{ALC}$ is a notational variant of the modal logic K with the quantifiers $\exists, \forall$ corresponding to the modal operators $\Diamond, \Box$.

$\mathsf{TSAT}_Q(B)$	I_0	I	V_0	V	E_0	E	N_2, N	S_{11} to R_0	M	L_0	L_3 to BF	else
$Q = \emptyset$	t	NL	t	P	t	P	NL	t	NP	t	NP	t
$Q = \{\exists\}$	t	P	t	EXP	t	P	EXP	t	EXP	t	EXP	t
$Q = \{\forall\}$	t	P	t	P	t	EXP	EXP	t	EXP	t	EXP	t
$Q = \{\exists, \forall\}$	t	EXP	t	EXP	t	EXP	EXP	t	EXP	t	EXP	t

remaining												
$Q = \emptyset$	NL			P			NL		NP			t
$Q = \{\exists\}$	P		EXPTIME		P			EXPTIME				t
$Q = \{\forall\}$	P					EXPTIME						t
$Q = \{\exists, \forall\}$	EXPTIME											t

Table 3: Results for TSAT and the *remaining* three problems TCSAT, OSAT, OCSAT. All entries denote completeness. "EXP" abbreviates "EXPTIME", and "t" stands for "trivial".

It can easily be seen that there are some reducibilities between the satisfiability problems independently of B and Q:

$$\mathsf{CSAT}_Q(B) \leq_m^{\log} \mathsf{TCSAT}_Q(B) \text{ and}$$
$$\mathsf{TSAT}_Q(B) \leq_m^{\log} \mathsf{TCSAT}_Q(B) \leq_m^{\log} \mathsf{OSAT}_Q(B) \equiv_m^{\log} \mathsf{OCSAT}_Q(B)$$

2.4.2 Results and Discussion

Our results are given in Table 3 and Figures 11–14 (Appendix C). They are complete with respect to arbitrary sets of Boolean operators and quantifiers. We can furthermore extract the tractability border by listing the maximal tractable sub-Boolean fragments of $\mathcal{ALC}$, i.e., the maximal combinations of B and Q for which $\mathsf{TSAT}_Q(B)$ (and the other three problems) are tractable:

1. $B = \mathsf{R}_1$ (1-reproducing functions) and Q arbitrary
2. $B = \mathsf{R}_0$ (1-reproducing functions) and Q arbitrary – only TSAT
3. $B = \mathsf{E}$ (conjunction and both constants) and $Q \subseteq \{\exists\}$
4. $B = \mathsf{V}$ (disjunction and both constants) and $Q \subseteq \{\forall\}$
5. $B = \mathsf{N}$ (negation and both constants) and $Q = \emptyset$

In other words, the maximal lightweight DLs obtained by restricting Boolean operators and quantifiers are given by this short list. We can now answer our original question by observing that this list contains no interesting candidate for a lightweight DL other than those that are already known:

- Item 3 is $\mathcal{EL}^{\bot}$, the extension of $\mathcal{EL}$ with the $\bot$-operator, which is subsumed by the tractable logic $\mathcal{EL}^{++}$ underlying the OWL EL profile.

- Item 4 is a logic allowing only the duals of the operators in $\mathcal{EL}^{\bot}$, which can safely be ruled out as a reasonable modeling language.
- Item 5 is a very simple logic in which one can only express subsumption and disjointness between concept names, i.e., this logic allows to model only taxonomies with disjointness information.
- Items 1 and 2 denote the maximal fragments for which satisfiability is even trivial. In this case, subsumption is the relevant decision problem. Subsumption was studied by Meier [Mei11], whose results surprisingly yield that subsumption for the cases $B = \mathsf{R}_1$ and $B = \mathsf{R}_0$ is intractable. The maximal fragments with tractable subsumption (and trivial satisfiability) are determined by $B = \mathsf{E}_1, Q = \{\exists\}$ and $B = \mathsf{V}_1, Q = \{\forall\}$, but these are already subsumed by items 3 and 4.

Our study therefore provides a systematic underpinning of the folklore assumption that the restrictions to the Boolean operators and quantifiers that underlie the known families of light-weight DLs lead to the only useful sub-Boolean $\mathcal{ALC}$-fragments for which satisfiability in the presence of general TBoxes is tractable. On the one hand, this conclusion is more general than the results in [BBL05] that extensions of $\mathcal{EL}$ with $\neg$ and $\sqcup$ are intractable. On the other hand, it is restricted to the case of general TBoxes and can therefore not be transferred to, e.g., acyclic TBoxes. Another limitation of our study is that is does not allow any conclusion about unqualified use of quantifiers, which contributes to the good computational properties of logics in the DL-Lite family. A study that takes these two aspects into consideration would be a natural continuation; however, we consider it unlikely that it would yield significant further insights.

If we compare the results of this study with the previously discussed analyses of propositional and modal logic [Lew79, BHSS06, HSS10] or with our studies for temporal and hybrid logic in Sections 2.3 and 2.5, we can observe intractable fragments considerably closer to the bottom of the lattice – even down to the I-clones in the case $Q = \{\exists, \forall\}$. This difference is not too surprising, given that TBoxes reintroduce a limited form of implication and conjunction, which induce sufficient expressive power for encoding ExpTime-hard problems.

A natural question for future work is whether the complexity landscape, including the tractability border, changes if the use of general concept inclusions is restricted, for example, to acyclic terminologies, i.e., TBoxes where axioms are cycle-free definitions $A \equiv C$ with A being atomic. Theories so restricted are useful for establishing taxonomies, and concept satisfiability for $\mathcal{ALC}$ w.r.t. acyclic terminologies is still PSpace-complete [BH91, Cal96]. Furthermore, a large part of SNOMED CT is an acyclic $\mathcal{EL}$ terminology. Another natural direction for future work is mentioned above: the investigation of fragments with unqualified quantifiers.

2.5 Hybrid Logic

Hybrid logics extend modal logic with the ability to refer explicitly to states in Kripke structures, using nominals, the satisfaction operator @, and/or the downarrow binder $\downarrow$. This additional expressive power is sometimes paid with unsatisfactory computational properties: while the basic uni-modal language K extended with nominals and @ remains PSpace-complete [ABM99], the addition of only $\downarrow$ to K leads to undecidability [ABM99].

In order to regain decidability, several syntactic and semantic restrictions of the hybrid binder language have been considered. Ten Cate and Franceschet in [tF05] have reestablished decidability by restricting the interactions between $\downarrow$ and universal modal operators – such as

□, its inverse, and its global counterpart – and, separately, by restricting attention to Kripke structures of bounded width: depending on the severity of the single restrictions and on the question whether they are combined, $\mathsf{K}^{\downarrow}$ is NP-, ExpTime-, NExpTime- or 2ExpTime-complete. Our previous work was concerned with different semantic restrictions for regaining decidability: $\mathsf{K}^{\downarrow}$ is NExpTime-complete over transitive and complete Kripke structures, and while $\mathsf{K}^{\downarrow,@}$ is undecidable over transitive Kripke structures, it is still NExpTime-complete over complete Kripke structures [MSSW10]. Over equivalence relations, even K extended with the more powerful "jumping" binder ∃ is decidable, namely N2ExpTime-complete [MS09]. Furthermore, over acyclic Kripke structures such as linear structures and transitive trees, ↓ on its own does not add any expressivity; extensions such as $\mathsf{K}^{\downarrow,@}$ have been shown to be decidable but nonelementary in [FdS03, MSSW10]. Elementary fragments have been obtained by bounding the number of state variables [SW07, Web09, BL10].

We aim for a more detailed view of the complexity landscape of hybrid binder languages that exhibits a more fine-grained boundary between decidable and undecidable, between elementary and nonelementary, and between tractable and intractable fragments of $\mathsf{K}^{\downarrow,@}$. We systematically study fragments obtained by restrictions to the Boolean operators as in the previous sections, combined with restrictions to the modal and hybrid operators, over several classes of Kripke structures. The unrestricted hybrid language from which we start contains ◇ and □, as well as nominals, @, and ↓. The classes of Kripke structures that we consider fall into two groups:

- Classes of structures that allow cycles: all structures, transitive structures, total structures (where every state has at least one successor), and structures with equivalence relations (*ER structures* for short)
- Classes of acyclic structures: general linear orders and the special case of the natural numbers with the "less than" relation

One benefit that we expect from this systematic study is an insight into possible extensions of specification languages (such as LTL) and knowledge representation languages (such as DLs) with hybrid operators. Given the negative results listed above, there is no hope to gain computationally well-behaved logics by just adding ↓ without any restrictions. By considering sub-Boolean fragments, we hope to lay the foundations for the future design of "lightweight" hybrid extensions of LTL or DLs that stand a chance of being useful and well-behaved.

2.5.1 Basic Notions

We define the standard terminology of hybrid logic as in [At07]. Let Prop be a countable set of *propositional variables*, Nom a countable set of *nominals*, SVar a countable set of *state variables*, and Atom = Prop ∪ Nom ∪ SVar. The formulas of *hybrid (modal) logic* HL are defined by

$$\varphi ::= a \mid \circ_f(\varphi, \ldots, \varphi) \mid \Diamond\varphi \mid \Box\varphi \mid \downarrow x.\varphi \mid @_t\,\varphi,$$

where $a \in \mathsf{Atom}$, $\circ_f$ is a Boolean operator corresponding to the Boolean function f, and where $x \in \mathsf{SVar}$ and $t \in \mathsf{Nom} \cup \mathsf{SVar}$.

HL formulas are interpreted on *(hybrid) Kripke structures* $K = (W, R, \eta)$, consisting of a set of *states* W, a *transition relation* $R \subseteq W \times W$, and a *labeling function* $\eta : \mathsf{Prop} \cup \mathsf{Nom} \to 2^W$ satisfying $|\eta(i)| = 1$ for all $i \in \mathsf{Nom}$. In order to evaluate ↓-formulas, we use *assignments* $g : \mathsf{SVar} \to W$ similar to assignments in first-order logic. Given an assignment g, a state variable x and a state w, the *x-variant g^x_w of g* is the assignment satisfying $g^x_w(x) = w$ and $g^x_w(x') = g(x')$

for all $x \neq x'$. For any $a \in \mathsf{ATOM}$, let $[\eta, g](a) = \{g(a)\}$ if $a \in \mathsf{SVAR}$, and $[\eta, g](a) = \eta(a)$ otherwise. The satisfaction relation for HL-formulas is defined in Figure 3.

$K, g, w \models a$	iff $w \in [\eta, g](a) \qquad a \in \mathsf{ATOM}$
$K, g, w \models \circ_f(\varphi_1, \ldots, \varphi_n)$	iff $f(t_1, \ldots, t_n) = 1$, where $t_i = 1$ if $K, g, w \models \varphi_i$, and $t_i = 0$ otherwise
$K, g, w \models \Diamond\varphi$	iff $K, g, w' \models \varphi$ for some $w' \in W$ with wRw'
$K, g, w \models \Box\varphi$	iff $K, g, w' \models \varphi$ for all $w' \in W$ with wRw'
$K, g, w \models @_t\varphi$	iff $K, g, w' \models \varphi$ for $w' \in [\eta, g](t)$
$K, g, w \models \downarrow x.\varphi$	iff $K, g^x_w, w \models \varphi$

Figure 3: Satisfaction relation for HL-formulas

A hybrid formula φ is said to be *satisfiable* if $K, g, w \models \varphi$ for some Kripke structure $K = (W, R, \eta)$, some state $w \in W$, and some assignment $g : \mathsf{SVAR} \to W$.

The operator $@_t$ shifts evaluation to the state named by $t \in \mathsf{NOM} \cup \mathsf{SVAR}$. The downarrow binder $\downarrow x.$ binds the state variable x to the current state. The symbols $@_x$, $\downarrow x.$ are called *hybrid operators* whereas the symbols $\Diamond$ and $\Box$ are called *modal operators*.

Fragments of HL are defined as follows. Let B be a finite set of Boolean functions and H a set of hybrid and modal operators. We define $\mathsf{HL}_H(B)$ to be the set of HL-formulas using only hybrid and modal operators from H and Boolean operators corresponding to functions from B.

A *frame* F is a pair (W, R), where W is a set of states and $R \subseteq W \times W$ a transition relation. We are interested in the following properties of a frame: (W, R) is called

transitive	if	$uRv \wedge vRw \to uRw$ for all $u, v, w \in W$
symmetric	if	$uRv \to vRu$ for all $u, v \in W$
reflexive	if	uRu for all $u \in W$
total	if	for all $u \in W$ there is some $v \in W$ with uRv
trichotomous	if	for all $u, v \in W$, either uRv or $u = v$ or vRu

We will study the following frame classes.

all	all frames
trans	all transitive frames
total	all total frames
ER	all transitive, symmetric, and reflexive frames
lin	all transitive, irreflexive, and trichotomous frames
Nat	the singleton frame class $\{(\mathbb{N}, <)\} \subseteq$ lin

We say that the Kripke structure $K = (W, R, \eta)$ *is based on the frame* (W, R).

We study the *satisfiability problem* for the fragments $\mathsf{HL}_H(B)$ over frame classes $\mathfrak{F}$, which is defined as follows.

$\mathsf{HL}_H(B)$-$\mathfrak{F}$-SAT. Given a $\mathsf{HL}_H(B)$-formula φ, is there a Kripke structure $K = (W, R, \eta)$ based on a frame from $\mathfrak{F}$, an assignment $g : \mathsf{SVAR} \to W$, and a state $w \in W$ such that $K, g, w \models \varphi$?

The following theorem summarizes the results for hybrid binder languages with unrestricted Boolean operators $\wedge, \vee, \neg$ that are known from the literature. Since $\Box\varphi \equiv \neg\Diamond\neg\varphi$, the $\Box$-operator is implicitly present in all fragments containing $\Diamond$ and negation.

Theorem 7. [ABM99, ABM00, FdS03, MS09, MSSW10]

1. $\mathsf{HL}_{\Diamond,\downarrow}(\wedge,\vee,\neg)$-all-SAT *and* $\mathsf{HL}_{\Diamond,\downarrow,@}(\wedge,\vee,\neg)$-all-SAT *are* coRE*-complete.*
2. $\mathsf{HL}_{\Diamond,\downarrow}(\wedge,\vee,\neg)$-trans-SAT *is* NExpTime*-complete.*
3. $\mathsf{HL}_{\Diamond,\downarrow,@}(\wedge,\vee,\neg)$-trans-SAT *is* coRE*-complete.*
4. $\mathsf{HL}_{\Diamond,\downarrow}(\wedge,\vee,\neg)$-ER-SAT *is* NExpTime*-complete.*
5. $\mathsf{HL}_{\Diamond,\downarrow,@}(\wedge,\vee,\neg)$-ER-SAT *is* NExpTime*-complete.*
6. $\mathsf{HL}_{\Diamond,\downarrow,@}(\wedge,\vee,\neg)$-lin-SAT *and* $\mathsf{HL}_{\Diamond,\downarrow,@}(\wedge,\vee,\neg)$-Nat-SAT *are decidable and nonelementary.*
7. $\mathsf{HL}_{\Diamond,\downarrow}(\wedge,\vee,\neg)$-$\mathfrak{F}$-SAT, $\mathsf{HL}_{\Diamond,@}(\wedge,\vee,\neg)$-$\mathfrak{F}$-SAT, *and* $\mathsf{HL}_{\Diamond}(\wedge,\vee,\neg)$-$\mathfrak{F}$-SAT *with* $\mathfrak{F}\in\{\mathit{lin},\mathit{Nat}\}$ *are* NP*-complete.*

These results provide (not necessarily tight) upper bounds for the complexity of the problems we will study. Lower bounds for the fragments with at least one modal operator follow from Theorem 3.

Our study is divided into two parts. The first part includes frame classes that allow cycles: all, transitive, total and ER frames. The second part includes the remaining two frame classes from the list above, which are acyclic. One reason for this separation is the simple observation that the expressive power of $\downarrow$ differs dramatically over the two groups of frame classes: over acyclic frame classes, the addition of $\downarrow$ typically makes the basic modal logic K undecidable or at least very complex because states where a states variables are bound can be revisited. This is not possible without cycles, and K with and without $\downarrow$ are in fact equally expressive over acyclic frame classes, where interactions between $\downarrow$ and @ are required to add expressive power (and increase the complexity).

Another reason for the separation is the fact that, over frame classes that allow cycles, we can sometimes focus on the singleton reflexive frame when proving upper bounds. This obviously fails for acyclic frame classes, and different approaches are required.

To provide a more fine-grained analysis of tractable fragments in this section, we will additionally refer to the complexity class NC^1 and use the notion of *constant-depth* reductions $\leq_{\mathsf{cd}}$ for proving lower bounds, which is an appropriate refinement of logspace and polytime reductions. These notions are defined, for example, in [Vol99]. It is known that $\mathsf{NC}^1 \subseteq \mathsf{L}$.

2.5.2 Results for Kripke Structures with Cycles

We study the problems $\mathsf{HL}_H(B)$-$\mathfrak{F}$-SAT for the frame classes $\mathfrak{F}\in\{\mathsf{all},\mathsf{trans},\mathsf{total},\mathsf{ER}\}$, sets H of hybrid and modal operators with $\{\Diamond,\downarrow\}\subseteq H\subseteq\{\Diamond,\Box,\downarrow,@\}$, and all sets B of Boolean operators corresponding to clones in Post's lattice. The HL-fragments in our study are more numerous and more diverse than the LTL- or $\mathcal{ALC}$-fragments from the previous sections since we combine four subsets of modal/hybrid operators of very different expressivity with four frame classes (and an infinite number of sets of Boolean operators). For this reason, the complexity landscape turns out to be quite heterogeneous. Our results disclose the tractability and decidability border up to certain combinations of modal/hybrid and Boolean operators that have withstood complexity analysis, and we are able to isolate particular sources of (un)decidability or (in)tractability:

- Over all frame classes, the only source of undecidability (if any) is the presence of either the Boolean operator $a\wedge\neg b$ (clone S_1) or all self-dual Boolean operators (clone D).

$\mathfrak{F}$	H | B	I_2 to R_1	I_0, I	N_2, N	V_0, V	E_0, E	L_3, L_0, L	S_{11} to M	D, S_1 to BF
ER	◇↓	trivial	$\in$ NC^1	$\in$ NC^1	$\in$ NC^1	$\in$ NC^1	?	NC^1-c	NEXP-c
	◇□↓								
	◇↓@			L-c					
	◇□↓@								
total	◇↓	trivial	$\in$ NC^1	$\in$ NC^1	$\in$ NC^1	$\in$ NC^1	?	NC^1-c	coRE-c
	◇□↓								
	◇↓@			L-c					
	◇□↓@								
trans	◇↓	trivial	$\in$ NC^1	$\in$ NC^1	$\in$ NC^1	$\in$ NC^1	?	NC^1-c	NEXP-c
	◇□↓		$\in$ L		?	?		?	
	◇↓@		$\in$ NC^1	L-c	$\in$ NC^1	$\in$ NC^1		NC^1-c	coRE-c
	◇□↓@		L-c		L-h	NL-h		PS-h	
all	◇↓	trivial	$\in$ NC^1	$\in$ NC^1	$\in$ NC^1	$\in$ NC^1	?	NC^1-c	coRE-c
	◇□↓		$\in$ L		?	coNP-h		PS-h	
	◇↓@		$\in$ NC^1	L-c	$\in$ NC^1	$\in$ NC^1		NC^1-c	
	◇□↓@		L-c		L-h	coNP-h		PS-h	

Legend			
PS	PSPACE	h	hardness
NEXP	NEXPTIME	c	completeness

Table 4: Results for HL$_H(B)$-$\mathfrak{F}$-SAT for frame classes $\mathfrak{F}$ that allow cycles

- Over ER frames, all fragments are decidable, and the tractability border is the same as the decidability border in the previous bullet point. In particular, there is a strong dichotomy between NEXPTIME-complete fragments and fragments in L (most of which are even in NC^1).
- The picture over total frames is almost the same as over ER frames, with "NEXPTIME-complete" replaced by "coRE-complete".
- Over the other two frame classes (all frames, transitive frames), we obtain tractable fragments by allowing only negation and/or constants as Boolean operators, or disallowing □. The combination of □ with conjunction is a particular source of intractability over all frames, which is already due to results for Hemaspaandra's Poor Man's Logic [Hem01]. Over transitive frames, we can only conclude that □ together with @ and *all* monotone operators lead to intractability (because this combination is strong enough to encode QBF validity). Over these two frame classes too, all fragments found tractable are in L or even in NC^1.

For the fragments that are in NC^1, our original work [MMS+10] even differentiates between NC^1-completeness and membership in smaller classes of circuit complexity, but we do not make this distinction here, in order to simplify the presentation.

As in the case of LTL-SAT, we have not been able to classify the cases of Boolean operators from L_3, L_0, L based on the binary *xor* operator, with the same rationale for these cases being particularly difficult.

Table 4 and Figures 15–18 (Appendix C) give an overview of our results by frame classes.

Our most interesting result is L-hardness for HL$_{\Diamond,\Box,\downarrow,@}(I_2)$-$\mathfrak{F}$-SAT, i.e., the fragments containing all four hybrid and modal operators but only the constant 0 as a Boolean operator, over any of the

$\mathfrak{F}$	$H \subset \{\Box, \downarrow, @\}$	$H = \{\Box, \downarrow, @\}$	$\{\Diamond\} \subseteq H \subset \{\Diamond, \Box, \downarrow, @\}$	$H = \{\Diamond, \Box, \downarrow, @\}$
lin	NC^1	NC^1	NP	nonelem.
Nat	NC^1	L	NP	PSPACE

Table 5: Results for $\mathsf{HL}_H(\mathsf{M})$-$\mathfrak{F}$-SAT for acyclic frame classes $\mathfrak{F}$. "Nonelem." stands for "decidable and nonelementary"; all other entries denote completeness.

four frame classes. While this result confirms the general observation that fragments allowing for interactions of $\Box$ and $\downarrow$ tend to be harder than those without $\Box$, it is surprising to observe that this interaction already manifests itself in the absence of almost all Boolean operators. Given this lack of expressivity, our proof of the lower bound is far from trivial and is based on an intriguing way to encode of the L-complete problem "order between vertices".

Our technically most involved result is L-membership for $\mathsf{HL}_{\Diamond,\Box,\downarrow,@}(\mathsf{N})$-$\mathfrak{F}$-SAT, i.e., the fragments from above enriched with negation (and hence the constant 1). It relies on two central observations: first, any formula in this fragment can be equivalently written as a sequence of hybrid and modal operators followed by at most one negation operator and one atom. Second, if the given formula φ ends in $\neg x$ with x being a state variable, it needs to be checked whether, intuitively put, the "substring" of φ up to $\neg x$ is semantically "forbidden" to reach the state to which x is bound via $\downarrow x$. For this purpose, certain compatibility conditions between this substring and the substring leading to $\downarrow x$ need to be checked, which involves comparing single positions and computing scopes of $\downarrow$ operators.

2.5.3 Results for Acyclic Kripke Structures

The main open question from the previous subsection is the one for tight upper bounds for monotone (aka positive) fragments including the $\Box$-operator. It is difficult to make a reliable conjecture about the open cases: on the one hand, monotone fragments of propositional logic are simpler than the full Boolean case [Lew79], and the same can be said about LTL satisfiability (Section 2.3.2). On the other hand, they are as hard as the full Boolean case for modal logic [BHSS06, HSS10], LTL model-checking (Section 2.3.3), and the four standard decision problems studied for $\mathcal{ALC}$ in Section 2.4.2. We do not know whether the monotone fragments of our hybrid languages belong to the first or the second group of logics.

Monotone fragments of propositional logic are interesting because many real-life problems from various application areas can be formulated, for example, as equivalence tests between monotone formulas [Hag08].

In this subsection, we classify the computational complexity of satisfiability for fragments of $\mathsf{HL}_{\Diamond,\Box,\downarrow,@}(\mathsf{M})$ over the frame classes lin (linear orders) and Nat (the natural numbers). We thus study 32 logics, determined by arbitrary combinations of the two frame classes and subsets of $\{\Diamond, \Box, \downarrow, @\}$. We find that $\mathsf{HL}_{\Diamond,\Box,\downarrow,@}(\mathsf{M})$, which uses all four hybrid/modal operators, is the only logic that retains the nonelementary complexity from its "full Boolean" counterpart over lin. Over Nat, it is PSPACE-complete. For all smaller sets of hybrid/modal operators that contain at least the $\Diamond$-operator, we show NP-completeness over both frame classes. Over lin we obtain NC^1-completeness for all remaining fragments; over Nat they have the same complexity except for $\{\Box, \downarrow, @\}$, which is L-complete. Table 5 and Figure 4 give an overview of these results.

The technique used to establish the nonelementary lower bound reveals the ability of the four hybrid/modal operators to distinguish between dense and discrete linear orders and deserves

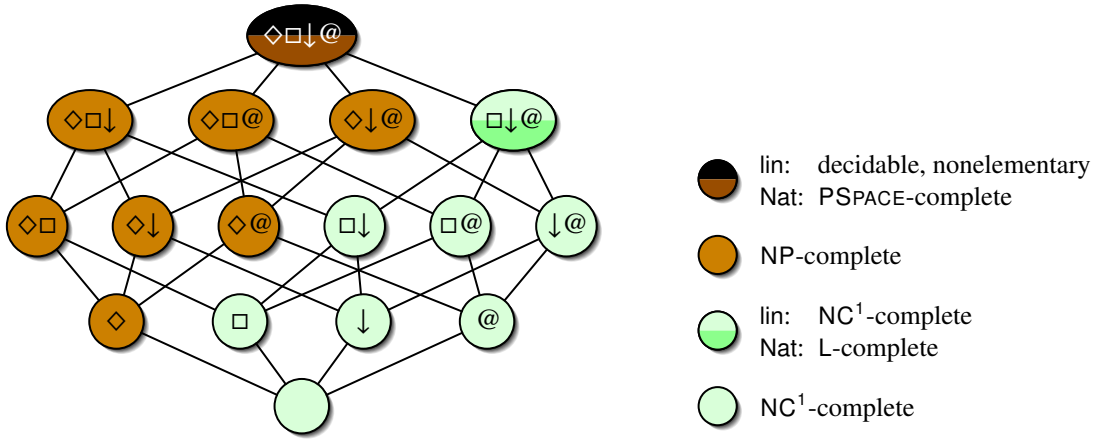


Figure 4: Results for $\mathsf{HL}_H(\mathsf{M})$-$\mathfrak{F}$-SAT for acyclic frame classes $\mathfrak{F} = \mathsf{lin}, \mathsf{Nat}$

a brief explanation. We use a reduction from satisfiability for first-order logic (FOL) with only one unary predicate P and one binary predicate $<$ which is interpreted as the natural "less than" relation over the natural numbers. This problem is nonelementary and decidable due to Stockmeyer's results [Sto74]. Although our monotone fragment does not have negation, we can encode P and $\neg P$ enforcing alternating sequences of dense and discrete intervals in a linear frame. A dense interval is a section of the linear order of states that is isomorphic to the closed interval $[0, 1]$ over the rational numbers; a discrete interval is a sequence of a finite number of states. The distinction of "dense" and "discrete" relies on enforcing or forbidding the existence of states between two given states, which requires all four hybrid/modal operators. This technique fails over Nat, whose underlying linear order is discrete. Over Nat we can embed the same hybrid logic into the above first-order language without the unary predicate, which is PSPACE-complete [FR79]. The corresponding lower bound is via a reduction from QBF validity.

Surprisingly, this difference in complexity between the two frame classes is reversed if we remove only $\Diamond$ from the allowed hybrid/modal operators: over lin, we can observe that all formulas of the form $\Box\alpha$ are easily satisfied in a state without successor, which is the essential insight needed to reduce $\mathsf{HL}_{\Box,\downarrow,@}(\mathsf{M})$-Nat-SAT to satisfiability of monotone *propositional* formulas, which is NC^1-complete [Sch07]. Over Nat, this idea fails because every state has a successor. However, we can find a decision procedure that runs in logarithmic space for $\mathsf{HL}_{\Box,\downarrow,@}(\mathsf{M})$ and in NC^1 for all smaller fragments.

In addition to the complexity results, we have shown that the 30 of the 32 logics that do not contain all four modal/hybrid operators have a *quasi-quadratic size model property*: over Nat, every satisfiable formula φ is satisfiable in a Kripke structure based on $(\mathbb{N}, <)$ that has a finite number of states where nominals and state variables are true, and where the distance between two consecutive such states is bounded by the modal depth of φ. Over lin, the finite interval between two such "nominal states" may be prefixed by at most one isomorphic copy of the dense open interval $(0, 1)$ over the rational numbers. Although models of this kind are generally infinite, they can be represented by a data structure of quadratic size describing the prefix up to the last nominal state. It is straightforward to extend known model-checking algorithms for hybrid logic [Fd06] to deal with these symbolic representations without affecting the complexity. This way we obtain a guess-and-check decision procedure for satisfiability, which entails not

only an alternative argument for NP-membership, but also a new NP-membership result for the same fragments over the frame class $\{(\mathbb{Q}, <)\}$.

The question remains whether the PSPACE-complete largest fragment over $(\mathbb{N}, <)$ admits some quasi-polynomial size model property. Furthermore, this study can be extended in several possible ways: by allowing negation on atomic propositions, by considering frame classes that consist only of dense frames, such as $(\mathbb{Q}, <)$, or by considering arbitrary sets of Boolean operators as in Section 2.5.2. For atomic negation, it follows quite easily that the largest fragment is of nonelementary complexity over $(\mathbb{N}, <)$, too, and that all fragments except $H = \{\Box, \downarrow, @\}$ are NP-complete. However, our proof of the quasi-quadratic size model property does not immediately go through in the presence of negated atomic propositions. Over $(\mathbb{Q}, <)$, we conjecture that all fragments, except possibly for the largest one, have the same complexity and model properties as over $(\mathbb{N}, <)$.

2.5.4 Lessons learned

Our study of hybrid languages exhibits a number of fragments of tractable or intermediate complexity, which raises hopes that restricting the interaction of $\downarrow$ with other operators may indeed lead to useful extensions of knowledge representation and specification languages. For example, our results in Table 4 make it seem likely that extensions of lightweight description logics such as $\mathcal{EL}$ with the $\downarrow$ operator have reasonable computational properties. However, this conjecture needs to be tested carefully because, in DLs, one is usually interested in satisfiability or subsumption with respect to a set of axioms, and it remains to investigate whether the interaction of those with $\downarrow$ will lead to a worse computational behavior.

As a second example, our results in Table 5 show that the monotone fragment of $\mathrm{LTL}_{\mathsf{F,G}}$ extended with both $\downarrow$ and @ is not harder than plain $\mathrm{LTL}_{\mathsf{F,G,X}}$. This fragment can be considered as a "lightweight" LTL fragment that imposes less restrictions than the successful lightweight description logic $\mathcal{EL}$ because it allows disjunctions and G. It is therefore conceivable that this fragment turns out to be a useful specification language. It still remains to be seen, however, whether this language is computationally well-behaved in terms of model checking too. Since model checking with hybrid binders is not as computationally expensive as satisfiability [Fd06], there is hope for a positive answer.

As an ambitious goal, it would be interesting to put this study on an even more systematic footing by systematizing all possible modal/hybrid operators and all possible frame classes. Similarly to the remarks at the end of Section 2.3, this is likely to be an overly ambitious research program. A quite substantial step into this more systematic direction has been taken by Hemaspaandra and Schnoor, who classified the satisfiability problem of the ("full Boolean") basic modal logic K over frame classes that can be defined by universal first-order Horn formulas into NP-complete and PSPACE-hard in [HS08].

Chapter 3

Module Extraction and Modularization

3.1 Introduction

Ontologies are logical theories that specify a vocabulary for a domain of interest and describe the relationships between the terms in that vocabulary. They have manifold applications in areas such as knowledge representation, knowledge management, semantic databases, the semantic web, biomedical informatics, the life sciences, linguistics, the geosciences. The needs of these applications are served by a variety of ontology languages. Description logics (DLs) [BCM+03], which we have already introduced in the previous chapters, comprise a widely used family of ontology languages and underlie the W3C Web Ontology Language OWL.

DLs are designed to provide a good trade-off between expressive power and computational complexity. Expressive DLs, such as $\mathcal{ALC}$ and $\mathcal{SROIQ}$, are designed to maximize expressive power while still retaining decidability of standard reasoning tasks such as satisfiability and subsumption. Their computational complexity is usually high: $\mathcal{ALC}$ is ExpTime-complete [Sch94, DM00] and $\mathcal{SROIQ}$ is N2ExpTime-complete [Kaz08]. Despite the high complexity, reasoning with these DLs can be feasible in practice thanks to highly optimized reasoning systems [HM01, TH06, BLS06, SPC+07, MSH09].

The success of DLs as ontology languages has been significantly determined by the availability of reasoning services within sophisticated ontology development systems such as Protégé,[1] Swoop [KPS+06], and the TopBraid composer. The traditional standard reasoning services for DLs are satisfiability and subsumption (studied in Section 2.4), asking whether all terms in the specified vocabulary are free of contradictions, or making the is-a hierarchy between the terms of the vocabulary explicit, which is useful for browsing and inspection. In recent years, however, the need for further reasoning services has arisen. Existing ontologies are more often considerably large: SNOMED CT [Spa00] has ≈400,000 logical axioms; the National Cancer Institute's (NCI) Thesaurus [GFH+03] has ≈110,000 logical axioms. These large ontologies pose serious challenges not only to the best optimized reasoners, but to all constituents of the ontology development process, such as navigation, editing, comprehension, and debugging. In particular, the following application scenarios are affected by the problems with handling large ontologies:

Scenario 1: Ontology Reuse. When engineers develop a new ontology, they often want to import knowledge that is already represented in existing ontologies. If those ontologies are large, it is not feasible to import them on the whole, for the reasons just stated. It is more desirable to import a subset of each external ontology that is self-contained with respect to the respective subdomain of interest, and there are several ways to define self-containment. Reasoning services are available to check whether a subset of an ontology is self-contained in this sense, and to extract such subsets.

[1] `http://protege.stanford.edu`

Scenario 2: Collaborative Ontology Development. When a team of developers maintains a large ontology, each expert wants to focus on a part of the ontology that corresponds to her subdomain of expertise (identified by a subset of its vocabulary) and make modifications under the assumption that they only touch this part. In particular, they want to make sure that their modifications have no impact on other, intuitively unrelated parts of the ontology. Since ontologies are complex logical theories, this is not automatically guaranteed and it is typically very difficult to check for such undesired consequences. Reasoning services are available to verify that unrelated parts of the ontology (again identified by parts of its vocabulary) have not changed, and signal a potential problem if this is not the case.

Scenario 3: Ontology comprehension. When an ontology developer wants to know whether the modeling in their large ontology corresponds to their understanding of the domain of interest, or when an ontology user wants to understand the topics of an ontology and their logical interrelations, then it is not useful to inspect a monolithic ontology axiom by axiom. The situation is similar to the task of understanding a large software project: instead of inspecting each single line of code, users and developers look at the dependencies between self-contained parts of the code, such as classes, packages, and modules. Analogously, the ontology user and developer can benefit from a partition of the ontology into logically self-contained parts that identify an ontology's main topics and represent the logical interactions within a topic as well as between topics.

To serve these application scenarios, additional reasoning services are needed: module extraction and modularization. **Module extraction** is concerned with determining a subset *(module)* of an ontology that "says the same" about a certain vocabulary of interest as the whole ontology. This task can be generalized to the question whether two arbitrary ontologies that are not necessarily in the subset relation "say the same" about a certain vocabulary, which is relevant, for example, for ontology versioning. Two ontologies with a positive answer to this question are called indistinguishable or *inseparable* (with respect to the corresponding vocabulary). In all these cases, there are several ways to define what "says the same" means, and each of them leads to a dedicated notion of a module or inseparability.

Modularization is concerned with decomposing an ontology into parts that represent its main topics – again represented by subsets of its vocabulary – and satisfy similarly strong properties regarding self-containment as modules. While it is straightforward to partition an ontology's vocabulary, it is not obvious how to obtain a corresponding partition of the axioms in the ontology into modules in the above sense. In particular, modules for disjoint subsets of the vocabulary typically overlap. Therefore, modularization should aim at identifying a partition whose constituents represent modules in a well-defined way.

From the above considerations, it becomes clear that modules play an important role as proxies of an ontology that can be used more conveniently because they are smaller and can thus be loaded, inspected, and reasoned over more efficiently. In this sense, this chapter on modularity contributes to the overall aim of the thesis: to identify ways to alleviate reasoning in expressive modal-like logics. In contrast to the previous chapter, where we systematically searched for restrictions *to the logic* that allow for efficient decision procedures, this chapter is devoted to studying ways of reducing the size *of a theory* in order to speed up the decision problems of the underlying logic, which are typically highly intractable, as in the case of OWL ($\mathcal{SROIQ}$). The strong logical guarantees required in Scenarios 1–3 and the respective module notions will be the central subject of study in this chapter.

The general topic of ontology modules and modularity is becoming increasingly important for the engineering of large-scale ontologies. In contrast to software development, where modularity is a well-understood and widely used paradigm, modularity of ontologies has only recently developed into an active field of research [CPSK06, KLWW08, CHKS08, JCS+08, Sun08, SPS09, CHKS10, ACH12, TP12], studying questions such as: How can a module of an ontology be defined? When is a module self-contained with respect to a certain vocabulary? How can this be verified algorithmically? How can a module for a given vocabulary be extracted from a large ontology? Can ontology modules help understand, develop, and efficiently reason over a large ontology written in an expressive DL?

The foundation for the strong logical guarantees required in the scenarios above is provided by conservative extensions (CEs) and the closely related notion of inseparability. The requirement that a subset $\mathcal{O}'$ of an ontology $\mathcal{O}$ be self-contained regarding a vocabulary (set of concept and role names) Σ can be formalized by the requirement that $\mathcal{O}$ be a *conservative extension (CE)* of $\mathcal{O}'$ with respect to the symbols in Σ. As mentioned above, it is sometimes more convenient to replace CEs with inseparability, which does not require that the compared ontologies are in the subset relation. Intuitively, ontologies $\mathcal{O}$ and $\mathcal{O}'$ are Σ-inseparable if the Σ-part of $\mathcal{O}$ and $\mathcal{O}'$ cannot be distinguished. In Scenario 2 it is important to ensure that the original ontology $\mathcal{O}$ is $\mathsf{sig}(\mathcal{O}) \setminus \Sigma$-inseparable from the new ontology $\mathcal{O}'$, where $\mathsf{sig}(\mathcal{O})$ is the overall vocabulary of $\mathcal{O}$ and Σ is the vocabulary corresponding to the respective domain expert's topic.

CEs have a long tradition in mathematical logic and are used in software specification to express that one specification refines another [TM87, BP90, Mai97] and to support modular specifications [DGS93, Mos04]. It was not until 2006 that the importance of CEs and inseparability for modularity was identified by Ghilardi et al. [GLW06]. In the years to follow, a whole host of research was dedicated to understanding and characterizing CEs of DL ontologies [KLWW09, LW10, KWZ10], to analyzing the computational properties of deciding CEs [LWW07, KLWW08, KPS+09, KWZ10, KKL+11], and to defining notions of modules based on CEs [KLWW08, CHKS08, KWZ10]. There are several variants of CEs, the most classical being the deductive and model-theoretic ones, see [KLWW09] and Section 3.2.2.

Unfortunately, CEs and inseparability have poor computational properties, which largely forbid their use in practical applications: the computational complexity of deciding deductive CEs is typically by an exponential higher than standard satisfiability, and it is undecidable for modest extensions of the basic DL $\mathcal{ALC}$ [GLW06, LWW07]. Model-theoretic CEs are undecidable already for the lightweight DL $\mathcal{EL}$ [LW10], with the notable exception that, when TBoxes are restricted to being acyclic, model-CEs are decidable in polynomial time for $\mathcal{EL}$ and Π_2^{p}-complete for $\mathcal{ALC}$ [KLWW08]. These predominantly negative results imply that, for DLs of medium or high expressivity, there is no hope to define a notion of a module based on dCEs if one expects to be able to extract *minimal* modules effectively and efficiently. For this reason, the notion of *locality* was introduced [CHKS08], which yields a sufficient condition for being a CE and for inseparability, and whose computational complexity is not higher than standard satisfiability (for the semantic version of locality) or even in polynomial time (for the syntactic version). Locality-based modules are thus a "safe approximation" of CE-based modules in the sense that they always contain the minimal CE-based module but are typically a bit larger.

Nowadays, locality-based modules (LBMs) play an important role in practical applications. They have been implemented, to a large part by the author of this thesis, in the OWL API[2] (a powerful Java interface for manipulating ontologies that underlies the ontology editor Protégé)

[2] `http://owlapi.sourceforge.net`

and in a web service for module extraction.[3] According to the traffic on various mailing lists related to OWL and Protégé, ontology engineers make use of LBMs. Furthermore, some principled approaches to incremental and modular reasoning are based on LBMs [CHKS10, ACH12, TP12], and some of the above questions have been answered. However, before we began the work reported in this chapter, locality-based modules had just been invented, and the following questions were yet to be answered.

1. Do the theoretical properties of LBMs justify their suitability in the scenarios above? For example, are they robust under imports – that is, is the property of being a module unaffected when importing a module of an ontology into an arbitrary other ontology?
2. If so, what is a systematic way to employ LBMs in the three scenarios above?
3. How do LBMs compare with other module notions in terms of size, extraction time, and usefulness? In particular, what is the typical difference in size between an LBM and the corresponding minimal CE-based module?
4. How does a user or engineer specify the vocabulary of interest? This is a nontrivial task because single topics (such as "diseases" or "anatomy" in a medical ontology) do not correspond to obvious sets of concept and role names.
5. How to determine the modular structure of an ontology, based on logically self-contained modules, and thus reveal the logical interactions within the ontology?
6. How to make use of this modular structure in order to present, understand, and (collaboratively) maintain an ontology, and to optimize reasoning?

In this chapter, we report on several theoretical and empirical studies aimed at making LBMs and other logic-based module notions usable for the reasoning services and applications discussed above. The chapter is divided into two parts: Section 3.2 studies LBMs, and Section 3.3 introduces a new modularization technique called Atomic Decomposition.

Our studies of LBMs deal with Questions 1, 2, and 3: we begin with devising a methodology that explicates the use of LBMs in Scenario 1, proving that LBMs have the required strong logical properties (Section 3.2.3). We then investigate and confirm robustness properties of LBMs relevant for all three scenarios (Section 3.2.4). We finish this part with a summary of empirical studies of LBMs, documenting the usefulness of LBMs and the applicability of our methodology to real-life import/reuse scenarios (Section 3.2.5).

Our work on Atomic Decomposition (AD) deals with Questions 5 and 6: we begin with providing empirical evidence that the typical number of modules of an ontology is indeed prohibitively large, dashing hopes to compute the modular structure from *all* modules (Section 3.3.2). We then introduce AD as a principled approach to representing all modules in a partition of the ontology's axioms, whose computation requires only a linear number of module extractions. This approach is universal in the sense that it is applicable to all module notions that provide certain logical guarantees (Section 3.3.3). We show how to use AD to speed up the extraction of a single module (Section 3.3.4), and provide empirical evidence that AD behaves well in practice, applying it to a large corpus of existing ontologies and reporting on the shape of the resulting decompositions (Section 3.3.5). Although we cannot answer Question 4 fully yet, we will argue that AD bears the potential to help identify of a vocabulary of interest.

[3] `http://mowl-power.cs.man.ac.uk:8080/modularity`

It remains to observe that our work focuses on *a-posteriori* modularity and modularization: applying module extraction and modularization techniques to existing ontologies. In contrast, *a-priori* approaches aim at imposing a modular structure on an ontology to foster its modular development. These approaches are typically based on heuristics without logical guarantees (such as the distribution over thematic files), or on extensions of DLs designed to specify ontology modules and syntactic links between them [BVSH09, ST09, CPS09]. However, AD seems to have the potential to become an a-priori modularization approach as well: the modular structure represented in the AD of an ontology at some point of its development process may turn out to be useful for its further modular development.

Bibliographic notes. The methodology for using LBMs discussed in Section 3.2.3 appeared in [JCS+08]. Our investigation of robustness properties (Section 3.2.4) is from [SSZ09]. The empirical studies reported in Section 3.2.5 are published in [JCS+08, SSZ09, DKP+13]. The foundations of atomic decomposition (Section 3.3.3) are introduced in [DPSS11]; the preparatory empirical study of module numbers (Section 3.3.2) is from [PS10]. Fast module extraction via AD (Section 3.3.4) and the study of AD and fast module extraction in practice (Section 3.3.5) are described in [DGK+11].

3.2 Logic-Based Module Extraction

3.2.1 A Reuse Scenario

We sketch a particular use case for Scenario 1, ontology reuse, in a real-world application – namely the development of an ontology, called JRAO, to describe a kind of arthritis called JRA (Juvenile Rheumatoid Arthritis) within the Health-e-Child project.[4] This project aimed at creating a repository of ontologies that can be used by clinicians in various applications.

The specific kinds of JRA to be modeled in JRAO are distinguished by several factors such as the joints affected, incurred responses (e.g., fever), and the treatment required. Well-established biomedical ontologies such as NCI (mentioned in Chapter 1) and GALEN[5] serve as reference ontologies: they contain information relevant to JRA, such as detailed descriptions of the human joints, diseases, symptoms. Figure 5 shows a fragment of NCI that defines JRA, together with our reuse scenario, where $C_1, \ldots, C_7$ refer to the kinds of JRA to be defined in JRAO.

The JRAO developers want to reuse knowledge from NCI and GALEN for three reasons: (a) they want to save time through reusing existing ontologies rather than writing their own; (b) they value knowledge that is commonly accepted by the community and used in similar applications; (c) they are not experts in all areas covered by NCI and GALEN. In doing so, they need to ensure that JRAO correctly reflects the reused knowledge but does not grow too much. An appropriate reuse methodology should therefore provide two guarantees.

(1) The use of certain concepts from NCI and GALEN in JRAO should not change their original meaning.

(2) The JRAO developers want to import only those axioms from NCI and GALEN that are relevant for JRAO. By importing only fragments of NCI and GALEN, they should not lose important information.

[4] http://www.health-e-child.org
[5] http://www.co-ode.org/galen

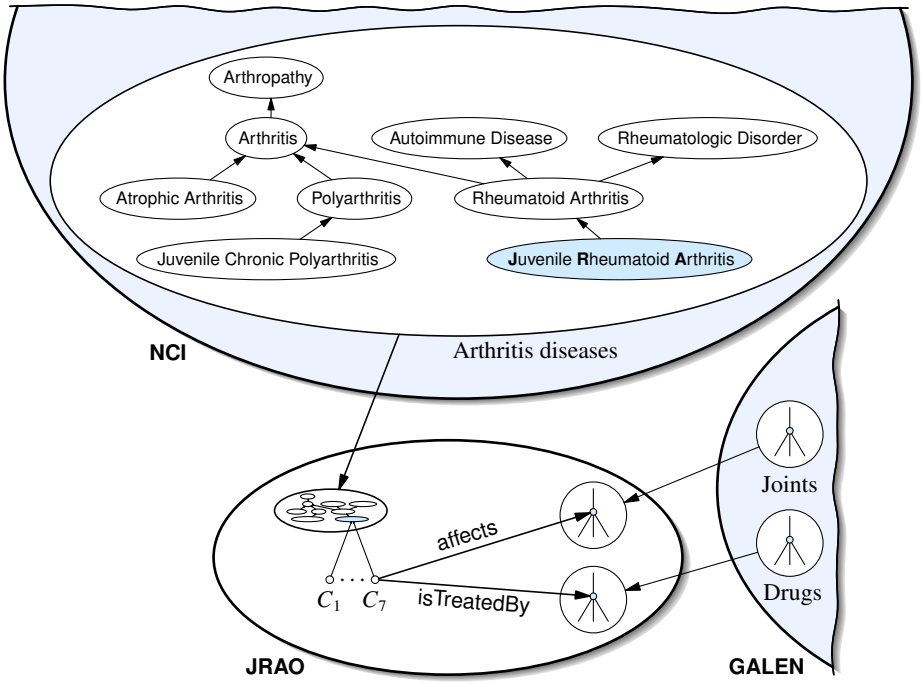


Figure 5: Constructing the ontology JRAO reusing fragments of GALEN and NCI

Our scenario has two main points in common with other ontology design scenarios: the ontology developer wants to reuse knowledge without changing it, and import only the relevant parts of an existing ontology. To support these scenarios whilst providing the two above guarantees, a logic-based approach to reuse is required. Current tools that support reuse, however, do not implement a logic-based solution and thus do not provide the above guarantees – and neither do existing guidelines and "best practices" for ontology design.

3.2.2 Basic Notions

We assume familiarity with DLs up to $\mathcal{SROIQ}$ [BCM+03, HKS06a], and we will use $\mathcal{L}$ to denote any such DL. We set $\mathcal{SROIQ}$ as an "upper bound" because it underlies OWL and because the original definition of locality-based modules for $\mathcal{SHIQ}$ in [CHKS08] can easily be extended to logics up to $\mathcal{SROIQ}$. We restrict attention to the terminological parts of ontologies, i.e., to TBoxes. The main reason is that we are concerned with module notions that preserve terminological knowledge. Nevertheless, these module notions are applicable to ontologies with ABoxes too.

A *signature* Σ is a set of terms (concept and role names). It specifies a topic of interest: axioms consisting solely of terms from Σ can be thought of as "on-topic", and all other axioms as "off-topic". Any concept or role name, TBox, axiom, etc. that uses only terms from Σ is called a Σ-*concept*, Σ-*role*, etc. Given any such object X, we call the set of terms in X the *signature of* X and denote it with $\mathsf{Sig}(X)$.

Given an interpretation $\mathcal{I}$, we denote its restriction to the terms in a signature Σ with $\mathcal{I}|_\Sigma$. Two interpretations $\mathcal{I}$ and $\mathcal{J}$ are said to *coincide on a signature* Σ, in symbols $\mathcal{I}|_\Sigma = \mathcal{J}|_\Sigma$, if $\Delta^\mathcal{I} = \Delta^\mathcal{J}$ and $X^\mathcal{I} = X^\mathcal{J}$ for all $X \in \Sigma$.

Conservative extensions, safety, and modules. Guarantee (1) above requires a preservation of knowledge ("should not change their meaning"), which is best captured by the notion of a *conservative extension (CE)* [GLW06, LWW07]. Depending on the degree of knowledge preservation required, CEs come in several variants. We focus on the following basic ones.

Definition 8. Let $\mathcal{L}$ be a DL, $\mathcal{T}_1 \subseteq \mathcal{T}$ be TBoxes, and Σ a signature.

1. $\mathcal{T}$ is a *deductive* Σ*-conservative extension (*Σ*-dCE)* of $\mathcal{T}_1$ if, for every axiom α over $\mathcal{L}$ with $\mathsf{Sig}(\alpha) \subseteq \Sigma$, we have $\mathcal{T} \models \alpha$ iff $\mathcal{T}_1 \models \alpha$.
2. $\mathcal{T}$ is a *deductive conservative extension (dCE)* of $\mathcal{T}_1$ if $\mathcal{T}$ is a $\mathsf{Sig}(\mathcal{T}_1)$-dCE of $\mathcal{T}_1$.
3. $\mathcal{T}$ is a *model-theoretic* Σ*-cons. extension (*Σ*-mCE)* of $\mathcal{T}_1$ if $\{\mathcal{I}|_\Sigma \mid \mathcal{I} \models \mathcal{T}_1\} = \{\mathcal{I}|_\Sigma \mid \mathcal{I} \models \mathcal{T}\}$.
4. $\mathcal{T}$ is a *model-theoretic conservative extension (dCE)* of $\mathcal{T}_1$ if $\mathcal{T}$ is a $\mathsf{Sig}(\mathcal{T}_1)$-mCE of $\mathcal{T}_1$.

It is clear that $\mathcal{T}$ being a (Σ-)mCE of $\mathcal{M}$ implies that $\mathcal{T}$ is a (Σ-)dCE of $\mathcal{M}$.

Definition 8 (1) applies to our example as follows: $\mathcal{T}_1$ is the ontology to be reused (NCI), $\mathcal{T}$ is the union of JRAO and NCI, and Σ represents the terms reused from NCI. Item (2) captures the case where we want to reuse *all* terms from NCI. Items (3) and (4) provide sufficient conditions.

Since the ontology to be reused (NCI) is usually under development beyond the control of the JRAO developers, it is convenient to make the axioms in NCI available on demand via a reference such that the developers of JRAO need not commit to a particular version of NCI. The notion of *safety* [CHKS08] is a generalization of dCE that abstracts away from the ontology to be reused and focuses on the reused *terms*.

Definition 9. Let $\mathcal{T}$ be a TBox and Σ a signature. $\mathcal{T}$ *is safe for* Σ if, for every TBox $\mathcal{T}'$ with $\mathsf{Sig}(\mathcal{T}) \cap \mathsf{Sig}(\mathcal{T}') \subseteq \Sigma$, we have that $\mathcal{T} \cup \mathcal{T}'$ is a Σ-dCE of $\mathcal{T}'$.

Guarantee (2) requires a preservation of knowledge ("should not lose important information") that can be captured by the notion of a *module* [CHKS08].

Definition 10. Let $\mathcal{T}_1' \subseteq \mathcal{T}'$ be TBoxes and Σ a signature. $\mathcal{T}_1'$ *is a* Σ*-module in* $\mathcal{T}'$ if, for every TBox $\mathcal{T}$ with $\mathsf{Sig}(\mathcal{T}) \cap \mathsf{Sig}(\mathcal{T}') \subseteq \Sigma$, we have that $\mathcal{T} \cup \mathcal{T}'$ is a $\mathsf{Sig}(\mathcal{T})$-dCE of $\mathcal{T} \cup \mathcal{T}_1'$.

The notions of safety and module are related as follows:

Proposition 11. [CHKS08] *If* $\mathcal{T}' \setminus \mathcal{T}_1'$ *is safe for* $\Sigma \cup \mathsf{Sig}(\mathcal{T}_1')$*, then* $\mathcal{T}_1'$ *is a* Σ*-module in* $\mathcal{T}'$.

Locality. The decision problems associated with dCEs, safety and modules are undecidable already for the fragment $\mathcal{ALCQIO}$ of $\mathcal{SROIQ}$ [LWW07], and mCEs are undecidable already for the lightweight DL $\mathcal{EL}$ [LW10]. *Locality* has been proposed as a decidable sufficient condition for safety: if all axioms in $\mathcal{T}$ satisfy the locality conditions, then $\mathcal{T}$ is safe, but the converse does not necessarily hold [CHKS08]. By means of Proposition 11, locality can be used for extracting modules too. The main reasons for the success of locality are the facts that (a) locality is a property of a single axioms and does not depend on the other axioms in the ontology, and (b) the syntactic variant can be checked in polynomial time.

The main intuition underlying locality is to check, given a signature Σ, whether the axiom in question can be satisfied independently of the interpretation of the Σ-terms, but in a restricted way: by interpreting all non-Σ terms either as the empty set ($\emptyset$-locality) or as the full domain[6]

[6] Or, in the case of roles, the set of all pairs of domain individuals.

(Δ-locality). This intuition leads to two dual variants of locality, which reflect distinct ways of reusing terms, namely generalization and refinement. The following definition introduces these two variants of *semantic locality* [CHKS08].

Definition 12. An axiom α over a logic $\mathcal{L}$ is called $\emptyset$*-local (Δ-local) w.r.t. signature* Σ if, for each interpretation $\mathcal{I}$, there exists an interpretation $\mathcal{J}$ such that $\mathcal{I}|_\Sigma = \mathcal{J}|_\Sigma$, $\mathcal{J} \models \alpha$, and for each $X \in \mathsf{Sig}(\alpha) \setminus \Sigma$, $X^\mathcal{J} = \emptyset$ (for each $C \in \mathsf{Sig}(\alpha) \setminus \Sigma$, $C^\mathcal{J} = \Delta$ and for each $R \in \mathsf{Sig}(\alpha) \setminus \Sigma$, $R^\mathcal{J} = \Delta \times \Delta$).

It has been shown in [CHKS08] that, if $\mathcal{T}_1 \subseteq \mathcal{T}$ and all axioms in $\mathcal{T} \setminus \mathcal{T}_1$ are $\emptyset$-local (or all axioms are Δ-local) w.r.t. $\Sigma \cup \mathsf{Sig}(\mathcal{T}_1)$, then $\mathcal{T}$ is a Σ-mCE of $\mathcal{T}_1$. The converse does not hold.

Both $\emptyset$- and Δ-locality can be tested using available DL reasoners: just check whether the axiom α' obtained from α by replacing all non-Σ terms with $\bot$ (or $\top$) is a tautology. In some cases, α' is not a $\mathcal{SROIQ}$ axiom, but standard reasoners can be extended in a straightforward way. Deciding semantic locality is thus considerably easier than deciding dCE. However, reasoning in expressive DLs is still complex, e.g. N2ExpTime-complete for $\mathcal{SROIQ}$.

In order to achieve *tractable* module extraction, a syntactic approximation of semantic locality was introduced in [CHKS08]. (Syntactic) $\bot$-locality is based on two grammars defining concept descriptions that are equivalent to $\bot$ or $\top$ if all non-Σ symbols are interpreted as $\bot$. An axiom $C \sqsubseteq D$ is then $\bot$-local if C is $\bot$- or D is $\top$-equivalent. The definition of $\top$-locality is analogous. It is straightforward to extend both definitions to cover all features of $\mathcal{SROIQ}$, as we have done in the implementation of locality in the OWL API. Obviously, $\bot$-locality ($\top$-locality) is sufficient for $\emptyset$-locality (Δ-locality). Therefore, all axioms in $\mathcal{T} \setminus \mathcal{T}_1$ being $\bot$-local (or all axioms being $\top$-local) w.r.t. $\Sigma \cup \mathsf{Sig}(\mathcal{T}_1)$ is sufficient for $\mathcal{T}$ to be a Σ-mCE of $\mathcal{T}_1$. The converse does not hold. Syntactic locality can be tested in polynomial time [CHKS08].

From now on, we use x to refer to any of the four locality variants. We say that a TBox is x-local if all of its axioms are x-local. All four variants of locality are sufficient for safety:

Proposition 13. [CHKS08] *If a TBox $\mathcal{T}$ is x-local w.r.t. Σ, for any $x \in \{\emptyset, \Delta, \bot, \top\}$, then $\mathcal{T}$ is safe for Σ.*

Propositions 11 and 13 suggest the following definition of modules in terms of locality.

Definition 14. Let $\mathcal{T}_1 \subseteq \mathcal{T}$ be TBoxes and Σ a signature. $\mathcal{T}_1$ is an *x-module for Σ in $\mathcal{T}$*, for any $x \in \{\emptyset, \Delta, \bot, \top\}$, if $\mathcal{T} \setminus \mathcal{T}_1$ is x-local w.r.t. $\Sigma \cup \mathsf{Sig}(\mathcal{T}_1)$.

It is known that x-modules are modules as in Definition 10:

Proposition 15. [CHKS08] *Let $\mathcal{T}_1$ be an x-module for Σ in $\mathcal{T}$, for any $x \in \{\emptyset, \Delta, \bot, \top\}$, and let $\Sigma' = \Sigma \cup \mathsf{Sig}(\mathcal{T}_1)$. Then $\mathcal{T}_1$ is a Σ'-module in $\mathcal{T}$.*

In order to be able to *extract* a module based on any of the four locality notions $x \in \{\emptyset, \Delta, \bot, \top\}$ for some $\mathcal{T}$ and Σ, it is important to observe that there is a *unique minimal* x-module for Σ in $\mathcal{T}$ [CHKS08]. These modules are obtained by starting with an empty set of axioms and subsequently adding axioms from $\mathcal{O}$ that are non-local, extending the signature against which locality is checked with the terms of the added axioms. To avoid confusion, the initial signature Σ is called the *seed signature* for the module. The procedure is given in Algorithm 1, and the unique minimal locality-based module (from now on called LBM) is introduced in Definition 16. LBMs can be made made smaller by nesting $\top$-extraction (Δ-extraction) into $\bot$-extraction

Algorithm 1: Extraction of a locality-based module

input : TBox $\mathcal{T}$, seed signature Σ, $x \in \{\emptyset, \Delta, \bot, \top\}$
output : x-module $\mathcal{M}$ of $\mathcal{O}$ w.r.t. Σ

$M \leftarrow \emptyset$ $\quad$ $\mathcal{T}' \leftarrow \mathcal{T}$
repeat
  changed $\leftarrow$ `false`
  foreach $\alpha \in \mathcal{T}'$ **do**
    if α *not x-local w.r.t.* $\Sigma \cup \mathsf{Sig}(\mathcal{M})$ **then**
      $\mathcal{M} \leftarrow \mathcal{M} \cup \{\alpha\}$ $\quad$ $\mathcal{T}' \leftarrow \mathcal{T}' \setminus \{\alpha\}$ $\quad$ changed $\leftarrow$ `true`
until *changed* = `false`
return $\mathcal{M}$

($\emptyset$-extraction) and vice versa, see Definition 16 (2); the resulting $\top\bot$-, $\bot\top$-modules etc. are still mCE-based modules in the sense of Definition 10. Finally, iterating this nesting can lead to even smaller modules, called $\top\bot^*$-, $\bot\top^*$-modules etc.; see Definition 16 (3). While locality and locality-based modules have been introduced in [CHKS08], the nested and iterated variants are our contribution [JCS+08, SSZ09].

Definition 16. Let $x \in \{\emptyset, \Delta, \bot, \top\}$, and $\mathcal{T}$ be a TBox and Σ a signature.

1. *The x-module of $\mathcal{T}$ w.r.t.* Σ, written $x\text{-mod}(\Sigma, \mathcal{T})$, is the output of Algorithm 1.
2. *The $\top\bot$-module of $\mathcal{T}$ w.r.t.* Σ is defined by $\top\bot\text{-mod}(\Sigma, \mathcal{T}) = \top\text{-mod}(\Sigma, \bot\text{-mod}(\Sigma, \mathcal{T}))$. The *$\bot\top$-, $\Delta\emptyset$-, and $\emptyset\Delta$-module of $\mathcal{T}$ w.r.t.* Σ are defined analogously.
3. Let $(\mathcal{M}_i)_{i \geqslant 0}$ be a sequence of TBoxes with $\mathcal{M}_0 = \mathcal{T}$ and $\mathcal{M}_{i+1} = \top\bot\text{-mod}(\Sigma, \mathcal{M}_i)$ for all $i \geqslant 0$. For the smallest n with $\mathcal{M}_n = \mathcal{M}_{n+1}$, we call $\mathcal{M}_n$ *the $\top\bot^*$-module of $\mathcal{T}$ w.r.t.* Σ, written $\top\bot^*\text{-mod}(\Sigma, \mathcal{T})$. Analogously for the *$\bot\top^*$-, $\Delta\emptyset^*$-, and $\emptyset\Delta^*$-module of $\mathcal{T}$ w.r.t.* Σ.

It is easy to see that $\top\bot^*\text{-mod}(\Sigma, \mathcal{T}) = \bot\top^*\text{-mod}(\Sigma, \mathcal{T})$ and $\Delta\emptyset^*\text{-mod}(\Sigma, \mathcal{T}) = \emptyset\Delta^*\text{-mod}(\Sigma, \mathcal{T})$.

Modulo the locality check, Algorithm 1 runs in time cubic in $|\mathcal{T}| + |\Sigma|$ [CHKS08]. Syntactic LBMs are therefore a feasible approximation for semantic LBMs.

3.2.3 A Logic-Based Methodology for Safe and Economic Reuse of Ontologies

In this section, we propose a methodology, based on LBMs, for designing an ontology in a reuse scenario where knowledge is to be borrowed from several external ontologies. This methodology provides precise guidelines for ontology developers to follow, and it ensures that a set of logical guarantees will hold at certain stages of the design process. Our original work [JCS+08] also describes a tool that implements this methodology; Section 3.2.5 reports on experiments.

We propose the working cycle given in Figure 6. This cycle consists of an *offline phase* – which is performed independently of the current contents of the external ontologies – and an *online phase* – where knowledge from the external ontologies is extracted and transferred into the current ontology. The separation between offline and online is not strict: The first phase can be performed offline or online, at the user's discretion. Our original work [JCS+08] contains a detailed description of both phases. The three marks on the right-hand side of Figure 6 indicate that the corresponding stages of the methodology require three guarantees:

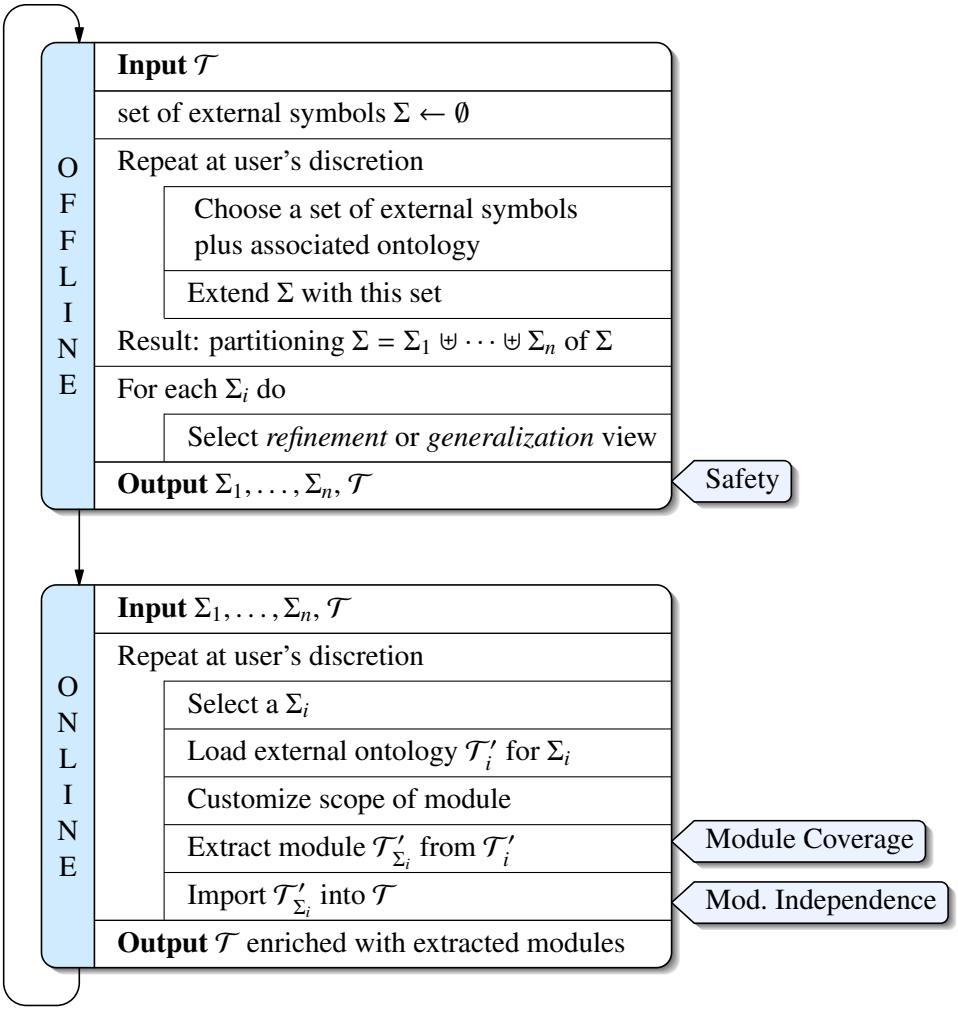


Figure 6: The two phases of import with the required guarantees

- *Safety:* The designer of $\mathcal{T}$ does not change the original meaning of the reused concepts, independently of what their particular meaning is in the external ontologies.
- *Coverage:* The extracted fragment $\mathcal{T}'_{\Sigma_i}$ is a module of the external TBox $\mathcal{T}'$ for the customized signature Σ_i according to Definition 10.
- *Module Independence:* The TBox $\mathcal{T} \cup \mathcal{T}_{\Sigma_i}$, which evolves from $\mathcal{T}$ after importing $\mathcal{T}_{\Sigma_i}$, should continue not to violate safety for the remaining imports.

The precise formulation of the three guarantees is as follows.

Definition 17.

1. The TBox $\mathcal{T}$ *guarantees safety w.r.t. the signatures* $\Sigma_1, \ldots, \Sigma_n$ if $\mathcal{T}$ is safe for Σ_i for all $i \leq n$.
2. Let Σ be a signature and $\mathcal{T}'_\Sigma \subseteq \mathcal{T}'$ TBoxes. $\mathcal{T}'_\Sigma$ *guarantees coverage of* Σ *in* $\mathcal{T}'$ if $\mathcal{T}'_\Sigma$ is a module for Σ in $\mathcal{T}'$.

3. Let $\mathcal{T}$ be a TBox and Σ_1, Σ_2 be signatures. $\mathcal{T}$ *guarantees module independence* if, for all $\mathcal{T}_1'$ with $\mathsf{Sig}(\mathcal{T}) \cap \mathsf{Sig}(\mathcal{T}_1') \subseteq \Sigma_1$ and for all $\mathcal{T}_2'$ with $\mathsf{Sig}(\mathcal{T}) \cap \mathsf{Sig}(\mathcal{T}_2') \subseteq \Sigma_2$ and $\mathsf{Sig}(\mathcal{T}_1') \cap \mathsf{Sig}(\mathcal{T}_2') = \emptyset$, it holds that $\mathcal{T} \cup \mathcal{T}_1' \cup \mathcal{T}_2'$ is a conservative extension of $\mathcal{T} \cup \mathcal{T}_1'$.

Both safety and coverage can be achieved using $\bot$- and $\top$-locality:

Proposition 18.

1. *Let $\mathcal{T}$ be a TBox and $\Sigma = \Sigma_1 \uplus \ldots \uplus \Sigma_n$ be the union of disjoint signatures. If, for each Σ_i, $\mathcal{T}$ is either $\bot$-local or $\top$-local w.r.t. Σ_i, then $\mathcal{T}$ guarantees safety w.r.t. $\Sigma_1, \ldots, \Sigma_n$.*
2. *Any of the modules $\bot$-mod$(\Sigma, \mathcal{T}'), \ldots, \top\bot$-mod$(\Sigma, \mathcal{T}'), \ldots, \top\bot^*$-mod$(\Sigma, \mathcal{T}'), \ldots$ guarantees coverage of Σ in $\mathcal{T}'$.*

Module independence can be guaranteed only in a restricted way; see the technical report[7] for details and a discussion of the restriction.

Proposition 19. *Let $\mathcal{T}$ be a TBox and Σ_1, Σ_2 disjoint signatures. If the following conditions are satisfied, where x denotes either $\bot$ or $\top$, then $\mathcal{T} \cup \mathcal{T}_1'$ is x-local, and thus safe, w.r.t. Σ_2.*

(a) $\mathcal{T}$ is x-local with respect to Σ_2.

(b) $\mathsf{Sig}(\mathcal{T}_1') \cap \Sigma_2 = \emptyset$.

(c) $\mathcal{T}_1'$ is x-local with respect to the empty signature.

We have implemented our methodology in a preliminary tool, a prototype plugin for the Protégé ontology editor. Section 3.2.5 reports on experiments that indicate that LBMs are indeed of acceptable size. The problem that remains to be solved is the difficulty to determine a suitable signature, in particular the seed signature for a module, see Section 3.2.6.

3.2.4 Robustness Properties of Locality-Based Modules

In this section we survey existing logic-based approaches to extracting a module in the sense of Definition 10, focusing on syntactic approximations via locality. We will compare different kinds of LBMs with each other and with CE-based modules. In particular, we will explore nested LBMs more in-depth than the previous literature. The comparison of module kinds will be based on examining properties relevant for ontology reuse. It will help learning which modules are best suited for which requirements.

We have seen that a module $\mathcal{T}_1$ of a TBox $\mathcal{T}$ w.r.t. a signature Σ is a subset of $\mathcal{T}$ that is indistinguishable from $\mathcal{T}$ w.r.t. Σ for certain classes of entailments. Inseparability relations a more general: they compare arbitrary ontologies that are not necessarily in the subset relation, and provide a uniform framework for comparing arbitrary definitions of modules. We study robustness properties of inseparability relations that have been identified in [KLWW09], and which are relevant for the corresponding module notions because they ensure that an ontology developer

- does not need to import a different module when she restricts the set of terms that she is interested in *(robustness under vocabulary restrictions)*;

[7] `http://www.informatik.uni-bremen.de/%7Ets/publ/safe-eco-reuse-report.pdf`

- does not need to import a different module when she extends the set of terms that she is interested in with terms that she does not use in her ontology *(robustness under vocabulary extensions)*;
- can always import the same module in different ontologies whenever she is interested in the same set of relevant terms *(robustness under replacement)*;
- does not need to import two indistinguishable versions of the same ontology *(robustness under joins)*.

The notions of modules defined so far were induced by CEs and different notions of locality. We will now put them into a more general context of modules generated by inseparability relations. For a given logic $\mathcal{L}$, an *inseparability relation* is a family $S = \{\equiv_\Sigma^S \mid \Sigma \text{ is a signature}\}$ of equivalence relations on the set of $\mathcal{L}$-TBoxes. The intuition behind this notion is as follows: $\mathcal{T}_1 \equiv_\Sigma^S \mathcal{T}_2$ means that $\mathcal{T}_1$ and $\mathcal{T}_2$ are indistinguishable w.r.t. Σ, i.e., they represent the same knowledge about the topic represented by Σ. The exact meaning of "indistinguishable" and "the same knowledge" depends on the precise definition of the inseparability relation. $\mathcal{T}_1$ being a module for Σ of $\mathcal{T}$ should be equivalent to $\mathcal{T}_1 \subseteq \mathcal{T}$ and $\mathcal{T}_1$ being inseparable w.r.t. Σ from $\mathcal{T}$.

The requirement to preserve entailments or models leads to the following inseparability relations, which have been examined, among others, in [KLWW09].

- $\mathcal{T}_1$ and $\mathcal{T}_2$ are Σ-*concept name inseparable*, written $\mathcal{T}_1 \equiv_\Sigma^c \mathcal{T}_2$, if for all Σ-concept names C, D, it holds that $\mathcal{T}_1 \models C \sqsubseteq D$ if and only if $\mathcal{T}_2 \models C \sqsubseteq D$.
- $\mathcal{T}_1$ and $\mathcal{T}_2$ are Σ-*subsumption inseparable* w.r.t. a logic $\mathcal{L}$, written $\mathcal{T}_1 \equiv_\Sigma^s \mathcal{T}_2$, if for all terms X and Y that are concept expressions over Σ or role names from Σ, it holds that $\mathcal{T}_1 \models X \sqsubseteq Y$ if and only if $\mathcal{T}_2 \models X \sqsubseteq Y$.
- $\mathcal{T}_1$ and $\mathcal{T}_2$ are Σ-*model inseparable*, written $\mathcal{T}_1 \equiv_\Sigma^{\text{sem}} \mathcal{T}_2$, if $\{\mathcal{I}|_\Sigma \mid \mathcal{I} \models \mathcal{T}_1\} = \{\mathcal{I}|_\Sigma \mid \mathcal{I} \models \mathcal{T}_2\}$.

We denote the respective sets of inseparability relations with S^c, S^s, and S^{sem}. It is easy to see that, for each signature Σ, it holds that $\equiv_\Sigma^{\text{sem}} \subseteq \equiv_\Sigma^s \subseteq \equiv_\Sigma^c$.

Inseparability relations induce modules as follows.

Definition 20. Let S be an inseparability relation, $\mathcal{T}_1 \subseteq \mathcal{T}$ TBoxes, and Σ a signature. $\mathcal{T}_1$ is

1. an S_Σ-*module of* $\mathcal{T}$ if $\mathcal{T}_1 \equiv_\Sigma^S \mathcal{T}$;
2. a *self-contained* S_Σ-*module of* $\mathcal{T}$ if $\mathcal{T}_1 \equiv_{\Sigma \cup \mathsf{Sig}(\mathcal{T}_1)}^S \mathcal{T}$;
3. a *depleting* S_Σ-*module of* $\mathcal{T}$ if $\emptyset \equiv_{\Sigma \cup \mathsf{Sig}(\mathcal{T}_1)}^S \mathcal{T} \setminus \mathcal{T}_1$.

$\mathcal{T}_1$ is called a *minimal* (self-contained, depleting) $\equiv_\Sigma$-module of $\mathcal{T}$ if $\mathcal{T}_1$, but no proper subset of $\mathcal{T}_1$, is a (self-contained, depleting) $\equiv_\Sigma$-module of $\mathcal{T}$.

Due to the shift from Σ to $\Sigma \cup \mathsf{Sig}(\mathcal{T}_1)$, it is not necessarily the case that every self-contained (or depleting) S_Σ-module of $\mathcal{T}$ is an S_Σ-*module of* $\mathcal{T}$. However, under certain robustness properties, this implication holds. While there can be exponentially many minimal S_Σ-modules, minimal depleting modules are uniquely determined – under mild conditions involving inseparability relations. This last property makes depleting modules essential for at least two applications:

The first application is the above described import scenario. If $\mathcal{T}_1$ is a depleting S_Σ-module of $\mathcal{T}$ and S satisfies certain robustness properties, then, for every external TBox $\mathcal{T}'$ that shares with $\mathcal{T}$ only terms from $\Sigma \cup \mathsf{Sig}(\mathcal{T}_1)$, we can import $\mathcal{T}'$ into $\mathcal{T} \setminus \mathcal{T}_1$ because they do not interfere with each other: $(\mathcal{T} \setminus \mathcal{T}_1) \cup \mathcal{T}' \equiv_{\Sigma'}^S \mathcal{T}'$.

The second application is the computation of all justifications for an entailment η of an ontology $\mathcal{T}$ [HPS08]. For appropriate inseparability relations S, Definition 20 (1) ensures that each $S_{\mathsf{Sig}(\eta)}$-module of $\mathcal{T}$ contains at least one justification for η, but not necessarily all. It is relatively easy to show that depletion, together with mild additional properties, guarantees that every $S_{\mathsf{Sig}(\eta)}$-module of $\mathcal{T}$ contains all justifications for η.

We define new inseparability relations for LBMs, proceeding by analogy to inseparability relations for conservativity: TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ are inseparable if they have the same modules, i.e., if the module extraction algorithm returns the same output for each of them. We have thus replaced the semantic criterion "the same entailments/models" in conservativity-based inseparability with a syntactic criterion "the same extraction result", given that LBMs are defined algorithmically. While this new type of inseparability imposes much stronger requirements ("have the same modules" is syntax-dependent), it is the best available computationally cheap approximation of the computationally hard inseparability relations $\equiv^{s}_{\Sigma}$ and $\equiv^{\text{sem}}_{\Sigma}$, analogously to how LBMs approximate CE-based modules.

We consider the following inseparability relations for LBMs.

Relation	$\mathcal{T}_1$ and $\mathcal{T}_2$ are in relation if ...
$\equiv^{\emptyset}_{\Sigma}$	$\emptyset$-mod$(\Sigma, \mathcal{T}_1) = \emptyset$-mod$(\Sigma, \mathcal{T}_2)$
$\equiv^{\Delta}_{\Sigma}$	Δ-mod$(\Sigma, \mathcal{T}_1) = \Delta$-mod$(\Sigma, \mathcal{T}_2)$
$\equiv^{\bot}_{\Sigma}$	$\bot$-mod$(\Sigma, \mathcal{T}_1) = \bot$-mod$(\Sigma, \mathcal{T}_2)$
$\equiv^{\top}_{\Sigma}$	$\top$-mod$(\Sigma, \mathcal{T}_1) = \top$-mod$(\Sigma, \mathcal{T}_2)$

Relation	$\mathcal{T}_1$ and $\mathcal{T}_2$ are in relation if ...
$\equiv^{\top\bot}_{\Sigma}$	$\top\bot$-mod$(\Sigma, \mathcal{T}_1) = \top\bot$-mod$(\Sigma, \mathcal{T}_2)$
$\equiv^{\bot\top}_{\Sigma}$	$\bot\top$-mod$(\Sigma, \mathcal{T}_1) = \bot\top$-mod$(\Sigma, \mathcal{T}_2)$
$\equiv^{\top\bot^*}_{\Sigma}$	$\top\bot^*$-mod$(\Sigma, \mathcal{T}_1) = \top\bot^*$-mod$(\Sigma, \mathcal{T}_2)$

Evidently, they all are equivalence relations. Furthermore, they are related to the previously defined inseparability relations as follows.

Proposition 21. *Let Σ be a signature. For any $\equiv^{\bullet}_{\Sigma}$ of the above 7 relations, we have $\equiv^{\bullet}_{\Sigma} \subseteq \equiv^{\text{sem}}_{\Sigma}$.*

We can now define the four robustness properties described above.

Definition 22. Let $\mathcal{L}$ be a DL. The inseparability relation S is called

1. *robust under vocabulary restrictions* if, for all $\mathcal{L}$-TBoxes $\mathcal{T}_1, \mathcal{T}_2$ and all signatures Σ, Σ' with $\Sigma \subseteq \Sigma'$, the following holds: if $\mathcal{T}_1 \equiv^{S}_{\Sigma'} \mathcal{T}_2$, then $\mathcal{T}_1 \equiv^{S}_{\Sigma} \mathcal{T}_2$.
2. *robust under vocabulary extensions* if, for all $\mathcal{L}$-TBoxes $\mathcal{T}_1, \mathcal{T}_2$ and all signatures Σ, Σ' with $\Sigma' \cap (\mathsf{Sig}(\mathcal{T}_1) \cup \mathsf{Sig}(\mathcal{T}_2)) \subseteq \Sigma$: if $\mathcal{T}_1 \equiv^{S}_{\Sigma} \mathcal{T}_2$, then $\mathcal{T}_1 \equiv^{S}_{\Sigma'} \mathcal{T}_2$.
3. *robust under replacement* if, for all $\mathcal{L}$-TBoxes $\mathcal{T}_1, \mathcal{T}_2$, all signatures Σ and every $\mathcal{L}$-TBox $\mathcal{T}$ with $\mathsf{Sig}(\mathcal{T}) \cap (\mathsf{Sig}(\mathcal{T}_1) \cup \mathsf{Sig}(\mathcal{T}_2)) \subseteq \Sigma$, the following holds: if $\mathcal{T}_1 \equiv^{S}_{\Sigma} \mathcal{T}_2$, then $\mathcal{T}_1 \cup \mathcal{T} \equiv^{S}_{\Sigma} \mathcal{T}_2 \cup \mathcal{T}$.
4. *robust under joins* if, for all $\mathcal{L}$-TBoxes $\mathcal{T}_1, \mathcal{T}_2$ and signatures Σ with $\mathsf{Sig}(\mathcal{T}_1) \cap \mathsf{Sig}(\mathcal{T}_2) \subseteq \Sigma$ and every $i = 1, 2$, the following holds: if $\mathcal{T}_1 \equiv^{S}_{\Sigma} \mathcal{T}_2$, then $\mathcal{T}_i \equiv^{S}_{\Sigma} \mathcal{T}_1 \cup \mathcal{T}_2$.

We have examined the properties of the inseparability relations $S^{\emptyset}$, S^{Δ}, $S^{\bot}$, $S^{\top}$, $S^{\top\bot}$, $S^{\bot\top}$, $S^{\top\bot^*}$, and compared them with those of S^{sem}, S^{s}, S^{c}. The results are given in Table 6. The first four lines give the module notion which is generalized by the respective inseparability relation, and indicate whether each such module for Σ is a (minimal) S_{Σ}-module, or a (minimal) self-contained S_{Σ}-module, or a (minimal) depleting S_{Σ}-module. For S^{c}, this question is meaningless because there is no corresponding standard module notion, apart from redefining Σ-dCEs to take only

↓ property relation →	S^{sem}	S^s	S^c	$S^\emptyset, S^\Delta, S^\bot, S^\top$	$S^{\top\bot}, S^{\bot\top}$	$S^{\top\bot^*}$
corresponding module notion	mCE	dCE	—	x-mod	yz-mod	yz^*-mod
(min.) modules	✓$^-$	✓$^-$	—	✓	✗	✓
(min.) self-contained mod.s	✗	✗	—	✓	✓	✓
(min.) depleting modules	✗	✗	—	✓	✓	✓
robustness voc. restr.	✓	✓	✓	✓	✗	✓
robustness voc. ext.	✓	(✗)	(✓)	✓	✗	✓
robustness replacement	✓	(✗)	(✗)	✓	✓	✓
robustness joins	✓	(✗)	(✗)	✓	✓	✓

Symbols:		
	✓, ✗	property holds/fails
	(✓), (✗)	property holds/fails for many standard description logics
	✓$^-$	property holds except for minimality
	—	property not considered: no corresponding module notion

Table 6: Properties of inseparability relations for different module notions

Σ-concept inclusions into account. For S^{sem} and S^s, it is clear that a dCE-based module according to Definition 10 (and the analogous notion of an mCE-based module) is not necessarily minimal, self-contained, or depleting. These properties can, however, easily be achieved by adopting stronger module notions as in [KLWW08, KPS$^+$09]. Hence, the negative entries in this part of the table are not problematic, and we can say that LBMs and conservativity-based modules are equally "good" – except for nested, non-iterated $\top\bot$- and $\bot\top$-modules: a $\top\bot$-module ($\bot\top$-module) is not always an $S_\Sigma^{\top\bot}$-module ($S_\Sigma^{\bot\top}$-module), which is critical. However, it is always a minimal self-contained and depleting $S_\Sigma^{\top\bot}$-module ($S_\Sigma^{\bot\top}$-module).

The remaining four lines indicate whether the respective inseparability relation satisfies the four robustness properties from Definition 22. The results for S^{sem}, S^s and S^c are taken from [CHKS08, KLWW09] those for the locality-based relations are our contribution. This part of the table reveals the following insights: Both S^c and S^s lack some of the important robustness properties (but this was already known [KLWW09]). As for the locality-based inseparability relations, it is truly surprising that $S^{\top\bot}$ and $S^{\bot\top}$ lack two of four robustness properties, while $S^\emptyset$, S^Δ, $S^\bot$, $S^\top$, as well as $S^{\top\bot^*}$ appear to be flawless and as good as S^{sem}.

In summary, semantic and syntactic LBMs are very robust: two out of three variants enjoy the same robustness properties as mCE-based modules and are therefore even more robust than modules based on dCEs. In addition, they are self-contained and depleting. Furthermore, syntactic LBMs can be extracted efficiently for all logics up to $\mathcal{SROIQ}$. For most DLs, where neither semantic LBMs nor dCE- or mCE-based modules can be extracted efficiently, syntactic LBMs seem best suited to module extraction scenarios because their smallest variant $\top\bot^*$ combines desirable robustness properties with computational feasibility.

3.2.5 Empirical Studies of Locality-Based Modules

To demonstrate the suitability of LBMs for the reuse scenario, we carried out three experimental studies evaluating the size of the resulting modules and their extraction time, and comparing them with other module notions. The driving force for these experiments was the natural question of whether LBMs differ from minimal coverage-providing modules in practice and, if yes, by how

much. In other words, are LBMs a good approximation of minimal coverage-providing modules? Unfortunately, this question is impossible to answer in general because undecidability of mCEs and dCEs for DLs of higher expressivity forbids the extraction of minimal coverage-providing modules. However, it is possible to empirically answer the following questions:

1. How large can LBMs of real-life ontologies become? Can we expect LBMs to be significantly smaller than the whole ontology, for seed signatures of reasonable size?
2. For ontologies in lightweight DLs, which permit effective extraction of mCE- or dCE-based modules, do LBMs differ in size and extraction time from the latter and, if yes, by how much?
3. Do syntactic and semantic LBMs differ in size and extraction time and, if yes, by how much? That is, are syntactic LBMs a good approximation of semantic LBMs?

Answers to these questions will allow for a more informed choice of which module extraction technique to select under a given set of circumstances.

Initial experiment evaluating $\bot$- and $\top\bot$-modules. In [JCS+08] we report on an experiment towards Question 1. We extracted $\bot$- and $\top\bot$-modules from Galen, NCI, and SNOMED CT for (a) random seed signatures of varying size and (b) real-life seed signatures from the Health-e-Child project. The modules obtained in (a) were small on average, and their growth relative to the signature size was smooth and always linear up to signature size 100. For (b), seed signatures of size 40–131 from Galen (4170 axioms) led to LBMs of size 490–1151; signatures of size 48–356 from NCI ($\approx$ 400,000 axioms) led to LBMs of size 300–1258.

In this study we also compared $\bot$- and $\top\bot$-modules for Galen and NCI with ontology fragments obtained by Seidenberg's segmentation approach [Sei09], which does not provide coverage or a similar logical guarantee. The $\top\bot$-modules were always smaller than the Seidenberg fragments for the same seed signatures; in the case of Galen, they were orthogonal.

Comparison with other approaches. In [SSZ09] we report on experiments towards Questions 1 and 2. We extracted $\bot$- and $\top\bot^*$-modules for ontologies in lightweight DLs and compared them with mCE- or dCE-based modules, which can be extracted effectively in these DLs.

We first took three real-life seed signatures for SNOMED CT with terms relevant for an intensive care unit, and compared the $\bot$- and $\top\bot^*$-modules for these signatures with the depleting mCE-based modules extracted efficiently via the MEX approach [KLWW08]. The LBMs extracted from the $\approx$ 400,000 axioms in SNOMED for these signatures of size 2,700–24,000 were of size 15,000–56,000 and thus comprised 4–15% of the whole ontology. In contrast, the size of the MEX modules ranged between 1,7 and 10%. All modules were extracted in 1–4 seconds.

Second, we extracted LBMs and minimal conservativity-based modules from two DL-Lite$^{\mathcal{N}}_{\text{bool}}$ ontologies from a commercial supply-chain management system in [KPS+09] (not part of this thesis). We extracted $\top\bot^*$-modules as a preprocessing step to computing minimal S^c-, S^q- (based on query-inseparability), and depleting S^q-modules using QBF solvers. We included all singleton seed signatures and randomly generated seed signatures of size 10; the two ontologies were of size 1247 and 1283. The difference in size between LBMs and S^c- or S^q-modules was often considerable, particularly when the S^c-modules consisted only of a few axioms. However, most of the difference seems to be caused by depletion because, in 5 out of 6 cases, the average size of LBMs was limited to 150% of the average size of depleting S^q-modules. Each LBM was extracted in a few seconds, while a depleting S^q-module took up to 30 minutes to compute. The

experimental results in our follow-up work [KKL+11] (not part of this thesis) on modules based on query inseparability for DL-Lite$^{\mathcal{N}}_{\text{core}}$, which underlies the OWL 2 QL profile, are similar.

In-depth comparison of syntactic LBMs with semantic LBMs and mCE-based modules. In [DKP+13] we report on a systematic experiment towards Questions 2 and 3. We chose a large corpus consisting of most ontologies from the NCBO BioPortal ontology repository[8] and selected ontologies from the TONES repository[9]. The 242 ontologies in this corpus cover a wide range of expressivity (from $\mathcal{AL}$ to $\mathcal{SROIQ}(\mathcal{D})$) and size (10–16,066 axioms, 10–16,068 terms). Each ontology was additionally trimmed to a subset that is acyclic and in $\mathcal{ELI}$, in order to be able to apply MEX. Furthermore we extended the reasoner FaCT++ to be able to test (semantic) Δ-locality, which involves tautology checks for axioms using the $\top$-role in a way that violates the OWL syntax.

We specified Questions 2 and 3 more precisely: we asked (i) whether, for a given seed signature Σ, it is likely that there is a difference between the syntactic, the semantic, and the MEX modules for Σ; if so, the size of the difference;[10] and (ii) how feasible the extraction of semantic LBMs is. For this purpose, we compared (a) $\emptyset$-semantic with $\bot$-syntactic locality, Δ-semantic with $\top$-syntactic locality, (b) $\emptyset$- with $\bot$-modules, Δ- with $\top$-modules, $\Delta\emptyset^*$- with $\top\bot^*$-modules, and (c) MEX modules with $\Delta\emptyset^*$-modules.

Due to the recursive nature of Algorithm 1, our investigation is both on a

- *per-axiom-basis:* given axiom α and signature Σ, is it likely that α is $\emptyset$-local (Δ-local, resp.) w.r.t. Σ but not $\bot$-local ($\top$-local, resp.) w.r.t. Σ?
- *per-module basis:* given a seed signature Σ, is it likely that
  - $\bot$-mod$(\Sigma, \mathcal{O}) \neq \emptyset$-mod$(\Sigma, \mathcal{O})$, or
  - $\top$-mod$(\Sigma, \mathcal{O}) \neq \Delta$-mod$(\Sigma, \mathcal{O})$, or
  - $\top\bot^*$-mod$(\Sigma, \mathcal{O}) \neq \Delta\emptyset^*$-mod$(\Sigma, \mathcal{O})$, or
  - $\Delta\emptyset^*$-mod$(\Sigma, \mathcal{O}) \neq$ MEX-mod$(\Sigma, \mathcal{O})$?

  If yes, is it likely that the difference is large?

We considered two kinds of signatures: genuine and random signatures. Genuine signatures are the signatures of single axioms in the ontology, and they are interesting for two reasons: they lead to coherent modules (i.e., modules that cannot be decomposed into the union of two $\subseteq$-incomparable modules), and every ontology has only linearly many genuine modules, as opposed to the exponential number of all signatures. For random signatures, we chose 400 distinct signatures (or all signatures if the ontology contained < 9 terms), each of which includes every term with probability 0.5. We have set the sample size to 400 because this ensures statistical significance of the results for the overall population of all modules *independently of the ontology size*. Our selection of signatures thus contains small (genuine) and large (random) signatures.

The results of our experiment show that, for the vast majority of the ontologies in our corpus, no difference between syntactic and semantic locality is observed, for all three variants $\bot$ vs. $\emptyset$, $\top$ vs. Δ, and $\top\bot^*$ vs. $\Delta\emptyset^*$. More precisely, for 209 out of 242 ontologies and *all* sampled signatures, there is no difference between syntactic LBMs and the corresponding semantic LBMs, and

[8] http://bioportal.bioontology.org

[9] http://owl.cs.manchester.ac.uk/repository/

[10] Recall: the MEX module for Σ is a subset of the semantic Σ-module, which is a subset of the syntactic Σ-module.

between any axiom being syntactically or semantically local. Hence, for these 209 ontologies, there is no statistically significant difference between semantic and syntactic locality of any kind, and between semantic and syntactic LBMs of any kind. The 209 ontologies include Galen and People, which are renowned for having unusually large $\bot$-modules [CHKS08, PS10]. In most cases, extracting a semantic and syntactic LBM each took only a few milliseconds. For two ontologies, the semantic LBM took considerably longer to extract than the syntactic: up to 5 times for nested-modules in MoleculeRole, and up to 34 times in Galen.

Differences between syntactic and semantic locality have been observed in the remaining 33 ontologies. In 6 of these, the differences are solely due to simple tautologies that have entered the published version by accident. In 21 of the remaining 27 ontologies, the differences affect only locality of single axioms, and never modules. The remaining 6 ontologies with differences in modules comprise 4 from BioPortal and 2 from TONES; their sizes range from 170 to 1925, and their expressivity from $\mathcal{AL}$ to $\mathcal{SRIQ}(\mathcal{D})$. Summarizing the detailed tables in our original work [DKP+13] and its accompanying technical report,[11] it can be argued that the observed differences are negligible for all 6 ontologies: in four cases, the syntactic and corresponding semantic LBM differ by at most 12 axioms. For a fifth ontology, differences greater than 11 axioms (or 2% of the semantic LBM) occur only for very small seed signatures, and the sixth ontology exhibits large differences only between $\top$- and Δ-modules – and never between the more economic $\top\bot^*$- and $\Delta\emptyset^*$-modules. Module extraction time was always a few milliseconds.

To explain the reasons for the observed differences between semantic and syntactic locality, we have identified and extensively described four types of axioms that cause these differences by being $\bot$-local ($\top$-local, respectively) w.r.t. some signature Σ but not $\emptyset$-local (Δ-local, respectively) w.r.t. Σ. We call these four types *culprits*. Sometimes, culprit axioms "pull" additional axioms into the syntactic LBM, due to the signature extension during module extraction (Alg. 1). We conjecture that one culprit type can be avoided by a straightforward extension to the definition of syntactic locality. For two culprit types, there is no hope for an exhaustive such extension. The fourth type simply encompasses the above mentioned tautologies.

Differences between LBMs and MEX modules have been observed for 66 of the 242 ontologies ($\approx 27\%$). The average difference in module size is at most 13 axioms for genuine seed signatures and 26 axioms (13% of the MEX module) for random seed signatures. The occurrence of differences is not correlated with the size of the original ontology, but with its expressivity – and thus with the amount of trimming necessary to obtain an acyclic $\mathcal{ELI}$ subontology. In particular, all 66 ontologies with differences had been pruned by at least 30 (and at most 12,185) axioms. Furthermore, these 66 ontologies coincide exactly with those where equivalences occur in the $\mathcal{ELI}$-TBox. Because of the low expressivity of $\mathcal{ELI}$, we conjecture that MEX- and $\Delta\emptyset^*$-modules can only ever differ due to the presence of equivalences. We cannot compare extraction times because the MEX implementation combines loading and module extraction.

In summary, we obtain three main observations from our experiments. (1) In general, there is no or little difference between semantic and syntactic locality. Hence, the computationally cheaper syntactic locality is a good approximation of semantic locality. (2) In most cases, there is no or little difference between LBMs and MEX modules. (3) Though in principle hard to compute, semantic LBMs can be extracted quite fast in practice, although often not as fast as syntactic LBMs. Due to results (1) and (2), hardly any benefit can be expected from preferring potentially smaller modules (MEX or semantic LBMs) to cheaper syntactic LBMs. In the course of our study we furthermore provided the first implementation of semantic LBMs.

[11] https://sites.google.com/site/cheapischeerful/technical-report

3.2.6 Discussion

The task of extracting one coverage-providing module given a seed signature is well understood and starting to be deployed in standard ontology development environments, such as Protégé 4 and in standalone tools. The extraction of LBMs has already been effectively used in the field for ontology reuse [JJBR08] as well as a subservice for incremental reasoning [CHKS10] and modular reasoning [ACH12, TP12].

The difficulty that still has to be overcome is the necessity for the user to know in advance the right seed signature for a suitable module; i.e., Question 4 from Section 3.1 remains open. There is currently no general method for constructing a seed signature from an intuitive description of a topic (such as "all the knowledge about anatomy in NCI"). Even if the final signature of the module were known, it would not help to determine the seed signature because of the non-obvious relations between the two. As a result, users are often unsure how to generate a proper request and confused by the results. To provide guidance for choosing the right seed signature, an appropriate methodology has yet to be developed, and there are at least two possibilities. The first is to develop a fundamental theory of topicality, connecting the cognitive notion of a topic with the logical notion of a vocabulary. First steps in this direction were made by Del Vescovo et al. in [DPS11], who identified five notions of logical coherence as being central for a theory of topicality, and showed how to base one of these five on labeled atomic decomposition, see also the discussion in Section 3.3.6. The second possibility is more heuristic: it consists in providing tool support that allows the user to "shop" for symbols to include in the seed signature and inspect a preview of the resulting module until she is satisfied with it.

3.3 Logic-Based Modularization

In the previous section, we have studied the task of extracting one module given a seed signature. This task, from now on called GetOne, is an important and useful service but, by itself, it tells us nothing about the modular structure of the ontology as a whole. The modular structure is determined by the set of *all* modules and their inter-relations, or at least a suitable subset thereof. The task of a-posteriori determining the modular structure of an ontology is called GetStruct, and the task of extracting all modules is called GetAll. While GetOne is well-understood and often computationally cheap, GetAll and GetStruct have not been examined for module notions with strong logical guarantees, with a few preliminary exceptions, see Sections 3.3.1 and 3.3.2. If ontology engineers had access to the overall modular structure of the ontology determined by GetStruct, they might be able to use it to guide their extraction choices and, supported by the experience described in [CPSK06], to understand its topicality, connectedness, structure, superfluous parts, or agreement between actual and intended modeling. For example, by inspecting the modular structure and observing unconnected parts that are intended to be connected, ontology designers could learn of weakly modeled parts of their ontology.

In the worst case, however, the number of all modules of an ontology is exponential in the number of terms or axioms in the ontology, as we will see in Section 3.3.2. More importantly, even for very small ontologies, the number of all modules is far too large for them to be inspected by a user or even computed; e.g., the tutorial ontology Koala consisting of 42 axioms has $3,660$ modules, and GetAll fails even on many ontologies consisting of less than one hundred axioms.

In Section 3.3.3, we report on new insights regarding the modular structure of ontologies which leads to a new algorithm for GetStruct that generates a linear-sized, partially ordered set of modules and *atoms* which succinctly represent *all* modules of an ontology. This algorithm can

be used with arbitrary module notions that satisfy reasonable properties, including LBMs, and it runs in polynomial time modulo module extraction. In Section 3.3.4, we describe an algorithm for fast module extraction based on the result of our algorithm for GetStruct. In Section 3.3.5, we report on systematic experiments carried out with an implementation of the algorithms for AD and fast module extraction.

3.3.1 Existing Approaches

One solution to GetStruct is described in [CPSK06] via partitions related to $\mathcal{E}$-connections. When this approach succeeds, it partitions an ontology into three kinds of subsets: (A) those which import vocabulary (and axioms) from others, (B) those whose vocabulary (and axiom set) is imported, and (C) isolated subsets. In various experiments, the numbers of subsets extracted were quite low, and the graph induced by the "imports" relation between the subsets often corresponded usefully to user understanding. For example, the tutorial ontology Koala, consisting of 42 logical axioms, was partitioned into one A-subset about animals and three B-subsets about genders, degrees and habitats, which are "imports"-related to the former. Furthermore, certain combinations of these subsets provide coverage [CPSK06]. For Koala, the only such combination is the whole ontology (though smaller parts of Koala have coverage as well).

This kind of ontology partitions requires rather strong conditions to ensure modular separation. They have been observed to force together axioms and terms which are logically separable. As a consequence, some ontologies with a fairly elaborate modular structure have impoverished $\mathcal{E}$-connections based structures. Furthermore, the robustness properties of the parts (e.g., under vocabulary extension) are not as well-understood as those of locality-based modules. Partitions ensure, however, a linear upper bound on the number of subsets.

Another approach to GetStruct was described in [BFZE08] and implemented in ModOnto, a tool aiming at providing support for working with ontology modules, borrowing intuitions from software modules. To the best of our knowledge, however, it is not known whether such modules provide coverage. Furthermore, ModOnto does not aim at obtaining a *complete* modularization.

A further method for partitioning an ontology is developed in [SK04]. It only takes the concept hierarchy of the ontology into account and does therefore not provide coverage.

In [KLPW10], a method for decomposing the *signature* of an ontology is described, with the aim of disclosing the dependencies between its terms. In contrast to previous such approaches, this one is syntax-independent. While disclosing term dependencies is one goal of the approach we develop in Section 3.3.3, we are also interested in the *modules* of the ontology.

3.3.2 The Modular Structure of an Ontology

We are interested in the question whether the task GetAll can be used as a basis for GetStruct, that is, whether the modular structure of the ontology as a whole can be determined by extracting the set of *all* modules, for some module notion with strong logical guarantees. In the worst case, an ontology can have exponentially many subsets that are modules, which would make GetAll infeasible. However, for module notions such as LBMs, which are determined by a seed signature, it is easy to observe that one module can have several seed signatures. Hence it is conceivable that, for most real ontologies, only a small proportion of their subsets are in fact modules; ideally, those could be used directly for GetStruct.

In this section, we report on experiments testing the hypothesis that real ontologies tend to have a small number of modules. We extracted all syntactic LBMs from real ontologies and

counted their *number*, to find out whether the suspected combinatorial explosion occurs. In order to determine the *growth rate* of module numbers depending on the ontology size, we also sampled subsets of the ontologies and counted their module numbers.

We performed the experiments on several existing ontologies that we consider to be well designed and sufficiently diverse. "Well designed" means that these ontologies cover a specific domain to a certain level of detail; they are axiomatically rich, for example, they do not only connect terms via atomic subsumptions, which would make module extraction rather uninteresting because the terms in the signature of a module would hardly cause other terms to be included in the module. We concentrate on well-designed ontologies because we want to *understand* their structure. "Diverse" means that these ontologies have different sizes, expressivities, ratios of axiom and term numbers, and cover different domains. We also selected two ontologies which were successfully and insightfully modularized in [CPSK06]: Koala and OWL-S. Our selection constitutes a set of ontologies which are commonly discussed in ontology engineering circles and for which people have strong instincts about their modular structure. For practical reasons, we restricted ourselves to small ontologies.

Table 7 gives an overview; most ontologies can be found in the TONES repository.[12]

Name	DL expressivity	#Logical axioms	#Terms (concepts, roles)
Koala	$\mathcal{ALCON}(D)$	42	25
Mereology	$\mathcal{SHIN}$	44	25
University	$\mathcal{SOIN}(D)$	52	39
People	$\mathcal{ALCHOIN}$	108	73
miniTambis	$\mathcal{ALCN}(D)$	173	226
OWL-S	$\mathcal{ALCHOIN}(D)$	277	137
Tambis	$\mathcal{ALCN}(D)$	595	494
Galen	$\mathcal{ALEHF}+$	4,528	3,161

Table 7: Ontologies used in the experiments

Only for the two smallest ontologies, Koala and Mereology, were we able to determine all modules within less than two hours. For the other five, we canceled the computation and proceeded to subset sampling, as reported below.

Table 8 shows module numbers, overall extraction time, and distribution of module size for Koala and Mereology. We extracted syntactic $\top$-, $\bot$-, and $\top\bot^*$-LBMs (see Section 3.2.2), and we considered a special kind of $\top\bot^*$-LBMs that we have called *genuine* and denoted $\top\bot^*_g$. This notion of a genuine module is an early version and was subsequently strengthened to coincide with what is called a genuine module in Sections 3.2.5 and 3.3.3. Here, genuine $\top\bot^*$-LBMs exclude modules that are disjoint unions of smaller $\top\bot^*$-LBMs – intuitively, such disjoint unions are uninteresting for the modular structure because they witness that the seed signatures of their smaller constituent modules do not interact with each other.

The number of modules increases from $\top$- via $\bot$- to $\top\bot^*$-LBMs, which is due to the fact that, typically, $\top\bot^*$-LBMs are smaller than $\bot$-LBMs and those are smaller than $\top$-LBMs. In particular, $\top$-LBMs are notorious for comprising almost the whole ontology, which makes them unsuitable for the task GetStruct. The module numbers for the other three notions are well below the theoretical upper bound of 2^{25}, but already too large to be useful for GetStruct for these small ontologies with 42 and 44 logical axioms.

[12] `http://owl.cs.manchester.ac.uk/repository`

	Koala				Mereology			
	$\top$	$\bot$	$\top\bot^*$	$\top\bot^*_g$	$\top$	$\bot$	$\top\bot^*$	$\top\bot^*_g$
#Modules	**12**	**520**	**3,660**	**2,143**	**40**	**552**	**1,952**	**272**
Time [s]	*0*	*1*	*9*	*34*	*0*	*6*	*158*	*158*
Min. module size	29	6	0	0	18	0	0	0
Avg. module size	35	27	23	23	26	25	20	22
Max. module size	42	42	42	42	40	40	40	38
Standard deviation	4	6	6	6	6	7	8	8

Table 8: Full modularization of Koala and Mereology

To test the hypothesis that the module size grows sub-exponentially for the examined ontologies, we sampled subontologies, ordered them by size, and modularized them as before in increasing order until a single modularization exceeded a preset timeout. The results clearly refute the hypothesis: Figure 7 shows exemplary scatterplots of the number of $\top\bot^*$-LBMs (genuine $\top\bot^*$-LBMs) versus the size of the subset for People and Koala. Each chart additionally shows an exponential trend line, the least-squares fit through the data points by using an exponential equation. For more charts and spreadsheets, see the accompanying technical report.[13]

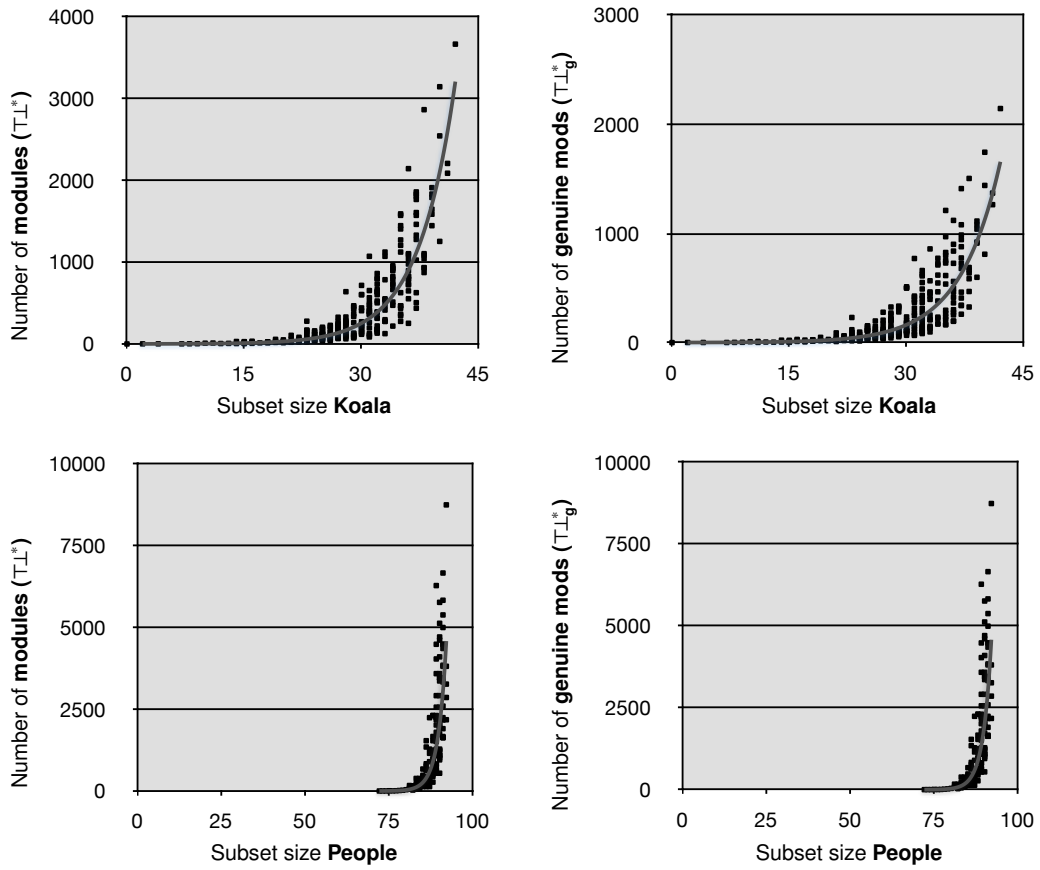


Figure 7: Numbers of modules ($\top\bot^*$, genuine $\top\bot^*$) versus subset sizes for Koala and People

[13] http://www.informatik.uni-bremen.de/%7Ets/publ/modstrucreport.pdf

Table 9 shows the trend-line equations, together with their confidence (R^2 values), and the estimated number of modules for the full ontology as per the trend-line equation. The results with confidence $> .8$ strongly suggest an exponential dependence of the module number on the subset size.

	Confidence		Trend-line equation		Estimate	
Ontology	R^2	R^2 (g)	$\top\bot^*$	$\top\bot^*_g$	$\top\bot^*$	$\top\bot^*_g$
People	.95	.95	$2 \cdot 10^{-13} e^{.41n}$		10^6	10^6
Mereology	.87	.94	$1.2e^{.16n}$	$1.1e^{.13n}$	10^3	10^2
Koala	.90	.88	$.45e^{.21n}$	$.50e^{.19n}$	10^3	10^3
Galen	.94	.86	$1.2e^{.24n}$	$1.6e^{.16n}$	$\gg 10^{99}$	$\gg 10^{99}$
University	.84	.83	$1.7e^{.19n}$	$1.6e^{.14n}$	10^4	10^3
OWL-S	.82	.84	$.0027e^{.17n}$	$.0032e^{.16n}$	10^{17}	10^{17}
Tambis	.75	.70	$1.1e^{.22n}$	$1.4e^{.13n}$	10^{58}	10^{33}
miniTambis	.47	.52	$2.6e^{.18n}$	$2.5e^{.14n}$	10^{14}	10^{10}

R^2, R^2 (g) Determination coefficient of trend lines ($\top\bot^*$, $\top\bot^*_g$)
Estimate Module numbers for full ontology as per trend line

Table 9: Witnesses for exponential behavior

The fundamental conclusion is that the number of modules is exponential in the size of the ontology for real ontologies, refuting the initial hypothesis. Our estimates show that full modularization is practically impossible already for midsize ontologies. As a consequence, GetStruct cannot be solved via GetAll but rather requires a more sophisticated approach that contents itself with a small amount of module extraction. We develop such an approach in the following.

3.3.3 Atomic Decomposition: Foundations

In this section, we are presenting an approach to GetStruct that represents the set of all modules of an ontology $\mathcal{O}$ by a small number of subsets of $\mathcal{O}$, called *atoms*, and by a dependency relation between those. This approach is called *atomic decomposition*. It can be applied to any notion of a module that is determined by a seed signature and is monotonic, self-contained, and depleting (and we know from our work in [SSZ09] that robustness under replacement and depletion implies self-containment). In what follows we will use m to denote any such module notion unless we make explicit restrictions.

As usual, we view $\mathcal{O}$ as a set of logical axioms. To ease presentation we assume, without loss of generality, that $\mathcal{O}$ contains no syntactic tautologies (those axioms occur in no m-module) and no global axioms (these are axioms that occur in all m-modules). They can be easily removed by computing $m\text{-mod}(\mathsf{Sig}(\mathcal{O}), \mathcal{O})$ and $m\text{-mod}(\emptyset, \mathcal{O})$ and considered separately.

Borrowing standard notions from algebra, we consider the field of subsets of $\mathcal{O}$ that is induced by the family of all m-modules of $\mathcal{O}$:

Definition 23. Let $\mathcal{O}$ be an ontology and $\mathfrak{F}_m(\mathcal{O})$ the family of m-modules of $\mathcal{O}$. The *(induced) field of modules* $\mathcal{B}(\mathfrak{F}_m(\mathcal{O}))$ is the closure of $\mathfrak{F}_m(\mathcal{O})$ under union, intersection, and complement.

$\mathcal{B}(\mathfrak{F}_m(\mathcal{O}))$ is thus a family of subsets of $\mathcal{O}$ that is closed under the set operations. This field of sets is partially ordered by the $\subseteq$ relation and, since $\mathcal{O}$ is finite, can be represented via its Hasse diagram. The $\subseteq$-minimal elements of $\mathcal{B}(\mathfrak{F}_m(\mathcal{O})) \setminus \{\emptyset\}$ are called *atoms*. They are pairwise disjoint

and thus form a partition of $\mathcal{O}$; in particular the number of atoms is bounded by the number of axioms in $\mathcal{O}$. Furthermore, every module is the union of some atoms. Hence the set of atoms is a succinct representation of all modules of $\mathcal{O}$ – provided that we equip it with information that allows to reconstruct the modules. Even more, we will see that the set of atoms can be computed in polynomial time modulo module extraction.

Since modules provide coverage, it is natural to say that an axiom α logically depends on another axiom β if, whenever α occurs in a module $\mathcal{M}$, then β also belongs to $\mathcal{M}$. This idea can be generalized to atoms:

Definition 24. Let $\mathfrak{a}$ and $\mathfrak{b}$ be two distinct atoms of an ontology $\mathcal{O}$. We say that

- $\mathfrak{a}$ *depends* on $\mathfrak{b}$ (written $\mathfrak{a} \geq \mathfrak{b}$) if, for every module $\mathcal{M} \in \mathfrak{F}_m(\mathcal{O})$ such that $\mathfrak{a} \subseteq \mathcal{M}$, we have $\mathfrak{b} \subseteq \mathcal{M}$.
- $\mathfrak{a}$ and $\mathfrak{b}$ are *independent* if there exist two disjoint modules $\mathcal{M}_1, \mathcal{M}_2 \in \mathfrak{F}_m(\mathcal{O})$ such that $\mathfrak{a} \subseteq \mathcal{M}_1$ and $\mathfrak{b} \subseteq \mathcal{M}_2$.

For a monotonic, self-contained, and depleting module notion, Definition 24 describes the possible relations between atoms exhaustively, which relies on the following insight.

Proposition 25. *For every atom $\mathfrak{a}$ in $\mathcal{B}(\mathfrak{F}_m(\mathcal{O}))$ and every nonempty set of axioms $\{\alpha_1, \ldots, \alpha_k\} \subseteq \mathfrak{a}$, we have that m-mod$(\{\mathsf{Sig}(\alpha_1) \cup \ldots \cup \mathsf{Sig}(\alpha_k)\}, \mathcal{O})$ is the smallest m-module containing $\mathfrak{a}$.*

Let $\mathcal{M}_\alpha = m\text{-mod}(\mathsf{Sig}(\alpha), \mathcal{O})$ for any axiom $\alpha \in \mathcal{O}$. As a consequence of Proposition 25, we get:

Corollary 26. *Given an atom $\mathfrak{a}$, for any axiom $\alpha \in \mathfrak{a}$ we have that $\mathcal{M}_\alpha = m\text{-mod}(\mathsf{Sig}(\mathfrak{a}), \mathcal{O})$. Moreover, $\mathfrak{a}$ depends on all atoms belonging to $\mathcal{M}_\alpha \setminus \mathfrak{a}$.*

The binary relation $\geq$ is a partial order over the set of atoms in $\mathcal{B}(\mathfrak{F}_m(\mathcal{O}))$.

We call the set of atoms with the dependency relation $\geq$ *the atomic decomposition* of $\mathcal{O}$, written AD($\mathcal{O}$). Definition 24 and Corollary 26 allow us to draw a Hasse diagram for AD($\mathcal{O}$), too, see Figure 8 for that of Koala. Atoms are scaled by their size; the arrows represent the $\geq$ relation. There are 23 atoms; four of them are isolated (numbers 9, 13, 15, 22), and the remaining atoms all depend on two large ones (numbers 7, 5). Those two consist of definitions of the terms central to the ontology: `Animal`, `Male`, `Female`, `Gender`, `hasGender`, and `hasHabitat`.

As an immediate consequence of our observations so far, a module is a disjoint finite union of atoms:

Definition 27. The *principal ideal of an atom* $\mathfrak{a}$ is the set $(\mathfrak{a}] = \{\alpha \in \mathfrak{b} \mid \mathfrak{b} \prec \mathfrak{a}\} \subseteq \mathcal{O}$.

Proposition 28. *For every atom $\mathfrak{a}$, $(\mathfrak{a}]$ is a module.*

Conversely, not every union of atoms is necessarily a module. However, we can compute the modules from AD($\mathcal{O}$) if we store the $\subseteq$-minimal seed signatures that lead to $(\mathfrak{a}]$: we say that an atom $\mathfrak{a}$ is *relevant* for a module $\top\bot^*\text{-mod}(\Sigma, \mathcal{O})$ if there is a seed signature Σ' for $(\mathfrak{a}]$ with $\Sigma' \subseteq \Sigma$.

Proposition 29. $\top\bot^*\text{-mod}(\Sigma, \mathcal{O}) = \bigcup \{(\mathfrak{a}] \mid \mathfrak{a}$ *is relevant for* $\Sigma\}$.

As a consequence, AD($\mathcal{O}$) is indeed a succinct representation of all modules: its linearly many atoms represent all its worst-case exponentially many modules. We can even compute AD($\mathcal{O}$) efficiently, i.e., by a polynomial-time algorithm with a linear number of oracle calls to a module extractor. If we use syntactic LBMs, then this algorithm has a polynomial-time overall runtime. The main ingredients are modules "generated" by single axioms, i.e., the linearly many m-mod$(\mathsf{Sig}(\alpha), \mathcal{O})$ for $\alpha \in \mathcal{O}$. We call these modules *genuine* because they can be shown to be exactly those that are not the union of two incomparable (w.r.t. set inclusion) modules.

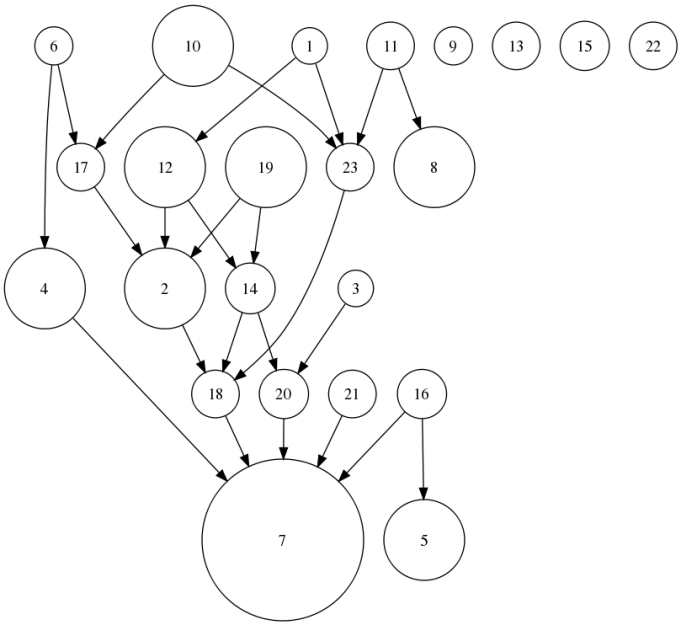


Figure 8: The atomic decomposition of Koala

3.3.4 Fast Module Extraction via Atomic Decomposition

The ability to extract modules quickly is important for ontology reuse in semantic web services such as SSWAP[14] (Simple Semantic Web Architecture and Protocol [GSM+09]), or SADI (Semantic Automated Discovery and Integration [WVM09]). The goal is to speed up module extraction by precomputing the modular structure and extracting single modules from it with little to no additional effort. This is clearly useful when many modules need to be extracted from the same ontology. In this section we describe a module extraction algorithm, called FME for "Fast Module Extraction", which is (a) usually faster than the standard ME algorithm and (b) does not require loading the entire ontology into memory.

As observed in Section 3.3.3, every module is a union of atoms, but the opposite does not hold. In general, it is hard to determine the set of atoms constituting the module for a given seed signature Σ: even axioms whose signatures are disjoint with Σ may turn out to be a part of the module. One way to help determining relevant atoms is to *label* them, i.e., add information regarding seed signatures. Here we consider labels which, for every atom $\mathfrak{a}$, provide all *minimal seed signatures* for the genuine module $(\mathfrak{a}]$, denoted by $\mathsf{MSS}(\mathfrak{a})$.

The labels $\mathsf{MSS}(\mathfrak{a})$ can have several uses. First, every $\Sigma \in \mathsf{MSS}(\mathfrak{a})$ can be regarded as a (minimal) topic that determines $(\mathfrak{a}]$ and $\mathfrak{a}$. Then all MSSs of all atoms constitute all relevant minimal topics about which the ontology speaks, which can be exploited for comprehension. Second, the collection of all MSSs guides ontology developers in the extraction of a single module by suggesting possible topics (MSSs as inputs of the extraction algorithm). In both cases, the number of topics can be exponential and thus needs to be controlled, e.g., by adjusting the granularity of the presentation, which is worth being investigated in the future.

Our algorithm for computing MSSs is restricted to $\bot$- and $\top$-modules because it crucially relies on computing the *minimal globalizing signatures (MGSs)* of α, which are the minimal signatures

[14] `http://sswap.info`

Σ such that α is local w.r.t. Σ. The notion of MGSs is not meaningful for a nested module notion such as $\top\bot^*$. The desirable transfer to $\top\bot^*$-modules would thus require an entirely new approach.

The algorithm first computes the set $\mathsf{MGS}(\alpha)$ for every axiom $\alpha \in \mathfrak{a}$, comprising the MGSs of α. For atoms $\mathfrak{a}$ that do not depend on any other atoms, $\mathsf{MSS}(\mathfrak{a})$ is the union of the $\mathsf{MGS}(\alpha)$ for all its axioms α. For the remaining atoms, the $\Sigma \in \mathsf{MGS}(\mathfrak{a})$ are not necessarily *minimal* seed signatures for $(\mathfrak{a}]$: some $\Sigma' \subsetneq \Sigma$ might be a seed signature for a module $(\mathfrak{b}]$ with $\mathfrak{b} \preceq \mathfrak{a}$ if Σ is contained in the *extended signature* $\Sigma' \cup \mathsf{Sig}((\mathfrak{b}])$ against which locality is checked when the module $(\mathfrak{b}]$ is extracted, see Algorithm 1. This constellation needs to be detected for every seed signature $\Sigma \in \mathsf{MGS}(\mathfrak{a})$ and every atom $\mathfrak{b}$ on which $\mathfrak{a}$ (transitively) depends; we call this subroutine *elaborating* Σ.

The algorithm requires time exponential in the size of the ontology in the worst case, but has the *anytime* property: the subroutine for elaborating a seed signature can be interrupted upon some timeout, which will result in computing some subset of the MSS set for an atom – in this case we call the atom *dirty*. Dirty atoms require special handling during module extraction because their relevance cannot be determined: our FME algorithm will need to include them in the module even though they may be irrelevant. Consequently, FME performance depends directly on the presence of atoms left dirty by the MSS algorithm. The amount of "dirtiness" and its impact on FME is subject of the evaluation in Section 3.3.5.

The final FME algorithm is now a simple variant of the plain module extraction algorithm (Algorithm 1), where the addition of all non-local axioms is replaced by the addition of all atoms relevant for $\Sigma \cup \mathsf{Sig}(\mathcal{M})$ (and all dirty atoms).

The FME algorithm has two important advantages over the standard ME algorithm. First, it should be faster for most ontologies because it benefits from the labeled AD in two ways: (i) it exploits labels to quickly detect relevant atoms; (ii) once an atom $\mathfrak{a}$ is established to be relevant, the corresponding module $(\mathfrak{a}]$ is added to the module without further checks. Second, it consumes substantially less memory since only relevant atoms (and their principal ideals) need to be loaded. This is especially significant when modules are small compared to the size of the ontology, which we have observed for most of the BioPortal ontologies (the median module size for small seed signatures is $< 1\%$), see Section 3.3.5.

In addition, the FME algorithm only needs access to the AD graph and labels, not to the axioms. This is relevant for maintaining large ontologies, or even large repositories thereof. Contrary to the standard ME, the FME algorithm can extract modules by loading only the axioms of relevant (and dirty) atoms. For example, if BioPortal ontologies were maintained in the decomposed form, it would be possible to provide clients, such as SSWAP, with modules for a required seed signature in a scalable (from the memory perspective) way.

3.3.5 Atomic Decomposition in Practice

This section is concerned with testing whether AD is useful in practice, providing answers to the questions of whether all or which ontologies decompose "well", what it means to decompose well, and which properties of an ontology lead to "good" decomposability. We discuss and evaluate the performance of ADs w.r.t. three specific tasks:

Task 1: Ontology reuse and collaborative development (see Scenarios 1 and 2 from Section 3.1)

Task 2: Topicality for ontology comprehension (relevant for Scenario 3 from Section 3.1)

Task 3: Fast module extraction (as discussed in Section 3.3.4)

Our results show that AD is generally a good decomposition for these three scenarios. Of course "good decomposability" may have a different meaning in other scenarios.

Task 1 involves several ontology engineers working on distinct modules of an ontology. The aim is to minimize the risk of conflicts caused by two or more collaborators making changes to logically related parts of the ontology (i.e., one engineer could be changing the semantics of terms used by another). Safety (see Section 3.2.2) defines conditions under which there is no such risk. We assume that each engineer works within *their* module and uses other terms in a safe way, and that modules assigned to different engineers do not overlap. A fine-grained decomposition is desirable here.

Task 2 is based on the assumption that understanding what an ontology deals with is fostered by searching for its "topics" and their interrelations [DPS11]. In this case, a good decomposition should provide a "bird's-eye" view of the topical structure of an ontology. This means that a very fine-grained decomposition is undesirable because it does little to help understanding. On the other hand, large clumps of axioms could aggregate, hence hide, specific topical relations. In this scenario, a good decomposition should be only modestly fine-grained. In addition, a meaningful way to label the atoms with their topics is desired.

We tested the AD on a corpus of 181 ontologies from the NCBO BioPortal, which were built by domain experts and vary greatly in size and expressivity. In order to evaluate granularity of the AD, we computed three variants, based on $\bot$-, $\top$-, and $\top\bot^*$-modules. The $\top\bot^*$-AD is a refinement, w.r.t. set inclusion between atoms, of both $\bot$- and $\top$-AD. The open-source Java implementation used for our experiments is available: `http://tinyurl.com/bioportalFME`

To evaluate AD for Tasks 1 and 2, we measured, for every ontology and AD variant, the average and maximal size of atoms and genuine modules (GMs), and the number of connected components (CCs) in the AD. Table 10 shows these five measures, averaged over the 181 ontologies.

Notion of locality	Average average size of atoms	Average maximum size of atoms	Average average size of GMs	Average maximum size of GMs	Average number of CCs
$\top\bot^*$	1.73	86	66	143	826
$\bot$	2.19	93	73	156	45
$\top$	330.45	1417	1166	2093	1.64

Table 10: Decomposition results for the 181 BioPortal ontologies

Table 10 shows that the $\top\bot^*$-AD is quite fine-grained: the average size of an atom is < 2 axioms; indeed, only few ontologies have atoms of size > 10. Next, $\bot$-AD is surprisingly close in granularity to $\top\bot^*$-AD; the two are correlated with strong statistical significance. They even coincide on 34 of the 181 ontologies, which will become interesting for Task 3, see below.

In contrast, $\top$-AD is substantially coarser; in particular, the average atom is by two orders of magnitude larger. Given the nature of $\top$-modules observed in Section 3.3.2, this is not surprising; thus $\top$-ADs are simply not a good choice when small atoms and modules are required.

For the majority of ontologies in our corpus, we observe a good decomposability in terms of atom size. There are, however some exceptions: 13 ontologies have $\top\bot^*$-atoms with more than 200 axioms (or 50% of all axioms). Some of these large atoms are due to the abundance of axiom pairs of the form $\{A \equiv (B_0 \sqcup \ldots \sqcup B_n), \text{PairwiseDisjoint}(B_0, \ldots, B_n)\}$, introduced by a specific usage pattern of ontology editors. They lead to large modules due to the signature

extension in Algorithm 1; this effect can be contained by splitting up the big disjointness axioms or making disjointness implicit. A more systematic investigation of patterns that lead to huge atoms remains for future work.

For Task 1, these results are promising since they exhibit a good decomposability of ontologies for $\top\bot^*$-AD and $\bot$-AD, i.e., the existence of small, disjoint sets of axioms that can be safely updated in parallel. In contrast, for Task 2 we observe that atoms reflect only very fine-grained topics of an ontology: small atoms do not provide much summarization over axioms. However, the dependency structure reflects the logical dependency between atoms, and thus can be used to consider, e.g., dependent components which in turn may better reflect the topics of an ontology. Of course, to really support ontology comprehension, we might have to consider "most relevant" atoms of an ontology and, definitely, suitable labelings of modules. Both directions are subject to further investigation; first steps have been made by Del Vescovo et al. [DPS11, Del11, DPS12].

To evaluate AD for Task 3, we proceeded in two steps. First, we evaluated the MSS labeling algorithm on the corpus, to assess the feasibility of computing all MSS for all atoms. We set a 5-second timeout for computing labels per atom. The results are given in Table 11.

Total no. of ont.s	Avg. size of MSS(α)	Avg. number of terms in all MSS(α)	Max. size of MSS(α)	Number of ont. with dirty atoms	Max. number of dirty atoms
181	1.4	2.1	4252	5	554

Table 11: Summary of the MSS algorithm evaluation on BioPortal ontologies

For all but 5 ontologies the algorithm was able to compute all MSS for all atoms. Also, the average label size (that is, the number of MSSs per atom) and the average number of terms in all MSSs per atom are small: 1.4 and 2.1, respectively (when averaged first within an ontology then over all ontologies). This is mainly a consequence of the simplicity of the BioPortal ontologies: their atoms are relevant to only a small number of terms, which leads to a small average number of atoms (and consequently, axioms) per module, see also below.

The remaining 5 ontologies either do not decompose well or have an interesting property of the AD graph: certain atoms depend *non-transitively* on a high number of other atoms. Both reasons apply to two ontologies for which the MSS algorithm left a large number of atoms dirty and managed to compute up to several thousand MSS per dirty atom. It would be interesting to systematically investigate cases where a subset of an ontology turns out to be relevant for such a high number of distinct, but overlapping, seed signatures, but this is subject to future work.

In the second step, we ran the FME algorithm on the obtain labeled ADs. We randomly generated 300 seed signatures consisting of concept names for every ontology, in three groups of sizes $2, 5, 10$. For each size group, we ran both FME and ME algorithms and averaged the results over all 100 runs. Cumulative statistics for all 181 ontologies are presented in Table 12. Correctness of the FME algorithm was verified by comparison with the standard ME algorithm.

The results show that good decomposability of BioPortal ontologies indeed implies small modules on average (column 2) and that the FME algorithm is typically faster than the standard ME algorithm – but this depends on several factors: decomposability of the ontology (affecting the size of the module), the average number of labels per atom (determining performance of the relevance check), and the size of the seed signature (determining the number of relevant atoms). The results include the ontologies with abundant dirty atoms, where the FME algorithm was up to 5 times slower, which supports the importance of efficient labeling.

Size of seed sig.	Avg. (median) relative module size in %	Number of positive cases	Avg. ME runtime (ms)	Avg. FME speed-up	Max. FME speed-up
2	0.77 (0.04)	173	1.09	7.33	37.28
5	0.91 (0.08)	169	1.15	3.86	27.12
10	0.99 (0.13)	150	1.18	2.48	8.34

"Relative module size" = size of the module divided by the size of the ontology
"Positive cases" = ontologies for which FME was faster than ME
"Avg. (max.) speed-up" = average (maximal) value of ME time divided by FME time

Table 12: Summary of the FME evaluation on BioPortal ontologies

3.3.6 Discussion and Outlook

We have introduced the atomic decomposition of an ontology, and shown how it is a succinct, tractable representation of the modular structure of an ontology. Moreover, it can be used to assemble all other modules without touching the whole ontology and without invoking a direct module extractor. We have furthermore shown that the majority of ontologies from BioPortal decompose well, discussed possible reasons for poor decomposability, and implications of decomposability for possible use cases, including semantic web service annotation and discovery. In addition, we have devised novel AD-based algorithms for labeling and fast module extraction.

Future work should investigate in depth applications and applicability of AD in ontology engineering. In particular, precise methodologies are sought for using AD in Scenarios 2 (collaborative development) and 3 (comprehension) from Section 3.1, and for answering Question 4 (identifying a suitable module for reuse). All these tasks require a more intuitive labeling of the atoms; alternative labeling approaches have been described by Del Vescovo et al. in [Del11], where further applications of AD are described, as well as in [DPS11] together with a theory of topicality, and in [DPS12] together with a rigid notion of logical relevance.

In addition, AD bears the potential to support the maintenance of ontologies in a decomposed form, which requires further investigation. The benefits are not restricted to reuse (through fast module extraction); we also expect benefits for ontology engineering (through the ability to load, work on, and reason over a smaller part), collaborative development (by obtaining a suitable partition from the AD), and ontology comprehension (through the ability to focus on small parts at a time and understand their interactions). Since the computation of AD can be time-consuming, it is desirable to be able to incrementally update the AD. Our first steps in this direction are restricted to axiom additions [KDS12]. The maintenance of ontologies in a decomposed form also requires a better understanding of good and poor decomposability, and modeling guidelines for developing well decomposable ontologies. Furthermore, since LBMs are syntax-dependent, it is important to understand the robustness of AD based on LBMs under certain "refactorings" that naturally occur in the development cycle (such as definitorial extensions – the introduction of new names for reoccurring complex concepts).

Finally, an extension of our FME algorithms to $\top\bot^*$-modules is desirable but nontrivial.

Chapter 4

Branching-Time Temporal Description Logics

4.1 Introduction

Classical description logics (DLs) aim at the representation of, and reasoning about, *static* knowledge. The objective of enhancing DLs with means to capture *temporal* aspects of knowledge has, over the past 20 years, led to much effort being spent on the study of temporal description logics (TDLs), as discussed in detail in the surveys [AF00, LWZ08] and the references therein. TDLs are useful for applications where knowledge involving a dynamic aspect is to be represented. For example, it is desirable to model in SNOMED CT concepts such as "a patient who may need a blood transfusion in the future" or "a disease whose occurrence may cause certain other diseases in the future".

A prominent approach to TDLs, following Schild's original proposal [Sch93], is to combine DLs with the standard temporal logics LTL and CTL(*), based on a two-dimensional product-like semantics in the style of many-dimensional DLs [GKWZ03]. In addition to choosing the component logics, the design of a TDLs requires further choices: whether temporal operators are applied to concepts, roles and/or TBoxes, or whether we want to define some DL roles or concepts as rigid, which would mean that they will not change their interpretation over time. For example, since every human has the same genetic disorders during their life, the relation "has genetic disorder" between humans and diseases should be modeled by a rigid role hasGeneticDisease. Consequently, it has been argued that rigid roles are required in applications of TDLs, e.g., in temporal data modeling [AKRZ14] or in medical ontologies such as SNOMED CT.

If the temporal component is chosen to be LTL, we speak of *linear-time TDLs*. For these *LTDLs*, the landscape of the expressivity and computational complexity is well-understood [AKL$^+$07, BGL08, FT11, AKRZ14]. Various design choices have been explored, and both lightweight and expressive DLs have been considered. An important insight gained is that LTDLs based on $\mathcal{ALC}$ become undecidable in the presence of general TBoxes as soon as temporal operators are applied to concepts *and* roles, or if rigid roles are allowed.

More generally, combinations of DLs and modal logics allowing for rigid roles and general TBoxes tend to have very high computational complexity. This can be partly explained by the well-known correspondence of DL combinations with product modal logics (PMLs) [GKWZ03, Chapters 3, 14]: the standard reasoning problem in TDLs, concept satisfiability with respect to a TBox, is then a variant of the global consequence problem in the corresponding PML. Global consequence is typically undecidable in PMLs [GKWZ03].

In the context of LTDLs, recent work has aimed at regaining decidability in the presence of rigid roles and general TBoxes, investigating LTDLs based on *lightweight DLs* from the $\mathcal{EL}$ and *DL-Lite* families [AKL$^+$07, AKRZ14]. The results exhibit a stark contrast between decidable *DL-Lite* based and undecidable $\mathcal{EL}$-based LTDLs. Undecidability in the $\mathcal{EL}$ case is explained by an interaction of *non-convexity* (the ability to simulate disjunctions, here using only

LTL-operators) with the known fact that $\mathcal{EL}$ with disjunctions can encode $\mathcal{ALC}$; hence these $\mathcal{EL}$-based LTDLs are as hard as their $\mathcal{ALC}$-counterpart.

From the viewpoint of some DL applications, it has been argued that representing the existence of different *possible futures*, represented by a branching-time structure, is necessary for a more appropriate modeling [GJL12]: in the medical domain, symptoms or diseases may evolve in different ways in the future. As an example, consider the statement: "any patient with diabetes will *possibly* develop glaucoma in the future". In an LTDL it can be modeled as

$$\mathsf{Patient} \sqcap \exists \mathsf{hasDisease}.\mathsf{Diabetes} \sqsubseteq \Diamond \exists \mathsf{develops}.\mathsf{Glaucoma},$$

which is too strong – it says that each diabetes patient will *eventually* develop glaucoma. In branching-time TDLs we can use the existential path quantifier **E** together with $\Diamond$ to state more carefully that there exists a *possible future* where the patient develops glaucoma.

In contrast to LTDLs, the study of *branching time TDLs (BTDLs)* has been limited, mainly focusing on decidability boundaries obtained in the context of first-order branching temporal logic [HWZ02, BHWZ04]. Only recently were tight elementary bounds presented for BTDLs based on $\mathcal{ALC}$ and $\mathcal{EL}$ in the case where only *(local)* (i.e., non-rigid) roles are allowed [GJL12].

An aspect which, to our knowledge, has received no attention in the context of TDLs is the restriction of TBoxes to classical or acyclic TBoxes. This is surprising because this restriction is a fairly obvious design choice: given two observations: (a) it is likely to "ease" the prohibitive computational behavior in the presence of general TBoxes of LTDLs (and, as we will show, of BTDLs), and (b) TDLs designed this way seem well-suited as modeling languages for applications – for example, in the biomedical domain, given that large parts of SNOMED CT and the Gene Ontology GO [Gen00] are acyclic $\mathcal{EL}$-TBoxes.

The work reported in this chapter thus aims at answering the following two questions.

1. Do BTDLs with rigid roles behave computationally better than LTDLs? Since the standard way to express a disjunction using LTL operators fails for CTL(*), there is justifiable hope for convex $\mathcal{EL}$-based BTDLs, which can reasonably be expected to have good computational properties.
2. Does the restriction to acyclic TBoxes improve the computational properties of TDLs?

To answer these questions, we will study the decidability and complexity of BTDLs obtained from combining CTL with $\mathcal{ALC}$ and lightweight DLs from the $\mathcal{EL}$ and *DL-Lite* families, with rigid roles, with temporal operators restricted to be applied to concepts only, and with general as well as acyclic TBoxes. We will analyze the decidability and complexity of the standard reasoning problems.

Bibliographic notes. Our results for general TBoxes appeared in [GJS14]; those for restricted TBoxes were published in [GJS15].

4.2 Preliminaries

We introduce $\mathrm{CTL}_{\mathcal{ALC}}$. Let $\mathsf{N_C}$ and $\mathsf{N_R}$ be countably infinite sets of *concept names* and *role names*. We assume that $\mathsf{N_R}$ is partitioned into two countably infinite sets $\mathsf{N_{rig}}$ and $\mathsf{N_{loc}}$ of *rigid role names* and *local role names*, respectively. $\mathrm{CTL}_{\mathcal{ALC}}$*-concepts* C are defined by the following grammar

$$C := \top \mid A \mid \neg C \mid C \sqcap D \mid \exists r.C \mid \mathbf{E}\bigcirc C \mid \mathbf{E}\Box C \mid \mathbf{E}(C \,\mathcal{U}\, D)$$

where A ranges over $\mathsf{N_C}$, r over $\mathsf{N_C}$. The operators $\bot, \sqcup, \forall, \mathbf{E}\Diamond, \mathbf{A}\Box, \mathbf{A}\Diamond, \mathbf{A}\mathcal{U}$ can be defined in terms of the above in the usual way, see also [CGP99]. We moreover consider the *strict* ($\cdot^<$) versions of the temporal operators: $\mathbf{E}\Box^< C = \mathbf{E}\bigcirc\mathbf{E}\Box C$, $\mathbf{E}(C\mathcal{U}^< D) = \mathbf{E}\bigcirc\mathbf{E}(C\mathcal{U}D)$.

The semantics of CTL$_{\mathcal{ALC}}$ is given in terms of temporal interpretations, which are infinite trees whose every node is associated with a classical DL interpretation. Here a *tree* is a directed graph $T = (W, E)$ where $W \subseteq (\mathbb{N} \setminus \{0\})^*$ is a prefix-closed nonempty set of *nodes* and $E = \{(w, wc) \mid wc \in W, w \in \mathbb{N}^*, c \in \mathbb{N}\}$ a set of *edges*; we generally assume that $wc \in W$ and $c' < c$ implies $wc' \in W$ and that E is a total relation. The node $\varepsilon \in W$ is the *root* of T. In the context of temporal DLs we refer to nodes of T as *time points or worlds*.

A *temporal interpretation* is a structure $\mathfrak{I} = (\Delta, T, \{\mathcal{I}_w\}_{w\in W})$ where $T = (W, E)$ is a tree and, for each $w \in W$, $\mathcal{I}_w$ is an interpretation with domain Δ such that $r^{\mathcal{I}_w} = r^{\mathcal{I}_{w'}}$ for all $r \in \mathsf{N_{rig}}$ and $w, w' \in W$. We usually write $A^{\mathfrak{I},w}$ instead of $A^{\mathcal{I}_w}$, and intuitively $d \in A^{\mathfrak{I},w}$ means that, in the interpretation $\mathfrak{I}$, the object d is an instance of the concept name A *at time point* w. Moreover, we make the *constant domain assumption*, that is, all time points share the same domain Δ. For Boolean-complete TDLs, CDA is more general than assuming expanding, decreasing and varying domains because those can all be reduced to CDA [GKWZ03, Prop. 3.32]. For the sub-Boolean logics studied here, CDA is not w.l.o.g. Indeed, we will identify a logic where reasoning with expanding domains cannot be reduced to the constant domain case (unless $\mathsf{P} = \mathsf{PSPACE}$).

We now define the semantics of CTL$_{\mathcal{ALC}}$-concepts. A *path* in a tree $T = (W, E)$ starting at a node w is a minimal set $\pi \subseteq W$ such that $w \in \pi$ and, for each $w' \in \pi$, there is a $c \in \mathbb{N}$ with $w'c \in \pi$. We use $\mathsf{Paths}(w)$ to denote the set of all paths starting at the node w. For a path $\pi = w_0 w_1 w_1 \cdots$ and $i \geq 0$, we use $\pi[i]$ to denote w_i. The mapping $\cdot^{\mathfrak{I},w}$ is extended from concept names to CTL$_{\mathcal{ALC}}$-concepts as shown in Figure 9.

$\top^{\mathfrak{I},w}$	$= \Delta$
$(\neg C)^{\mathfrak{I},w}$	$= \Delta \setminus C^{\mathfrak{I},w}$
$(C \sqcap D)^{\mathfrak{I},w}$	$= C^{\mathfrak{I},w} \cap D^{\mathfrak{I},w}$
$(\exists r.C)^{\mathfrak{I},w}$	$= \{d \in \Delta \mid \exists e.(d, e) \in r^{\mathfrak{I},w} \wedge e \in C^{\mathfrak{I},w}\}$
$(\mathbf{E}\bigcirc C)^{\mathfrak{I},w}$	$= \{d \in \Delta \mid d \in C^{\mathfrak{I},\pi[1]}$ for some $\pi \in \mathsf{Paths}(w)\}$
$(\mathbf{E}\Box C)^{\mathfrak{I},w}$	$= \{d \in \Delta \mid \forall j \geq 0. d \in C^{\mathfrak{I},\pi[j]}$ for some $\pi \in \mathsf{Paths}(w)\}$
$(\mathbf{E}(C\mathcal{U}D))^{\mathfrak{I},w}$	$= \{d \in \Delta \mid \exists j \geq 0.(d \in D^{\mathfrak{I},\pi[j]} \wedge (\forall\, 0 \leq k < j.\, d \in C^{\mathfrak{I},\pi[k]}))$ for some $\pi \in \mathsf{Paths}(w)\}$

Figure 9: Satisfaction relation for CTL$_{\mathcal{ALC}}$-concepts

A *general CTL$_{\mathcal{ALC}}$-TBox* $\mathcal{T}$ is a finite set of *concept inclusions (CIs)* $C \sqsubseteq D$ with C, D CTL$_{\mathcal{ALC}}$ concepts. An *acyclic CTL$_{\mathcal{ALC}}$-TBox* $\mathcal{T}$ is a finite set of *concept definitions (CDs)* $A \equiv D$ with $A \in \mathsf{N_C}$ and D a CTL$_{\mathcal{ALC}}$ concept, such that (1) no two CDs have the same left-hand side, and (2) there are no CDs $A_1 \equiv C_1, \ldots, A_k \equiv C_k$ in $\mathcal{T}$ such that A_{i+1} occurs in C_i for $1 \leq i \leq k$, where $A_{k+1} = A_1$.

A temporal interpretation $\mathfrak{I}$ *satisfies* a concept C if $C^{\mathfrak{I},\varepsilon} \neq \emptyset$; it *satisfies* a CI $C \sqsubseteq D$, written $\mathfrak{I} \models C \sqsubseteq D$, if $C^{\mathfrak{I},w} \subseteq D^{\mathfrak{I},w}$ for all $w \in W$; it is a *model* of a TBox $\mathcal{T}$ if it satisfies all CIs in $\mathcal{T}$. Thus, a TBox $\mathcal{T}$ is interpreted globally in the sense that it has to be satisfied at *every* time point.

A concept C is *satisfiable with respect to a CTL$_{\mathcal{ALC}}$-TBox* $\mathcal{T}$ if there is a model of $\mathcal{T}$ that satisfies C. A concept D *subsumes* a concept C with respect to a TBox $\mathcal{T}$, written $\mathcal{T} \models C \sqsubseteq D$, if $\mathfrak{I} \models C \sqsubseteq D$ for all models $\mathfrak{I}$ of $\mathcal{T}$. For CTL$_{\mathcal{ALC}}$ and *DL-Lite*$^{\mathcal{N}}_{bool}$, we consider the *concept satisfiability problem*: given a concept C and a TBox $\mathcal{T}$, decide whether C is satisfiable with

Rigid roles?	no	yes	yes	yes
TBoxes	general	general	acyclic	empty
$\mathrm{CTL}_{\mathcal{ALC}}$	= ExpTime[1]	undecidable	nonelementary and decidable	
$\mathrm{CTL}_{\mathcal{EL}}^{\mathbf{E}\Diamond}$, $\mathrm{CTL}_{\mathcal{EL}}^{\mathbf{EO}}$	≤ P[1]	nonelem./undecid.	≤ P	≤ P
$\mathrm{CTL}_{\mathcal{EL}}^{\mathbf{EO},\mathbf{E}\Diamond}$	= ExpTime[1,2]	undecidable	≥ coNP, ≤ coNExpTime	= coNP
$\mathrm{CTL}_{\mathcal{EL}}^{\mathbf{E}\Diamond,\mathbf{A}\Box}$	= PSpace[1]	nonelementary	= PSpace	≤ PSpace

[1] [GJL12] [2] lower bound follows from *our* results ≥ hardness ≤ membership = completeness

Table 13: Overview of previous and new complexity results

respect to $\mathcal{T}$. For $\mathcal{EL}$, we consider the *subsumption problem*: given a TBox $\mathcal{T}$ and two concepts C and D, decide whether D subsumes C with respect to $\mathcal{T}$.

We will denote fragments of $\mathrm{CTL}_{\mathcal{ALC}}$ determined by restrictions to the temporal operators by putting the available temporal operators as a superscript; for example, $\mathrm{CTL}_{\mathcal{ALC}}^{\mathbf{E}\Diamond,\mathbf{A}\Box}$ denotes the fragment with the only temporal operators $\mathbf{E}\Diamond$ and $\mathbf{A}\Box$. Analogously, we use $\mathcal{EL}$ and $\textit{DL-Lite}_{bool}^{\mathcal{N}}$ as subscripts for TDLs based on these lightweight DLs.

4.3 Results

Our results for $\mathcal{ALC}$- and $\mathcal{EL}$-based TDLs are summarized – and put into the context of previous work – in Table 13.

General TBoxes. We start with the combination of CTL and $\mathcal{ALC}$, showing undecidability of concept satisfiability already for the fragment with only the operators $\mathbf{E}\Diamond$ and $\mathbf{A}\Box$. For this result, we combine results for products of "transitive" PMLs [GKWZ05] with a well-known DL-technique for encoding transitivity in a TBox [Tob01]. Our result is more general: it says that the global satisfiability problem for all PMLs determined by frame classes $\mathfrak{C}_1$ and $\mathfrak{C}_2$ is undecidable if $\mathfrak{C}_2$ contains only transitive frames and if $\mathfrak{C}_2$ and the class of all transitive frames in $\mathfrak{C}_1$ contain frames of arbitrarily large finite or infinite depth.

We continue by investigating the subsumption problem for BTDLs based on $\mathcal{EL}$ and several fragments of CTL. These CTL fragments are subsets of the CTL operators that were previously shown to lead to a decrease in complexity for the standard CTL satisfiability problem [MMTV09]. We first identify the following non-convex BTDLs.

$$\mathrm{CTL}_{\mathcal{EL}}^{\mathbf{E}\Diamond,\mathbf{A}\Diamond} \quad \mathrm{CTL}_{\mathcal{EL}}^{\mathbf{E}\Diamond\mathbf{E}\bigcirc} \quad \mathrm{CTL}_{\mathcal{EL}}^{\mathbf{E}\mathcal{U}} \quad \mathrm{CTL}_{\mathcal{EL}}^{\mathbf{E}\mathcal{U}^<} \quad \mathrm{CTL}_{\mathcal{EL}}^{\mathbf{E}\Diamond^<,\mathbf{E}\Box^<} \quad \mathrm{CTL}_{\mathcal{EL}}^{\mathbf{E}\Diamond,\mathbf{E}\Box}$$

We show their undecidability using the technique described above. Next we identify convex BTDLs based on $\mathcal{EL}$, showing closure under direct products of FO-models:

$$\mathrm{CTL}_{\mathcal{EL}}^{\mathbf{EO}} \quad \mathrm{CTL}_{\mathcal{EL}}^{\mathbf{E}\Diamond} \quad \mathrm{CTL}_{\mathcal{EL}}^{\mathbf{E}\Diamond,\mathbf{A}\Box}$$

It is reasonable to hope that these three BTDLs would behave computationally well, given that this is typically the case for convex logics, including two-dimensional ones: e.g, the combination of $\mathcal{EL}$ with the CTL operator $\mathbf{E}\Diamond$ in the case where only *local* roles are allowed [GJL12] is tractable, and various combinations of the modal logic S5 with $\mathcal{EL}$ are easier than the corresponding $\mathcal{ALC}$ variant [LS10, GJLS11]. Surprisingly, we show that the above three logics are undecidable or at

least nonelementary, see Table 13. The proofs of these results are challenging and technically involved because the absence of disjunction makes it difficult to use standard techniques for establishing lower complexity bounds, such as tilings. For the undecidability of $\mathrm{CTL}^{\mathbf{E}\mathsf{O}}_{\mathcal{EL}}$ we use a reduction from the halting problem of 2-counter automata introduced by Minsky [Min67]; for the nonelementary lower bound of $\mathrm{CTL}^{\mathbf{E}\Diamond}_{\mathcal{EL}}$ we show how to encode k-exponential counters, adopting Stockmeyer's technique [Sto74], and then reduce from the word problem of k-exponentially space-bounded Turing machines.

Finally, we investigate concept satisfiability for BTDLs based on $\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}$, showing that the technique developed in [AKRZ14] for LTDLs is robust in the sense that it can be carried over to *most* BTDLs based on CTL. This technique consists in reducing $\mathrm{CTL}_{\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}}$ to the one-variable fragment of first-order temporal logic, and then eliminating existential quantifiers to obtain an equivalent CTL formula. We thus obtain tight complexity bounds, namely PSPACE- and EXPTIME-completeness, according to the complexity for the corresponding CTL fragments [MMTV09]. Unfortunately, elimination of the quantifiers does not work for full $\mathrm{CTL}_{\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}}$ because the original shifting technique [AKRZ14, Lemma 4.2] relies on the past being unbounded, which is not the case for the standard CTL semantics. However, the problem can be circumvented by (1) disallowing local roles – which makes shifting superfluous – or (2) restricting the temporal operators to those that are tolerant to a certain variant of unraveling into the temporal direction, related to stutter invariance [Lam83]. We thus obtain:

Theorem 30. *Concept satisfiability relative to general TBoxes is* EXPTIME*-complete for* $\mathrm{CTL}_{\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}}$ *without local roles and for* $\mathrm{CTL}^{\mathbf{E}\mathcal{U},\mathbf{E}\Box}_{\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}}$*, and* PSPACE*-complete for* $\mathrm{CTL}^{\mathbf{E}\Diamond}_{\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}}$.

A more systematic investigation of $\mathrm{CTL}_{\mathit{DL\text{-}Lite}^{\mathcal{N}}_{bool}}$ is left for future work; we plan to pursue an automata-based approach in the style of [GJL12].

Restricted TBoxes. As above, we start with $\mathrm{CTL}_{\mathcal{ALC}}$, showing that satisfiability relative to empty and acyclic TBoxes is nonelementary and decidable. To establish the upper bound, we proceed in two steps analogously to Hodkinson et al. [HWZ00, HWZ02]: we characterize satisfiability by the existence of a quasimodel [GKWZ03] and encode the latter as a formula in monadic second-order logic, which is decidable over countably branching trees [Rab69]. Thanks to unfolding [Neb90b], there is no difference between the cases of acyclic TBoxes and the empty TBox. The nonelementary lower bound holds already for $\mathrm{CTL}^{\mathbf{E}\Diamond}_{\mathcal{ALC}}$ and $\mathrm{CTL}^{\mathbf{E}\mathsf{O}}_{\mathcal{ALC}}$: they are notational variants of the PMLs $\mathsf{S4} \times \mathsf{K}$ and $\mathsf{K} \times \mathsf{K}$, whose satisfiability problem is inherently nonelementary [GJL15].

We continue by replacing $\mathcal{ALC}$ with $\mathcal{EL}$ and maintaining the restriction to $\mathbf{E}\Diamond, \mathbf{E}\mathsf{O}$ and empty TBoxes. We show that the resulting TDLs are decidable in P with one of the two operators, and coNP-complete with both. The P upper bound is obtained by extending the known characterization of $\mathcal{EL}$ subsumption via homomorphisms between concept description trees [BKM99] to the two-dimensional case: given two $\mathrm{CTL}^{\mathbf{E}\mathsf{O}}_{\mathcal{EL}}$-concepts C, D, we have $C \sqsubseteq D$ if and only if D is satisfied in the root of the canonical model of C. While constructing the latter is straightforward, showing its canonicity is technically intricate. The same technique goes through with minor modifications for $\mathrm{CTL}^{\mathbf{E}\Diamond}_{\mathcal{EL}}$. The coNP upper bound for $\mathrm{CTL}^{\mathbf{E}\mathsf{O},\mathbf{E}\Diamond}_{\mathcal{EL}}$ is obtained by combining the previous machinery with expansion vectors, which were introduced by Haase and Lutz [HL08] for proving coNP membership of $\mathcal{EL}^{\mathsf{trans}}$, i.e., $\mathcal{EL}$ extended with transitive roles. The matching lower bound is due to a straightforward reduction from $\mathcal{EL}^{\mathsf{trans}}$.

Our main results are the P- and PSPACE-completeness for $\text{CTL}_{\mathcal{EL}}$ fragments over acyclic TBoxes: we lift the P upper bound for $\text{CTL}_{\mathcal{EL}}^{\mathbf{EO}}$ and $\text{CTL}_{\mathcal{EL}}^{\mathbf{E}\Diamond}$ to the case of acyclic TBoxes and show that the addition of $\mathbf{A}\Box$ leads to PSPACE-completeness. For the P upper bound, we use a two-dimensional variant of the standard $\mathcal{EL}$ completion algorithm [BBL05], which relies on the input TBox being acyclic. For the PSPACE upper bound, a suitable extension of the same procedure would require an exponentially-sized data structure. To avoid its full computation, our PSPACE procedure computes only single paths, explores each such *trace* in a tableau-like fashion, and enumerates all traces in an appropriate order. Here, acyclicity is crucial for bounding the length of traces polynomially. The matching PSPACE lower bound is via a reduction from QBF validity. By a slight variation of that reduction, we are able to show that $\text{CTL}_{\mathcal{EL}}^{\mathbf{EO}}$ and $\text{CTL}_{\mathcal{EL}}^{\mathbf{E}\Diamond}$ with rigid roles, acyclic TBoxes, and *expanding domains* are already PSPACE-hard.

As for $\text{CTL}_{\mathcal{EL}}^{\mathbf{EO},\mathbf{E}\Diamond}$ over acyclic TBoxes, unfolding immediately yields a coNEXPTIME upper bound. This time we cannot obtain a lower bound via a reduction from $\mathcal{EL}$ with transitive roles (which is PSPACE-hard over acyclic TBoxes [HL08]) because Haase and Lutz' hardness proof uses two roles, which cannot both be modeled by the temporal dimension. It remains for future work to search for tight complexity bounds; we are currently trying to combine the general idea behind expansion vectors with our completion procedure.

All upper bounds are unaffected if rigid concepts are included.

4.4 Discussion

With our results, we have made progress towards understanding the landscape of the computational complexity and expressivity of BTDLs allowing for rigid roles. We identified $\mathcal{EL}$-based BTDLs that are convex but nevertheless computationally badly behaved over general TBoxes.

By restricting the TBoxes, we have obtained a relatively low complexity for the same BTDLs, which is in stark contrast with the negative results for general TBoxes. The restriction to acyclic TBoxes has thus enabled us to identify the first computationally well-behaved TDLs with rigid roles based on $\mathcal{EL}$ and classical temporal logics.

Our positive results for $\text{CTL}_{\mathcal{EL}}$ over empty TBoxes are of significance for PMLs too: they entail that implication for the positive fragments of $\mathsf{K} \times \mathsf{K}$ and $\mathsf{S4} \times \mathsf{K}$ is in P.

The picture of the complexity landscape of TDLs with rigid roles obtained thus far can be extended in several ways. First, over general TBoxes, decidability could be regained by certain modifications, such as considering expanding domains (which might cause the complexity to drop over general TBoxes), or by replacing *DL-Lite*$_{bool}$ with *DL-Lite*$_{core}$ (which was successful for LTDLs [AKRZ14]). Second, the positive results obtained over acyclic TBoxes encourage attempts to enrich the language, e.g., by adding role inclusions, by allowing *temporal roles*, by studying cyclic TBoxes, or by changing the temporal component to LTL or even the μ-calculus.

Bibliography

[AB09] S. Arora and B. Barak. *Computational Complexity – A Modern Approach*. Cambridge University Press, 2009.

[ABI+09] E. Allender, M. Bauland, N. Immerman, H. Schnoor, and H. Vollmer. The complexity of satisfiability problems: Refining Schaefer's theorem. *J. of Computer and System Sciences*, 75(4):245–254, 2009.

[ABM99] C. Areces, P. Blackburn, and M. Marx. A road-map on complexity for hybrid logics. In *Proc. of CSL-99*, volume 1683 of *LNCS*, pages 307–321. Springer-Verlag, 1999.

[ABM00] C. Areces, P. Blackburn, and M. Marx. The computational complexity of hybrid temporal logics. *Logic J. of the IGPL*, 8(5):653–679, 2000.

[ACD90] R. Alur, C. Courcoubetis, and D. L. Dill. Model-checking for real-time systems. In *Proc. of LICS-90*, pages 414–425. IEEE Computer Society, 1990.

[ACH12] A. Armas Romero, B. Cuenca Grau, and I. Horrocks. MORe: Modular combination of OWL reasoners for ontology classification. In *Proc. of ISWC-12 (1)*, volume 7649 of *LNCS*, pages 1–16. Springer-Verlag, 2012.

[ACKZ07] A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. DL-Lite in the light of first-order logic. In *Proc. of AAAI-07*, pages 361–366, 2007.

[ACKZ09a] A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. Adding weight to DL-Lite. In *Proc. of DL 2009*, CEUR-WS, 2009.

[ACKZ09b] A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. The DL-Lite family and relations. *J. of Artificial Intelligence Research*, 36:1–69, 2009.

[AF00] A. Artale and E. Franconi. A survey of temporal extensions of description logics. *Ann. of Mathematics and Artificial Intelligence*, 30(1-4):171–210, 2000.

[AKL+07] A. Artale, R. Kontchakov, C. Lutz, F. Wolter, and M. Zakharyaschev. Temporalising tractable description logics. In *Proc. of TIME-07*, pages 11–22. IEEE Computer Society Press, 2007.

[AKRZ14] A. Artale, R. Kontchakov, V. Ryzhikov, and M. Zakharyaschev. A cookbook for temporal conceptual data modelling with description logics. *ACM Trans. Computational Logic*, 15(3):25, 2014.

[At07] C. Areces and B. ten Cate. Hybrid logics. In *Handbook of Modal Logic*, volume 3, chapter 14, pages 821–868. Elsevier, 2007.

[Baa96] F. Baader. Using automata theory for characterizing the semantics of terminological cycles. *Ann. of Mathematics and Artificial Intelligence*, 18(2-4):175–219, 1996.

[Baa03] F. Baader. Terminological cycles in a description logic with existential restrictions. In *Proc. of IJCAI-03*, pages 325–330. Morgan Kaufmann, 2003.

[BBC+10] M. Bauland, E. Böhler, N. Creignou, S. Reith, H. Schnoor, and H. Vollmer. The complexity of problems for quantified constraints. *Theory of Computing Systems*, 47(2):454–490, 2010.

[BBL05] F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proc. of IJCAI-05*, pages 364–369, 2005.

[BBL08] F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope further. In *Proc. of OWLED-08DC*, 2008.

[BCC+04] M. Bauland, P. Chapdelaine, N. Creignou, M. Hermann, and H. Vollmer. An algebraic approach to the complexity of generalized conjunctive queries. In *Proc. of SAT-04*, 2004.

[BCM+03] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[BCRV03] E. Böhler, N. Creignou, S. Reith, and H. Vollmer. Playing with Boolean blocks, part I: Post's lattice with applications to complexity theory. *SIGACT News*, 34(4):38–52, 2003.

[BDL04] G. Behrmann, A. David, and K. G. Larsen. A tutorial on Uppaal. In *Revised Lectures of SFM-RT '04*, volume 3185 of *LNCS*, pages 200–236. Springer-Verlag, 2004.

[BFZE08] C. Bezerra, F. Freitas, A. Zimmermann, and J. Euzenat. ModOnto: A tool for modularizing ontologies. In *Proc. of WONTO-08*, volume 427 of *CEUR-WS.org*, 2008.

[BGL08] F. Baader, S. Ghilardi, and C. Lutz. LTL over description logic axioms. In *Proc. of KR-08*, pages 684–694. AAAI Press, 2008.

[BH91] F. Baader and B. Hollunder. A terminological knowledge representation system with complete inference algorithms. In *Proc. of PDK-91*, volume 567 of *LNCS*, pages 67–86. Springer-Verlag, 1991.

[BH09] M. Bauland and E. Hemaspaandra. Isomorphic implication. *Theory of Computing Systems*, 44(1):117–139, 2009.

[BHSS06] M. Bauland, E. Hemaspaandra, H. Schnoor, and I. Schnoor. Generalized modal satisfiability. In *Proc. of STACS-06*, volume 3884 of *LNCS*, pages 500–511. Springer-Verlag, 2006.

[BHWZ04] S. Bauer, I. Hodkinson, F. Wolter, and M. Zakharyaschev. On non-local propositional and weak monodic quantified CTL. *J. of Logic and Computation*, 14(1):3–22, 2004.

[BK08] C. Baier and J.-P. Katoen. *Principles of Model Checking*. The MIT Press, 2008.

[BKM99] F. Baader, R. Küsters, and R. Molitor. Computing least common subsumers in description logics with existential restrictions. In *Proc. of IJCAI-99*. Morgan Kaufmann, 1999.

[BL10] L. Bozzelli and R. Lanotte. Complexity and succinctness issues for linear-time hybrid logics. *Theoretical Computer Science*, 411(2):454–469, 2010.

[BLS06] F. Baader, C. Lutz, and B. Suntisrivaraporn. CEL – a polynomial-time reasoner for life science ontologies. In *Proc. of IJCAR-06*, volume 4130 of *LNCS*, pages 287–291. Springer-Verlag, 2006.

[BMS+09] 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. In *Proc. of M4M-5*, volume 231 of *ENTCS*, pages 277–292, 2009.

[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 Trans. Computational Logic*, 12(2):13, 2011.

[BMTV12] O. Beyersdorff, A. Meier, M. Thomas, and H. Vollmer. The complexity of reasoning for fragments of default logic. *J. of Logic and Computation*, 22(3):587–604, 2012.

[BP90] P. Byers and D. Pitt. Conservative extensions: A cautionary note. *EATCS-Bulletin*, 41, 1990.

[Bra04] S. Brandt. Polynomial time reasoning in a description logic with existential restrictions, GCI axioms, and—what else? In *Proc. of ECAI-04*, pages 298–302. IOS Press, 2004.

[BS85] R. J. Brachman and J. G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216, 1985.

[BS95] P. Blackburn and J. Seligman. Hybrid languages. *J. of Logic, Language and Information*, 4(3):251–272, 1995.

[BSS+09] M. Bauland, T. Schneider, H. Schnoor, I. Schnoor, and H. Vollmer. The complexity of generalized satisfiability for linear temporal logic. *Logical Methods in Computer Science*, 5(1), 2009.

[Bul70] R. Bull. An approach to tense logic. *Theoria*, 36:282–300, 1970.

[BVSH09] J. Bao, G. Voutsadakis, G. Slutzki, and V. Honavar. Package-based description logics. In Stuckenschmidt et al. [SPS09], pages 349–371.

[Cal96] D. Calvanese. Reasoning with inclusion axioms in description logics: Algorithms and complexity. In *Proc. of ECAI-96*, pages 303–307. John Wiley & Sons, 1996.

[CCGR00] A. Cimatti, E. M. Clarke, F. Giunchiglia, and M. Roveri. NuSMV: a new symbolic model checker. *Int. J. on Software Tools for Technology Transfer*, 2(4):410–425, 2000.

[CDL+05] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. DL-Lite: Tractable description logics for ontologies. In *Proc. of AAAI-05*, pages 602–607. AAAI Press / The MIT Press, 2005.

[CDLN01] D. Calvanese, G. De Giacomo, M. Lenzerini, and D. Nardi. Reasoning in expressive description logics. In J. A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, pages 1581–1634. Elsevier and MIT Press, 2001.

[CGP99] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. MIT Press, 1999.

[CHKS08] B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Modular reuse of ontologies: Theory and practice. *J. of Artificial Intelligence Research*, 31:273–318, 2008.

[CHKS10] B. Cuenca Grau, C. Halaschek-Wiener, Y. Kazakov, and B. Suntisrivaraporn. Incremental classification of description logics ontologies. *J. of Automated Reasoning*, 44(4):337–369, 2010.

[CHS07] P. Chapdelaine, M. Hermann, and I. Schnoor. Complexity of default logic on generalized conjunctive queries. In *Proc. of LPNMR-07*, volume 4483 of *LNCS*, pages 58–70. Springer-Verlag, 2007.

[CL93] C.-C. Chen and I-P. Lin. The computational complexity of satisfiability of temporal Horn formulas in propositional linear-time temporal logic. *Information Processing Letters*, 45(3):131–136, 1993.

[CMVT12] N. Creignou, A. Meier, H. Vollmer, and M. Thomas. The complexity of reasoning for fragments of autoepistemic logic. *ACM Trans. Computational Logic*, 13(2), 2012.

[CPS09] B. Cuenca Grau, B. Parsia, and E. Sirin. Ontology integration using $\mathcal{E}$-connections. In Stuckenschmidt et al. [SPS09], pages 293–320.

[CPSK06] B. Cuenca Grau, B. Parsia, E. Sirin, and A. Kalyanpur. Modularity and web ontologies. In *Proc. of KR-06*, pages 198–209, 2006.

[CST12] N. Creignou, J. Schmidt, and M. Thomas. Complexity classifications for propositional abduction in Post's framework. *J. of Logic and Computation*, 22(5):1145–1170, 2012.

[Dal00] V. Dalmau. *Computational Complexity of Problems over Generalized Formulas*. PhD thesis, Universitat Politécnica de Catalunya, 2000.

[Del11] C. Del Vescovo. The modular structure of an ontology: Atomic decomposition towards applications. In *Proc. of DL 2011*, volume 745 of *CEUR-WS.org*, 2011.

[DFR00] C. Dixon, M. Fisher, and M. Reynolds. Execution and proof in a Horn-clause temporal logic. In H. Barringer, M. Fisher, D. Gabbay, and G. Gough, editors, *Advances in Temporal Logic*, volume 16 of *Applied Logic Series*, pages 413–433. Kluwer, 2000.

[DGK+11] C. Del Vescovo, D. Gessler, P. Klinov, B. Parsia, U. Sattler, T. Schneider, and A. Winget. Decomposition and modular structure of BioPortal ontologies. In *Proc. of ISWC-11*, volume 7031 of *LNCS*, pages 130–145. Springer-Verlag, 2011.

[DGS93] R. Diaconescu, J. Goguen, and P. Stefaneas. Logical support for modularisation. In G. Huet and G. Plotkin, editors, *Logical Environments*. Cambridge University Press, 1993.

[DKP+13] C. Del Vescovo, P. Klinov, B. Parsia, U. Sattler, T. Schneider, and D. Tsarkov. Empirical study of logic-based modules: Cheap is cheerful. In *Proc. of ISWC-13 (1)*, volume 8218 of *LNCS*, pages 84–100. Springer-Verlag, 2013.

[DL94] G. De Giacomo and M. Lenzerini. Boosting the correspondence between description logics and propositional dynamic logics. In *Proc. of AAAI-94*, pages 205–212. AAAI Press, 1994.

[DLN+92] F. M. Donini, M. Lenzerini, D. Nardi, B. Hollunder, W. Nutt, and A. Marchetti-Spaccamela. The complexity of existential quantification in concept languages. *Artificial Intelligence*, 53(2-3):309–327, 1992.

[DLNN97] F. M. Donini, M. Lenzerini, D. Nardi, and W. Nutt. The complexity of concept languages. *Information and Computation*, 134(1):1–58, 1997.

[DM00] F. M. Donini and F. Massacci. EXPTIME tableaux for $\mathcal{ALC}$. *Artificial Intelligence*, 124(1):87–138, 2000.

[Don03] F. M. Donini. Complexity of reasoning. In Baader et al. [BCM+03], pages 96–136.

[DPS11] C. Del Vescovo, B. Parsia, and U. Sattler. Topicality in logic-based ontologies. In *Proc. of ICCS-11*, volume 6828 of *LNCS*, pages 187–200. Springer-Verlag, 2011.

[DPS12] C. Del Vescovo, B. Parsia, and U. Sattler. Logical relevance in ontologies. In *Proc. of DL 2012*, volume 846 of *CEUR-WS.org*, 2012.

[DPSS10] C. Del Vescovo, B. Parsia, U. Sattler, and T. Schneider. The modular structure of an ontology: an empirical study. In *Proc. of DL 2010*, volume 573 of *CEUR-WS.org*, 2010.

[DPSS11] C. Del Vescovo, B. Parsia, U. Sattler, and T. Schneider. The modular structure of an ontology: Atomic decomposition. In *Proc. of IJCAI-11*, pages 2232–2237, 2011.

[DS02] S. Demri and P. Schnoebelen. The complexity of propositional linear temporal logics in simple cases. *Information and Computation*, 174(1):84–103, 2002.

[EC82] E. A. Emerson and E. M. Clarke. Using branching time temporal logic to synthesize synchronization skeletons. *Science of Computer Programming*, 2:241–266, 1982.

[EES90] E. A. Emerson, M. Evangelist, and J. Srinivasan. On the limits of efficient temporal decidability. In *Proc. of LICS-90*, pages 464–475. IEEE Computer Society Press, 1990.

[EH86] E. A. Emerson and J. Y. Halpern. "Sometimes" and "not never" revisited: on branching versus linear time temporal logic. *J. of the ACM*, 33(1):151–178, 1986.

[Fd06] M. Franceschet and M. de Rijke. Model checking for hybrid logics (with an application to semistructured data). *J. of Applied Logic*, 4(3):279–304, 2006.

[FdS03] M. Franceschet, M. de Rijke, and B.-H. Schlingloff. Hybrid logics on linear structures: Expressivity and complexity. In *Proc. of TIME-03*, pages 166–173. IEEE Computer Society Press, 2003.

[FR79] J. Ferrante and C. W. Rackoff. *The Computational Complexity of Logical Theories*. Springer-Verlag, 1979.

[FT11] E. Franconi and D. Toman. Fixpoints in temporal description logics. In *Proc. of IJCAI-11*, pages 875–880. AAAI Press, 2011.

[Gen00] The Gene Ontology Consortium. Gene ontology: Tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

[GFH+03] J. Golbeck, G. Fragoso, F. Hartel, J. Hendler, J. Oberthaler, and B. Parsia. The National Cancer Institute's thesaurus and ontology. *J. of Web Semantics*, 1(1):75–80, 2003.

[GJL12] V. Gutiérrez-Basulto, J. C. Jung, and C. Lutz. Complexity of branching temporal description logics. In *Proc. of ECAI-12*, volume 242 of *Frontiers in AI and Appl.*, pages 390–395. IOS Press, 2012.

[GJL15] S. Göller, J. C. Jung, and M. Lohrey. The complexity of decomposing modal and first-order theories. *ACM Trans. Computational Logic*, 16(1):9:1–9:43, 2015.

[GJLS11] V. Gutiérrez-Basulto, J. C. Jung, C. Lutz, and L. Schröder. A closer look at the probabilistic description logic Prob-$\mathcal{EL}$. In *Proc. of AAAI-11*. AAAI Press, 2011.

[GJS14] V. Gutiérrez-Basulto, J. C. Jung, and T. Schneider. Lightweight description logics and branching time: a troublesome marriage. In *Proc. of KR-14*. AAAI Press, 2014.

[GJS15] V. Gutiérrez-Basulto, J. C. Jung, and T. Schneider. Lightweight temporal description logics with rigid roles and restricted TBoxes. In *Proc. of IJCAI-15*, pages 3015–3021. AAAI Press, 2015.

[GKS10] D. Götzmann, M. Kaminski, and G. Smolka. Spartacus: A tableau prover for hybrid logic. *ENTCS*, 262:127–139, 2010.

[GKWZ03] D. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyaschev. *Many-dimensional modal logics: theory and applications*, volume 148 of *Studies in Logic*. Elsevier, 2003.

[GKWZ05] David Gabelaia, Agi Kurucz, Frank Wolter, and Michael Zakharyaschev. Products of 'transitive' modal logics. *J. of Symbolic Logic*, 70(3):993–1021, 2005.

[GLW06] S. Ghilardi, C. Lutz, and F. Wolter. Did I damage my ontology? A case for conservative extensions in description logics. In *Proc. of KR-06*, pages 187–197, 2006.

[GMM+12] S. Göller, A. Meier, M. Mundhenk, T. Schneider, M. Thomas, and F. Weiß. The complexity of monotone hybrid logics over linear frames and the natural numbers. In *Proc. of AiML-9*, pages 261–278. College Publications, 2012.

[GMWK02] R. Givan, D. A. McAllester, C. Witty, and D. Kozen. Tarskian set constraints. *Information and Computation*, 174(2):105–131, 2002.

[GSM+09] D. Gessler, G. S. Schiltz, G. D. May, S. Avraham, C. D. Town, D. M. Grant, and R. T. Nelson. SSWAP: A simple semantic web architecture and protocol for semantic web services. *BMC Bioinformatics*, 10, 2009.

[HA09] G. Hoffmann and C. Areces. HTab: a terminating tableaux system for hybrid logic. *ENTCS*, 231:3–19, 2009.

[Hag08] M. Hagen. *Algorithmic and Computational Complexity Issues of* MONET. PhD thesis, University of Jena, 2008.

[Hal95] J. Y. Halpern. The effect of bounding the number of primitive propositions and the depth of nesting on the complexity of modal logic. *Artificial Intelligence*, 75(2):361–372, 1995.

[HB91] B. Hollunder and F. Baader. Qualifying number restrictions in concept languages. In *Proc. of KR-91*, pages 335–346. Morgan Kaufmann, 1991.

[Hem01] E. Hemaspaandra. The complexity of Poor Man's Logic. *J. of Logic and Computation*, 11(4):609–622, 2001.

[Hin62] J. Hintikka. *Knowledge and Belief.* Cornell University Press, 1962.

[HKS06a] I. Horrocks, O. Kutz, and U. Sattler. The even more irresistible $\mathcal{SROIQ}$. In *Proc. of KR-06*, pages 57–67. AAAI Press, 2006.

[HKS06b] I. Horrocks, O. Kutz, and U. Sattler. The irresistible $\mathcal{SRIQ}$. In *Proc. of OWLED-05*, volume 188 of *CEUR-WS.org*, 2006.

[HL08] C. Haase and C. Lutz. Complexity of subsumption in the $\mathcal{EL}$ family of description logics: Acyclic and cyclic TBoxes. In *Proc. of ECAI-08*, 2008.

[HM01] V. Haarslev and R. Möller. RACER system description. In *Proc. of IJCAR-01*, volume 2083 of *LNAI*. Springer-Verlag, 2001.

[Hof05] Martin Hofmann. Proof-theoretic approach to description-logic. In *Proc. of LICS-05*, pages 229–237. IEEE Computer Society, 2005.

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

[Hol04] G. J. Holzmann. *The SPIN Model Checker – primer and reference manual*. Addison-Wesley, 2004.

[HPS08] M. Horridge, B. Parsia, and U. Sattler. Laconic and precise justifications in OWL. In *Proc. of ISWC-08*, volume 5318 of *LNCS*, pages 323–338. Springer-Verlag, 2008.

[HPSv03] I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. *J. of Web Semantics*, 1(1):7–26, 2003.

[HS08] E. Hemaspaandra and H. Schnoor. On the complexity of elementary modal logics. In *Proc. of STACS-08*, volume 1 of *LIPIcs*, pages 349–360. Schloss Dagstuhl, 2008.

[HSS10] E. Hemaspaandra, H. Schnoor, and I. Schnoor. Generalized modal satisfiability. *J. of Computer and System Sciences*, 76(7):561–578, 2010.

[HWZ00] I. Hodkinson, F. Wolter, and M. Zakharyaschev. Decidable fragments of first-order temporal logics. *Ann. of Pure and Applied Logic*, 106:85–134, 2000.

[HWZ02] I. Hodkinson, F. Wolter, and M. Zakharyaschev. Decidable and undecidable fragments of first-order branching temporal logics. In *Proc. of LICS-02*, pages 393–402. IEEE Computer Society Press, 2002.

[JCS+08] E. Jiménez-Ruiz, B. Cuenca Grau, U. Sattler, T. Schneider, and R. Berlanga Llavori. Safe and economic re-use of ontologies: A logic-based methodology and tool support. In *Proc. of ESWC-08*, volume 5021 of *LNCS*, pages 185–199. Springer-Verlag, 2008.

[JJBR08] A. Jimeno, E. Jiménez-Ruiz, R. Berlanga, and D. Rebholz-Schuhmann. Use of shared lexical resources for efficient ontological engineering. In *Proc. of SWAT4LS-08*, volume 435 of *CEUR-WS.org*, 2008.

[JT52a] B. Jónsson and A. Tarski. Boolean algebras with operators, Part I. *American J. of Mathematics*, 73:891–939, 1952.

[JT52b] B. Jónsson and A. Tarski. Boolean algebras with operators, Part II. *American J. of Mathematics*, 74:127–162, 1952.

[Kaz08] Y. Kazakov. $\mathcal{RIQ}$ and $\mathcal{SROIQ}$ are harder than $\mathcal{SHOIQ}$. In *Proc. of KR-08*, pages 274–284. AAAI Press, 2008.

[Kd03] Y. Kazakov and H. de Nivelle. Subsumption of concepts in $\mathcal{FL}_0$ for (cyclic) terminologies with respect to descriptive semantics is PSPACE-complete. In *DL-2003*, volume 81 of *CEUR-WS.org*, 2003.

[KDS12] P. Klinov, C. Del Vescovo, and T. Schneider. Incrementally updateable and persistent decomposition of OWL ontologies. In *Proc. of OWLED-12*, volume 849 of *CEUR-WS.org*, 2012.

[KKL+11] B. Konev, R. Kontchakov, M. Ludwig, T. Schneider, F. Wolter, and M. Zakharyaschev. Conjunctive query inseparability of OWL 2 QL TBoxes. In *Proc. of AAAI-11*. AAAI Press, 2011.

[KLPW10] B. Konev, C. Lutz, D. Ponomaryov, and F. Wolter. Decomposing description logic ontologies. In *Proc. of KR-10*. AAAI Press, 2010.

[KLWW08] B. Konev, C. Lutz, D. Walther, and F. Wolter. Semantic modularity and module extraction in description logics. In *Proc. of ECAI-08*, volume 178 of *Frontiers in AI and Appl.*, pages 55–59. IOS Press, 2008.

[KLWW09] B. Konev, C. Lutz, D. Walther, and F. Wolter. Formal properties of modularization. In Stuckenschmidt et al. [SPS09], pages 25–66.

[KNP04] M. Z. Kwiatkowska, G. Norman, and D. Parker. Probabilistic symbolic model checking with PRISM: a hybrid approach. *Int. J. on Software Tools for Technology Transfer*, 6(2):128–142, 2004.

[KPS+06] A. Kalyanpur, B. Parsia, E. Sirin, B. Cuenca Grau, and J. Hendler. Swoop: A Web ontology editing browser. *J. of Web Semantics*, 4(2):144–153, 2006.

[KPS+09] R. Kontchakov, L. Pulina, U. Sattler, T. Schneider, P. Selmer, F. Wolter, and M. Zakharyaschev. Minimal module extraction from DL-Lite ontologies using QBF solvers. In *Proc. of IJCAI-09*, pages 836–841, 2009.

[Kri59] S. Kripke. A completeness theorem in modal logic. *J. of Symbolic Logic*, 24(1):1–14, 1959.

[Kri63a] S. Kripke. Semantical analysis of modal logic I: Normal modal propositional calculi. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 6:67–96, 1963.

[Kri63b] S. Kripke. Semantical considerations on modal logic. *Acta Philosophica Fennica*, 16:83–94, 1963.

[KWZ10] R. Kontchakov, F. Wolter, and M. Zakharyaschev. Logic-based ontology comparison and module extraction, with an application to DL-Lite. *Artificial Intelligence*, 174(15):1093–1141, 2010.

[Lam83] Leslie Lamport. What good is temporal logic? In *Prof. of IFIP Congress*, pages 657–668, 1983.

[Lew18] C. I. Lewis. *A survey of symbolic logic*. University of California Press, 1918. Reprint, New York, 1960.

[Lew79] H. Lewis. Satisfiability problems for propositional calculi. *Mathematical Systems Theory*, 13:45–53, 1979.

[LL32] C. I. Lewis and C. H. Langford. *Symbolic logic*. Dover, 1932. Corrected reprint, New York, 1959.

[LS77] E. Lemmon and D. Scott. *An Introduction to Modal Logic*. Blackwell, 1977.

[LS10] C. Lutz and L. Schröder. Probabilistic description logics for subjective uncertainty. In *Proc. of KR-10*. AAAI Press, 2010.

[LW10] C. Lutz and F. Wolter. Deciding inseparability and conservative extensions in the description logic $\mathcal{EL}$. *J. of Symbolic Computation*, 45(2):194–228, 2010.

[LWW07] C. Lutz, D. Walther, and F. Wolter. Conservative extensions in expressive description logics. In *Proc. of IJCAI-07*, pages 453–458, 2007.

[LWZ08] C. Lutz, F. Wolter, and M. Zakharyaschev. Temporal description logics: A survey. In *Proc. of TIME-08*, pages 3–14. IEEE Computer Society Press, 2008.

[Mac91] R. MacGregor. The evolving technology of classification-based knowledge representation systems. In J. F. Sowa, editor, *Principles of Semantic Networks*, pages 385–400. Morgan Kaufmann, 1991.

[Mai97] T. Maibaum. Conservative extensions, interpretations between theories and all that! In *Proc. of CAAP/FASE-97*, volume 1214 of *LNCS*, pages 40–66. Springer-Verlag, 1997.

[Mar04] N. Markey. Past is for free: on the complexity of verifying linear temporal properties with past. *Acta Informatica*, 40(6–7):431–458, 2004.

[MDW91] E. Mays, R. Dionne, and R. Weida. K-REP system overview. *SIGART Bull.*, 2(3):93–97, 1991.

[Mei11] A. Meier. *On the Complexity of Modal Logic Variants and their Fragments*. PhD thesis, Leibniz University of Hannover, 2011.

[Min67] M. L. Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, Inc., 1st edition, 1967.

[MMS+10] A. Meier, M. Mundhenk, T. Schneider, M. Thomas, V. Weber, and F. Weiß. The complexity of satisfiability for fragments of hybrid logic – part I. *J. of Applied Logic*, 8(4):409–421, 2010.

[MMTV09] A. Meier, M. Mundhenk, M. Thomas, and H. Vollmer. The complexity of satisfiability for fragments of CTL and CTL*. *Int. J. Found. Comput. Sci.*, 20(5):901–918, 2009.

[Mos04] P. Mosses, editor. *CASL Reference Manual*, volume 2960 of *LNCS*. Springer-Verlag, 2004.

[MS09] M. Mundhenk and T. Schneider. The complexity of hybrid logics over equivalence relations. *J. of Logic, Language and Information*, 18(4):493–514, 2009.

[MS13] A. Meier and T. Schneider. Generalized satisfiability for the description logic ALC. *Theoretical Computer Science*, 505:55–73, 2013.

[MSH09] B. Motik, R. Shearer, and I. Horrocks. Hypertableau reasoning for description logics. *J. of Artificial Intelligence Research*, 36:165–228, 2009.

[MSSW10] M. Mundhenk, T. Schneider, T. Schwentick, and V. Weber. Complexity of hybrid logics over transitive frames. *J. of Applied Logic*, 8(4):422–440, 2010.

[MW05] A. Muscholl and I. Walukiewicz. An NP-complete fragment of LTL. *Int. J. Found. Comput. Sci.*, 16(4):743–753, 2005.

[Neb90a] B. Nebel. *Reasoning and Revision in Hybrid Representation Systems*, volume 422 of *LNCS*. Springer-Verlag, 1990.

[Neb90b] B. Nebel. Terminological reasoning is inherently intractable. *Artificial Intelligence*, 43(2):235–249, 1990.

[Nor05] G. Nordh. A trichotomy in the complexity of propositional circumscription. In *Proc. of LPAR 2004*, volume 3452 of *LNCS*, pages 257–269. Springer-Verlag, 2005.

[Pap94] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[Pel91] C. Peltason. The BACK system – an overview. *SIGART Bull.*, 2(3):114–119, 1991.

[Pip97] N. Pippenger. *Theories of Computability*. Cambridge University Press, 1997.

[Pnu77] A. Pnueli. The temporal logic of programs. In *Proc. of FOCS-77*, pages 46–67. IEEE Computer Society Press, 1977.

[Pos41] E. Post. The two-valued iterative systems of mathematical logic. *Annals of Mathematical Studies*, 5:1–122, 1941.

[Pra76] V. R. Pratt. Semantical considerations on Floyd-Hoare logic. In *Proc. of FOCS-76*, pages 109–121. IEEE Computer Society Press, 1976.

[Pra78] V. R. Pratt. A practical decision method for propositional dynamic logic: Preliminary report. In *Proc. of STOC-78*, pages 326–337. ACM, 1978.

[Pri57] A. N. Prior. *Time and Modality*. Oxford: University Press, 1957.

[Pri58] A. Prior. The syntax of time-distinctions. *Franciscan Studies*, 18:105–120, 1958.

[Pri67] A. N. Prior. *Past, Present and Future*. Oxford: Clarendon Press, 1967.

[Pri68] A. N. Prior. *Papers on Time and Tense*. Oxford: Clarendon Press, 1968.

[PS10] B. Parsia and T. Schneider. The modular structure of an ontology: An empirical study. In *Proc. of KR-10*. AAAI Press, 2010.

[Rab69] M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. of the American Math. Society*, 141:1–35, 1969.

[Rei01] S. Reith. *Generalized Satisfiability Problems*. PhD thesis, Universität Würzburg, 2001.

[RV03] S. Reith and H. Vollmer. Optimal satisfiability for propositional calculi and constraint satisfaction problems. *Information and Computation*, 186(1):1–19, 2003.

[RW00] S. Reith and K. W. Wagner. The complexity of problems defined by Boolean circuits. In *Proc. of Int. Conf. Mathematical Foundation of Informatics '99*, pages 25–28, 2000.

[SC85] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. *J. of the ACM*, 32(3):733–749, 1985.

[Sch91] K. Schild. A correspondence theory for terminological logics: Preliminary report. In *Proc. of IJCAI-91*, pages 466–471. Morgan Kaufmann, 1991.

[Sch93] K. Schild. Combining terminological logics with tense logic. In *Proc. of EPIA-93*, volume 727 of *LNCS*, pages 105–120. Springer-Verlag, 1993.

[Sch94] K. Schild. Terminological cycles and the propositional μ-calculus. In *Proc. of KR-94*, pages 509–520. Morgan Kaufmann, 1994.

[Sch07] H. Schnoor. *Algebraic Techniques for Satisfiability Problems*. PhD thesis, Universität Hannover, 2007.

[Sei09] J. Seidenberg. Web ontology segmentation: Extraction, transformation, evaluation. In Stuckenschmidt et al. [SPS09], pages 211–243.

[SK04] H. Stuckenschmidt and M. Klein. Structure-based partitioning of large concept hierarchies. In *Proc. of ISWC-04*, volume 3298 of *LNCS*, pages 289–303. Springer-Verlag, 2004.

[SL94] R. Segala and N. A. Lynch. Probabilistic simulations for probabilistic processes. In *Proc. of CONCUR-94*, volume 836 of *LNCS*, pages 481–496. Springer-Verlag, 1994.

[Spa00] K.A. Spackman. Managing clinical terminology hierarchies using algorithmic calculation of subsumption: Experience with SNOMED-RT. *J. of the Amer. Med. Informatics Ass.*, 2000.

[SPC+07] E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. *J. of Web Semantics*, 5(2):51–53, 2007.

[SPS09] H. Stuckenschmidt, C. Parent, and S. Spaccapietra, editors. *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, volume 5445 of *LNCS*. Springer-Verlag, 2009.

[SS07] H. Schnoor and I. Schnoor. Enumerating all solutions for constraint satisfaction problems. In *Proc. of STACS-07*, volume 4393 of *LNCS*, pages 694–705. Springer-Verlag, 2007.

[SS08] H. Schnoor and I. Schnoor. Partial polymorphisms and constraint satisfaction problems. In *Complexity of Constraints*, volume 5250 of *LNCS*, pages 229–254. Springer-Verlag, 2008.

[SSZ09] U. Sattler, T. Schneider, and M. Zakharyaschev. Which kind of module should I extract? In *Proc. of DL 2009*, volume 477 of *CEUR-WS.org*, 2009.

[ST09] L. Serafini and A. Tamilin. Composing modular ontologies with Distributed Description Logics. In Stuckenschmidt et al. [SPS09], pages 321–347.

[Sto74] L. J. Stockmeyer. *The complexity of decision problems in automata theory and logic*. PhD thesis, Massachusetts Institute of Technology (MIT), 1974.

[Sun08] B. Suntisrivaraporn. Module extraction and incremental classification: A pragmatic approach for $\mathcal{EL}^+$ ontologies. In *Proc. of ESWC-08*, volume 5021 of *LNCS*, pages 230–244. Springer-Verlag, 2008.

[SV01] U. Sattler and M. Y. Vardi. The hybrid μ-calculus. In *Proc. of IJCAR-01*, volume 2083 of *LNCS*, pages 76–91. Springer-Verlag, 2001.

[SW07] T. Schwentick and V. Weber. Bounded-variable fragments of hybrid logics. In *STACS-07*, volume 4393 of *LNCS*, pages 561–572. Springer-Verlag, 2007.

[tF05] B. ten Cate and M. Franceschet. On the complexity of hybrid logics with binders. In *CSL-05*, volume 3634 of *LNCS*, pages 339–354. Springer-Verlag, 2005.

[TH06] D. Tsarkov and I. Horrocks. FaCT++ description logic reasoner: System description. In *Proc. of IJCAR-06*, volume 4130 of *LNCS*, pages 292–297. Springer-Verlag, 2006.

[Tho12] M. Thomas. The complexity of circumscriptive inference in Post's lattice. *Theory of Computing Systems*, 50(3):401–419, 2012.

[TM87] W. Turski and T. Maibaum. *The Specification of Computer Programs*. Addison Wesley, 1987.

[Tob01] S. Tobies. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation*. PhD thesis, RWTH Aachen, 2001.

[TP12] D. Tsarkov and I. Palmisano. Divide et impera: Metareasoning for large ontologies. In *Proc. of OWLED-12*, volume 849 of *CEUR-WS.org*, 2012.

[Vol99] H. Vollmer. *Introduction to Circuit Complexity – a Uniform Approach*. Texts in Theoretical Computer Science. Springer-Verlag, 1999.

[VW86] M. Y. Vardi and P. Wolper. Automata-theoretic techniques for modal logics of programs. *J. of Computer and System Sciences*, 32(2):183–221, 1986.

[Web09] V. Weber. Branching-time logics repeatedly referring to states. *J. of Logic, Language and Information*, 18(4):593–624, 2009.

[Wol83] P. Wolper. Temporal logic can be more expressive. *Information and Control*, 56(1/2):72–99, 1983.

[WVM09] M. D. Wilkinson, B. Vandervalk, and L. McCarthy. SADI semantic web services – 'cause you can't always GET what you want! In *Proc. of APSCC-09*, pages 13–18, 2009.

[Yov97] S. Yovine. KRONOS: A verification tool for real-time systems. *Int. J. on Software Tools for Technology Transfer*, 1(1-2):123–133, 1997.

Appendix A

List of Submitted Papers

Papers are listed per chapter and in the order of their appearance in this thesis.

Chapter 2: Complexity of Sub-Boolean Fragments

[BSS+09] M. Bauland, T. Schneider, H. Schnoor, I. Schnoor, and H. Vollmer. The Complexity of Generalized Satisfiability for Linear Temporal Logic. *Logical Methods in Computer Science,* 5(1), 2009.

[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,* 12(2), 2011.

[MS13] A. Meier and T. Schneider. Generalized Satisfiability for the Description Logic $\mathcal{ALC}$. *Theoretical Computer Science,* 505:55–73, 2013.

[MMS+10] A. Meier, M. Mundhenk, T. Schneider, M. Thomas, V. Weber, and F. Weiß. The Complexity of Satisfiability for Fragments of Hybrid Logic – Part I. *Journal of Applied Logic,* 8, 409–421, 2010.

[GMM+12] S. Göller, A. Meier, M. Mundhenk, T. Schneider, M. Thomas, and F. Weiß. The Complexity of Monotone Hybrid Logics over Linear Frames and the Natural Numbers. In *Proc. of AiML-12*, College Publications, pg. 261–278, 2012.

Chapter 3: Module Extraction and Modularization

[JCS+08] E. Jiménez-Ruiz, B. Cuenca Grau, U. Sattler, T. Schneider, and R. Berlanga Llavori. Safe and Economic Re-Use of Ontologies: A Logic-Based Methodology and Tool Support. In *Proc. of ESWC-08,* volume 5021 of *LNCS*, pg. 185–199. Springer-Verlag, 2008. Nominated for Best Paper Award.

[SSZ09] U. Sattler, T. Schneider, and M. Zakharyaschev. Which Kind of Module Should I Extract? In *Proc. of DL 2009,* volume 477 of CEUR-WS.org, 2009.

[DKP+13] C. Del Vescovo, P. Klinov, B. Parsia, U. Sattler, T. Schneider, and D. Tsarkov. Empirical Study of Logic-Based Modules: Cheap Is Cheerful. In *Proc. of ISWC-13 (1)*, LNCS 8218, pg. 84–100. Springer-Verlag, 2013. Nominated for Best Paper Award.

[PS10] B. Parsia and T. Schneider. The Modular Structure of an Ontology: an Empirical Study. In *Proc. of KR-10*, pg. 584–586, 2010.

[DPSS11] C. Del Vescovo, B. Parsia, U. Sattler, and T. Schneider. The Modular Structure of an Ontology: Atomic Decomposition. In *Proc. of IJCAI-11*, pg. 2232–2237, 2011.

[DGK+11] C. Del Vescovo, D. Gessler, P. Klinov, B. Parsia, U. Sattler, T. Schneider, and A. Winget. Decomposition and Modular Structure of BioPortal Ontologies. In *Proc. of ISWC-11*, LNCS 7031, pg. 130–145, 2011.

Chapter 4: Branching-Time Temporal Description Logics

[GJS14] V. Gutiérrez Basulto, J. C. Jung, and T. Schneider. Lightweight Description Logics and Branching Time: a Troublesome Marriage. In *Proc. of KR-14*, AAAI Press, 2014.

[GJS15] V. Gutiérrez Basulto, J. C. Jung, and T. Schneider. Lightweight Temporal Description Logics with Rigid Roles and Restricted TBoxes. In *Proc. of IJCAI-15*, pg. 3015–3021, AAAI Press, 2015.

Appendix C

Illustrative Figures for Chapter 2

The following pages contain overviews of our results in the sections 2.3.2, 2.4.2, and 2.5.2, arranged in Post's lattice.

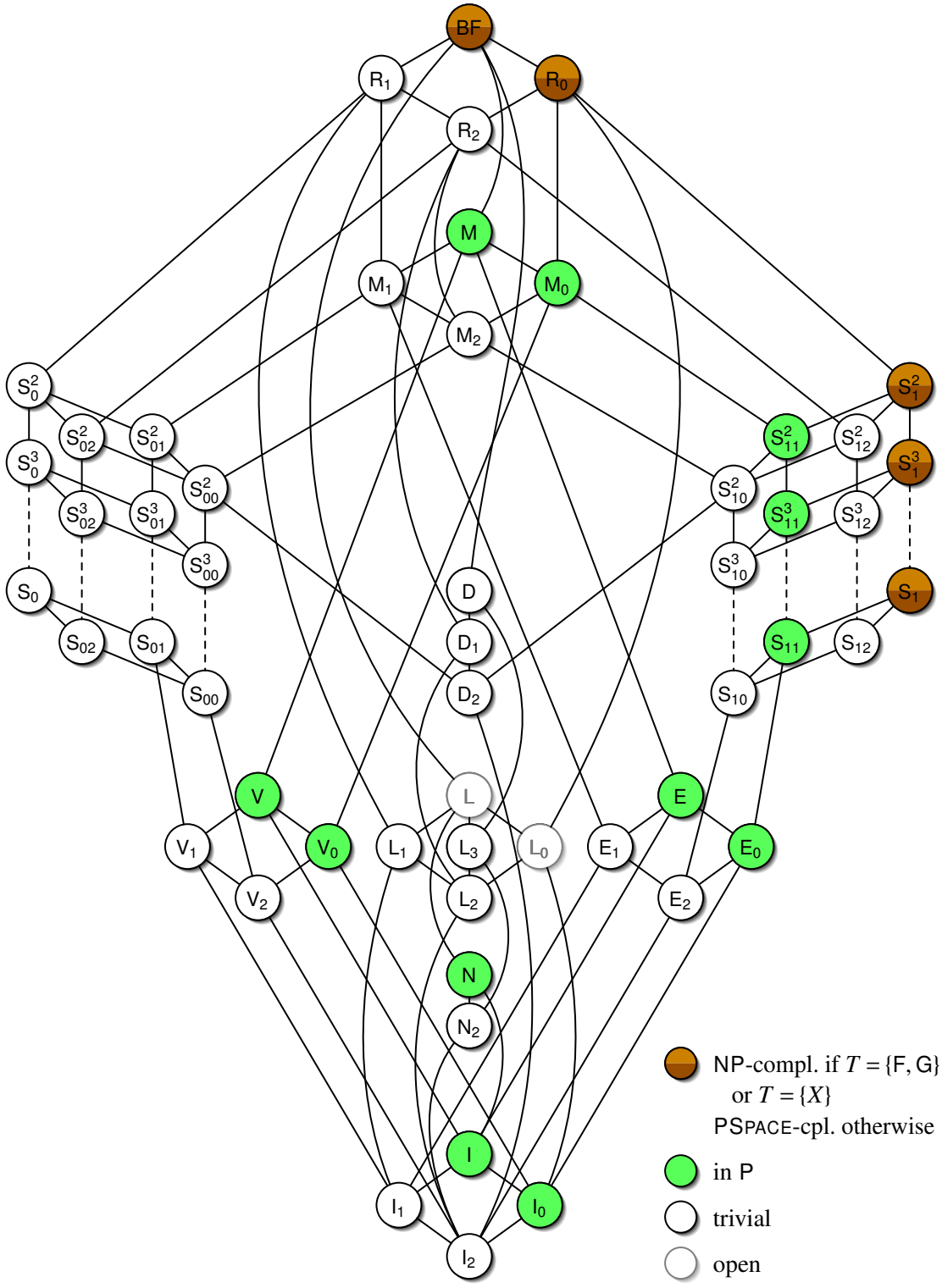


Figure 10: Results for $\mathsf{LTL}_T(B)$-SAT, Section 2.3.2

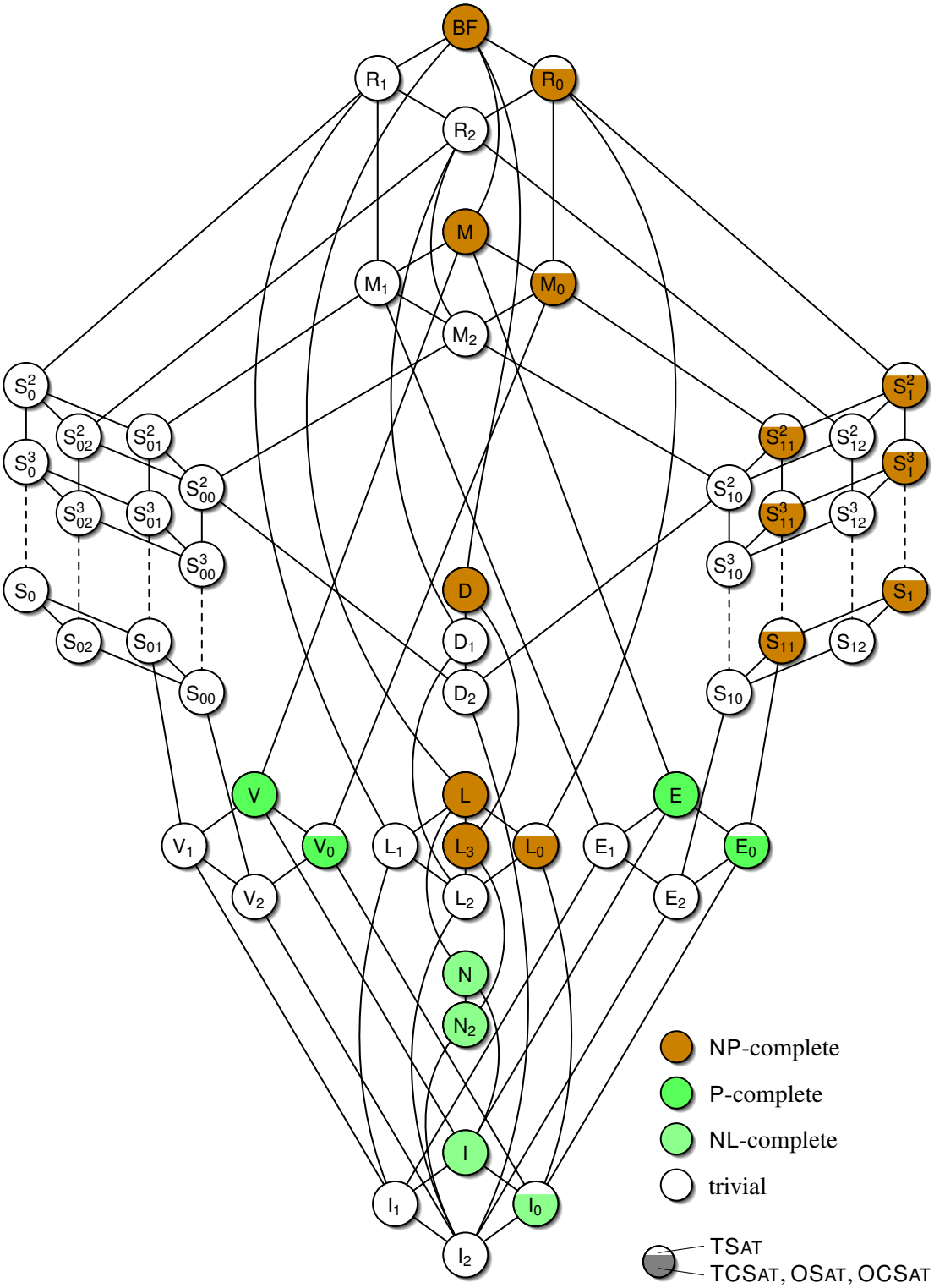


Figure 11: Results for $\mathrm{TSAT}_\emptyset(B)$, $\mathrm{TCSAT}_\emptyset(B)$, $\mathrm{OSAT}_\emptyset(B)$, and $\mathrm{OCSAT}_\emptyset(B)$, Section 2.4.2

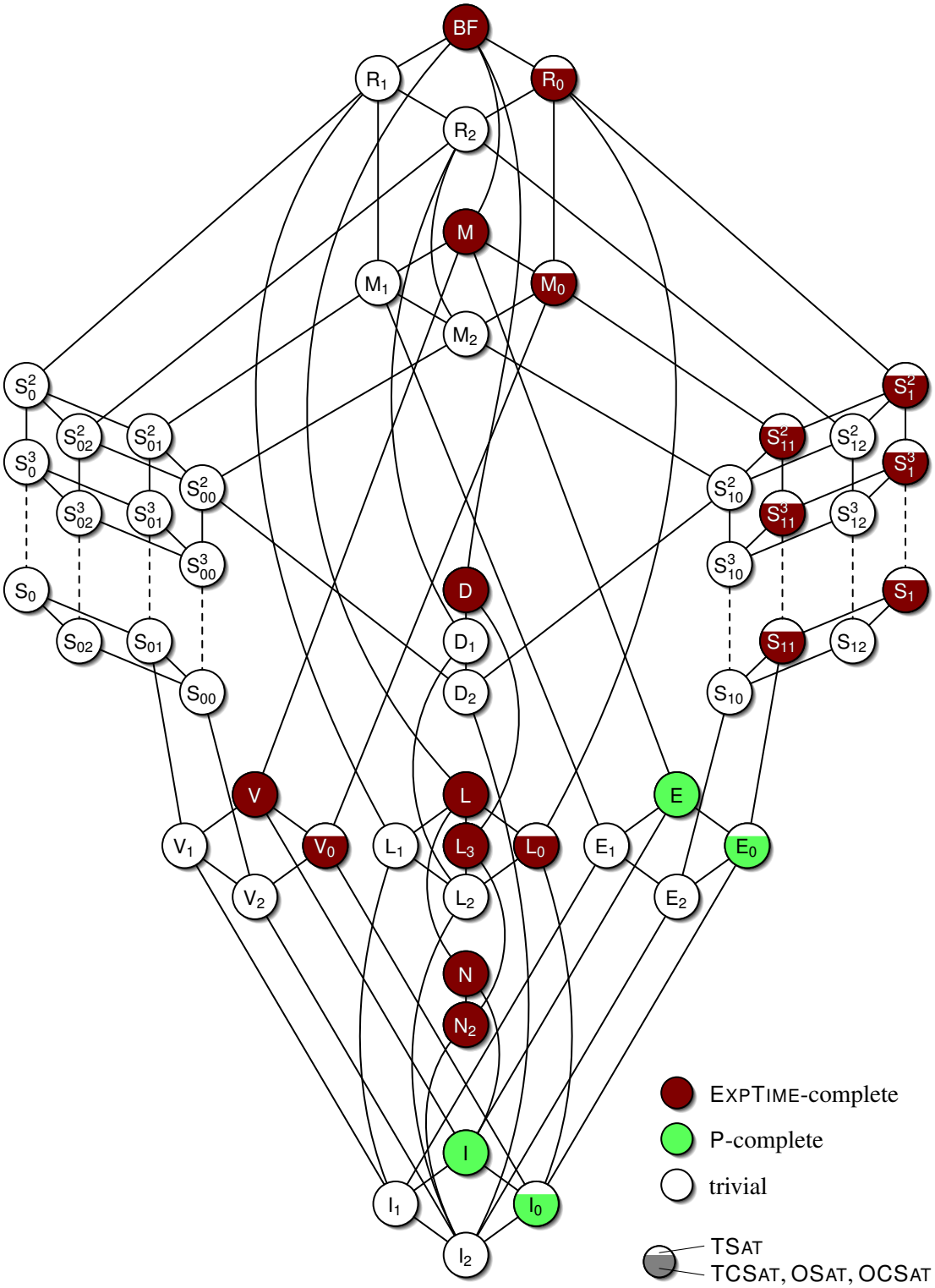


Figure 12: Results for $\mathsf{TSAT}_\exists(B)$, $\mathsf{TCSAT}_\exists(B)$, $\mathsf{OSAT}_\exists(B)$, and $\mathsf{OCSAT}_\exists(B)$, Section 2.4.2

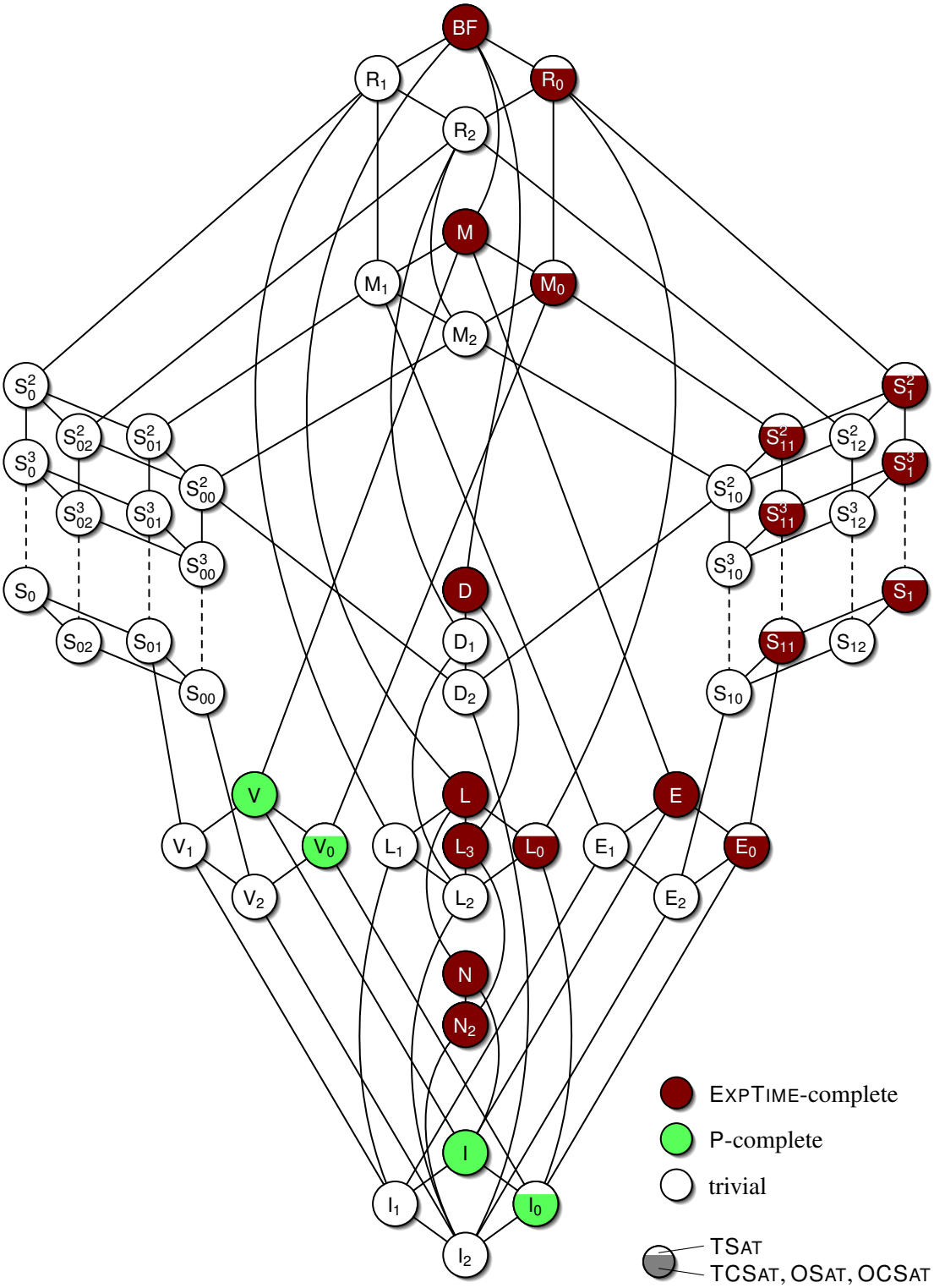


Figure 13: Results for $\mathsf{TSAT}_\forall(B)$, $\mathsf{TCSAT}_\forall(B)$, $\mathsf{OSAT}_\forall(B)$, and $\mathsf{OCSAT}_\forall(B)$, Section 2.4.2

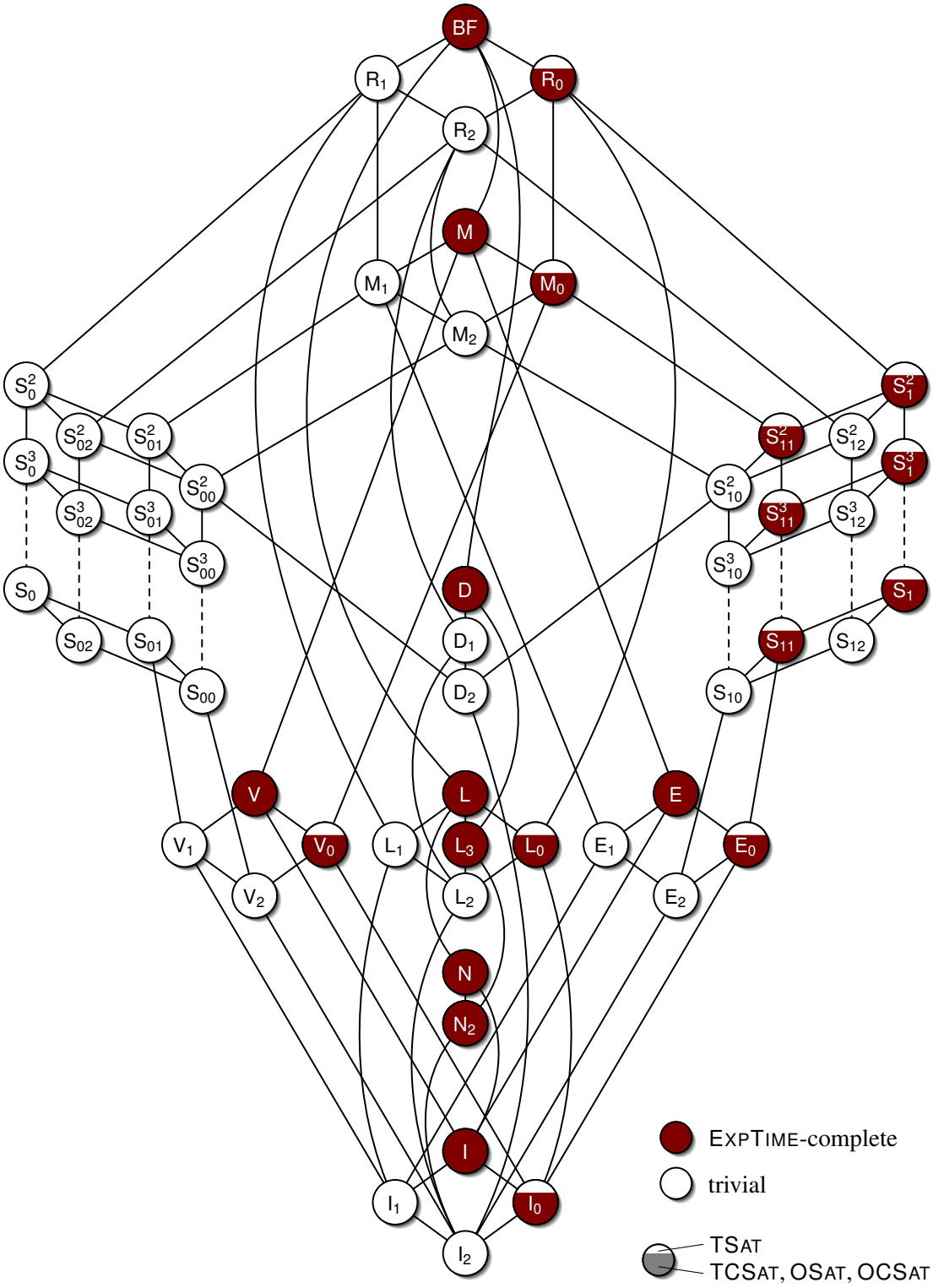


Figure 14: Results for $\mathsf{TSAT}_{\exists,\forall}(B)$, $\mathsf{TCSAT}_{\exists,\forall}(B)$, $\mathsf{OSAT}_{\exists,\forall}(B)$, and $\mathsf{OCSAT}_{\exists,\forall}(B)$, Section 2.4.2

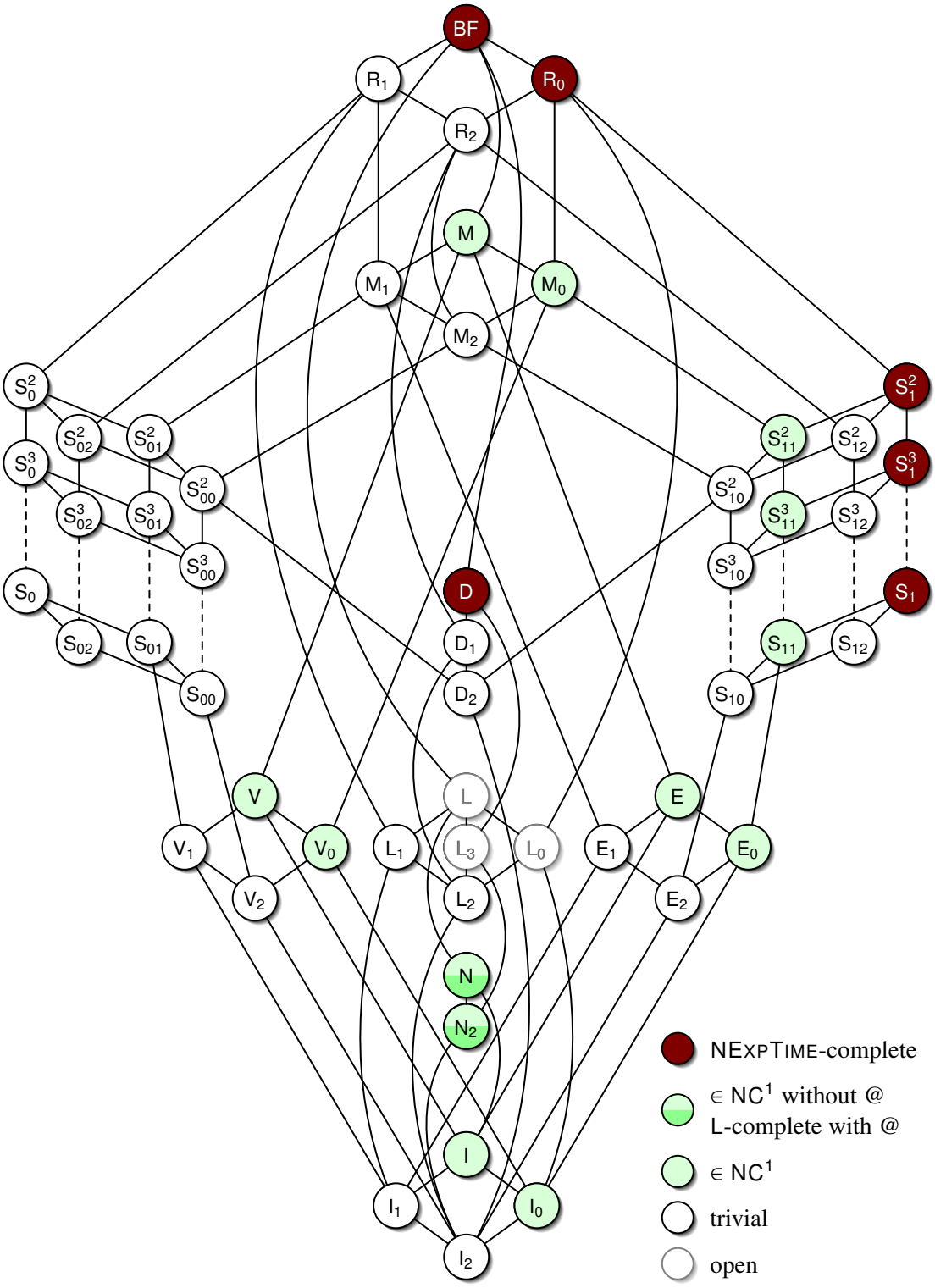


Figure 15: Results for $\mathsf{HL}_H(B)$-ER-SAT, Section 2.5.2

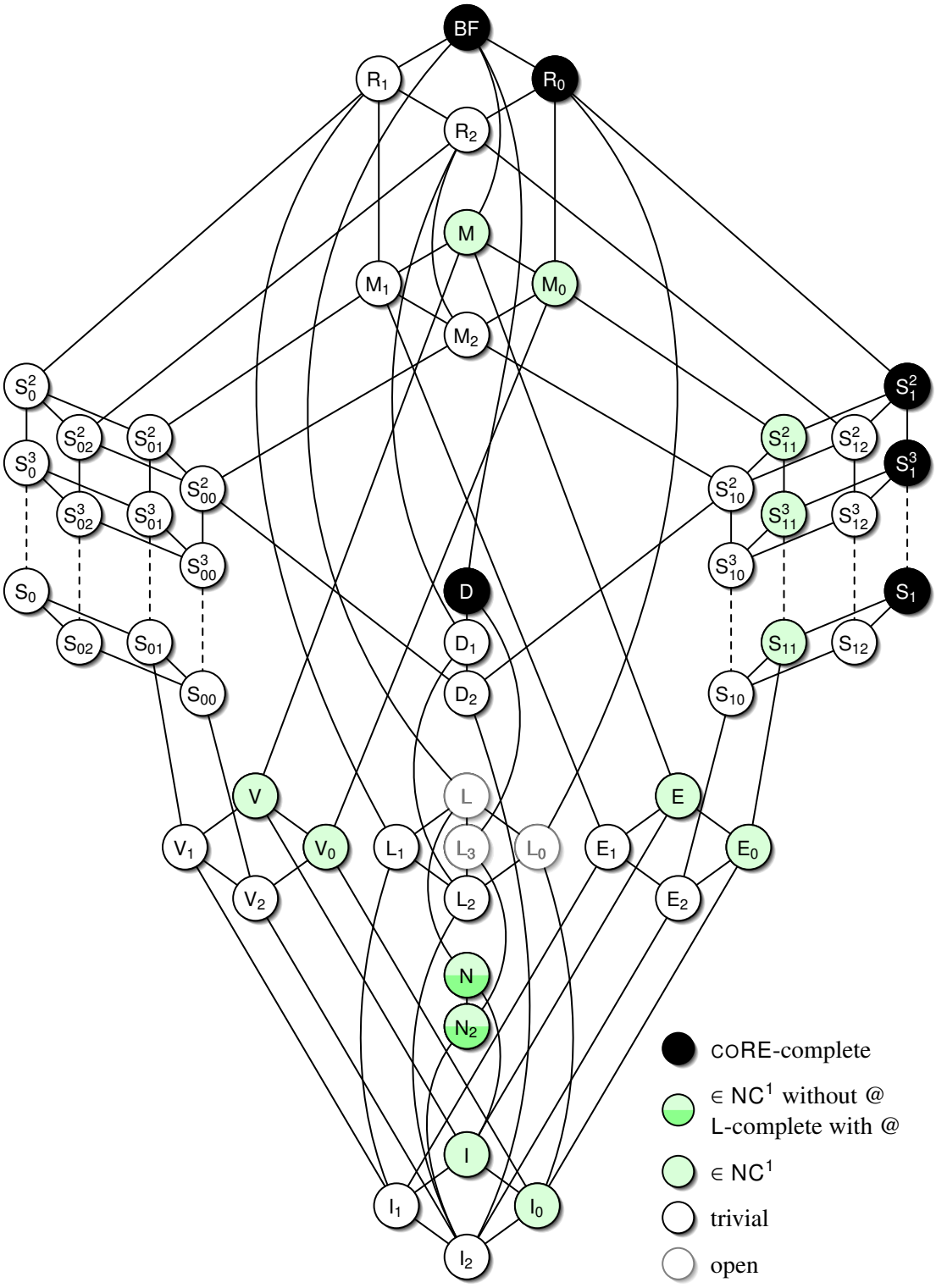


Figure 16: Results for $\mathsf{HL}_H(B)$-total-SAT, Section 2.5.2

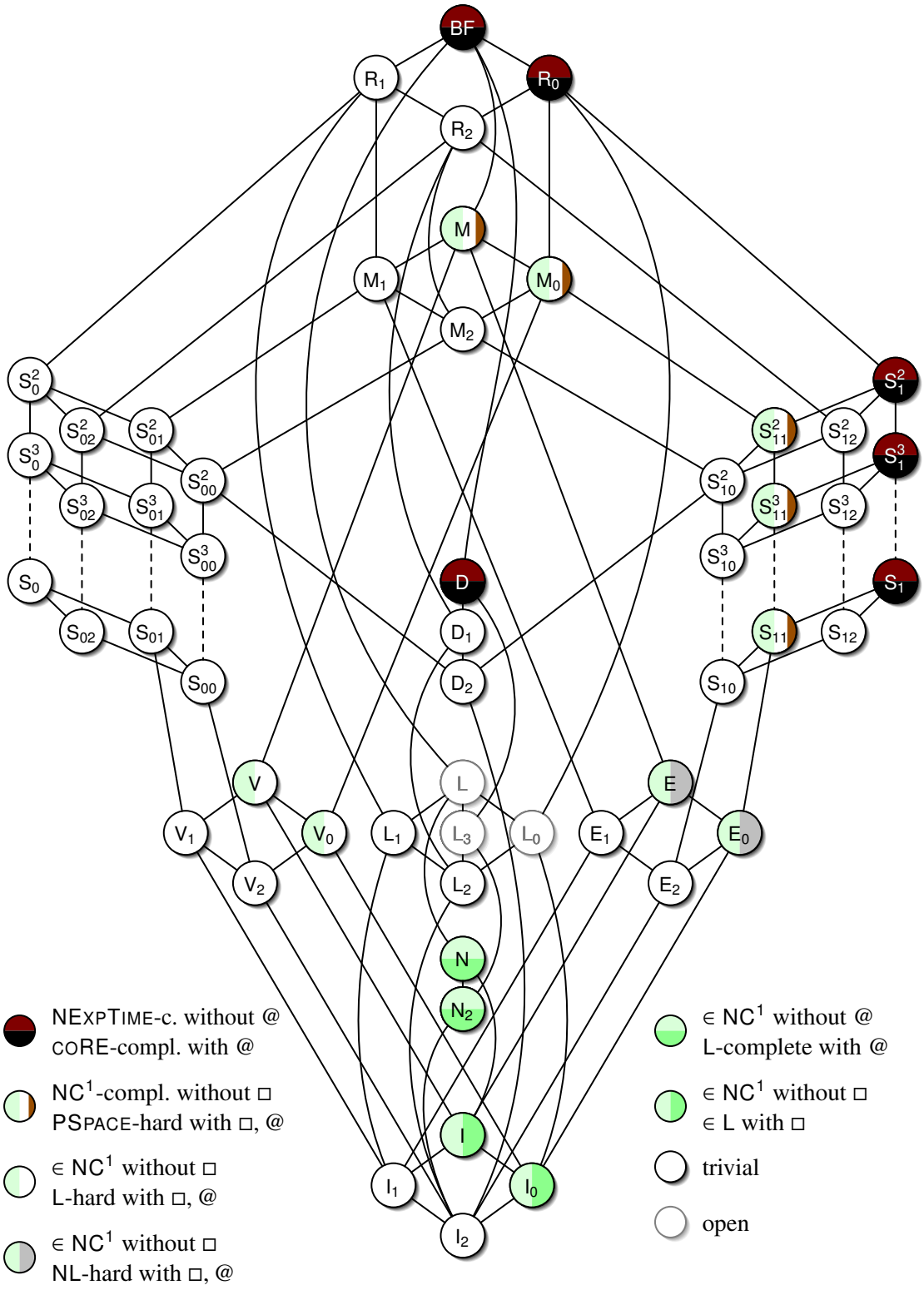


Figure 17: Results for $\mathsf{HL}_H(B)$-trans-SAT, Section 2.5.2

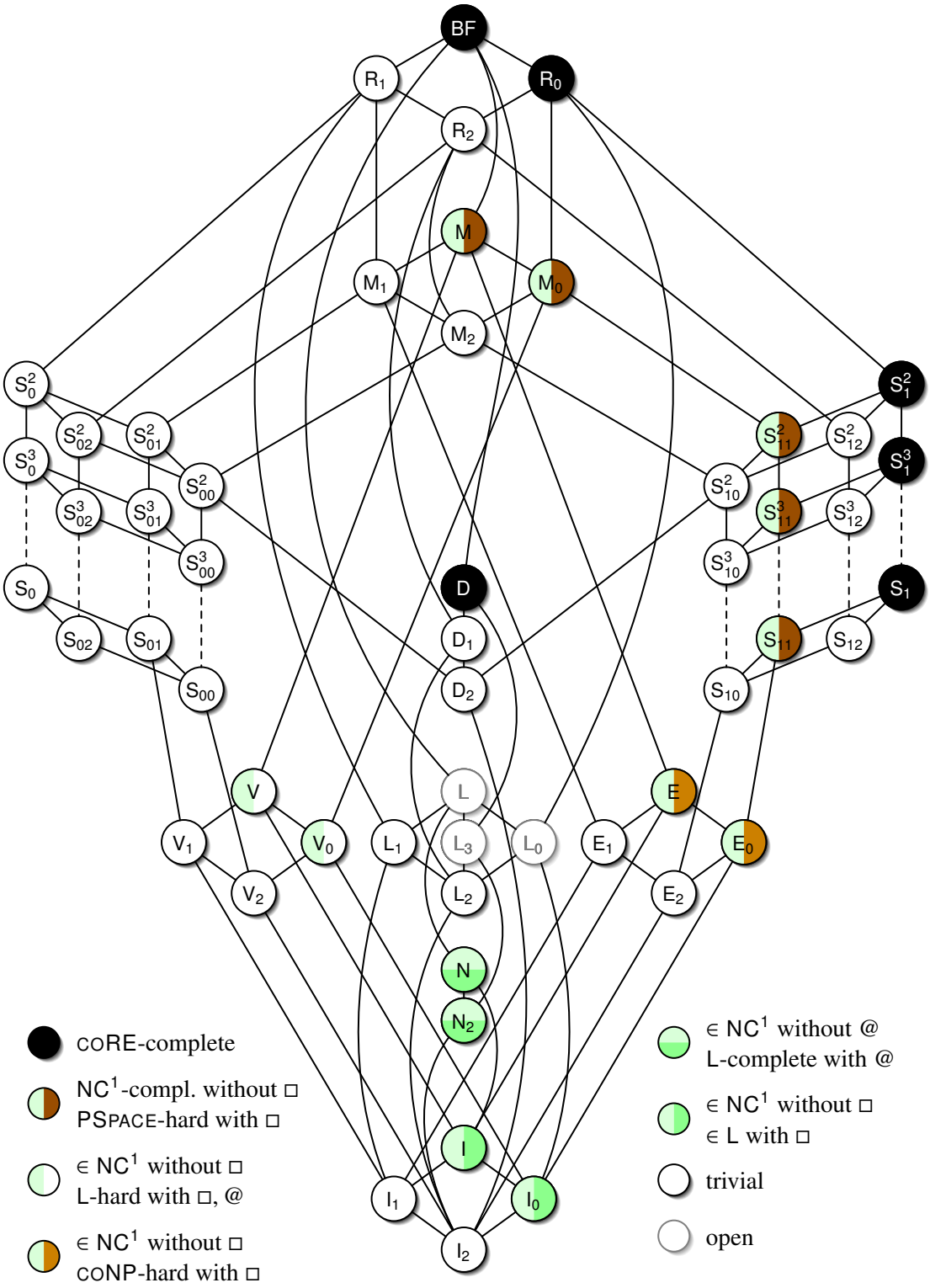


Figure 18: Results for $\mathsf{HL}_H(B)$-all-SAT, Section 2.5.2