Computational Complexity and Anytime Algorithm for Inconsistency Measurement[*]

Yue Ma[1], Guilin Qi[2], Guohui Xiao[3,5], Pascal Hitzler[4], and Zuoquan Lin[3]

[1] (Institute LIPN, Université Paris 13-CNRS, France)

[2] (School of Computer Science and Engineering, Southeast University, Nanjing 210096, China)

[3] (Department of Information Science, Peking University, Beijing 100871, China)

[4] (Kno.e.sis Center, Wright State University, Dayton, OH, USA)

[5] (Institut für Informationssysteme, Technische Universität Wien, Austria)

Abstract   Measuring inconsistency degrees of inconsistent knowledge bases is an important problem as it provides context information for facilitating inconsistency handling. Many methods have been proposed to solve this problem and a main class of them is based on some kind of paraconsistent semantics. In this paper, we consider the computational aspects of inconsistency degrees of propositional knowledge bases under 4-valued semantics. We first give a complete analysis of the computational complexity of computing inconsistency degrees. As it turns out that computing the exact inconsistency degree is intractable, we then propose an anytime algorithm that provides tractable approximations of the inconsistency degree from above and below. We show that our algorithm satisfies some desirable properties and give experimental results of our implementation of the algorithm.

Key words: knowledge representation; inconsistency measurement; multi-valued logic; computational complexity; algorithm

1   Introduction

Inconsistency handling is one of the central problems in the field of knowledge representation. Recently, there is an increasing interest in quantifying inconsistency in an inconsistent knowledge base. This is because it is not fine-grained enough to simply say that two inconsistent knowledge bases contain the same amount of inconsistency. Indeed, it has been shown that analyzing inconsistency is helpful to decide how to act on inconsistency[1], i.e. whether to ignore it or to resolve it. Furthermore, measuring inconsistency in a knowledge base can provide some context information which can

be used to resolve inconsistency[2−4], and proves useful in different scenarios such as Software Engineering[5].

Different approaches to measuring inconsistency are based on different views of atomic inconsistency[3]. Syntactic ones put atomicity to formulae, such as taking maximal consistent subsets of formulae[6] or minimal inconsistent sets[7]. Semantic ones put atomicity to propositional letters, such as considering the conflicting propositional letters based on some kind of paraconsistent model [2,3,8−10]. In this paper, we focus on the computational aspect of a 4-valued semantics based inconsistency degree which is among the latter view.

The main contributions of this paper are two-folded. One is to give a complete study of the computation complexity of the decision and functional problems related to measuring inconsistency degree. We show that computing exact inconsistency degrees is a computational problem of high complexity (Θ_2^p-complete). To conquer such a high complexity in computation, we present an anytime algorithm that provides tractable approximations of the inconsistency degree from above and below, by computing *the lower* and *upper bounds*. We show that our algorithm satisfies some desirable properties. Furthermore, we give some experimental explanations of the algorithm. Compared to many existing work on measuring inconsistency, our work complements them in that (1) it analyzes the complexity issues of computing the inconsistency degree and that (2) it attempts to alleviate the intractability of computing the exact inconsistency degree for full propositional logic by approximating it from above and from below in an anytime manner. Our results show that the computation of approximating inconsistency degree can be done tractable; and can be performed to full propositional knowledge bases, unlike the restriction to CNF for designing a tractable paraconsistent reasoning under the Quasi-Classical semantics[11].

The paper is structured as follows. Section 2 gives a discussion of related work. Then after recalling some preliminaries on Belnap's four-valued semantics and knowledge of inconsistency degree in Section 3, we give the complexity analysis of problems of computing inconsistency degree in Section 4. We deal in turn with an approach to approximating inconsistency degree and the corresponding anytime algorithm in Section 5. The implementation of the anytime algorithm is given in Section 6. Finally we conclude the work in Section 7.

2　Related Work

Most effort has been directed at theoretical accounts of inconsistency measures, i.e. its definitions, properties, and possible applications. But few papers focus on the computational aspect of inconsistency degree. Among the syntactic approaches, Ref.[6] shows the possibility to compute inconsistency degrees using the *simplex method*. Among the semantics methods, Refs.[13, 14] and Ref.[10] provide algorithms for computing inconsistency degrees that can be implemented. The algorithm in Ref.[10] only deals with KBs consisting of first-order formulas in the form $Q_1 x_1, ..., Q_n x_n. \bigwedge_i (P_i(t_1, \ldots, t_{m_i}) \wedge \neg P_i(t_1, \ldots, t_{m_i}))$, where $Q_1, \ldots, Q_n$ are universal or existential quantifiers. In Ref.[13], an algorithm is proposed for full FOL logic. Although it can be applied to measure inconsistency in propositional logic, its computational complexity is too high to be used in general cases. In Ref.[14], approximating inconsistency degrees are defined but without detailed study of an anytime algorithm.

The anytime algorithm proposed in this paper for computing approximating inconsistency degrees can avoid these shortcomings.

Although our algorithm is inspired by the algorithms given in Refs.[13, 14], it is significantly different from the existing ones. Firstly, this paper develops the work in Ref.[14] by an anytime algorithm which can return approximating inconsistency degrees in tractable time. We show that for some special knowledge bases, this algorithm will return their exact inconsistency degree in polynomial time. In contrast, the algorithm in Ref.[13] is based on a reduction to hard SAT instances, which makes it inherently intractable. Secondly, ours is designed towards obtaining an approximation with guaranteed lower and upper bounds that gradually converge to the exact solution. The approximations have a meaningful sense in terms of *bounding models*. Thirdly, based on the monotonicity of *S-4 semantics*, we implement a new truncation strategy to limit the search space for better polynomial time approximations. We also present the preliminary evaluation results of the implementation of the algorithm. Our evaluation results show our algorithm outperforms that given in Ref.[13] and develops the results given in Ref.[14], which in all show that the approximating values are reasonable to replace the exact inconsistency degree.

3   Preliminaries

Let $\mathcal{P}$ be a countable set of propositional letters. We concentrate on the classical propositional language formed by the usual Boolean connectives $\wedge$ (conjunction), $\vee$ (disjunction), $\rightarrow$ (implication), and $\neg$ (negation). A propositional knowledge base K consists of a finite set of formulae over that language. We use $Var(K)$ for the set of propositional letters used in K and $|S|$ for the cardinality of S for any set S.

Next we give a brief introduction on Belnap's four-valued (4-valued) semantics (See to Appendix section of this paper for more details that are used in the proofs). Compared to two truth values used by classical semantics, the set of truth values for four-valued semantics[15,16] contains four elements: *true, false, unknown* and *both*, written by t, f, N, B, respectively. The truth value B stands for contradictory information, hence four-valued logic leads itself to dealing with inconsistencies. The four truth values together with the ordering $\preceq$ defined below form a lattice, denoted by $\mathsf{FOUR} = (\{t, f, B, N\}, \preceq)$: $f \preceq N \preceq t, f \preceq B \preceq t, N \npreceq B, B \npreceq N$. The four-valued semantics of connectives $\vee, \wedge$ are defined according to the upper and lower bounds of two elements based on the ordering $\preceq$, respectively, and the operator $\neg$ is defined as $\neg t = f, \neg f = t, \neg B = B$, and $\neg N = N$. The designated set of FOUR is $\{t, B\}$. So a four-valued interpretation $\mathfrak{I}$ is a *4-model* of a knowledge base K if and only if for each formula $\phi \in K$, $\phi^{\mathfrak{I}} \in \{t, B\}$. A knowledge base which has a 4-model is called *4-valued satisfiable*. A knowledge base K 4-valued entails a formula φ, written $K \models_4 \varphi$, if and only if each 4-model of K is a 4-model of φ.

Every propositional knowledge base containing only connectives from $\{\vee, \wedge, \neg, \rightarrow\}$ has a 4-model which assigns B to each propositional letter[16]. Four-valued entailment can be reduced to the classical entailment[17]. We write K for a knowledge base, and $\mathcal{M}_4(K)$ for the set of 4-models of K throughout this paper. Four-valued semantics provides a novel way to define inconsistency measurements[1].

Definition 1.   Let $\mathfrak{I}$ be a four-valued model of K. The *inconsistency degree of K with respect to $\mathfrak{I}$*, denoted $\mathrm{Inc}_{\mathfrak{I}}(K)$, is a value in $[0, 1]$ defined as $\mathrm{Inc}_{\mathfrak{I}}(K) =$

$\frac{|Conflict(\mathfrak{I},K)|}{|Var(K)|}$, where $Conflict(\mathfrak{I}, K) = \{p \mid p \in Var(K), p^{\mathfrak{I}} = B\}$.

That is, the inconsistency degree of K w.r.t. $\mathfrak{I}$ is the ratio of the number of conflicting propositional letters under $\mathfrak{I}$ divided by the amount of all propositional letters used in K. It measures to what extent a given knowledge base K contains inconsistencies with respect to its 4-model $\mathfrak{I}$. Preferred models defined below are used to define inconsistency degrees and especially useful to explain our approximating algorithm later.

In Ref.[1], instead of $\text{Inc}_{\mathfrak{I}}(K)$, the *concordance degree* of a knowledge base is defined as $1 - \frac{|Conflict(I,K)|}{|Var(K)|}$, denoted $\text{Concordance}_{\mathfrak{I}}(K)$. It is clear that $\text{Inc}_{\mathfrak{I}}(K) = 1 - \text{Concordance}_{\mathfrak{I}}(K)$. So all the results we get in this paper for $\text{Inc}_{\mathfrak{I}}(K)$ can be easily applied to the concordance degree in [1].

Definition 2 (Preferred Models).   The set of *preferred models,* written $PreferModel(K)$, is defined as $PreferModel(K) = \{\mathfrak{I} \mid \forall \mathfrak{I}' \in \mathcal{M}_4(K), \text{Inc}_{\mathfrak{I}}(K) \leqslant \text{Inc}_{\mathfrak{I}'}(K)\}$.

By this definition and the fact that every knowledge base K containing only connectives from $\{\vee, \wedge, \neg, \rightarrow\}$ has 4-models, the inequation $PreferModel(K) \neq \emptyset$ always holds, and the inconsistency degree of K with respect to two preferred models are equal.

Definition 2 (Inconsistency Degree).   The *inconsistency degree of K*, denoted by $ID(K)$, is defined as the value $Inc_{\mathfrak{I}}(K)$, where $\mathfrak{I} \in PreferModels(K)$.

Example 1.   Let $K = \{p, \neg p \vee q, \neg q, r\}$. Consider two 4-valued models $\mathfrak{I}_1$ and $\mathfrak{I}_2$ of K with $p^{\mathfrak{I}_1} = t, q^{\mathfrak{I}_1} = B, r^{\mathfrak{I}_1} = t$; and $p^{\mathfrak{I}_2} = B, q^{\mathfrak{I}_2} = B, r^{\mathfrak{I}_2} = t$. We have $\text{Inc}_{\mathfrak{I}_1}(K) = \frac{1}{3}$, while $\text{Inc}_{\mathfrak{I}_2}(K) = \frac{2}{3}$. Moreover, $\mathfrak{I}_1$ is a preferred model of K because there is no other 4-model $\mathfrak{I}'$ of K such that $\text{Inc}_{\mathfrak{I}'}(K) < \text{Inc}_{\mathfrak{I}_1}(K)$. Then $ID(K) = \frac{1}{3}$.

One way to compute inconsistency degree is to recast the algorithm proposed in Ref.[13] to propositional knowledge bases, where *S-4 semantics* defined as follows is used:

Definition 4 (S-4 Model).   For any given set $S \subseteq Var(K)$, an interpretation $\mathfrak{I}$ is called an S-4 model of K if and only if $\mathfrak{I} \in \mathcal{M}_4(K)$ and satisfies the following condition:

$$\mathfrak{I}(p) \in \begin{cases} \{B\} & \text{if } p \in Var(K) \setminus S, \\ \{N, t, f\} & \text{if } p \in S. \end{cases}$$

That is, $\mathfrak{I}$ is an S-4 model of K if it is a 4-valued model of K which assigns the propositional letters not in S to the contradictory truth value, while it assigns others to non-contradictory truth values.

For a given $S \subseteq Var(K)$, the knowledge base K is called S-4 *unsatisfiable* iff. it has no S-4 model. Let φ be a formula and $Var(\{\varphi\}) \subseteq Var(K)$. φ is S-4 *entailed* by K, written $K \models_S^4 \varphi$, iff. each S-4 model of K is an S-4 model of φ. Obviously, $K \models_S^4 f$ if and only if K has no S-4 model, where f is the truth value symbol in FOUR.

Theorem 1[14]**.**   For any KB K, we have $ID(K) = 1 - A/|Var(K)|$, where $A = \max\{|S| : S \subseteq Var(K), K \text{ is } S\text{-4 satisfiable}\}$.

Theorem 1 shows that the computation of $ID(K)$ can be reduced to the problem of computing the maximal cardinality of subsets S of $Var(K)$ such that K is S-4 satisfiable.

4 Computational Complexities

Apart from any particular algorithm, let us study the computational complexity of the inconsistency degree to see how hard the problem itself is. First we define the following computation problems related to inconsistency degrees:

- $ID_{\leqslant d}$ (resp. $ID_{<d}$, $ID_{\geqslant d}$, $ID_{>d}$): Given a propositional knowledge base K and a number $d \in [0, 1]$, is $ID(K) \leqslant d$ (resp. $ID(K) < d$, $ID(K) \geqslant d$, $ID(K) > d$)?

- EXACT-ID: Given a propositional knowledge base K and a number $d \in [0, 1]$, is $ID(K) = d$?

- ID: Given a propositional knowledge base K, what is the value of $ID(K)$?

Obviously, we have two trivial instances of these decision problems $ID_{\leqslant}1$ and $ID_{\geqslant}0$ with the answer "yes"; And another two trivial instances $ID_{<0}$ and $ID_{>1}$ with the answer "no".

In more general cases, the complexities of these computational problems are indicated by following theorems.

Theorem 2. $ID_{\leqslant d}$ and $ID_{<d}$ are **NP**-complete; $ID_{\geqslant d}$ and $ID_{>d}$ are **coNP**-complete.

Proof: We prove these results separately as follows:

$ID_{\leqslant d}$ is NP-complete:

The membership to NP is achieved by the following non-deterministic algorithm:

1. Guess a 4-valued interpretation $\mathfrak{I}$ over $Var(K)$;

2. Check that $\mathfrak{I}$ is a 4-model of K and $\frac{|Conflict(\mathfrak{I})|}{|Var(K)|} \leqslant M$, which can be done in deterministic polynomial time.

The hardness to NP comes from the following reduction from checking the satisfiability of K under classical 2-valued semantics, which is known to NP-complete, to this problem. The reduction is that K is 2-valued satisfiable if and only if $ID(K) \leqslant 0$ which is obvious by the definition of inconsistency degree.

$ID_{<d}$ is NP-complete:

Similarly to the case of $ID_{\leqslant d}$, the membership to **NP** holds obviously. The hardness to **NP** holds by the reduction that K is 2-valued satisfiable if and only if $ID(K) < \frac{1}{2|Var|}$. This is because, by the definition of $ID(K)$, the smallest value of $ID(K)$ for an inconsistent knowledge base is $\frac{1}{|Var|}$.

$ID_{\geqslant d}$ and $ID_{>d}$ are coNP-complete:

This is because that $ID_{\geqslant d}$ is the complementary problem of $ID_{<d}$ and $ID_{>d}$ is the complementary problem of $ID_{\leqslant d}$.

Theorem 3. EXACT-ID is **DP**-complete[1].

Proof: To show that it is in **DP**, we have to exhibit two languages $L_1 \in$ **NP** and $L_2 \in$ **coNP** such that the set of all "yes" instances of EXACT-ID is $L_1 \cap L_2$. This is easy by setting $L_1 = \{K \mid ID(K) \leqslant M\}$ and $L_2 = \{K \mid ID(K) \geqslant M\}$.

---

[1] A language L is in the class **DP** [18] iff there are two languages $L_1 \in$ **NP** and $L_2 \in$ **coNP** such that $L = L_1 \cap L_2$.

To show the completeness, let $L = L_1 \cap L_2$ be any language in **DP**. We have to show that L can be reduced to EXACT-ID. To this end, recall that $ID_\leqslant$ is **NP**-complete and $ID_\geqslant$ is co**NP**-complete, that is, there is a reduction R_1 from L_1 to $ID_\leqslant$ and a reduction R_2 from L_2 to $ID_\geqslant$. Therefore, the reduction R from L to EXACT-ID can be defined as follows, for any input x: $R(x) = (R_1(x), R_2(x))$. We have that $R(x)$ is a "yes" instance of EXACT-ID if and only if $R_1(x)$ is a "yes" instance of $ID_\leqslant$ and $R_2(x)$ is a "yes" instance of $ID_\geqslant$, which is equal to $x \in L$.

Theorem 4.   ID is $\mathbf{FP^{NP[\log n]}}$-complete[2].

Proof: To show that ID is in $\mathbf{FP^{NP[\log n]}}$, consider problems of the following form: is the inconsistency degree less than $\frac{i}{|Var(K)|}$ (**NP**-complete by Theorem 2)? Through solving a *logarithmic* number of such problems (by dichotomy on $i \in \{0, \ldots, |Var(K)|\}$), we find an $\mathbf{FP^{NP[\log n]}}$ algorithm to compute the inconsistency degree, which shows that the problem belongs to $\mathbf{FP^{NP[\log n]}}$.

Next we prove that ID is $\mathbf{FP^{NP[\log n]}}$-hard, which is achieved by showing that MaxSAT[3] (the maximum satisfiability problem) can be polynomially reduced to an instance of ID.

W.o.l.g, assume the inconsistent propositional knowledge base is $K = \{\varphi_1, \ldots, \varphi_n\}$. Define a new knowledge base K_{new} as follows:

$$K_{new} = \{\varphi_i \vee \neg new_i, new_i \mid 1 \leqslant i \leqslant n\} \cup \{(p_i \wedge \neg p_i) \supset f \mid p_i \in Var(K)\},$$

where $\supset$ is the internal implication under four-valued semantics (see to Appendix for details). Clearly, the reduction is polynomial with respect to the size of K and $Var(K)$. Next, we aim to show that the maximal size of consistent subsets of K is M if and only if $ID(K_{new}) = \frac{n-M}{|Var(K)|+n}$, where $|Var(K)|+n = |Var(K_{new})|$. That is, the maximal size of consistent subsets of K is M if and only if $|Conflict(I, K_{new})| = n-M$ for any preferred model I of K_{new}.

(Only If) First, we show that $|Conflict(I, K_{new})| \leqslant n - M$. By the assumption, there is an M-size consistent subset of K, without loss of generality, written as $K_{cons} = \{\varphi_1, \ldots, \varphi_M\}$. Suppose J is a classical model of K_{cons} and define J' based on J as follows:

$$p^{J'} = \begin{cases} p^J, & \text{if } p \in Var(K), \\ t, & \text{if } p = new_i(i \leqslant M), \\ B, & \text{if } p = new_i(i > M). \end{cases}$$

By the fact that $p^{J'} = p^J$ for $p \in Var(K)$, we have that J' satisfies K_{cons} and in turn satisfies $\{\varphi_1 \vee \neg new_1, \ldots, \varphi_M \vee \neg new_M\}$. Obviously, J' satisfies $\{new_1, \ldots, new_M\}$. Moreover, we have J' satisfies $\{\varphi_{M+1} \vee \neg new_{M+1}, \ldots, \varphi_n \vee \neg new_n\}$ by the fact that $new_i^{J'} = B(i > M)$. Finally, note that J is a classical interpretation and J' equals J on all propositional letters in $Var(K)$. Therefore, J' interprets $p_i(p_i \in Var(K))$ classically. By the definition of semantics of internal implication, $(p_i \wedge \neg p_i) \supset f$ is satisfied by any classical interpretation, which means that J' satisfies $\{(p_i \wedge \neg p_i) \supset$

---

[2] Complexity $\mathbf{P^{NP[\log n]}}$ is defined to be the class of all languages decided by a polynomial-time oracle machine which on input x asks a total of $\mathcal{O}(\log |x|)$ SAT (or any other problem in **NP**) queries. $\mathbf{FP^{NP[\log n]}}$ is the corresponding class of functions. $\mathbf{FP^{NP[\log n]}}$ is also written as Θ_2^p.
[3] A MaxSAT problem is to ask for the maximum number of clauses which are satisfiable of a propositional knowledge base K.

$f \mid p_i \in Var(K)\}$. In all, we have that J' is 4-model of K_{new}. It is to see that $|Conflict(J', K_{new})| = |\{new_i \mid M < i \leqslant n\}| = n - m$. Then, by the definition of preferred model, we have $|Conflict(I, K_{new})| \leqslant n - M$.

Next we show that $|Conflict(I, K_{new})| \not< n - M$ for preferred models I of K_{new}. Otherwise, we can assume $|Conflict(I, K_{new})| < n - M$. By noting that $\{(p_i \wedge \neg p_i) \supset f \mid p_i \in Var(K)\} \subseteq K_{new}$ and by the semantics of internal implication, we have $p_i^I \neq B$ for all $p_i \in Var(K)$. That is, $p_i \notin Conflict(I, K_{new})$, which means $Conflict(I, K_{new}) \subseteq \{new_i \mid 1 \leqslant i \leqslant n\}$. By the assumption $|Conflict(I, K_{new})| < n - M$, there are at most $n - M - 1$ letters in $\{new_j \mid 1 \leqslant j \leqslant n\}$ values B in I. That is, there are at least $M + 1$ clauses $\varphi_{i_1} \vee \neg new_{i_1}, \dots, \varphi_{i_{M+1}} \vee \neg new_{i_{M+1}}$ in K_{new} with $new_{i_j}^I \in \{t, f, N\}$ for $j \in [1, M + 1]$. By the assumption that I is a model of K_{new} and $new_{i_j} \in K_{new}$, we have $new_{i_j}^I = t$. In all, $new_{i_j}^I \in \{t, N\}$ (i.e. $\neg new_{i_j}^I \in \{f, N\}$). Since $\varphi_{i_j} \vee \neg new_{i_j} \in K_{new}$, it holds that $\varphi_{i_j}^I \in \{t, B\}(1 \leqslant j \leqslant M + 1)$. That is, I satisfies at least $M + 1$ clauses of K. Consider the following classical interpretation I' for each $p \in Var(K)$:

$$p^{I'} = \begin{cases} t & \text{if } p^I = N, \\ p^I & \text{otherwise.} \end{cases}$$

Obviously, $I <_k I'$ where $<_k$ denotes the partial order w.r.t to amount of information in four-valued logic (Refer to Appendix for details). By the monotonicity of classical logical connectives ($\neg, \vee, \wedge, \rightarrow$) under the four-valued semantics (See to Proposition 8 in Appendix), we have I' satisfies $\{\varphi_{i_1}, \dots, \varphi_{i_{M+1}}\}$. Furthermore, by noting that $p \notin Conflict(I, K_{new})$, we can see that I' is a classical model of $\{\varphi_{i_1}, .., \varphi_{i_{M+1}}\}$, which conflicts with the fact that the maximal size of consistent subsets of K is M.

In all, we have $|Conflict(I, K_{new})| = n - M$.

(If) Similar to the proof of "only if" direction, the "if" direction can be proved.

5 Anytime Algorithm

According to the results shown in the previous section, computing inconsistency degrees is usually intractable. In this section, we propose an anytime algorithm to approximate the exact inconsistency degree. Our results show that in P-time we can get an interval containing the accurate value of $ID(K)$.

Firstly, by borrowing the idea of guidelines for a theory of approximating reasoning[19], we precise the requirements that an anytime approximating algorithm for computing inconsistency degrees should satisfy: It should be able to produce two sequences $r_1, \dots, r_m$ and $r^1, \dots, r^k$:

$$r_1 \leqslant \cdots \leqslant r_m \leqslant ID(K) \leqslant r^k \leqslant \cdots \leqslant r^1, \tag{1}$$

such that these two sequences have the following properties:

- The length of each sequence is polynomial w.r.t $|K|$;

- Computing r_1 and r^1 are both tractable. Generally, computing r_j and r^j becomes exponentially harder as j increases, but it is not harder than computing $ID(K)$.

- Since computing r_i and r^j could become intractable as i and j increase, we need to find functions $f(|K|)$ and $g(|K|)$ such that computing r_i and r^j both stay tractable as long as $i \leqslant f(|K|)$ and $j \leqslant g(|K|)$.

- Each r_i (r^j) in the two sequences is meaningful (in terms of corresponding to *approximating preferred models* in our case), which indicates the sense of the two sequences.

For the notation clarity, some definitions are necessary as given in the next section, which will be used to explain our algorithm.

5.1   Formal definitions

Definition 5.   (Bounding Values[14]) A real number x (resp. y) is a *lower (resp. an upper) bounding value* of the inconsistency degree of K, if and only if $x \leqslant ID(K)$ (*resp.* $ID(K) \leqslant y$).

Intuitively, a pair of lower and upper bounding values characterizes an interval containing the exact inconsistency degree of a knowledge base. For simplicity, lower (resp. an upper) bounding value is called *lower (resp. upper) bound*.

Corresponding to bounding values, we define a new concept called *bounding models* which are used to illustrate the sense of results of our anytime algorithm.

Definition 6.   (Bounding Models) A four-valued interpretation $\mathfrak{I}'$ is a *lower* (resp. an *upper*) bounding model of K if and only if for any preferred model $\mathfrak{I}$ of K, Condition 1 holds (resp. Condition 2 holds and $\mathfrak{I}' \in \mathcal{M}_4(K)$):

$$\text{Condition 1: } |Conflict(\mathfrak{I}', K)| \leqslant |Conflict(\mathfrak{I}, K)|$$
$$\text{Condition 2: } |Conflict(\mathfrak{I}', K)| \geqslant |Conflict(\mathfrak{I}, K)|$$

Intuitively, the lower and upper bounding models of K are approximations of preferred models from below and above. We call two-valued interpretations $\mathfrak{J}$ trivial lower bounding models since $Conflict(\mathfrak{J}, K) = 0$ and $ID(K) = 0$ always holds. We are only interested in nontrivial bounding models for inconsistent knowledge bases, which can produce a nonzero lower bound of $ID(K)$.

Example 2.   (Example 1 continued) K has a lower bounding model $\mathfrak{I}_3$ and an upper bounding model $\mathfrak{I}_4$ defined as: $p^{\mathfrak{I}_3} = t, q^{\mathfrak{I}_3} = t, r^{\mathfrak{I}_3} = t$; and $p^{\mathfrak{I}_4} = B, q^{\mathfrak{I}_4} = B, r^{\mathfrak{I}_4} = t$.

Next proposition gives a connection between *lower (resp. upper) bounds* and *lower (resp. upper) bounding models*.

Proposition 1.   If $\mathfrak{I}$ is a lower (an upper) bounding model of K, $Inc_{\mathfrak{I}}(K)$ is a lower (an upper) bounding value of $ID(K)$.

Proof: If $\mathfrak{I}$ is a lower bounding model of K, then $|Conflict(\mathfrak{I}, K)| \leqslant |Conflict(\mathfrak{I}_1, K)|$ for any preferred model $\mathfrak{I}_1$, which in turn leads to that $Inc_{\mathfrak{I}}(K) = \frac{|Conflict(\mathfrak{I}, K)|}{|Var(K)|} \leqslant \frac{|Conflict(\mathfrak{I}_1, K)|}{|Var(K)|} = ID(K)$. That is, $Inc_{\mathfrak{I}}(K)$ is a lower bounding value of $ID(K)$. Similarly, we can prove the conclusion in the case that $\mathfrak{I}$ is an upper bounding model of K.

5.2   Tractable approximations from above and below

By Theorem 1, we have an algorithm to compute inconsistency degrees via the computation of S-4 satisfiability. However, next theorem shows that S-4 entailment is generally intractable.

Theorem 5.   The decision problem of the S-4 satisfiability is **NP**-hard.

Proof: This theorem easily follows from the following reduction from SAT problem to S-4 satisfiability: A knowledge base K is classically two-valued satisfiable if and only if K is S-4 satisfiable, where $S = Var(K)$. The "only if" direction of the reduction is obvious because a classical model of K is also an S-4 model of K with $S = Var(K)$. For the "if" direction, suppose that K has an S-4 model I with $S = Var(K)$, that is, for any $p \in Var(K), I(p) \in \{N, t, f\}$. Define the following classical interpretation I':

$$I'(p) \in \begin{cases} I(p) & \text{if } I(p) \in \{t, f\}, \\ t & \text{if } I(p) = N. \end{cases}$$

By the monotonicity of classical logical connectives $(\neg, \vee, \wedge, \rightarrow)$ under the four-valued semantics (See to Proposition 8 in Appendix), we have I' satisfies K, that is, K is classically satisfiable.

This theorem shows that algorithms based on S-4 semantics to compute inconsistency degrees are time-consuming. In this section, by a tractable case of S-4 entailment (proportional to the size of input knowledge base)[14], we give an algorithm to compute approximating inconsistency degrees in tractable time.

Lemma 1 (Ref.[14]).   Let $S = \{p_1, \ldots, p_k\}$ be a subset of $Var(K)$ and φ be a formula such that $Var(\varphi) \subseteq Var(K)$. $K \models^4_S \varphi$ if and only if

$$K \wedge \bigwedge_{q \in Var(K) \backslash S} (q \wedge \neg q) \models_4 \varphi \vee (c_1 \vee \cdots \vee c_k)$$

holds for any combination $\{c_1, \ldots, c_k\}$, where each c_i is either p_i or $\neg p_i (1 \leqslant i \leqslant k)$.

This lemma shows a way to reduce the S-4 entailment to the 4-entailment. Specially note that if φ is in CNF (conjunctive normal formal), the righthand of the reduced 4-entailment maintains CNF form by a little bit of rewriting, as follows: Suppose $\varphi = C_1 \wedge \cdots \wedge C_n$. Then $\varphi \vee (c_1 \vee \cdots \vee c_k) = (C_1 \vee c_1 \vee \cdots \vee c_k) \wedge \cdots \wedge (C_n \vee c_1 \vee \cdots \vee c_k)$ which is still in CNF and its size is linear to that of $\varphi \vee (c_1 \vee \cdots \vee c_k)$.

Lemma 2[20]**.**   For K in any form and φ in CNF, there exists an algorithm for deciding if $K \models_4 \varphi$ in $\mathcal{O}(|K| \cdot |\varphi|)$ time.

By Lemma 1 and 2, we have the following theorem:

Theorem 6[14]**.**   There exists an algorithm for deciding if $K \models^4_S \varphi$ and deciding if K is S-4 satisfiable in $\mathcal{O}(|K||\varphi||S| \cdot 2^{|S|})$ and $\mathcal{O}(|K||S| \cdot 2^{|S|})$ time, respectively.

Theorem 6 shows that S-4 entailment and S-4 satisfiability can both be decided in polynomial time w.r.t the size of K, exponential w.r.t that of S, though. So they can be justified in P-time if $|S|$ is limited by a logarithmic function of $|K|$.

The following results in Ref.[14] are useful for our anytime algorithm.

Theorem 7[14]**.**   Given $S \subseteq Var(K)$, if K is S-4 satisfiable, then $ID(K) \leqslant 1 - |S|/|Var(K)|$.

Theorems 6 and 7 together show that for a monotonic sequence of sets $S_1,\ldots,S_k$, where $|S_i| < |S_{i+1}|$ for any $1 \leqslant i \leqslant k-1$, if we can show that K is S_i-4 $(i = 1,\ldots,k)$ satisfiable one by one, then we can get a sequence of decreasing upper bounding values of the inconsistency degree of K in time $\mathcal{O}(|K||S_i| \cdot 2^{|S_i|})$. If $|S_i| = \mathcal{O}(\log |K|)$, it is easy to see that the computation of an upper bound is done in polynomial time with respect to the size of K. In the worst case (i.e., when $S = Var(K)$), the complexity of the method coincides with the result that $\mathrm{ID}_{\leqslant}$ is NP-complete (Theorem 2).

Theorem 8[14]**.**   For a given w $(1 \leqslant w \leqslant |Var(K)|)$, if for each w-size subset S of $Var(K)$, K is S-4 unsatisfiable, then $ID(K) \geqslant 1 - (w-1)/|Var(K)|$.

Theorems 6 and 8 together show that for a monotonic sequence of sets $S_1,\ldots,S_m$ satisfying $|S_i| < |S_{i+1}|$, if we can prove that K is $|S_i|$-4 unsatisfiable[4] for each $i \in [1,m]$, then we can get a series of increasing lower bounds of the inconsistency degree of K. For each w, it needs at most $\binom{|Var(K)|}{w}$ times tests of S-4 unsatisfiability. So it takes $\mathcal{O}(\binom{|Var(K)|}{w}|K|w \cdot 2^w)$ time to compute a lower bound $1 - (w-1)/|Var(K)|$. If w is limited by a constant, we have that each lower bound is obtained in polynomial time.

Suppose r_i, r^j in Inequation 1 are defined as follows:

$$r^j = 1 - |S|/|Var(K)|, \text{ where } K \text{ is } |S|\text{-4 satisfiable}, j = |S|;$$

$$r_i = 1 - \frac{|S| - 1}{|Var(K)|}, \text{ where } K \text{ is } |S|\text{-4 unsatisfiable}, i = |S|.$$

By Theorems 6, 7 and 8, we get a way to compute the upper and lower bounds of $ID(K)$ which satisfy: if $j \leqslant \log(|K|)$ and $i \leqslant M$ (M is a constant independent of $|K|$), r^j and r_i are computed in polynomial time; Both i and j cannot be greater than $|Var(K)|$. This is a typical approximation process of a **NP**-complete problem $\mathrm{ID}_{\geqslant d}$ (resp. co**NP**-complete problem $\mathrm{ID}_{\leqslant d}$) via polynomial intermediate steps, because each intermediate step provides a partial solution which is an upper (resp. lower) bound of

Example 3.   Suppose $K = \{p_i \vee q_j, \neg p_i, \neg q_j \mid 1 \leqslant i,j \leqslant N\}$. So $|Var(K)| = 2N$. To know whether $ID(K) < \frac{3}{4}$, by Theorem 7 we only need to find an S of size $\lceil \frac{2N}{4} \rceil$ such that K is S-4 satisfiable. This is true by choosing $S = \{p_i \mid 1 \leqslant i \leqslant \lceil \frac{2N}{4} \rceil\}$. To know whether $ID(K) > \frac{1}{3}$, Theorem 8 tells us to check whether K is S-4 unsatisfiable for all S of size $\lfloor \frac{4N}{3} \rfloor + 1$. This is true also. So $ID(K) \in [\frac{1}{3}, \frac{3}{4}]$.

An interesting consequence of the above theoretical results is that we can compute the exact inconsistency of some knowledge bases in P-time. Let us first look at an example.

Example 4.   Consider a knowledge base $K = \{(p_i \vee p_{i+1}) \wedge (\neg p_i \vee \neg p_{i+1}), p_{i_1} \wedge \cdots \wedge p_{i_{N-5}}, \neg p_{j_1} \wedge \cdots \wedge \neg p_{j_{N-10}}, p_{2t}, \neg p_{3j+1} \vee \neg p_{5u+2}, \}(1 \leqslant i \leqslant N-1, 1 \leqslant 2t, 3j+1, 5u+2 \leqslant N)$. $Var(K) = N$. To approximate $ID(K)$, we can check whether K is l-4 satisfiable for l going larger from 1 by one increase on the value each time. Obviously, K's inconsistency degree is close to 1 if $N \gg 10$. By Theorem 6, we can see that all of these operations can be done in P-time before the exact value obtained.

More formally, we have the following proposition.

_____

[4] For the sake of simplicity, we say that K is l-4 satisfiable for $l \in \mathbb{N}$, if there is a subset $S \subseteq Var(K)$ such that K is S-4 satisfiable. We say that K is l-4 unsatisfiable if K is not l-4 satisfiable.

Proposition 2.   If $ID(K) \geqslant 1 - M/|Var(K)|$, where M is an arbitrary constant which is independent of $|K|$, then $ID(K)$ can be computed in polynomial time.

Proof: By the definition of inconsistency degree and the assumption $ID(K) \geqslant 1 - \frac{M}{|Var(K)|}$, we know that $\frac{|Conflict(\mathfrak{I},K)|}{|Var(K)|} \geqslant 1 - \frac{M}{|Var(K)|}$ for any preferred model $\mathfrak{I}$ of K. That is,

$$|Conflict(\mathfrak{I}, K)| \geqslant |Var(K)| - M, \tag{2}$$

We claim that K has no S-4 model for any $S \subseteq Var(K)$ whose size is strictly greater than M. If not, suppose $|S_0| > M$ makes K S_0-4 satisfiable. By the definition of S-4 semantics, we have $|Conflict(\mathfrak{I},K)| = |Var(K)| - |S| < |Var(K)| - M$. This is a contradiction with Inequation 2. Let us check whether K is l-4 satisfiable for l going larger from 1 by one increase on the value each time until K becomes l-4 unsatisfiable, then the accurate inconsistency degree $ID(K) = 1 - (l-1)/|Var(K)|$. By the claim above, the first l which makes K l-4 unsatisfiable is for $l = M + 1$ and K keeps l-unsatisfiable for $l > M$. Note that checking K l-4 satisfiability from $l = 1$ to $M + 1$ can be done in polynomial time by Theorem 6. Therefore, we find a way to get the exact inconsistency degree of K in P-time.

5.3   The anytime algorithm

Given a knowledge base K with $|Var(K)| = n$, it is natural to perform dichotomy on n to search for the maximal size of $S \in Var(K)$ such that K is S-4 satisfiable. However, we will see, in this section firstly, that it leads to intractability from the beginning. To avoid this, subsequently, we give an anytime algorithm which can return approximating inconsistency degrees in polynomial time.

By the analysis given after Theorem 7 and Theorem 8, we know that in the worst case, given $0 \leqslant w \leqslant n$, it takes $\mathcal{O}(\binom{n}{w}|K|w2^w)$ time to get an upper (resp. a lower) bounding value $1 - \frac{w}{|Var(K)|}$ (resp. $1 - \frac{w-1}{|Var(K)|}$). By Fermat's Lemma[5], the maximal value of $\mathcal{O}(\binom{n}{w}|K|w2^w)$ is near $w = \lceil \frac{2n+1}{3} \rceil$ when n is big enough. It means that *to do dichotomy directly on size $\frac{n}{2}$ will be of high complexity.* To get upper and lower bounding values in P-time instead of going to intractable computation directly, we propose Algorithm 1, which consists of two stages: The first one is to localize an interval $[l_1, l_2]$ that contains the inconsistency degree (line 1-8), while returning upper and lower bounding values in P-time; The second one is to pursue more accurate approximations within the interval $[l_1, l_2]$ by binary search (line 9-17).

Algorithm 1 is an anytime algorithm that can be interrupted at any time and returns a pair of upper and lower bounding values of the exact inconsistency degree. It has five parameters: the knowledge base K we are interested in; the precision threshold ε which is used to control the precision of the returned results; the constant $M \ll |Var(K)|$ to guarantee that the computation begins with tractable approximations; a pair of positive reals a, b which determines a linear function $h(l_2) = al_2 + b$ that updates the interval's right extreme point l_2 by $h(l_2)$ during the first stage (line 5). $h(\cdot)$ decides how to choose the sizes l to test l-4 satisfiability of K. For example, if $h(l_2) = l_2 + 2$, line 5 updates l from i to $i + 1$ (suitable for $ID(K)$ near 1); If $h(l_2) = 2l_2$, line 5 updates l from i to $2i$ (suitable for $ID(K)$ near 0.5); While if $h(l_2) = 2(|Var(K)| - M)$, line 5 updates l by $|Var(K)| - M$ (suitable for $ID(K)$ near

---

[5] V.A. Zorich, *Mathematical Analysis*, Springer, 2004.

0). We remark that $h(l_2)$ can be replaced by other functions.

Algorithm 1 Approx_Incons_Degree$(K, \varepsilon, M, a, b)$

Input: KB K; precision threshold $\varepsilon \in [0, 1[$; constant $M \ll |Var(K)|$; $a, b \in \mathbb{R}^+$

Output: Lower bound r_- and upper bound r_+ of $ID(K)$

1: $r_- \leftarrow 0; r_+ \leftarrow 1$ {Initial lower and upper bounds}
2: $\bar{\varepsilon} \leftarrow r_+ - r_-; n \leftarrow |Var(K)|; l_1 \leftarrow 0; l_2 \leftarrow M; l \leftarrow \lceil \frac{l_2}{2} \rceil$
3: **while** $\bar{\varepsilon} > \varepsilon$ and K is l-4 satisfiable **do**
4:     $r_+ \leftarrow (1 - l/n); \bar{\varepsilon} \leftarrow r_+ - r_-$ {Update upper bound}
5:     $l_1 \leftarrow l; l_2 \leftarrow h(l_2); l \leftarrow \lceil \frac{l_2}{2} \rceil$ {Update search interval}
6: **end while**
7: $r_- \leftarrow 1 - (l-1)/n; \bar{\varepsilon} \leftarrow r_+ - r_-; l_2 \leftarrow l$
8: **if** $\bar{\varepsilon} \leqslant \varepsilon$ **then**   **return** r_+ and r_- **end if**
9: **while** $\bar{\varepsilon} > \varepsilon$ **do**
10:     $l \leftarrow l_1 + \lceil \frac{l_2 - l_1}{2} \rceil$
11:     **if** K is l-4 satisfiable **then**
12:         $r_+ \leftarrow (1 - l/n); \bar{\varepsilon} \leftarrow r_+ - r_-; l_1 \leftarrow l$
13:     **else**
14:         $r_- \leftarrow 1 - (l-1)/n; \bar{\varepsilon} \leftarrow r_+ - r_-; l_2 \leftarrow l$
15:     **end if**
16: **end while**
17: **return** r_+ and r_-

Next we give detailed explanations about Algorithm 1. To guarantee that it runs in P-time at the beginning to return approximations, we begin with a far smaller search interval $[l_1, l_2] = [0, M]$ compared to $|Var(K)|$. The *while* block (line 3) iteratively tests whether the difference between upper and lower bounding values is still lager than the precision threshold and whether K is l-satisfiable, where $l = \lceil \frac{l_2}{2} \rceil$. If both yes, the upper bound r_+ is updated, the testing interval becomes $[l, h(l_2)]$, and the iteration continues; Otherwise (line 7), the lower bound r_- is updated and the search interval becomes $[l_1, l]$. This completes the first part of the algorithm to localize an interval. If $r_+ - r_-$ is already below the precision threshold, the algorithm terminates (line 8). Otherwise, we get an interval $[l_1, l_2]$ such that K is l_1-4 satisfiable and l_2-4 unsatisfiable. Then the algorithm turns to the second "while" iteration (line 9) which executes binary search within the search internal $[l_1, l_2]$ found in the first stage. If there is a subset $|S| = l_1 + \lceil \frac{l_2 - l_1}{2} \rceil$ such that K is S-4 satisfiable, then the search internal shortens to the right half part of $[l_1, l_2]$ (line 12), otherwise to the left half part (line 14). *During this stage, K keeps l_2-4 unsatisfiable and l_1-4 satisfiable for $[l_1, l_2]$.* Until $r_+ - r_-$ below the precision threshold, the algorithm finishes and returns upper and lower bounds.

Theorem 9 (Correctness of Algorithm 1).   Let r_+ and r_- be values computed by Algorithm 1. We have $r_- \leqslant ID(K)$ and $r_+ \geqslant ID(K)$. Moreover, $r_+ = r_- = ID(K)$ if $\varepsilon = 0$.

Proof: By analyzing Algorithm 1, r_+ is updated as $1 - l/|Var(K)|$ only if K is l-4 satisfiable. By Theorem 7, $r_+ \geqslant ID(K)$. Similarly, r_- is updated as $1 - (l-1)/|Var(K)|$ only if K is l-4 unsatisfiable. By Theorem 8, $r_- \leqslant ID(K)$. Note

that Algorithm 1 terminates only when $\bar{\varepsilon} \leqslant \varepsilon$. If $\varepsilon = 0$, $r_+ - r_- = \bar{\varepsilon} \leqslant \varepsilon$. So $r_+ = r_- = ID(K)$.

The following example gives a detailed illustration.

Example 5 (Example 3 contd). Let $\varepsilon = 0.1, h(l_2) = 2l_2$, and $M = 4 \ll N$. Algorithm 1 processes on K as follows:

Denote the initial search interval $[l_1^0, l_2^0] = [0, 4]$. After initializations, $l = 2$ and line 3 is executed. Obviously, K is S-4 satisfiable for some $|S| = l$ (e.g. $S = \{p_1, p_2\}$). So we get a newer upper bound $r_+ = \frac{2N-l}{2N}$. Meanwhile, the difference between upper and lower bounds $\bar{\varepsilon}$ becomes $\frac{2N-l}{2N} > \varepsilon$, and the search interval is updated as $[l_1^1, l_2^1] = [l, 2l_2]$ and $l = 4$.

Stage 1. The *while* iteration from line 3 is repeatedly executed with double size increase of l each time. After c times such that $2^{c-1} \leqslant N < 2^c$, $l = 2^c$ and K becomes l-4 unsatisfiable. The localized interval is $[2^{c-1}, 2^c]$. It turns to line 7 to update the lower bound by $1 - \frac{l-1}{2N}$. The newest upper bound is $1 - 2^{c-2}/N$, so $\bar{\varepsilon} = 2^{c-2}/N$. If $\bar{\varepsilon} \leqslant \varepsilon$, algorithm ends by line 8. Otherwise, it turns to stage 2.

Stage 2. By dichotomy in the interval $[2^{c-1}, 2^c]$, algorithm terminates until $\bar{\varepsilon} \leqslant \varepsilon$.

Unlike Example 5, for the knowledge base in Example 4, since its inconsistency degree is quite close to 1, it becomes S-4 satisfiable for an S such that $|S|$ is less than a constant M. Therefore, after the first stage of Algorithm 1 applying on this knowledge base, the localized interval $[l_1, l_2]$ is bounded by M. For such an interval, the second stage of the algorithm runs in P-time according to Theorem 6. So Algorithm 1 is a P-time algorithm for the knowledge base given in Example 4. However, it fails for other knowledge bases whose inconsistency degrees are far less than 1. Fortunately, the following proposition shows that by setting the precision threshold ε properly, Algorithm 1 can be executed in P-time to return approximating values.

Proposition 3. Let s be an arbitrary constant independent of $|K|$. If $\varepsilon \geqslant 1 - \frac{h^s(M)}{2|Var(K)|}$, where $h^s(\cdot)$ is s iterations of $h(\cdot)$, Algorithm 1 terminates in polynomial time with the difference between upper and lower bounds less than ε ($r_+ - r_- \leqslant \varepsilon$).

Proof: Algorithm 1 terminates if and only if $\bar{\varepsilon} \leqslant \varepsilon$. At the beginning of the algorithm, $r_+ = 1, r_- = 0$, and $\bar{\varepsilon} = 1$. Suppose $r_+ = 1 - l/|Var(K)|$ after the first *while* block beginning line 3. At this moment, $\bar{\varepsilon} = r_+ - r_- = 1 - l/|Var(K)|$. It has two cases:

- $1 - l/|Var(K)| \leqslant \varepsilon$ holds such that the algorithm terminates. It is not difficult to see that the while block (line 3) will end if l reaches to $h^s(M)/2$ because $\varepsilon \geqslant 1 - \frac{h^s(M)}{2|Var(K)|}$ and $\bar{\varepsilon} = 1 - l/|Var(K)|$. Note that $l = 0, \lceil M/2 \rceil, \lceil h(M)/2 \rceil, \cdots$ in each iteration of the while block. Therefore, it takes s times of l-4 satisfiability tests of K, each of which is P-time by Theorem 6. Because s is a constant independent of $|Var(K)|$, the computation time is P-time in all.

- $1 - l/|Var(K)| > \varepsilon$ which means that the while block runs for less than s times. So the localized interval $[l_1, l_2]$ satisfies $0 \leqslant l_2 - l_1 \leqslant h^s(M)/2$, that is, it is bounded by a constant independent of $|VarK|$. Then the binary search in this interval costs P-time because logarithmic times of P-time computations is still in P-time.

In all, the algorithm terminates in P-time.

The following proposition shows that r_- and r_+ computed by Algorithm 1 have a sound semantics in terms of *upper* and *lower bounding models* defined in Definition 6.

Proposition 4. There is a lower (an upper) bounding model $\mathfrak{J}'$ ($\mathfrak{J}''$) of K such that $Inc_{\mathfrak{J}'}(K) = r_-$ ($Inc_{\mathfrak{J}''}(K) = r_+$).

Proof: For r_+, there is an $S \subseteq \Sigma$ such that K is S-4 satisfiable and $r_+ = 1 - \frac{|S|}{|\Sigma|}$. Therefore, K has an S-4 model, written $\mathfrak{J}'$, and $Inc_{\mathfrak{J}'}(K) = 1 - \frac{|S|}{|\Sigma|}$. Obviously, $\mathfrak{J}'$ is an upper bounded model and $Inc_{\mathfrak{J}'}(K) = r_+$.

For r_-, if $r_- = 1 - \frac{l'-1}{|Var(K)|}$, then by Algorithm 1, K is S'-unsatisfiable for all l'-size subsets S' and there is at least one S'' such that $|S''| = l' - 1$ and K is S''-4 satisfiable. So K has an S''-4 model, written $\mathfrak{J}''$. By the proof of Theorem 8, we know that for any preferred model $\mathfrak{J}$ of K, $|Conflict(\mathfrak{J}, K)| \geqslant |Var(K)| - |S''|$, then $\mathfrak{J}''$ is a lower bounded model of K and $Inc_{\mathfrak{J}''}(K) = 1 - \frac{|S'|-1}{|\Sigma|} = r_-$.

Summing up, we have achieved an anytime algorithm for approximately computing inconsistency degrees which is:

- *computationally tractable:* Each approximating step can be done in polynomial time if $|S|$ is limited by a logarithmic function for upper bounds (Theorems 6 and 7) and by a constant function for lower bounds (Theorems 6 and 8).

- *dual and semantical well-founded:* The accurate inconsistency degree is approximated both from above and from below (Theorem 9), corresponding to inconsistency degrees of some upper and lower bounding models of K (Proposition 4).

- *convergent:* More computation resource available, more precise values returned (Theorems 7 and 8). It always converges to the accurate value if there is no limitation of computation resource (Theorem 9) and terminates in polynomial time for special knowledge bases (Proposition 2).

Proposition 5. Given two sets S and S' satisfying $S \subseteq S' \subseteq \mathcal{P}$, if a theory K is S-4 unsatisfiable, then it is S'-4 unsatisfiable.

Proof: Assume that K is S-4 unsatisfiable and that there exists an S'-4 interpretation $\mathfrak{J}_{S'}$ satisfying K. We construct an S-4 interpretation $\mathfrak{J}_S$ as follows. For each propositional letter $p \in \mathcal{P}$:

$$p^{\mathfrak{J}_S} = \begin{cases} B & \text{if } p \in S' \setminus S, \\ p^{\mathfrak{J}_{S'}} & \text{otherwise.} \end{cases}$$

Obviously, $\mathfrak{J}_S$ is an S-4 model of K, a contradiction.

Proposition 5 says that if we have known that K is S-4 unsatisfiable, then there's no necessity to test its S'-4 satisfiability for $S' \subset S$. By this proposition, we can get a truncation strategy to limit the search space in the implementation of our algorithm discussed in the next section:

Definition 7 (Truncation Strategy). For any knowledge base K, if an $S \in Var(K)$ is found which makes K being S-4 satisfiable, then all supersets S' of S are pruned.

6　Evaluation

Our algorithm has been implemented in Java using a computer with Intel E7300 2.66G, 4G, Windows Server 2008. Algorithm 1 gives a general framework to approximate inconsistency degrees from above and below. In our implementation, we set $M = 2, h(l_2) = l_2 + 2$. That is, the first *while* loop (see line 3) keeps testing l-4 satisfiability of K from $l = i$ to $i + 1$. So the interval $[l_1, l_2]$ localized in the first stage of the algorithm satisfies $l_2 = l_1 + 1$ and the second binary search is not necessary. According to our analysis in Section 5, this avoids direct binary search which needs to test all $\frac{n!}{(n/2)!(n/2)!}$ subsets of $Var(K)$, where $n = |Var(K)|$.

There are tow main sources of complexity to compute approximating inconsistency degrees: *the complexities of S-4 satisfiability* and *of search space*. The S-4 satisfiability that we implemented is based on the reduction given in Lemma 1 and the tractable algorithm for 4-satisfiability in Ref.[20]. Our experiments told us that search space could heavily affect efficiency. So we carefully designed a truncation strategy to limit the search space based on the monotonicity of S-4 unsatisfiability. That is, if we have found an S such that K is S-4 unsatisfiable, then we can prune all supersets S' of S which makes K S'-4 unsatisfiable. We implemented this strategy in breadth-first search on the binomial tree[21,22] of subsets of $Var(K)$.
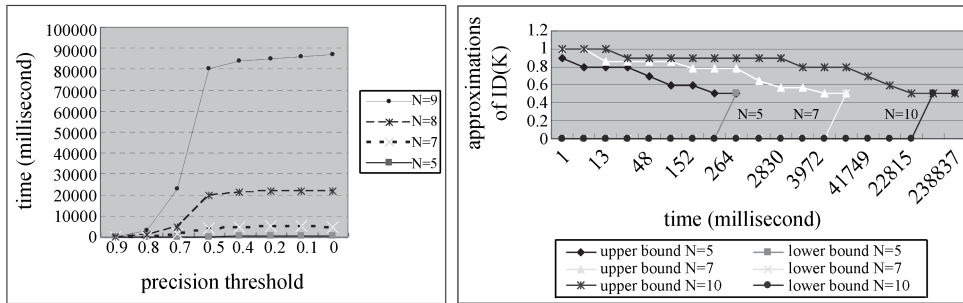

Figure 1.　Evaluation results over KBs in Example 3 with $|K| = N^2 + 2N$ and $|Var(K)| = 2N$ for $N = 5, 7, 8, 9, 10$.

Figure 1 shows the evaluation results over knowledge bases[6] in Example 3 with $|K| = N^2 + 2N$ and $|Var(k)| = 2N$ for $N = 5, 7, 8, 9, 10$. The left part of the figure shows how the preset precision threshold ε affects the run time performance of our algorithm: the smaller ε is, the longer it executes. If $\varepsilon \geqslant 0.7$, the algorithm terminated easily (at most $18.028s$ for $N = 9$ and much less for $N < 9$). The quality of the approximations at different time points is shown on the right part of the figure. The decreasing (resp. increasing) curves represent upper (resp. lower) bounds for $N = 5, 7, 10$, respectively. Note that the inconsistency degrees of all the three knowledge bases are 0.5.

For large knowledge bases, it is time-consuming to compute the exact inconsistency degrees. For example, for $N = 10$, our algorithm took $239.935s$ to get the accurate inconsistency degree. In contrast, by costing much less time, approximating

---

[6] We use instances of Example 3 because they are the running examples through the paper and meet the worst cases of the algorithm (e.g. the *truncation strategy* discussed later cannot be applied). We want to show the performance of our algorithm in its worst case.

values (upper bounds for these examples) can provide a good estimation of the exact value and are much easier to compute. For example, when $N = 10$, the algorithm told us that the inconsistency degree is less than 0.8 at 3.9s; and when $N = 5$, we got the upper bound 0.6 at 0.152s. Note that in these experiments, the lower bounds were updated slowly. In fact, the exact inconsistency degrees were obtained as soon as the first nonzero lower bounding values were returned. This is because we set $M = 2, h(l_2) = l_2 + 2$ in our implementation. If we set M and $h(\cdot)$ differently, the results will be changed, as shown in Example 3 in Section 5.

We need to point out that our truncation strategy cannot be applied to the test data used in the experiments because no subsets can be pruned. Therefore, although our experiments show the benefits of the approximations, our algorithm can increase significantly when the truncation strategy is applicable and if we carefully set M and $h(\cdot)$. Take $\{p_i, \neg p_j \mid 0 \leqslant i, j < 20, j \text{ is odd}\}$ for example, our optimized algorithm run less than 1s whilst it run over 5min without the truncation strategy.

7   Conclusion

In this paper, we investigated computational aspects of the inconsistency degree. We showed that the complexities of several decision problems about inconsistency degree are high in general. To compute inconsistency degrees more practically, we proposed an general framework of an anytime algorithm which is *computationally tractable, dual and semantical well-founded, and improvable and convergent.* The experimental results of our implementation show that computing approximating inconsistency degrees is much faster than computing the exact inconsistency degrees in general. The approximating inconsistency degrees can be useful in many applications, such as knowledge base evaluation and merging inconsistent knowledge bases. We will further study on the real applications of approximating inconsistency degrees in the future work.

References

[1] Hunter A. How to act on inconsistent news: Ignore, resolve, or reject. Data & Knowlnowledge Engineering, 2006, 57: 221–239. [doi: 10.1016/j.datak.2005.04.005]

[2] Hunter A. Measuring inconsistency in knowledge via quasi-classical models. Proceedings of the Eighteenth National Conference on Artificial Intelligence and Fourteenth Conference on Innovative Applications of Artificial Intelligence (AAAI'02). July 28 - August 1, 2002, Edmonton, Alberta, Canada, AAAI Press, 2002. 68–73.

[3] Hunter A, Konieczny S. Approaches to measuring inconsistent information. In: Bertossi LE, Hunter A, Schaub T, eds. Inconsistency Tolerance. LNCS 3300, Springer, 2005. 191–236.

[4] Hunter A, Konieczny S. Shapley inconsistency values. In: Patrick Doherty, John Mylopoulos CAW, ed. Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning KR'06, AAAI Press, 2006. 249–259.

[5] Mu K, Jin Z, Lu R, *et al.* Measuring inconsistency in requirements specifications. In: Godo L, ed. Proceedings of the 8th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU'05). Barcelona, Spain, July 6-8, 2005, Proceedings. LNCS 3571, Springer, 2005. 440–451.

[6] Knight K. Measuring inconsistency. Journal of Philosophical Logic, 2002, 31(1): 77–98.

[7] Hunter A, Konieczny S. Measuring inconsistency through minimal inconsistent sets. In: Brewka G, Lang J, eds. Proceedings of the 11th International Conference Principles of Knowledge Representation and Reasoning (KR'08). Sydney, Australia, September 16-19, 2008, AAAI Press,

2008. 358–366.

[8] Grant J. Classifications for inconsistent theories. Notre Dame J. of Formal Logic, 1978, 19: 435–444.

[9] Grant J, Hunter A. Measuring inconsistency in knowledge bases. Journal of Intelligent Information Systems, 2006, 27: 159–184. [doi: 10.1007/s10844-006-2974-4]

[10] Grant J, Hunter A. Analysing inconsistent first-order knowledge bases. Artificial Intelligence, 2008, 172: 1064–1093. [doi: 10.1016/j.artint.2007.11.006]

[11] Coste-Marquis S, Marquis P. A unit resolution-based approach to tractable and paraconsistent reasoning. In: de Mántaras RL, Saitta L, eds. Proceedings of the 16th European Conference on Artificial Intelligence (ECAI'04), including Prestigious Applicants of Intelligent Systems, PAIS 2004, Valencia, Spain, August 22-27, 2004, IOS Press, 2004. 803–807.

[12] Ma Y, Qi G, Xiao G, *et al*. An anytime algorithm for computing inconsistency measurement. In: Karagiannis D, Jin Z, eds. Proceedings of the Third International Conference Knowledge Science, Engineering and Management (KSEM'09). Vienna, Austria, November 25-27, 2009. LNCS 5914, Springer, 2009. 29–40.

[13] Ma Y, Qi G, Hitzler P, *et al*. An algorithm for computing inconsistency measurement by paraconsistent semantics. In: Mellouli K, ed. Proceedings of the 9th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU'07). Hammamet, Tunisia, October 31 - November 2, 2007. LNCS 4724, Springer, 2007. 91–102.

[14] Ma Y, Qi G, Hitzler P. Computing inconsistency measure based on paraconsistent semantics. Journal of Logic and Computation. To appear.

[15] Belnap ND. A useful four-valued logic. In: Modern uses of multiple-valued logics, Reidel Publishing Company, 1977. 7–73.

[16] Arieli O, Avron A. The value of the four values. Artificial Intelligence, 1998, 102: 97–141.

[17] Arieli O, Denecker M. Reducing preferential paraconsistent reasoning to classical entailment. Journal of Logic and Computation, 2003, 13: 557–580.

[18] Papadimitriou CH. Computational Complexity. Addison Wesley, 1994.

[19] Schaerf M, Cadoli M. Tractable reasoning via approximation. Artificial Intelligence, 1995, 74: 249–310.

[20] Cadoli M, Schaerf M. On the complexity of entailment in propositional multivalued logics. Annals of Mathematics and Artificial Intelligence, 1996, 18: 29–50.

[21] Malouf R. Maximal consistent subsets. Computational Linguistics, 2007, 33: 153–160

[22] Mu K, Jin Z, Liu W, *et al*. An approach to measuring the significance of inconsistency in viewpoints framework. Technical report, Peking University, 2008.

[23] Arieli O, Avron A. Reasoning with logical bilattices. Journal of Logic, Language and Information, 1996, 5: 25–63.

Appendix. Four-valued Logic

Four-valued logic is based on the idea of having four truth values, instead of the classical two. The four truth values stand for *true, false, unknown* (or *undefined*) and *both* (or *overdefined, contradictory*). We use the symbols t, f, N, B, respectively, for these truth values, and the set of these four truth values is denoted by FOUR. The truth value B stands for contradictory information, hence four-valued logic lends itself to dealing with inconsistent knowledge. The value B thus can be understood to stand for *true and false*, while N stands for *neither true nor false*, i.e. for the absence of any information about truth or falsity.

Syntactically, four-valued logic is very similar to classical logic. Care has to be taken, however, in defining meaningful notions of implication, as there are several ways to do this. Indeed, there are three major notions of implication in the literature, namely the material implication $\mapsto$, the internal implication $\supset$, and the strong

implication $\rightarrow$, which are discussed in detail in Refs.[16, 23]. Thus the set of logical connectives allowed in four-valued logic is $\{\neg, \vee, \wedge, \mapsto, \supset, \rightarrow\}$.

Four-valued interpretations for formulae (i.e. 4-interpretations) are obviously mappings from formulae to (the set of four) truth values, respecting the truth tables for the logical connectives, as detailed in Table 1

Table 1 Truth Table for 4-valued Connectives

α	f	f	f	f	t	t	t	t	B	B	B	B	N	N	N	N
β	f	t	B	N	f	t	B	N	f	t	B	N	f	t	B	N
$\neg\alpha$	t	t	t	t	f	f	f	f	B	B	B	B		N	N	N
$\alpha \wedge \beta$	f	f	f	f	f	t	B	N	f	B	B	f	f	N	f	N
$\alpha \vee \beta$	f	t	B	N	t	t	t	t	B	t	B	t	N	t	t	N
$\alpha \mapsto \beta$	t	t	t	t	f	t	B	N	B	t	B	t	N	t	t	N
$\alpha \supset \beta$	t	t	t	t	f	t	B	N	f	t	B	N	t	t	t	t
$\alpha \rightarrow \beta$	t	t	t	t	f	t	f	N	f	t	B	N	N	t	N	t

Four-valued models (4-models) are defined in the obvious way, as follows, where t and B are the designated truth values.

Definition 8. Let I be a 4-interpretation, let Σ be a theory (i.e. set of formulae) and let φ be a formula in four-valued logic. Then we call that I is a 4-model of φ if and only if $I(\varphi) \in \{t, B\}$; We say that I is a 4-model of Σ if and only if I is a 4-model of each formula in Σ; And we name that Σ four-valued entails φ, written $\Sigma \models_4 \varphi$, if and only if every 4-model of Σ is a 4-model of φ.

Proposition 6. We note the following general properties.

- The language $L = \{\neg, \vee, \wedge, \supset, N, B\}$ is functional complete for the set **FOUR** of truth values, i.e. every function from $\textbf{FOUR}^n$ to **FOUR** is representable by some formula in L [16, Theorem 12].

- Any formula containing only connectives from $\{\neg, \vee, \wedge, \supset\}$ always has a four-valued model.

Some general remarks about the different notions of implication are in order. The basic rationales behind them are the following: Material implication can be defined by means of negation and disjunction as known from classical logic. However, it does not satisfy Modus Ponens or the deduction theorem, and is thus of limited use as an *implication* in the intuitive sense. Internal implication satisfies Modus Ponens and the deduction theorem, but cannot be defined by means of other connectives. Furthermore, internal implication does not satisfy contraposition. Strong implication is stronger than internal implication, in that it additionally satisfies contraposition. Indeed, an alternative view on the truth tables for the implication connectives is as

follows.

$$\varphi \mapsto \psi \quad \text{is definable as} \quad \neg\varphi \vee \psi. \qquad \text{(Material Implication)}$$

$$\varphi \supset \psi \quad \text{evaluates to} \quad \begin{cases} \psi & \text{if } \varphi \in \{t, B\} \\ t & \text{if } \varphi \in \{f, N\} \end{cases} \qquad \text{(Internal Implication)}$$

$$\varphi \rightarrow \psi \quad \text{is definable as} \quad (\varphi \supset \psi) \wedge (\neg\psi \supset \neg\varphi) \qquad \text{(Strong Implication)}$$

Further properties of the implication connectives are summarized in the following proposition (as shown in Ref.[16, Corollary 9] and Ref.[23]).

Proposition 7[16]. The following claims hold, where Γ is a theory and ψ, ϕ are formulae.

- Internal implication is not definable in terms of the connectives $\neg, \vee, \wedge$.

- $\Gamma, \psi \models_4 \phi$ iff $\Gamma \models_4 \psi \supset \phi$.

- If $\Gamma \models_4 \psi$ and $\Gamma \models_4 \psi \supset \phi$ then $\Gamma \models_4 \phi$.

- $\psi \rightarrow \phi$ implies that $\neg\phi \rightarrow \neg\psi$.

The other partial order defined on the four truth values $\{t, f, B, N\}$, denote $<_k$, is to reflect differences in the amount of knowledge or information that each truth value exhibits: $N <_k t <_k B, N <_k f <_k B, t \not<_k f$. That is, $(\{t, f, B, N\}, <_k)$ is a lattice where $<_k$ is its minimal element N, its maximal element B, and t, f are incomparable. The truth operators $\wedge, \vee$, and $\neg$ are monotone with respect to $<_k$. For two four-valued interpretations I, I', we call $I <_k I'$ if and only if $p^I <_k p^{I'}$ for any propositional letter p in the considered language.

Proposition 8[16]. For any given four-valued interpretations I, I' and any formula ϕ containing only connections from $\{\vee, \wedge, \neg, \rightarrow\}$, suppose $I <_k I'$, then $\phi^I <_k \phi^{I'}$. Moreover, if $I \models_4 \phi$, $I' \models_4 \phi$.