Model-based Algorithm for Belief Revisions between Normal Conjunctive Forms

Alma Delia García, Guillermo De Ita, Fernando Zacarias

Benemerita Universidad Autonoma de Puebla, Computer Sc. Dpt., Puebla, México
deliakzy@gmail.com, deita@cs.buap.mx, fzflores@yahoo.com.mx

Abstract. We consider a knowledge base (KB) K and a new information ϕ, both expressed in conjunctive form (CF), and present here, a novel, deterministic and correct algorithm for belief revision of ϕ in K. We denote our revision operator as: $K' = K \circ \phi$. We introduce a novel logical binary operator Ind between two conjunctive forms, such that $Ind(\phi, K)$ generates also a conjunctive form. The operator $Ind(\phi, K)$ works building independent clauses with the clauses of K, and whose falsifying assignments of the resulting formula cover exactly the space of assignments $Fals(\phi) - Fals(K)$, this is essential for performing the process of belief revision $K' = K \circ \phi$, and where $K' \models \Phi$. Furthermore, our proposal satisfies the KM postulates. We also present the correctness proof of our belief revision method, and the analysis of its time complexity.

Keywords: Propositional Inference, Belief Revision, Model Based Inference, Postulates KM.

1 Introduction

A widely accepted reference framework for reasoning in intelligent systems is the systems approach based on knowledge bases. In this case, the sentences are stored in a Knowledge Base (KB) provided with a reasoning mechanism [5]. A fundamental challenge of these systems is the automation of deductive reasoning from the KB. Propositional deductive reasoning is generally summarized as follows: given a KB K, which contains knowledge about a domain ("the world"), and ϕ a statement representing the query that captures the current situation, both expressed in propositional logic, the goal is to decide whether K implies ϕ (in symbols: $K \models \phi$), which is known as the *problem of propositional entailment.*

The propositional implication is an important task in problems such as estimating the degree of belief revision and updating of beliefs, working with abductive explanations, and many other procedures in applications of Artificial Intelligence. For example, when working in planning, designing multi-agent systems, logical diagnosis, approximate reasoning, among other applications [4], [7]. In general, the problem of logical implication is a difficult challenge in the area of automatic reasoning, and turns out to be a problem in the class Co-NP complete, even in the propositional case [9].

Belief revision problem consists in incorporate new beliefs to a knowledge base (KB) already established, changing as little as possible the original beliefs and maintaining consistency of the KB. This article shows that the use of falsifying patterns clauses help to determine whether an conjunctive form (CF) is inferred from another CF, and therefore, it allows us to construct an algorithm for belief revision between CF's.

In short, the main contributions of our work are:

- We propose a method that works on the set of falsifying assignments of the involved formulas, in order to review: $K \models \Phi$.
- We introduce a logical operator between two terms, $Ind(\varphi_i, C_j)$, resulting in a CF Fs, such that $Fals(Fs) = Fals(\varphi_i) - Fals(C_j)$.
- We show that our proposal for the belief revision is correct, and holds the Katsuno and Mendelzon postulates.
- $Ind\ operator(\varphi_i, C_j)$ was implemented to work in linear time on number of variables involved, and it is the base of our procedure for belief revision.
- Although the operator $Ind(\varphi_i, C_j)$ performs efficiently, the number of clauses in Fs, that allows $(K \wedge Fs) \models \phi$, can lead to an exponential growth (the order $O(2^{(n-min\{|\varphi_i|:\varphi_i \in \phi\})})$).

1.1 State of the Art

The paradigm best known for belief revision is the AGM paradigm [1]. Subsequently, Mendelzon and Katsuno [8] unified the different belief revision approaches to semantic, and reformulated the AGM postulates, which were called now, KM postulates. Subsequently, Darwiche and Pearl [3] proposed the iterated revision, where his proposal establishes a representation based on model assumptions.

There are some proposals for belief revisions based on models, and they are identified by the name of their authors; Dalal, Satoh, Winslett, Borguida and Forbus [11]. Dalal [2] operator suggests to review, based on the minimum Hamming distance between satisfied interpretations, and extend the distance between interpretations and bases. In practice, this proposal involves the calculation of the set of models, which is very expensive. One of the drawbacks of the approach from Dalal is limited to the case of consistent knowledge bases. Therefore in [13] is proposed a new method for computing Dalal's revision that avoids the computation of belief bases models, but it works only on disjunctive normal forms.

The Satoh's proposal [14] is similar to Dalal, with the difference that the distance between two models is defined as the set of different literals between both models. In the case of Winslett, the proposal is based on a comparison between all possible consistent maximum systems. The Borguida and Forbus' proposals are similar of the Winslett, with the difference that Borguida considers incompatible models, and Forbus use the Hamming distance. More recently, the belief revision has gained attention in the framework of symbolic logic, including Horn fragments, and many operators for belief revision have been proposed according to syntactic or semantic points of view [10, 12, 14].

2 Preliminaries

Let $X = \{x_1, \ldots, x_n\}$ be a set of n Boolean variables. A *literal* denoted as *lit* is, a variable x_i or a denied variable $\neg x_i$. As usual, each $x \in X$, $x^0 = \neg x$ y $x^1 = x$. A *clause* is a disjunction of different literals. For $k \in N$, a *k-clauses* is a clause with exactly k literals, and $(\leq k)$-*clause* is a clause with at most k literals. A *phrase* is a conjunction of literals. A *k-phrase* is a phrase with exactly k literals. A variable $x \in X$ appears in a clause C (or phrase) if x or $\neg x$ is an element of C.

A *conjunctive normal form* (CF) is a conjunction of clauses, and *k-CF* is a CF containing only *k-clauses*. A *disjunctive normal form* (DF) is a disjunction of sentences, and *k-DF* is a DF containing only *k-phrases*. A CF F with n variables is a *n-ary* Boolean function F:$\{0,1\}^n \rightarrow \{0, 1\}$. Rather, any Boolean function F has infinitely many equivalent representations, among these, some in CF and DF.

We denote with Y any of the basic logic elements that we are using, such as a literal, a clause, a phrase, a DF or a CF. v (Y) denotes the set of variables involved in the object Y. For example, $v(\neg x_1 \vee x_2) = \{x_1, x_2\}$. While $lit(Y)$ denotes the set of literals involved in object Y. For example, if $X = v(Y)$ then $lit(Y) = X \cup \neg X = \{x_1, \neg x_1, \ldots, x_n, \neg x_n\}$. We also use $\neg Y$ as the negation operator on the object Y. We denote to {1, 2, 3, ...,n} by [[1, n]], and the cardinality of a set A by $|A|$.

An *assignment* s for a formula F is a Boolean mapping $s : v(F) \rightarrow \{1, 0\}$. An assignment s can also be considered as a non-complementary set of literals: $l \in s$ if and only if s assigns true to l and $\neg l$ false. s is a partial *assignment* for the formula F when s has determined a logical value only to variables of a proper subset of F, namely $s : Y \rightarrow \{1, 0\}$ and $Y \subset v(F)$. A CF F is satisfied by an assignment s if each clause F is satisfied by s; A model of F is an assignment on $v(F)$ satisfying F.

A phrase f is satisfied by an assignment s, if $f \subseteq s$. Otherwise, s falsifies f. A DF F is satisfied by s if a sentence in F is satisfied by s. F is contradicted by s if all sentences in F are contradicted by s. Denote by $SAT(F)$ to the set of assignment in $S(F)$ which they are models for F. $Fals(F)$ denotes the set of assignments in $S(F)$ that falsifies F.

3 Inference Between Conjunctive Forms

In this section we analyze the computational complexity of the entail problem between conjunctive forms: CF $\models$ CF. As K and ϕ are in CF, the falsifying assignments $Fals(K)$ y $Fals(\phi)$ can be calculated efficiently [5]. Using falsifying strings as the base for reviewing $K \models \phi$, that is equivalent to check whether $SAT(K) \subseteq SAT(\phi)$, or that: $Fals(\phi) \subseteq Fals(K)$. The result of applying the operator of belief revision between a KB K and the new evidence ϕ is denoted as $K' = K \circ \phi$.

When $K \models \phi$ then $K' = K \circ \phi = K$. If $K \not\models \phi$ then $Fals(\phi) \not\subset Fals(K)$, which implies that there is a set of assignments S such that $S \subseteq Fals(\phi)$ y $S \not\subseteq$

$Fals(K)$. If $K \not\models \phi$, then $S = (Fals(\phi)$ - $Fals(K)$ $) \neq \emptyset$. In this case, our belief revision method works building such set S, which allows to build a new CF Fs, such that $S = Fals(Fs)$ and $K' = (K \wedge Fs)$, and it holds $K \models \phi$.

The proposed method obtains $S = (Fals(\phi) - Fals(K))$ as a set of falsifying strings, leading us to build a CF Fs directly, where $S = Fals(Fs)$ and such that $K' = K \wedge Fs$ is a new CF with less information than K (because K' has more clauses than K), in fact, it holds that if $S \neq \emptyset$ then $Fals(K) \subset Fals(K')$, and therefore, $SAT(K') \subset SAT(K)$.

3.1 Construction of Independent Sets of Clauses

Let K be a CF, i.e., $K = \bigwedge_{i=1}^{m} C_i$, where each $C_i, i = 1, \ldots, m$ is a disjunction of literals. The set of assignments forming $Fals(C_i)$ can be represented through a string A_i that consists of {0, 1, *}. If $C_i = \{x_{i_1} \vee \cdots \vee x_{i_k}\} \in K$, then the value; for each position from $i_1 - th$ to i_{k-th} of the string A_i, has to falsify the literals of C_i. If $x_{i_j} \in C_i$ then the $i_j - th$ element of A_i is set to 0. If $\neg x_{i_j} \in C_i$ then the i_{j-th} element of A_i is set to 1. The variables in $\upsilon(K)$ which do not appear in C_i are represented by the symbol *, meaning that they could take any logical value {0, 1} . In this way, the string A_i of length $n = |\upsilon(K)|$ represents the set of assignments falsifying the clause C_i. We will denote the formation of the string representing $Fals(C_i)$ as $A_i = string(Fals(C_i))$. It is easy to build $Fals(K)$ since each clause C_i determines a subset of falsifying assignments of K. The following lemma expresses how to form the falsifying set of assignments of a CF.

Lemma 1. *Given a CF* $K = \bigwedge_{i=1}^{m} C_i$, *which holds* $Fals(K) = \bigcup_{i=1}^{m} \{\sigma \in S(K) \mid Fals(C_i) \subseteq \sigma\}$

Definition 1. *[6] Given two C_i and C_j clauses, if they have at least one complementary literal, will be called independent clauses. Otherwise, it is said that both are dependent clauses.*

Two independent clauses C_i and C_j have complementary pair of literals, therefore their falsifying assignments also must have complementary literals, that is, $Fals(C_i) \cap Fals(C_j) = \emptyset$.

Definition 2. *Let $K = \{C_1, C_2, \ldots, C_m\}$ be a CF. K is called independent if for any pair of clauses $C_i, C_j, \in K$, $i \neq j$, the property of independence is met.*

Definition 3. *Given two falsifying strings A and B each of length n, if there is an $i \in \{1, \ldots, n\}$ such that $A[i] = x$ and $B[i] = 1 - x$, $x \in \{0, 1\}$, it is said that they have the* independence *property. Otherwise, we say that both strings are* dependent.

Notice that falsifying strings for independent clauses have complementary values (0 and 1) in at least one of their fixed values.

Let $F = \{C_1, C_2, \cdots, C_m\}$ be a CF, $n = |\upsilon(F)|$. Let $C_i, i \in [[i..m]]$ be a clause in F and $x \in \upsilon(F) \setminus \upsilon(C_i)$ be any variable, we have that

$$C_i \equiv (C_i \vee \overline{x}) \wedge (C_i \vee x) \tag{1}$$

Furthermore, this reduction preserves the number of falsifying assignments of C_i with respect to F, since $\#FAL(C_i) = 2^{n-|C_i|} = 2^{n-(|C_i|+1)} + 2^{n-(|C_i|+1)} = \#FAL((C_i \vee \overline{x}) \wedge (C_i \vee x))$, because $(C_i \vee \overline{x})$ and $(C_i \vee x)$ are two independent clauses.

Definition 4. *Given a pair of dependent clauses C_1 and C_2, if $lit(C_1) \subseteq lit(C_2)$ we say that C_2 is subsumed by C_1.*

If C_1 subsumes C_2 then $FAL(C_2) \subseteq FAL(C_1)$. On the other hand, if C_2 is not subsumed by C_1 and they are dependents, there is a set of indices $I = \{1, \ldots, p\} \subseteq \{1, \ldots, n\}$ such that for each $i \in I, x_i \in C_1$ but $x_i \notin C_2$. There exists a reduction to transform C_2 to be independent with C_1, we call this transformation as the *independent reduction* between two clauses that works as follows: let C_1 and C_2 be two dependent clauses. Let $\{x_1, x_2, \ldots, x_p\} = Lit(C_1) \setminus Lit(C_2)$. By (1) we can write: $C_1 \wedge C_2 = C_1 \wedge (C_2 \vee \neg x_1) \wedge (C_2 \vee x_1)$. Now C_1 and $(C_2 \vee \neg x_1)$ are independent. Applying (1) to $(C_2 \vee x_1)$:

$$C_1 \wedge C_2 \equiv C_1 \wedge (C_2 \vee \neg x_1) \wedge (C_2 \vee x_1 \vee \neg x_2) \wedge (C_2 \vee x_1 \vee x_2)$$

The first three clauses are independent. Repeating the process of making the last clause independent with the previous ones, until x_p is considered; we have that $C_1 \wedge C_2$ can be written as:
$C_1 \wedge (C_2 \vee \neg x_1) \wedge (C_2 \vee x_1 \vee \neg x_2) \wedge \ldots \wedge (C_2 \vee x_1 \vee x_2 \vee \ldots \vee \neg x_p) \wedge (C_2 \vee x_1 \vee x_2 \vee \ldots \vee x_p)$.

The last clause contains all literals of C_1, so it is subsumed by C_1, and then

$$C_1 \wedge C_2 \equiv C_1 \wedge (C_2 \vee \neg x_1) \wedge (C_2 \vee x_1 \vee \neg x_2) \wedge \ldots \wedge (C_2 \vee x_1 \vee x_2 \vee \ldots \vee \neg x_p) \tag{2}$$

We obtain on the right side of (2) an independent set of $p+1$ clauses which we denote as $indep\_reduction(C_1, C_2)$.

We will use the independent reduction between two clauses C_1 and φ (or between their respective falsifying strings) to define:

$$Ind(C, \varphi) = \begin{cases} \varphi & \text{If } \varphi \text{ and C are independent} \\ \emptyset & \text{If Lit}(C) \setminus \text{Lit}(\varphi) = \emptyset \\ indep\_reduction(C, \varphi) - C & \text{in other case} \end{cases} \tag{3}$$

It is straightforward to redefine the operator Ind in terms of the falsifying strings representing $FAL(C)$ and $FAL(\varphi)$. The operation $Ind(C, \varphi)$ forms a conjunction of clauses whose falsifying assignments are exactly $FAL(\varphi) - FAL(C)$.

Theorem 1. *If φ and C are two clauses, then $FAL(Ind(C, \varphi)) = FAL(\varphi) - FAL(C)$.*

Proof. If $Ind(C,\varphi) = \emptyset$ then $FAL(\varphi) \subseteq FAL(C)$, so $FAL(\varphi) \setminus FAL(C) = \emptyset$. Now, we assume that $Ind(C,\varphi) \neq \emptyset$. Let s be an assignment such that $s \in FAL(Ind(C,\varphi))$. We will show that $s \in FAL(\varphi)$ and $s \notin FAL(C)$. If $s \in FAL(Ind(C,\varphi))$ then s falsifies φ because each clause in $Ind(C,\varphi)$ has the form $(\varphi \vee R)$, where R is a disjunctive set of literals (possibly R is empty). If s falsifies $(\varphi \vee R)$ then s has to falsify φ and thus $s \in FAL(\varphi)$. On the other hand, each clause $(\varphi \vee R) \in Ind(C,\varphi)$ is independent to C by construction of the operator Ind; therefore, $FAL(C) \cap FAL(Ind(C,\varphi)) = \emptyset$. Furthermore, $s \notin FAL(C)$.

4 Belief Revision Between Conjunctive Forms

Our belief revision method is based on the following two properties:

1. If $\forall\ s \in Fals(\phi)$ holds that $s \in Fals(K)$, then $K \models \phi$.
2. If $\exists\ s \in Fals(\phi)$, and $s \notin Fals(K)$, then $K \not\models \phi$.

The first case considers all assignments $Fals(\phi)$ are in the set $Fals(K)$, which show that $K \models \phi$. And in this case $K' = K$, since no need to change the KB K. In the second case, the sets of assignments S such that $\exists \varphi \in \phi$, $S \subseteq Fals(\varphi)$ and $S \not\subseteq Fals(K)$ are detected.

Algorithm 1 Procedure $Ind(\varphi_i, K)$

```
Input: K: A KB, φ_i: Clause with new information
Push(φ_i, V); Fs = ∅; {Output in Fs a CF (Set of clauses)}
for all C_j ∈ K do
  while (V ≠ ∅) do
    φ = Pop(V); {Try following clause}
    Fs = Fs − φ; {Remove the exit clause}
    Nc = Ind(φ, C_j); {Form: Nc ∧ C_j ⊨ φ }
    if (Nc ≠ ∅) then
      Fs = Fs ∪ Nc; {Only if there are clauses to add}
    end if
  end while
  V = Fs; {Next iteration considers new clauses}
end for
Return(Fs)
```

Example 1. Let $K = (\neg p \vee q \vee s) \wedge (\neg q \vee \neg r \vee s) \wedge (\neg q \vee r \vee \neg s) \wedge (\neg p \vee \neg q \vee r)$ and $\phi = (\neg p \vee \neg r) \wedge (\neg q \vee r) \wedge (p \vee q \vee \neg r \vee \neg s) \wedge (\neg t)$. To prove $K \models \phi$, it is equivalent to check $Fals(\phi) = \{1{*}1{**}, {*}10{**}, 0011{*}, {****}1\} \subseteq Fals(K) = \{10{*}0{*}, {*}110{*}, {*}101{*}, 110{**}\}$. In each cell of column 2 of the table 1, it will show the result of $Ind(\varphi_i, C_j)$.

For example, let $C_i = (x \vee q)$, and $C_j = (\neg x \vee q)$, then $C_i \wedge C_j = (q)$. In terms of the falsifying strings clauses, we denote such a reduction as $Varcom(A_i, A_j)$. In

Table 1. Building $Ind(\phi, K)$ operator

ϕ \ K	10*0*	*110*	*101*	110**	***S***
1*1**	111**	1111*	1111*	1111*	**1111***
	1011*	1011*	1011*	1011*	**1011***
*10**	*10**	*10**	*100*	0100*	**0100***
0011*	0011*	0011*	0011*	0011*	**0011***
****1	0***1	00**1	00**1	00**1	**00**1**
		010*1	01001	01001	**01001**
		01111	01111	01111	**01111**
	11**1	110*1	11001	$\emptyset$	$\emptyset$
		11111	11111	11111	**11111**
	10*11	10*11	10*11	10*11	**10*11**

the case of our example, we have: $Varcom${1111*,1011*} = {1*11*}. It is relevant to apply the reduction operation by complementary literals on the strings in S, for minimizing the total number of clauses. Thus, $Varcom$ and subsumed clauses are applied in order to reduce the resulting CF, from the example 1, we have:

S = {1*11*, 0100*, 0011*, 0***1, *1111, 10*11}. Writing S as CF, $Fs = (\neg p \vee \neg r \vee \neg s) \wedge (p \vee \neg q \vee r \vee s) \wedge (p \vee q \vee \neg r \vee \neg s) \wedge (p \vee \neg t) \wedge (\neg q \vee \neg r \vee \neg s \vee \neg t) \wedge (\neg p \vee q \vee \neg s \vee \neg t)$. So, the new KB $K' = K \wedge Fs$ holds $K' \models \phi$.

5 Method Properties of Belief Revision

Theorem 2. *Given two clauses φ_i and C_j, it holds that $(C_j \wedge Ind(\varphi_i, C_j)) \models \varphi_i$.*

Proof. If φ_i and C_j are independent clauses, then $\varphi_i = Ind(\varphi_i, C_j)$ and therefore $(C_j \wedge Ind(\varphi_i, C_j)) = (C_j \wedge \varphi_i)$. So $(C_j \wedge \varphi_i) \models \varphi_i$, propositional property by: $((p \wedge q) \supset q$ and the reflexivity of the logical inference: $\varphi_i \models \varphi_i$i. If φ_i and C_j are not independent, but $Ind(\varphi_i, C_j) = \emptyset$, this implies that $Fals(\varphi_i) \subseteq Fals(C_j)$ and such $C_j \models \varphi_i$. As $C_j = (C_j \wedge Ind(\varphi_i, C_j))$, then $(C_j \wedge Ind(\varphi_i, C_j)) \models \varphi_i$. When φ_i and C_j are not independent, and $Ind(\varphi_i, C_j) \neq \emptyset$, it holds that $(C_j \wedge Ind(\varphi_i, C_j)) \equiv (C_j \wedge \varphi_i)$ by (2), fulfilling that $(C_j \wedge \varphi_i) \models \varphi_i$, propositional property by: $((p \wedge q) \supset q$, and reflexivity: $\varphi_i \models \varphi_i$. Thus, for any of the three possible outcomes of $Ind(\varphi_i, C_J)$, it is true: $(C_J \wedge Ind(\varphi_i, C_J)) \models \varphi_i$.

Corollary 1. $Fals(Ind(\varphi_i, K)) \subseteq Fals(\varphi_i)$.

Proof. $Fals(Ind(\varphi_i, C_j)) = Fals(\varphi_i) - Fals(C_j)$, to iterate over each C_j of K is satisfied that $Fals(Ind(\varphi_i, K)) = Fals(\varphi_i) - Fals(K)$. And properties between sets, it holds that $Fals(Ind(\varphi_i, K)) \subseteq Fals(\varphi_i)$.

Example 2. Let $K = (\neg p \vee q \vee r \vee s) \wedge (\neg p \vee \neg q) \wedge (\neg p \vee \neg r)$ y $\phi = (\neg p \vee \neg s \vee \neg t) \wedge (q \vee \neg r \vee \neg s \vee \neg t) \wedge (\neg p \vee q \vee r \vee \neg s \vee t) \wedge (\neg p)$, check if $K \models \phi$. It is equivalent to check whether$Fals(\phi)$ = {1**11, *0111, 10010, 1****} $\subseteq Fals(K) =$

Table 2. Application of $Ind(\phi, K)$ operator

ϕ \ K	1000*	11***	1*1**	$\boldsymbol{S}$
1**11	1**11	10*11	10011	**10011**
*0111	*0111	00111	00111	**00111**
10010	10010	10010	10010	**10010**
1****	11***	$\emptyset$	$\emptyset$	$\emptyset$
	101**	101**	$\emptyset$	$\emptyset$
	1001*	1001*	1001*	**1001***

$\{1000*, 11***, 1*1**\}$. In each cell of the table 2 is shown $Ind(\varphi_i, C_j)$. As shown in Table 2, $Ind(\varphi, K)$ is applied to a greater number of strings to those shown in Table 3, because in table 3, before to apply the independence operator, the clauses $C_i \in K$ are ordered according to the size $|lit(C_j) - lit(\varphi_i)|$. Because the number of literals C_j different with $varphi_i$ determine the number of independent clauses to be generated by $Ind(\Phi, K)$

Table 3. Calculation Ind with the $C_i \in K$ sorted

ϕ \ K	1*1**	11***	1000*	$\boldsymbol{S}$
1**11	1*011	10011	10011	**10011**
φ_2 \ K	1000*	11***	1*1**	$\boldsymbol{S}$
*0111	00111	00111	00111	**00111**
φ_3 \ K	11***	1*1**	1000*	$\boldsymbol{S}$
10010	10010	10010	10010	**10010**
φ_4 \ K	1*1**	11***	1000*	$\boldsymbol{S}$
1****	1*0**	100**	1001*	**1001***

Therefore, before applying operator $Ind(\varphi_i, K)$, it is appropriate to order the clauses of K according to the literals of each φ_i, as shown in Table 3. It gives us an strategy on the number of independent clauses to be generated. Also, the number of clauses to be generated is reduced via reduce clauses with complementary literals, it results in the case of example 2, $S = \{1001*, 00111\}$, whose CF is $Fs = (\neg p \vee q \vee r \vee \neg s) \wedge (p \vee q \vee \neg r \vee \neg s \vee \neg t)$. Thus, $K' = K \wedge Fs = (\neg p \vee q \vee r \vee s) \wedge (\neg p \vee \neg q) \wedge (\neg p \vee \neg r) \wedge (\neg p \vee q \vee r \vee \neg s) \wedge (p \vee q \vee \neg r \vee \neg s \vee \neg t)$.

6 The KM Postulates

We present here the analysis of the postulates on our proposed belief revision operator $K' = K \circ \Phi = K \wedge Ind(\Phi, K)$. Consider now the KM postulates.

- **(R1)** $K \circ \phi \models \phi$.
- **(R2)** If $K \wedge \phi$ is satisfiable then $K \circ \phi \equiv K \wedge \phi$.
- **(R3)** If ϕ is satisfiable, then so is $K \circ \phi$.
- **(R4)** If $K1 \equiv K2$ and $\phi 1 \equiv \phi 2$, then $K1 \circ \phi 1 \equiv K2 \circ \phi 2$.
- **(R5)** $(K \circ \phi) \wedge \gamma \models K \circ (\phi \wedge \gamma)$.
- **(R6)** If $(K \circ \phi) \wedge \gamma$ is satisfiable then also $K \circ (\phi \wedge \gamma) \models (K \circ \phi) \wedge \gamma$.

Theorem 2 shows that our belief revision operator meets the postulate **R1**. If $K \wedge \Phi$ is satisfiable and $K \models \Phi$, then each $\varphi_i \in \Phi$ is inferred from K and therefore $Ind(\varphi_i, K) = \varphi_i, i = 1, \ldots, k$. As, $K \circ \Phi = K \wedge Ind(\Phi, K) = K \wedge \Phi$, fulfilling the postulate **R2**. The **R3** postulate is fulfilled if $K \circ \phi$ is satisfiable (for **R2**). But if $Fals(K) \cup Fals(Ind(\phi, K)$ would cover the entire space assignments: 2^n, then only in this case, $K \circ \phi$ is redefined. $K \circ \phi$ is redefined to hold **R3**. And it is redefined $K \circ \Phi = ((K \wedge Ind(\phi, K)) - C_j)$, selecting the clause $C_j \in K$ with the least information (note that $|SAT(C_j)|$ is minimal on the cardinality of the set of models if $|C_j|$ is maximum in K).

R4 and **R5** postulates are satisfied because our review operator is closed on conjunctive forms. On the other hand, $Fals(K \circ (\phi \wedge \gamma)) = Fals(K \wedge S \wedge Ind(\gamma, K)) = Fals(K \wedge S) \cup Fals(Ind(\gamma, K)) = Fals(K \circ \phi) \cup Fals(Ind(\gamma, K)) \subseteq Fals(K \circ \phi) \cup Fals(\gamma)$, by Corollary 1. Thus, $Fals(K \circ (\phi \wedge \gamma)) \subseteq Fals(K \circ \phi) \cup Fals(\gamma) = Fals((K \circ \phi) \wedge \gamma)$ **R5** fulfilled. If $(K \circ \phi) \wedge \gamma$ is satisfied, then $K \circ (\phi \wedge \gamma) \models (K \circ \phi) \wedge \gamma$. As $Fals((K \circ \phi) \wedge \gamma) = Fals(K \wedge Ind(\phi, K) \wedge \gamma)$, but (γ) would equal $Ind(\gamma, K)$ iff γ is independent with each clause of K, and then only in that case, we have that Fals $(K \wedge Ind(\phi, K) \wedge \gamma) = Fals(K \wedge Ind(\phi, K) \wedge Ind(\gamma, K))$ $= Fals(K \circ (\phi \wedge \gamma)$ and then, the postulate **R6** is held.

7 Time-Complexity Analysis

The time function for our belief operator $K \circ \phi$ will be denoted as $T_0(|\phi|, |K|)$, and it depends mainly on the runtime operator of independence: $Ind(\phi, K)$. $Ind(\phi, K))$ is obtained from the iterative calculation of $Ind(\varphi_i, K)$.

The time complexity of $Ind(\varphi_i, K)$ process is of order $O(|K| \cdot n \cdot f(|\varphi_i|, |K|))$, where $f(|\varphi_i|, |K|)$ is an entire function , that given a clause φ_i and a CF K, determines the number of terms that will return the $Ind(\varphi_i, K)$ process.Let us now analyze the maximum possible number of clauses that can be generated through the process $Ind(\varphi_i, K)$. In some cases, $Ind(\varphi_i, K)$ can generate null sets (when $\exists\ C_j \in K$, such that $C_j \models \varphi_i)$), but in the worst cases, the time complexity of calculating $Ind(\varphi_i, K)$ depends on the length of the sets: $S_{ij} = \{x_1, x_2, \ldots, x_P\} = lit(C_j)$ - $lit(\varphi_i)$, $j = 1, \ldots, m$.

As noted in Example 2, a fixed $\varphi_i \in \phi$, it is appropriate to order the clauses $C_j \in K$ according to the cardinality of the S_{ij}, $j = 1, \ldots, m$ from lowest to highest, and eliminating the clauses that are independent with φ_i. When there is no

independent clauses with φ_i, neither $S_{ij} = \emptyset$ for $j = 1, \ldots, m$, the time complexity for calculating $Ind(\varphi_i, K)$) is bounded by the number of resulting clauses. In other words, it must $|Ind(\varphi_i, K)| \leq |S_{i1}| * |S_{i2}| * \ldots * |S_{im}| * Poly(n)$. Where $Poly(n)$ summarizes a polynomial time on the number of variables generated applying the operator $Ind(\varphi_i, Cj)$. Then, we can infer that the time complexity $T_0(|\phi|, |K|)$ for our belief revision operator, in the worst case, is upper bounded by $Max\{|S_{i1}| * |S_{i2}| * \ldots * |S_{im}| : \forall \varphi_i \in \phi\}$, eliminating polynomial factors on n (the number of variables) and the size of the KB K and the maximum value for this upper bound is $2^{(n-min\{|\varphi_i|:\varphi_i \in \phi\})}$. Thus, $T_0(|\phi|, |K|) \leq Max\{|S_{i1}| * |S_{i2}| * \ldots * |S_{im}| : \forall\ \varphi_i \in \phi\} \in O(2^{(n-min\{|\varphi_i|:\varphi_i \in \phi\})})$.

8 Conclusions

We present a new method for belief revision betweeen CF's. As K and ϕ are CF's, the review process between K and ϕ is reduced to make the review between each $\varphi_i \in \phi$ and each $C_j \in K$, simplifying the overall problem to do $|K| * |\phi|$ subproblems of review between two clauses. A logical operator, called $Ind(\varphi_i, C_j)$, was built. $Ind(\varphi_i, C_j)$ finds the clauses whose falsifying assignments is $Fals(\phi) - Fals(K)$. The correctness of our proposal is proved, and we show that it also holds the KM postulates. We also show the complexity-time of our proposal.

References

1. Alchourron, C., Gardenfords, P., Makinson, D.: On the logic of theory change: Partial meet contraction and revision functions. Journal of Symbolic Logic 50, 510–530 (1985)
2. Dalal, M.: Investigations into theory of knowledge base revision. In: Proc. The Seventh National Conference on Artificial Intelligence. pp. 475–479. AAAI (1988)
3. Darwiche, A., Pearl, J.: On the logic of iterated belief revision. Artificial Intelligence 89, 1–29 (1997)
4. Darwiche, A.: On tractable counting of theory models and its application to truth maintenance and belief revision. Applied Non-Classical Logics 11, 11–34 (2001)
5. De Ita, G., Zacarias., F.: Supervised and unsupervised lexical knowledge methods for word sense disambiguation. Computers and the Humanities 34, 103–108 (2000)
6. Doubois, O.: Counting the number of solutions for instances of satisfiability. Theoretical Computer Science 81, 49–64 (1991)
7. Ellis, D.: Irredundant families of subcubes. Mathematical Proceedings of the Cambridge Philosophical Society 150, 257–272 (2011)
8. Katsuno, H., Mendelzon, A.O.: On the difference between updating a knowledge base and revising it. KR'91 Cambridge, MA, USA 1, 387–394 (1991)
9. Khardon, R., Roth, D.: Reasoning with models. Artificial Intelligence 87, 187–213 (1996)
10. Liberatore, P., Schaerf, M.: Belief revision and update: Complexity of model checking. Journal of Computer and System Sciences 62, 43–72 (2001)
11. Liberatore, P., Schaerf, M.: The coplexity of model checking for belief revision and update. Journal of Computer and Systems Sciences 62, 43–72 (2001)

12. Nebel, B.: How hard is it to revise a belief base? Handbook of Defeasible Reasoning and Uncertainty Management Systems 3, 77–145 (1998)
13. Pilar Pozos Parra, W.L., Perrussel, L.: Dalal's revision without hamming distance. In: 12th Mexican International Conference on Artificial Intelligence. pp. 41–53 (2013)
14. Satoh, K.: Nonmonotonic reasoning by minimal belief revision. Institute for New Generation Computer Technology 358, 157–170 (1988)