Efficient Revision of Propositional Knowledge Bases

Guillermo De Ita[1], Carlos Guillén[2], A. López-López[2]

[1]Universidad Politécnica de Puebla , [2]INAOEP
deita@cs.buap.mx, cguillen@ccc.inaoep.mx, allopez@inaoep.mx

Abstract. We address the problem of update-revision of a knowledge base under the principle of minimal change. In order to perform the revision operation in polynomial time, we consider an initial knowledge base Σ expressed as conjunctions of unitary or binary clauses (set of formulas in 2-CF). Given a new Boolean formula F, we introduce the revision operator $*_I$, where the computation of $(\Sigma *_I F)$ relies heavily on a selected model I of F. We present a polynomial time algorithm to compute $(\Sigma *_I F)$ being F a 2-CF, but if F has not restrictions, the algorithm uses only one NP oracle call, resulting with an exponential time over the length of F and with polynomial time over the length of Σ, being common to consider the length of Σ much larger than the length of F. Then our proposal of revision operation is in the complexity class $P^{NP[1]}$.

Keywords. Intelligent Agents, Belief revision, Knowledge Representation, Satisfiability Problem.

1 Introduction

Artificial Intelligence has motivated the researchers to explore new reasoning issues and methods, and to combine disparate reasoning modalities into a uniform unified framework, so as to deal with incomplete, imprecise, contradictory, and changing information. Classical logic has been developed long time ago to study unchanging mathematical objects, being well-founded and consistent. However, it consequently acquired a static nature.

However, the agent paradigm demands to represent, besides of static knowledge, dynamic knowledge too. In this sense, logic theories have claimed a major role in defining the trends of modern research, increasing its influence in the development of algorithms to update-revise the Knowledge Bases of intelligent agents.

We consider in this work, the belief revision of a satisfiable Knowledge Base (KB) of an agent given new information and, if a modification of the KB is required, this has to be done losing the minimum of knowledge (i.e. according to the principle of minimal change).

Various approaches for incorporating dynamically a single or a sequence of changes into a KB have been proposed [2,4,8,10,13]. Almost all of these proposals are plagued by serious complexity-theoretic impediments, even in the Horn case [4,8].

Thus, an important problem to explore is the computational complexity of the revision approaches, and although all methods are clearly intractable in the general case [4,9,10], is relevant to find under which restrictions some methods would become tractable. In our case, we are interested in determining the class of formulas where the incremental recompilation of knowledge with minimal-change can be done efficiently.

To achieve a polynomial time when doing belief revision, we consider that Σ is in 2-CF (conjunctions of unitary and binary clauses) since for this class of formulas, the SAT problem (revision of satisfiability) is solved in polynomial time. However, for the same class of formulas (2-CF's), some related problems are NP-Hard. For example, given a formula F in 2-CF, the Max-2SAT problem (i.e. determining the maximum number of clauses of F that can be satisfied simultaneously), and the #2SAT problem (counting the number of models that F has) are both NP-Hard [11,12].

The main algorithm (Revise) presented here, is based on strategies for determining the satisfiability of a 2-CF Σ, starting by the construction of the transitive closure of Σ. This closure of Σ also allows to find a model when Σ is satisfiable. Revise is an update-revision scheme that finds a subset $\Sigma' \subset \Sigma$, where $(\Sigma' \cup F)$ is satisfiable, being F a propositional formula without restrictions. Such subset Σ' holds the Ginsberg's revision property [4], this means that does not exist γ, where $\Sigma' \subset \gamma \subseteq \Sigma$ and such that $(\gamma \cup F)$ will be satisfiable.

Once determined the subset $\Sigma' \subseteq \Sigma$ such that $(\Sigma' \cup F)$ is satisfiable, we present two different proposals for recovering the consistency of $(\Sigma \cup F)$. The first proposal keeps the structure of Σ in such way that the revision scheme is inductive. Also, we present a second proposal that, although it may extend the structure of the clauses of Σ, changing from k-clauses to $(k+1)$-clauses, with $1 \leq k$, the process for satisfiability revision continues requiring polynomial time.

2 Notation and Preliminaries

Let $X=\{x_1,\ldots,x_n\}$ be a set of n Boolean variables. A literal l is either a variable x or a negated variable $\sim x$. $\sim l$ denotes the negation of the literal l. We use $v(l)$ to indicate the variable involved by the literal l. For any variable x, the pair $\{x,\sim x\}$ is said to be complementary. We employ $\perp$ and T as constants of the language which represent the truth values *false* and *true*, respectively.

A *clause* is a disjunction of literals. The empty clause signals a contradiction. A clause is *tautological* if it has a complementary pair of literals. For $k \in \mathcal{N}$, a k-clause is a clause consisting of exactly k literals, and a *$(\leq k)$-clause* is a clause with k literals at the most. A unitary clause has just one literal and a binary clause has exactly two literals. A variable $x \in X$ *appears* in a clause c if either $\sim x$ or x is an element of c. Let $v(c)=\{x \in X: x$ appears in $c\}$. Let $P(c)$ and $N(c)$ be the sets of variables occurring positively and negatively in c, $P(c) \cap N(c) = \emptyset$. A clause c is *Horn*, if $|P(c)| \leq 1$.

A *phrase* is a conjunction of literals while a *Disjunctive Form* (DF) is a disjunction of phrases. A *Conjunctive Form* (CF) is a conjunction of clauses. A k-CF is a CF containing only k-clauses, and $(\leq k)$-CF denotes a CF containing $(\leq k)$-clauses. In the latter case, the CF is also called a non strict k-CF. An $s\mu$-CF is a CF in which no variable occurs more than s times. A $(k,s\mu)$-CF is a k-CF where each variable appears no more than s times, similarly a $(\leq k,s\mu)$-CF is a $(\leq k)$-CF where each variable appears s times at most. For any CF F, let $v(F)=\{x \in X : x$ appears in $F\}$, while $\text{Lit}(F) =\{l\colon l \in c \wedge c \in F\}$. A literal l is pure in a formula F if l appears in F and its negation $\sim l$ does not occur in F.

An *assignment s* for F is a function s: $v(F) \rightarrow \{0,1\}$. An assignment s can be also considered as a set of literals where there is no complementary pair of literals. If l is an element of an assignment s, then s sets l to *true* ($s(l)=1$) and $\sim l$ to *false* ($s(l)=0$). If we have $F_1 \subseteq F_2$ then $v(F_1) \subseteq v(F_2)$. Given two assignments: s_1, s_2 such that $s_1 \subseteq s_2$, s_1 being an assignment for F_1 and s_2 for F_2 respectively, then s_1 is a partial assignment of F_2 while s_2 is a total assignment for F_2.

A clause c is *satisfied* by an assignment s if and only if $(c \cap s) \neq \emptyset$. Otherwise we can say that c is *falsified* or *contradicted* by s. A CF F is *satisfied* by an assignment s if each clause of F is satisfied by s. F is *falsified* by s if any clause of F is contradicted by s. A model of F is an assignment M over $v(F)$ that satisfies F, and is denoted by $M \models F$, indicating that the assignment M is a model of F.

Let SAT(F) be the set of models that F has over its set of variables. If $\text{SAT}(F) \neq \emptyset$ then F is *satisfiable* otherwise F is *unsatisfiable*. Let $\#\text{SAT}(F)=|SAT(F)|$ be the cardinality of the set of models of F. Given F a CF, the SAT problem consists of determining if F has a model. The #SAT problem is the enumeration version of SAT, i.e. counting the number of models of F.

A Knowledge Base (KB) Σ is a set of formulae. Given a KB Σ and a propositional formula F, we say that Σ entails F, written $\Sigma \models F$, if F is true in every model of Σ. We say that Σ supports M, an assignment, if $M \in \text{SAT}(\Sigma)$, or equivalently, $M \models \Sigma$.

Let *LANG*-SAT and #*LANG*-SAT be the notation for the SAT and #SAT problem respectively, when propositional formulas in the class *LANG*-CF are involved, i.e. 2-SAT problem is the SAT problem for formulas in 2-CF, and #3-SAT denotes #SAT for formulas in 3-CF.

3 Graph Notation and the Transitive Closure

Let Σ be a 2-CF, we express as $G_\Sigma=(V,E)$ the *dependence-graph* originated by the restrictions of Σ. The set of nodes of G_Σ is $V=\text{Lit}(\Sigma) \cup \{\text{T},\perp\}$, T being a distinguished node associated with "True", and $\perp$ a distinguished node associated with "False". For each binary clause $c=(x,y) \in \Sigma$ there are two directed edges in E, given by: $\sim x \rightarrow y$ and $\sim y \rightarrow x$. For each unitary clause $c=(u) \in \Sigma$ there are also two directed edges in E: $\text{T} \rightarrow u$ and $\sim u \rightarrow \perp$. Those edges reflect the fact that each clause is equivalent to a pair of implications. We call these two directed edges originated by c, the *representative edges* of c.

Let "$\Rightarrow$" be the reflexive and transitive closure over any node of V. It follows immediately that for any two literals x and y, if $x \Rightarrow y$ then if a feasible solution M (a model of Σ) has x set to true, and then y must also be true. T(x) denotes the set of all such literals y forced by x, $\forall\, x \in \text{Lit}(\Sigma)$, $\text{T}(x) =\{ y \in \text{Lit}(\Sigma) : x \Rightarrow y \}$. As $x \Rightarrow y \equiv \sim y$

$\Rightarrow \sim x$, then if $y \in$ T(x) then $\sim x \in$ T($\sim y$). Note that T($\sim x$) can be seen as the set of unitary clauses resulting by applying unit resolution over Σ taking into account the unitary clause ($\sim x$).

Furthermore, note that if $\sim x \in$ T(x) then in any model of Σ, we cannot have x set true. Similarly, if $x \in$ T($\sim x$), then x cannot be set to false in any model of Σ. $\forall l$ Lit(Σ), we say that T(l) is *contradictory* if there is a variable $x \in v(\Sigma)$ such that x T(l) and $\sim x \in$ T(l) or well, the distinguished node $\perp \in$ T(l), otherwise T(l) is *sound*. Σ is a ($\leq$2)-CF and there exist unitary clauses, i.e. (u) $\in \Sigma$, then $\perp \in$ T($\sim u$) and T($\sim u$) is contradictory.

If there is a variable $x \in v(\Sigma)$ such that T(x) and T($\sim x$) are contradictory then Σ unsatisfiable. For a Σ satisfiable and $\forall l$, $l \in$ Lit(Σ), if T($\sim l$) is contradictory then Σ l.

Let Cont(Σ) = {$l \in$ Lit(Σ): T(l) is contradictory}, note that for any literal Cont(Σ), ($\Sigma \cup$ {(l)}) is unsatisfiable. Let Base(Σ) = {$l \in$ Lit(Σ) : T(l) is sound T($\sim l$) is contradictory}. The set Base(Σ) determines the set of variables which appear in any model of Σ with only one of the two possible logical values. In fact, the logical value with which such variables appear is the same as they appear in Base(Σ).

We have that Base(Σ) $\cap$ Cont(Σ) = $\emptyset$. Let Any(Σ) = Lit(Σ)-(Base(Σ) $\cup$ Cont(Σ)), then $\forall l \in$ Any(Σ), l is a literal which could appear with whatever of the two logical values in any model of Σ. Note that for any literal $l \in$ Cont(Σ) we have that $\sim l$ Base(Σ), so Base(Σ) $\cap$ Any(Σ) = $\emptyset$. Then the set {Cont(Σ),Base(Σ),Any(Σ)} specifies a partition of Lit(Σ). If SAT(Σ) $\neq \emptyset$ then for any model M of Σ we have Base(Σ) $\subseteq M$, and if Any(Σ) $\neq \emptyset$ then Any(Σ) $\cap M \neq \emptyset$. So, the following observation is pertinent:

Remark 1. If Σ is a satisfiable ($\leq$2)-CF then for any literal $l \in$ (Base(Σ) $\cup$ Any(Σ)), there is a model $M \in$ SAT(Σ) such that $l \in M$.

In terms of the graph G_Σ, given a node (literal) l, T(l) is the set of vertices of that can be reached from l. A *path* from l to any node v of G_Σ is a sequence of edges v_0v_1, $v_1v_2, \ldots, v_{k-1}v_k$ where $l = v_0$ and $v_k = v$. The length of the path is k. A *simple path* a path such that v_0, $v_1,\ldots,v_k$ are all different. If T(l) is contradictory, then there is least one path from l where two contradictory nodes: x and $\sim x$ appear in such path and, we say that such path is contradictory.

Given a formula Σ in the class ($\leq$2)-CF with m clauses and n variables, we refer *closure* to the procedure that computes the sets: T(x) and T($\sim x$), $\forall x \in v(\Sigma)$. *Closure* runs in polynomial time, in fact, the complexity is O($n \cdot m$) [7].

After computing the closure sets T(l), $\forall\, l \in$ Lit(Σ), we also obtain the sets of literals: Base(Σ), Cont(Σ) and Any(Σ). Moreover, when |SAT(Σ)| $\leq$ 1, from the Base(Σ) we can find the unique model of Σ or otherwise, determine that Σ is unsatisfiable. The second possibility emerges when $\exists\, l \in$ Lit(Σ) such that $l \in$ Cont(Σ) and $\in$ Cont(Σ). So, checking for the satisfiability of Σ in ($\leq$2)-CF is done in polynomial time. Furthermore, if Σ is satisfiable, the construction of a model M of Σ is also done in polynomial time by the same procedure *closure*[7].

Proposition 1. Let Σ be a satisfiable ($\leq$2)-CF and c a ($\leq$2)-clause, then the revision of satisfiability of $(\Sigma \wedge c)$ can be done in polynomial time.

As discussed above, the *closure* procedure checks for satisfiablity of any ($\leq$2)-CF in polynomial time, and as $(\Sigma \wedge c)$ is a ($\leq$2)-CF, then proposition 1 follows at once. In fact, if we know T(l), $\forall l \in$ Lit(Σ), then we can revise if any clause c is contradictory with Σ iff $\forall l \in c$, $l \in$ Cont(Σ). Furthermore, if F is a new CF, the revision for the satisfiability of $(\Sigma \wedge F)$ is done in linear time over the length of F, if we know Cont(Σ).

Example 1. Let $\Sigma = \{c_1=(x_1, \sim x_2), c_2=(x_1, x_2), c_3=(x_1, x_3), c_4=(\sim x_1, x_3), c_5=(\sim x_2, x_4), c_6=(\sim x_2, \sim x_4), c_7=(x_2, x_5), c_8=(x_3, \sim x_5)\}$, whose representative edges are:

c_1: $\sim x_1 \rightarrow \sim x_2$; $x_2 \rightarrow x_1$, c_2: $\sim x_1 \rightarrow x_2$; $\sim x_2 \rightarrow x_1$,

c_3: $\sim x_1 \rightarrow x_3$; $\sim x_3 \rightarrow x_1$, c_4: $x_1 \rightarrow x_3$; $\sim x_3 \rightarrow \sim x_1$,

c_5: $x_2 \rightarrow x_4$; $\sim x_4 \rightarrow \sim x_2$, c_6: $x_2 \rightarrow \sim x_4$; $x_4 \rightarrow \sim x_2$,

c_7: $\sim x_2 \rightarrow x_5$; $\sim x_5 \rightarrow x_2$, c_8: $\sim x_3 \rightarrow \sim x_5$; $x_5 \rightarrow x_3$.

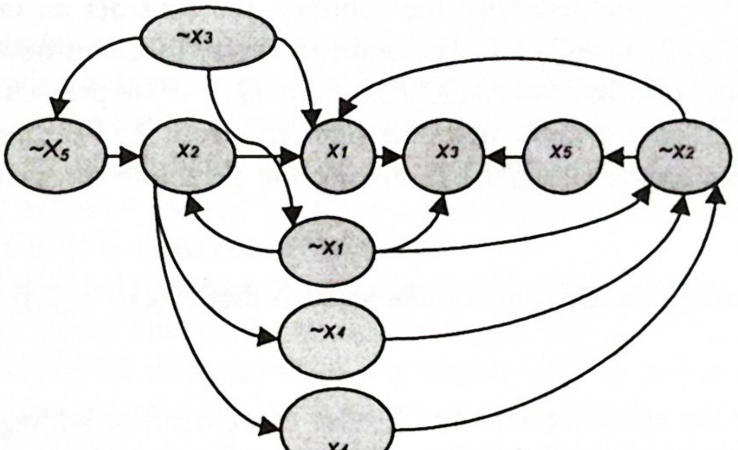

Fig. 1. The Dependence-graph for the formula Σ from example 1.

In Figure 1, we present the *dependence-graph* G_Σ for the formula Σ. The different sets T(l), $\forall l \in$ Lit (Σ) are also shown as well as the sets: Base(Σ), Cont(Σ), and Any(Σ).

$T(\sim x_1) = \{ \sim x_1, \sim x_2, x_2, x_3, x_1, x_4, \sim x_4, x_5 \}$, $T(x_1) = \{x_1, x_3\}$,

$T(\sim x_2) = \{ \sim x_2, x_1, x_3, x_5 \}$, $T(x_2) = \{x_2, x_1, x_3, x_4, \sim x_4, \sim x_2\}$,

$T(\sim x_3) = \{ \sim x_3, \sim x_1, x_3, x_1, \sim x_5, x_2 \}$, $T(x_3) = \{ x_3 \}$,

$T(\sim x_4) = \{ \sim x_4, \sim x_2, x_1, x_3, x_5 \}$, $T(x_4) = \{x_4, \sim x_2, x_1, x_3, x_5 \}$,

$T(\sim x_5) = \{ \sim x_5, x_2, x_1, x_3, x_4, \sim x_4, \sim x_2 \}$, $T(x_5) = \{ x_5, x_3 \}$.

We notice that T($\sim x_1$), T(x_2), T($\sim x_3$), T($\sim x_5$) are all contradictory. Then, Cont(Σ) = $\{\sim x_1, x_2, \sim x_3, \sim x_5\}$, Base($\Sigma$) = $\{x_1, \sim x_2, x_3, x_5\}$ and Any(Σ) = $\{x_4, \sim x_4\}$. And the two models of SAT(Σ) are: $\{x_1, \sim x_2, x_3, x_5, x_4\}$ and $\{x_1, \sim x_2, x_3, x_5, \sim x_4\}$.

Given a formula Σ, we call the reduction of Σ by a literal l, (also called forcing l) and denoted by $\Sigma[l]$ to the set of clauses generated from Σ by removing all clauses containing l and then removing $\sim l$ from all the remaining clauses. For example, if Σ the formula of the example 1, $\Sigma[\sim x_2] = \{(x_1), (x_1), (x_3), (\sim x_1, x_3), (x_5), (x_3, \sim x_5)\}$.

For an assignment $s = \{l_1,l_2,\ldots, l_k\}$ and a formula Σ, we define the reduction of by s, as: $\Sigma[s] = \Sigma\ [l_1][l_2]\ldots[l_k]$. For short, we write $\Sigma[l_1,l_2,\ldots,l_k]$ instead $\Sigma[\{l_1,l_2,\ldots,l_k\}]$. For example, considering again the formula in example 1, and performing the reduction given by the partial assignment $s=\{x_2,\sim x_3\}$, we obtain $\Sigma[x_2,\sim x_3]$ $= \{(x_1), (x_1), (\sim x_1), (x_4), (\sim x_4), (\sim x_5)\}$.

The *unit clause reduction* over a formula Σ consists of, given a unitary clause $c=(u)$, performing a reduction of Σ for the literal of the unitary clause, that is, $\Sigma[u]$. unit clause reduction can generate new unit clauses. *Unit Propagation* (*UP*) is the erative process of doing unit clause reduction over a set of clauses Σ until there is additional unitary clause in the reductions of Σ.

4 Incremental Recompilation Procedures

From now on and as long as we do not establish the contrary, we consider an initial satisfiable knowledge base (KB) Σ in ($\leq$2)-CF. We want to revise Σ given new formation expressed by a formula F, denoted as (Σ^*F), while $(\Sigma + F)$ expresses that F is simply added to Σ and $(\Sigma - F)$ denotes that F is removed from Σ. We present two different proposals for the operation (Σ^*F) following the principle of minimal change.

Proposition 2. Let Σ be a satisfiable CF and c a clause such that $(v(c) - v(\Sigma)) \neq$ then $(\Sigma \wedge c)$ is satisfiable.

$(\Sigma \wedge c)$ is satisfiable since for every model M of Σ, we can extend M adding literals that appear in c and do not appear in Lit(Σ) and such extended assignment is model for $(\Sigma \wedge c)$.

Given a new clause c (independent of its number of literals) such that $(v(c) - v(\Sigma))$ $\neq \emptyset$, then $(\Sigma^*c) = (\Sigma \wedge c)$, because $(\Sigma \cup \{ c \})$ is always satisfiable based on proposition 2. Then, if the KB Σ is satisfiable, the only way that $(\Sigma \cup \{ c \})$ can be unsatisfiable is by aggregating a clause c such that $v(c) \subseteq v(\Sigma)$ and where c contradicts some clause(s) of Σ.

Let consider that the new information for the revision process consists of a propositional formula F which is satisfiable as well as the KB Σ. We want to revise Σ that $(\Sigma * F)$ becomes satisfiable. We work based on the rejection principle, i.e., new information F for revising Σ is preferred over older information which could be overridden [5].

4.1 Formula-Based Change

If a KB Σ is inconsistent with a new formula F, a straightforward method to gain consistency with F is to remove formulas from Σ. Let $W(\Sigma,F)$ be the set of maximal subsets of Σ which are consistent with the revising formula F:
$W(\Sigma,F) = \{\Sigma' \subseteq \Sigma : \sim(\Sigma' \cup F \vdash \bot) \ \& \sim\exists\gamma: \Sigma' \subset \gamma \subseteq \Sigma \ \& \ \sim(\gamma \cup F \vdash \bot)\}$.

The set $W(\Sigma,F)$ contains all the plausible subsets of Σ that we may retain when inserting F [10].

One of the most common ways to achieve a revision operation is by a formula-based change applying the "Ginsberg's approach", where: $\Sigma *_G F = \{\Sigma' \cup F : \Sigma' \in W(\Sigma,F)\}$. That is, the result of revising Σ with F is the family of maximal subsets Σ' $\subseteq \Sigma$ that are consistent with F, plus F. Note that in $W(\Sigma,F)$ could exist more than one subset with the same number of clauses, so there could also exist more than one subset Σ' with the maximum number of clauses over $W(\Sigma,F)$.

4.2 Model-Based Change

Model-based revision operators refer only to the extensions, i.e. the model M of a Knowledge Base, regardless of the syntactical knowledge representation. The considered model-based change operators rely heavily on the notion of model closeness defined by $M\Delta M'$ being M a model of Σ and M' a model of F. The symmetric set difference $M\Delta M' = (M \cup M')-(M \cap M')$, then $v(M\Delta M')$ is the set of variables of $v(\Sigma)$ whose value in M is different to that in M'.

Dalal [14] proposes to interpret minimality of change in terms of minimal cardinality of model change. In his approach, $(\Sigma *_D F)$ has those models of F which have as few variables different from any model of Σ as possible. The *distance* between two formulas p and q is: $|\Delta|^{min}(p,q) = \min\{|v(M\Delta M')| : M \in SAT(p) \ \& \ M' \in SAT(q)\}$, that is $|\Delta|^{min}(p,q)$ is the minimum number of variables in which models of p and q diverge. And Dalal's approach is defined as: $SAT(\Sigma *_D F) = \{M \in SAT(F): \exists M' \in SAT(\Sigma) \ \& \ |v(M\Delta M')|=|\Delta|^{min}(\Sigma,F)\}$.

Note that under the previous definition, an element of $SAT(\Sigma *_D F)$ is not necessary a total assignment for $(\Sigma \cup F)$, specially if $v(\Sigma) - v(F) \neq \varnothing$.

In the work of Eiter[4], the complexity of the counterfactual revision problem is treated. Such problem consists of, given a KB Σ, a revision formula F, and a formula q, deciding whether q is derivable from $\Sigma * F$ considering different revision operations. For the two revision operations discussed here (Ginsberg's and Dalal's approaches) this counterfactual problem is at the second level of the polynomial hierarchy, thus harder than classical implication in propositional logic. Furthermore, in the Horn case, the complexity level is lowered by one, but the problem remains intractable for each approach. In exact way, for propositional formulas Σ, F and q without any restriction, and considering the counterfactual problem: $(\Sigma *_G F) \vdash q$ is a Π_2^P-complete problem. While, if both Σ as F are Horn formulas, the above counterfactual problem is co-NP-complete.

In the case of Dalal's approach, for formulas in general, the counterfactual problem is a $P^{NP[O(\log n)]}$-complete problem, while if both Σ as F are Horn formulas the counterfactual problem remains in the complexity class $P^{NP[O(\log n)]}$-complete.

In order to avoid these serious complexity-theoretic impediments, we consider this work, a variation of the Ginsberg's revision operator.

We have observed that the results of the revision operation between two satisfiable formulas ($\Sigma * F$) depend heavily on which model I of F is being considered, so we refer to our revision operator as: ($\Sigma *_I F$), and we show later that $\Sigma *_I F \subseteq \Sigma *_G F$.

Let $W(\Sigma,I) = \{\Sigma' \subseteq \Sigma : \sim(\Sigma' \cup I \vdash \bot) \;\&\; \sim\exists\gamma: \Sigma' \subset \gamma \subseteq \Sigma \;\&\; \sim(\gamma \cup I \vdash \bot)\}$ be the possible consistent subsets of Σ with an assignment I, then: $\Sigma *_I F = \{\Sigma' \cup F : I$ SAT(F) and $\Sigma' \in W(\Sigma,I)\}$.

Notice that in this case as I is a model of the formula F, this guarantees that each element of $W(\Sigma,I)$ satisfies F. The model I can be considered as a phrase of the Disjunctive Form of F. Then, depending on the model I of F that we consider, we obtain a specific set of clauses for $\Sigma *_I F$.

Let $CT_I(\Sigma) = \{S \subseteq \Sigma: \exists\Sigma' \in W(\Sigma,I) \;\&\; S = (\Sigma - \Sigma')\}$. Each element in $CT_I(\Sigma)$ contains a subset of clauses from Σ which are conflicting with F (in fact, with one of models: I). We refer to the set $CT_I(\Sigma)$ as the *conflicting clauses* of Σ with I and we show in the following procedure how to compute this set.

4.3 A Procedure for Computing ($\Sigma *_I F$)

Input: A satisfiable KB Σ in (≤ 2)-CF and a satisfiable formula F such that $(\Sigma \cup F)$ is unsatisfiable.

Output: $\Sigma' \subseteq \Sigma$ where $\Sigma' \in W(\Sigma, I)$ being I a model of F.

Procedure Revise:

1: Call *closure*(Σ), obtaining the sets: Cont(Σ), Base(Σ), Any(Σ) as well as a model M of Σ.

2: Let I be a model of F (we will explain later on how to obtain that model). The model I is a conjunction of literals, so each literal $l \in I$ must be considered as unitary clause in $(\Sigma \cup I)$, i.e., $(\Sigma \cup I) = (\Sigma \wedge_{l \in I} (l))$.

3: As $(\Sigma \cup F)$ is unsatisfiable, $(\Sigma \cup I)$ is unsatisfiable too, but not all the literals l are in conflict with Σ. For any literal l of I if $l \notin$ Cont(Σ) then $I = I - \{l\}$. Notice that the resulting assignment I could not longer be a model of F.

4: Let $A = \Sigma[I]$ and $B = \Sigma \cap \Sigma[I]$. A contains the clauses of Σ conflicting with and B the clauses of Σ satisfied by I. Notice that $(v(A) \cap v(I)) = \emptyset$.

5: Each clause from Σ that changed to a nil clause in $\Sigma[I]$ is part of $CT_I(\Sigma)$. We remove all nil clauses from A.

6: We split the set of clauses of A in A1 and A2, where $A1 = \{c \in A: c \in \Sigma\}$ and $A2 = A - A1$. As $A1 \subseteq \Sigma$ and we consider that Σ is satisfiable, then A1 is satisfiable too.

7: Call *closure*(A1), obtaining the sets: Base(A1), Cont(A1), Any(A1) and model M of A1.

8: Apply the pure literal rule: For each literal l in Lit(A), if l is a pure literal in A, then A1=A1[l] and A2=A2[l].
9: Apply the non conflicting literals rule: For each literal l in Base(A1), if $\sim l \notin$ Lit(A2) then A1=A1[l] and A2=A2[l].
We iterate these two rules as many times as can be done.
The resulting clauses of A1 are in conflict with the clauses of A2.
10: As A2[M] contains just the conflicting clauses of Σ with ($I \cup M$), then for any nil clause in A2[M], we retrieve its original clause c in Σ and add it to $CT_I(\Sigma)$.
11: Let $\Sigma' = (\Sigma - CT_I(\Sigma))$.
12: Apply a last revision process: For each clause c in $CT_I(\Sigma)$, we check if ($\Sigma' \cup \{c\}$) is satisfiable and then $\Sigma' = \Sigma' \cup \{c\}$.

This last step guarantees that $\Sigma' \in W(\Sigma, I)$ and based on proposition 1, this step executes in polynomial time since $\Sigma' \cup \{c\}$ is a ($\leq$2)-CF.

Notice that each one of the steps in the above procedure runs in polynomial time given that the reductions used, the closure procedure, and the revision of satisfability are all polynomial time procedures.

The procedure *Revise* returns only one of the plausible subsets Σ' of Σ that we could retain when inserting F. In this sense, *Revise* works as a satisfiability algorithm to find an assignment M which satisfies ($\Sigma' \cup F$) and where M is an extended assignment of I (a phrase of the DF of F).

4.4 Recovering the Satisfiability of ($\Sigma \cup$ F)

Once that we had applied the procedure Revise(Σ,F), we obtain the set $CT_I(\Sigma) = \Sigma$-Σ'. We have two proposals in order to recover the satisfiability of ($\Sigma \cup F$).

One of these proposals, and also very straightforward is to eliminate from Σ the clauses of $CT_I(\Sigma)$.

In this case, this approach of adding a new formula and retracting some others from Σ, follows the *formula-based* change principle, and then, as pointed out by Eiter [4], heavily depends on the syntax of Σ. Thus, it is not expected that this approach respects the irrelevance of syntax postulate.

The second proposal consists of weakening the information coded by each clause c, $c \in CT_I(\Sigma)$, i.e. extending the clause c with a new literal outside of the language. Then, each clause $c \in CT_I(\Sigma)$ changes to ((c) $\vee$ l_i) being l_i, $i \geq 1$, a new literal which does not appear before in Lit(Σ). The aggregation of this dummy literal allows to satisfy the new clause ((c) $\vee$ l_i) as well as the formula F simultaneously, since we must assign to the new literal l_i the appropriate logical value to have ((c) $\vee$ l_i) with a true value, as shown by proposition 2. In this way, the new clause ((c) $\vee$ l_i) will not be in contradiction with F.

Each dummy literal l_i is not further used in this process of extending clauses.

Although, the size of some clauses from Σ were extended, its representative edges do not change in the dependence graph G_Σ, since we are taking l_i as a virtual literal used only to satisfy ((c) $\vee$ l_i). In fact, for every model M of Σ, $l_i \in M$.

This latter proposal is commonly used in updating logic programming, where the weakening procedure is applied over logic programs which are contradictory with new logical program which is being aggregated. This proposal was inspired by the previous works in [1,3,5].

Both proposals can be used depending on the application, considering which one more representative regarding to the minimal change. This will depend on the working scenario given that, as it is now widely believed, there is no general purpose, domain independent means of updating knowledge bases that will "do the right thing" under all circumstances [13].

Notice that in both proposals, the recovering of the satisfiability of $(\Sigma \cup F)$ polynomial over the length of $(\Sigma \cup F)$.

To design a procedure for computing a model of a formula F is a classic NP-Complete problem (SAT problem) and, as it is well known, has an exponential time complexity, but not for Horn and 2-CF formulas which are solved in polynomial time.

For formulas F in general (not Horn, not (≤ 2)-CF's), let consider that we make call to an NP oracle O with argument F and this oracle returns a model I of F. Thus, our revision operator $*_I$ has an exponential time complexity over the length of F but continues being polynomial over the length of Σ, which is considered in general, much larger than F. In our proposals, the recovering of the satisfiability of $(\Sigma \cup$ continues with a polynomial complexity.

5 Conclusions

We introduce a new revision operator $*_I$ where, given a KB Σ in (≤ 2)-CF and a satisfiable formula F, the computation of $(\Sigma *_I F)$ relies heavily on the selection of model I of F. Once determined a model of F, we present a procedure to compute subset $\Sigma' \subseteq \Sigma$ such that Σ' follows the Ginsberg's approach, this is, $(\Sigma' \cup F)$ is satisfiable and there is not γ such that $\Sigma' \subset \gamma \subseteq \Sigma$ and $(\gamma \cup F)$ will be satisfiable.

The results presented here, show that the *revision operator* $*_I$ in the face of single 2-CF formulas can be carried out efficiently. Moreover, as our proposals are inductive, we extend these results for considering an efficient incremental recompilation over 2-CF's.

Once determined a maximum subset $\Sigma' \subseteq \Sigma$ such that $(\Sigma' \cup F)$ is satisfiable, we also present two different proposals to recover the satisfiability of $(\Sigma \cup F)$ working polynomial time.

Given Σ and F two 2-CF's, the algorithm *Revise* computes $(\Sigma *_I F)$ in polynomial time, but if F has not restrictions then, the computation of $(\Sigma *_I F)$ is solved by our polynomial time algorithm with only one NP oracle call, showing that our procedure is in the complexity class $P^{NP[1]}$.

References

[1] Alferes J.J., Leite J.A., Pereira L.M., Przymusinska H., Przymusinski T., "Dynamic updates of non-monotonic knowledge bases", *Jour. Of Logic Programming* 45, 2000, pp. 43-70.
[2] Darwiche A., "On the tractable counting of theory models and its application to truth maintenance and belief revision", *Jour. Of Applied Non-classical Logics* 11, 2001, pp.11-34.
[3] Alferes J.J., Pereira L.M., "Logic Programming updating – a guided approach", F. Sadri, A. Kakas, editors, Essays in honor of Robert Kowalsky, *LNAI*, Vol. 2, 2408, 2000.
[4] Eiter T., Gottlob G., "On the Complexity of Propositional Knowledge Base Revision, Updates, and Counterfactuals", *Artificial Intelligence* 57, 1992, pp. 227-270.
[5] Eiter T., M. Fink, G. Sabattini, H. Thompits, "Considerations on updates of logic programs", *Lecture Notes in Artificial Intelligence*, Vol. 1, (JELIA'00), 2000, pp.212-226.
[6] Eiter T., Makino K., "Generating all abductive explanations for queries on propositional Horn Theories", *INFSYS Research Report* 1843, (2003), pp.03-09.
[7] Gusfield D., Pitt L., "A Bounded Approximation for the Minimum Cost 2-Sat Problem", *Algorithmica* 8, (1992), pp. 103-117.
[8] Gogic G., Papdimitriou C., Sideri M., "Incremental Recompilation of Knowledge", Jour. Of Artificial Intelligence Research 8, 1998, pp. 23-37.
[9] Katsuno H., Mendelzon A., "On the Difference Between Updating a Knowledge Base and Revising it", In Proceedings KR-91, pp. 387-395, 1991.
[10] Liberatore P., Schaerf M., "The Complexity of Model Checking for Belief Revision and Update", Procs. Thirteenth Nat. Conf. on Artificial Intelligence (AAAI'96).
[11] Roth D., On the hardness of approximate reasoning, *Artificial Intelligence* 1996; 82:pp.273-302.
[12] Russ B., *Randomized Algorithms: Approximation, Generation, and Counting*, Distinguished dissertations, Springer, 2001.
[13] Winslett M., *Updating Logical Databases*, Cambridge University Press, 1990.
[14] Dalal M. "Investigations Into a Theory of Knowledge Base Revision: Preliminary Report", Procs. Seventh Nat. Conf. on Artif. Intelligence (AAAI-88), St. Paul, Minnesota, August 1988, pp. 475-479.