Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17)

Strong Inconsistency in Nonmonotonic Reasoning

Gerhard Brewka[1] and **Matthias Thimm**[2] and **Markus Ulbricht**[1]
[1]Department of Computer Science, Leipzig University, Germany
[2]Institute for Web Science and Technologies (WeST), University of Koblenz-Landau, Germany
{brewka,mulbricht}@informatik.uni-leipzig.de thimm@uni-koblenz.de

Abstract

Minimal inconsistent subsets of knowledge bases play an important role in classical logics, most notably for repair and inconsistency measurement. It turns out that for nonmonotonic reasoning a stronger notion is needed. In this paper we develop such a notion, called *strong inconsistency*. We show that—in an arbitrary logic, monotonic or not—minimal strongly inconsistent subsets play the same role as minimal inconsistent subsets in classical reasoning. In particular, we show that the well-known classical duality between hitting sets of minimal inconsistent subsets and maximal consistent subsets generalises to arbitrary logics if the strong notion of inconsistency is used. We investigate the complexity of various related reasoning problems and present a generic algorithm for computing minimal strongly inconsistent subsets of a knowledge base. We also demonstrate the potential of our new notion for applications, focusing on repair and inconsistency measurement.

1 Introduction

Various notions which are highly useful and thus have been studied intensively in classical logic turn out to be of rather limited value when it comes to nonmonotonic reasoning based on formalisms like Reiter's default logic [Reiter, 1980], answer set programming (ASP) [Brewka *et al.*, 2011], or abstract argumentation [Dung, 1995]. An excellent example is the notion of equivalence. In classical logic equivalence is important as it guarantees substitutability: whenever two formulas F and F' are equivalent, that is, possess the same models, and F is a subformula of G, then replacing F by F' in G yields a formula equivalent to G. In nonmonotonic formalisms this is no longer the case. For instance the two logic programs $P_1 = \{c.\}$ and $P_2 = \{c \leftarrow \text{not } b.\}$ have the same (single) stable model $\{c\}$, but substitutability is not given.[1] For example, replacing the first rule in $P_3 = \{c \leftarrow \text{not } b.,\ b.\}$ with the fact "c." changes the semantics of P_3. This observation has led to a body of literature on so-called *strong equivalence*, a more adequate notion of equivalence for nonmonotonic reasoning (see for instance [Lifschitz *et al.*, 2001; Eiter *et al.*, 2005; Oikarinen and Woltran, 2011]).

In this paper we study another notion, namely the notion of minimal inconsistent subsets. Again, this notion is highly interesting for classical, or more generally monotonic logics, at least for the following reasons:

- Diagnosis and repair of knowledge bases: consistency of an inconsistent knowledge base $\mathcal{K}$ can be restored by computing a minimal hitting set of the minimal inconsistent subsets of $\mathcal{K}$ and eliminating the elements of the hitting set. The result will be a maximal consistent subset of $\mathcal{K}$ [Reiter, 1987].
- Inconsistency measures: various prominent numerical measures of the degree of inconsistency of a knowledge base exist in the literature which depend on (the number of) minimal inconsistent subsets, see for instance [Hunter and Konieczny, 2008].

Minimal inconsistent subsets cannot play the same role for nonmonotonic formalisms. Consider the logic program $P_4 = \{a \leftarrow \text{not } a, \text{not } b.,\ b.\}$. The program is consistent and has the stable model $\{b\}$. However, it has an inconsistent subset, namely $\{a \leftarrow \text{not } a, \text{not } b.\}$ which is also a minimal inconsistent subset. Yet, since P_4 is consistent there is no inconsistency to be resolved. Note also that one of the standard assumptions in the literature on inconsistency measurement is that the inconsistency value of a knowledge base $\mathcal{K}$ should be 0 iff $\mathcal{K}$ is consistent. The example thus also shows that the notion of minimal inconsistent subsets is of no use in defining inconsistency measures for nonmonotonic formalisms.

The goal of this paper is to develop a stronger notion of inconsistency. We will introduce so-called *strongly inconsistent* subsets, which generalise the classical notion adequately to the nonmonotonic case, and study the minimal ones among these sets. In particular, we show that our objects of study can indeed play the same role for nonmonotonic reasoning as regular minimal inconsistent subsets for monotonic reasoning.

The paper is organised as follows: since our main results are independent of the actual logic used, we first present in Section 2 an abstract notion of logics which is based on a similar account in the area of multi-context systems [Brewka and Eiter, 2007]. Section 3 then introduces strong inconsistency, proves a generalised duality result between strongly inconsistent and maximal consistent sets, and investigates further properties of our new notion. Computational aspects of

[1]A brief introduction to logic programs is provided in Section 2.

strong inconsistency, including a complexity analysis and a generic algorithm for computing strongly inconsistent subsets, are studied in Section 4. Section 5 discusses applications of our new notion in diagnosis and inconsistency measurement. Section 6 concludes.

Proofs of technical results can be found in an extended version of this paper[2].

2 Preliminaries

We first describe what we mean by a—potentially nonmonotonic—logic in an abstract manner, following the characterisation of logics in [Brewka and Eiter, 2007]. Here a logic is specified by a set KB of knowledge bases, a set BS of belief sets, and an acceptability function $\mathsf{ACC} : \mathsf{KB} \rightarrow 2^{\mathsf{BS}}$. The analysis in this paper assumes that knowledge bases are sets of formulas. Moreover, the distinction between consistent and inconsistent belief sets is crucial. For this reason we extend Brewka and Eiter's characterisation as follows:

Definition 2.1. A logic L is a tuple $L = (\mathsf{WF}, \mathsf{BS}, \mathsf{INC}, \mathsf{ACC})$ where WF is a set of well-formed formulas, BS is a set of belief sets, $\mathsf{INC} \subseteq \mathsf{BS}$ is an upward closed[3] set of inconsistent belief sets, and $\mathsf{ACC} : 2^{\mathsf{WF}} \rightarrow 2^{\mathsf{BS}}$ assigns a collection of belief sets to each subset of WF. A *knowledge base* $\mathcal{K}$ of L is a finite subset of WF. A knowledge base $\mathcal{K}$ is called *inconsistent* iff $\mathsf{ACC}(\mathcal{K}) \subseteq \mathsf{INC}$.

Note that a knowledge base $\mathcal{K}$ can be inconsistent because it has no belief sets, and consistent even if some (but not all) of its belief sets are in INC. We illustrate the generality of the above definition by giving instantiations for propositional logic, answer set programming [Brewka *et al.*, 2011], and abstract argumentation frameworks [Dung, 1995].

Example 2.2. Let A be a set of propositional atoms. A propositional logic L_{P} can be defined as $L_{\mathrm{P}} = (\mathsf{WF}_{\mathrm{P}}, \mathsf{BS}_{\mathrm{P}}, \mathsf{INC}_{\mathrm{P}}, \mathsf{ACC}_{\mathrm{P}})$ where WF_{P} are the well-formed formulas over A, BS_{P} are the deductively closed sets of formulas, $\mathsf{INC}_{\mathrm{P}}$ has WF_{P} as its single element, and $\mathsf{ACC}_{\mathrm{P}}$ assigns to each $\mathcal{K} \subseteq \mathsf{WF}_{\mathrm{P}}$ the set containing its set of theorems.

Example 2.3. Let A be a set of propositional atoms. Extended logic programs under answer set semantics over A is the logic $L_{\mathrm{ASP}} = (\mathsf{WF}_{\mathrm{ASP}}, \mathsf{BS}_{\mathrm{ASP}}, \mathsf{INC}_{\mathrm{ASP}}, \mathsf{ACC}_{\mathrm{ASP}})$ where $\mathsf{WF}_{\mathrm{ASP}}$ is the set of all rules over A, $\mathsf{BS}_{\mathrm{ASP}}$ consists of the sets of literals over A, $\mathsf{INC}_{\mathrm{ASP}}$ are the belief sets containing a complementary pair of literals, and $\mathsf{ACC}_{\mathrm{ASP}}$ assigns to a logic program $P \subseteq \mathsf{WF}_{\mathrm{ASP}}$ the set of all answer sets of P.

Example 2.4. Abstract argumentation frameworks under e. g. stable semantics [Dung, 1995] can be modelled as a logic $L_{\mathrm{AAF}} = (\mathsf{WF}_{\mathrm{AAF}}, \mathsf{BS}_{\mathrm{AAF}}, \mathsf{INC}_{\mathrm{AAF}}, \mathsf{ACC}_{\mathrm{AAF}})$ in the following way: given some set of abstract arguments Arg, the elements of $\mathsf{WF}_{\mathrm{AAF}}$ are either elements of Arg, or pairs of such elements, called attacks, i. e., $\mathsf{WF}_{\mathrm{AAF}} = Arg \cup (Arg \times Arg)$. The former are needed to represent arguments which do not participate in any attack. Belief sets are arbitrary sets of arguments, $\mathsf{INC}_{\mathrm{AAF}}$ is empty, and $\mathsf{ACC}_{\mathrm{AAF}}$ assigns to each knowledge base $\mathsf{AF} \subseteq \mathsf{WF}_{\mathrm{AAF}}$ the stable extensions of the argumentation framework.

[2] `http://mthimm.de/misc/btu_ijcai17_ext.pdf`

[3] S upward closed means $B \in S$ and $B \subseteq B'$ implies $B' \in S$.

As the above examples show, the abstract notion is general enough to model monotonic and nonmonotonic logics.

Definition 2.5. A logic $L = (\mathsf{WF}, \mathsf{BS}, \mathsf{INC}, \mathsf{ACC})$ is weakly monotonic whenever $\mathcal{K} \subseteq \mathcal{K}' \subseteq \mathsf{WF}$ implies

1. if $B' \in \mathsf{ACC}(\mathcal{K}')$ then $B \subseteq B'$ for some $B \in \mathsf{ACC}(\mathcal{K})$.

L is monotonic if $\mathcal{K} \subseteq \mathcal{K}' \subseteq \mathsf{WF}$ in addition implies

2. if $B \in \mathsf{ACC}(\mathcal{K})$ then $B \subseteq B'$ for some $B' \in \mathsf{ACC}(\mathcal{K}')$,

Note that this definition generalises [Brewka and Eiter, 2007] where in addition ACC is required to be unique for monotonic logics. The two conditions are needed to guarantee that both skeptical and credulous inference based on intersection, respectively union of belief sets are monotonic.

It is easy to see that—as expected—propositional logic is monotonic whereas logic programs under the answer set semantics and abstract argumentation frameworks are not.

Lemma 2.6. *Let $L = (\mathsf{WF}, \mathsf{BS}, \mathsf{INC}, \mathsf{ACC})$ be weakly monotonic and $\mathcal{K} \subseteq \mathcal{K}'$. If $\mathcal{K}$ is inconsistent then so is $\mathcal{K}'$.*

We will call a knowledge base (weakly) monotonic whenever its associated logic is. Also, whenever there is no risk of confusion we will leave the actual logic implicit.

Throughout the paper we use examples based on logic programs with two kinds of negation, classical negation $\neg$ and default negation not, under answer set semantics. Such programs consist of rules of the form

$$l_0 \leftarrow l_1, \ldots, l_m, \text{not } l_{m+1}, \ldots, \text{not } l_n. \tag{1}$$

where the l_i are classical literals. For programs without default negation ($m = n$) the unique answer set is the smallest set of literals closed under all rules, where a set is closed under a rule of form (1) iff the head literal l_0 is in the set whenever the body literals $l_1, \ldots, l_m$ are. For a program P with default negation, a set M of literals is an answer set iff M is the answer set of the reduced program P^M obtained from P by (i) deleting rules with not l_j in the body for some $l_j \in M$, and (ii) deleting default negated literals from the remaining rules. P is inconsistent iff all of its answer sets contain a complementary pair of literals. This includes the case where P has no answer sets at all. Readers are referred to [Brewka *et al.*, 2011] for more details.

3 Strong Inconsistency

Let $L = (\mathsf{WF}, \mathsf{BS}, \mathsf{INC}, \mathsf{ACC})$ be a logic and $\mathcal{K} \subseteq \mathsf{WF}$ a knowledge base of L. We will leave L implicit in the rest of this section. We use $I(\mathcal{K})$ to denote the collection of all inconsistent subsets of $\mathcal{K}$. A set $H \in I(\mathcal{K})$ is called *minimal inconsistent* if $H' \subsetneq H$ implies H' is consistent. $I_{min}(\mathcal{K})$ is the set of all minimal inconsistent subsets of $\mathcal{K}$.

A consistent subset H of $\mathcal{K}$ is called *maximal $\mathcal{K}$-consistent* if $H \subsetneq H' \subseteq \mathcal{K}$ implies H' is inconsistent. We let $C(\mathcal{K})$ and $C_{max}(\mathcal{K})$ denote the set of all consistent and maximal $\mathcal{K}$-consistent subsets of $\mathcal{K}$, respectively.

In weakly monotonic logics, whenever a knowledge base $\mathcal{K}$ is inconsistent, then so is $\mathcal{K}'$ for any knowledge base $\mathcal{K} \subseteq \mathcal{K}'$. This follows directly from weak monotonicity and upward-closedness of the inconsistent belief sets. In nonmonotonic logics this property does not hold in general as additional information may resolve inconsistency.

Another property of monotonic logics is a specific duality between minimal inconsistent and maximal consistent sets. For that we need the following notion:

Definition 3.1. Let $\mathcal{M}$ be a set of sets. We call $\mathcal{H}$ a *hitting set* of $\mathcal{M}$ if $\mathcal{H} \cap M \neq \emptyset$ for each $M \in \mathcal{M}$. A hitting set $\mathcal{H}$ of $\mathcal{M}$ is a *minimal hitting set* of $\mathcal{M}$ if $\mathcal{H}' \subsetneq \mathcal{H}$ implies $\mathcal{H}'$ is not a hitting set of $\mathcal{M}$.

In the monotonic case, we have the following duality result (see [Reiter, 1987] for a proof of the first-order case in the setting of diagnosis).

Theorem 3.2 (MinHS duality). *Let $\mathcal{K}$ be a weakly monotonic knowledge base. Then, $\mathcal{H}$ is a minimal hitting set of $I_{min}(\mathcal{K})$ if and only if $\mathcal{K} \setminus \mathcal{H} \in C_{max}(\mathcal{K})$.*

For nonmonotonic logics, this is not true anymore because a consistent knowledge base may contain inconsistent subsets. In order to generalise Theorem 3.2 to the nonmonotonic case, we need a different, stronger notion of inconsistency. Here is the definition of the central notion of this paper:

Definition 3.3. For $H, \mathcal{K} \subseteq \mathsf{WF}$ with $H \subseteq \mathcal{K}$, H is called *strongly $\mathcal{K}$-inconsistent* if $H \subseteq H' \subseteq \mathcal{K}$ implies H' is inconsistent. The set H is called *strongly inconsistent* if it is strongly WF-inconsistent.

In other words, a subset of a knowledge base $\mathcal{K}$ is strongly $\mathcal{K}$-inconsistent if all its supersets within the knowledge base $\mathcal{K}$ are inconsistent as well. Note that this is a generalisation of standard inconsistency: both notions coincide in the monotonic case (cf. Proposition 3.5 below).

Definition 3.4. For $H, \mathcal{K} \subseteq \mathsf{WF}$ with $H \subseteq \mathcal{K}$, H is *minimal strongly $\mathcal{K}$-inconsistent* if H is strongly $\mathcal{K}$-inconsistent and $H' \subsetneq H$ implies that H' is not strongly $\mathcal{K}$-inconsistent.

Let $SI(\mathcal{K})$ denote the set of all strongly $\mathcal{K}$-inconsistent subsets of $\mathcal{K}$ and let $SI_{min}(\mathcal{K})$ denote the set of all minimal strongly $\mathcal{K}$-inconsistent subsets of $\mathcal{K}$. The following results are immediate, respectively easy to show:

Proposition 3.5. *Let $\mathcal{K}$ be a knowledge base.*

1. *If $\mathcal{K}$ is weakly monotonic, then $I(\mathcal{K}) = SI(\mathcal{K})$.*
2. *If $\mathcal{K}$ is weakly monotonic, then $I_{min}(\mathcal{K}) = SI_{min}(\mathcal{K})$.*
3. *$\mathcal{K}$ is inconsistent iff $SI(\mathcal{K}) \neq \emptyset$ iff $\mathcal{K} \in SI(\mathcal{K})$.*
4. *If H is strongly $\mathcal{K}$-inconsistent and $H \subseteq \mathcal{K}' \subseteq \mathcal{K}$, then H is strongly $\mathcal{K}'$-inconsistent.*

We are now in a position to present one of our main results.

Theorem 3.6 (Generalised MinHS duality). *Let $\mathcal{K}$ be a knowledge base. Then, $\mathcal{H}$ is a minimal hitting set of $SI_{min}(\mathcal{K})$ if and only if $\mathcal{K} \setminus \mathcal{H} \in C_{max}(\mathcal{K})$.*

Example 3.7. Consider again the logic program $P_4 = \{a \leftarrow \text{not } a, \text{not } b., \ b.\}$ from before. The single (and thus also minimal) inconsistent subset is $H = \{a \leftarrow \text{not } a, \text{not } b.\}$. Since it contains one rule only, it coincides with the minimal hitting set. $P_4 \setminus H$ is consistent, but not maximal consistent as P_4 itself is consistent. Using strong inconsistency leads to the intended result: H is inconsistent, but not strongly P_4-inconsistent. In fact, $SI_{min}(P_4)$ is empty, and so is the minimal hitting set. The single maximal consistent subset is P_4, as stated in Theorem 3.6.

Theorem 3.6 suggests that strong inconsistency is indeed an adequate notion that allows us to generalise results from monotonic logics to arbitrary ones. Even though fixing one part of $\mathcal{K}$ by removing a formula could potentially render another part of $\mathcal{K}$ inconsistent, we can restore consistency as in the monotonic case, using the notion of strong inconsistency.

In the literature on classical inconsistency handling the notion of *free formulas* plays a special role [Hunter and Konieczny, 2008] since such a formula has no influence whatsoever regarding consistency. We will next investigate a similar notion for our general setting.

Definition 3.8. Let $\mathcal{K}$ be a monotonic knowledge base. A formula $\alpha \in \mathcal{K}$ is called *free* if

$$\alpha \in \mathcal{K} \setminus \bigcup_{H \in I_{min}(\mathcal{K})} H = \bigcap_{H \in C_{max}(\mathcal{K})} H. \tag{2}$$

Due to Theorem 3.6, an equality similar to the one in (2) also holds in nonmonotonic logics.

Corollary 3.9. *Let $\mathcal{K}$ be a knowledge base. Then*

$$\mathcal{K} \setminus \bigcup_{H \in SI_{min}(\mathcal{K})} H = \bigcap_{H \in C_{max}(\mathcal{K})} H.$$

Consider a monotonic knowledge base $\mathcal{K}$. Since a free formula $\alpha \in \mathcal{K}$ is contained in any maximal consistent set $H \subseteq \mathcal{K}$, we see that for a free formula α the following implication holds.

$$\forall H \subseteq \mathcal{K} : \ H \in C(\mathcal{K}) \Rightarrow H \cup \{\alpha\} \in C(\mathcal{K}) \tag{3}$$

However, in a nonmonotonic framework, (3) does not necessarily mean that α is irrelevant regarding consistency of $\mathcal{K}$. For example, $\mathcal{K}$ could be consistent while $\mathcal{K} \setminus \{\alpha\}$ is not. Hence, in order to obtain a similar notion of free formulas we need to strengthen (3).

Definition 3.10. Let $\mathcal{K}$ be a knowledge base. A formula $\alpha \in \mathcal{K}$ is called *neutral* if it satisfies

$$\forall H \subseteq \mathcal{K} : \ H \in C(\mathcal{K}) \Leftrightarrow H \cup \{\alpha\} \in C(\mathcal{K}). \tag{4}$$

A formula $\alpha \in \mathcal{K}$ is called *consistency restoring* if it satisfies (3), but not (4). The neutral and the consistency restoring formulas in $\mathcal{K}$ are denoted $Ntr(\mathcal{K})$ and $Res(\mathcal{K})$, respectively.

Proposition 3.11. *If $\mathcal{K}$ is weakly monotonic, then* (3) *and* (4) *coincide and hence, $Ntr(\mathcal{K}) = Free(\mathcal{K})$ and $Res(\mathcal{K}) = \emptyset$.*

Proposition 3.12. *Let $\mathcal{K}$ be a knowledge base. Then*

$$Ntr(\mathcal{K}) \cup Res(\mathcal{K}) \subseteq \bigcap_{H \in C_{max}(\mathcal{K})} H.$$

In the Introduction we mentioned equivalence as another notion that has been strengthened for nonmonotonic logics. We conclude this section with a connection between strong equivalence and strong inconsistency. Strong equivalence, mainly studied in logic programming and argumentation, can be generalised to arbitrary logics in the following way: let $L = (\mathsf{WF}, \mathsf{BS}, \mathsf{INC}, \mathsf{ACC})$ be a logic. The knowledge bases $\mathcal{K}$ and $\mathcal{K}'$ are strongly equivalent iff $\mathsf{ACC}(\mathcal{K} \cup H) = \mathsf{ACC}(\mathcal{K}' \cup H)$ for each $H \subseteq \mathsf{WF}$. We obtain the following:

Proposition 3.13. *Let $\mathcal{K}$, $\mathcal{K}'$ and H be knowledge bases. If $\mathcal{K}$ and $\mathcal{K}'$ are strongly equivalent, then $\mathcal{K}$ is strongly $\mathcal{K} \cup H$-inconsistent iff $\mathcal{K}'$ is strongly $\mathcal{K}' \cup H$-inconsistent.*

The result does not hold if equivalence is used rather than strong equivalence.

4 Computational Complexity

Theorem 3.6 suggests that strong $\mathcal{K}$-inconsistency (Definition 3.3) naturally generalizes inconsistency to nonmonotonic frameworks. However, this notion requires consideration of all supersets of a given set, which is apparently more involved than considering inconsistency in monotonic logics. So, we are interested in the computational complexity of deciding (minimal) strong $\mathcal{K}$-inconsistency and in particular the difference between monotonic and nonmonotonic logics.

We assume the reader to be familiar with the classes Σ_m^p and Π_m^p, $m \geq 0$, of the polynomial hierarchy. We also make use of the classes D_m^p, which are the classes of languages that are intersections of a language in Σ_m^p and a language in Π_m^p [Papadimitriou, 1994].

In order to assess the computational complexity of deciding (minimal) strong inconsistency, we compare it to the classical setting: in [Papadimitriou and Wolfe, 1988], it has been shown that Minimal Unsatisfiability (MU) is D_1^p-complete. MU is the following problem: "Given a propositional formula ϕ in CNF, is it true that it is unsatisfiable, but removing an arbitrary clause renders it satisfiable?" Our first observation in this section is a generalisation of this result to higher levels of the polynomial hierarchy.

A *quantified Boolean formula* (QBF) Φ is a formula

$$\Phi = Q_1 X_1 \ldots Q_m X_m \, \phi$$

with quantifiers $Q_1, \ldots, Q_m \in \{\forall, \exists\}$, pair-wise disjoint sets of variables $X_1, \ldots, X_m$, and a propositional formula ϕ over the variables $X_1 \cup \ldots \cup X_m$. A QBF Φ is *true* if ϕ evaluates to true with respect to the quantifiers, e. g., $\forall x_1 \exists x_2 (x_1 \vee \neg x_2)$ is true as for every truth value of x_1 one can find a truth value of x_2 such that $x_1 \vee \neg x_2$ evaluates to true. A QBF Φ is in *prenex normal form* if the quantifiers $Q_1, \ldots, Q_m$ alternate between $\forall$ and $\exists$. The problem of deciding whether a QBF Φ with m alternating quantifiers starting with $\exists$ (resp. starting with $\forall$) is true is the canonical Σ_m^p-complete (resp. Π_m^p-complete) problem [Papadimitriou, 1994].

Let now QBF-MU$(Q_1, ..., Q_m)$ be the following problem:

> Given a QBF $\Phi = Q_1 X_1 \ldots Q_m X_m \, \phi$ in prenex normal form with $\phi = C_1 \wedge \ldots \wedge C_r$ and formulas $C_1, \ldots, C_r$, is it true that Φ is false, but removing any conjunct C_k from ϕ renders Φ true?

For this problem, we obtain a similar result as [Papadimitriou and Wolfe, 1988], which, to our knowledge, has not been stated explicitly before.

Theorem 4.1. *If $m \geq 2$, then* QBF-MU$(Q_1, ..., Q_m)$ *is D_m^p-complete.*

Combining Theorem 4.1 and the result in [Papadimitriou and Wolfe, 1988] (i. e., the case where $m = 1$ and $Q_1 = \exists$), one can observe that the case where Φ is of the form $\Phi = \forall X \phi$ is missing. Indeed, it turns out to be easier.

Proposition 4.2. QBF-MU$(\forall)$ *is NP-complete.*

Remark 4.3. We can cast the logic of quantified Boolean formulas into our general logical framework as well. Given quantifiers $Q_1, \ldots, Q_m$ and sets of variables $X_1, \ldots, X_m$, we define a corresponding logic $L = L(Q_1, \ldots, Q_m, X_1, \ldots, X_m) = (\mathsf{WF}, \mathsf{BS}, \mathsf{INC}, \mathsf{ACC})$ as follows: $\mathsf{WF} = \mathsf{WF}(X_1 \cup \ldots \cup X_m)$ is the set of all well-formed Boolean formulas over the atoms in $X_1 \cup \ldots \cup X_m$, $\mathsf{BS} = \{\bot, \top\}$, $\mathsf{INC} = \{\bot\}$ and for a knowledge base $\mathcal{K} = \{C_1, \ldots, C_r\}$ we let $\phi(\mathcal{K}) = C_1 \wedge \ldots \wedge C_r$ and define $\mathsf{ACC} = \mathsf{ACC}(Q_1, \ldots, Q_m)$ via

$$\mathsf{ACC}(\mathcal{K}) = \begin{cases} \{\top\}, & \text{if } Q_1 X_1 \ldots Q_m X_m \, \phi(\mathcal{K}), \\ \{\bot\}, & \text{otherwise.} \end{cases}$$

Now, deciding whether Φ is a "yes" instance of QBF-MU$(Q_1, ..., Q_m)$ corresponds to checking whether $\mathcal{K}$ is minimal (strongly) inconsistent.

We now turn to the general discussion on the computational complexity of problems related to strong inconsistency. For that, we assume an arbitrary but fixed logic $L = (\mathsf{WF}, \mathsf{BS}, \mathsf{INC}, \mathsf{ACC})$ for the remainder of this section. To be able to assess how difficult it is to check whether a subset $H \subseteq \mathcal{K}$ of a knowledge base $\mathcal{K}$ is (minimal) strongly $\mathcal{K}$-inconsistent, we consider checking satisfiability of $\mathcal{K}$ as the basis of our investigation. More precisely, we consider the following decision problems:

$\mathrm{SAT}_{\mathcal{K}}$	**Input**:	$\mathcal{K} \subseteq \mathsf{WF}$
	Output:	TRUE iff $\mathcal{K}$ is consistent
S-INC$_{\mathcal{K}}(H)$	**Input**:	$\mathcal{K} \subseteq \mathsf{WF}$, $H \subseteq \mathcal{K}$
	Output:	TRUE iff $H \in SI(\mathcal{K})$
MIN-S-INC$_{\mathcal{K}}(H)$	**Input**:	$\mathcal{K} \subseteq \mathsf{WF}$, $H \subseteq \mathcal{K}$
	Output:	TRUE iff $H \in SI_{min}(\mathcal{K})$

In other words, $\mathrm{SAT}_{\mathcal{K}}$ is the generalisation of the satisfiability problem in our general logic L, S-INC$_{\mathcal{K}}(H)$ is about deciding whether H is strongly $\mathcal{K}$-inconsistent, and MIN-S-INC$_{\mathcal{K}}(H)$ is about deciding whether H is a minimal strongly $\mathcal{K}$-inconsistent set. If L is monotonic and $\mathrm{SAT}_{\mathcal{K}} \in \mathcal{C}$ for some class $\mathcal{C}$, then S-INC$_{\mathcal{K}}(H)$ is in co-$\mathcal{C}$. However, in a nonmonotonic framework, checking whether a given subset $H \subseteq \mathcal{K}$ is strongly $\mathcal{K}$-inconsistent involves considering all sets H' with $H \subseteq H' \subseteq \mathcal{K}$ and corresponding consistency checks. This may increase computational complexity in some cases, but, interestingly, not always as the following result shows.

Theorem 4.4. *Let $\mathcal{K}$ be a knowledge base. Let $m \geq 1$. If the decision problem* $\mathrm{SAT}_{\mathcal{K}}$ *is in*

(a) Σ_m^p*, then* S-INC$_{\mathcal{K}}(H)$ *is in* Π_m^p*,*

(b) Π_m^p*, then* S-INC$_{\mathcal{K}}(H)$ *is in* Π_{m+1}^p*,*

(c) Π_m^p *and $\mathcal{K}$ is weakly monotonic, then* S-INC$_{\mathcal{K}}(H)$ *is in* Σ_m^p*.*

Theorem 4.1 already showed how difficult MIN-S-INC$_{\mathcal{K}}(H)$ is compared to the decision problem $\mathrm{SAT}_{\mathcal{K}}$ in the generic framework of QBFs (cf. Remark 4.3). As stated in Theorem 4.4, checking strong inconsistency is in general more difficult in nonmonotonic frameworks and we obtain a similar result in the case of MIN-S-INC$_{\mathcal{K}}(H)$. However, the increase of the computational complexity stems from checking the "strong" part in "strong minimal inconsistency" rather than the "minimal" part. For that reason and as the following result shows, moving from the problem S-INC$_{\mathcal{K}}(H)$ to the problem MIN-S-INC$_{\mathcal{K}}(H)$—i. e., additionally asking for minimality—does not involve going up an additional level in the polynomial hierarchy but only moving to the corresponding D^p_m class.

Theorem 4.5. *Let $\mathcal{K}$ be a knowledge base. Let $m \geq 1$. If the decision problem* $\text{SAT}_\mathcal{K}$ *is in*

(a) Σ^p_m*, then* $\text{MIN-S-INC}_\mathcal{K}(H)$ *is in* D^p_m*,*

(b) Π^p_m*, then* $\text{MIN-S-INC}_\mathcal{K}(H)$ *is in* D^p_{m+1}*,*

(c) Π^p_m *and $\mathcal{K}$ is weakly monotonic, then* $\text{MIN-S-INC}_\mathcal{K}(H)$ *is in* D^p_m*.*

In Theorems 4.4 and 4.5 only membership statements are given. Thus, they leave open whether $\text{S-INC}_\mathcal{K}(H)$ and $\text{MIN-S-INC}_\mathcal{K}(H)$ can be strictly less difficult than Π^p_{m+1} and D^p_{m+1}, respectively, if $\mathcal{K}$ is nonmonotonic and $\text{SAT}_\mathcal{K}$ in Π^p_m. However, we obtain completeness for a specific (artificial) nonmonotonic logic, showing that the bounds from both cases (b) from Theorems 4.4 and 4.5 are tight in general.

Theorem 4.6. *There is a logic $L = (\mathsf{WF}, \mathsf{BS}, \mathsf{INC}, \mathsf{ACC})$ such that* $\text{SAT}_\mathcal{K}$ *is in* $\mathsf{coNP} = \Pi^p_1$ *and*

(a) $\text{S-INC}_\mathcal{K}(H)$ *is* Π^p_2*-complete and*

(b) $\text{MIN-S-INC}_\mathcal{K}(H)$ *is* D^p_2*-complete.*

We give two more hardness results for case (a) of Theorem 4.5 for the framework of logic programming. Be reminded that deciding whether a given logic program P is consistent is NP-complete [Eiter and Gottlob, 1995].

Theorem 4.7. *The problem* $\text{MIN-S-INC}_P(H)$ *is* D^p_1*-complete for logic programs.*

As a second example, we consider disjunctive logic programs, i. e., logic programs with rules such as (1) but that may have disjunctions in the head rather than a single literal. Due to [Eiter and Gottlob, 1995], deciding whether a given disjunctive program is consistent is Σ^p_2-complete.

Theorem 4.8. *The problem* $\text{MIN-S-INC}_P(H)$ *is* D^p_2*-complete for disjunctive logic programs.*

To conclude this discussion on computational complexity, we present a generic algorithm for computing $SI(\mathcal{K})$. Algorithm 1 computes strongly $\mathcal{K}$-inconsistent subsets in the order of decreasing cardinality, starting with $\mathcal{K}$. It is based on the observation that a proper subset S of $\mathcal{K}$ can only be strongly $\mathcal{K}$-inconsistent if all subsets of $\mathcal{K}$ which contain one additional element are also strongly $\mathcal{K}$-inconsistent (this property is checked during the computation of New). This additional check presumably reduces the search space in many cases, but a detailed evaluation of this algorithm is left for future work. The algorithm is somewhat reminiscent of the Apriori algorithm for computing frequent sets in data mining [Agrawal and Srikant, 1994], but rather than working bottom up from smaller to bigger sets, it works in the opposite direction. The algorithm can easily be turned into one for $SI_{min}(\mathcal{K})$ by deleting non-minimal elements whenever New is added to H'.

Proposition 4.9. *Algorithm 1 is sound, complete, and has runtime $O(2^n * n * f(n))$ where $f(n)$ is the runtime of an algorithm for checking consistency in the given logic.*

We expect that for specific logics one can do better. For instance, for logic programs without classical negation it is

```
Input: a knowledge base 𝒦
Result: SI(𝒦)
n := |𝒦|; H := ∅; H' := ∅;
if 𝒦 inconsistent then H' := {𝒦};
while H ≠ H' do
    n := n − 1; H := H'; New := ∅ ;
    for each S ∈ H with |S| = n + 1 do
        for each S' ⊆ S with |S'| = n do
            if S' inconsistent and
            S' ∪ {φ} ∈ H for each φ ∈ 𝒦 \ S'
            then New := New ∪ {S'};
        end
    end
    H' := H' ∪ New;
end
return H.
```

Algorithm 1: A generic algorithm for computing $SI(\mathcal{K})$

well-known that inconsistency can only arise if there are certain negative loops in the dependency graph. The analysis of such loops may lead to more direct algorithms. This topic is currently under investigation.

5 Applications

We will now discuss some of the potential applications of strong inconsistency in nonmonotonic reasoning, namely knowledge base repair and inconsistency measurement.

5.1 Diagnosis and Repair

Theorem 3.6 already shows how consistency of a knowledge base $\mathcal{K}$ can be restored by deleting a minimal subset of formulas. As in the classical case, the key is to compute certain inconsistent subsets of the knowledge base. The hitting sets of these subsets then are the candidates for deletion. Unlike in monotonic logics, in the general case one has to compute hitting sets of minimal strongly inconsistent subsets. Our theorem shows that this guarantees minimality of the modification performed on $\mathcal{K}$.

Example 5.1. Consider the following logic program P_5:

$$\begin{array}{lllll} P_5: & a \leftarrow \text{not } b. & b \leftarrow \text{not } c. & a \leftarrow \text{not } d. & e. \\ & c \leftarrow \text{not } a. & & d. & \neg e. \end{array}$$

Minimal inconsistent subsets of P_5 are $H_1 = \{a \leftarrow \text{not } b., b \leftarrow \text{not } c., c \leftarrow \text{not } a.\}$ and $H_2 = \{e., \neg e.\}$. Whereas H_2 is also minimal strongly P_5-inconsistent, H_1 is not as adding "$a \leftarrow$ not d." resolves inconsistency. The second minimal strongly P_5-inconsistent subset is $H_3 = H_1 \cup \{d\}$. Minimal hitting sets consist of one element of H_3 and one of H_2. The program can be repaired by deleting the rules in any of the hitting sets.

It is worth mentioning that in the nonmonotonic case, this is not the only way of repairing a knowledge base, as adding formulas may also lead to consistency. However, deletion-based repair is also important for nonmonotonic knowledge bases for various reasons. First of all, in many cases it is far from clear how to select and justify the added formulas. Secondly, there are situations where modelling errors are more probable than modelling gaps, and where identifying such errors simply is the better option. And finally, there are cases where there is simply no choice as some inconsistencies cannot be repaired by additions alone.

The results of this paper are not only relevant for knowledge base repair, but also for model-based diagnosis of technical systems along the lines of [Reiter, 1987; de Kleer *et al.*, 1992]. In this approach a system description SD is given in terms of first-order logic. SD describes the correct behaviour of a set of components $Comp$ and uses ab predicates for this purpose. The idea is then to identify minimal sets of components C such that $SD \cup Obs \cup \{ab(c) \mid c \in C\}$ is inconsistent. The results of this paper allow us to capture system descriptions expressed in more general logics. All we have to do is replace the inconsistency check with a strong inconsistency check. A more detailed analysis of this generalisation of model-based diagnosis is beyond the scope of this paper and will be discussed elsewhere.

5.2 Measuring Inconsistency

An inconsistency measure Inc is a function that maps knowledge bases to non-negative real numbers. The intuition behind such functions is that larger values indicate severe inconsistencies in the knowledge base and the value 0 indicates minimal inconsistency, i. e., consistency. Different approaches to measuring inconsistency have been proposed in the literature, mostly for classical propositional logic, see [Thimm, 2017] for a recent survey. In this context, a simple but popular approach to measure inconsistency is to take the number of minimal inconsistent subsets [Hunter and Konieczny, 2008], i. e., to define $Inc_{\mathsf{MI}}(\mathcal{K}) = |I_{min}(\mathcal{K})|$ for a classical knowledge base $\mathcal{K}$. This measure already complies with some basic ideas of inconsistency measurement, in particular $Inc_{\mathsf{MI}}(\mathcal{K}) = 0$ iff $\mathcal{K}$ is consistent. By also taking the size and the relationships of minimal inconsistent subsets into account, a wide variety of different inconsistency measures can be defined on top of that idea, see e. g. [Hunter and Konieczny, 2008; Jabbour *et al.*, 2016; Jabbour and Sais, 2016].

Measuring inconsistency in nonmonotonic logics has only recently gained some attention [Ulbricht *et al.*, 2016] and a thorough study is still needed. In this setting, a measure such as Inc_{MI} is not applicable as a consistent nonmonotonic knowledge base $\mathcal{K}$ may contain minimal inconsistent subsets, recall the logic program $P_4 = \{a \leftarrow \text{not } a, \text{not } b., \ b.\}$ from the introduction. However, using our notion of strong inconsistency the wide spectrum of measures based on minimal inconsistent subsets can be lifted to the general case. Here, we only consider the measure Inc_{MI}.

Definition 5.2. Define Inc_{MSI} via $Inc_{\mathsf{MSI}}(\mathcal{K}) = |SI_{min}(\mathcal{K})|$ for every knowledge base $\mathcal{K}$.

If $\mathcal{K}$ is monotonic then $Inc_{\mathsf{MSI}}(\mathcal{K}) = Inc_{\mathsf{MI}}(\mathcal{K})$ due to Proposition 3.5, item 2. So the measure Inc_{MSI} faithfully generalises Inc_{MI} to all kinds of logics.

Example 5.3. For the logic program $P_4 = \{a \leftarrow \text{not } a, \text{not } b., b.\}$ we obtain $Inc_{\mathsf{MSI}}(P_4) = 0$, despite the fact that P_4 contains a (classical) minimal inconsistent subset. For $P_5 = \{a., \neg a., b \leftarrow \text{not } b.\}$ we have $Inc_{\mathsf{MSI}}(P_5) = 2$.

The field of inconsistency measurement is driven by rationality postulates, i. e., the development of general properties that should hold for an inconsistency measure, cf. [Thimm, 2017]. Many of them specify desirable behaviour in terms of minimal inconsistent subsets and can thus easily be lifted to the general case. The following result shows the compliance of our generalised measure with some important properties.

Theorem 5.4. *Let $\mathcal{K}$ be a (monotonic or nonmonotonic) knowledge base.*

Consistency $Inc_{\mathsf{MSI}}(\mathcal{K}) = 0$ *if and only if $\mathcal{K}$ is consistent.*

Independence *If $\alpha \in Ntr(\mathcal{K})$ then $Inc_{\mathsf{MSI}}(\mathcal{K}) = Inc_{\mathsf{MSI}}(\mathcal{K} \setminus \{\alpha\})$.*

Separability *If $SI_{min}(\mathcal{K}_1 \cup \mathcal{K}_2) = SI_{min}(\mathcal{K}_1) \cup SI_{min}(\mathcal{K}_2)$ and $SI_{min}(\mathcal{K}_1) \cap SI_{min}(\mathcal{K}_2) = \emptyset$ then $Inc_{\mathsf{MSI}}(\mathcal{K}_1 \cup \mathcal{K}_2) = Inc_{\mathsf{MSI}}(\mathcal{K}_1) + Inc_{\mathsf{MSI}}(\mathcal{K}_2)$.*

The measure Inc_{MSI} violates one important property though, which is usually demanded for classical measures: the *monotonicity postulate*. This postulate requires $Inc(\mathcal{K}) \leq Inc(\mathcal{K}')$ whenever $\mathcal{K} \subseteq \mathcal{K}'$ and formalises the intuition that inconsistency can only increase when adding new information. However, this intuition is inadequate for nonmonotonic logics as the addition of new information may resolve inconsistencies. Therefore, satisfaction of the *monotonicity postulate* is indeed *not* desirable in general, see [Ulbricht *et al.*, 2016] for a discussion on this topic.

In the same vein, other approaches that utilise minimal inconsistent sets for inconsistency measurement [Hunter and Konieczny, 2008; Jabbour *et al.*, 2016; Jabbour and Sais, 2016] can also be lifted to the general case. We leave a deeper investigation of this topic for future work.

6 Conclusions

In this paper we studied inconsistency in an abstract setting covering arbitrary logics, including nonmonotonic ones. We showed that in the general case the standard notion of inconsistency is unable to play the same role it does in monotonic reasoning. Our main contribution is the identification of an adequate strengthening of inconsistency. One of our main results shows that the duality between minimal inconsistent subsets and maximal consistent subsets of a knowledge base, which does not hold for nonmonotonic logics, can be restored when minimal strongly inconsistent subsets are used. We established encouraging complexity results for problems related to strong inconsistency, presented a generic algorithm for computing (minimal) strongly inconsistent subsets, and demonstrated possible applications of our new notion in diagnosis/repair and inconsistency measurement.

Although there is a rich literature on inconsistency handling (see [Bertossi *et al.*, 2005] for an introduction and [Bienvenu *et al.*, 2016] for a recent approach), we are not aware of any work addressing the issues we studied in this paper.

In future work we will investigate algorithms for specific nonmonotonic logics, elaborate the use of strong inconsistency in model-based diagnosis and continue the study of inconsistency measures based on strong inconsistency.

Acknowledgements

This work was partially funded by DFG (Research Training Group 1763; project BR 1817/7-2).

References

[Agrawal and Srikant, 1994] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases*, pages 487–499, 1994.

[Bertossi *et al.*, 2005] Leopoldo E. Bertossi, Anthony Hunter, and Torsten Schaub, editors. *Inconsistency Tolerance*, volume 3300 of *Lecture Notes in Computer Science*. Springer, 2005.

[Bienvenu *et al.*, 2016] Meghyn Bienvenu, Camille Bourgaux, and François Goasdoué. Query-driven repairing of inconsistent DL-lite knowledge bases. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 957–964, 2016.

[Brewka and Eiter, 2007] Gerhard Brewka and Thomas Eiter. Equilibria in heterogeneous nonmonotonic multi-context systems. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*, pages 385–390, 2007.

[Brewka *et al.*, 2011] Gerhard Brewka, Thomas Eiter, and Miroslaw Truszczynski. Answer set programming at a glance. *Communications of the ACM*, 54(12):92–103, 2011.

[de Kleer *et al.*, 1992] Johan de Kleer, Alan K. Mackworth, and Raymond Reiter. Characterizing diagnoses and systems. *Artificial Intelligence*, 56(2-3):197–222, 1992.

[Dung, 1995] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–358, 1995.

[Eiter and Gottlob, 1995] Thomas Eiter and Georg Gottlob. On the computational cost of disjunctive logic programming: Propositional case. *Annals of Mathematics and Artificial Intelligence*, 15(3-4):289–323, 1995.

[Eiter *et al.*, 2005] Thomas Eiter, Michael Fink, Hans Tompits, and Stefan Woltran. Strong and uniform equivalence in answer-set programming: Characterizations and complexity results for the non-ground case. In *Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, 2005, Pittsburgh, Pennsylvania, USA*, pages 695–700, 2005.

[Hunter and Konieczny, 2008] Anthony Hunter and Sébastien Konieczny. Measuring inconsistency through minimal inconsistent sets. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Eleventh International Conference, KR 2008, Sydney, Australia, September 16-19, 2008*, pages 358–366, 2008.

[Jabbour and Sais, 2016] Said Jabbour and Lakhdar Sais. Exploiting MUS Structure to Measure Inconsistency of Knowledge Bases. In *Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI'16)*, 2016.

[Jabbour *et al.*, 2016] Saïd Jabbour, Yue Ma, Badran Raddaoui, Lakhdar Sais, and Yakoub Salhi. A MIS partition based framework for measuring inconsistency. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016.*, pages 84–93, 2016.

[Lifschitz *et al.*, 2001] Vladimir Lifschitz, David Pearce, and Agustín Valverde. Strongly equivalent logic programs. *ACM Transactions on Computational Logic*, 2(4):526–541, 2001.

[Oikarinen and Woltran, 2011] Emilia Oikarinen and Stefan Woltran. Characterizing strong equivalence for argumentation frameworks. *Artificial Intelligence*, 175(14-15):1985–2009, 2011.

[Papadimitriou and Wolfe, 1988] Christos H Papadimitriou and David Wolfe. The complexity of facets resolved. *Journal of Computer and System Sciences*, 37(1):2–13, 1988.

[Papadimitriou, 1994] Christos Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[Reiter, 1980] Raymond Reiter. A logic for default reasoning. *Artificial Intelligence*, 13(1-2):81–132, 1980.

[Reiter, 1987] Raymond Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95, 1987.

[Thimm, 2017] Matthias Thimm. On the compliance of rationality postulates for inconsistency measures: A more or less complete picture. *Künstliche Intelligenz*, 31(1):31–39, 2017.

[Ulbricht *et al.*, 2016] Markus Ulbricht, Matthias Thimm, and Gerhard Brewka. Measuring inconsistency in answer set programs. In *Logics in Artificial Intelligence - 15th European Conference, JELIA 2016, Larnaca, Cyprus, November 9-11, 2016, Proceedings*, pages 577–583, 2016.