Handling Incoming Beliefs

Philippe Besnard, Eric Grégoire

To link to this article : doi: 10.1007/978-3-642-39787-5_17
URL : http://dx.doi.org/10.1007/978-3-642-39787-5_17

Handling Incoming Beliefs

Philippe Besnard[1] and Éric Grégoire[2,⋆]

[1] IRIT UMR CNRS 5505
118 route de Narbonne F-31062 Toulouse France
besnard@irit.fr

[2] CRIL
Université d'Artois - CNRS UMR 8188
rue Jean Souvraz SP18 F-62307 Lens France
gregoire@cril.univ-artois.fr

Abstract. Most logic-based approaches to knowledge and belief change in artificial intelligence assume that when a new piece of information comes up, it should be merely added to the current beliefs or knowledge when this does not lead to inconsistency. This paper addresses situations where this assumption does not hold. The focus is on the construction of Boolean standard-logic knowledge and belief bases in this context. We propose an approach to handle incoming beliefs that can require some formulas reconstruction or a form of preemption to be performed.

Keywords: Knowledge Engineering, Knowledge Representation, Logic-based Artificial Intelligence.

1 Introduction

These last two decades, a fertile domain of research in knowledge representation and reasoning has concerned the way knowledge and beliefs should change in light of new information. Especially, so-called (logic-based) belief revision, belief update [1,2,3,4,5] and knowledge fusion [6,7] have become Artificial Intelligence research fields in their own rights [1,8,9,10,11,12]. Their main focus is on situations where a new piece of information is logically contradicting the pre-existing knowledge or beliefs. When no logical contradiction arises, these approaches assume that this new piece of information should be adopted as such, without any specific treatment. However, this latter assumption does not hold in frequent situations. Accordingly, in this paper we adapt the construction of Boolean standard-logic belief and knowledge bases[1], allowing them to encompass situations of that kind in an adequate way. Let us give some motivating examples.

⋆ This work has been supported in part by the *Région Nord/Pas-de-Calais* and the EC through a FEDER grant.

[1] From now on, we do not distinguish between the words belief and knowledge.

A first example is about the necessity to *merge pieces of knowledge themselves*. Assume that a standard-logic knowledge base Δ is under development and already contains R_1 = *If the switch is on and the lamp bulb is ok then the light is on*. Later on, we are given another rule or belief R_2 = *If the switch is on and the switch is not broken then the light is on*. R_2 does not contradict R_1 from a logical point of view. If R_2 is merely inserted within Δ then we can deduce *Light is on* from Δ when the switch is broken, according to R_1 and provided that both *The switch is on* and *The lamp bulb is ok* are established. This is clearly counter-intuitive. Actually, to match our intuitions, both rules should be merged to form R_3 = *If the switch is on and the lamp bulb is ok and the switch is not broken then the light is on*, which must replace both rules R_1 and R_2.

If we had represented the rules in a converse manner like *If the light is on then the switch is on and the lamp bulb is ok*, the above problem would have been avoided; however, we would have lost the intended capacity to infer that the light is on based on the observations that the switch is on and the lamp bulb is ok. Hence, the selected way to represent rules. Actually, the problem addressed here is not founded on a specific way to represent rules; the necessity to merge rules can occur whenever rules do not a priori capture all their possible exceptions. Although this basic example is straightforward, its treatment in the general case is not direct. First, not all rules should be merged. In the example, R_1 and R_2 must be merged because each of them captures a kind of *compulsory* condition for a shared conclusion to be derived. We should thus be able to distinguish between compulsory and non-compulsory conditions for a rule to apply. Moreover, rules are not necessarily explicit in Δ but can be mere implicit deductive consequences of Δ, only. Also, it must be investigated to which extent this issue might also concern formulas that do not encode rules but simply share some variables. Especially, when a syntax-independent logic-based representation of knowledge is adopted, there is no way to distinguish between a rule and e.g. its representation as a clause, i.e. as a disjunction of signed variables. Finally, what must be dropped from Δ after the merged formulas is inserted must also be defined carefully. We address these questions in this paper.

The issue of *preempting subsuming knowledge* is a slightly related problem. Assume now that Δ contains only the rule R_0 = *If the switch is on then the light is on*. From a logical point of view, R_1 is a mere logical deductive consequence of R_0 and is thus subsumed by R_0. As such, from a logical point of view, R_1 does not bring any actual additional information to R_0. Indeed, in the presence of both rules R_0 and R_1, whenever the switch is on, the light is on (not depending on whether the lamp bulb is ok or not). In order to enforce R_1 over R_0 and thus require *The lamp bulb is ok* for the light to be on, we need the ability to make some subsumed information (namely, R_1) prevail over (or say, preempt) the subsuming knowledge (R_0). Remark that this issue of *preempting subsuming knowledge* also occurs as a sub-part of the *merging pieces of knowledge* one: when the merged formula has been inserted within Δ, things must be settled so that the latter formula is not subsumed by any of the initial ones, which by construction subsume it.

Let us stress that in the general case situations where subsumption links must be annihilated are neither specific to rules, nor due to the selected form of implication connective and its use, nor due to causality issues. On the contrary they can concern any logical formula. To illustrate that, assume that a new piece of information $office \vee home \vee bar$ comes up and must prevail over $office \vee home$ that is already present [13]. From a logical point of view the new piece of information is a deductive consequence of the former one and there is no inconsistency involved. However, we need the subsumed information to replace or prevail over the subsuming one.

In this context, our contribution in this paper builds on and extends some previous works. First, an approach to characterize and solve the subsumption issue has been introduced in [14]. Candidate rationality postulates for belief change operators that allow beliefs to be preempted by subsumed ones have been presented in [15]. The formal characterization of this specific handling of subsumption has been extended to a general non-monotonic setting in [16] and applied to the legal domain in [13]. Starting from this, the contribution of this paper is at least twofold. First, we present a new formal solution to the subsumption issue that avoids rules to be preempted when their preconditions are not satisfied and that also accommodates the problem of merging pieces of knowledge themselves (none of these issues was encompassed in any of the aforementioned pieces of work). Then, the focus is on knowledge engineering issues. More precisely, we study the insertion of knowledge from a pragmatic point of view, taking the above issues into account, when a distinction is made between explicit and implicit information in a knowledge base.

The logical setting in this paper is standard (clausal) Boolean logic. On the one hand, it is the simplest possible framework for presenting and addressing the above subsumption-related issues. On the other hand, recent dramatic progress in Boolean search and reasoning has now revived Boolean logic as a realistic and attractive framework for representing large knowledge bases and solving numerous complex reasoning tasks in artificial intelligence [17].

The paper is organized as follows. In the next section, basic notions about standard Boolean logic are recalled. The question of merging pieces of knowledge and the subsumption issue are presented and solved in section 3. In section 4, a concept of compulsory clauses is introduced. Section 5 focuses on an adapted prime implicate representation. A concept of restrictive clauses is presented in section 6. Main issues about the interaction of a new belief with a preexisting base are addressed in section 7, together with some computational issues. The paper ends with perspectives and promising paths for further research.

2 Logic-Based Framework

To concentrate on the aforementioned conceptual problems, we consider the simple framework of standard (clausal) Boolean logic. Let $\mathcal{L}$ be a language of formulas over a finite alphabet $\mathcal{P}$ of Boolean variables, also called *atoms*. Atoms are noted $a, b, c, \ldots$ The $\wedge, \vee, \neg, \Rightarrow$ and $\Leftrightarrow$ symbols represent the standard conjunctive, disjunctive, negation, material implication and equivalence connectives, respectively. A *literal* is an atom or a negated atom. Formulas are built in the usual way from atoms, connectives and parentheses; they are noted f, g, h, etc. A formula is in conjunctive normal form (CNF) when expressed as a conjunction of *clauses*, where a clause is a disjunction of literals. For convenience, clauses can be represented by their set of involved literals. The empty clause represents *false*. Also for convenience, the set of involved literals of a clause can be enriched by the value *false*, while still representing the clause. Also for convenience, the disjunction forming a clause can be safely enriched by a disjunct representing *false*.

Interpretations are functions assigning either *true* or *false* to every atom. A *model* of a set of formulas Δ is an interpretation that satisfies every formula of Δ. Δ is *consistent* (also said *satisfiable*) when its set of models is not empty. $\Delta \vdash f$ expresses that the formula f can be deduced from Δ, i.e., that it is *true* in all models of Δ.

A *knowledge base* Δ is a consistent finite set of (non-tautological) clauses and the incoming belief f is a consistent non-tautological clause. We distinguish between Δ, which represents the *explicit* clauses of the base, from the set of all the deductive conclusions of Δ, noted $Th(\Delta)$: $\Delta \vdash f$ iff $f \in Th(\Delta)$.

A word of caution can be needed for readers who are familiar with rule-based systems but not with logic. We exploit the sound and complete deductive capabilities of Boolean logic. Especially, we do not only simply allow for mere forward and backward chaining on $\Rightarrow$ as in traditional rule-based systems. For example, from the rule $a \Rightarrow b$ and $\neg b$, we derive $\neg a$ using contraposition. Also, keep in mind that a rule of the form $(a \wedge b \wedge \neg c) \Rightarrow (d \vee e)$ is logically equivalent to $\neg a \vee \neg b \vee c \vee d \vee e$ (which is also represented by $\{\neg a, \neg b, c, d, e\}$) and will be treated as such.

3 Preempting Subsuming Knowledge and Merging Clauses

In the following, two central concepts are *strict implicant* and *subsumption*.

Definition 1. *Let f and g be two formulas. f is a strict implicant of g iff $f \vdash g$ but $g \nvdash f$. Δ strictly subsumes (in short, subsumes) g iff $\Delta \vdash f$ for some strict implicant f of g.*

Interestingly, when f and g are clauses under their set-theoretic representation, f is a (strict) implicant of g when f is a (strict) non-empty subset of g. Moreover, when f is an implicant made of $n-1$ literals from the n different variables involved in g (i.e., when f is one longest sub-clause of g), f is said to be a *prime implicant* of g. For example, $\Delta = \{office \vee home\}$ subsumes $office \vee home \vee bar$ and $office \vee home$ is a prime implicant of $office \vee home \vee bar$.

3.1. Preempting Subsuming Knowledge

All strict implicants g of f must be expelled after f has been introduced inside Δ when f must prevail. Interestingly, as formulas of Δ are under CNF format, it is sufficient to expel the prime implicants of f. For example, when we ensure that the prime implicant *office*$\vee$*home* of *office*$\vee$*home*$\vee$*bar* is expelled, we are guaranteed that the smaller implicants *office* and *home* are expelled, too. Indeed, when any of these implicants remains, *office*$\vee$*home* remains derivable, too.

Actually, the problem of making a formula prevail over all its strict implicants in Δ must sometimes be adapted by replacing Δ by one of its subsets, say Δ', in Definition 1. Typically, Δ' is selected as containing the permanent information involving generic rules or other permanent knowledge, whereas the rest of Δ contains facts that are temporary or related to a specific case or result from the instantiation of the generic rules to a specific situation. When using such Δ' only, we reason about generic rules independently of specific facts and the process of transforming and expelling formulas considers the generic rules, only. Note that facts subsume any rule that contains them as part of its conclusion. For example, *Light is on* subsumes *If the switch is on and the lamp bulb is ok then light is on*. By considering rules only, we do not consider facts that are related to a specific situation and avoid expelling these facts although they subsume rules. In the rest of the paper, for notational convenience, we assume that any formula from Δ can be expelled in the process of accommodating an incoming belief.

3.2. About Merging Pieces of Knowledge themselves

Consider now the first example from the introduction, which requires some rules to be merged. Assume Δ contains the rules R_1 and R_2 in clausal form; namely, $\neg$*switch-on*$\vee\neg$*bulb-ok*$\vee$*light-on* and $\neg$*switch-on*$\vee\neg$*switch-ok*$\vee$*light-on*. R_1 and R_2 need be replaced by $R_3 = \neg$*switch-on*$\vee\neg$*bulb-ok*$\vee\neg$*switch-ok*$\vee$*light-on*. In this example, the operation is straightforward but more complex situations are also to handled.

First, assume R_1 is the clausal representation of *If A and condition-1 and condition-2 then B* whereas R_2 is intended to represent *If A and condition-3 then B*. Thus, R_3 = *If A and condition-1 and condition-2 and condition-3 then B* is inserted in clausal form within Δ. Clearly, retracting R_2 and R_1 might not been enough to make this new rule prevail. Indeed, Δ might also entail for example *If A and condition-1 and condition-3 then B*, which would subsume R_3. Thus, after having inserted the newly formed clause R_3 in Δ, we must make it prevail over all its strict implicants using the aforementioned process of preempting subsuming knowledge.

Then, not every clause should be merged. Intuitively, we need to merge two clauses when they are about a same subject and when at least one of them translates a form of compulsory condition about this subject. In the last example, we merged R_1 and R_2 because they (both) express compulsory conditions for having B on the basis of A.

In the next sections, concepts of compulsory and restrictive clauses are proposed. They will help us distinguish, capture and solve the various situations about merging pieces of knowledge themselves and about making knowledge prevail over the subsuming one.

4 Compulsory Clauses about f'

In the following, Δ is a consistent finite set of non-tautological clauses that has accommodated an incoming non-contradictory and non-tautological clause f. Unless explicitly indicated, when a clause is referred to, it is not tautological.

Definition 2. *When f can be rewritten as $f = g \vee f'$ where g and f' are two clauses with an empty intersection and where g is not the empty clause. f is said to be about f'. g is called the anti-condition for f' in f.*

When the anti-condition g for f' in f is *false*, f entails f': hence the "anti" prefix. However, by convenience, we will write "condition" instead of "anti-conclusion". Please, note that the definition entails that f is not about itself: f is about *false* and about any of its strict subsets. In our framework, the incoming clause f can be asserted together with the additional information that f is *specifically intended* to be about one specific given f', or without any information about such a possible intent. In the first, case this does not prevent the above definition from concluding that f is also about other f' when f is not just a literal. As motivated earlier, $f = g \vee f'$ might have been asserted together with some clause-labeling information expressing that f need be compulsory about f' in the final Δ.

A first family of such situations occurs when f is not allowed to be subsumed, i.e. when no strict sub-clause of f is allowed to belong to $Th(\Delta)$. In the motivating example, *office*$\vee$*home*$\vee$*bar* was such a subsumption-free clause in Δ. To express this, our proposed convention is to select $f' =$ *false* (and thus $g = f$) and provide some labeling indication that f must be compulsory about f' in Δ.

In the example encoding rules, we might express that $f = g \vee f' = \neg$*switch-on*$\vee\neg$*switch-ok*$\vee$*light-on*, which encodes the rule *If the switch is on and the switch is ok then the light is on*, is about $f' =$ *light-on*. We might additionally express that f translates compulsory conditions about f' and that the following requirements must be met.

Requirements (for $f = g \vee f'$ about f' to be compulsory for f' in Δ)

1. $f \in \Delta$. We require any compulsory clause to be explicit, i.e. to belong to Δ.
2. $f' \notin Th(\Delta)$. Otherwise, f could be interpreted as being a mere deductive consequence of f'. Remember that we have assumed that Δ is intended to contain the generic rules and clauses; more generally, we have assumed that any formula in Δ can be expelled to make f must prevail over the knowledge subsuming f. Moreover, when $f' =$ *false*, $f' \in Th(\Delta)$ would contradict the prerequisite that Δ is consistent.
   From these two first requirements, we conclude that $\neg g \notin Th(\Delta)$.
3. When $f' \neq$ *false*, we have $\neg f' \notin Th(\Delta)$. Otherwise, f' could never be derived because the prerequisite that Δ is consistent must be satisfied. If f' were always blocked from derivation then there could not exist at the same time any additional condition g that would authorize such a derivation.

4. When $f' = \textit{false}$, $\nexists h = g' \vee f'$ about f' in $Th(\Delta)$ such that h is a strict subset of f, like *office*$\vee$*bar* in the motivating example.
5. For any $h = g' \vee f'$ about f' where $h \in Th(\Delta)$, we have $g \subseteq g'$ unless $f' = \textit{false}$ or $g' \in Th(\Delta)$. This requirement does not exclude from $Th(\Delta)$ all clauses containing a strict subpart of g. For example, if we are able to derive that *in-Dalian-KSEM* is true, a standard-logic reasoner is justified to also conclude e.g. $h =$ *in-Dalian-KSEM* $\vee$ *light-on*, or $h =$ *in-Dalian-KSEM* $\vee\neg$*switch-on* $\vee$ *light-on*. These last inferences can be made independently of the actual truth value of *light-on* and despite any possible compulsory clause about *light-on*.
6. A compulsory clause about f' is *also* implicitly expressing conditions about any f'' where $\emptyset \subset f'' \subset f'$. More precisely, when a clause $f = g \vee f'$ is compulsory about f', $h = g \vee f''$ is compulsory about f'' for any f'' such that $\emptyset \subset f'' \subset f'$, *provided that* h belongs to $Th(\Delta)$. For example, when $\neg$*vacation* $\vee$ *go-to-beach* $\vee$ *go-to-mountain* is compulsory about *go-to-beach* $\vee$ *go-to-montain*, we have that $h = \neg$*vacation* $\vee$ *go-to-beach* is compulsory for *go-to-beach*, provided that $h \in Th(\Delta)$.

Hence, the formal definition.

Definition 3. *When $f = g \vee f'$ is about f'. f is compulsory about f' in Δ iff*

1. *$f \in \Delta$, and*
2. *$f' \notin Th(\Delta)$ and, when $f' \neq$ false, $\neg f' \notin Th(\Delta)$, and*
3. *when $f' =$ false, no strict implicant of f belongs to $Th(\Delta)$, and*
4. *$\forall\ (g' \vee f') \in Th(\Delta)$, $g \subseteq g'$ unless $g' \in Th(\Delta)$ or $f' =$ false.*

Example. Let $f = a \vee b \vee c1 \vee c2$ be compulsory about $c1 \vee c2$ in Δ. According to 3.1, $f \in \Delta$. According to 3.2, $c1 \vee c2 \notin Th(\Delta)$ and thus $c1 \notin Th(\Delta)$ and $c2 \notin Th(\Delta)$. Also, $\neg c1 \wedge \neg c2 \notin Th(\Delta)$. As a consequence of both 3.1 and 3.2, $\neg g = \neg a \wedge \neg b \notin Th(\Delta)$. According to 3.4, $f = a \vee d \vee c1 \vee c2 \in Th(\Delta)$ requires that $d \in Th(\Delta)$.

Property 1. When $f = g \vee f'$ is compulsory about f' in Δ, $g \vee f''$ is compulsory about f'' for any f'' s.t. $\emptyset \subset f'' \subset f'$.

Indeed: first note that $f'' \neq \textit{false}$. Then, assume $\exists\ g' \vee f'' \in Th(\Delta)$: $g \not\subseteq g'$ unless $g' \in Th(\Delta)$. Since $g' \vee f'' \vdash g' \vee f'$, all this would contradict $\forall\ (g' \vee f') \in Th(\Delta)$, we have $(g \subseteq g')$ unless $g' \in Th(\Delta)$ or $f' = \textit{false}$.

Clearly, this property can be of practical computational importance since none of the conditions in Definition 3 needs to be checked to decide whether $g \vee f''$ is compulsory about f'' when we know that $g \vee f''$ is compulsory about f'' and that $g \vee f'' \in Th(\Delta)$.

Example (Cont'd). Assume also that $a \vee b \vee c1 \in Th(\Delta)$. Then, $a \vee b \vee c1 \in \Delta$ and this clause is compulsory about $c1$.

Property 2. When a clause $f = g \vee f'$ is compulsory about f' in Δ, we have that

1. all the aforementioned *Requirements (for* $f = g \vee f'$ *about* f' *to be compulsory for* f' *in* Δ*)* are satisfied,
2. when $f' \neq false$, there is no other compulsory clause about f' in Δ,
3. $\neg g \notin Th(\Delta)$,
4. when f is compulsory about f' in Δ, f is also compulsory about f' for any $\Delta' \subseteq \Delta$ such that $f \in \Delta$.

The last three properties are also of a practical computational importance. For example, the last one allows us to retract information from Δ without altering the compulsory status of a clause f when f remains derivable.

A labeling is used to mark clauses in order to recognize compulsory clauses in Δ that were actually *intended to be so*. We do not label formulas in $Th(\Delta) \setminus \Delta$ since all compulsory clauses belong to Δ.

Definition 4. *Let* $f = g \vee f' \in \Delta$. f *is either explicitly labeled "required-compulsory " (in short, RC) or implicitly labeled "not-required-compulsory" (in short, NRC) about* f' *in* Δ*. By default,* f *is implicitly labeled NRC about* f' *in* Δ*. Only clauses that are compulsory about* f' *can be labeled RC about* f'*.*

For convenience, when the context does not make it ambiguous, we say that f is *RC* (resp. NRC), implicitly referring to Δ, f' and to the condition g for f' in f.

By default, a clause in Δ is thus (implicitly) marked NRC about any of its sub-clauses, including the empty one, as we expect *RC* clauses to be outnumbered by NRC ones. Let us stress again that all compulsory clauses about f' are not necessarily labeled *RC* about f': they are marked NRC when the agent/user did not require them to be compulsory.

Condition 4 in Definition 3 does not provide any hint about which g' should be checked in that condition. To circumvent the latter issue, we turn to a *prime implicate* representation of Δ.

5 Prime Implicate Representation

Definition 5. *A prime implicate of a finite set* Δ *of formulas is any clause* h *that satisfies both conditions below*

(1) $h \in Th(\Delta)$
(2) $h' \Leftrightarrow h \in Th(\Delta)$ *for every clause* h' *s.t.* $\Delta \vdash h'$ *and* $h' \vdash h$

Δ_{PI} denotes the set of all prime implicates in Δ.

Accordingly, h is a prime implicate of Δ iff h is a minimal (w.r.t. $\subseteq$) non-tautological clause amongst the set formed of the clauses l such that $\Delta \vdash l$.

Prime implicates have already been investigated in belief revision and change mainly because they provide a compact and syntax-independent yet complete representation of a belief base (see e.g. [18] and [19]) and because interesting computational tasks (like satisfiability checking and entailment) are tractable in this framework [20]. In the worst case, computing the set of prime implicates of Δ containing a clause l (a task that we will often refer to) is however not in polynomial total time unless P=NP (it is in polynomial total time when for example the clause is positive and Δ is Horn) [21]. Although the compactness and some of the computational features of a prime implicates representation happen to be welcome properties, the motivation for focusing on prime implicates is here different and stems from their intrinsic epistemological nature, as shown by the following properties.

The first two properties are straightforward and well-known. The third one shows how this prime implicate representation will help us in dealing with compulsory clauses.

Property 3. Let f be a consistent non-tautological clause.

1. $f \in Th(\Delta)$ iff $f \in Th(\Delta_{PI})$.
2. f is not subsumed in Δ iff $f \in \Delta_{PI}$.
3. $f = g \vee f'$ is the only clause about f' in Δ_{PI} iff f is compulsory in Δ.

It would be tempting to adopt Δ_{PI} as the actual representation for Δ. However, Δ can contain formulas that are not prime implicates of Δ and that thus do not belong to Δ_{PI} but in $Th(\Delta_{PI}) \setminus \Delta_{PI}$. Accordingly, we assume that Δ is a superset of Δ_{PI} and that $\Delta = \Delta_{PI} \cup \Delta_{nonPI}$, where Δ_{nonPI} contains the explicit clauses that are not prime implicates of Δ. As a consequence, $Th(\Delta) = Th(\Delta_{PI}) = Th(\Delta_{PI} \cup \Delta_{nonPI})$.

6 Restrictive Clauses about f'

Being compulsory about f' is sometimes a too strong requirement. It must sometimes be softened as follows: any strict implicant of $f = g \vee f'$ containing f' is not allowed to belong to $Th(\Delta)$ while, at the same time, other clauses about f' are allowed to exist in $Th(\Delta)$. For example, we might require that *If first-class-passenger and valid-boarding-pass then fast-lane* does not coexist with any shorter rule containing *fast-lane* in Δ whereas *If VIP then fast-lane* is allowed to exist in $Th(\Delta)$. The first clause is called restrictive about f' (where $f' =$ *fast-lane*) in Δ.

Definition 6. *When $f = g \vee f'$ is about f', f is restrictive about f' in Δ iff*

1. *$f \in \Delta$, and*
2. *$f' \notin Th(\Delta)$ and, when $f' \neq$ false, $\neg f' \notin Th(\Delta)$, and*
3. *no strict implicant of f containing f' belongs to $Th(\Delta)$.*

Useful properties of restrictive clauses are as follows.

Property 4.

1. Any compulsory clause about f' in Δ is restrictive about f' in Δ.
2. When Δ contains a restrictive clause about f' that is not compulsory about f' and when $f' \neq$ *false*, there is no compulsory clause about f' in Δ.

3. Assume a clause $f = g \vee f'$ is restrictive about f' in Δ. Let f'' s.t. $\emptyset \subset f'' \subset f'$ and $g \vee f'' \in Th(\Delta)$, we have that $g \vee f''$ is restrictive about f'' in Δ.
4. $f = g \vee f'$ is restrictive about f' in Δ iff $f \in \Delta_{PI}$.

Since permissive clauses are expected to outnumber restrictive ones, and since not all restrictive clauses are required to be so by the agent or user, the following labeling convention is followed.

Definition 7. *Let f be a clause about f' in Δ. f is marked either "required-restrictive" (in short RR) about f' in Δ, or implicitly "not-required-restrictive" (in short NRR) about f' in Δ. By default, clauses labeled NRC about f' are marked NRR about f'. Clauses marked RC about f' are marked RR about f'.*

For convenience, NRR clauses about f' are also called permissive clauses. Note that restrictive clauses are thus permissive when they are not intended to be required restrictive by the user or the agent.

7 Handling an Incoming Belief

As in most belief change approaches, we adopt a form of preference for more recent information and apply a principe of minimal change. A total ordering of RC or RR clauses (based on the time-stamp expressing when the labeling occurred) is assumed available to direct the selection of clauses to be expelled or that must have their labeling changed, when such a choice among several candidates occurs. The treatment is also intended to be well-suited for iteration, when a succession of incoming beliefs occurs and when Δ is built incrementally, at least in the sense that $\Delta = \Delta_{PI} \cup \Delta_{nonPI}$ is assumed to comply with the $\{RC, NRC, RR, NRR\}$ labeling before and after an incoming belief shows up. Due to space limitation, we only sketch the main steps of the approach when the incoming belief is a clause $f = g \vee f'$ that is intended to be compulsory about one given f'. This is the most complex situation and it allows us to illustrate principles that also apply to other cases.

Two main approaches can be distinguished based on the understanding of the precise role of f in that respect: f can be intended to either *replace* any formula about f' in Δ' (actually, as we have seen, all prime implicates containing f' in Δ'_{PI}), or *weaken* those clauses by enforcing g within their condition about f'. Consider the second situation as a case study.

Several situations can occur, depending on the current $\{RC, NRC, RR, NRR\}$ labeling of clauses in Δ'. Interestingly, we take advantage of the aforementioned properties of the prime implicate representation and of compulsory/restrictive clauses. For example, we know that when a clause is compulsory about f', it is unique in that respect and belongs to Δ_{PI}. When a clause is retracted from Δ, the remaining compulsory/restrictive clauses do not change their status (and thus the corresponding labeling of clauses does not change). When a clause $g \vee f'$ is compulsory/restrictive about f' then a same status is derived for any $g \vee f''$ where $\emptyset \subset f'' \subset f'$ when $g \vee f'' \in Th(\Delta)$.

However, let us stress on the following points. Δ_{PI} can need some updating operations when an incoming clause must belong to it. When an incoming clause is to be RC or NRC, other RC or RR clauses might need to be expelled. When they can co-exist together, their status can however need to be downgraded into RR or NRR with respect to the concerned sub-clause. In the following, we do not mention how Δ_{nonPI} is handled, because this does not involve any technical difficulty or complexity.

The specific focus is on when $f = g \vee f'$ about f' (when $f' \neq \mathit{false}$) is intended to be compulsory about f' with respect to a pre-existing Δ', in order to deliver a final base, noted Δ. Consider the case where there exists a clause $h = k \vee f'$ that is compulsory about f' in Δ. There is thus no other restrictive clause about f' in Δ.
First, Δ is initialized to Δ'. h is the unique clause about f' in Δ_{PI}. $m = g \vee k \vee f'$ must become compulsory about f' in Δ. When $m = h$ this means that m is already compulsory about f' in Δ: the procedure ends. Otherwise, h is retracted from Δ_{PI}. Clauses about f'' where $\emptyset \subset f'' \subset f$ are retracted from Δ_{PI}, too. Δ_{PI} is updated with the constraint that $m \in \Delta_{PI}$. Let $N = \{n$ s.t. $n = g \vee k \vee f''$ where $\emptyset \subset f'' \subset f'$ and $n \in Th(\Delta_{PI})\}$. Mark all elements of N by RC about f''. As a clause has been introduced within $Th(\Delta_{PI})$, it might happen that RC and RR clauses about some sub-clauses are no longer compulsory (restrictive) but only restrictive (permissive) about those sub-clauses. Accordingly, all RC and RR clauses must be checked again, according to the aforementioned total order translating a preference for more recent information. When a clause cannot be compulsory, it is checked whether it can be downgraded to RR and, in the negative case, it becomes merely permissive about the concerned sub-clause. Also, clauses that remain in Δ but cannot be any longer RR become permissive about the concerned sub-clause.

8 Conclusion and Perspectives

Most research efforts about knowledge and belief change have taken the consistent case for granted. On the contrary, we claim that taking into account a new piece of information that does not contradict the preexisting knowledge is not always a straightforward issue; it might actually involve complex reasoning paradigms. This paper intends to be a contribution to the study of these paradigms by focusing on situations where the novel information can need to prevail over the existing knowledge. In the future, we plan to extend this work to the first-order case and to non-monotonic logics. In this last respect, a first result is that the approach in the paper directly applies to fragments of non-monotonic logics that include forms of negation as failure, provided that Δ does not entail any literal. In this case, negation as failure can be replaced by standard negation to analyze in an adequate manner the interactions between generic rules, independently of any specific concrete case or data.

References

1. Fermé, E., Hansson, S.: AGM 25 years. twenty-five years of research in belief change. J. of Philosophical Logic 40, 295–331 (2011)
2. Alchourrón, C., Gärdenfors, P., Makinson, D.: On the logic of theory change: Partial meet contraction and revision functions. J. of Symbolic Logic 50(2), 510–530 (1985)
3. Katsuno, H., Mendelzon, A.: On the difference between updating a knowledge base and revising it. In: Proc. of KR 1991, pp. 387–394 (1991)
4. Gärdenfors, P.: Knowledge in Flux: Modeling the Dynamics of Epistemic States, vol. 103. MIT Press (1988)
5. Hansson, S.O.: A Textbook of Belief Dynamics. Theory Change and Database Updating. Kluwer Academic (1999)
6. Konieczny, S., Pino Pérez, R.: On the logic of merging. In: Proc. of KR 1998, pp. 488–498 (1998)
7. Konieczny, S., Grégoire, É.: Logic-based information fusion in artificial intelligence. Information Fusion 7(1), 4–18 (2006)
8. Doyle, J.: A truth maintenance system. Artificial Intelligence 12, 231–272 (1979)
9. Dalal, M.: Investigations into a theory of knowledge base revision (preliminary report). In: Proc. of AAAI 1988, vol. 2, pp. 475–479 (1988)
10. Revesz, P.Z.: On the semantics of theory change: Arbitration between old and new information. In: Proc. of PODS 1993, pp. 71–82 (1993)
11. Subrahmanian, V.S.: Amalgamating knowledge bases. ACM Transactions on Database Systems 19, 291–331 (1994)
12. Fagin, R., Ullman, J.D., Vardi, M.Y.: On the semantics of updates in databases. In: Proc. of PODS 1983, pp. 352–365 (1983)
13. Besnard, P., Grégoire, É., Ramon, S.: Logic-based fusion of legal knowledge. In: Proc. of Fusion 2012, pp. 587–592. IEEE Press, Singapor (2012)
14. Besnard, P., Grégoire, É., Ramon, S.: Enforcing Logically Weaker Knowledge in Classical Logic. In: Xiong, H., Lee, W.B. (eds.) KSEM 2011. LNCS, vol. 7091, pp. 44–55. Springer, Heidelberg (2011)
15. Besnard, P., Grégoire, É., Ramon, S.: Preemption operators. In: Proc. of ECAI 2012, pp. 893–894 (2012)
16. Besnard, P., Grégoire, É., Ramon, S.: Overriding subsuming rules. In: Liu, W. (ed.) ECSQARU 2011. LNCS, vol. 6717, pp. 532–544. Springer, Heidelberg (2011)
17. Sakallah, K.A., Simon, L. (eds.): SAT 2011. LNCS, vol. 6695. Springer, Heidelberg (2011)
18. Zhuang, Z.Q., Pagnucco, M., Meyer, T.: Implementing iterated belief change via prime implicates. In: Orgun, M.A., Thornton, J. (eds.) AI 2007. LNCS (LNAI), vol. 4830, pp. 507–518. Springer, Heidelberg (2007)
19. Bienvenu, M., Herzig, A., Qi, G.: Prime implicate-based belief revision operators. In: 20th European Conference on Artificial Intelligence (ECAI 2012), pp. 741–742 (2008)
20. Darwiche, A., Marquis, P.: A knowledge compilation map. J. Artif. Intell. Res. (JAIR) 17, 229–264 (2002)
21. Eiter, T., Makino, K.: Generating all abductive explanations for queries on propositional horn theories. In: Baaz, M., Makowsky, J.A. (eds.) CSL 2003. LNCS, vol. 2803, pp. 197–211. Springer, Heidelberg (2003)