Analysing rational properties of change operators based on forward chaining

Hassan Bezzazi, Stéphane Janot, Sébastien Konieczny and Ramón Pino Pérez

LIFL U.A. 369 du CNRS, Université de Lille I
59655 Villeneuve d'Ascq, FRANCE
E-mail: {bezzazi,janot,konieczn,pino}@lifl.fr

Abstract. We propose an abstract framework to analyse the rationality of change operators defined in a syntactical way. More precisely we propose "syntactical" postulates of rationality stemming from AGM ones. Then we introduce five change operators based on forward chaining. Finally we apply our abstract framework to analyse the rationality of our operators.

Introduction

Revision is the process of according an old knowledge base with a new evidence. In order to have a good behaviour, a revision operator must obey a minimal set of rationality requirements. For example it must obey the principle of primacy of update that demands the new evidence to be true in the new knowledge base, and the principle of minimal change that imposes that the new knowledge base has to be as close as possible to the old one. These properties are intuitive requirements one can expect from revision operators. These operators and their properties have been formally studied in philosophy, artificial intelligence and databases [1, 12, 16] and several operators have already been proposed [5, 9, 27, 28, 25].

In general, revision is a complex process [8, 20] and is not efficiently computable. The problem is that revision operators usually handle theories closed under logical consequences. Then, the computation of (all the consequences of) the new theory according to the old one and to the new information is generally prohibitive. One solution is to work with theories that are not closed under logical consequences [11, 14, 15, 22, 23, 26] and to take their logical closure only when one needs them. Of course such an approach is syntax sensitive. Another solution is to work in a restrained framework, a "weaker" (tractable) logic, instead of the classical one. It is a combination of these two approaches that is proposed here.

In this paper, we investigate five change operators based on forward chaining. The use of forward chaining provides us with an efficient way to compute the revision of a knowledge base.

Furthermore these operators are readily suitable for expert systems based on the same kind of inference. Thus, we have an easy way to include non-monotonic reasoning in such systems. Such operators may have numerous other applications, in diagnosis systems for example.

For some operators, our approach is close to REVISE [6] and Revision Programming [21] but is simpler, since we use only forward chaining on propositional formulae; in particular, we don't assume negation by failure.

We propose five knowledge change operators. The first one, called *factual update*, updates a set of facts with another set of facts coding a new evidence, according to a set of rules which can be seen as integrity constraints of the system. The other operators revise programs by programs. More precisely the second one, called *ranked revision*, is based on a hierarchy over the rules which denotes how *exceptional* the rules are, and when a new evidence arrives, it finds the least exceptional rules consistent with this new information. The third one, called *hull revision*, extends the result of ranked revision to a set which remains consistent with the new information. The fourth operator, *extended hull*, combines the approaches of hull revision and factual update operators. The fifth operator, called *selection hull*, is actually a family of operators defined by selection functions.

One of the main contributions of this work is the study of the rationality properties of these operators. To do that we introduce syntactical relativizations of the main postulates proposed in the literature for theory change. We prove that factual update can be seen as an update operator, *i.e.* satisfying a syntactical version of Katsuno and Mendelzon postulates [17]. In the same way, ranked revision and selection hull (when the selection function used to define it has good properties) can be seen as revision operators, according to Alchourrón-Gärdenfors-Makinson (AGM) postulates [1, 12]. Concerning hull revision and extended hull revision, although they seem to be rational extensions of the ranked operator, we prove that only some basic postulates of change hold.

The paper is organized as follows: section 1 contains the basic definitions of our abstract framework; in section 2.1 we define factual revision (an algorithm to compute it is given in the appendix); section 2.2 is devoted to definitions of ranked revision, hull revision and extended hull revision; in section 3 we study the properties of these knowledge change operators. In section 4 we introduce the selection hull operators. Finally, we conclude with some remarks and some perspectives for future work.

1 Preliminaries

Our framework is finite propositional logic.

A literal (or fact) is an atom or a negation of an atom. The set of literals will be denoted *Lit*. A rule is a formula of the shape $l_1 \wedge l_2 \wedge \cdots \wedge l_n \rightarrow l_{n+1}$ where l_i is a literal for $i = 1, \ldots, n+1$. A rule as before will be denoted $l_1, l_2, \ldots, l_n \rightarrow l_{n+1}$. We admit rules of the form $\rightarrow l$ which actually code facts.

Let R and L be a finite set of rules and a finite set of literals (both possibly empty) respectively. A program P is a set of the form $R \cup L$ and we will say that the elements of R are the rules of P and the elements of L are the facts of P. The set of programs will be denoted by $Prog$

Let $P = R \cup L$ be a program. We define the set of consequences by forward chaining of P, denoted $C_{fc}(P)$, as the smallest set of literals L' such that:
(i) $L \subseteq L'$.
(ii) If $l_1, l_2, \ldots, l_n \rightarrow l$ is in R and $l_i \in L'$ for $i = 1, \ldots, n$ then $l \in L'$.
(iii) If L' contains two opposite literals then $L' = Lit$

A program P is said to be consistent iff $C_{fc}(P) \neq Lit$.

Let P and L be a program and a finite set of literals respectively. L is said to be P-consistent iff $P \cup L$ is consistent (with respect to forward chaining).

In the sequel ω will denote the set of positive integers.

1.1 Revision and update postulates

We begin this section by recalling the rationality postulates that have been proposed [1, 12, 16, 17] in the area of revision theory, i.e. properties that an operator has to satisfy in order to have a "good" behaviour as a change operator. Then we will give a relativization of these notions to a syntactical abstract framework.

First let's consider the postulates for revision operators. Let φ be a formula representing a knowledge base and let μ be a formula representing a new piece of information. $\varphi \circ \mu$ will denote a formula representing the changes that μ produces on φ. The operator $\circ$ is a revision operator [1, 16] if it satisfies the following postulates:

(R1) $\vdash (\varphi \circ \mu) \rightarrow \mu$.
(R2) If $\varphi \wedge \mu$ is consistent then $\vdash (\varphi \circ \mu) \leftrightarrow (\varphi \wedge \mu)$.
(R3) If μ is consistent then $\varphi \circ \mu$ is consistent.
(R4) If $\vdash (\varphi_1 \leftrightarrow \varphi_2)$ and $\vdash (\mu_1 \leftrightarrow \mu_2)$ then $\vdash (\varphi_1 \circ \mu_1) \leftrightarrow (\varphi_2 \circ \mu_2)$.
(R5) $\vdash ((\varphi \circ \mu) \wedge \phi) \rightarrow (\varphi \circ (\mu \wedge \phi))$.
(R6) If $(\varphi \circ \mu) \wedge \phi$ is consistent then $\vdash (\varphi \circ (\mu \wedge \phi)) \rightarrow ((\varphi \circ \mu) \wedge \phi)$.

The intuitive meaning of these postulates is the following: the new piece of information must be true in the new knowledge base, which is required by (R1). (R2) states that if the new piece of information is consistent with the old knowledge base then the revision is reduced to the addition of the new piece of information to the old knowledge base. (R3) assures that if the new piece of information is consistent then the new knowledge base must be consistent. (R4) is the so called *Dalal's principle of irrelevance of syntax* and says that the result of the revision depends neither on the syntax of the new piece of information nor on the one of the knowledge base. (R5) and (R6) assure that the result of the revision is "closest" to the old base and that this notion of closeness behaves well. For more explanations on the meaning of these postulates see [12, 16].

Revision is adequate to model change of belief about a static world but, as shown in [17], is not able to cope with change in a dynamic world. Katsuno and Mendelzon have defined update operators for this case. The rationality postulates for update operators they propose are given next.

The operator $\circ$ is an update operator [17] if it satisfies the following postulates:

(U1) $\vdash (\varphi \circ \mu) \rightarrow \mu$.
(U2) If $\vdash \varphi \rightarrow \mu$ then $\vdash (\varphi \circ \mu) \leftrightarrow \varphi$.
(U3) If both φ and μ are consistent then $\varphi \circ \mu$ is also consistent.
(U4) If $\vdash (\varphi_1 \leftrightarrow \varphi_2)$ and $\vdash (\mu_1 \leftrightarrow \mu_2)$ then $\vdash (\varphi_1 \circ \mu_1) \leftrightarrow (\varphi_2 \circ \mu_2)$.
(U5) $\vdash ((\varphi \circ \mu) \wedge \phi) \rightarrow (\varphi \circ (\mu \wedge \phi))$.
(U6) If $\vdash (\varphi \circ \mu_1) \rightarrow \mu_2$ and $\vdash (\varphi \circ \mu_2) \rightarrow \mu_1$ then $\vdash (\varphi \circ \mu_1) \leftrightarrow (\varphi \circ \mu_2)$.
(U7) If φ is complete then $\vdash ((\varphi \circ \mu_1) \wedge (\varphi \circ \mu_2)) \rightarrow (\varphi \circ (\mu_1 \vee \mu_2))$.
(U8) $\vdash ((\varphi_1 \vee \varphi_2) \circ \mu) \leftrightarrow ((\varphi_1 \circ \mu) \vee (\varphi_2 \circ \mu))$.

These postulates are close to those for revision. Postulates (U1)-(U5) correspond to postulates (R1-R5) and the intuitive meaning of these postulates is: (U1) is exactly (R1),*i.e.* the new piece of information must be true in the new knowledge base, (U2) states that if the new piece of information is weaker than the knowledge base then updating by this new piece of information has no effect on the knowledge base. Notice that if the knowledge base is consistent then (U2) is weaker than (R2). (U3) assures that if the new piece of information and the old knowledge base are consistent then the new knowledge base is consistent. (U4) is exactly (R4), the principle of irrelevance of syntax. (U5) is exactly (R5). It assures that the notion of "minimal change" behaves well. (U6) says that if μ_1 is true when we update the knowledge base by μ_2 and if μ_2 is true when we update the knowledge base by μ_1, then the two updates are equivalent. (U7) states that for a complete knowledge base the conjunction of two updates contains the information of the update by the disjunction of the two pieces of information. (U8) is the disjunction rule: from a semantical point of view a knowledge base can be considered as the sum of all its possible worlds, so (U8) states that updating this sum is the sum of updating. This assures that each possible world of the knowledge base is given independent consideration.

Note that in revision and update postulates the notions of consequence, consistency and equivalence are the classical ones. We will investigate the instantiation of these postulates to different "logics". This point is essential in this paper because when manipulating syntactical objects (such as databases) we have to define some abstract notions of *consequence, conjunction, disjunction* in order to be able to analyse the properties of the operators. Particularly, we will focus in this paper on a "forward chaining logic".

More precisely, a first case of this sort of instantiation concerns the postulates for revision. This is done in next definition.

Definition 1. *Suppose that we are manipulating objects of a set Ω (a set of "formulas" or "knowledge bases"), a set $\Gamma \subseteq \Omega$ and a set $\mathcal{L}$ (the "deducible atoms") such that $\mathcal{P}(\mathcal{L}) \subseteq \Omega$. Consider we have a map $\mathcal{C} : \Omega \longrightarrow \mathcal{P}(\mathcal{L})$ ($\mathcal{C}$ is a consequence operator for the chosen logic). Finally suppose we have a function (a change operator) $\vartriangle : \Omega \times \Gamma \longrightarrow \Omega$ and a function (the "conjunction") $\otimes : \Omega \times \Gamma \longrightarrow \Omega$, such that $\otimes : \Gamma \times \Gamma \longrightarrow \Gamma$,* i.e. *the restriction of $\otimes$ to couples in Γ takes its values in Γ. Then $\vartriangle$ is said to be a **syntactical revision operator***

(with respect to $\mathcal{C}$ and $\otimes$) if the following postulates hold:

(SR1) $\mathcal{C}(Y) \subseteq \mathcal{C}(X \vartriangle Y)$.
(SR2) *If* $\mathcal{C}(X \otimes Y) \neq \mathcal{L}$ *then* $\mathcal{C}(X \vartriangle Y) = \mathcal{C}(X \otimes Y)$.
(SR3) *If* $\mathcal{C}(Y) \neq \mathcal{L}$ *then* $\mathcal{C}(X \vartriangle Y) \neq \mathcal{L}$.
(SR5) $\mathcal{C}(X \vartriangle (Y \otimes Z)) \subseteq \mathcal{C}((X \vartriangle Y) \otimes Z)$.
(SR6) *If* $\mathcal{C}((X \vartriangle Y) \otimes Z) \neq \mathcal{L}$ *then* $\mathcal{C}((X \vartriangle Y) \otimes Z) \subseteq \mathcal{C}(X \vartriangle (Y \otimes Z))$.

Let us remark that the postulates **(SRi)** are the natural counterparts of postulates **(Ri)** when we interpret $\otimes$ as the conjunction of formulas and thinking X consistent iff $\mathcal{C}(X) \neq \mathcal{L}$.

There is no postulate corresponding to **(R4)** (alias **U4**), the postulate of irrelevance of syntax, because the operators we will define are in general syntax-sensitive. Nevertheless in our abstract setting we could define the counterpart of **(R4)** in the following way:

(IS) If $\mathcal{C}(X) = \mathcal{C}(Y)$ and $\mathcal{C}(Z) = \mathcal{C}(W)$ then $\mathcal{C}(X \vartriangle Z) = \mathcal{C}(Y \vartriangle W)$.

Unfortunately this does not hold for our operators as we will see in observation 6.

The second case of instantiation we consider concerns the postulates for update. This is the object of next definition.

Definition 2. *Consider two sets Ω and $\mathcal{L}$ such that there is a set $\Gamma \subseteq \mathcal{P}(\mathcal{L})$ with $\Gamma \subseteq \Omega$. Let $\mathcal{C}$ be a function $\mathcal{C} : \Omega \longrightarrow \mathcal{P}(\mathcal{L})$. Suppose we have a function (a change operator) $\vartriangle : \Omega \times \Gamma \longrightarrow \Omega$. Now suppose that we have an associative "connector" (our "disjunction") $\oplus : \Gamma \times \Gamma \longrightarrow \Gamma$ such that $\mathcal{C}(X \oplus Y) = \mathcal{C}(X) \cap \mathcal{C}(Y)$,* i.e. *the behaviour of $\oplus$ with respect to $\mathcal{C}$ is like a disjunction. We also suppose we have a function $\otimes : \Omega \times \Gamma \longrightarrow \Omega$, such that $\otimes : \Gamma \times \Gamma \longrightarrow \Gamma$,* i.e. *the restriction of $\otimes$ to couples in Γ takes its values in Γ. The operator $\vartriangle$ is said to be a* ***syntactical update operator*** *with respect to $\mathcal{C}$, $\otimes$ and $\oplus$ if the following postulates hold:*

(SU1) $\mathcal{C}(Y) \subseteq \mathcal{C}(X \vartriangle Y)$.
(SU2) *If* $\mathcal{C}(Y) \subseteq \mathcal{C}(X)$ *then* $\mathcal{C}(X \vartriangle Y) = \mathcal{C}(X)$.
(SU3) *If* $\mathcal{C}(X) \neq \mathcal{L}$ *and* $\mathcal{C}(Y) \neq \mathcal{L}$ *then* $\mathcal{C}(X \vartriangle Y) \neq \mathcal{L}$.
(SU5) $\mathcal{C}(X \vartriangle (Y \otimes Z)) \subseteq \mathcal{C}((X \vartriangle Y) \otimes Z)$.
(SU6) *If* $\mathcal{C}(Y_1) \subseteq \mathcal{C}(X \vartriangle Y_2)$ *and* $\mathcal{C}(Y_2) \subseteq \mathcal{C}(X \vartriangle Y_1)$ *then* $\mathcal{C}(X \vartriangle Y_1) = \mathcal{C}(X \vartriangle Y_2)$.
(SU8) $\mathcal{C}((X \oplus Y) \vartriangle Z) = \mathcal{C}((X \vartriangle Z) \oplus (Y \vartriangle Z))$.

The postulates **(SUi)** are the natural counterparts of postulates **(Ui)** (notice that we have asked the "connector" $\oplus$ to have the behaviour of a disjunction with respect to the notion of consequence $\mathcal{C}$). Remark also that there is no postulate corresponding to **(U7)** because in general the image of a couple of elements of Γ under $\oplus$ will not belong to Γ.

As for syntactical revision operators, there is no postulate corresponding to **U4**.

2 Some syntactical change operators

The purpose of this section is to define some change operators essentially based on forward chaining. The first one, factual update, updates a set of facts by a set of facts coding a change in the world according to a set of integrity constraints. The following three ones, namely ranked revision, hull revision and extended hull revision are based on a ranking of sentences of the programs. We will analyse the rational properties they satisfy in section 3.

2.1 Factual update

Let P be a fixed program which in this context can be seen as our background theory or our integrity constraints. Let L be a set of facts which can be seen as our beliefs about the world. We would like to define the change produced by a set of facts L' coding a new piece of information about the world. The following definition describes the result of this change:

$$L \diamond_P L' = \begin{cases} Lit & \text{if } L \text{ or } L' \text{ is not } P\text{-consistent} \\ \langle L_1 \cup L', \ldots, L_n \cup L' \rangle & \text{otherwise} \end{cases}$$

where $\{L_1, \ldots, L_n\}$ is the set of subsets of L which are maximal and $P \cup L'$-consistent.

So more generally than a set of facts L we are considering unordered tuples of sets of facts $\langle L_1, \ldots, L_n \rangle$ called *flocks* in the literature [9]. Such flocks can be also seen as multisets.

We define the concatenation of flocks '$\cdot$' in the obvious way:

$$\langle L_1, \ldots, L_n \rangle \cdot \langle L'_1, \ldots, L'_m \rangle \overset{def}{=} \langle L_1, \ldots, L_n, L'_1, \ldots, L'_m \rangle$$

and we define the change produced in a flock by a new piece of information by the following

$$\langle L_1, \ldots, L_n \rangle \diamond_P L' \overset{def}{=} \begin{cases} Lit & \text{if } L' \text{ or all the } L_i \text{ are not } P\text{-consistent} \\ (L_{i_1} \diamond_P L') \cdot (L_{i_2} \diamond_P L') \cdots (L_{i_k} \diamond_P L') & \text{otherwise} \end{cases}$$

where $\{L_{i_1}, \ldots, L_{i_k}\}$ is the set of all sets in $\{L_1, \ldots, L_n\}$ which are consistent with P.

In order to investigate the relation between $\diamond_P$ and the postulates of change we need to define the intensional content (the consequences) of a flock $\mathcal{F} = \langle L_1, \ldots, L_n \rangle$. So we define the consequences by forward chaining (with respect to P) of such a flock, denoted $C^P_{fc}(\mathcal{F})$, by the following:

$$C^P_{fc}(\mathcal{F}) = \bigcap_{i=1}^{n} C_{fc}(L_i \cup P)$$

So, we adopt here a sceptical point of view, since the consequences of a flock are the facts that are true in every elements of the flock.

In section 3 we will show that $\diamond_P$ is a syntactical update operator.

Example 1. We consider the program P and the set of literals L, defined by $P = \{a, b \rightarrow c \; ; \; a, d \rightarrow c\}$ and $L = \{a, b, d\}$. Put $L' = \{\neg c\}$. L' is not $P \cup L$ consistent and then some facts must be "retracted" from the old base L. Then it is easy to see that

$$L \diamond_P L' = \langle \{a\} \cup \{\neg c\}, \{b, d\} \cup \{\neg c\} \rangle$$

For the sake of completeness we give in the appendix an algorithm to compute $L \diamond_P L'$.

2.2 Ranked revision, hull revision and extended hull revision

In the case of factual update the program is fixed and we restrain the new piece of information to be a set of facts. When it is not the case a natural question that one may ask is how to change a program when a new piece of information arrives. The aim of this section is to give an answer to this question even when the new piece of information is a program.

The change operators introduced in this section are inspired by the duality existing between revision and rational inference relations [10, 13]. So the first operator can be seen as the 'relativization' of the rational closure [19] to the forward chaining logic. The second operator is an extension of the first one and it is aimed to satisfy a little bit more of transitivity [3, 4].

Definition 3 (Exceptional sets of literals and rules). *Let P be a program. A set of literals L is said to be exceptional with respect to P iff L is not P-consistent and a rule $L \longrightarrow l$ of P is said to be exceptional in P iff L is exceptional in P.*

Notice that, when the body of a rule is empty, this rule will be exceptional iff P is not consistent. In this case all the rules are exceptional.

A similar definition of exceptionality for a formula can be found in [19].

Definition 4 (Base). *Let P be a program. Let $(P_i)_{i \in \omega}$ be the decreasing sequence defined by: P_0 is P and P_{i+1} is the set of all exceptional rules of P_i. Since P is finite there is a smallest integer n_0 such that for all $m \geq n_0$ we have $P_m = P_{n_0}$. If $P_{n_0} \neq \emptyset$ we say that $\langle P_0, \ldots, P_{n_0}, \emptyset \rangle$ is the base of P. If $P_{n_0} = \emptyset$ we say that $\langle P_0, \ldots, P_{n_0} \rangle$ is the base of P.*

Thus a program P has intrinsically a hierarchy, its base, in which the greater n is, the more exceptional the information in P_n is.

Definition 5 (Rank function). *Fix a program P and let $\langle P_0, \ldots, P_n \rangle$ be the base of P. Let $\rho : Prog \longrightarrow \omega$ be the rank function defined as follows: $\rho(P') =$*

$\min\{i \in \omega : P'$ *is* P_i*-consistent*$\}$ *if* P *is consistent, otherwise* $\rho(P') = n$. *It is a fact that if* $P' \subseteq P''$ *then* $\rho(P') \leq \rho(P'')$.

Notice that actually the rank function has two parameters. Thus in the notation $P_{\rho(P')}$, $\rho(P')$ *denotes the rank of* P' *with respect to* P.

The rank of a new program P' denotes how this program is exceptional according to the old program P.

Definition 6 (Ranked revision). *Let* P *and* P' *be two programs. We define the ranked revision of* P *by* P', *denoted* $P \circ_{rk} P'$, *as follows:*

$$P \circ_{rk} P' = P_{\rho(P')} \cup P'$$

In other words we take from the base of P the first program (the least exceptional) that agrees with the new piece of information.

We will now slightly generalize the ranked revision operator, and define the hull revision operator from the definition of the "hull of P" $((h(P))$.

Let $I_P(P')$ be the set of maximal subsets of P which are consistent with P' and which contain $P_{\rho(P')}$. Define $h : Prog \longrightarrow \mathcal{P}(P)$ by $h(P') = \bigcap I_P(P')$

Definition 7 (Hull revision). *The hull revision of a program* P *by a program* P' *denoted* $P \circ_h P'$ *is defined as follows*

$$P \circ_h P' = h(P') \cup P'$$

Remark 1. By the definitions it is easy to see that

$$C_{fc}(P \circ_{rk} P') \subseteq C_{fc}(P \circ_h P')$$

Thus one can say that $\circ_h$ is a conservative extension of $\circ_{rk}$. It keeps information that does not come into account in the contradiction.

Remember that in the definition of hull revision of P by P' we first calculate the set $I_P(P')$ of subsets of P maxiconsistent with P' and containing $P_{\rho(P')}$; then we take a very sceptical approach putting $P \circ_h P' = (\bigcap I_P(P')) \cup P'$. What we want now is to be more permissive and we are going to manipulate $I_P(P')$ as a flock. The ideas here are very close to the ones of factual update (cf section 2.1).

First given two programs P and P' we are going to define $P \circ_{eh} P'$. Let $I_P(P')$ be as before. Then we put:

$$P \circ_{eh} P' = \begin{cases} \langle H_1 \cup P', \ldots, H_n \cup P' \rangle & \text{if } I_P(P') = \{H_1, \ldots, H_n\} \\ P' & \text{if } I_P(P') = \emptyset \end{cases}$$

Remark that the result is a flock of programs.

Now suppose that $\mathcal{F}$ is a flock of programs, say $\mathcal{F} = \langle Q_1, \ldots, Q_n \rangle$. Then we define $\mathcal{F} \circ_{eh} P$ by putting

$$\mathcal{F} \circ_{eh} P = (Q_1 \circ_{eh} P) \cdot (Q_2 \circ_{eh} P) \cdots (Q_n \circ_{eh} P)$$

where as in section 2.1 '$\cdot$' is the concatenation of flocks.

If $\mathcal{F} = \langle Q_1, \ldots, Q_n \rangle$ is a flock of programs we define $C_{fc}(\mathcal{F})$ by putting $C_{fc}(\mathcal{F}) = \bigcap_{i=1}^{n} C_{fc}(Q_i)$.

Remark 2. Notice that with this definition $\circ_{eh}$ is a conservative extension of $\circ_h$, *i.e.* $C_{fc}(P \circ_h P') \subseteq C_{fc}(P \circ_{eh} P')$. For this reason the operator $\circ_{eh}$ is called *extended hull revision* .

We will identify a program P with the flock $\langle P \rangle$.

In section 3 we will show that $\circ_{rk}$ is a syntactical revision operator and that the operators $\circ_h$ and $\circ_{eh}$ enjoy some of the properties of syntactical revision operators.

2.3 Examples of ranked revision, hull revision and extended hull revision.

In this subsection we give examples that illustrate the behaviour of ranked revision, hull, and extended hull operators and at the same time the differences between them.

In the following examples we are interested in facts one can infer from $P \circ P'$, *i.e.* the facts $l \in C_{fc}(P \circ P')$ when $\circ$ is $\circ_{rk}$, $\circ_h$ or $\circ_{eh}$. For simplicity we will take for P a set of rules with non empty body and P' a set of facts.

Example 2. Consider $P = \{b \to f \ ; \ b \to w \ ; \ o \to b \ ; \ o \to \neg f\}$ where b, o, f, w stand respectively for birds, ostriches, fly and have wings. It is easy to see that the base is $< P_0, P_1, P_2 >$ where

$P_0 = \{b \to f \ ; \ b \to w \ ; \ o \to b \ ; \ o \to \neg f\}$
$P_1 = \{o \to b \ ; \ o \to \neg f\}$
$P_2 = \emptyset$

Notice that $\rho(b) = 0$ so $I_P(b) = P_0 = P$ and therefore

$$P \circ_{rk} \{b\} = P \circ_h \{b\} = P \circ_{eh} \{b\} = P \cup \{b\}$$

$$C_{fc}(P \circ_{rk} \{b\}) = \{b, f, w\}$$

Thus on this example the three operators have exactly the same behaviour.

For the same P, an easy computation shows that $\rho(o) = 1$. Since the set $\{o \to b \ ; \ o \to \neg f \ ; \ b \to w\}$ is the unique extension of P_1 consistent with $\{o\}$ we have $h(o) = \{o \to b \ ; \ o \to \neg f \ ; \ b \to w\}$ so

$$P \circ_h \{o\} = P \circ_{eh} \{o\} = \{o \to b \ ; \ o \to \neg f \ ; \ b \to w\} \cup \{o\}$$

Since $\rho(o) = 1$ we have $P \circ_{rk} \{o\} = \{o \to b \ ; \ o \to \neg f\} \cup \{o\}$. Therefore

$$C_{fc}(P \circ_h \{o\}) = C_{fc}(P \circ_{eh} \{o\}) = \{b, o, \neg f, w\}$$

but $C_{fc}(P \circ_{rk} \{o\}) = \{b, o, \neg f\}$. Thus this example shows that hull revision and extended hull revision keep more information from the old program than ranked revision.

Another classic taxonomic example (the calculations are left to the reader) is given by

Example 3. $P = \{m \to s \ ; \ c \to m \ ; \ c \to \neg s \ ; \ n \to c \ ; \ n \to s\}$ where m, s, c, n stand respectively for mollusc, shell, cephalopod and nautili. The base is $< P_0, P_1, P_2, P_3 >$ where

$P_0 = \{m \to s \ ; \ c \to m \ ; \ c \to \neg s \ ; \ n \to c \ ; \ n \to s\}$
$P_1 = \{c \to m \ ; \ c \to \neg s \ ; \ n \to c \ ; \ n \to s\}$
$P_2 = \{n \to c \ ; \ n \to s\}$
$P_3 = \emptyset$

We have

$$C_{fc}(P \circ_h \{n\}) = \{n, c, s, m\} = C_{fc}(P \circ_{eh} \{n\})$$

and

$$C_{fc}(P \circ_{rk} \{n\}) = \{n, c, s\}$$

this shows that the hull (and extended hull) revision allows more inferences than ranked revision. In some other cases the revisions coincide, for instance

$$C_{fc}(P \circ_h \{c, \neg n\}) = C_{fc}(P \circ_{eh} \{c, \neg n\}) = \{c, \neg n, m, \neg s\} = C_{fc}(P \circ_{rk} \{c, \neg n\})$$

Now we consider an example showing that in general the extended hull revision allows more inferences than the hull revision

Example 4. Take the following program

$$P = \{a, b \to c \ ; \ a, \neg c \to d \ ; \ b, \neg c \to d \ ; \ a \ ; \ b\}$$

Put $P' = \{\neg c\}$. The base of P is $\langle P_0, P_1, P_2 \rangle$ with

$P_0 = \{a, b \to c \ ; \ a, \neg c \to d \ ; \ b, \neg c \to d \ ; \ a \ ; \ b\}$
$P_1 = \{a, \neg c \to d \ ; \ b, \neg c \to d\}$
$P_2 = \emptyset$

Clearly $\rho(P') = 1$ and $I_P(P') = \{P_1 \cup \{a, b \to c, a\}, P_1 \cup \{a, b \to c, b\}, P_1 \cup \{a, b\}\}$. Thus $h(P') = P_1$ and therefore $d \notin C_{fc}(P \circ_h P') = C_{fc}(P_1 \cup \neg c)$. Whereas $d \in C_{fc}(P \circ_{eh} P')$ because for any $Q \in I_P(P')$ we have $d \in C_{fc}(Q \cup \{\neg c\})$.

2.4 Computing hull revision and extended hull revision

In this subsection we show how, via a simple coding, we can compute the hull revision by using the factual revision defined in section 2.1.

The base $< P_0, \ldots, P_n >$ of P is easily computed.

To compute the class of maximal subsets of P which are consistent with L and which contain $P_{\rho(P')}$ we use the update algorithm given in the appendix in the following way.

Let $\ell' : P \longrightarrow \{r_1, \ldots, r_m\}$ and $\ell'' : P' \longrightarrow \{q_1, \ldots, q_k\}$ be two bijections where the r_i and the q_j are new atoms. Define $\ell : P \cup P' \longrightarrow \{r_1, \ldots, r_m\} \cup \{q_1, \ldots, q_k\}$ by $\ell(r) = \ell'(r)$ if $r \in R$ and $\ell(r') = \ell''(r')$ if $r' \in P'$. Let $M(P)$ be the modification of P in the following way: each rule $L \to l$ of P is replaced by the rule $r, L \to l$ where $r = \ell(L \to l)$. Analogously, let $M(P')$ be the modification of P' in the following way: each rule $L \to l$ of P' is replaced by the rule $r, L \to l$ where $r = \ell(L \to l)$. Note that the maximal subsets of P which are consistent with P' and which contain $P_{\rho(P')}$ are then those corresponding to the maximal subsets of the base of facts $\ell(P)$ computed as being its possible updatings with respect to $M(P) \cup M(P')$ by $\ell(P') \cup \ell(P_{\rho(P')})$. More precisely we have:

$$P \circ_h P' = \ell^{-1}\{ \bigcap [\ell(P) \diamond_{M(P) \cup M(P')} (\ell(P') \cup \ell(P_{\rho(P')}))\ ]\ \}$$

In an analogous way

$$P \circ_{eh} P' = \ell^{-1}[\ell(P) \diamond_{M(P) \cup M(P')} (\ell(P') \cup \ell(P_{\rho(P')}))\ ]$$

In order to illustrate this method take the following example:

Example 5. $P = \{b \to f, b \to w, o \to b, o \to \neg f\}$. Let $P' = \{o\}$. Define $\ell : P \cup P' \longrightarrow \{1, 2, 3, 4, 5\}$ such that $M(P) = \{1, b \to f\ ,\ 2, b \to w\ ,\ 3, o \to b\ ,\ 4, o \to \neg f\}$ and $M(P') = \{5 \to o\}$. We have seen above that $P_{\rho(P')} = \{o \to b, o \to \neg f\}$ so $\ell(P_{\rho(P')}) = \{3, 4\}$. Therefore

$$\begin{aligned}\ell(P) \diamond_{M(P) \cup M(P')} (\ell(P') \cup \ell(P_{\rho(P')})) &= \{1, 2, 3, 4\} \diamond_{M(P) \cup M(P')} \{5\} \cup \{3, 4\} \\ &= \langle \{2, 3, 4, 5\} \rangle\end{aligned}$$

and so $C_{fc}(P \circ_h P') = C_{fc}(\ell^{-1}(\{2, 3, 4, 5\})) = C_{fc}(\{o \to b, o \to \neg f, b \to w, o\}) = \{o, b, \neg f, w\}$.

3 Change properties for $\diamond_P$, $\circ_{rk}$, $\circ_h$ and $\circ_{eh}$

In this section we analyse the rationality of our operators.

More exactly we will show that factual update can be seen as an update operator in our relativized version of the Katsuno-Mendelzon postulates and that ranked revision can be considered as a revision operator in in our relativized version of the Alchourrón-Gärdenfors-Makinson postulates. And we give some properties satisfied by hull revision and extended hull revision.

We begin with an observation the proof of which is straightforward.

Observation 1 *The functions C_{fc} and C^P_{fc} are idempotent and monotonic,* i.e. *$\mathcal{C}(\mathcal{C}(X)) = \mathcal{C}(X)$ and $\mathcal{C}(X) \subseteq \mathcal{C}(X \cup Y)$. Thus if $X \subseteq \mathcal{C}(Y)$ then $\mathcal{C}(X) \subseteq \mathcal{C}(Y)$ for $\mathcal{C} = C_{fc}$ or $\mathcal{C} = C^P_{fc}$ and X and Y in the appropriate domains.* □

We will show that $\diamond_P$ is a syntactical update operator. In order to do that we must give the instantiations for $\mathcal{L}$, Ω, $\mathcal{C}$ $\oplus$ and $\otimes$ used in definition 2. We do that in next definition.

Definition 8. *$\mathcal{L} = Lit$; $\Gamma = \mathcal{P}(\mathcal{L})$; Ω is the set of flocks in which each element is in Γ. We identify $L \in \Gamma$ with the flock $\langle L \rangle$. With this identification we have $\Gamma \subseteq \Omega$. The function $\mathcal{C} : \Omega \longrightarrow \mathcal{P}(\mathcal{L})$ is defined by $\mathcal{C} = C^P_{fc}$. The function $\oplus : \Omega \times \Omega \longrightarrow \Omega$ is defined by $\mathcal{F}_1 \oplus \mathcal{F}_2 = \mathcal{F}_1 \cdot \mathcal{F}_2$ (notice that this definition satisfies the requirement $\mathcal{C}(\mathcal{F}_1 \oplus \mathcal{F}_2) = \mathcal{C}(\mathcal{F}_1) \cap \mathcal{C}(\mathcal{F}_2)$. The function $\otimes : \Omega \times \Gamma \longrightarrow \Omega$ is defined in the following way:*

$$\langle L_1, \ldots, L_n \rangle \otimes L = \langle L_1 \cup L, \ldots, L_n \cup L \rangle$$

Notice that with the previous identification we have $L \otimes L' = L \cup L'$, that is $\otimes$ satisfies the requirement that its restriction to couples of Γ takes its values in Γ.

With this definition we can state the following theorem:

Theorem 2. *The operator $\diamond_P$ is a syntactical update operator. More precisely, taking $\mathcal{L}$, Γ, Ω, $\mathcal{C}$ and $\otimes$ as in definition 8 the postulates* ***SU1****,* ***SU2****,* ***SU3****,* ***SU5****,* ***SU6*** *and* ***SU8*** *hold.*

Proof: SU1: We want to show that $\mathcal{C}(L') \subseteq \mathcal{C}(\mathcal{F} \diamond_P L')$, *i.e.* that $C^P_{fc}(L')$ is a subset of $C^P_{fc}(\mathcal{F} \diamond_P L')$. If L' is P-inconsistent or all the elements of $\mathcal{F}$ are P-inconsistent, the result follows trivially.

Now suppose that $\mathcal{F} \diamond_P L' = \langle L_1, \ldots, L_n \rangle$. By definition of $\diamond_P$, we have $L' \subseteq L_i$ for $i = 1, \ldots, n$. Therefore $L' \subseteq \cap_{i=1}^n C^P_{fc}(L_i) = C^P_{fc}(\mathcal{F} \diamond_P L')$. We conclude using observation 1

SU2: Suppose that $C^P_{fc}(L') \subseteq C^P_{fc}(\mathcal{F})$. We want to show that $C^P_{fc}(\mathcal{F}) = C^P_{fc}(\mathcal{F} \diamond_P L')$. If $C^P_{fc}(\mathcal{F}) = Lit$ or $C^P_{fc}(L') = Lit$ then the result follows trivially from definitions. Thus, assume that $C^P_{fc}(\mathcal{F}) \neq Lit$ and $C^P_{fc}(L') \neq Lit$ By the assumption, we can suppose that $\mathcal{F} = \langle L_1, \ldots, L_n \rangle$ and $\mathcal{F} \diamond_P L' = (L_{i_1} \diamond_P L') \cdots (L_{i_K} \diamond_P L')$ where the set $\{L_{i_1} \ldots L_{i_K}\}$ is the maximal subset of $\{L_1 \ldots L_n\}$ such that each L_{i_j} is P-consistent. Since $C^P_{fc}(L') \subseteq C^P_{fc}(L_{i_j})$ we have that $L_{i_j} \diamond_P L' = L_{i_j} \cup L'$ and by observation 1 that $C^P_{fc}(L_{i_j} \cup L') = C^P_{fc}(L_{i_j})$. Thus

$$C^P_{fc}(\mathcal{F} \diamond_P L') = C^P_{fc}(L_{i_1} \cup L', \ldots, L_{i_k} \cup L') = \bigcap_{j=1}^{k} C^P_{fc}(L_{i_j}) = \bigcap_{i=1}^{n} C^P_{fc}(L_i) = C^P_{fc}(\mathcal{F})$$

where the next to last equality is due to the fact that if L_i is different of all L_{i_j} then $C^P_{fc}(L_i) = Lit$.

SU3: It is straightforward by definition of $\diamond_P$.

SU5: We want to show that $C^P_{fc}(\mathcal{F} \diamond_P (L \otimes L')) \subseteq C^P_{fc}((\mathcal{F} \diamond_P L) \otimes L')$. First we prove the result when $\mathcal{F}$ is a flock with one element, say $\mathcal{F} = \langle H \rangle$. If H is P-inconsistent or $L \cup L'$ is P-inconsistent the result is quite straightforward. Now suppose that both H and $P \cup P'$ are P-consistent. By definition we have

$$\begin{aligned} H \diamond_P (L \otimes L') &= \langle L_1 \cup L \cup L', \ldots, L_n \cup L \cup L' \rangle \\ (H \diamond_P L) \otimes L' &= \langle K_1 \cup L, \ldots, K_p \cup L \rangle \otimes L' \\ &= \langle K_1 \cup L \cup L', \ldots, K_p \cup L \cup L' \rangle \end{aligned}$$

where L_i is a maximal subset of H such that $L_i \cup L \cup L'$ is P-consistent, for $i = 1, \ldots, n$, K_j is a maximal subset of H such that $K_j \cup L$ is P-consistent, for $j = 1, \ldots, p$. Notice that either $C^P_{fc}(K_j \cup L \cup L') = Lit$ or there exists an m such that $K_j = L_m$. Therefore if $l \in \bigcap_{i=1}^n C_{fc}(P \cup L_i \cup L' \cup L'')$ then $l \in \bigcap_{j=1}^r C_{fc}(P \cup K_{i_j} \cup L' \cup L'')$.

Now we prove the general case. Suppose $\mathcal{F} = \langle L_1, \ldots, L_n \rangle$. If $\mathcal{F}$ is P-inconsistent or $L \cup L'$ is P-inconsistent the result is straightforward. Thus assume that $\mathcal{F}$ and $P \cup P'$ are P-consistent. Then

$$\begin{aligned} \mathcal{F} \diamond_P (L \otimes L') &= (L_{i_1} \diamond_P (L \cup L')) \cdots (L_{i_k} \diamond_P (L \cup L')) \\ (\mathcal{F} \diamond_P L') \otimes L' &= ((L_{i_1} \diamond_P L \cdots L_{i_k} \diamond_P L)) \otimes L' \\ &= ((L_{i_1} \diamond_P L) \otimes L') \cdots ((L_{i_k} \diamond_P L) \otimes L') \end{aligned}$$

where $\{L_{i_1}, \ldots, L_{i_k}\}$ is the maximal subset of $\{L_1, \ldots, L_n\}$ such that each element is P-consistent. By the first case we have

$$C^P_{fc}((L_{i_j} \diamond_P (L \otimes L')) \subseteq C^P_{fc}((L_{i_j} \diamond_P L) \otimes L') \quad \text{for } j = 1, \ldots, k$$

Therefore $C^P_{fc}((\mathcal{F} \diamond_P (L \otimes L')) \subseteq C^P_{fc}((\mathcal{F} \diamond_P L) \otimes L')$.

SU6: Suppose $C^P_{fc}(L_2) \subseteq C^P_{fc}(\mathcal{F} \diamond_P L_1)$ and $C^P_{fc}(L_1) \subseteq C^P_{fc}(\mathcal{F} \diamond_P L_2)$. We want to show that $C^P_{fc}(\mathcal{F} \diamond_P L_1) = C^P_{fc}(\mathcal{F} \diamond_P L_2)$. When one of $\mathcal{F}$, L_1, L_2 is P-inconsistent the result is trivial. So suppose all three of them P-consistent.

First we suppose that $\mathcal{F} = L$. By the assumption it is clear that $L_2 \subseteq C^P_{fc}(L \diamond_P L_1)$ and $L_1 \subseteq C^P_{fc}(L \diamond_P L_2)$ Put

$$\begin{aligned} L \diamond_P L_1 &= \langle L^1_1, \ldots, L^1_{n_1} \rangle \otimes L_1 \\ L \diamond_P L_2 &= \langle L^2_1, \ldots, L^2_{n_2} \rangle \otimes L_2 \end{aligned}$$

where L^1_i is a maximal subset of L such that $L^1_i \cup L_1$ is P-consistent for $i = 1, \ldots, n_1$ and L^2_j is a maximal subset of L such that $L^2_j \cup L_2$ is P-consistent for $j = 1, \ldots, n_2$. From the hypothesis it is easy to see that:
(a) $L_1 \cup L^2_j$ is P-consistent for $j = 1, \ldots n_2$ and
(b) $L_2 \cup L^1_i$ is P-consistent for $i = 1, \ldots n_1$.
From (b) we have $\forall\ i \in \{1, \ldots, n_1\}\ \exists\ j \in \{1, \ldots, n_2\}$ such that $L^1_i \subseteq L^2_j$ and from (a) we have $\forall\ j \in \{1, \ldots, n_2\}\ \exists\ i \in \{1, \ldots, n_1\}$ such that $L^2_j \subseteq L^1_i$. But

this implies, by maximality of sets L_i^1 and L_j^2, that $n_1 = n_2$ and there is a permutation σ of $\{1,\ldots,n_1\}$ such that $L_i^1 = L_{\sigma(i)}^2$. Without loss of generality we can suppose that σ is the identity. Finally note that

$$\begin{aligned}C_{fc}^P(\langle L_1^1,\ldots,L_{n_1}^1\rangle \otimes L_1) &= C_{fc}^P(\langle L_1^1,\ldots,L_{n_1}^1\rangle \otimes (L_1 \cup L_2))\\ &= C_{fc}^P(\langle L_1^2,\ldots,L_{n_1}^2\rangle \otimes (L_2 \cup L_1))\\ &= C_{fc}^P(\langle L_1^2,\ldots,L_{n_1}^2\rangle \otimes L_2)\end{aligned}$$

The first and third equalities follow from the hypothesis and observation 1.

Now we prove the general case. Put $\mathcal{F} = \langle H_1,\ldots,H_n\rangle$ and suppose $C_{fc}^P(L_2) \subseteq C_{fc}^P(\mathcal{F} \diamond_P L_1)$ and $C_{fc}^P(L_1) \subseteq C_{fc}^P(\mathcal{F} \diamond_P L_2)$. Then

$$\mathcal{F} \diamond_P L_i = (H_{j_1} \diamond_P L_i)\cdots(H_{j_k} \diamond_P L_i) \quad i = 1,2$$

But it is easy to see that $L_1 \subseteq C_{fc}^P(H_{j_m} \diamond_P L_2)$ and $L_2 \subseteq C_{fc}^P(H_{j_m} \diamond_P L_1)$ for $m = 1,\ldots,k$. Then, because of the first case, $C_{fc}^P(H_{j_m} \diamond_P L_1) = C_{fc}^P(H_{j_m} \diamond_P L_2)$ and therefore $C_{fc}^P(\mathcal{F} \diamond_P L_1) = C_{fc}^P(\mathcal{F} \diamond_P L_2)$.

SU8: It is trivially verified because of definitions. □

We will show now that $\circ_{rk}$ is a syntactical revision operator and that the operator $\circ_h$ has some of the properties of syntactical revision operators. In order to do that we give the instantiations of the sets $\mathcal{L}$, Ω and the functions $\mathcal{C}$ and $\otimes$ used in 1.

Definition 9. *$\mathcal{L} = Lit$. $\Omega = \Gamma = Prog$. Clearly $\mathcal{P}(\mathcal{L}) \subseteq \Omega$. The consequence operator $\mathcal{C} : \Omega \longrightarrow \mathcal{P}(\mathcal{L})$ is defined by $\mathcal{C} = C_{fc}$. The function $\otimes : \Omega \times \Omega \longrightarrow \Omega$ is defined by $P \otimes P' = P \cup P'$.*

Theorem 3. *The operator $\circ_{rk}$ is a syntactical revision operator. More precisely, taking $\mathcal{L}$, Ω, $\mathcal{C}$ and $\otimes$ like in definition 9, the postulates **SR1**, **SR2**, **SR3**, **SR5** and **SR6** hold.*

*The operator $\circ_h$ satisfies the postulates **SR1**, **SR2** and **SR3** but it does not satisfy **SR5** nor **SR6**.*

Proof: We do the verifications for $\circ_h$ concerning the postulates **SR1**, **SR2**, **SR3** (the postulates for $\circ_{rk}$ are verified in an analogous way). Then we verify **SR5** and **SR6** for $\circ_{rk}$. Finally we show counterexamples to **SR5** and **SR6** for $\circ_h$.

SR1: We want to show that $C_{fc}(P') \subseteq C_{fc}(P \circ_h P')$. This is clearly true because $P \circ_h P' = h(P') \cup P'$.

SR2: Suppose that $\mathcal{C}(P \otimes P') \neq \mathcal{L}$, *i.e.* $C_{fc}(P \cup P') \neq Lit$. We want to show that $P \circ_h P' = P \cup P'$. This is straightforward because $C_{fc}(P \cup P') \neq Lit$ implies $\rho(P') = 0$.

SR3: Suppose $\mathcal{C}(P') \neq \mathcal{L}$, *i.e.* P' is consistent. We want to show that $P \circ_h P'$ is also consistent, *i.e.* $\mathcal{C}(P \circ_h P') \neq \mathcal{L}$. This is true because $P \circ_h P' = h(P') \cup P'$

and $h(P')$ is by definition contained in a set consistent with P'.

SR5 and **SR6** for $\circ_{rk}$: We suppose that $\mathcal{C}((P \circ_{rk} P') \otimes P'') \neq \mathcal{L}$, *i.e.* $(P \circ_{rk} P') \cup P''$ is consistent (otherwise **SR5** is trivial). We want to prove that $\mathcal{C}((P \circ_{rk} P') \otimes P'') = \mathcal{C}(P \circ_{rk} (P' \otimes P''))$. In order to do that it is enough to show that $P \circ_{rk} (P' \cup P'') = (P \circ_{rk} P') \cup P''$. By hypothesis $(P_{\rho(P')} \cup P') \cup P''$ is consistent. Thus $\rho(P' \cup P'') \leq \rho(P')$ and then $\rho(P' \cup P'') = \rho(P')$. From this we conclude easily.

In order to show that **SR5** does not hold it is enough to consider the following example: the program P is defined by $P = \{b \to w, w \to w', w' \to f, o \to b, o \to \neg f\}$. The base is in this case $\langle P_0, P_1, P_2 \rangle$ with $P_0 = P$, $P_1 = \{o \to b, o \to \neg f\}$ and $P_2 = \emptyset$. Put $P' = \{o\}$ and $P'' = \{w'\}$. It is not hard to establish that $h(P') = P_1$ and $h(P' \cup P'') = P_1 \cup \{b \to w, w \to w'\}$. Thus $C_{fc}((P \circ_h P') \cup P'') = \{o, w', b, \neg f\}$ and $C_{fc}(P \circ_h (P' \cup P'')) = \{o, w', b, \neg f, w\}$. Therefore $C_{fc}(P \circ_h (P' \cup P'')) \nsubseteq C_{fc}((P \circ_h P') \cup P'')$, that is **SR5** does not hold.

To prove that **SR6** does not hold we consider the following example: Put $P = \{r_0, r_1, r_2\}$ where $r_0 = a \to c$, $r_1 = e \to \neg c$, $r_2 = b \to \neg c$. Put $P' = \{a, e\}$ and $P'' = \{b\}$. The base for P is $\langle P, \emptyset \rangle$. Then it is easy to see that $P_{\rho(P')} = P_{\rho(P' \cup P'')} = \emptyset$ and

$$\begin{array}{ll} I_P(P') & = \{\{r_0, r_2\}, \{r_1, r_2\}\} \quad \text{and} \\ I_P(P' \cup P'') & = \{\{r_0\}, \{r_1, r_2\}\} \end{array}$$

Thus $h(P') = \{r_2\}$ and $h(P' \cup P'') = \emptyset$. Therefore $\neg c \in C_{fc}((P \circ_h P') \cup P'')$ and $\neg c \notin C_{fc}(P \circ_h (P' \cup P''))$, so **R6** fails. □

Now, in order to analyse the postulates of syntactical revision satisfied by $\circ_{eh}$, we need to state in a very precise way what are the sets and functions of definition 1. This is the subject of the next definition.

Definition 10. *We put $\mathcal{L} = Lit$; Ω is the set of flocks of programs; $\Gamma = Prog$; notice that with the above identification we have $\Gamma \subseteq \Omega$; $\circ_{eh} : \Omega \times \Gamma \longrightarrow \Omega$ as defined above; the function $\otimes : \Omega \times \Gamma \longrightarrow \Omega$ is defined by:*

$$\langle Q_1, \ldots, Q_n \rangle \otimes P = \langle Q_1 \cup P, \ldots, Q_n \cup P \rangle$$

Notice that the restriction of $\otimes$ to couples of elements in Γ takes its values in Γ; and finally we define $\mathcal{C} : \Omega \longrightarrow \mathcal{P}(\mathcal{L})$ by putting $\mathcal{C} = C_{fc}$.

The extended hull revision operator satisfies some of our syntactical postulates. More precisely we have the following theorem:

Theorem 4. *The operator $\circ_{eh}$ satisfy **SR1**, **SR3** and **SR5**, when we take the definitions of 10. It satisfies a weak version of **SR2**: if P is consistent with all the elements of $\mathcal{F}$ then $\mathcal{F} \circ_{eh} P = \mathcal{F} \otimes P$.*

Proof: **SR1** is proved in an analogous way than the same postulate (**SU1**) for the operator $\diamond_P$ (see the proof of theorem 2).

SR3 and the weak form of **SR2** are straightforward from definitions.

Now we prove **SR5**. First, we consider the case $\mathcal{F} = P$. Then we want to prove that $\mathcal{C}(P \circ_{eh} (P' \otimes P'')) \subseteq \mathcal{C}((P \circ_{eh} P') \otimes P'')$. Suppose that

$$P \circ_{eh} P' = \langle Q_1 \cup P', \ldots, Q_n \cup P' \rangle$$
$$P \circ_{eh} (P' \otimes P'') = P \circ_{eh} (P' \cup P'') = \langle H_1 \cup P' \cup P'', \ldots, H_k \cup P' \cup P'' \rangle$$

Then $(P \circ_{eh} P') \otimes P'' = \langle Q_1 \cup P' \cup P'', \ldots, Q_n \cup P' \cup P'' \rangle$.
If $C_{fc}((P \circ_{eh} P') \otimes P'') = Lit$ we are done. Otherwise there is a Q_i such that $Q_i \cup P' \cup P''$ is consistent. But since Q_i is a subset of P maxiconsistent with P' and containing $P_{\rho(P')}$ necessarily $\rho(P') = \rho(P' \cup P'')$. Now we claim that for all $i = 1, \ldots, n$ either $Q_i \cup P' \cup P''$ is inconsistent or there is a $j \leq k$ such that $Q_i = H_j$. To see that suppose that $Q_i \cup P' \cup P''$ is consistent then Q_i is a subset maximal consistent with $P' \cup P''$ containing $P_{\rho(P')} = P_{\rho(P' \cup P'')}$, *i.e.* $Q_i = H_j$ for a j, by definition of sets H_q for $q = 1, \ldots, k$. Finally from the claim we get easily $\bigcap_{j=1}^{k} C_{fc}(H_j \cup P' \cup P'') \subseteq \bigcap_{i=1}^{n} C_{fc}(Q_i \cup P' \cup P'')$, *i.e.* $\mathcal{C}(P \circ_{eh} (P' \otimes P'')) \subseteq \mathcal{C}((P \circ_{eh} P') \otimes P'')$.

The general case, when $\mathcal{F} = \langle P_1, \ldots, P_n \rangle$, follows from the first case by definition of $\circ_{eh}$ and using the same trick that we used in the proof of **SU5** in theorem 2. □

Observation 5 *The postulates **SR2**, **SU2**, **SU6** and **SR6** don't hold for* $\circ_{eh}$.

Proof: We build counterexamples for each of those postulates.

For **SR2**: Take $\mathcal{F} = \langle \{a\}, \{\neg b\} \rangle$ and $P = \{b\}$. Then

$$C_{fc}(\mathcal{F} \otimes P) = C_{fc}(\langle \{a, b\}, \{\neg b, b\} \rangle) = \{a, b\}$$

So $\mathcal{F} \otimes P$ is consistent. But $C_{fc}(\mathcal{F} \circ_{eh} P) = C_{fc}(\langle \{a, b\}, \{b\} \rangle) = \{b\}$; so **SR2** fails.

For **SU2**: Take $P = \{a\}$ and $P' = \{a \to b\}$. Then $\emptyset = C_{fc}(P') \subseteq C_{fc}(P) = \{a\}$. But $C_{fc}(P \circ_{eh} P') = C_{fc}(P \cup P') = \{a, b\}$; so **SU2** fails.

For **SU6**: Take $P_1 = \{a \to b\}$, $P_2 = \{c \to d\}$ and $Q = \{a\}$. Clearly $C_{fc}(P_1) = C_{fc}(P_2) = \emptyset$, so the hypothesis of **SU6** is true. But

$$C_{fc}(Q \circ_{eh} P_1) = C_{fc}(Q \cup P_1) = \{a, b\} \neq \{a\} = C_{fc}(Q \cup P_2) = C_{fc}(Q \circ_{eh} P_2)$$

For **SR6**: The same counterexample given in the proof of theorem 3 to show that **SR6** fails for $\circ_h$, works in this case. □

Observation 6 *All the operators previously defined are syntax-sensitive,* i.e. *(IS) fails for the operators* $\diamond_P$, $\circ_{rk}$, $\circ_h$ *and* $\circ_{eh}$.

Proof: First we give a counterexample for $\diamond_P$. Put $P = \{a \to b\}$. Define $L_1 = \{a, b\}$ and $L_2 = \{a\}$. Put $L' = \{\neg a\}$. Clearly $C_{fc}^{P}(L_1) = C_{fc}^{P}(L_2) = \{a, b\}$. It is easy to see that $L_1 \diamond_P L' = \{b, \neg a\}$. Thus $C_{fc}^{P}(L_1 \diamond_P L') = \{b, \neg a\}$. But it is easy to see that $L_2 \diamond_P L' = \{\neg a\}$ so $C^P_{fc}(L_2 \diamond_P L') = \{\neg a\}$. Therefore $C^P_{fc}(L_1 \diamond_P L') \neq C^P_{fc}(L_2 \diamond_P L')$.

Now we give a counterexample for $\circ_{rk}$, $\circ_h$ and $\circ_{eh}$. Put $P_1 = \{a \to b\}$, $P_2 = \{a \to c\}$ and $P' = \{a\}$. Then $C_{fc}(P_1) = \emptyset = C_{fc}(P_2)$. But $C_{fc}(P_1 \circ P') = C_{fc}(P_1 \cup P') = \{a, b\} \neq \{a\} = C_{fc}(P_2 \cup P') = C_{fc}(L_2 \circ P')$
for any $\circ \in \{\circ_{rk}, \circ_h, \circ_{eh}\}$. □

4 Still another operator: selection hull

In the previous section we have seen that the two sceptical approaches to extend the ranked revision fail to be syntactical revision operators. In this section we give another extension of ranked revision based on the idea of using a selection function.

Let S be a function mapping sets of programs into programs, *i.e.*
$S : \mathcal{P}(Prog) \longrightarrow Prog$. We will say that S is a selection function iff the following properties hold:

(i) $S(\emptyset) = \emptyset$
(ii) If $D \neq \emptyset$ then $S(D) \in D$

Let us remark that this kind of selection functions are known as maxichoice functions in the literature [1, 12].

Let P and P' be two programs. Let $I_P(P')$ be as defined in section 2.2, *i.e.* the set of subsets of P which are maxiconsistent with P' and which contain $P_{\rho(P')}$. Let S be a selection function. We define the operator $\circ_{sh}$ by the following:

$$P \circ_{sh} P' = S(I_P(P')) \cup P'$$

We will take the same instantiations as in definition 9 in order to analyse the postulates satisfied by $\circ_{sh}$.

The following definition gives us a class of selection functions for which the operator $\circ_{sh}$ is a syntactical revision operator.

Definition 11. *A selection function S is said to be sensible iff the following property holds: for any programs P, P' and P''*

$$I_P(P') \cap I_P(P' \cup P'') \neq \emptyset \;\Rightarrow\; S(I_P(P')) = S(I_P(P' \cup P''))$$

Theorem 7. *If S is an sensible selection function then $\circ_{sh}$ is a syntactical revision operator when we consider $\mathcal{L}$, Ω, Γ, $\mathcal{C}$ and $\otimes$ as in definition 9.*

Proof: **SR1** and **SR3** follow easily from definition of $\circ_{sh}$. The verification of **SR3** is also easy using the property (ii) of selection function. Now we prove **SR5** and **SR6**. Actually with the hypothesis on S we are going to prove that if $(P \circ_{sh} P') \cup P''$ is consistent then $(P \circ_{sh} P') \cup P'' = P \circ_{sh} (P' \cup P'')$ which is obviously stronger than **SR5** and **SR6**.

Suppose that $I(P') = \{Q\} \cup D_1$ and $S(I(P')) = Q$. By hypothesis $Q \cup P' \cup P''$ is consistent so $\rho(P') = \rho(P' \cup P'')$ and $Q \in I(P' \cup P'')$. Therefore $I(P' \cup P'') =$

$\{Q\} \cup D_2$. Since $\rho(P') = \rho(P' \cup P'')$ and $P' \subseteq P' \cup P''$ we have that for any R in D_2 there exists R' in D_1 such that $R \subseteq R'$. Then by the property assumed for S we have $S(I(P' \cup P'')) = Q$; and from this we conclude. $\square$

We remark that the property required for S in definition 11 can be explained in an intuitive way: if you are choosing among elements of $\{Q\} \cup D_1$ and your preference is Q, it means that you think Q is the set which best fits P, then in the situation where you are choosing among elements of $\{Q\} \cup D_2$ with elements of D_2 contained in elements of D_1, you must reasonably choose Q.

Notice also that with this definition $\circ_{sh}$ is a conservative extension of $\circ_{eh}$, *i.e.* $C_{fc}(P \circ_{eh} P') \subseteq C_{fc}(P \circ_{sh} P')$. Thus we have

$$C_{fc}(P \circ_{rk} P') \subseteq C_{fc}(P \circ_{h} P') \subseteq C_{fc}(P \circ_{eh} P') \subseteq C_{fc}(P \circ_{sh} P')$$

A natural question one can ask is if there are selection functions with the property required in theorem 7. We show next a method for building such selection functions.

4.1 Building sensible selection functions

An easy way to build a sensible selection function is to use a linear ordering among propositions, that codes agent preferences in the program.

Let $|Q|$ be the cardinality of a set Q. Let $\{r_1, \ldots, r_n\}$ be an enumeration without repetition of rules and facts. Let π be a function (weighting) $\pi : \{r_1, \ldots, r_n\} \longrightarrow \omega$ such that $\pi(r_i) = 2^i$. We extend in a natural way π to $\mathcal{P}(Prog)$ by putting

$$\pi(\{r_{i_1}, \ldots, r_{i_k}\}) = \sum_{j=1}^{k} 2^{i_j}$$

Notice that if $P, P' \in Prog$ and $P \neq P'$ then $\pi(P) \neq \pi(P')$.

Let $<_\ell$ the lexicographical order on ω^2. Then define $S : \mathcal{P}(Prog) \longrightarrow Prog$ by $S(\emptyset) = \emptyset$ and if D is nonempty

$$S(D) = Q \text{ iff } Q \in D \text{ and } \forall R \in D \; (R \neq Q \;\Rightarrow\; (|R|, \pi(R)) <_\ell (|Q|, \pi(Q)) \,)$$

That is S chooses among the sets of greatest cardinality the set with higher weighting.

Because of definition of π it is quite easy to see that S is a sensible selection function.

Conclusion

We have proposed in this paper a methodological framework in order to analyse rational properties for syntactical change operators. We have also introduced five knowledge change operators based on forward chaining. The ideas behind the definition of these operators are very natural and simple. In most of the cases, they are connected with well known methods [9, 2, 21]. The originality of our work relies on the definition of the rank function for the revision operators and concerning the factual update on the fact that we consider the flocks with a logical content: their consequences by forward chaining. This point -endowing the result of an operator with a logical content- is particularly important because it makes possible the analysis of the operators in our abstract framework.

The following table summarizes the main results:

Operator	Satisfied postulates	Unsatisfied postulates
$\diamond_P$	**SU1+SU2+SU3+SU5+SU6+SU7+SU8**	
$\circ_{rk}$	**SR1+SR2+SR3+SR5+SR6**	
$\circ_h$	**SR1+SR2+SR3**	**SR5+SR6**
$\circ_{eh}$	**SR1+SR3+SR5**	**SR2+SU2+SR6+SU6**
$\circ_{sh}$	**SR1+SR2+SR3+SR5+SR6**	

Thus three of our operators have desirable properties. In particular factual update is a syntactical update operator according to our version of Katsuno and Mendelzon postulates. Ranked revision and selection hull are, syntactical revision operators according to our version of Alchourrón-Mendelzon-Gärdenfors postulates. The other operators, hull and extended hull revision, don't have good rational properties. Nevertheless they extend ranked revision in order to keep more information from the old program and thus seem to be less drastic than ranked revision.

These operators based on forward chaining are easily computable. Notice that, in particular, the operator of ranked revision is polynomial. The factual update operator, from a complexity point of view, is exponential (actually it is NP-complete since it requires the computation of all minimal hitting sets).

The operators based on hull revision are more complicated but can be computed with the help of the two others.

An interesting further work is to investigate the properties of our operators with respect to some relativisation of iteration postulates [7, 18]. Another interesting point to develop is the extension of these operators to the first order. By the way, our results can be translated in an obvious way to Datalog without negation by failure.

References

1. C.E. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change: partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50:510–530, 1985.
2. C. Baral, S. Kraus, J. Minker and V.S. Subrahmanian. Combining knowledge bases consisting of first order theories. *Computational Intelligence*, 80:45–71, 1992.
3. H. Bezzazi and R. Pino Pérez. Rational transitivity and its models. In *Proc. of the Twenty-Sixth International Symposium on Multiple-Valued Logic*, Santiago de Compostela, Spain, May, 1996, pp. 160–165, IEEE Computer Society Press.

4. H. Bezzazi, D. Makinson, and R. Pino Pérez. Beyond rational monotony: some strong non-horn rules for nonmonotonic inference relations. *Journal of Logic and Computation*, 7:605–631, 1997.
5. M. Dalal. Investigations into a theory of knowledge base revision: Preliminary report. In *Proc. of the Seventh National Conference on Artificial Intelligence (AAAI'88)*, pp. 475–479, 1988.
6. C. Damasio, W. Nedjdl, and L. M. Pereira. Revise: An extended logic programming system for revising knowledge bases. In *Proc. of the 4th International Conference on Principles of Knowledge Representation and Reasoning*, pp. 607-618, Morgan Kaufmann, 1994.
7. A. Darwiche, and J. Pearl. On the logic of iterated belief revision. *Artificial Intelligence*, 89:1–29, 1997.
8. T. Eiter and G. Gottlob. On the complexity of propositional knowledge base revision, updates and counterfactuals. In *Proc. of 11th Symposium on Principles of Database Systems*, pp. 261–273, ACM Press, 1992.
9. R. Fagin, G. Kuper, J.D. Ullman, and M.Y. Vardi. Updating logical databases. *Advances in Computing Research*, 3:1–18, 1986.
10. M. Freund and D. Lehmann. Belief revision and rational inference. Technical Report 94-16, Institute of Computer Science, The Hebrew University of Jerusalem. 1994.
11. A. Fuhrmann. Theory contraction through base contraction. *Journal of Philosophical Logic*, 20:175–203, 1991.
12. P. Gärdenfors. *Knowledge in Flux: modeling the dynamics of epistemic states.* MIT press, Cambridge, MA, 1988.
13. P. Gärdenfors and D. Makinson. Relations between the logic of theory change and nonmonotonic logic. In *The Logic of Theory Change, Workshop, Konstanz, FRG, Octuber 1989*, pages 185–205. Springer-Verlag, 1989. Lecture Notes in Artificial Intelligence 465.
14. S.O. Hansson. Theory contraction and base contraction unified. *Journal of Symbolic Logic*, 58:602–625, 1993.
15. S.O. Hansson. Reversing the Levi identity. *Journal of Philosophical Logic*, 22:637–669, 1993.
16. H. Katsuno and A.O. Mendelzon. Propositional knowledge base revision and minimal change. *Artificial Intelligence*, 52:263–294, 1991.
17. H. Katsuno and A.O. Mendelzon. On the difference between updating a knowledge database and revising it. In *Belief Revision*, P. Gärdenfors Ed. Cambrigde tracts in theoretical computer science 29. Cambridge University Press, 1992.
18. D. Lehmann. Belief revision, revised. In *Proceedings IJCAI'95*, 1995, pages 1534–1540.
19. D. Lehmann and M. Magidor. What does a conditional knowledge base entail? *Artificial Intelligence*, 55:1–60, 1992.
20. Paolo Liberatore. The complexity of iterated belief revision. In *Proceedings of the Sixth International Conference on Database Theory -ICDT'97*, Delphi, Greece, January 8-10, 1997. Lecture Notes in Computer Science, Vol. 1186, Springer, 1997, pages 276–290.
21. V. Marek and M.Truszczynski. Revision programming, Database Updates and Integrity Constraints. In *Proc. of 5th International Conference of Database Theory*, Prague, Czech Republic, January 11-13, 1995. Lecture Notes in Computer Science, Vol. 893, Springer, 1995, pages 368–382.
22. A.C. Nayak. Foundational Belief Change. *Journal of Philosophical Logic*, 23:495–533, 1994.

23. B. Nebel. Syntax-Based Approches to Belief Revision. In *Belief Revision*, P. Gärdenfors Ed. Cambrigde tracts in theoretical computer science 29. Cambridge University Press, 1992, pages 52–88
24. R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987.
25. K. Satoh. Nonmonotonic reasoning by minimal belief revision. In *Proceedings International Conference on Fifth Generation Computer Systems*, Tokio, 1988, pages 455–462.
26. N. Tennant. Changing the Theory of Theory Change: Towards a Computational Approach. *British Journal for Philosophy of Science* , 45:865–897, 1994.
27. A. Weber. Updating propositional formulas. In *Proceedings First Conference on Database Systems*, 1986, pages 487–500.
28. M. Winslett. Reasoning about action using a possible models approach. In *Proceedings AAAI'88*, 1988, pages 89–93.

Appendix: Update algorithm

Let P be a fixed program which in this context can be seen as our background theory or our integrity constraints. Let L be a set of facts which can be seen as our beliefs about the world. We would like to define the change produced by a set of facts L' coding a new piece of information about the world. The following definition describes the result of this change:

$$L \diamond_P L' = \begin{cases} Lit & \text{if } L \text{ or } L' \text{ is not } P\text{-consistent} \\ \langle L_1 \cup L', \ldots, L_n \cup L' \rangle & \text{otherwise} \end{cases}$$

where $\{L_1, \ldots, L_n\}$ is the set of subsets of L which are maximal and $P \cup L'$-consistent.

This is computed in two steps: first, we compute sets of facts that lead to inconsistency, called *contradictory sets.* Then, given the set SC of contradictory sets, we compute the minimal hitting sets of SC. The maximal subsets of L such that $L \cup L'$ is P-consistent are the sets $L \setminus H$, where H is a minimal hitting set of SC.

First step. A contradictory set C is a subset of L corresponding to a way of proving a pair of opposite literals from $P \cup L \cup L'$: C is a contradictory set if C is a subset of L and there exists a minimal subset P' of P such that $P' \cup C \cup L'$ is not consistent and, for each l in C, l appears in the body of a rule of P'.

We assume that, for every atom a that appears in the knowledge base, we have an implicit rule $a, \neg a \rightarrow \perp$.

To compute the contradictory sets when updating L with L', we build a contradiction tree $T_{L,L'}$, starting from $\perp$: a node is a pair$(L; C)$, where L is a list of literals to prove to obtain contradiction and C is a partial contradictory set. We start with the node $(\perp; \{\})$. Let $N = (l_1, l_2, \ldots, l_n; C)$ be a node of $T_{L,L'}$. The successors of N are computed as follows:

- if $l_1 \in L'$ or $l_1 \in P$ or if l_1 is already in C, then $(l_2, \ldots, l_n; C)$ is the only successor of N

- else for each rule $g_1, g_2, \dots, g_p \to l_1$, $(g_1, g_2, \dots, g_p, l_2, \dots, l_n\ ;\ C$ is a successor of N and if $l_1 \in L$, then $(l_2, \dots, l_n ; C \cup \{l_1\}))$ is a successor of N.

A branch terminates with an empty list of literals or with a node that cannot be developed. If a branch ends with $(\emptyset; C)$, then C is a contradictory set of facts. Note that if we suppose that L' is consistent with P, we can't obtain $(\emptyset; \emptyset)$. $T_{L,L'}$ doesn't give only the minimal contradictory sets, but all the ways to entail a pair of opposite literals from P and $L \cup L'$.

Example. We consider the program P and the set of literals L, with $P = \{a, b \to c\ ;\ a, d \to c\}$ and $L = \{a, b, d\}$. When updating L with $\{\neg c\}$, we obtain two contradictory sets $\{a, d\}$ and $\{a, b\}$. Fig 1 shows the contradiction tree (to simplify, we consider only the rule $\neg c, c \to \perp$ at the first step, since the only pair of contradictory literals that actually appears in this case is $c, \neg c$).

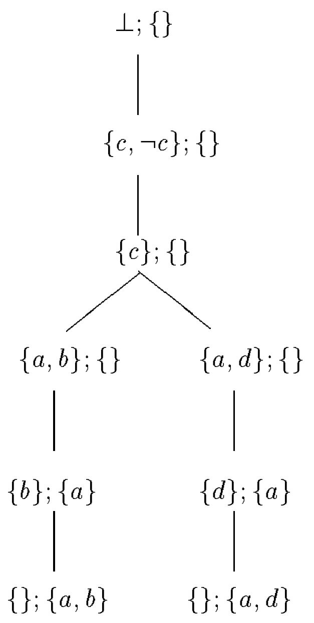

Fig 1. Contradiction tree

Second step. The contradiction tree produces a set of contradictory sets $SC = \{C_1, ..., C_n\}$. To update L with L' we compute all the maximal subsets S of L such that $S \cup L'$ is P-consistent. The subsets S of L are obtained by removing from L at least one element of each contradictory set: if H is a set of facts such that, for each element C_i of SC, $S \cap C_i \neq \emptyset$, then $(L \setminus H) \cup L'$ is consistent. H is usually called a hitting set of SC. To find the maximal consistent subsets of L, we need all the minimal hitting sets (by set inclusion) of SC.

The figure 2 illustrates the algorithm we implemented in Prolog to compute the minimal hitting sets.

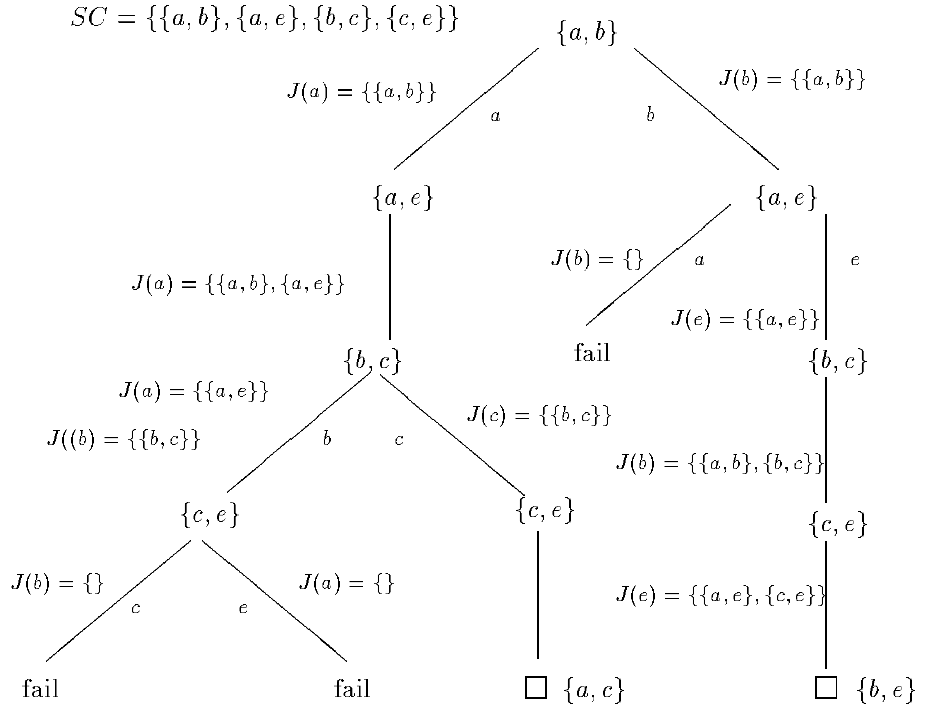

Fig 2. Minimal Hitting sets: $\{a,c\}$ and $\{b,e\}$

This algorithm is very close to the one given by Reiter in [24]. Let $SC = \{C_1, C_2, ..., C_n\}$. We try to construct a hitting set of SC by examining the elements of SC one by one: if the current set C_i is not already hit by the partial hitting set HS, we add one of the literals of C_i in HS. To know if a hitting set is minimal, we maintain a set of justifications $J(l)$ for each literal l of the hitting set: $J(l)$ contains all the sets of literals that are hit only by l. When a new literal l must be added to the hitting set, the justification sets are updated by removing all the sets containing l. If one of the justification sets becomes empty, the current hitting set is not minimal anymore and so the construction fails.

Concerning the relationships between our algorithm and Reiter's notice that we construct the same kind of HS-tree, where nodes are labeled with elements of SC and edges are labeled with elements of the hitting sets. The main difference is the use of justification sets instead of tree pruning. Tree pruning is used in Reiter's algorithm in order to compute not all the hitting sets but only the minimal ones. In our algorithm, this is done with justification sets and each minimal hitting set is computed in a unique branch.