The Complexity of Limited Belief Reasoning — The Quantifier-Free Case

Yijia Chen\textsuperscript{1} and Abdallah Saffidine\textsuperscript{2} and Christoph Schwering\textsuperscript{3}
\textsuperscript{1} Fudan University, Shanghai 201203, China
\textsuperscript{2} Australian National University, Canberra ACT 2600, Australia
\textsuperscript{3} University of New South Wales, Sydney NSW 2052, Australia
yijiachen@fudan.edu.cn, abdallah.saffidine@anu.edu.au, c.schwering@unsw.edu.au

Abstract
The classical view of epistemic logic is that an agent knows all the logical consequences of their knowledge base. This assumption of logical omniscience is often unrealistic and makes reasoning computationally intractable. One approach to avoid logical omniscience is to limit reasoning to a certain belief level, which intuitively measures the reasoning “depth.”

This paper investigates the computational complexity of reasoning with belief levels. First we show that while reasoning remains tractable if the level is constant, the complexity jumps to \textit{PSPACE}-complete— that is, beyond classical reasoning—when the belief level is part of the input. Then we further refine the picture using parameterized complexity theory to investigate how the belief level and the number of non-logical symbols affect the complexity.

1 Introduction

The standard way of modeling knowledge and belief\textsuperscript{1} in epistemic logic is in terms of possible worlds: an agent knows a proposition if and only if it is true in all worlds the agent considers possible. A side-effect of this model is that agents are assumed to be logically omniscient; that is, they know all the consequences of what they know [Hintikka, 1975].

Unfortunately, the assumption of logical omniscience is inappropriate for most resource-bounded agents like humans or robots: it drives up the computational cost of reasoning and is usually far beyond the their capabilities. Theories of limited belief therefore aim to lift the omniscience assumption.

A number of theories of limited belief have been proposed, predominantly in the 1980s and 1990s [Konolige, 1986; Kaplan and Schubert, 2000; Vardi, 1986; Fagin and Halpern, 1987; Levesque, 1984; Patel-Schneider, 1990; Lakemeyer, 1994; Delgrande, 1995]. A common problem with these approaches is, however, that either their model of limiting belief is too fine-grained or it misses out on simple inferences.

A novel approach to limited belief developed over a series of papers [Liu et al., 2004; Lakemeyer and Levesque, 2013; 2014; Klassen et al., 2015; Schwering and Lakemeyer, 2016; Lakemeyer and Levesque, 2016; Schwering, 2017] attempts to address this issue. The basic idea is to stratify beliefs into belief levels, where the first one, level 0, only comprises the explicit beliefs, that is, what is written down expressly in the knowledge base, and higher belief levels \( k + 1 \) draw additional conclusions based on what is believed at level \( k \). Semantically the logic can be characterized using sets of clauses instead of possible worlds, and through case splits, that is, by branching on all the values some term can take and propagating the value.

As an example, consider the following knowledge base:

\[ \text{fatherOf}(\text{Sally}) = \text{Frank} \lor \text{fatherOf}(\text{Sally}) = \text{Fred} \]
\[ \text{fatherOf}(\text{Sally}) = n \supset \text{rich}(n) = \top \text{ for } n \in \{\text{Frank}, \text{Fred}\}. \]

Here, Sally, Frank, Fred, \( \top \) name distinct individuals (\( \top \) is an auxiliary name for modeling propositions), whereas fatherOf and rich represent functions in the classical sense. From this knowledge we can deduce \( \text{rich}(\text{Frank}) = \top \lor \text{rich}(\text{Fred}) = \top \) at level 1 by splitting on all potential fathers of Sally: if Frank is the father, then Frank is rich; if Fred is the father, then he is rich; every other potential father contradicts the first clause.

Logics of limited belief in general and the belief level mechanism in particular aim to provide means of controlling the reasoning effort in a comprehensible and explainable way, as contrasted with using a classical reasoner and terminating it after a timeout, for example. The rationale behind the belief level approach is that reasoning at small belief levels should be relatively cheap but still sufficient for the average problem a human or a robot faces during their daily operation. Experiments confirm this hypothesis for the confined domains of Sudoku and Minesweeper [Schwering, 2017].

Contribution
In this paper, we analyze reasoning with belief levels from the perspective of complexity theory. More precisely, we study the problem of deciding whether a knowledge base entails a query at a certain belief level.

For a constant belief level, the problem is indeed in \textit{PTIME} and hence known to be tractable; the same holds when the knowledge base and query only mention a constant number of function terms.

However, we shall see that if both the belief level and the number of function terms are part of the problem input, then the complexity jumps to \textit{PSPACE}-complete! This may come as a surprise given that classical, unlimited reasoning is in \textit{co-NP}. So (large) belief levels appear to make reasoning harder.
Intuitively, the jump is caused by the belief level limiting a possibly scarce resource, namely the number of case splits, which needs to be utilized in an optimal way.

The gap between PTIME and PSPACE-completeness calls for a more refined analysis, which we carry out using parameterized complexity theory. We investigate three dimensions of parameters: (1) the belief level, (2) the number of function terms mentioned in the reasoning problem (in the above example, the function terms are fatherOf(Sally), rich(Frank), rich(Fred)), and (3) the number of mentioned so-called standard names (in the example, these names are Sally, Frank, Fred, ⊤). Parameterized complexity theory offers the W- and A-hierarchies to classify problems between PTIME and NP and between NP and PSPACE, respectively. We locate the parameterized variants of our problem within these hierarchies.

A comprehensive overview of the paper’s findings is given in Table 1. Figure 1 illustrates the relationships between the complexity classes we deal with in this paper.

The paper is structured as follows. The next section introduces the logic of limited belief and defines the reasoning problem that we shall study. Section 3 introduces a gadget that we use in several reductions. Section 4 begins the complexity analysis from the perspective of classical complexity theory with PTIME and PSPACE results. Section 5 refines the picture using parameterized complexity theory. Then we conclude.

Full proofs of our results can be found in [Chen et al., 2018].

2 The Logic of Limited Belief

In its most recent form, the logic of limited belief is a first-order logic with functions, equality, and epistemic modal operators [Lakemeyer and Levesque, 2016; Schwering, 2017]. In this paper, we limit our consideration to the quantifier-free case.

This section first introduces the syntax and semantics of this logic, and then defines the reasoning problem whose complexity we will study in the remainder of the paper: if we know K5 explicitly, do we believe α at level k? The definitions and results of this section are adopted from [Schwering, 2017] with some minor simplifications to ease the technical treatment; these simplifications do not affect the expressivity or complexity of the reasoning task at hand.

2.1 The Language

A term is either a standard name (or name for short) or a function term f(n₁, . . . , nₖ), where f is a function symbol and every nᵢ is a standard name. Standard names can be understood as special constants that satisfy the unique-names assumption and an infinitary version of domain closure. We assume an infinite supply of standard names as well as of function symbols.

A literal is an expression of the form t = n or ¬t = n, where t is a function term and n is a standard name. A formula is a literal or an expression of the form ¬α, (α ∨ β), or Bₖα, where α, β are formulas and k ≥ 0 is a natural number. We read Bₖα as “α is known at belief level k”; in case k = 0 we also say “α is known explicitly.” We use the usual abbreviations t ̸= n, (α ∧ β), (α ⊃ β), and may omit brackets to ease readability.

A formula without Bₖ is called objective. Schwering [2017] has shown that there is a linear Turing reduction from the reasoning problem with nested beliefs to the non-nested case. Hence to simplify the presentation we henceforth assume that α in Bₖα is objective. As usual, a conjunction of disjunctions of literals is said to be in conjunctive normal form (CNF).

2.2 The Semantics

The semantics of limited belief is based on clause subsumption, unit propagation, and case splits. A clause is a set of literals. We abuse notation and identify a non-empty clause {ℓᵰ₁, . . . , ℓᵰₙ} with the formula (ℓᵰ₁ ∨ . . . ∨ ℓᵰₙ). In the rest of this paragraph, we implicitly assume that n, n’ refer to distinct standard names. A clause c₁ subsumes another clause c₂ iff for every t = n ∈ c₁, either t = n ∈ c₂ or t ̸= n’ ∈ c₂, and for every t ̸= n ∈ c₁, also t ̸= n ∈ c₂. We say two literals ℓᵰ₁, ℓᵰ₂ are complementary when they are of the form t = n and t ̸= n or of the form t = n and t = n’. The unit propagation of a clause c and a literal ℓ is obtained by removing from c all literals complementary to ℓ. For a set of clauses s, we let UP(s) be the closure of s under unit propagation and subsumption.
The truth relation $\models$ is defined between a formula $\alpha$ and a set of clauses $s$. Intuitively, $s$ acts as a partial model. At belief level 0, $\alpha$ is broken down to clause level and then checked for subsumption by $UP(s)$. Higher belief levels allow to branch on a function term $t$ and all its values $n$ and add $t = n$ to $s$, which may then trigger unit propagation in $UP(s)$ and thus produce new inferences. The formal definition is as follows:

1. if $c$ is a clause: $s \models c$ iff $c \in UP(s)$
2. if $(\alpha \lor \beta)$ is not a clause: $s \models (\alpha \lor \beta)$ iff $s \models \alpha$ or $s \models \beta$
3. $s \models \neg(\alpha \lor \beta)$ iff $s \models \neg \alpha$ and $s \models \neg \beta$
4. $s \models \neg \neg \alpha$ iff $s \models \alpha$
5. $s \models B_0 \alpha$ iff $s \models \alpha$
6. $s \models B_{k+1} \alpha$ iff for some function term $t$, for all names $n$, $s \cup \{t = n\} \models B_k \alpha$
7. $s \models \neg B_k \alpha$ iff $s \not\models B_k \alpha$

In the remainder, we refer to these definitions as Rules 1–7. As usual, a formula $\alpha$ is valid, written $\models \alpha$, iff $s \models \alpha$ for every set of clauses $s$.

The belief level $k$ in $B_k \alpha$ specifies the number of case splits, which corresponds to the maximum permitted reasoning effort for proving $\alpha$. Limited belief is monotonic in the belief level:

**Lemma 1** $B_k \alpha \supset B_{k+1} \alpha$.

Moreover, belief stabilizes at a high-enough belief level in the following sense:

**Lemma 2** Let $F$ contain all function terms in $s$ and $\alpha$, and let $k \geq |F|$. Then $s \models B_k \alpha$ iff $s \models B_{|F|} \alpha$.

**Example**
Let us revisit the example from the introduction to illustrate how the semantics works. Let $s$ contain the clauses

- $fatherOf(Sally) = Frank \lor fatherOf(Sally) = Fred$
- $fatherOf(Sally) \neq Frank \lor rich(Frank) = \top$
- $fatherOf(Sally) \neq Fred \lor rich(Fred) = \top$

and let $c$ denote the clause $rich(Frank) = \top \lor rich(Fred) = \top$. Then $s \models B_1 c$ holds by splitting the cases for Sally’s father:

- $UP(s \cup \{fatherOf(Sally) = Frank\}) \models rich(Frank) = \top$ by unit propagation with the second clause.
- $UP(s \cup \{fatherOf(Sally) = Fred\}) \models rich(Fred) = \top$ by unit propagation with the third clause.
- $UP(s \cup \{fatherOf(Sally) = n\})$ for $n \notin \{Frank, Fred\}$ contains the empty clause by unit propagation with the first clause.

In each case, we obtain a clause that subsumes $c$, so for every potential father $n, c \in UP(s \cup \{fatherOf(Sally) = n\})$.

**Classical Semantics**
For future reference, we briefly give the classical, “unlimited” semantics of objective formulas. A world $w$ is a function from function terms to standard names. Truth of an objective formula $\alpha$ in a world $w$, written $w \models \alpha$, is defined as follows:

- $w \models t = n$ iff $w(t) = n$
- $w \models \neg \alpha$ iff $w \not\models \alpha$
- $w \models (\alpha \lor \beta)$ iff $w \models \alpha$ or $w \models \beta$
- $w \models \neg(\alpha \lor \beta)$ iff $w \models \neg \alpha$ and $w \models \neg \beta$

We write $s \models \alpha$ to say that for all $w$, if $w \models c$ for all $c \in s$, then $w \models \alpha$. Moreover, we write $\models \alpha$ for $\emptyset \models \alpha$.

Limited belief is sound as well as eventually complete with respect to classical semantics in the following sense:

**Proposition 3** For all finite $s$ and all $\alpha$, there is a (large-enough) belief level $k \geq 0$ such that $s \models B_k \alpha$ iff $s \models \alpha$.

**Proof sketch.** Soundness holds because Rule 6 branches over all names. Eventual completeness holds because $k$ can be chosen large enough to split all terms in $s, \alpha$.

2.3 The Limited Belief Reasoning Problem
The fundamental problem of reasoning about limited belief is to decide whether for a given knowledge base $KB$ and a query $\alpha$, if $KB$ is known explicitly, then $\alpha$ is known at belief level $k$. In limited belief, $KB$ is typically assumed to be CNF [Lakemeyer and Levesque, 2016]. The formal definition is:

<table>
<thead>
<tr>
<th>Limited Belief Reasoning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Instance:</strong> Objective formulas $KB$ and $\alpha$ over function terms $F$ and standard names $\mathcal{N}$, $KB$ in CNF, belief level $k \geq 0$.</td>
</tr>
<tr>
<td><strong>Problem:</strong> Decide whether $\models B_k KB \supset B_k \alpha$.</td>
</tr>
</tbody>
</table>

We shall investigate this problem using classical complexity theory first and then refine the picture using parameterized complexity theory for parameters $k, |F|, |\mathcal{N}|$. An overview of the results is in Table 1.

Since the knowledge base in Limited Belief Reasoning is assumed to be in CNF, it corresponds to a unique (modulo UP) set of clauses and the problem can be equivalently expressed as a model checking problem:

**Lemma 4** Let $KB$ be in CNF with clauses $s = \{c_1, \ldots, c_j\}$. Then $\models B_0 KB \supset B_k \alpha$ iff $s \models B_k \alpha$.

Thus and by Lemma 2 and Proposition 3, Limited Belief Reasoning is sound and eventually complete with respect to classical reasoning: $\models B_0 KB \supset B_{|F|} \alpha$ iff $KB \supset \alpha$.

Finally, the following lemma says that in Rule 6 a finite number of function terms and standard names is sufficient.

**Lemma 5** Let $F$ (resp. $\mathcal{N}$) contain all function terms (resp. standard names) in $s, \alpha$, and let $n \notin F$ be an additional name. Then $s \models B_{k+1} \alpha$ iff for some $t \in F$, for all $n \in F \cup \{n\}$, $s \cup \{t = n\} \models B_k \alpha$.

Together, Lemmas 4 and 5 give rise to a decision procedure for Limited Belief Reasoning, which works as follows. First, the problem is turned into the equivalent model checking problem using Lemma 4. Then the procedure applies Lemma 5 to reduce the belief level, and finally follows Rules 1–5 to break $\alpha$ down to clause level and check the clauses for subsumption. It is already known that this procedure runs in time $O(2^k(|KB| + |\alpha|)^{k+3})$ [Schwering, 2017].

3 Ordering Gadget
It is easy to see that the ordering in which terms are split can be relevant. For example, let $s$ contain the following four clauses:

- $f = n \lor g_1 = n \lor h = n$ 
- $f \not\models n \lor g_2 = n \lor h = n$
- $f = n \lor g_1 \not= n \lor h = n$ 
- $f \not\models n \lor g_2 \not= n \lor h = n$
We can prove \( s \models B_2 h = n \) by splitting \( f \) first and then, depending on the value of \( f \), splitting \( g_1 \) or \( g_2 \) next, but not the other way around.

In this section we construct a gadget that generalizes this idea in order to enforce that a goal formula can only be proved by splitting terms in a certain order (at polynomial cost in space). This gadget is used repeatedly in the proofs of Sections 4 and 5. For example, in Theorem 9 we use it to preserve the quantifier ordering of the quantified Boolean formula.

To begin with, the following lemma shows how to make sure that one of the terms from a set \( F \) is split no later than at belief level \( k \). We use the notation \( [k] \) for \( \{1, \ldots, k\} \).

**Lemma 6** Let \( F \) be a non-empty finite set of function terms, and \( L \) be a set of literals where every term from \( F \) occurs exactly once. Let \( B_k \alpha \) be a formula with \( k \geq 1 \). Let \( s \) be a set of clauses such that for all \( t \notin F \), for some name \( n \), \( s \models \{t = n\} \not\models \lor_{\ell \in L} \neg \ell \) and \( s \models \{t = n\} \not\models \lor_{\ell \in L} \ell \).

Let \( c_i^0 \) and \( c_i^0 \) for \( i \in [k - 1] \), \( j \in [k] \) be literals with distinct function terms \( f^0_i \) and \( f^0_j \) that do not occur in \( s \) or \( \alpha \). Let \( c_i^0 \) stand for \( c_i^0 \lor \cdots \lor c_i^0 \). Let \( s_k \) be the least set that includes \( s \) and for every \( \ell \in L \) contains the clauses

\[
- \ell \lor c_k^0 \\
\ell \lor c_i^0 \lor \cdots \lor c_i^0 \lor c_k^0 \\
\ell \lor -c_i^0 \lor c_i^0 , \ldots , \ell \lor -c_i^0 \lor c_k^0
\]

Then

\[ s_k \models B_k(c_k^0 \land \lor_{\ell \in L} \neg \ell \lor \alpha) \]

iff for some \( t \in F \), for all names \( n \), \( s \models \{t = n\} \not\models B_{k-1} \lor_{\ell \in L} \neg \ell \lor \alpha \).

**Proof.** The proof proceeds in four steps.

**Claim 1.** \( s_k \setminus s \not\models B_{k-1} c_k^0 \).

**Proof of Claim 1.** By assumption, no two literals in \( L \) are complementary. Hence the only way of proving \( c_k^0 \) is by splitting some \( t \in F \) and \( f^0_1, \ldots, f^0_{k-1} \), which requires belief level \( k \). The proof is by induction on \( k \).

**Claim 2.** For all \( t \notin F \cup \{f^0_1, \ldots, f^0_{k-1}, f^0_k, \ldots, f^0_k\} \), for some \( n \), \( s_k \cup \{t = n\} \not\models B_{k-1} c_k^0 \).

**Proof of Claim 2.** By assumption, there is some \( n \) such that \( s \models \{t = n\} \not\models \lor_{\ell \in L} \neg \ell \lor \alpha \) for all \( \ell \in L \). Hence and since \( f^0_i \) do not occur in \( s \), \( s_k \cup \{t = n\} \not\models B_{k-1} c_k^0 \) iff \( s_k \setminus s \not\models B_{k-1} c_k^0 \), which holds by Claim 1.

**Claim 3.** For all \( t \in F \), for all \( n \), \( s_k \cup \{t = n\} \not\models B_{k-1} c_k^0 \).

**Proof of Claim 3.** Let \( t \in F \) and let \( n \) be an arbitrary name. Then for all names \( n_1, \ldots, n_k \), \( c_k^0 \in UP(s_k \cup \{t = n, f^0_1 = n_1, \ldots, f^0_{k-1} = n_k\}) \). Thus \( s_k \cup \{t = n\} \not\models B_{k-1} c_k^0 \).

**Proof of the lemma.** For the only-if direction, by Claim 2, \( s_k \cup \{t = n\} \not\models B_{k-1} (c_k^0 \land \lor_{\ell \in L} \neg \ell \lor \alpha) \) for all \( n \) for some \( t \in F \cup \{f^0_1, \ldots, f^0_{k-1}, f^0_k, \ldots, f^0_k\} \). By Claim 3 and since \( f^0_i, f^0_j \) do not occur in \( s \) or \( \alpha \), we can assume \( t \in F \).

For the converse direction, if \( n \) is such that \( t = n \) is complementary to \( \neg \ell \) for some \( \ell \in L \), then \( c_k^0 \in UP(s_k \cup \{t = n\}) \).

Otherwise, \( t = n \) subsumes \( \neg \ell \) for some \( \ell \in L \) and by Claim 3, the remaining splits suffice to prove \( c_k^0 \).

The next lemma represents our gadget. Despite its somewhat intimidating interface, it simply plugs together repeated applications of the previous lemma to completely determine the ordering of splitting terms from \( F_1, \ldots, F_l \).

**Lemma 7** Let \( F_1, \ldots, F_l \) be non-empty finite sets of function terms, \( F = F_1 \cup \cdots \cup F_l \) and \( L_1, \ldots, L_l \) be sets of literals such that every term from \( F_i \) occurs exactly once in \( L_i \). Let \( B_\alpha \) be a formula with \( k \leq l \). Let \( s \) be a set of clauses such that for all \( k \in [l] \), for all \( t_1 \in F_1, \ldots, t_{k+1} \in F_{k+1}, t_k \notin F \), for all \( n_1, \ldots, n_{k+1}, n_k, s \models \{t_k = n_k, \ldots, t_1 = n_1\} \not\models \lor_{\ell \in L_k} \ell \).

For every set of clauses \( s \), let \( s'_1 \) and \( c_i^0 \) be as in Lemma 6 with respect to \( F_i, L_1, \alpha \). Let \( \alpha_0 \) be \( \alpha \) and \( \alpha_i \) for \( i > 0 \) be \( c_i^0 \lor \lor_{\ell \in L_i} \neg \ell \lor \alpha_{i-1} \). Then

\[ ((s_1) \ldots k) \models B_k \alpha_k \]

iff for some \( t_k \in F_k \), for all names \( n_k, \ldots \),

for some \( t_1 \in F_1 \), for all names \( n_1 \),

\[ s \models \{t_1 = n_1, \ldots, t_k = n_k\} \not\models \lor_{\ell \in L_k, i \in [k]} \ell \lor \alpha \].

**Proof.** By induction on \( k \), where Lemma 6 can be applied since \( s' \not\models \lor_{\ell \in L_k, \ell} \).

**4 Classical Complexity**

This section analyzes the complexity of Limited Belief Reasoning using classical complexity theory. The next tractability result follows from the decision procedure from Section 2.3:

**Corollary 8** Limited Belief Reasoning with constant \( k \) or constant \( |F| \) is in \( \text{PTIME} \).

**Proof.** The decision procedure runs in time polynomial with degree \( k + 3 \). By Lemmas 1 and 2, \( |F| \) suffices.

Next, we consider the case where neither \( k \) nor \( |F| \) is constant. It comes as no surprise that the complexity then significantly increases with the number of case splits. Proposition 3 and Lemma 4 showed that Limited Belief Reasoning is sound and eventually complete with respect to classical reasoning. So clearly, Limited Belief Reasoning must be \( \text{co}-\text{NP}-hard \), and eventual completeness may suggest that it is \( \text{co}-\text{NP}-complete \) as well. However, limiting the number of case splits further adds to the computational complexity: whereas in classical reasoning a decision procedure may “simply” split all function terms, a decision procedure for limited belief needs to make sure it makes use of the available case splits in the best possible way. This leads to the following result:

**Theorem 9** Limited Belief Reasoning with constant \( |N| \) is \( \text{PSPACE}-complete \). The result also holds when \( |N| \) is input.

**Proof.** Membership. The decision procedure from Section 2.3 runs in space \( O(m + k) \) where \( m = |KB| + |\alpha| \), since \( \text{UP}(s) \) can be represented in space \( O(|\alpha|) \) because minimal clauses suffice.

**Hardness.** We reduce from True Quantified Boolean Formula, which is \( \text{PSPACE}-complete \) [Arora and Barak, 2009]. The problem input is a fully quantified Boolean formula \( Q_k x_k \ldots Q_1 x_1 \psi \) for \( Q_i \in \{\forall, \exists\} \) and a propositional formula \( \psi \). Without loss of generality, we assume that \( \psi \) mentions only negation in front of variables. The question is whether
this formula evaluates to TRUE, that is, for all (if $Q_k = \forall$) / some (if $Q_k = \exists$) assignment(s) of $x_k$, ..., for all / some assignment(s) of $x_1, (x_1), ... , x_k$) satisfies $\psi$.

Let $N = \{\top, W\}$ contain two standard names. Let $F = \{f_j | Q_k = \forall\}, F_2 = \{f_j, f_j' | Q_k = \exists\}$. $F = F_1 \cup F_2$, where $f_j, f_j'$ are pairwise distinct function terms. We define a mapping $\cdot$ * from QBF to limited belief formulas: let $x_k^*$ be $f_j = \top$, let $(\neg x_k^*)'$ be $f_j \neq \top$ if $Q_k = \forall$ and $f_j' = \top$ if $Q_k = \exists$, let $(x_k^* \lor \psi_2^*)'$ be $(x_k^* \lor \psi_2^*)$, and let $(x_k^* \land \psi_2^*)'$ be $(x_k^* \land \psi_2^*)$. For $Q_k = \forall$, let $F_2 = \{f_j\}, N_i = \{n |$ name $n$ is distinct from $W\}$, and $L_i = \{f_j \neq W\};$ for $Q_k = \exists$, let $F_1 = \{f_j, f_j'\}, N_i = \{\top\}$, and $L_i = \{f_j = \top, f_j' = \top\}$.

The idea is as follows. Universally quantified $x_k$ are naturally translated to literals $f_j = \top$ so that the truth values TRUE and FALSE of $x_k$ correspond to $f_j = \top$ and $f_j \neq \top$, respectively. For existentially quantified $x_k$, positive occurrences of $x_k$ are replaced with $f_j = \top$ and negative ones with $f_j' = \top$ so that the truth values TRUE and FALSE of $x_k$ correspond to $f_j = \top$ and $f_j' = \top$. The $f_j$ or $f_j'$ (if $Q_k = \exists$) then need to be split in the appropriate order.

We can show by induction on $k$ and substitution on $\psi$ that $\phi$ evaluates to TRUE iff for some $t_k \in F_k$, for all $n_k \in N_k, ..., for some t_1 \in F_1$, for all $n_1 \in N_1, \{t_1 = n_1, ..., t_k = n_k\} \models \psi^*$. The theorem then follows because the restriction $n_k \in N_k$ on the right-hand side can be lifted by replacing $\psi^*$ with $\forall t \in L_i \land i \in [k] - t \lor \psi^*$, which then reduces in polynomial time to Limited Belief Reasoning by Lemmas 7 and 4. □

It is noteworthy that this reduction only uses two standard names. Thus even the propositional case (where an atomic name $p$ is simulated by $p = \top$) is PSPACE-complete.

5 Parameterized Complexity

The gap between tractability and PSPACE-completeness from the previous section calls for a more refined analysis. In this section we use parameterized complexity theory to investigate how the parameters $k, |F|$, and/or $|N|$ affect the complexity of Limited Belief Reasoning.

While many unparameterized problems can be classified with the classical classes PTIME, NP, or PSPACE, parameterized versions of these problems fall into a variety of complexity classes [Flum and Grohe, 2006]. The role of PTIME in parameterized complexity is taken on by FPT, which includes problems parameterized by $k$ that are solvable in $f(k) \cdot p(n)$, where $f$ is a computable function and $p$ a polynomial. Other important parameterized classes come from the W- and A-hierarchies: the classes $\text{W}[1] \subseteq \text{W}[2] \subseteq \ldots \subseteq \text{W}[\text{SAT}] \subseteq \text{W}[\text{P}]$ include parameterized versions of different natural NP-complete problems; similarly, the classes $\text{A}[1] \subseteq \text{A}[2] \subseteq \ldots \subseteq \text{AW}[\text{SAT}] \subseteq \text{AW}[\text{P}]$ can be seen as a parameterized version of the polynomial hierarchy. This analogy is a simplification—links between PSPACE and W[1] have recently been exhibited [Bonnet et al., 2017]—but it suffices for our purpose and we display an intuitive representation of the relationships between these classes in Figure 1.

Membership in classes of the W- and A-hierarchies can be shown using machines that restrict the number of nondeterministic steps [Chen et al., 2005]. An NRAM is a random access machine with a nondeterministic EXISTS instruction which guesses a natural number less than or equal to a certain register and stores the number in that register. A problem is in $\text{W}[\text{P}]$ iff it is decidable by an NRAM in $f(k) \cdot p(n)$ steps, at most $g(k)$ of them being nondeterministic, using at most the first $f(k) \cdot p(n)$ registers and only numbers $\leq f(k) \cdot p(n)$. A problem is in $\text{AW}[\text{P}]$ iff it is decidable by an ARAM with the same constraints, where an ARAM is an NRAM with an additional nondeterministic FORALL instruction, the dual to EXISTS. Hardness in parameterized complexity is shown by way of FPT-reductions, which are reductions computable in time $f(k) \cdot p(n)$ and such that $k' \geq g(k)$, where $k$ and $k'$ are the parameters of the problems reduced from and reduced to, respectively, $n$ is the input size, $f$ and $g$ are computable functions, and $p$ is a polynomial.

Before starting our analysis, we introduce the Quantified Monotone Circuit Satisfiability problem. A circuit is a directed acyclic graph $(V, E)$ whose vertices are partitioned into input-nodes $X$ of in-degree $0$, not-nodes of in-degree $1$, and or-nodes of in-degree $\geq 0$, and a distinguished output-node $v_0$ of out-degree $0$. An assignment $S \subseteq X$ sets inputs $S$ to TRUE and the other ones to FALSE and propagates the values to the output node, whose value then determines whether or not $S$ satisfies the circuit. A monotone circuit contains no not-nodes.

**Quantified Monotone Circuit Satisfiability**

**Instance:** A monotone circuit $C$ with input-nodes partitioned into sets $X_1, \ldots, X_i$.

**Parameter:** $k_1 + \ldots + k_l$

**Problem:** Decide whether for all $S_1 \subseteq X_1$ with $|S_1| = k_1$, for some $S_2 \subseteq X_2$ with $|S_2| = k_2$, ..., the assignment $S_1 \cup \ldots \cup S_l$ satisfies $C$.

The following lemma states AW[P]-completeness for the problem, which has been claimed elsewhere before without explicit proof [Abrahamson et al., 1995].

**Lemma 10** Quantified Monotone Circuit Satisfiability is AW[P]-complete.

**Proof.** It is sufficient to reduce from Quantified Circuit Satisfiability, which is AW[P]-complete [Flum and Grohe, 2006]. Consider an instance with circuit $C = (V, E)$ and inputs $X_1, \ldots, X_i$. By De Morgan’s laws we can assume the not-nodes are right above the inputs. Observe that a not-node $w \in V$ with input $x \in X_i$ is TRUE iff at least $k_i$ variables in $X_i \setminus \{x\}$ are set to TRUE. The latter property can be expressed in a monotone circuit of polynomial size using or-nodes $b_{i_1,i_2,t}$ that represent that at least $t$ of $x_{i_1}, \ldots, x_{i_2}$ are set to TRUE, and nodes conjoining pairs of these nodes to express that $t'$ of $x_{i_1}, \ldots, x_{i_2}$ and $t - t'$ of $x_{i_3}, \ldots, x_{i_2}$ are set to TRUE. □

With this lemma we can establish the complexity of Limited Belief Reasoning parameterized by the belief level:

**Theorem 11** Limited Belief Reasoning with parameter $k$ is AW[P]-complete.

**Proof.** Membership. We implement the decision procedure from Section 2.3 using an ARAM. Model checking at belief level $0$ can clearly be done on a RAM in time $p(m)$, where $p$ is a polynomial and $m = |KB| + |\alpha|$. At belief level $k > 0$ we select one of the function terms from $F$ with EXISTS, and
the corresponding name from \( \mathcal{N} \cup \{ \bar{n} \} \) with \textsc{forall}. This amounts to \( 2 \cdot k \cdot n \) nondeterministic steps and a total runtime \( 2 \cdot k \cdot p(m) \), so the problem is in \textsc{aw[p]}.\footnote{\textsc{aw[p]}-complete.}

\textbf{Hardness.} We reduce from \textsc{quantified monotone circuit satisfiability}, which is \textsc{aw[p]}-complete by Lemma 10. Let 
\[ C = (V, E) \] 
be a monotone circuit with inputs \( x_1, \ldots, x_i \). We say \( x_i \) or \( x_i \not\in \mathcal{N} \) is \textit{universal} (\textit{existential}) if \( i \) is odd (even). For a parameter as well, this gives us leverage to replace the non-

\textit{Proof. Membership.} The idea is to represent that a node \( v \) is the least set such that \( v \subseteq \mathcal{N} \) and for each we select a standard name from \( \mathcal{N} \cup \{ \bar{n} \} \).

It is then straightforward to show by induction on \( l \) and substitution on the depth of \( C \) that for all \( S \subseteq X \) with \( |S| = k \), some \( S \subseteq X \) with \( |S| \leq k \cdot 2, \cdots, \), the truth assignment \( S_1 \cup \cdots \cup S_l \) satisfies \( C \) iff for some \( t_1, t_2, \cdots, t_l \in F_{1,1}, \) for all \( n_1, \ldots, n_l \in N_1 \), \( s \subseteq \cup \{ t_1, t_2, \cdots, t_l \} \), \( t(n_1, \ldots, t(l) = n_l, t(n)) \} \approx f_{\bar{n}} \) for \( v \). For each \( s \) and its inputs \( W = \{ \{ w \} \} \), and \( f_{\bar{n}} \not\in v \) for all inputs \( v \). We then prove that every \( S \subseteq X \) with \( |S| = k \) falsifies \( C \) iff for all \( n_1, \ldots, n_k \in \{ n_v \} \), \( s \subseteq \cup \{ t_1, t_2, \cdots, t_l \} \), \( t(n_1, \ldots, t(l) = n_l, t(n)) \} \approx f_{\bar{n}} \) by induction on the depth of \( C \). The right-hand side can be rewritten to match Lemma 7 using \( L_i \) and \( f_{\bar{n}} \) for all \( i \) and \( j \), encoded by the and/or nodes 4 and 5, which gives us \textsc{aw[p]}

\textbf{Membership in \textsc{aw[p]}.} This is quite natural due to the alternation of existential and universal quantifications of case splits in Lemma 5. When the number of standard names \( |\mathcal{N}| \) becomes a parameter as well, this gives us leverage to replace the non-

deterministic \textsc{forall} steps that select the standard names with a simple loops. It is therefore not surprising that Limited Belief Reasoning parameterized by \( k \) and \( |\mathcal{N}| \) is in \textsc{w[p]}, the hardest NP-analogue of the \( \mathcal{W} \)-hierarchy. The following result shows that the problem is in fact \textsc{w[p]}-complete:

\textbf{Proposition 12 Limited Belief Reasoning with parameters \( k \) and \( |\mathcal{N}| \) is \textsc{w[p]}-complete. The result also holds when \( |\mathcal{N}| \) is constant.}

\textit{Proof. Membership.} We build an \textsc{nram}. For \( k = 0 \), it be-

\textit{Hardness.} We reduce from \textsc{weighted monotone circuit satisfiability}, which is \textsc{w[p]}-complete [Abrahamsson et al., 1995] and identical to the quantified problem with only a single block of existential variables, and the proof accordingly carries over from Theorem 11.\footnote{\textsc{w[p]}-hardness.}

Next we consider the case where \( |\mathcal{F}| \) becomes a parameter. The below theorem specifies \textsc{co-w[p]}-completeness:

\textbf{Theorem 13 Limited Belief Reasoning with parameters \( k \) and \( |\mathcal{F}| \) is \textsc{co-w[p]}-complete. The result also holds when \( k \) is input.}

\textit{Proof. Membership.} We show that the co-problem is in \textsc{w[p]} using an \textsc{nram} that finds a falsifying assignment of names for all split terms. As in Theorem 11, the case \( k = 0 \) is straightforward. For \( k > 0 \) we loop over all function terms in \( \mathcal{F} \) and for each we select a standard name from \( \mathcal{N} \cup \{ \bar{n} \} \) with \textsc{exists}. This requires \( |\mathcal{F}|^{|\mathcal{F}|} \) nondeterministic steps.

\textit{Hardness.} We reduce from the complement of \textsc{weighted anti-monotone circuit satisfiability}, which is \textsc{w[p]}-complete [Flum and Grohe, 2006]. A circuit is anti-monotone when all inputs have out-degree 1 and feed into a not-node and there are no other not-nodes except those on top of some input. Let 
\[ C = (V, E) \] 
be an anti-monotone circuit with inputs \( X \). For \( x \in X \) we denote the associated not-node by \( v_x \).

Let \( \mathcal{F} = \{ f_i \mid i \in [k] \} \) \( \mathcal{F} \) is function terms. Let 
\[ \mathcal{N} = \{ n_v \} \cup [\forall x, f \in X] \cup [\forall w \in W] \] 
be standard names.

The idea is to represent that a node \( v \) is set to \textsc{false} by \( f \not\equiv n_v \). The truth assignment is selected by splitting \( f_1, \ldots, f_k \) \( f \). Truth of an input \( x \) is represented by \( f_{\bar{n}} \) for some \( \bar{n} \), which triggers \( f \not\equiv n_v \); these values are propagated to the output node, so that \( f_{\bar{n}} \equiv v_x \) indicates that the circuit is falsified.

This is encoded in a set of clauses \( s \) as follows. For every \( i \in [k] \), let \( s_i \) be the least set that for all \( j \in [k] \) \( f_{\bar{n}} \not\equiv n_v \) \( f \not\equiv n_v \) for all \( x \in X \), and \( \forall x \in X, f_{\bar{n}} \not\equiv n_v \). For existential let \( \mathcal{F}_{\bar{n}} \not\equiv v_x \). Now let \( s \) be the least set such that \( s \supseteq s_i \) for all \( i \in [l] \) \( s \subseteq s \) and \( \forall x \in X, f_{\bar{n}} \not\equiv v_x \). For all \( x \in X \), \( f_{\bar{n}} \not\equiv v_x \). This is encoded in a set of clauses \( s \) as follows. For every \( i \in [l] \), let \( s_i \) be the least set that contains \( f_{\bar{n}} \not\equiv v_x \) \( f \not\equiv n_v \) \( f \) for every \( x \in X \), and \( \forall x \in X, f_{\bar{n}} \not\equiv v_x \). Let \( s \) be the least set such that \( s \supseteq s_i \) for all \( i \in [l] \), \( f_{\bar{n}} \not\equiv v_x \) for every \( i \), \( j \in [k] \) with \( i \not= j \) and \( x \in X \), and an encoding of the and/or nodes analogous to the one from the proof of

We then prove that every \( s \subseteq X \) with \( |\mathcal{S}| = k \) falsifies \( C \) iff for all \( n_1, \ldots, n_k \in \{ n_v \} \), \( s \subseteq \cup \{ t_1, t_2, \cdots, t_l \} \), \( t(n_1, \ldots, t(l) = n_l, t(n)) \} \approx f_{\bar{n}} \) by induction on the depth of \( C \). The right-hand side can be rewritten to match Lemma 7 using \( L_1 \) and \( f_{\bar{n}} \not\equiv v_x \), and thus \textsc{fpt-reduces} to Limited Belief Reasoning by Lemmas 7 and 4, which gives us \textsc{co-w[p]-hardness}.

Finally, the only remaining case is when Limited Belief Reasoning is parameterized by both \( |\mathcal{F}| \) and \( |\mathcal{N}| \):

\textbf{Proposition 14 Limited Belief Reasoning with parameters \( |\mathcal{F}| \) and \( |\mathcal{N}| \) is in \textsc{fpt}. This also holds when \( |\mathcal{N}| \) is constant.}

\textit{Proof.} The decision procedure runs in time \( (|\mathcal{F}| \cdot (|\mathcal{N}| + k)^k \cdot p(m) \). By Lemma 2 we can estimate \( k \leq |\mathcal{F}| \).\footnote{\textsc{fpt}-completeness.}

\section{Conclusion}

We have analyzed the complexity of Limited Belief Reasoning. While tractable for constant belief levels, the complexity jumps to \textsc{pspace}-complete in the general case. Using parameterized complexity theory, we showed how parameterized versions of the problem populate the space between these two extremes.
We believe our findings are relevant to the future development of the theory of limited belief. In particular, the insight that the limited belief level can actually increase the computational cost should be considered in future versions.

In light of PSPACE-completeness, one might implement a reasoning system using an off-the-shelf QBF-solver. Also, limited belief may be suitable as a modeling language for other problems in PSPACE.

So far, we have only considered Limited Belief Reasoning without first-order quantification; lifting this restriction would be a natural next step. Moreover, additional parameters could be studied, for example, parameters exploiting the structure of the knowledge base and the query, like backdoors [Gaspers and Szeider, 2012].

Another interesting question is whether our findings carry over to other approaches to resource-bounded reasoning using a similar splitting technique [D’Agostino, 2015].

References


