The PMA Revisited

Andreas Herzig
IRIT, Université Paul Sabatier
118 route de Narbonne, F-31062 Toulouse Cédex, France
herzig@irit.fr
http://www.irit.fr/ACTIVITES/EQ_ALG/Herzig/home.html
Fax: (+33) 6155.8325

Abstract

The so-called Possible Models Approach (PMA) is a minimal change semantics for database updates. In this paper we give a sound and complete decision procedure for the PMA which is based on normal forming. We prove complexity results in particular for the case of updates restricted to conjunctions or disjunctions of literals. Finally, we sketch how integrity constraints can be taken into account.

1 INTRODUCTION

The Possible Models Approach (PMA for short) of Winslett (1988, 1990, 1995) is one of the most popular and successful semantical approaches in the domain of database updates. From the formal point of view, the PMA has a beautiful mathematical structure, with a notion of minimal change based on a simple closeness criterion between interpretations. Nevertheless, it has been criticized as being inadequate in particular in applications where integrity constraints appear.

The aim of this paper is twofold: first, we show a new way to correctly handle integrity constraints. Second we give a decision procedure for the PMA. Its core is a set of rewrite rules that allows us to transform formulas to formulas of classical logic. In this way theorems can be proved in a two-step procedure: first put the formula in normal form, and then apply a decision procedure for classical logic.

The rest of the paper is organized as follows: in section 2 we review the PMA, and in section 3 we reformulate it in conditional logic. In section 4 we present the decision procedure and give examples. Then we prove termination (section 5) and correctness (section 6) of the algorithm, and give complexity results (section 7). Finally, in section 8 we sketch how integrity constraints can be integrated into our presentation of the PMA.

2 THE PMA

2.1 The Original Semantics

We present the PMA semantics as described in (Winslett 1988, 1990, 1995).

We suppose a standard language of classical logic built from a set of atoms $ATM = \{p, q, r, \ldots\}$ with the classical connectives $\wedge, \vee, \neg, \top, \bot$. Formulas are denoted by $A, B, C, \ldots$. We consider $A \rightarrow B$ and $A \leftrightarrow B$ to be abreviations of $\neg A \vee B$ and $(A \rightarrow B) \wedge (B \rightarrow A)$.

We view interpretations as subsets of the set of atoms ATM. Given a formula A, $|A| = \{w \subseteq ATM : w \models A\}$ is the set of interpretations satisfying A.

Definition 1 *Given two interpretations $w, v \subseteq ATM$, the distance between w and v is the symmetric difference*

$$w \dot{-} v = (w - v) \cup (v - w)$$

In other words, $w \dot{-} v$ is the set of atoms whose interpretation differs: $w \dot{-} v = \{p :$ either $p \in w$ and $p \notin v$, or $p \notin w$ and $p \in v\}$. E.g. for $w = \{p, q, r\}$ and $v = \{p, r, t\}$ we have $w \dot{-} v = \{q, t\}$.

The update of an interpretation w by a set of interpretations U is the set of those elements of U which are closest, i.e. whose distance to w is minimal.

Definition 2 *Let $w \subseteq ATM$ and $U \subseteq 2^{ATM}$.*
$w \star U =$

$$\{u \in U \text{ : there is no } u' \in U \text{ such that } w \dot{-} u' \subset w \dot{-} u\}$$

E.g. $w \star \emptyset = \emptyset$, i.e. updating with the empty set (which is the interpretation of a contradiction) results in the empty set, and $\emptyset \star \{\{p\}, \{q\}, \{p, q\}\} = \{\{p\}, \{q\}\}$.

Slightly abusing notation we define the update of a set of interpretations.

Definition 3 *Let* $U, V \subseteq 2^{ATM}$.

$$U \star V = \bigcup_{w \in U} w \star V$$

In the PMA, the update $KB \star A$ of a database KB by a formula A is identified with the set of interpretations $|KB| \star |A|$.

Now under the standard notion of semantical consequence we have

$$KB \star A \models C \text{ iff } |KB| \star |A| \subseteq |C|$$

2.2 Two Criticisms

The first criticism concerns the interpretation of disjunction in the PMA, and the second one the implementation of integrity constraints.

Suppose we are given the database $KB = \{\neg p, \neg q\}$. Its update by $p \vee q$ is

$$|\{\neg p, \neg q\}| \star |p \vee q| = \{\emptyset\} \star \{\{p\}, \{q\}, \{p, q\}\}$$
$$= \{\{p\}, \{q\}\} = |\{p \vee q, \neg p \vee \neg q\}|$$

Thus the result of updating KB by the inclusive or is the exclusive or. There are several counterexamples where such an exclusive interpretation of the inclusive or is inadequate, and the PMA has been criticized for that (Eiter et al. 1992).

Second, suppose we are given a finite set of integrity constraints IC (that is confused with the conjunction of its elements). We must define an update operation $\star_{IC}$ which takes into account such integrity constraints. It has often been proposed (see e.g. (Katsuno and Mendelzon 1991)) to define such an operation by $KB \star_{IC} A = KB \star (A \wedge IC)$. This amounts to selecting the closest $A \wedge IC$-interpretations.

By means of the following counterexample it has been shown by Ginsberg (see e.g. (Lifschitz 1990)) that such a proposal is problematic in particular if $\star$ is the PMA update operator: let up_1 mean 'switch 1 is up', up_2 'switch 2 is up', and l 'the light is on'. Suppose there is a circuit such that the light is on exactly when both switches are in the same position. Hence

$$IC = (up_1 \leftrightarrow up_2) \leftrightarrow l$$

Let $KB = \{up_1, up_2, l\}$. Then one would expect that $KB \star (\neg up_1 \wedge IC) \models \neg l$. As well, we would expect $KB \star (\neg up_1 \wedge IC) \models up_2$, i.e. the second switch does not move. It turns out that neither is the case in the PMA.

We respond to these criticisms in section 8. In particular we shall propose a new, more sophisticated way of implementing integrity constraints.

2.3 The Existing Decision Procedures

Several decision procedures have been proposed for the PMA. The first one has been given in (Chou and Winslett 1991, 1994, Winslett 1995). Others can be found in (Del Val 1992) and (Grahne and Mendelzon 1995).

In the approaches of Chou and Winslett and of Grahne and Mendelzon, the algorithm directly manipulates models of the database KB. This amounts to enumerating the models of the database and updating each model.

As noted in (Del Val 1992), such procedures might be adequate in situations of almost complete information, in which there are only a few models. They clearly run into difficulties in cases where information is usually incomplete, as it is the case in many applications such as robotics and more generally commonsense reasoning. In such contexts, the number of models can be exponential w.r.t. the size of the database.

The approach of Del Val manipulates partial models of the database KB. His procedure enumerates the minimal models and updates each of them. Although he claims it to be a syntactical characterization of the PMA, it has a strong model-based flavour: in the basic mechanism, with the database and the update formula in disjunctive normal form, his algorithm uses a function *Diff(.,.)* computing the difference between two conjunctions of literals, which is basically the symmetric difference function $\dot{-}$ of the semantics.

All these methods are in terms of an update operator (involving a database and an update) and not in terms of a conditional operator (involving an update and a goal). This leads to procedures operating on models of the database, and makes their complexity exponential in the size of the database. Our procedure for the PMA will avoid at least that type of exponential complexity. The price to pay for that is exponential increase in the size of the update formula. This comes with putting the update in disjunctive normal form (which can be viewed as working with partial models of the update). Nevertheless, as databases tend to be big and updates to be small, our procedure should be preferrable in practical applications. As well, we shall give complexity results that are not in the above papers for the cases of updates by conjunctions or disjunctions of literals.

3 A REFORMULATION IN CONDITIONAL LOGIC

In this section we give an equivalent presentation of the PMA in terms of conditionals. This has been done before in (Fariñas and Herzig 1988) for the case where updates are restricted to literals, and in (Fariñas and Herzig 1994c, 1997) for the general case. In both papers, the PMA has been characterized axiomatically. In the more general setting of update operations satisfying the postulates of Katsuno and Mendelzon (1991) this has been done in (Grahne 1991, Eiter and Gottlob 1992, Ryan and Schobbens 1996, 1997).

The reason for such a presentation of the PMA is that our decision procedure introduces formulas with nested conditional operators (which cannot be expressed equivalently with update operators). As well, it becomes clear that the PMA is a particular case of Burgess's (1981) semantics for conditional logics.

The semantical apparatus is the same as before, based on the update $w \star U$ of an interpretation w by a set of interpretations U. But at the syntactical level, instead of re-using the update operator $\star$ we introduce a conditional operator $>$. The conditional $A > C$ is read as a hypothetical update: "if the current database is updated with A then C follows".

We have a standard language of conditional logic resulting from the addition of a conditional operator $>$ to the language of classical logic. In the formula $A > C$, A is called the *hypothesis* and C the *consequence*. Note that nested conditional operators as in $(p > q) > r$ are allowed. We call a formula *classical* if no conditional operator occurs in it. We stipulate that $\neg$ binds stronger than $>$, which in turn binds stronger than the rest of the connectives and associates to the right. E.g. $p > \neg q > r \vee t$ is $(p > ((\neg q) > r)) \vee t$.

We need a function *atm* which associates to every formula the set of atoms appearing in it.

Definition 4 *Let A be a formula.*

$$atm(A) = \{p \in ATM : p \text{ occurs in } A\}$$

E.g. $atm(\bot) = \emptyset$, and $atm(p \to (r \wedge q)) = \{p, q, r\}$. Now we inductively define the extension $|.|$ of a formula, which is a set of interpretations.

Definition 5 *Let A and C be formulas.*

$$|A > C| = \{w \subseteq ATM : w \star |A| \subseteq |C|\}$$

For the other connectives, $|.|$ is defined as in classical logic.

As usual, semantical consequence $KB \models C$ is defined by $|KB| \subseteq |C|$.

Remark In this way, the PMA is a particular case of Burgess's (1981) relational semantics, in the sense that its semantics corresponds to a unique "big" model. In that model, every classical interpretation is identified with some world. To see that, it is sufficient to prove that every relation $R(w) = \{(w_1, w_2) : w_1 \dot{-} w \subseteq w_2 \dot{-} w\}$ is a partial preorder, i.e. a reflexive and transitive relation. Note that the PMA-semantics is not an instance of Lewis's sphere models, because the preorder $R(w)$ is only partial and not total.

Finally we formally link the two presentations of the PMA by the so-called Ramsey Test (Gärdenfors 1988, Grahne 1991):

Proposition 1 *Let KB, A and C be classical. Then $KB \star A \models C$ iff $KB \models A > C$.*

Proof $(KB \star A) \models C$ means $|KB| \star |A| \subseteq |C|$. This is equivalent to $w \star |A| \subseteq |C|$ for every $w \in |KB|$. But this is nothing else than $w \in |A > C|$ for every $w \in |KB|$, which means $KB \models A > C$ in our conditional logic presentation.

4 THE DECISION PROCEDURE

In this section we show that every formula can be rewritten to a classical formula. This gives us the decision procedure.

First of all we recall some notions: as usual, a *literal* is an atom or the negation of an atom. Literals are denoted by L, L', L_1, ... Note that $\top$ and $\bot$ are not literals. *Clauses* are disjunctions of literals, and *cubes* are conjunctions of literals. E.g. $p \vee q \vee \neg p$ is a clause, and $p \wedge \neg r \wedge t$ is a cube.

A classical formula is in *negation normal form* (NNF for short) if it is constructed from literals, $\top$ and $\bot$ with conjunction and disjunction. In other words, for a formula in NNF, negations only govern atoms. A classical formula is in *disjunctive normal form* (DNF for short) if it is either $\top$ or $\bot$, or a non-tautological disjunction of consistent cubes (i.e. every cube does not contain both an atom and its negation). E.g. $p \wedge q$ and $p \vee q$ are in DNF, and neither $p \vee \neg p$ nor $p \wedge \bot$ nor $p \vee \bot$ nor $(p \wedge q \wedge \neg p) \vee r$ are in DNF.

Lemma 1 *For every classical formula C there is a classical formula C' such that $C \leftrightarrow C'$, $atm(C) = atm(C')$, and C' is in DNF.*

A key notion will be that of interference:

Definition 6 *A and B interfere iff $atm(A) \cap atm(B) \neq \emptyset$.*

Hence two formulas interfere if they have some atom in common. E.g. $p \vee q$ and $r \wedge (\neg q \vee t)$ interfere, and $p \vee q$ and $r \wedge t$ don't.

Next we state a useful property of classical logic saying that for every formula of classical logic, we can always 'factor out' some A which does not interfere with the rest of the formula.

Lemma 2 *Let C be a classical formula with $atm(C) \neq \emptyset$. Then there is a classical formula C' such that $C \leftrightarrow C'$, $atm(C) = atm(C')$, and*

- *C' is of the form $(A \wedge B) \vee (\neg A \wedge B') \vee B''$,*
- *A and B, A and B', and A and B'' do not interfere.*

Proof One can apply the following algorithm: First put the hypothesis in DNF. Whenever there is a cube containing both p and $\neg p$ as conjuncts, replace that cube by $\bot$. Choose some atom p occurring in the DNF. (There is at least one because $atm(C) \neq \emptyset$.) As by the definition of the DNF there is no cube with p occurring both positively and negatively in it, factoring p and $\neg p$ we can put the formula into the form

$$(p \wedge B) \vee (\neg p \wedge B') \vee B''$$

such that p does not occur in B, B' and B'' any longer (note that B, B', and B'' might be $\top$ or $\bot$). Hence p and B, p and B', and p and B'' do not interfere.

4.1 The Normal Forming Algorithm

Here comes the normal forming algorithm which rewrites every conditional with classical hypothesis and consequence into a classical formula. This algorithm will be iterated in the decision procedure.

$A, B, \ldots$ denote general formulas, $L, L_1, \ldots$ literals, and $p, q, \ldots$ atoms.

procedure eliminate-cond
input: a conditional $A > C$ such that A and C are classical
output: a formula G that is classical

1. put A in DNF and C in NNF
2. apply the following rewrite rules:

   a. **if** $n \geq 2$ **then**

   $$(L_1 \wedge \ldots \wedge L_n) > C' \rightsquigarrow L_1 > (\ldots > (L_n > C') \ldots)$$

   b. **if** $n \geq 2$ **then**
   $(L_1 \vee \ldots \vee L_n) > C' \rightsquigarrow$

   $$((L_1 \vee \ldots \vee L_n) \wedge C') \vee (L_1 > C' \wedge \ldots \wedge L_n > C')$$

   c. **if** card$(atm(A')) \geq 2$, A' is neither a clause nor a cube, and A' can be transformed by factoring [1] into $(p \wedge B) \vee (\neg p \wedge B') \vee B''$ with p and B, p and B', and p and B'' not interfering and $atm(A') = atm((p \wedge B) \vee (\neg p \wedge B') \vee B'')$, **then**

   $A' > C' \rightsquigarrow$

   $$(p \wedge (B \vee B'') > C' \wedge (B \vee B' \vee B'') > (B \vee B'' \vee \neg p > C')) \vee$$

   $$(\neg p \wedge (B' \vee B'') > C' \wedge (B \vee B' \vee B'') > (B' \vee B'' \vee p > C'))$$

   d. **if** card$(atm(A')) \geq 2$, A' is neither a clause nor a cube, and A' can be transformed by factoring into $(L \wedge B) \vee B'$ with L and B, and L and B' not interfering and $atm(A') = atm((L \wedge B) \vee B')$ **then**

   $A' > C' \rightsquigarrow$

   $$(L \wedge (B \vee B') > C') \vee (\neg L \wedge B' > C' \wedge (B \vee B') > (B' \vee L > C'))$$

3. apply the following rewrite rules:

   a. $\top > C' \rightsquigarrow C'$
   b. $\bot > C' \rightsquigarrow \top$
   c. $L > (C_1 \wedge C_2) \rightsquigarrow L > C_1 \wedge L > C_2$
   d. $L > (C_1 \vee C_2) \rightsquigarrow L > C_1 \vee L > C_2$
   e. $L > L \rightsquigarrow \top$
   f. $p > \neg p \rightsquigarrow \bot$
   g. $\neg p > p \rightsquigarrow \bot$
   h. **if** $atm(L) \neq atm(L')$ **then** $L > L' \rightsquigarrow L'$
   i. $L > \top \rightsquigarrow \top$
   j. $L > \bot \rightsquigarrow \bot$
   k. $A' \wedge \top \rightsquigarrow A'$
   l. $\top \wedge A' \rightsquigarrow A'$
   m. $A' \wedge \bot \rightsquigarrow \bot$
   n. $\bot \wedge A' \rightsquigarrow A'$

[1] v. Lemma 2

Remarks In step 1, if $atm(A') = \emptyset$ then A' is replaced by either $\top$ or $\bot$, and if $atm(A') = \{p\}$ then A' is replaced by either p or $\neg p$. In step 2.a, the hypothesis is a single cube, and as we are in DNF, the cube is consistent. In step 2.b, the hypothesis is a clause, and as we are in DNF, the clause is non-tautological.

Iterating step 2 we eliminate all conditionals $A' > C'$ with classical hypothesis such that that A' is "complex", i.e. such that hypotheses are of the form $\top$, $\bot$, $L_1 \wedge \ldots \wedge L_n$, or $L_1 \vee \ldots \vee L_n$. This means that at the end of step 2 we have $atm(A') \leq 1$. Note that in step 2.c, as $atm(A') \geq 2$, factoring out some p is always possible by Lemma 2. Moreover, as $\mathrm{card}(atm(A')) \geq 2$, we must have $\mathrm{card}(atm(B) \cup atm(B') \cup atm(B'')) \geq 1$. (Note also that nested conditionals are introduced here.) Step 2.d is a simplification of step 2.c for the case where either B or B' is $\bot$.

Iterating step 3.a and 3.b we eliminate all conditionals $A' > C'$ with a classical hypothesis A' such that $\mathrm{card}(atm(A')) \leq 1$. Iterating steps 3.c - d we distribute the literal hypotheses until every conditional with a classical hypothesis is of the form $L > L'$, where L and L' are literals. Steps 3.k - n correspond to standard simplification rules of classical logic.

Iterating step 3.a - n we eliminate all conditionals $A' > C'$ with a classical hypothesis A' such that $\mathrm{card}(atm(A')) \leq 2$.

4.2 The Decision Procedure

Now it is easy to build a procedure from the normal forming algorithm which decides whether a given formula is valid in the PMA. The story goes as follows:

procedure decide
input: a formula F
output: { *valid*, *non-valid* }

1. **while** there is a conditional operator occurring in F **do**
   (a) choose some conditional $A > C$ with A and C classical
   (b) apply the normal forming algorithm to $A > C$ resulting in some G
   (c) in the formula F, replace $A > C$ by G
2. apply some decision procedure for classical logic to F
3. **if** F is classically valid **then** return *valid*, **else** return *non-valid*

Remark In practice we want to know whether a formula C follows from the update of a classical database KB by a classical formula A. In this case there is a slightly better procedure: first, rewrite the conditional $A > C$ into an equivalent classical formula G by the normal forming algorithm. Then use a decision procedure for classical logic to check whether $KB \models G$.

4.3 Examples

Example Let $A = p \rightarrow (q \wedge \neg p) > (q \wedge \neg p)$. First, the conditional $(q \wedge \neg p) > (q \wedge \neg p)$ is chosen and is rewritten in step 2.a of **eliminate-cond** to $q > \neg p > (q \wedge \neg p)$.

Then from $p \rightarrow q > \neg p > (q \wedge \neg p)$, the conditional $\neg p > (q \wedge \neg p)$ is chosen and rewritten to $\neg p > q \wedge \neg p > \neg p$ in step 3.c. In step 3.e, this is rewritten to $\neg p > q \wedge \top$ and then in step 3.h to $q \wedge \top$.

Now the resulting $p \rightarrow q > (q \wedge \top)$ is first put into DNF in step 1 and becomes $p \rightarrow q > q$, and then $q > q$ is chosen and rewritten to $\top$.

Finally, the classical formula resulting from the normal forming is $p \rightarrow \top$, i.e. $\top$, and the decision procedure returns *valid*.

Example Let $A = p \vee (\neg p \wedge q \wedge r)) > \neg q$. The hypothesis is already in DNF. In step 2.c we factor out the atom p and replace the hypothesis $((p \wedge \top) \vee (\neg p \wedge q \wedge r) \vee (\bot)) > \neg q$ by

$$(p \wedge (\top \vee \bot) > \neg q \wedge (\top \vee (q \wedge r) \vee \bot) > (\top \vee \bot \vee \neg p > \neg q)) \vee$$

$$(\neg p \wedge ((q \wedge r) \vee \bot) > \neg q \wedge (\top \vee (q \wedge r) \vee \bot) > ((q \wedge r) \vee \bot \vee p > \neg q))$$

After eliminating $\top$ and $\bot$ by classical equivalences, we iteratedly eliminate $\top$ and $\bot$ with steps 3.a and 3.b and get

$$(p \wedge \neg q) \vee (\neg p \wedge (q \wedge r) > \neg q \wedge ((q \wedge r) \vee p > \neg q))$$

Then we obtain $(p \wedge \neg q) \vee (\neg p \wedge (q \wedge r) > \neg q \wedge ((q \wedge r) \vee \neg q))$ by 2.h, and finally $(p \wedge \neg q) \vee (\neg p \wedge q > r > \neg q \wedge ((q \wedge r) \vee \neg q))$ by 3.a. Now steps 3.h and 3.f apply and eliminate the remaining conditional operators. We end up with the classical formula $(p \wedge \neg q) \vee (\neg p \wedge \bot \wedge ((q \wedge r) \vee \neg q))$, which is equivalent to $p \wedge \neg q$.

5 TERMINATION

Theorem 1 (Termination) *The decision procedure terminates for every input formula F.*

Proof If there is no conditional in F then we are done.

Else a conditional $A > C$ with A and C classical is chosen and the normal forming procedure **eliminate-cond** is applied to it.

In step 1, termination of the transformations in DNF and NNF is a property of classical logic. Note that the number of atoms remains the same.

In step 2, all the hypotheses of the conditionals coming from a rewrite step are less complex than the original hypothesis (contain strictly less atoms). Precisely, to each formula F we associate a sequence $\{c_i\}_{i \in \mathbb{N}}$ such that c_i is the number of conditionals $A' > C'$ in F with $\mathrm{card}(atm(A')) = i$. Under the lexigraphic ordering this parameter decreases by every rewrite rule of step 2. Note that due to Lemma 2, step 2 is applicable whenever there is a conditional $A' > C'$ with $\mathrm{card}(atm(A')) \geq 1$. Note also that factoring out does not increase the number of atoms.

In step 3, after the application of a rewrite rule the hypotheses have the same complexity as before and the consequences of the conditonals are less complex (in particular when $L > (B \wedge C)$ and $L > (B \vee C)$ are replaced). Precisely, to each formula F we associate a sequence $\{c_i\}_{i \in \mathbb{N}}$ such that c_i is the number of conditionals $A' > C'$ in F with i symbols in C'. Under the lexicographic ordering this parameter decreases by every rewrite rule of step 3.

6 CORRECTNESS

Correctness of both the normal forming algorithm and the decision procedure follows from the fact that every replacement in the normal forming algorithm corresponds to a valid equivalence of the PMA.

First we tackle the simple things.

6.1 Correctness of the Rewrite Steps 3.a – n, 2.a, 2.b

Lemma 3 *The following equivalences are valid in the PMA.*

1. $\top > C \leftrightarrow C$
2. $\bot > C \leftrightarrow \top$
3. $L > (B \wedge C). \leftrightarrow .L > B \wedge L > C$
4. $L > (B \vee C) \leftrightarrow L > B \vee L > C$
5. $L > L \leftrightarrow \top$
6. $L > \neg L \leftrightarrow \bot$
7. $\neg L > L \leftrightarrow \bot$
8. $L > L' \leftrightarrow L'$ *if* $atm(L) \neq atm(L')$
9. $A > \top \leftrightarrow \top$
10. $L > \bot \leftrightarrow \bot$
11. $(L_1 \wedge \ldots \wedge L_n) > C \leftrightarrow L_1 > (\ldots > (L_n > C) \ldots)$ *if* $L_1 \wedge \ldots \wedge L_n$ *is consistent*
12. $(L_1 \vee \ldots \vee L_n) > C \leftrightarrow$

$$((L_1 \vee \ldots \vee L_n) \wedge C) \vee (L_1 > C \wedge \ldots \wedge L_n > C)$$

if $L_1 \vee \ldots \vee L_n$ *is non-tautological*

Proof We only prove the validity of the last equivalence, the others are similar.

Let $w \subseteq ATM$ be any classical interpretation, and let $L_1 \vee \ldots \vee L_n$ be non-tautological, i.e. there are no L_i, L_j such that $L_i = \neg L_j$.

There are two cases:

i. $w \in |L_1 \vee \ldots \vee L_n|$.

Then $w \star |L_1 \vee \ldots \vee L_n| = \{w\}$.

Now suppose $w \in |(L_1 \vee \ldots \vee L_n) > C|$. By the truth condition, $w \in |C|$ as well. Hence $w \in |(L_1 \vee \ldots \vee L_n) \wedge C|$ and as well

$$w \in |((L_1 \vee \ldots \vee L_n) \wedge C) \vee (L_1 > C \wedge \ldots \wedge L_n > C)|$$

The other way round, suppose

$$w \in |((L_1 \vee \ldots \vee L_n) \wedge C) \vee (L_1 > C \wedge \ldots \wedge L_n > C)|$$

If $w \in |(L_1 \vee \ldots \vee L_n) \wedge C|$ then $w \in |C|$ and we are done. Else we must have $w \in |L_i > C|$ for every L_i. As $w \in |L_1 \vee \ldots \vee L_n|$, there must be some L_j such that $w \in |L_j|$. Then by the truth condition, $w \in |C|$ as well, and consequently $w \in |(L_1 \vee \ldots \vee L_n) > C|$.

ii. $w \notin |L_1 \vee \ldots \vee L_n|$.

Then as there are no L_i, L_j such that $L_i = \neg L_j$, it follows from the definition of a distance that $w \star |L_1 \vee \ldots \vee L_n| = w \star |L_1| \cup \ldots \cup w \star |L_n|$.

Suppose $w \star |L_1 \vee \ldots \vee L_n| \subseteq |C|$. This means that $w \star |L_i| \subseteq |C|$ for every L_i. Hence $w \in |L_1 > C \wedge \ldots \wedge L_n > C|$ and as well

$$w \in |((L_1 \vee \ldots \vee L_n) \wedge C) \vee (L_1 > C \wedge \ldots \wedge L_n > C)|$$

The other way round, suppose

$w \in |((L_1 \vee \ldots \vee L_n) \wedge C) \vee (L_1 > C \wedge \ldots \wedge L_n > C)|$

As $w \notin |L_1 \vee \ldots \vee L_n|$, we must have $w \in |L_i > C|$ for every L_i. The truth condition warrants that $w \star |L_i| \subseteq |C|$ for every L_i. As we have said above, we also have $w \star |L_1 \vee \ldots \vee L_n| \subseteq |C|$, which means $w \in |(L_1 \vee \ldots \vee L_n) > C|$.

6.2 Some Semantical Properties

Before establishing the correctness of step 2.c, we first prove several more involved properties of the PMA semantics. They concern the interaction between the notion of minimal change of the PMA and the classical connectives.

Proposition 2 *Let $v \in w \star |A|$.*

1. *$w \in |A|$ iff $w \dot{-} v = \emptyset$ iff $w \star |A| = \{w\}$.*
2. *$w \dot{-} v \subseteq atm(A)$.*
3. *If $w \notin |A|$ then $(w \dot{-} v) \cap atm(A) \neq \emptyset$.*
4. *For every B such that A and B do not interfere, $(w \dot{-} v) \cap atm(B) = \emptyset$.*

Proof
(1.) follows from the definition of distance.
To prove (2.) suppose the contrary: then the distance between w and v would not be minimal (contradicting that $v \notin w \star |A|$).
(3.) follows from (1), and (4.) follows from (2).

Proposition 3 *Suppose A and B do not interfere.*

1. *If $w \in |A|$ then $w \star |A \wedge B| = w \star |B|$.*
2. *If $w \notin |A|$ then $w \star |B| \subseteq w \star |(A \wedge B') \vee B|$.*
3. *$w \star |A \wedge B| = (w \star |A|) \star |B|$*

Proof

1. First, $w \star |A \wedge B| = w \star |A| \star |B|$ by (3) of Proposition 3. By (1) of Proposition 2, $w \star |A| = \{w\}$, hence $w \star |A| \star |B| = \{w\} \star |B| = w \star |B|$.
2. Suppose $v \in w \star |B|$. We prove that there is no $v' \in w \star |A \wedge B'|$ closer to w than v, establishing thus that $v \in w \star |(A \wedge B') \vee B|$. As $w \notin |A|$, we must have $(w \dot{-} v') \cap atm(A) \neq \emptyset$ by (3.) of Proposition 2. On the other hand, as A and B do not interfere, $(w \dot{-} v) \cap atm(A) = \emptyset$. Hence we can never have $w \dot{-} v' \subset w \dot{-} v$, and therefore $v \in w \star |(A \wedge B') \vee B|$.
3. This follows from (4.) of Proposition 2.

Proposition 4 *Suppose A and B do not interfere, and $w \in |A|$. Then*

$$w \star |(A \wedge B) \vee B'| = w \star |B \vee B'|$$

Proof $w \star |(A \wedge B) \vee B'|$ is the set of those elements of $w \star |A \wedge B| \cup w \star |B'|$ whose distance to w is minimal. (To see that, suppose $v \in w \star |B|$ and $v \notin w \star |(A \wedge B) \vee B'|$. The latter means that there is a $v' \in w \star |A \wedge B|$ such that $w \dot{-} v' \subset w \dot{-} v$.) As $w \in |A|$ and A and B do not interfere, by (1.) of Proposition 3 this set is equal to the minimal elements of $w \star |B| \cup w \star |B'|$. But the latter is nothing else than $w \star |B \vee B'|$.

Proposition 5 *Suppose A and B as well as A and B' do not interfere, and $w \notin |A|$. Then*

$$w \star |(A \wedge B) \vee B'| = (((w \star |B \vee B'|) - |B'|) \star |A|) \cup (w \star |B'|)$$

Proof From the right to the left: If $v \in w \star |B'|$, then as $w \notin |A|$ and A and B' do not interfere, $v \in w \star |(A \wedge B) \vee B'|$ follows from (6) of Proposition 3. Hence suppose $v \in (w \star |B \vee B'| - |B'|) \star |A|$. This means that there is some $v_B \in w \star |B \vee B'|$ such that $v \in v_B \star |A|$ and $v_B \notin |B'|$. Thus $v_B \in w \star |B|$, and there is no $v_{B'} \in v \star |B'|$ closer to w. In other words, for every $v_{B'}$ we have $w \dot{-} v_{B'} \subseteq atm(B')$, and there is some atom $p \in w \dot{-} v_{B'}$ such that $p \notin w \dot{-} v_B$. Hence (and this is the crucial step) as A and B' do not interfere, by (4) of Proposition 2 we have also that $p \notin w \dot{-} v$. This means $v \in w \star |B| \star |A|$, and moreover there is no $v_{B'} \in v \star |B'|$ such that $w \dot{-} v_{B'} \subset w \dot{-} v$. As A and B do not interfere, by (3) of Proposition 3 we have $v \in w \star |A \wedge B|$, and thus $v \in w \star |(A \wedge B) \vee B'|$ because there is no $v_{B'} \in v \star |B'|$ closer to w.

From the left to the right: Suppose $v \in w \star |(A \wedge B) \vee B'|$. Then either $v \in w \star |B'|$ and we are done, or $v \in w \star |A \wedge B|$, and there is no $v' \in w \star |B'|$ which is closer to w than v. In the latter case $v \in w \star |B| \star |A|$ because A and B do not interfere. This means that there is some $v_B \in w \star |B \vee B'|$ such that $v \in v_B \star |A|$ and $v_B \notin |B'|$. Hence $v \in (w \star |B \vee B'| - |B'|) \star |A|$.

6.3 Correctness of Rewrite Step 3.c

Proposition 6 *Suppose A and B, A and B' and A and B'' do not interfere. Then the following formulas are valid in the PMA.*

1. $A \to .((A \wedge B) \vee B') > C \leftrightarrow (B \vee B') > C$
2. $\neg A \to .((A \wedge B) \vee B') > C \leftrightarrow (B' > C \wedge (B \vee B') > (B' \vee (A > C)))$

3. $((A \wedge B) \vee (\neg A \wedge B') \vee B'') > C \leftrightarrow.$

$$(A \wedge (B \vee (\neg A \wedge B') \vee B'') > C) \vee (\neg A \wedge ((A \wedge B) \vee B' \vee B'') > C$$

4. $((A \wedge B) \vee B')) > C \leftrightarrow.$

$$(A \wedge (B \vee B') > C) \vee (\neg A \wedge B' > C \wedge (B \vee B') > (B' \vee (A > C)))$$

Proof
(1.) and (2.) are the syntactical counterpart of Proposition 4.
Let us prove (3.): First, $((A \wedge B) \vee (\neg A \wedge B') \vee B'') > C$ is equivalent to

$$((A \wedge B) \vee (\neg A \wedge B') \vee B'') > C \wedge A . \vee . ((A \wedge B) \vee (\neg A \wedge B') \vee B'') > C \wedge \neg A$$

Then the equivalence follows with (1.). To establish (4.), (1.) and (2.) are applied in the same manner as in the proof of (3.).

Lemma 4 *Suppose A and B, A and B', and A and B'' do not interfere. Then the equivalence*

$$((A \wedge B) \vee (\neg A \wedge B') \vee B'') > C \leftrightarrow$$

$$(A \wedge (B \vee B'') > C \wedge (B \vee B' \vee B'') > (B \vee B'' \vee \neg A > C)) \ \vee$$

$$(\neg A \wedge (B' \vee B'') > C \wedge (B \vee B' \vee B'') > (B' \vee B'' \vee A > C))$$

is valid in the PMA.

Proof Apply the equivalence (3.) of Proposition 6 and then twice (4.).

6.4 Correctness of the Decision Procedure

Theorem 2 (Correctness) *Let F be a formula. If F is valid in the PMA then the decision procedure returns* valid, *and else it returns* non-valid.

Proof First, by the Termination Theorem 1, the decision procedure terminates.

Now all the steps in the normal forming algorithm **eliminate-cond** replace formulas by equivalent ones: for step 1 this is ensured by Lemma 1. For the rewrite rules of step 2.a - n, the equivalence is warranted by Lemma 4 and Lemma 2. Steps 2.a and 2.b and step 3.a - d are covered by Lemma 3. Hence each time the normal forming algorithm is applied, it returns a classical formula that is equivalent to its input formula.

As the number of conditional operators decreases with each call of **eliminate-cond**, by iteration a classical formula is obtained that is equivalent to the original formula. Finally, the decision procedure for classical logic establishes the result.

7 COMPLEXITY

It has been proved by Eiter and Gottlob (1992) that the theoretical complexity of all the known update formalisms is strictly higher than that of classical logic. Precisely, they have shown that deciding in the PMA if a counterfactual $p > q$ is true over a database KB is Π_2^p-complete, and that Π_2^p-hardness holds even if $KB \subseteq ATM$ and q is an atom with $q \in KB$ (Eiter and Gottlob 1992, Theorem 6.4).

Clearly, our decision procedure is exponential in the size of the update formula. Nevertheless, it is only co-NP-complete in the size of the database (the size of the update formula p being less or equal some constant p). This is exactly the worst case complexity of the PMA given by Eiter and Gottlob (1992) in their Theorem 8.1 (see also their Table 3).

A considerably simpler normal form can be obtained when the hypotheses are restricted to clauses or conjunctions of literals. Thus we improve considerably the complexity bound for the general case as given in (Eiter and Gottlob 1992), answering at the same time an open question from the latter.

In the case of conjunctions of literals, normal forming can be done in linear time:

Theorem 3 *If every hypothesis of a formula F is a cube, then normal forming is in $O(n)$, where n is the number of symbols in F.*

The case of clauses is slightly more complex than that of cubes. Here normal forming can be done in quadratic time:

Theorem 4 *If every hypothesis of a formula F is a clause or a cube, then normal forming is in $O(n^2)$, where n is the number of symbols in F.*

As a corollary we get that in this case the complexity of the decision problem for the PMA is the same as that for classical logic.

Corollary 1 *For the class of formulas where every hypothesis is a clause or a cube, the validity problem is co-NP-complete.*

8 UPDATES UNDER INTEGRITY CONSTRAINTS

One story to learn from Lifschitz's counterexample of section 2.2 is that there are aspects of the domain structure which cannot be expressed by classical integrity constraints. In (Fariñas and Herzig 1994a, 1994b, 1996) (see also (Giunchiglia 1995)), we have argued for the use of information about dependences between formulas in change operations, and we have studied the formal properties of the appropriate dependence notion.

Dependence is a sort of weak causal connection. Our reading of 'q depends on p' is that updates concerning p *may* change the truth value of q. In the example, l depends on up_1, but up_2 does not depend on up_1: for the update of any database by up_1 or $\neg up_1$, the truth value of up_2 remains unchanged.

Formally, we suppose given a *dependence function* DEP mapping atoms to sets of atoms such that $p \in DEP(p)$ for all atoms p. We shall suppose in the sequel that $DEP(p)$ is *finite* for every p. E.g. $DEP(up_1) = \{l\}$, $DEP(up_2) = \{l\}$, and $DEP(l) = \{up_1, up_2\}$. Dependence can be extended to general formulas by stipulating $DEP(A) = \bigcup_{p \in atm(A)} DEP(p)$.

Now we must define an update operation $\star_{IC,DEP}$ such that for every update A:

- atoms that are independent of A 'survive': if $p \notin DEP(A)$ and $KB \models p$ then we want $KB \star_{IC,DEP} A \models p$, and
- atoms that depend on A are *a priori* retracted: if $p \in DEP(A)$ then we want $KB \star_{IC,DEP} A \not\models p$ and $KB \star_{IC,DEP} A \not\models \neg p$.

E.g. in our example we want $KB \star_{IC,DEP} \neg up_1 \models up_2$ because $up_2 \notin DEP(up_1)$. On the other hand, *a priori* neither l nor $\neg l$ will follow from $KB \star_{IC,DEP} \neg up_1$ because $l \in DEP(up_1)$. Now a way to obtain that l follows from the updated database is to simply add IC to it.

Therefore, we define an update operation $\star_{IC,DEP}$ under a set of integrity constraints and a dependence function DEP as follows:

$$KB \star_{IC,DEP} A = (\bigvee_{B \in CTX(A)} (KB \star B)) \wedge IC \wedge A$$

where $\star$ is the PMA update operation, and the function CTX is defined by

$$CTX(A) = \{L_1 \wedge \ldots \wedge L_n : L_i = p_i \text{ or } L_i = \neg p_i\}$$

if $DEP(A) = \{p_1, \ldots, p_n\}$. E.g. $CTX(up_1) = \{up_1 \wedge l, up_1 \wedge \neg l, \neg up_1 \wedge l, \neg up_1 \wedge \neg l\}$. Equivalently, we have

$$KB \star_{IC,DEP} A = \bigvee_{B \in CTX(A)} ((KB \star B) \wedge IC \wedge A)$$

We reformulate with conditionals and get

$KB \star_{IC,DEP} A \models C$

iff $\bigvee_{B \in CTX(A)} (KB \star B) \wedge IC \wedge A \models C$

iff $KB \star B \models (IC \wedge A) \rightarrow C$
for every $B \in CTX(A)$

iff $KB \rightarrow (B > ((IC \wedge A) \rightarrow C)$
for every $B \in CTX(A)$

In this way we have defined a new update operation, and a first thing to do is to relate it to the PMA: suppose $IC = \emptyset$ and $DEP(p) = \{p\}$ for all $p \in ATM$. It turns out that $KB \star_{IC,DEP} A = KB \star A$ for every A that is a cube. The behaviour is different if A is a clause: e.g. $\{\neg p, \neg q\} \star_{IC,DEP} (p \vee q) = \{\{p,q\}, \{p\}, \{q\}\}$, whereas we have seen in section 2.2 that $\{\neg p, \neg q\} \star (p \vee q) = \{\{p\}, \{q\}\}$. Note that (as we have already mentioned there) the PMA has been criticized for that exclusive interpretation of the inclusive or.

In our example, we have to check the validity of four formulas. We only give the case of $KB \rightarrow (up_1 \wedge l) > ((IC \wedge \neg up_1) \rightarrow \neg l)$.

First, the conditional $(up_1 \wedge l) > ((IC \wedge \neg up_1) \rightarrow \neg l)$ is rewritten in step 2.a of the normal forming algorithm to $up_1 > l > ((IC \wedge \neg up_1) \rightarrow \neg l)$. Then, the conditional $l > ((IC \wedge \neg up_1) \rightarrow \neg l)$ is rewritten applying several times steps 3.c and 3.d to $(l > ((up_1 \leftrightarrow up_2) \leftrightarrow l) \wedge l > \neg up_1) \rightarrow l > \neg l$. The first conditional $(l > ((up_1 \leftrightarrow up_2) \leftrightarrow l)$ is rewritten in several steps to $(up_1 \leftrightarrow up_2)$. [2] The second one $l > (\neg up_1)$ is rewritten to $\neg up_1$, and the third one $l > \neg l$ to $\perp$. Finally we obtain the classical formula $((up_1 \leftrightarrow up_2) \wedge \neg up_1) \rightarrow \perp$, which is equivalent to $up_1 \vee up_2$.

Now we still have to rewrite $l > (up_1 \vee up_2)$ to $up_1 \vee up_2$, and to check whether $KB \rightarrow (up_1 \vee up_2)$ is classically valid, which is the case.

Similarly, the three other updates are performed. It turns out that $KB \star_{IC,DEP} \models l$, i.e. our approach treats the counterexample correctly.

Generally, if card$(DEP(A)) = n$ we must do 2^n PMA-updates. Clearly, such a method is interesting only if knowledge is represented in a way such that there are very few dependences. On the other hand, a very advantageous feature of our approach is that updates are limited to conjunctions of literals. As we have seen in Theorem 3, normal forming can be done in linear time in this case. Therefore, if we suppose that there is

[2] Remember that we have kept $\leftrightarrow$ and $\rightarrow$ here for reasons of readability. In the algorithm, it is supposed that only $\wedge, \vee$ and $\neg$ appear in formulas.

some constant k such that card$(DEP(p)) \leq k$ for all $p \in ATM$, (which is the case in particular if our language is finite) then the complexity of deciding updates under integrity constraints and dependences becomes that of deciding the validity of a classical formula.

9 DISCUSSION

Although the theoretical complexity of the validity problem in the PMA is strictly higher than that for classical logic, in practical applications updates are often conjunctions or disjunctions of literals, for which we have given better complexity results.

In particular, we have shown how integrity constraints and dependences can be represented in our approach in a way such that the only updates involved are conjunctions of literals. We have pointed out that if moreover the dependences are bounded then the complexity of the decision problem for the PMA is the same as that for classical logic.

In a conditional language the updates are hypothetical and the database is not progressed (in Reiter's sense). It is an open problem how the updated database can be effectively constructed. Another open problem is whether our method can be extended to other approaches such as Dalal's (1988).

Acknowledgements

This paper continues work done with Luis Fariñas del Cerro that has been presented in (Fariñas and Herzig 1988, 1994c, 1997). Thanks to him as well as to Bernhard Nebel for several stimulating discussions.

References

J. P. Burgess (1981), Quick completeness proofs for some logics of conditionals. Notre Dame J. of Formal Logic 22 1, pp 76-84.

T.S. Chou and M. Winslett (1991), Immortal: A model-based belief revision system. Proc. Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'91), Morgan Kaufmann.

T.S. Chou and M. Winslett (1994), A model-based belief revision system. J. of Automated Reasoning 12, pp 157-208.

M. Dalal (1988), Investigations into a theory of knowledge base revision: preliminary report. Proc. 7th Nat. Conf. on Artificial Intelligence (AAAI'88).

A. Del Val (1992), Computing knowledge base updates. Proc. Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'92), Morgan Kaufmann.

Th. Eiter and G. Gottlob (1992), On the complexity of propositional knowledge base revision, updates, and counterfactuals. Journal of Artificial Intelligence 57, pp 227-270.

Th. Eiter, G. Gottlob, and Y. Gurevich (1993), Curb your theory! A circumscriptive approach for inclusive interpretation of disjunctive information. Proc. IJCAI'93.

L. Fariñas del Cerro and A. Herzig (1988), An automated modal logic for elementary changes. In: P Smets, A. Mamdani, D. Dubois and H. Prade (eds.), Non-Standard Logics for Automated Reasoning, Academic Press, pp 63-79.

L. Fariñas del Cerro and A. Herzig (1994a), Interference logic = conditional logic + frame axiom. Int. J. of Intelligent Systems, Special Issue on Revision and Updating in Knowledge Bases, 9(1), pp 119-130.

L. Fariñas del Cerro, A. Herzig (1994b), Possibility theory and independence. In: B. Bouchon-Meunier, R.R. Yager, L. A. Zadeh, eds., Advances in Intelligent Computing – IPMU-94. Springer Verlag, LNCS 945, pp 292-301.

L. Fariñas del Cerro et A. Herzig (1994c), A conditional logic for updating in the Possible Models Approach. Proc. 18th German Conf. on Artificial Intelligence (KI'94), eds. B. Nebel, L. Dreschler-Fischer, Springer Verlag, LNAI 861.

L. Fariñas and A. Herzig (1996), Belief change and dependence. Proc. 6th Conf. on Theoretical Aspects of rationality and knowledge (TARK VI), March 1996.

L. Fariñas and A. Herzig (1997), All other things being equal: on a notion of inertia in conditional logic", In: E. Ejerhed and S. Lindström (eds.), Logic, Action and Cognition - Selection of papers of the Umea colloquium on Dynamic approaches in Logic, Language and Information (UmLLI-93). KluwerAcademic Publishers, to appear. (http://www.irit.fr/ACTIVITES/EQ_ALG/Papers/umea.ps)

P. Gärdenfors (1988), Knowledge in Flux. MIT Press.

E. Giunchiglia and V. Lifschitz (1995), Dependent fluents. Proc. IJCAI'95, Montreal, Canada, pp 1964-1969.

G. Grahne (1991), Updates and counterfactuals. Proc.

Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'91), Morgan Kaufmann.

G. Grahne and A. O. Mendelzon (1995), Updates and subjunctive queries. Information and Computation 116, pp 241-252.

H. Katsuno and A.O. Mendelzon (1991), On the Difference between Updating a Knowledge Base and Revising it. In: P. Gärdenfors (ed.), Belief Revision. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, pp 183-203. Preliminary version in: Proc. Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'91), Morgan Kaufmann.

V. Lifschitz (1986), Frames in the space of situations. Journal of Artificial Intelligence 46, pp 365-376.

M. D. Ryan and P.-Y. Schobbens (1997), Intertranslating Counterfactuals and Updates. In: J. of Logic, Language and Information, special issue on TARK'96, to appear. (Preliminary version in: W. Wahlster (ed.), Proc. ECAI'96. John Wiley & Sons, Ltd., to appear.)

M. Winslett (1988), Reasoning about actions. Proc. 7th Nat. Conf. on Artificial Intelligence (AAAI'88).

M. Winslett (1990), Updating Logical Databases. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press.

M. Winslett (1995), Updating Logical Databases. In: D. Gabbay, A. Galton, J. A. Robinson (eds.), Handbook of Logic in Artificial Intelligence and Logic Programming, Vol. 4 "Epistemic and Temporal Reasoning", Oxford University Press, pp.133-174.