Probable Consistency Checking for Sets of Propositional Clauses

Anthony Hunter

Department of Computer Science
University College London
Gower Street, London WC1E 6BT, UK

Abstract. Inconsistencies inevitably arise in knowledge during practical reasoning. In a logic-based approach, this gives rise to the need for consistency checking. Unfortunately, this can be difficult. In classical propositional logic, this is intractable. However, there is a useful alternative to the notion of consistency called probable consistency. This offers a weakening of classical consistency checking where polynomial time tests are done on a set of formulae to determine the probability that the set of formulae is consistent. In this paper, we present a framework for probable consistency checking for sets of clauses, and analyse some classes of polynomial time tests.

1 Introduction

In order to manage inconsistency in knowledge, we need to undertake consistency checking. However, consistency checking is inherently intractable in the case of propositional classical logic. To address this problem, we can consider using (A) tractable subsets of classical logic (for example binary disjunctions of literals [3]), (B) heuristics to direct the search for a model[1] (for example in semantic tableau [7], GSAT [10], and constraint satisfaction [2]), (C) some form of knowledge compilation (for example [6, 1]), and (D) formalization of approximate consistency checking based on notions described below, such as approximate entailment, and partial and probable consistency.

Approximate entailment Proposed in [5], and developed in [9, 4], classical entailment is approximated by two sequences of entailment relations. The first is sound but not complete, and the second is complete but not sound. Both sequences converge to classical entailment. For a set of propositional formulae Δ, a formula α, and an approximate entailment relation $\models_i$, the decision of whether $\Delta \models_i \alpha$ holds or $\Delta \not\models_i \alpha$ holds can be computed in polynomial time.

Partial consistency Consistency checking does not necessarily involve an exponential search space. Furthermore, consistency checking for a set of formulae Δ can be

[1] Heuristic approaches can be either complete such as semantic tableau or incomplete such as in the GSAT system. Whilst in general, using heuristics to direct search has the same worst-case computational properties as undirected search, it can offer better performance in practice for some classes of theories. Note, heuristic approaches do not tend to be oriented to offering any analysis of theories beyond a decision on consistency.

prematurely terminated when the search space exceeds some threshold. When the checking of Δ is prematurely terminated, partial consistency is the degree to which Δ is consistent. This can be measured in a number of ways including the proportion of formulae from Δ that can be shown to form a consistent subset of Δ. Maximum generalized satisfiability [8] may be viewed as an example of this.

Probable consistency Determining the probability that a set of formulae is consistent on the basis of polynomial time classifications of those formulae. Classifications for the propositional case can be based on tests including counting the number of different propositional letters, counting the multiple occurrences of each propositional letter, and determining the degree of nesting for each logical symbol. The more a set of formulae is tested, the greater the confidence in the probability value for consistency/inconsistency, but this is at the cost of undertaking the tests.

Identifying approximate consistency for a set of formulae Δ is obviously not a guarantee that Δ is consistent. However, approximate consistency checking is useful because it helps focus where problems possibly lie in Δ, and so prioritize resolution tasks. For example, if Δ and Γ are two parts of a larger knowledgebase that is thought to be inconsistent, and the probability of consistency is much greater for Δ than Γ, then Γ is more likely to be problematical and so should be examined more closely. Similarly, if Δ and Γ are two parts of a larger knowledgebase that is thought to be inconsistent, and a partial consistency identified for Δ is greater than for Γ, then Γ seems to contain more problematical data and so should be examined more closely by the user.

The notions of probable consistency and partial consistency provide complementary means for reasoning with knowledge. For a set of formulae Δ, with partial consistency, we may identify subsets of Δ that are consistent, but have no certainty on whether Δ is consistent, whereas with probable consistency, we can obtain a view on the whole of Δ, but cannot guarantee to find consistent subsets even if they exist.

In this paper, we explore probable consistency checking in more detail. In the next section, we provide some basic definitions, then in the following sections we formalize probable consistency, and give an example in detail.

2 Basic definitions for syntax

In this section, we recall some of the usual definitions for classical logic, and then provide some additional definitions for analysing the syntax of formulae that will be used in the rest of the paper.

Definition 1. *Let $\mathcal{L}_{atoms}$ be a set of* **atoms** *and let $\mathcal{L}$ be the set of* **classical propositional formulae** *formed from $\mathcal{L}_{atoms}$, and the $\wedge, \vee, \rightarrow$ and $\neg$ connectives. For each atom $\alpha \in \mathcal{L}_{atoms}$, α is a* **positive literal** *and $\neg\alpha$ is a* **negative literal**. *Let $\mathcal{L}_{literals}$ be the set of literals in $\mathcal{L}$.*

Example 1. From the propositional atoms α, β and γ, members of $\mathcal{L}$ include $\alpha, \beta \wedge \gamma$, $\neg\alpha \wedge \alpha$ and $(\alpha \wedge \beta) \rightarrow \neg\neg\gamma$.

Definition 2. *For $\alpha_1 \vee .. \vee \alpha_n \in \mathcal{L}$, $\alpha_1 \vee .. \vee \alpha_n$ is a* **clause** *iff each of $\alpha_1, .., \alpha_n$ is a literal. Let $\mathcal{C} \subset \mathcal{L}$ be the set of clauses. Let $\mathcal{C}_i \subset \mathcal{C}$ be the set of clauses of arity i (i.e. the clauses that are a disjunction of i literals).*

So, $\mathcal{C}_1$ is the set of literals, $\mathcal{C}_2$ is the set of binary clauses, and $\mathcal{C}_3$ is the set of ternary clauses.

Definition 3. *Let α be a literal, then α^* is the complementary literal. So if β is an atom, then β^* is $\neg\beta$, and $(\neg\beta)^*$ is β.*

Definition 4. *For clauses, $\alpha \vee \beta$, $\neg\beta \vee \gamma$, $\alpha \vee \gamma \in \mathcal{L}$, $\alpha \vee \gamma$ is a* **resolvent** *of $\alpha \vee \beta$ and $\neg\beta \vee \gamma$.*

Definition 5. *Let the* **atoms function***, denoted* Atoms*, be a function from $\wp(\mathcal{L})$ into $\mathcal{L}_{atoms}$ such that $\mathsf{Atoms}(\Gamma)$ gives the set of atoms used in Γ.*

Example 2. For $\alpha, \beta, \gamma \in \mathcal{L}_{atoms}$, $\mathsf{Atoms}(\{\alpha \wedge \beta\}) = \{\alpha, \beta\}$, and $\mathsf{Atoms}(\{\alpha \wedge \alpha, (\neg\alpha \wedge \beta \wedge \gamma) \rightarrow \alpha\}) = \{\alpha, \beta, \gamma\}$.

Definition 6. *Let $\mathcal{A}$ be a finite subset of $\mathcal{L}_{atoms}$. Let $\mathcal{L}^{\mathcal{A}}$ be the subset of $\mathcal{L}$ where*

$$\mathcal{L}^{\mathcal{A}} = \{\alpha \in \mathcal{L} \mid \mathsf{Atoms}(\{\alpha\}) \subseteq \mathcal{A}\}$$

So, $\mathcal{L}^{\mathcal{A}}_{atoms} = \mathcal{A}$, and $\mathcal{L}^{\mathcal{A}}_{literals} = \mathcal{A} \cup \{\neg\alpha \mid \alpha \in \mathcal{A}\} = \mathcal{C}^{\mathcal{A}}_1$, and $\mathcal{C}^{\mathcal{A}}_i = \mathcal{C}_i \cap \mathcal{L}^{\mathcal{A}}$.

3 Probable consistency for sets of clauses

Suppose a set of formulae Γ is either $\{\alpha \vee \alpha, \neg\alpha \vee \neg\alpha\}$ or $\{\alpha \vee \neg\alpha, \beta \vee \gamma\}$, and it is equally likely that it is either of them, then the probability of Γ being consistent is $1/2$. Using this idea, if we can determine whether a given set of formulae Γ is in a particular class of sets of formulae where the probability of consistency is known, then we have a probability of consistency for Γ. To do this, we need polynomial time tests to analyse each set of formulae. These tests may include a count of the number of propositional letters used in the formulae in the set, a count of the number of formulae in the set, and a count of the number of types of logical symbols used in the formulae in the set. These polynomial time tests delineate classes of sets of formulae. To support this, we need to determine, in advance, the proportion of inconsistent formulae for these classes.

Example 3. Let $\Gamma \in \wp(\mathcal{C})$. If $\mathsf{Atoms}(\Gamma) = \{\alpha\}$, and $|\Gamma| = 2$, where one formula is a positive literal and the other is a negative literal, then the sample space containing Γ has just one element, which is $\{\alpha, \neg\alpha\}$. So the probability that Γ is inconsistent is 1.

Example 4. Let $\Gamma \in \wp(\mathcal{C})$. If $\mathsf{Atoms}(\Gamma) = \{\alpha, \beta\}$, and $|\Gamma| = 2$, where one is a positive literal and the other is a negative literal, then the sample space containing Γ has 2 elements $\{\alpha, \neg\beta\}$ and $\{\beta, \neg\alpha\}$. So the probability that Γ is inconsistent is 0.

In this paper, we restrict consideration to sets of clauses. Informally, given some set of clauses Γ, and the results of some tests on Γ, we want to determine the conditional probability that Γ is inconsistent. For this paper, we will assume a uniform distribution over the sample space containing Γ. Assuming a uniform distribution may be too restrictive for some applications. Alternatives include giving a weighted distribution that is determined by either past usage, or predicted usage, of the formulae, or the propositional letters used in the language. Using the uniform distribution, if $\Theta \subseteq \wp(\mathcal{C})$, and m elements of Θ are inconsistent, and $|\Theta| = n$, then the probability of inconsistency of a randomly selected member Γ of Θ is m/n. Also, the probability of consistency of a randomly selected member Γ of Θ is $1 - m/n$.

Example 5. Consider $\Delta = \{\alpha, \neg\alpha, \beta, \neg\beta\}$. Here the probability of a randomly selected element of $\wp(\Delta)$ being inconsistent is $7/16$.

Definition 7. *Let $\mathcal{Q} = \wp(\mathcal{C})$. Let $\mathcal{Q}_{incon}$ be the class of inconsistent sets in $\mathcal{Q}$ and let $\mathcal{Q}_{mis}$ be the class of minimal inconsistent sets in $\mathcal{Q}$.*

$$\mathcal{Q}_{incon} = \{\Gamma \in \mathcal{Q} \mid \Gamma \vdash \bot\}$$
$$\mathcal{Q}_{mis} = \{\Gamma \in \mathcal{Q}_{incon} \mid \neg\exists\Gamma' \in \mathcal{Q}_{incon} \text{ s.t. } \Gamma' \subset \Gamma\}$$

Suppose $\mathcal{Q}_t \subseteq \mathcal{Q}$. So if by some test t, we show a set Γ is a member of $\mathcal{Q}_t$, then we use the conditional probability statement $P(\mathcal{Q}_{incon} \mid \mathcal{Q}_t)$ to get a valuation of probable consistency/inconsistency for Γ.

Example 6. Let $\mathcal{Q}_{t_1} = \wp(\{\alpha, \neg\alpha, \beta, \neg\beta\})$. So $P(\mathcal{Q}_{incon} \mid \mathcal{Q}_{t_1}) = 7/16$. If we know by test $t1$, $\Gamma \in \mathcal{Q}_{t_1}$, then according to this, the probability of inconsistency for Γ is $7/16$.

Example 7. Let $\mathcal{Q}_{t_2} = \wp(\{\alpha, \neg\alpha\})$. So $P(\mathcal{Q}_{incon} \mid \mathcal{Q}_{t_2}) = 1/4$. If we know by test t_2, $\Gamma \in \mathcal{Q}_{t_2}$, then according to this, the probability of inconsistency for Γ is $1/4$.

Clearly, if $\Gamma \in \mathcal{Q}_t$ and $P(\mathcal{Q}_{incon} \mid \mathcal{Q}_t) = 1$ then $\Gamma \vdash \bot$ holds. Similarly, if for all $\Gamma \in \mathcal{Q}_t$, $\Gamma \vdash \bot$ holds, then $P(\mathcal{Q}_{incon} \mid \mathcal{Q}_t) = 1$.

Definition 8. *Let $\mathcal{Q}_{t_1} \subseteq \mathcal{Q}$ and ... and $\mathcal{Q}_{t_n} \subseteq \mathcal{Q}$. A* **probable consistency system** *is a set Π of conditional probability statements defined as follows, where $p_1, .., p_n \in [0, 1]$.*

$$\Pi = \{P(\mathcal{Q}_{incon} \mid \mathcal{Q}_{t_1}) = p_1, .., P(\mathcal{Q}_{incon} \mid \mathcal{Q}_{t_n}) = p_n\}.$$

The **universe** *for Π is $\mathcal{Q}_{t_1} \cup ... \cup \mathcal{Q}_{t_n}$.*

In general, we may find by using a battery of test functions that a set of clauses Γ is a member of a number of delineated subsets of $\mathcal{Q}$. If we denoted these subsets $\mathcal{Q}_{t_1}, ..., \mathcal{Q}_{t_n}$, then this can be captured by a conditional probability of the form:

$$P(\mathcal{Q}_{incon} \mid \mathcal{Q}_{t_1} \cap ... \cap \mathcal{Q}_{t_n})$$

Couching test functions in terms of observations means that we can undertake a series of tests on a set of clauses, and that the net result is independent of the order in which the tests are done. In other words, we can easily show that all sequences of a set of tests give the same result.

For a probable consistency system Π, and a set of clauses Γ, the conditional probability chosen is the conditional probability with the most specific reference class for the test results.

Definition 9. *If Π is a probable consistency system, then Π_Γ and Π_Γ^{min} are subsets defined as follows, where $\Gamma \in \mathcal{Q}$ and $\mathcal{Q}_{t_i} \subseteq \mathcal{Q}$.*

$$\Pi_\Gamma = \{P(\mathcal{Q}_{incon} \mid \mathcal{Q}_{t_i}) \in \Pi \mid \Gamma \in \mathcal{Q}_{t_i}\}$$

$$\Pi_\Gamma^{min} = \{P(\mathcal{Q}_{incon} \mid \mathcal{Q}_{t_i}) \in \Pi_\Gamma \mid \text{there is no } P(\mathcal{Q}_{incon} \mid \mathcal{Q}_{t_j}) \in \Pi_\Gamma \text{ such that } \mathcal{Q}_{t_j} \subset \mathcal{Q}_{t_i}\}$$

This is used in the following classification, where τ is a fixed threshold such as 0.5.

> *If for all $P(\mathcal{Q}_{incon} \mid \mathcal{Q}_{t_i}) \in \Pi_\Gamma^{min}$, $P(\mathcal{Q}_{incon} \mid \mathcal{Q}_{t_i}) < \tau$ holds, then Γ is* **probably consistent** *according to Π.*

In the same way, for $\Gamma \in \mathcal{Q}_t$, we can have analogous definitions for Γ is probably inconsistent according to Π, and for Γ is probably minimally inconsistent according to Π. The later definition is potentially important in finding faults in knowledgebases and in finding arguments from knowledgebases.

Whilst for small examples, the total cost of using test functions may be greater than undertaking a classical consistency check, in general, they can be cost-effective when used for larger sets of formulae and/or used for more expensive tasks such as searching for minimal inconsistent sets.

4 Case study with binary clauses

Whilst consistency checking for sets of binary clauses is tractable, we will use them to illustrate the probable consistency checking approach. It is simple to check whether a set of formulae is a set of binary clauses. We can define a polynomial test in terms of a small finite state machine. Further simple tests can determine the number of atom symbols used and the number of clauses in a set. The focus of this case study is therefore on various classes of sets of binary clauses to determine the proportion of inconsistent sets within each class, and in particular, the probability that a set of binary clauses is a minimal inconsistent set.

Obviously, if $\Gamma \in \wp(\mathcal{C}_2^{\mathcal{A}})$ is a singleton a set, then the probability that Γ is an minimal inconsistent set is 0. For sets of binary clauses of cardinality greater than 1, we need to consider the nature of minimal inconsistent subsets in a little more detail. In particular, we need to consider the different ways that we can construct conflicting arguments from sets of binary clauses.

4.1 Types of inconsistency in sets of binary clauses

First we consider the types of contradictory arguments that can be constructed from a set of binary clauses.

Definition 10. *A* **resolution sequence** *$\Gamma \in \wp(\mathcal{C}_2^{\mathcal{A}})$ is a sequence of clauses $(\phi_1, ..., \phi_n)$, where $n \geq 1$, such that for each clause ϕ_i (except ϕ_1 and ϕ_n) in the sequence, one of the disjuncts in ϕ_i resolves with one of the disjuncts in the immediate predecessor clause*

ϕ_{i-1} in the sequence, and the other disjuncts in ϕ_i resolves with one of the disjuncts in the immediate successor clause ϕ_{i+1} in the sequence. The literal in ϕ_1 that does not resolve with a literal in ϕ_2 is a **tail**. *The literal in ϕ_n that does not resolve with a literal in ϕ_{n-1} is also a* **tail**.

A resolution sequence $(\phi_1, ..., \phi_n)$ gives a proof of a clause $\alpha \vee \beta$ where α is the tail of ϕ_1 and β is the tail of ϕ_n.

Definition 11. *A* **chain** *Γ is a resolution sequence with tails α and β and there is no subset of $\Delta \subset \Gamma$ such that Δ is a resolution sequence with tails α and β*

Example 8. $(\delta \vee \alpha, \neg\delta \vee \gamma, \beta \vee \neg\gamma)$ is a chain.

A chain $(\phi_1, ..., \phi_n)$ gives a minimal proof of a clause $\alpha \vee \beta$ where α is the tail of ϕ_1 and β is the tail of ϕ_n.

Proposition 1. *Let $\vdash$ be the classical consequence relation. For any $\Gamma \in \wp(\mathcal{C}_2^{\mathcal{A}})$, $\phi \in \mathcal{C}_2^{\mathcal{A}}$, if $\Gamma \vdash \phi$, and $\Gamma \not\vdash \bot$, and there is no $\Gamma' \subset \Gamma$ such that $\Gamma' \vdash \phi$, then Γ is a chain.*

Clearly, $(\phi_1, ..., \phi_n)$ is a chain iff $(\phi_n, ..., \phi_1)$ is a chain.

Definition 12. *An* **isochain** *is a chain $(\phi_1, ..., \phi_n)$ where for ϕ_1 and ϕ_n, there is a disjunct in common. This disjunct is called the* **head** *of the isochain. The other disjunct in ϕ_1 resolves with one of the disjuncts in ϕ_2. The other disjunct in ϕ_n resolves with one of the disjuncts in ϕ_{n-1}.*

An isochain gives a minimal proof of a literal. So for each isochain $(\phi_1, .., \phi_n)$ there is a literal α such that $\{\phi_1, .., \phi_n\} \vdash \alpha$, where α is the head of the isochain.

Example 9. The following are two examples of an isochain. Both have α as head.

$$(\alpha \vee \beta, \neg\beta \vee \gamma, \neg\gamma \vee \delta, \neg\delta \vee \alpha)$$
$$(\beta \vee \alpha, \neg\beta \vee \gamma, \alpha \vee \neg\gamma)$$

Example 10. An isochain can use one or more of the same clauses at the start and end of the chain. In the following the first two clauses and the last two clauses are the same.

$$(\alpha \vee \beta, \neg\beta \vee \gamma, \neg\gamma \vee \delta, \neg\delta \vee \neg\gamma, \neg\beta \vee \gamma, \alpha \vee \beta)$$

Definition 13. *A* **superchain** *is a chain $(\phi_1, ..., \phi_n)$ such that for some i, $(\phi_1, ..., \phi_i)$ is an isochain and $(\phi_{i+1}, ..., \phi_n)$ is a chain and so ϕ_i has a disjunct that resolves with ϕ_{i+1}.*

The isochain gives a minimal proof of a literal β, and the chain gives a minimal proof of a binary clause $\neg\beta \vee \alpha$, and so the superchain gives a minimal proof of α.

Example 11. The following are two examples of a superchain

$$(\beta \vee \beta, \alpha \vee \neg\beta)$$
$$(\neg\delta \vee \neg\delta, \delta \vee \neg\gamma, \neg\beta \vee \gamma, \beta \vee \alpha)$$

Example 12. A superchain can use one or more of the same clauses in the isochain and chain parts of the sequence. Consider the superchain

$$(\alpha \vee \beta, \neg\beta \vee \gamma, \neg\gamma \vee \alpha, \neg\alpha \vee \beta, \neg\beta \vee \gamma, \neg\gamma \vee \delta)$$

which is composed from the isochain $(\alpha \vee \beta, \neg\beta \vee \gamma, \neg\gamma \vee \alpha)$ and the chain $(\neg\alpha \vee \beta, \neg\beta \vee \gamma, \neg\gamma \vee \delta)$ and have the chain $(\neg\beta \vee \gamma)$ in common.

Proposition 2. *Let $(\phi_1, ..., \phi_n)$ be a superchain. If $(\phi_1, ..., \phi_i)$ is an isochain, then $(\phi_{i+1}, ..., \phi_n)$ is not an isochain.*

Proposition 3. *Let $(\phi_1, ..., \phi_n)$ be a chain. If $(\phi_1, ..., \phi_n)$ is an isochain, then $(\phi_1, ..., \phi_n)$ is not a superchain.*

Proof: Let $(\phi_1, ..., \phi_n)$ be an isochain with head α. So α is a tail in ϕ_1 and α is a tail in ϕ_n. Now suppose, $(\phi_1, ..., \phi_n)$ is also a superchain. Then this superchain incorporates an isochain with head α. So, there is a subset of $\{\phi_1, .., \phi_n\}$ that has the same tails as $(\phi_1, .., \phi_n)$. Therefore, $(\phi_1, ..., \phi_n)$ is not a chain. This contradiction means that it cannot be a superchain.$\square$

Definition 14. *For any $\Gamma \in \wp(\mathcal{C}_2^{\mathcal{A}})$, Γ is an* **argument** *for the literal α iff $\Gamma \vdash \alpha$, and there is no $\Gamma' \subset \Gamma$ such that $\Gamma' \vdash \alpha$, and Γ is an isochain or a superchain.*

Definition 15. *Let α be a literal. A* **chainconflict** *is a pair of chains*

$$((\phi_1, ..., \phi_n), (\psi_1, ..., \psi_m))$$

such that $(\phi_1, ..., \phi_n)$ is an argument for α and $(\psi_1, ..., \psi_m)$ is an argument for α^*.

Proposition 4. *For any $\Delta \in \wp(\mathcal{C}_2^{\mathcal{A}})$, if Δ is a minimal inconsistent set then there is a chainconflict $((\phi_1, .., \phi_n), (\psi_1, .., \psi_m))$ such that $\Delta = \{\phi_1, .., \phi_n\} \cup \{\psi_1, .., \psi_m\}$.*

The converse does not hold as illustrated below.

Example 13. Let $(\gamma \vee \gamma, \neg\gamma \vee \alpha)$ be an argument for α, and let $(\neg\gamma \vee \neg\gamma, \gamma \vee \neg\alpha)$ be an argument for $\neg\alpha$. But there is subset of $\{\gamma \vee \gamma, \neg\gamma \vee \alpha\} \cup \{\neg\gamma \vee \neg\gamma, \gamma \vee \neg\alpha\}$ that is inconsistent.

Definition 16. *Let $((\phi_1, ..., \phi_n), (\psi_1, ..., \psi_m))$ be a chainconflict. It is a* **disjoint chainconflict***, if $\{\phi_1, ..., \phi_n\} \cap \{\psi_1, ..., \psi_m\} = \emptyset$, otherwise it is a* **joint chainconflict***.*

Example 14. The following is a disjoint chainconflict with the first item being an argument for α and the second being an argument for $\neg\alpha$.

$$((\alpha \vee \beta, \neg\beta \vee \gamma, \neg\gamma \vee \alpha), (\delta \vee \delta, \neg\delta \vee \neg\alpha))$$

Example 15. The following is a joint chainconflict, with the first item being an argument for α and the second being an argument for $\neg\alpha$, and $(\gamma \vee \beta, \neg\beta \vee \gamma)$ is a subsequence in common with both superchains.

$$((\gamma \vee \beta, \neg\beta \vee \gamma, \neg\gamma \vee \delta, \neg\delta \vee \alpha), (\gamma \vee \beta, \neg\beta \vee \gamma, \neg\gamma \vee \neg\epsilon, \epsilon \vee \neg\alpha))$$

Example 16. The pair $((\alpha \vee \beta, \neg\beta \vee \neg\delta, \delta \vee \alpha), (\neg\alpha \vee \beta, \neg\beta \vee \neg\delta, \delta \vee \neg\alpha)$ is a joint chainconflict. So we have the isochain $(\alpha \vee \beta, \neg\beta \vee \neg\delta, \delta \vee \alpha)$ and the isochain $(\neg\alpha \vee \beta, \neg\beta \vee \neg\delta, \delta \vee \neg\alpha)$. Together they conflict on α with $(\neg\beta \vee \neg\delta)$ being the chain in common.

We will use this characterization of isochains and superchains in joint and disjoint chainconflicts to enumerate the possible minimal inconsistent sets of binary clauses.

4.2 Combinatorics of sets of binary clauses

We now consider the combinatorics for constructing chainconflicts, and thereby gain the proportion of sets of binary clauses that are minimal inconsistent sets. Our approach here is to consider the formats for minimal inconsistent sets of binary clauses of various cardinalities.

Definition 17. *A* **clause scheme** *is one of the following, where $i, j \in \mathbb{N}$, and X_i and X_j are meta-variables symbols (place holders for literals), and X_i^* (respectively X_j^*) has to be instantiated with the complement of X_i (respectively X_j).*

$$X_i \vee X_j \qquad X_i \vee X_j^* \qquad X_i^* \vee X_j \qquad X_i^* \vee X_j^*$$

Definition 18. *A* **grounding** *is an assignment of a literal to a meta-variable where the meta-variable is given on the left-hand-side of the $=$ symbol and the literal is in the right-hand-side. A* **grounding set** *is a set of groundings where each meta-variable occurs at most once, and each meta-variable is ground with a different atom symbol in the literal (i.e. for all grounding sets if $X_i = \alpha_i$ and $X_j = \alpha_j$ and $i \neq j$, then $\mathsf{Atoms}(\{\alpha_i\}) \neq \mathsf{Atoms}(\{\alpha_j\})$).*

Definition 19. *Let Φ be a clause scheme, and G be a grounding. The $\mathsf{Instantiate}(\Phi, G)$ function uses the groundings in G to instantiate the meta-variables in Φ. The result is an* **instantiation** *of Φ.*

When a set of clause schema is instantiated, we obtain a clause from each clause scheme.

Example 17. Let $\{X_1 \vee X_2, X_2^* \vee X_3, X_2 \vee X_4^*\}$ be a set of clause schema, and let $\{X_1 = \neg\alpha, X_2 = \neg\beta, X_3 = \delta, X_4 = \gamma\}$ be a grounding set. Then we obtain the set of clauses $\{\neg\alpha \vee \neg\beta, \beta \vee \delta, \neg\beta \vee \neg\gamma\}$ which is an instantiation.

Clause schema are used to define a set of chainconflicts.

Definition 20. *A* **conflict scheme** *is a set of clause schema $\{\Phi_1, .., \Phi_n\}$, such that if there is a grounding set G that can instantiate each of these clause schema, then $\mathsf{Instantiate}(\Phi_1, G) \cup .. \cup \mathsf{Instantiate}(\Phi_n, G)$ is a minimal inconsistent set.*

Example 18. The set $\{X_1 \vee X_1, X_1^* \vee X_1^*\}$ is a conflict scheme. If we let $X_1 = \alpha$, where $\alpha \in \mathcal{A}$, then we obtain $\{\alpha \vee \alpha, \neg\alpha \vee \neg\alpha\}$, which is a minimal inconsistent set.

We also require some subsidiary definitions.

Definition 21. *Let Φ be a set of clause schema. The function* $\mathsf{Letters}(\Phi)$ *gives the number of different meta-variable symbols used in Φ.*

Example 19. Let $\Phi = \{X_1 \vee X_2, X_2^* \vee X_3, X_2 \vee X_4^*\}$. Then $\mathsf{Letters}(\Phi) = 4$.

Definition 22. *Let Φ be a set of clause schema. The function* $\mathsf{Heteroclauses}(\Phi)$ *gives the* **number of schema** *in Φ where the first disjunct has a different meta-variable symbol to the second disjunct.*

Example 20. Let $\Phi = \{X_1 \vee X_2, X_2^* \vee X_3, X_2 \vee X_2^*\}$. Then $\mathsf{Heteroclauses}(\Phi) = 2$. Here the first and second disjuncts of $X_2 \vee X_2^*$ have the same meta-variable symbol.

Some conflict schema are symmetrical in the sense that there are n different grounding sets for each instantiation. For instance, for $\{X_1 \vee X_1, X_1^* \vee X_1^*\}$, there are two groundings for the each instantiation. So for the instantiation $\{\alpha \vee \alpha, \neg\alpha \vee \neg\alpha\}$, we have $G_1 = \{X_1 = \alpha\}$ and $G_2 = \{X_1 = \neg\alpha\}$.

Definition 23. *Let Φ be a set of clause schema. The function* $\mathsf{Symmetry}$ *is defined as follows: If there are n different grounding sets $G_1, .., G_n$ such that* $\mathsf{Instantiate}(\Phi, G_1) = .. = \mathsf{Instantiate}(\Phi, G_n)$*, for each instantiation of Φ, then* $\mathsf{Symmetry}(\Phi) = 1/n$.

Example 21. To illustrate the $\mathsf{Symmetry}$ function, consider the following:

$$\mathsf{Symmetry}(\{X_1 \vee X_2, X_2^* \vee X_3, X_2 \vee X_4^*\}) = 1$$
$$\mathsf{Symmetry}(\{X_1^* \vee X_1, X_1 \vee X_1^*\}) = 1/2$$
$$\mathsf{Symmetry}(\{X_1 \vee X_2, X_2^* \vee X_1, X_1^* \vee X_2, X_2^* \vee X_1^*\}) = 1/2$$

Now we consider how we get the number of chainconflicts from a conflict scheme and a set of atoms.

Definition 24. *For a conflict scheme Φ, the* **number of chainconflicts** *that can be obtained with a set of atoms $\mathcal{A}$ is determined by the function* $\mathsf{Conflictcount}(\Phi, \mathcal{A})$. *We assume a chainconflict $((\phi_1, .., \phi_n), (\psi_1, .., \psi_m))$ is the same as $((\psi_1, .., \psi_m), (\phi_1, .., \phi_n))$ and so would count them only once.*

Proposition 5. *Let $f = \mathsf{Heteroclauses}(\Phi) - 1$, if $\mathsf{Heteroclauses}(\Phi) \neq 0$, and $f = 0$ otherwise, $g = \mathsf{Symmetry}(\Phi)$, $h = \mathsf{Letters}(\Phi)$, and $a = |\mathcal{A}|$. Hence*

$$\mathsf{Conflictcount}(\Phi, \mathcal{A}) = \frac{a!}{(a-h)!} \times 2^h \times 2^f \times g \textit{ if } a \geq h$$

$$\mathsf{Conflictcount}(\Phi, \mathcal{A}) = 0 \textit{ if } a < h$$

Proof: For $a < h$, there are insufficient atom symbols in $\mathcal{A}$ to form a grounding set, and so there are 0 instantiations. For $a \geq h$, we need to justify the four terms as follows. The first term gives all permutations of the a atom symbols in $|\mathcal{A}|$ used to instantiate the sequence of h atomic meta-variable symbols in Φ. The second term gives the number of choices of positive and negative literals for the sequence of h atomic meta-variable symbols in Φ. The third term gives the permutations for each disjunct in each clause being the first or second disjunct in the heterogeneous clauses except for the first heterogeneous clause. The fourth term eliminates the duplicate counts obtained in the second and third terms in cases of symmetry. □

Definition 25. *A* **conflict profile***, denoted Ψ, of degree k is a set of conflict schema where for all $\Phi \in \Psi$, $|\Phi| = k$, and for each minimal inconsistent set $\Gamma \in \wp(\mathcal{C}_2^{\mathcal{A}})$, if $|\Gamma| = k$, then there is exactly one conflict scheme $\Phi' \in \Psi$ such that an instantiation of Φ' is Γ.*

Conflict profiles for degrees 2 to 4, are given in Tables 1 to 3.

Proposition 6. *The number of minimal inconsistent sets of cardinality k in $\wp(\mathcal{C}_2^{\mathcal{A}})$ is calculated as follows, where $\Psi = \{\Phi_1, .., \Phi_n\}$ is a conflict profile of degree k:*

$$\sum_{i=1}^{n} \mathsf{Conflictcount}(\Phi_i, \mathcal{A})$$

Proof: Each chainconflict corresponds to a minimal inconsistent set. According to Definition 25, the set of minimal inconsistent sets generated by each conflict scheme is disjoint from those generated by the other conflict schema. Therefore, the number of minimal inconsistent sets of cardinality k is the sum of the minimal inconsistent sets generated by each conflict scheme. □

Example 22. Let Φ_{2a} be defined by 2a in Table 1. Let $|\mathcal{A}| = 5$. Since $a = 5$, $f = 0$, $g = 1/2$, and $h = 1$, we have the following, and hence the number of minimal inconsistent sets of cardinality 5 in $\wp(\mathcal{C}_2^{\mathcal{A}})$ is 5.

$$\mathsf{Conflictcount}(\Phi_{2a}, \mathcal{A}) = \frac{5!}{(5-1)!} \times 2^1 \times 2^0 \times 1/2 = 5$$

Example 23. Let $\Phi_{3a}, .., \Phi_{3c}$ be defined by 3a,..,3c in Table 2. Let $|\mathcal{A}| = 2$. The number of minimal inconsistent sets of cardinality 3 in $\wp(\mathcal{C}_2^{\mathcal{A}})$ is 40, as follows.

For $\Phi_{3a}, f = 1, g = 1,$ and $h = 2,$ so $\mathsf{Conflictcount}(\Phi_{3a}, \mathcal{A}) = 16$
For $\Phi_{3b}, f = 0, g = 1,$ and $h = 2,$ so $\mathsf{Conflictcount}(\Phi_{3b}, \mathcal{A}) = 8$
For $\Phi_{3c}, f = 1, g = 1,$ and $h = 2,$ so $\mathsf{Conflictcount}(\Phi_{3c}, \mathcal{A}) = 16$

It is straightforward to obtain conditional probabilities of inconsistency using the count of minimal inconsistent sets.

Proposition 7. *Let $a = |\mathcal{A}|$. The number of sets of cardinality k in $\wp(\mathcal{C}_2^{\mathcal{A}})$ is the following number of combinations of sets of size k, where $n = (2a)^2$ is the number of binary clauses that can be formed from the $2a$ literals obtained from $\mathcal{A}$.*

$$\frac{n!}{k!(n-k)!}$$

Proposition 8. *The probability that any set of cardinality k is a minimal inconsistent set given by the following ratio, where the numerator is obtained from Proposition 6 and the denominator is obtained from Proposition 7.*

$$\frac{\textit{number of min inconsistent sets of cardinality } k \textit{ in } \wp(\mathcal{C}_2^{\mathcal{A}})}{\textit{number of sets of cardinality } k \textit{ in } \wp(\mathcal{C}_2^{\mathcal{A}})}$$

This case study is intended to indicate the promise of probable consistency checking. In order to count larger minimal inconsistent sets in $\wp(\mathcal{C}_2^{\mathcal{A}})$, we need to generate conflict profiles of degree greater than 4. In order to render this viable, we are currently exploring the development of an algorithm to generate conflict profiles. In parallel, we are seeking more general results that would enable the counting of minimal inconsistent sets in $\wp(\mathcal{C}_2^{\mathcal{A}})$, and in $\wp(\mathcal{C}_i^{\mathcal{A}})$ for $i > 2$, more directly.

	Conflict scheme	Type
2a	$X_1 \vee X_1$ $X_1^* \vee X_1^*$	disjoint isochains

Table 1. A conflict profile of degree 2 is composed of conflict scheme 2a. The first line is for the first argument and the second line is for the second argument.

	Conflict scheme	Type
3a	$X_1 \vee X_2, X_2^* \vee X_1$ $X_1^* \vee X_1^*$	disjoint isochains
3b	$X_1 \vee X_1, X_1^* \vee X_2$ $X_2^* \vee X_2^*$	disjoint isochain + superchain
3c	$X_1 \vee X_1, X_1^* \vee X_2$ $X_1 \vee X_1, X_1^* \vee X_2^*$	joint superchains

Table 2. A conflict profile of degree 3 is composed of conflict schema 3a - 3c.

	Conflict scheme	Type
4a	$X_1 \vee X_2, X_2^* \vee X_3, X_3^* \vee X_1$ $X_1^* \vee X_1^*$	disjoint isochains
4b	$X_1 \vee X_2, X_2^* \vee X_1, X_1^* \vee X_3$ $X_3^* \vee X_3^*$	disjoint isochain + superchain
4c	$X_1 \vee X_2, X_2^* \vee X_1$ $X_1^* \vee X_3^*, X_3 \vee X_1^*$	disjoint isochains
4d	$X_1 \vee X_2, X_2^* \vee X_1$ $X_1^* \vee X_2^*, X_2 \vee X_1^*$	disjoint isochains
4e	$X_1 \vee X_1, X_1^* \vee X_2$ $X_3 \vee X_3, X_3^* \vee X_2^*$	disjoint superchains
4f	$X_1 \vee X_1, X_1^* \vee X_2, X_2^* \vee X_3$ $X_1 \vee X_1, X_1^* \vee X_3^*$	joint superchains

Table 3. A conflict profile of degree 4 is composed of conflict schema 4a - 4f.

5 Discussion

Consistency checking is increasingly important in artificial intelligence, data and knowledge engineering, and software engineering. In approaches to knowledge representation and reasoning such as truth maintenance systems, argumentation systems based on identifying consistent subsets, and default reasoning systems, consistency checking is an integral part of inferencing. Probable consistency checking potentially offers more efficient inferencing, either by directing conflict resolution to the more problematical areas of the data, or by allowing inferences before eliminating all possibilities of an inconsistency.

More generally, probable consistency checking is potentially important in information integration, requirements engineering, negotiation, and multi-agent interaction. It seems that the real advantage with probable consistency checking in all these activities is the relative ordering we can obtain rather than the absolute probability values. The relative ordering helps prioritize search or further analysis.

Of course, none of these probabilities take into account psychological or cognitive factors in developing or handling knowledgebases. For example, psychological observations may reveal an increased error resulting from an increased complexity of a specification. If such an observation were sufficiently precise, then perhaps there should be some weighting applied to the probability of inconsistency of formulae so that larger formulae are more likely *a priori* to be inconsistent.

References

1. A Darwiche. Compiling knowledge into decomposible negation normal form. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 284–289, 1999.
2. R Dechter and J Pearl. Network-based heuristics for constraint-satisfaction problems. *Artificial Intelligence*, 34:1–38, 1987.
3. M Garey and D Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, 1979.
4. F Koriche. On anytime coherence-based reasoning. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty (Ecsqaru'01)*, volume 2143 of *Lecture Notes in Computer Science*, pages 556–567. Springer, 2001.
5. H Levesque. A logic of implicit and explicit belief. In *Proceedings of the National Conference on Artificial Intelligence (AAAI'84)*, pages 198–202, 1984.
6. P Marquis. Knowledge compilation using prime implicates. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 837–843, 1995.
7. F Oppacher and E Suen. HARP: A tableau-based theorem prover. *Journal of Automated Reasoning*, 4:69–100, 1988.
8. C Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
9. M Schaerf and M Cadoli. Tractable reasoning via approximation. *Artificial Intelligence*, 74:249–310, 1995.
10. B Selman, H Levesque, and D Mitchell. A new method for solving hard satisfiability problems. In *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI'92)*, pages 440–446, 1992.