View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Scientific Publications of the University of Toulouse II Le Mirail

Propositional update operators based on formula/literal dependence

Andreas Herzig, Jerome Lang, Pierre Marquis

To cite this version:

Andreas Herzig, Jerome Lang, Pierre Marquis. Propositional update operators based on formula/literal dependence. ACM Transactions on Computational Logic, Association for Computing Machinery, 2013, vol. 14 (n 3), pp. 1-31. <10.1145/2499937.2499945>. <hal-01128955>

HAL Id: hal-01128955

https://hal.archives-ouvertes.fr/hal-01128955

Submitted on 10 Mar 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Open Archive TOULOUSE Archive Ouverte (OATAO)
OATAO is an open access repository that collects the work of Toulouse researchers and makes it freely available over the web where possible.

This is an author-deposited version published in : http://oatao.univ-toulouse.fr/
Eprints ID : 12416

To link to this article : DOI :10.1145/2499937.2499945
URL : http://dx.doi.org/10.1145/2499937.2499945

To cite this version : Herzig, Andreas and Lang, Jerome and Marquis, Pierre *Propositional update operators based on formula/literal dependence*. (2013) ACM Transactions on Computational Logic, vol. 14 (n° 3). pp. 1-31. ISSN 1529-3785

Any correspondance concerning this service should be sent to the repository administrator: staff-oatao@listes-diff.inp-toulouse.fr

Propositional Update Operators Based on Formula/Literal Dependence

ANDREAS HERZIG, IRIT-CNRS, Toulouse
JEROME LANG, LAMSADE-CNRS, Paris
PIERRE MARQUIS, CRIL-CNRS, Lens

We present and study a general family of belief update operators in a propositional setting. Its operators are based on formula/*literal* dependence, which is more fine-grained than the notion of formula/*variable* dependence that was proposed in the literature: formula/variable dependence is a particular case of formula/literal dependence. Our update operators are defined according to the "forget-then-conjoin" scheme: updating a belief base by an input formula consists in first forgetting in the base every literal on which the input formula has a negative influence, and then conjoining the resulting base with the input formula. The operators of our family differ by the underlying notion of formula/literal dependence, which may be defined syntactically or semantically, and which may or may not exploit further information like known persistent literals and pre-set dependencies. We argue that this allows to handle the frame problem and the ramification problem in a more appropriate way. We evaluate the update operators of our family w.r.t. two important dimensions: the logical dimension, by checking the status of the Katsuno-Mendelzon postulates for update, and the computational dimension, by identifying the complexity of a number of decision problems (including model checking, consistency and inference), both in the general case and in some restricted cases, as well as by studying compactability issues. It follows that several operators of our family are interesting alternatives to previous belief update operators.

General Terms: Algorithms, Languages, Theory

Additional Key Words and Phrases: Knowledge representation, update, computational complexity

1. INTRODUCTION

Belief update mainly focuses on determining how a belief state (typically represented by a belief base) should evolve in order to take into account a new piece of information reflecting an explicit evolution of the world, typically the effect of an agent's action, or an event the occurrence of which the agent is aware of. In other words, updating by some piece of information μ roughly corresponds to projecting the consequences of the action "make μ true."

Authors' addresses: A. Herzig, IRIT-CNRS/Université de Toulouse, 118 route de Narbonne - F-31062 Toulouse - France; J. Lang, LAMSADE-CNRS/Université Paris Dauphine, Place du Maréchal de Lattre de Tassigny - F-75775 Paris - France; P. Marquis, CRIL-CNRS/Université d'Artois, rue Jean Souvraz - F-62307 Lens - France; email: marquis@cril.univ-artois.fr.

Many belief update operators in a propositional setting have been proposed in the literature. In this setting, a belief base is a propositional formula (or a finite set of such formulas that is interpreted conjunctively). The first works were mostly based on principles of minimization of change. While this principle is attractive for many belief change operators, including belief revision[1], using it to define update operators has been criticized in many places (see e.g., Herzig and Rifi [1999]), and it is now more or less a common belief that an explicit minimization of change should not be the core principle for belief update. A more suitable principle for building update operators is the notion of *dependence*. The first dependence-based update operators were proposed by Winslett [1990], Doherty et al. [1998, 2000], Herzig [1996], Herzig and Rifi [1998, 1999]. In a nutshell, a belief base is updated by first forgetting every piece of belief whose variables depend on those of the input formula, and then conjoining ("expanding") the resulting base with the input formula itself. For example take the propositional belief base $a \wedge b$ and the input formula $\neg a$. If we consider that the propositional variable a depends on a and b does not, then forgetting a from $a \wedge b$ results in b, and expanding the latter with $\neg a$ results in $\neg a \wedge b$. Another example is the belief base $a \wedge b$ and the input formula $a \vee b$. If the propositional variable a depends on a and b depends on b then forgetting $a \vee b$ from $a \wedge b$ results in the empty belief base $\top$, and expanding the latter with $a \vee b$ results in $a \vee b$. Depending on the notion of dependence under consideration, many operators following such a "forget-then-conjoin" scheme can be defined.

However, some severe problems with these dependence-based operators were pointed out by Herzig et al. [2001]. They stem from the fact that dependence is not directed: as we shall explain below, in many cases this makes us forget too much. In this article we show how dependence-based update operators can be generalized to take into account the *truth value* of fluents (here, the propositional variables) on which the update formula depends (and not only the variables themselves). This leads to a better handling of the frame problem and is specifically important when persistent literals are considered: literals that remain true once they become true. The idea is that in order to update by μ one should forget the negations of all those literals μ depends on. Let us take up the second above example where forgetting the variables a and b in $a \wedge b$ resulted in the trivial belief base $\top$. It is natural to consider that a (positively) depends on the truth of a, but that $\neg a$ does not (positively) depend on the truth of a. Therefore in order to update by $a \vee b$ we should forget $\neg a$ and $\neg b$ from $a \wedge b$. As both a and b "survive" the forgetting of $\neg a$ and of $\neg b$, updating $a \wedge b$ by $a \vee b$ should result in $a \wedge b$.

In order to overcome these problems we introduce a family of update operators that is based on formula/literal dependence. We explore several possible definitions of dependence functions: they may be defined syntactically or semantically, and they may exploit further information such as knowledge about persistent literals and pre-set dependencies. Different notions of formula/literal dependence will lead to different update operators. We show that these belief update operators allow to grasp the effects of actions in a way better than the previous dependence-based operators, while avoiding a complexity shift. We also evaluate the operators with respect to their logical and computational properties: as to the former, we check the status of the Katsuno-Mendelzon (KM) postulates for update; as to the latter, we identify the complexity of a number of decision problems (including model checking, consistency, and inference), as well as compactability issues. We identify some operators from the family that provide interesting trade-offs, compared to previous belief update operators.

[1]Note that however some authors question the suitability of change minimization even for belief revision; see for instance [Delgrande et al. 2005].

The rest of the article is as follows. Some formal preliminaries are given in Section 2. Our family of dependence-based update operators is presented in Section 3. Logical properties are discussed in Section 4 and computational properties are studied in Section 5. Dependence-based update is positioned with respect to action languages in Section 6. Connections to related work are given in Section 7. Finally, Section 8 concludes the article.

2. FORMAL PRELIMINARIES

In this section, we recall the necessary background definitions from propositional logic and complexity theory.

2.1. Propositional logic

We consider a propositional language FML_{Var} built up from a finite (but arbitrarily large) set of propositional variables $Var = \{a, b, \ldots\}$, the Boolean constants $\top$ and $\bot$, and the connectives $\neg$, $\wedge$, $\vee$, $\Rightarrow$. Propositional formulas are denoted by Greek letters, typically α, β, μ, φ, ψ, etc. $Var(\psi)$ denotes the set of propositional variables occurring in formula ψ, and $|\psi|$ denotes the size of ψ, i.e., the number of symbols used to write it.

Interpretations over Var (or *worlds*) are denoted by ω, ω', etc. We represent them as (maximal consistent) sets or conjunctions of literals over Var. For instance, if $Var = \{a, b, c\}$, then the world assigning variables a and c to true and variable b to false is denoted by $\{a, \neg b, c\}$. Formulas are interpreted in the classical way. The set of models of a formula φ is the set of interpretations ω such that $\omega \models \varphi$; this set is denoted by $Mod(\varphi)$. $\models$ denotes classical entailment and $\equiv$ denotes logical equivalence.

A *literal* is a propositional variable (a positive literal) or the negation of a propositional variable (a negative literal). For every literal l, $\bar{l}$ is the complementary literal of l, defined by $\bar{l} = \neg x$ when $l = x \in Var$ is a positive literal, and $\bar{l} = x$ when $l = \neg x$ (with $x \in Var$) is a negative literal. For a set of literals L we note $\overline{L} = \{\bar{l} \mid l \in L\}$. Lit_{Var} is the set of literals of the language, i.e., $Lit_{Var} = \bigcup_{x \in Var} \{x, \neg x\} = Var \cup \overline{Var}$.

If ω is a world and L a consistent set of literals from Lit_{Var}, then $(\omega \setminus \overline{L}) \cup L$ is the world that gives the same truth value as ω to all variables except the variables of literals of L and such that every literal of L is true, that is, $(\omega \setminus \overline{L}) \cup L \models (\bigwedge L)$. For instance, if $\omega = \{a, \neg b, c, d\}$ and $L = \{b, c, \neg d\}$ then $(\omega \setminus \overline{L}) \cup L = \{a, b, c, \neg d\}$.

The set of literals of a formula is inductively defined as follows:

$$\begin{aligned}
Lit(\top) &= \emptyset \\
Lit(\bot) &= \emptyset \\
Lit(x) &= \{x\} \text{ for } x \in Var \\
Lit(\neg\varphi) &= \overline{Lit(\varphi)} \\
Lit(\varphi \wedge \psi) &= Lit(\varphi) \cup Lit(\psi) \\
Lit(\varphi \vee \psi) &= Lit(\varphi) \cup Lit(\psi) \\
Lit(\varphi \Rightarrow \psi) &= \overline{Lit(\varphi)} \cup Lit(\psi)
\end{aligned}$$

For instance, for $\varphi = \neg((\neg a \wedge b) \vee (a \wedge c))$ we have $Lit(\varphi) = \{a, \neg a, \neg b, \neg c\}$.

Equivalently, $Lit(\varphi)$ is the set of literals occurring in the Negation Normal Form of φ, noted $NNF(\varphi)$, that is obtained by "pushing down" every occurrence of $\neg$ in φ (including the implicit negations conveyed by $\Rightarrow$), then removing double negations (over variables). Formally, $NNF(\varphi)$ is inductively defined as follows:

— if $\varphi \in Var \cup \{\top, \bot\}$, then $NNF(\varphi) = \varphi$;

— if $\varphi = \alpha \wedge \beta$ then $NNF(\varphi) = NNF(\alpha) \wedge NNF(\beta)$;
— if $\varphi = \alpha \vee \beta$ then $NNF(\varphi) = NNF(\alpha) \vee NNF(\beta)$;
— if $\varphi = \alpha \Rightarrow \beta$ then $NNF(\varphi) = \neg NNF(\alpha) \vee NNF(\beta)$;
— if $\varphi = \neg\alpha$ with $\alpha \in Var \cup \{\top, \bot\}$ then $NNF(\varphi) = \neg\alpha$;
— if $\varphi = \neg(\neg\alpha)$ then $NNF(\varphi) = NNF(\alpha)$;
— if $\varphi = \neg(\alpha \wedge \beta)$ then $NNF(\varphi) = NNF(\neg\alpha) \vee NNF(\neg\beta)$;
— if $\varphi = \neg(\alpha \vee \beta)$ then $NNF(\varphi) = NNF(\neg\alpha) \wedge NNF(\neg\beta)$;
— if $\varphi = \neg(\alpha \Rightarrow \beta)$ then $NNF(\varphi) = NNF(\alpha) \wedge NNF(\neg\beta)$.

When the morphology of the language is limited to the connectives $\neg$, $\wedge$, $\vee$, $\Rightarrow$, and the Boolean constants—as it is the case here—, then $Lit(\varphi)$ can be computed in time linear in the size of φ.

The standard definition of uniform substitution stipulates that $\varphi[x \leftarrow \psi]$ is the formula obtained by uniformly replacing in φ every occurrence of propositional variable x by the formula ψ. It will be convenient to extend this to literals; we identify $\varphi[\neg x \leftarrow \psi]$ with $\varphi[x \leftarrow \neg\psi]$. For the example formula $\varphi = \neg((\neg a \wedge b) \vee (a \wedge c))$ we get $\varphi[\neg a \leftarrow \top] = \neg((\neg\neg\top \wedge b) \vee (\neg\top \wedge c))$.

2.2. Computational Complexity

We assume that the reader is familiar with some basic notions of computational complexity, especially the complexity classes P, NP, and coNP, as well as with the basic decision problems SAT and UNSAT (and their restrictions to 3CNF formulas) and with the classes Δ_k^p, Σ_k^p and Π_k^p of the polynomial hierarchy $\mathsf{PH} = \bigcup_{k\geq 0} \Delta_k^p = \bigcup_{k\geq 0} \Sigma_k^p = \bigcup_{k\geq 0} \Pi_k^p$; we refer the reader to Papadimitriou [1994] for the details.

Let us recall that a decision problem is said to be at the *kth* level of PH if and only if it belongs to Δ_{k+1}^p, and is either Σ_k^p-hard or Π_k^p-hard.

It is well-known that if there exists $i > 0$ such that $\Sigma_i^p = \Pi_i^p$ then for every $j > i$, we have $\Sigma_j^p = \Pi_j^p = \Sigma_i^p$: PH is then said to collapse to level i. It is strongly believed that PH does not collapse (to any level), that is, it is a truly infinite hierarchy: for every integer k, $\mathsf{PH} \neq \Sigma_k^p$.

An advice-taking Turing machine is a Turing machine that has associated with it a special "advice oracle" A, which can be any function (which does not have to be computable). On input s, a special "advice tape" is automatically loaded with $A(|s|)$ and from then on the computation proceeds as normal, based on the two inputs, s and $A(|s|)$.

An advice-taking Turing machine uses polynomial advice if its advice oracle A satisfies $|A(n)| \leq p(n)$ for some fixed polynomial p and all non-negative integers n; finally, P/poly is the class of all languages that can be decided in polynomial time by deterministic Turing machines augmented by polynomial advice. It is known that if $\mathsf{NP} \subseteq \mathsf{P/poly}$, then PH collapses at the second level [Karp and Lipton 1980].

3. A GENERAL FAMILY OF DEPENDENCE-BASED UPDATE OPERATORS

Belief update operators [Katsuno and Mendelzon 1991] map a belief base ψ representing the initial beliefs of a given agent and an input formula μ reflecting the effect of some action or event to a new belief base $\psi \diamond \mu$, which corresponds to the beliefs held by the agent after the action or event took place. In this article we consider propositional belief bases, that we identify with formulas. We typically use φ, ψ, ψ', etc. for belief bases and μ, μ' etc. for input formulas. To define $\psi \diamond \mu$ as the conjunction of the input formula μ with ψ—called an expansion in the belief revision literature—would not be a good idea because $\psi \wedge \mu$ is inconsistent as soon as the input formula μ contradicts ψ. Similarly, belief revision operations [Alchourrón et al. 1985; Katsuno and Mendelzon 1992] are not appropriate for updating belief bases; especially, the belief

revision postulate **(R2)** [Katsuno and Mendelzon 1992], stating that revision must reduce to expansion whenever μ is consistent with ψ, is not expected to hold. As a matter of illustration, consider the well-known example $\psi = d \vee w$ where d means "the door is open" and w means "the window is open." If ψ is revised by the more reliable formula $\mu = \neg d$, it makes sense to assume after the revision step that "the door is closed and the window is open" since no action occurred. However, this conclusion cannot reasonably be drawn if the change formula $\mu = \neg d$ reflects the effect of the action of closing the door; indeed, assume that in the actual state of the world the door is open and the window is closed (which is a model of ψ); for sure, opening the window is not an expected effect of closing the door!

Katsuno and Mendelzon [1991] proposed a general semantics for update operations. The most prominent feature of KM updates (distinguishing update from revision) is that they are performed modelwise: $Mod(\psi \diamond \mu) = \bigcup_{\omega \models \psi} Mod(\omega \diamond \mu)$. (This is a consequence of the KM postulate **(U8)**.) The literature contains many proposals for update operators. Several authors showed that there are good reasons for building an update operator from a dependence function [Doherty et al. 1998; Herzig and Rifi 1999].

The dependence-based update of a belief base ψ by an input formula μ consists in first forgetting in ψ all information "related" to μ (leaving unchanged the variables that are not related to the update), and then expanding the result with μ. It remains to define the notion of "being related to" and the operation of forgetting.

3.1. Formula/Variable Dependence

A *formula/variable dependence function* is modeled as a mapping Dep_v from FML_{Var} to 2^{Var}. (The mapping has to be computable, although this is typically left implicit in the definitions.)

Many choices for Dep_v are possible. Common examples of formula/variable dependence functions are the following.

(1) *Syntactical Dependence*. $Dep_v(\mu) = Var(\mu)$ is the set of propositional variables occurring in μ.
(2) *Semantical Dependence*. $Dep_v(\mu) = RelVar(\mu) \subseteq Var(\mu)$ is the set of relevant variables of μ, defined by: $x \in RelVar(\mu)$ if and only if x occurs in every formula that is logically equivalent to μ; or, equivalently, $x \in RelVar(\mu)$ if and only if $\mu[x \leftarrow \top]$ is not equivalent to $\mu[x \leftarrow \bot]$ [Doherty et al. 1998, 2000; Herzig and Rifi 1998, 1999; Lang and Marquis 1998];
(3) *Dependence Induced by Dependencies between Variables* [Herzig and Rifi 1999]. Let δ_v be a primitive variable dependence function, i.e., a mapping from 2^{Var} to 2^{Var} such that for every $X \subseteq Var$ we have $X \subseteq \delta_v(X)$. (When X is a singleton $\{x\}$, we write $\delta_v(x)$ instead of $\delta_v(X)$ in order to alleviate notation.) Often δ_v can by built from a more basic reflexive relation r_v on $Var \times Var$: $\delta_v(X) = \bigcup_{x \in X} \{x' \mid (x, x') \in r_v\}$. We extend the mapping δ_v to formulas by stipulating that $Var_{\delta_v}(\varphi) = \delta_v(Var(\varphi))$, and $RelVar_{\delta_v}(\varphi) = \delta_v(RelVar(\varphi))$. $Dep_v(\varphi)$ can be defined as any of these two sets, leading thus to two families of update operators.

Whatever the choice, given a dependence function Dep_v, the set of models of the dependence-based update of a world ω by a formula μ w.r.t. Dep_v, denoted by $\omega \diamond_{Dep_v} \mu$, is the set of all worlds ω' such that $\omega' \models \mu$, and such that ω and ω' assign the same truth value to every propositional variable x such that $x \notin Dep_v(\mu)$:

$$Mod(\omega \diamond_{Dep_v} \mu) = \{\omega' \mid \omega' \models \mu \text{ and } Var(\omega' \setminus \omega) \subseteq Dep_v(\mu)\}.^2$$

[2] Let us recall that interpretations are viewed as sets of literals (and not sets of variables).

Dependence function	Induced update operator $\diamond_{Dep_v}$
$Dep_v = Var$	WSS [Winslett 1990]
$Dep_v = RelVar$	MPMA [Doherty et al. 1998] or equivalently WSS$\downarrow$ [Herzig and Rifi 1998]
$Dep_v = Var_{\delta_v}$	WSS$^{\delta_v}$ [Herzig 1996]
$Dep_v = RelVar_{\delta_v}$	WSS$\downarrow^{\delta_v}$ [Herzig and Rifi 1998]
$Dep_v = P$	$\star^P$ [Doherty et al. 1998]

Fig. 1. Update operators based on formula/variable dependence.

Equivalently, $\omega' \in Mod(\omega \diamond_{Dep_v} \mu)$ if and only if the following two conditions are met:

(1) $\omega' \models \mu$;
(2) ω and ω' agree on all propositional variables in $Var \setminus Dep_v(\mu)$.

Example 3.1. Let $\omega = \{\neg a, \neg b, c, d\}$ and $\mu = (\neg a \wedge b \wedge c) \vee ((b \Rightarrow c) \wedge (c \Rightarrow b))$, and let the dependence function between variables δ_v be induced by $r_v = \{(a,a), (a,b), (b,b), (c,c), (c,d), (d,d)\}$. We have:

$$\begin{aligned}
Var(\mu) &= \{a,b,c\};\\
RelVar(\mu) &= \{b,c\};\\
Var_{\delta_v}(\mu) &= \{a,b,c,d\};\\
RelVar_{\delta_v}(\mu) &= \{b,c,d\};\\
Mod(\omega \diamond_{Var} \mu) &= \{\{\neg a,b,c,d\}, \{\neg a,\neg b,\neg c,d\}, \{a,b,c,d\}, \{a,\neg b,\neg c,d\}\};\\
Mod(\omega \diamond_{RelVar} \mu) &= \{\{\neg a,b,c,d\}, \{\neg a,\neg b,\neg c,d\}\};\\
Mod(\omega \diamond_{RelVar_{\delta_v}} \mu) &= \{\{\neg a,b,c,d\}, \{\neg a,\neg b,\neg c,d\}, \{\neg a,b,c,\neg d\}, \{\neg a,\neg b,\neg c,\neg d\}\}.
\end{aligned}$$

Finally, the *dependence-based update of a belief base $\psi \in FML_{Var}$ by the input formula μ w.r.t. Dep_v* is defined as follows:

$$Mod(\psi \diamond_{Dep_v} \mu) = \bigcup_{\omega \models \psi} Mod(\omega \diamond_{Dep_v} \mu).$$

Figure 1 sums up the various dependence functions Dep_v and the associated update operators. In the last line the set P is some (fixed) subset of Var.

Interestingly, $\diamond_{Dep_v}$ operators can be characterized through the notion of *variable forgetting* [Lang et al. 2003a; Lin and Reiter 1994]. Indeed, let us define quantification over propositional variables as $\exists x.\psi = \psi[x \leftarrow \top] \vee \psi[x \leftarrow \bot]$; and let us define quantification over a finite set of variables $X = \{x_1, \ldots, x_n\} \subseteq Var$ as $\exists X.\psi = \exists x_1 \ldots \exists x_n.\psi$. Then the following holds: $\psi \diamond_{Dep_v} \mu \equiv (\exists Dep_v(\mu).\psi) \wedge \mu$ [Doherty et al. 1998].

Note that there exist other closely related belief update operators that are explicitly based on a dependence function [Delgrande et al. 2005; Vo et al. 2006]; they will be discussed in the related work section.

3.2. Formula/Literal Dependence

Unfortunately, forgetting everything about the variables involved in the update often leads to *forgetting too much*. Let us consider the following scenario.

Example 3.2. Let us consider a mobile robot whose internal state is described by the truth values given to a number of fluents, like *powerOn*, *engineReady*, *batteryOK*, etc. Assume that we know at start that the robot satisfies $\psi = powerOn \wedge engineReady \wedge \neg batteryOK$; if the action of making $\mu = (powerOn \vee engineReady) \wedge batteryOK$ true

is performed, then under any formula/variable dependence function Dep_v such that $\{powerOn, engineReady, batteryOK\} \subseteq Dep_v(\mu)$, we get $\psi \diamond_{Dep_v} \mu \equiv \mu$. We thus forgot that initially power was on and the engine ready.

This is not what we expected: the input formula does not convey any evidence that our belief that power is on and the engine is ready should be changed; intuitively, we should not forget the initial information, because updating by μ has *negative* influence neither on *powerOn* nor on *engineReady*. Hence, only *negative* occurrences of *powerOn* and *engineReady* should be forgotten before expanding by the input formula, but not positive occurrences.

Another problem with the previous update operators based on formula/variable dependence is that they cannot handle persistent literals: literals that remain true once they become true.

Example 3.3. Suppose now that an additional fluent *engineDead* is added to the description of the internal state of the robot, and suppose that that the engine cannot be fixed. Assume that we know at start that $\psi = powerOn \wedge engineReady \wedge engineDead$. When the world evolves so that $\mu = \neg powerOn \vee \neg engineReady \vee \neg engineDead$ holds, then using any formula/variable dependence function Dep_v such that

$$\{powerOn, engineReady, engineDead\} \subseteq Dep_v(\mu),$$

we obtain $\psi \diamond_{Dep_v} \mu \equiv \mu$. Once again, this is not what we expected. Although $\neg engineDead$ has a negative influence on *engineDead*, *engineDead* should not be forgotten because it is a persistent fluent, while $\neg engineDead$ is not.

In the last example the truth value of the fluent is essential: formula/variable dependence is too coarse-grained to handle the problem. This calls for the following more fine-grained notion of dependence.

Definition 3.4 (*formula/literal dependence function*). A *formula/literal dependence function* is a computable mapping $Dep : FML_{Var} \to 2^{Lit_{Var}}$.

Intuitively, $Dep(\mu)$ is the set of literals that potentially become true when μ becomes true. The other way round, when l is absent from $Dep(\mu)$ then we know that under no circumstances an update by μ may make l true: if l is false before the update then it will remain false.

Many meaningful functions *Dep* can be considered. Similarly to formula/variable dependence functions, $Dep(\mu)$ can be basically defined as $Lit(\mu)$ or as $RelLit(\mu)$, where the latter is defined as follows.

Definition 3.5 (*formula/literal independence [Lang et al. 2003a]*). Let ψ be a formula from FML_{Var}. ψ is *Lit-independent of* x iff $\psi[x \leftarrow \top] \models \psi$, and ψ is *Lit-independent of* $\neg x$ iff $\psi[x \leftarrow \bot] \models \psi$. Finally, for $l \in Lit_{Var}$, ψ is Lit-dependent on l if and only if it is not Lit-independent of l. We denote by $RelLit(\psi)$ the set of literals ψ depends on.

Example 3.6. The formula $\neg a \vee (a \wedge b)$ is Lit-dependent on $\neg a$, and Lit-independent of a. The latter is the case because $(\neg a \vee (a \wedge b))[a \leftarrow \top] = \neg\top \vee (\top \wedge b) \equiv b$, and $b \models \neg a \vee (a \wedge b)$.

The formula $\psi = (a \vee b) \wedge (\neg a \vee c) \wedge (a \vee b \vee \neg c)$ is Lit-dependent on a, $\neg a$, b and c; but not on $\neg c$. The latter is the case because $\psi[c \leftarrow \bot] = (a \vee b) \wedge (\neg a \vee \bot) \wedge (a \vee b \vee \neg\bot)$, which is equivalent to $(a \vee b) \wedge \neg a$, and $(a \vee b) \wedge \neg a \models (a \vee b) \wedge (\neg a \vee c) \wedge (a \vee b \vee \neg c)$. We have $RelLit(\psi) = \{a, \neg a, b, c\}$.

An alternative, equivalent definition [Lang et al. 2003a] is: ψ is Lit-independent of l if and only if there is an equivalent formula φ such that $l \notin Lit(\varphi)$.

Here are some observations. First, $RelLit(\mu)$ is a subset of $Lit(\mu)$, while the converse is false. Second, μ is Lit-independent of x if and only if $\mu[x \leftarrow \top] \models \mu[x \leftarrow \bot]$, and μ is Lit-independent of $\neg x$ if and only if $\mu[x \leftarrow \bot] \models \mu[x \leftarrow \top]$. Third, $x \in RelVar(\mu)$ if and only if ψ is Lit-dependent on x or on $\neg x$.

More sophisticated dependence functions can be designed by presupposing the existence of a primitive literal dependence function, i.e., a mapping $\delta_l : 2^{Lit_{Var}} \longrightarrow 2^{Lit_{Var}}$, such that for every $L \subseteq Lit_{Var}$, $L \subseteq \delta_l(L)$. Some natural constraints may be imposed on δ_l; for example, monotonicity: if $L_1 \subseteq L_2$ then $\delta_l(L_1) \subseteq \delta_l(L_2)$. We can then define

$$\begin{aligned} Lit_{\delta_l}(\mu) &= \delta_l(Lit(\mu)), \\ RelLit_{\delta_l}(\mu) &= \delta_l(RelLit(\mu)). \end{aligned}$$

When $l \in \delta_l(\{l_1, \ldots, l_n\})$ then the truth of $l_1, \ldots, l_n$ releases the truth of l. This turns out to be particularly useful to solve the ramification problem: whenever a literal l is caused by the truth of some set of literals L then it is natural to have $l \in \delta_l(L)$. This can be ensured by defining $Dep = Lit_{\delta_l}$ (resp. $Dep = RelLit_{\delta_l}$), provided that $L \subseteq Lit(\mu)$ (resp. $L \subseteq RelLit(\mu)$) with δ_l monotone. The truth values of such derived fluents can then be updated through the use of some integrity constraints (reflecting static laws), which have to be conjoined with the input formula.

Example 3.7. Suppose now that an additional fluent *readyToMove* is added to the description of the internal state of the robot. It is a derived fluent and its connections with the previous three primitive fluents are given by the static law

$$\varphi = ((\neg engineDead \wedge powerOn \wedge engineReady) \Rightarrow readyToMove) \wedge$$

$$(readyToMove \Rightarrow (\neg engineDead \wedge powerOn \wedge engineReady)).$$

Assume that initially

$$\psi = powerOn \wedge engineReady \wedge \neg engineDead \wedge readyToMove.$$

Assume also that

$$\mu = \neg powerOn \vee \neg engineReady \vee engineDead.$$

Suppose $Dep = Lit_{\delta_l}$ where δ_l is such that

$$\neg readyToMove \in \delta_l(\{\neg powerOn, \neg engineReady, engineDead\}).$$

If the world evolves so that μ holds then *readyToMove* is forgotten, as expected.

Using our formula/literal dependence functions, one can also easily take into account persistent literals, i.e., literals remaining true whatever happens, such as *engineDead* (contrarily to $\neg engineDead$). For example, a possible dependence function is $Dep(\mu) = RelLit(\mu) \setminus \overline{PersLit}$, where *PersLit* is a given set of persistent literals.

Example 3.8. Let $\psi = powerOn \wedge engineReady \wedge engineDead$ and let $\mu = \neg powerOn \vee \neg engineReady \vee \neg engineDead$. If $PersLit = \{engineDead\}$, then

$$Dep(\mu) = RelLit(\mu) \setminus \overline{PersLit} = \{\neg powerOn, \neg engineReady\},$$

and *engineDead* will not be forgotten in ψ.

3.3. Update Based on Formula/Literal Dependence

We are now ready to define our update operators based on formula/literal dependence:

Definition 3.9 (*update based on formula/literal dependence*). Let *Dep* be a formula/literal dependence function. Let φ be a formula representing integrity constraints. The

set of models of the *update based on formula/literal dependence* $\omega \diamond^{\varphi}_{Dep} \mu$ of the world ω by the formula μ w.r.t. *Dep* and φ is the set of all worlds ω' such that $\omega' \models \mu \wedge \varphi$ and $\omega' \setminus \omega \subseteq Dep(\mu)$. In symbols:

$$Mod(\omega \diamond^{\varphi}_{Dep} \mu) = \{\omega' \models \mu \wedge \varphi \mid \omega' \setminus \omega \subseteq Dep(\mu)\}.$$

Last, if ψ is a formula from FML_{Var}, then

$$Mod(\psi \diamond^{\varphi}_{Dep} \mu) = \bigcup_{\omega \models \psi} Mod(\omega \diamond^{\varphi}_{Dep} \mu).$$

An equivalent formulation of $Mod(\omega \diamond^{\varphi}_{Dep} \mu)$ is:

$$Mod(\omega \diamond^{\varphi}_{Dep} \mu) = \bigcup_{L \subseteq Dep(\mu)} \{\omega' \models \mu \wedge \varphi \mid \omega = (\omega' \setminus L) \cup \overline{L}\}.$$

Example 3.10. Let $\omega = \{powerOn, engineReady\}$ and $\mu = powerOn \vee engineReady$. Using any formula/literal dependence function *Dep* such that $Dep(powerOn \vee engineReady)$ neither contains $\neg powerOn$ nor $\neg engineReady$, we get $\omega \diamond^{\top}_{Dep} \mu \models powerOn \wedge engineReady$.

The next result gives us a more practical way to compute $\diamond^{\varphi}_{Dep}$. We first need to recall the notion of literal forgetting.

Definition 3.11 (*literal forgetting [Lang et al. 2003a]*). Let ψ be a formula and let $L \subseteq Lit_{Var}$. Then $\exists_l L.\psi$ is the formula inductively defined by:

$$\begin{array}{rcl} \exists_l \emptyset.\psi & = & \psi \\ \exists_l \{l\}.\psi & = & \psi[l \leftarrow \top] \vee (\neg l \wedge \psi) \\ \exists_l (\{l\} \cup L).\psi & = & \exists_l L.(\exists_l \{l\}.\psi) \end{array}$$

$\exists_l L.\psi$ is (up to logical equivalence) the logically strongest consequence of ψ that is Lit-independent of L, i.e., it is Lit-independent of each $l \in L$ (see Proposition 16 from Lang et al. [2003a]).

On this ground, the following characterization of $\psi \diamond^{\varphi}_{Dep} \mu$ follows.

PROPOSITION 3.12. *Let Dep be a formula/literal dependence function. Let φ be a formula representing integrity constraints. Let ψ and μ be two formulas. We have:*

$$\psi \diamond^{\varphi}_{Dep} \mu \equiv \mu \wedge \varphi \wedge \exists_l \overline{Dep(\mu)}.\psi$$

PROOF. From Proposition 15 of Lang et al. [2003a], we get that

$$\begin{array}{rcl} Mod(\mu \wedge \varphi \wedge \exists_l \overline{Dep(\mu)}.\psi) & = & \{\omega' \models \mu \wedge \varphi \mid (\omega' \setminus \overline{L}) \cup L \models \psi \text{ where } L \subseteq \overline{Dep(\mu)}\} \\ & = & \bigcup_{\omega \models \psi} \{\omega' \models \mu \wedge \varphi \mid \omega = (\omega' \setminus \overline{L}) \cup L \text{ where } L \subseteq \overline{Dep(\mu)}\} \\ & = & \bigcup_{\omega \models \psi} \{\omega' \models \mu \wedge \varphi \mid \omega = (\omega' \setminus L) \cup \overline{L} \text{ where } L \subseteq Dep(\mu)\} \\ & = & Mod(\psi \diamond^{\varphi}_{Dep} \mu) \end{array}$$ □

Example 3.13. Let $\psi = (\neg a \vee \neg d) \wedge (a \vee b) \wedge (a \vee c \vee d)$, $\mu = d$ and let $Dep = RelLit$ and $\varphi = \top$. We have $Dep(\mu) = \{d\}$ and $\exists_l \{\neg d\}.\psi \equiv (a \vee b) \wedge (a \vee c \vee d)$; hence $\psi \diamond^{\varphi}_{Dep} \mu \equiv d \wedge (a \vee b)$.

A direct consequence of Proposition 3.12 is that we also have $\psi \diamond^{\varphi}_{Dep} \mu \equiv \mu \wedge \varphi \wedge \exists_l (\overline{Dep(\mu)} \cap \overline{Lit(\psi)}).\psi$. Indeed, if $l \in \overline{Dep(\mu)}$ and $l \notin \overline{Lit(\psi)}$, then $\exists_l \overline{Dep(\mu)}.\psi$ is

equivalent to $\exists_l(\overline{Dep(\mu)} \setminus \{l\}).\psi$: forgetting literals not occurring in a formula does not change anything.

Clearly enough, our family of operators contains all previous operators based on formula/variable dependence.

PROPOSITION 3.14. *Every update operator based on formula/variable dependence is an update operator based on formula/literal dependence.*

PROOF. Given that $\exists\{x\}.\psi \equiv \exists_l\{x, \neg x\}.\psi$ for every formula ψ and variable x (see Proposition 20 from Lang et al. [2003a]), any operator $\diamond_{Dep_v}$ based on formula/variable dependence is the same as the update operator $\diamond^{\top}_{Dep}$ based on formula/literal dependence, where Dep is defined by $Dep(\alpha) = \bigcup_{x \in Dep_v(\alpha)}\{x, \neg x\}$ for any formula α. □

It will be useful to consider two extreme cases of formula/literal dependence. Call *full* the formula/literal dependence function Dep_f such that $Dep_f(\mu) = Lit_{Var}$ for every formula μ; and call *empty* the formula/literal dependence function Dep_e such that $Dep_e(\mu) = \emptyset$ for every formula μ.

PROPOSITION 3.15. *Let Dep_e be the empty formula/literal dependence function and let Dep_f be the full formula/literal dependence function. Then*

$$\psi \diamond^{\top}_{Dep_e} \mu = \psi \wedge \mu$$

$$\psi \diamond^{\top}_{Dep_f} \mu = \begin{cases} \mu & \text{if } \psi \text{ is consistent} \\ \bot & \text{otherwise} \end{cases}$$

PROOF. For the case of the empty dependence function we use that $\exists_l\overline{\emptyset}.\psi$ is equivalent to ψ. For the case of the full dependence function, we use that $\exists_l\overline{Lit_{Var}}.\psi \equiv \bot$ if ψ is inconsistent and $\exists_l\overline{Lit_{Var}}.\psi \equiv \top$ otherwise. □

In the literature, $\diamond^{\top}_{Dep_e}$ is called the expansion operator.

4. RATIONALITY POSTULATES

We now examine the status of the KM postulates. We do so both in the general case and for specific formula/literal dependence functions.

Katsuno and Mendelzon [1991] proposed the following eight postulates for characterizing rational update operators.

(U1) $\psi \diamond \mu \models \mu$.
(U2) If $\psi \models \mu$ then $\psi \diamond \mu \equiv \psi$.
(U3) If ψ and μ are consistent then $\psi \diamond \mu$ is consistent.
(U4) If $\psi_1 \equiv \psi_2$ and $\mu_1 \equiv \mu_2$ then $\psi_1 \diamond \mu_1 \equiv \psi_2 \diamond \mu_2$.
(U5) $(\psi \diamond \mu_1) \wedge \mu_2 \models \psi \diamond (\mu_1 \wedge \mu_2)$.
(U6) If $\psi \diamond \mu_1 \models \mu_2$ and $\psi \diamond \mu_2 \models \mu_1$ then $\psi \diamond \mu_1 \equiv \psi \diamond \mu_2$.
(U7) If ψ is complete then $(\psi \diamond \mu_1) \wedge (\psi \diamond \mu_2) \models \psi \diamond (\mu_1 \vee \mu_2)$.
(U8) $(\psi_1 \vee \psi_2) \diamond \mu$ is equivalent to $(\psi_1 \diamond \mu) \vee (\psi_2 \diamond \mu)$.

These postulates are designed for updates without integrity constraints. For this reason, we assume in this section (and in this section only) that $\varphi = \top$. Note also that is possible to extend such postulates in order to account for integrity constraints (see Herzig and Rifi [1999] for such a contribution).

4.1. General Results

The relationships between the family of KM update operators and the family of update operators based on formula/literal dependence are made precise by the following propositions. First, Proposition 4.1 shows that the two families are disjoint because no update operator $\diamond^{\top}_{Dep}$ based on formula/literal dependence satisfies both **(U2)** and **(U3)**. Furthermore, giving up any of those two postulates is enough to overcome this impossibility:

PROPOSITION 4.1. *No update operator $\diamond^{\top}_{Dep}$ based on formula/literal dependence is a KM update operator. More precisely:*

(1) No update operator $\diamond^{\top}_{Dep}$ based on formula/literal dependence satisfies both **(U2)** *and* **(U3)**.
(2) There exists an update operator based on formula/literal dependence that satisfies all KM postulates but **(U2)**.
(3) There exists an update operator based on formula/literal dependence that satisfies all KM postulates but **(U3)**.

PROOF.

(1) Let $\diamond^{\top}_{Dep}$ be an update operator based on formula/literal dependence. If $\diamond^{\top}_{Dep}$ satisfies **(U2)**, then for any formulas ψ and μ such that $\psi \models \mu$ holds, it must be the case that $\psi \diamond^{\top}_{Dep} \mu \equiv \psi$. Equivalenty, if $\psi \models \mu$ holds, then we must have $\mu \wedge \exists_l \overline{Dep(\mu)}.\psi \equiv \psi$, which implies that $\exists_l \overline{Dep(\mu)}.\psi \models \neg\mu \vee \psi$ must be satisfied. Now, let $Var = \{a, b\}$, $\mu = a \vee \neg b$, and let us consider the following formulas:
   (a) $\psi_1 = a \wedge b$. Since $\psi_1 \models \mu$ holds, we must have

$$\exists_l \overline{Dep(a \vee \neg b)}.(a \wedge b) \models \neg(a \vee \neg b) \vee (a \wedge b).$$

   $\neg(a \vee \neg b) \vee (a \wedge b)$ is equivalent to b, hence this entailment can be the case only if $\neg b \notin Dep(a \vee \neg b)$.
   (b) $\psi_2 = \neg a \wedge \neg b$. Since $\psi_2 \models \mu$ holds, we must have

$$\exists_l \overline{Dep(a \vee \neg b)}.(\neg a \wedge \neg b) \models \neg(a \vee \neg b) \vee (\neg a \wedge \neg b).$$

   $\neg(a \vee \neg b) \vee (\neg a \wedge \neg b)$ is equivalent to $\neg a$, hence this entailment can be the case only if $a \notin Dep(a \vee \neg b)$.
   (c) $\psi_3 = \neg a \wedge b$. By definition, $\psi_3 \diamond^{\top}_{Dep} \mu$ is equivalent to

$$(a \vee \neg b) \wedge \exists_l \overline{Dep(a \vee \neg b)}.(\neg a \wedge b).$$

   From items (a) and (b) above, neither a nor $\neg b$ belongs to $Dep(a \vee \neg b)$. As a consequence, we have

$$\exists_l \overline{Dep(a \vee \neg b)}.(\neg a \wedge b) \equiv \neg a \wedge b.$$

   Accordingly, $\psi_3 \diamond^{\top}_{Dep} \mu$ is inconsistent. Since neither ψ_3 nor μ is inconsistent, this contradicts the fact that $\diamond^{\top}_{Dep}$ satisfies **(U3)**.

(2) It is easy to check that the update operator based on full formula/literal dependence ($Dep_f(\mu) = Lit_{Var}$ for every formula μ) satisfies all KM postulates but **(U2)**. (We use that $\psi \diamond^{\top}_{Dep_f} \mu$ is equivalent to μ if ψ is consistent and is equivalent to $\bot$ otherwise, cf. Proposition 3.15.)

(3) It is also easy to check that the update operator based on empty formula/literal dependence (defined by $Dep_e(\mu) = \emptyset$ for every formula μ) satisfies all KM postulates but **(U3)**. (We use that $\psi \diamond^{\top}_{Dep_e} \mu$ equals $\psi \wedge \mu$, cf. Proposition 3.15.) □

Now, Proposition 4.2 makes precise the KM postulates that are offered by every update operator based on formula/literal dependence.

PROPOSITION 4.2. *Update operators $\diamond^{\top}_{Dep}$ based on formula/literal dependence satisfy the two postulates* **(U1)** *and* **(U8)**. *The postulates* **(U2)**, **(U3)**, **(U4)**[3], **(U5)**, **(U6)** *and* **(U7)** *are not satisfied by update operators $\diamond^{\top}_{Dep}$ in the general case.*

PROOF.

(U1) Obvious from the definition of update operators based on formula/literal dependence.
(U2) Not satisfied by MPMA [Doherty et al. 1998].
(U3) Let $\psi = a$, $\mu = \neg a$ and let *Dep* be such that $\neg a \notin Dep(\mu)$ (e.g., $Dep(\mu) = \emptyset$ for every μ); then ψ and μ are consistent but $\psi \diamond^{\top}_{Dep} \mu$ is not.
(U4) Let $\psi_1 = \psi_2 = a$, $\mu_1 = \top$, $\mu_2 = a \vee \neg a$ and $Dep = Lit$; we have $\psi_1 \equiv \psi_2$ and $\mu_1 \equiv \mu_2$, but $\psi_1 \diamond^{\top}_{Dep} \mu_1 \equiv a$ is not equivalent to $\psi_2 \diamond^{\top}_{Dep} \mu_2 \equiv \top$.
(U5) Not satisfied by MPMA [Doherty et al. 1998].
(U6) Not satisfied by MPMA [Doherty et al. 1998].
(U7) Not satisfied by MPMA [Doherty et al. 1998].
(U8) Obvious from the definition of update operators based on formula/literal dependence. □

Finally, while Proposition 4.1 shows that the KM update operators and the update operators based on formula/literal dependence form two disjoint families, every operator from their union bounds in the same way the belief base $\psi \diamond \mu$ resulting from the update of ψ by μ. Furthermore, these bounds are reached for both families of operators. Formally, we have the following.

PROPOSITION 4.3. *Let $\diamond$ be a KM update operator or an update operator based on formula/literal dependence, of the form $\diamond^{\top}_{Dep}$. Then for every formulas ψ and μ we have:*

$$\psi \wedge \mu \models \psi \diamond \mu \models \mu.$$

PROOF.

— If $\diamond$ is a KM update operator then it satisfies **(U1)**, therefore $\psi \diamond \mu \models \mu$. It remains to show that $\psi \wedge \mu \models \psi \diamond \mu$. Consider the instance $(\psi \diamond \top) \wedge \mu \models \psi \diamond (\top \wedge \mu)$ of **(U5)**. By **(U2)**, $\psi \diamond \top \equiv \psi$, and by **(U4)**, $\psi \diamond (\top \wedge \mu) \equiv \psi \diamond \mu$. It follows from all this that $\psi \wedge \mu \models \psi \diamond \mu$.
— If $\diamond$ is an update operator based on formula/literal dependence, of the form $\diamond^{\top}_{Dep}$, then $\psi \diamond \mu \equiv \mu \wedge \exists_l \overline{Dep(\mu)}.\psi$. By definition of literal forgetting, $\exists_l \overline{Dep(\mu)}.\psi$ is a logical consequence of ψ. Then the result follows trivially. □

4.2. The Cases of Specific Dependence Functions

We now study some specific formula/literal dependence functions that are obtained by imposing some requirements on *Dep*. Proposition 3.15 already characterized the update operators that are based on empty and on full dependence functions.

[3]We note that logically equivalent bases ψ_1 and ψ_2 are however updated equivalently: if $\psi_1 \equiv \psi_2$ then $\psi_1 \diamond^{\varphi}_{Dep} \mu \equiv \psi_2 \diamond^{\varphi}_{Dep} \mu$.

Definition 4.4. Let *Dep* be a formula/literal dependence function.

— *Dep* is *standard* iff $RelLit(\mu) \subseteq Dep(\mu)$ for every μ.
— *Dep* is *syntax-independent* iff $\mu_1 \equiv \mu_2$ implies $Dep(\mu_1) = Dep(\mu_2)$ for every μ_1, μ_2.
— *Dep* is *monotone* iff $\mu_1 \models \mu_2$ implies $Dep(\mu_1) \supseteq Dep(\mu_2)$ for every μ_1, μ_2.
— *Dep* is *antimonotone* iff $\mu_1 \models \mu_2$ implies $Dep(\mu_1) \subseteq Dep(\mu_2)$ for every μ_1, μ_2.
— *Dep* is *confined* iff $Dep(\mu) \subseteq Lit(\mu)$ for every μ.
— *Dep* is *constant* iff $Dep(\mu_1) = Dep(\mu_2)$ for every μ_1, μ_2.

The mappings *Lit* and *RelLit* are neither monotone nor antimonotone: to see the former take $\mu_1 = a$ and $\mu_2 = a \vee b$; to see the latter take $\mu_1 = a \wedge b$ and $\mu_2 = a$. They are also not constant. *RelLit* is syntax-independent while *Lit* is not. Finally, both mappings are standard and confined.

Clearly enough, since $\bot \models \mu$ and $\mu \models \top$ for every formula μ, dependence functions that are both monotone and antimonotone are constant (and therefore syntax-independent). Furthermore, for the same reason, given that $Lit(\bot) = Lit(\top) = \emptyset$, there is a unique function that is both monotone and confined, or both antimonotone and confined, viz. the empty dependence function leading to the expansion operator (cf. Proposition 3.15. Moreover, there is exactly one dependence function that is both standard and monotone, viz. the full dependence function (because $\mu \models \mu \vee l$ for every $l \in Lit$), and there is exactly one dependence function that is both standard and antimonotone, viz. the full dependence function (because $\mu \wedge l \models \mu$ for every $l \in Lit$).

These restrictions lead to operators satisfying more KM postulates as follows.

PROPOSITION 4.5. *Let Dep be a formula/literal dependence function and let* $\diamond^{\top}_{Dep}$ *be the induced update operator.*

(1) If Dep is standard then $\diamond^{\top}_{Dep}$ *satisfies* **(U3)**.
(2) If Dep is syntax-independent then $\diamond^{\top}_{Dep}$ *satisfies* **(U4)**.
(3) If Dep is monotone then $\diamond^{\top}_{Dep}$ *satisfies* **(U5)**.
(4) If Dep is constant then $\diamond^{\top}_{Dep}$ *satisfies* **(U6)**.
(5) If Dep is antimonotone then $\diamond^{\top}_{Dep}$ *satisfies* **(U7)**.

Before proceeding to the proof of Proposition 4.5, we first state a lemma that will be used in the proof of Proposition 4.5 and then used again in Section 5. It relates Levesque's notion of separability [1998] and the following notion of *Lit-separability*:

Definition 4.6 (*Lit-separability*). Two formulas α and β are said to be *Lit-separable* if and only if there is no $l \in Lit_{Var}$ such that $l \in RelLit(\alpha)$ and $\bar{l} \in RelLit(\beta)$.

Since $\exists_l \overline{Dep(\mu)}.\psi$ is Lit-independent of $\overline{Dep(\mu)}$ (see Proposition 16 from Lang et al. [2003a]), we have that if *Dep* is a standard function then μ and $\exists_l \overline{Dep(\mu)}.\psi$ are Lit-separable. This implies that they are also separable in Levesque's sense.

LEMMA 4.7. *If* α *and* β *are Lit-separable formulas, then they are separable in Levesque's sense [1998], that is, for every clause* γ*, we have* $\alpha \wedge \beta \models \gamma$ *if and only if* $\alpha \models \beta$ *or* $\alpha \models \gamma$.

PROOF. From Theorem 5 of Lang et al. [2003a], there exist α' and β' such that $\alpha \equiv \alpha'$ and $\beta \equiv \beta'$ and α' and β' are Lit-simplified, that is, such that $RelLit(\alpha') = Lit(\alpha')$ and

$RelLit(\beta') = Lit(\beta')$. Since $\alpha \equiv \alpha'$ and $\beta \equiv \beta'$, we have $RelLit(\alpha') = RelLit(\alpha)$ and $RelLit(\beta') = RelLit(\beta)$; hence, α' and β' are two Lit-separable formulas and we have for every literal $l \in Lit_{Var}$, if $l \in Lit(\alpha')$ then $\bar{l} \notin Lit(\beta')$. We can assume w.l.o.g. that α' and β' are CNF formulas, so that $\alpha' \wedge \beta'$ is also a CNF formula. Now, let l be any literal occurring in $\alpha' \wedge \beta'$ and let x_l be the corresponding variable; we know that for every literal l', if $l' \in Lit(\alpha')$ then $\overline{l'} \notin Lit(\beta')$; as a consequence, x_l is pure in $\alpha' \wedge \beta'$ (i.e., $\bar{l}$ does not occur in $\alpha' \wedge \beta'$) or x_l occurs either in α or in β, but not in both. Subsequently, there is no possible resolution step in $\alpha' \wedge \beta'$ between a clause from α and a clause from β: any resolvent γ of two clauses of $\alpha' \wedge \beta'$ is such that the two clauses are in α or the two clauses are in β. Assume that γ is the resolvent of two clauses of α (the other case is similar); then we still have that every variable occurring in $(\alpha \wedge \gamma) \wedge \beta$ is pure or such that it occurs either in $\alpha \wedge \gamma$ or in β, but not in both. A straightforward induction on the length of resolution proofs is enough to conclude that every clause that can be derived by (general) resolution from $\alpha \wedge \beta$ can be derived from α alone or from β alone. Since (general) resolution is complete in consequence-finding, we conclude that every prime implicate of $\alpha \wedge \beta$ is a prime implicate of α or a prime implicate of β (see [Marquis 2000] for details). Finally, Corollary 3 of Lang et al. [2002] shows that α and β are separable in Levesque's sense if and only if every prime implicate of $\alpha \wedge \beta$ is a prime implicate of α or a prime implicate of β, and this concludes the proof. □

It is easy to check that Lit-separability alone is a sufficient condition for Levesque's separability. For instance, $\alpha = a \vee b$ and $\beta = \neg a \vee \neg b$ are separable in Levesque's sense but are not Lit-separable. Now we proceed to the proof of Proposition 4.5.

PROOF.

(1) This is a direct consequence of Proposition 4.7. If Dep is a standard function then μ and $\exists_l \overline{Dep(\mu)}.\psi$ are Lit-separable, hence $\mu \wedge \exists_l \overline{Dep(\mu)}.\psi \models \bot$ (where $\bot$ is viewed as the empty clause) if and only if $\mu \models \bot$ or $\exists_l \overline{Dep(\mu)}.\psi \models \bot$.
(2) Easy from the fact that $\exists_l \overline{Dep(\mu_1)}.\psi_1 \equiv \exists_l \overline{Dep(\mu_2)}.\psi_2$ when $\psi_1 \equiv \psi_2$ and $Dep(\mu_1) = Dep(\mu_2)$.
(3) The result easily comes from the fact that $\exists_l \overline{Dep(\mu)}.\psi \models \exists_l \overline{Dep(\mu \wedge \varphi)}.\psi$ when Dep is monotone.
(4) By definition, Dep is constant if and only if for any $\mu_1, \mu_2 \in FML_{Var}$, $Dep(\mu_1) = Dep(\mu_2) = P$. Hence $\psi \diamond^{\top}_{Dep} \mu_1 \equiv \mu_1 \wedge \exists_l \overline{P}.\psi$ and $\psi \diamond^{\top}_{Dep} \mu_2 \equiv \mu_2 \wedge \exists_l \overline{P}.\psi$. If $\mu_1 \wedge \exists_l \overline{P}.\psi \models \mu_2$, then $\mu_1 \wedge \exists_l \overline{P}.\psi \models \mu_2 \wedge \exists_l \overline{P}.\psi$. Similarly, if $\mu_2 \wedge \exists_l \overline{P}.\psi \models \mu_1$, then $\mu_2 \wedge \exists_l \overline{P}.\psi \models \mu_1 \wedge \exists_l \overline{P}.\psi$. Accordingly, if $\psi \diamond^{\top}_{Dep} \mu_1 \models \mu_2$ and $\psi \diamond^{\top}_{Dep} \mu_2 \models \mu_1$, then we have $\psi \diamond^{\top}_{Dep} \mu_1 \equiv \psi \diamond^{\top}_{Dep} \mu_2$.
(5) Similarly, the result for **(U7)** comes from the fact that $\exists_l \overline{Dep(\mu)}.\psi \models \exists_l \overline{Dep(\mu \wedge \varphi)}.\psi$ when Dep is antimonotone. □

COROLLARY 4.8. *Let δ_l be a primitive literal dependence function.*

— *The update operator $\diamond_{RelLit_{\delta_l}}$ satisfies* **(U1)**, **(U3)**, **(U4)** *and* **(U8)**.
— *The update operator $\diamond_{Lit_{\delta_l}}$ satisfies* **(U1)**, **(U3)** *and* **(U8)**.

Taking $\delta_l(L) = L$ for any L, we obtain, in particular, that $\diamond_{RelLit}$ satisfies **(U1)**, **(U3)**, **(U4)** and **(U8)** and that $\diamond_{Lit}$ satisfies **(U1)**, **(U3)** and **(U8)**.

A further property of decomposability has been proposed in Hoffmann et al. [2009]: formulas talking about disjoint sets of variables can be updated separately.[4] Formally, we have the following.

(D) if $Var(\varphi_1) \cap Var(\varphi_2) = \emptyset$, $Var(\varphi_1) \cap Var(\mu_2) = \emptyset$, $Var(\mu_1) \cap Var(\varphi_2) = \emptyset$ and $Var(\mu_1) \cap Var(\mu_2) = \emptyset$, then $(\varphi_1 \wedge \varphi_2) \diamond (\mu_1 \wedge \mu_2) \equiv (\varphi_1 \diamond \mu_1) \wedge (\varphi_2 \diamond \mu_2)$.

We now say that a formula/literal dependence function *Dep* is *decomposable* if and only if for any pair of consistent formulas μ_1 and μ_2, if $Var(\mu_1) \cap Var(\mu_2) = \emptyset$, then $Dep(\mu_1 \wedge \mu_2) = Dep(\mu_1) \cup Dep(\mu_2)$. This property is quite innocuous, and it is easy to check that it is verified by all the basic dependence functions *Var*, *RelVar*, *Lit*, *RelLit* and *P*.

PROPOSITION 4.9. *If Dep is confined and decomposable, then* $\diamond_{Dep}$ *satisfies* **(D)**.

PROOF. Let *Dep* be confined and decomposable, and suppose $\varphi_1, \varphi_2, \mu_1, \mu_2$ satisfy the conditions given in **(D)**. By the definition of $\diamond_{Dep}$, if at least one of μ_1 and μ_2 is inconsistent then $(\varphi_1 \wedge \varphi_2) \diamond_{Dep} (\mu_1 \wedge \mu_2)$ is inconsistent and at least one of $\varphi_1 \diamond \mu_1$ and $\varphi_2 \diamond \mu_2$ is inconsistent. Hence **(D)** is satisfied. Assume now that both μ_1 and μ_2 are consistent. Because *Dep* is confined, we have $Dep(\mu_1) \subseteq Lit(\mu_1)$ and $Dep(\mu_2) \subseteq Lit(\mu_2)$. $Var(\mu_1) \cap Var(\mu_2) = \emptyset$ implies $Lit(\mu_1) \cap Lit(\mu_2) = \emptyset$, therefore (1) $Dep(\mu_1) \cap Dep(\mu_2) = \emptyset$. Now, $\exists_l \overline{Dep(\mu_1 \wedge \mu_2)}.(\varphi_1 \wedge \varphi_2) \equiv \exists_l \overline{Dep(\mu_1) \cup Dep(\mu_2)}.(\varphi_1 \wedge \varphi_2)$ because *Dep* is decomposable, which together with (1) and the definition of $\exists_l$, leads to $\exists_l \overline{Dep(\mu_1 \wedge \mu_2)}.(\varphi_1 \wedge \varphi_2) \equiv (\exists_l \overline{Dep(\mu_1)}.\varphi_1) \wedge (\exists_l \overline{Dep(\mu_2)}.\varphi_2)$. Therefore,

$$\begin{aligned}
(\varphi_1 \wedge \varphi_2) \diamond (\mu_1 \wedge \mu_2) &\equiv (\exists_l \overline{Dep(\mu_1)}.\varphi_1) \wedge (\exists_l \overline{Dep(\mu_2)}.\varphi_2) \wedge (\mu_1 \wedge \mu_2) \\
&\equiv ((\exists_l \overline{Dep(\mu_1)}.\varphi_1) \wedge \mu_1) \wedge ((\exists_l \overline{Dep(\mu_2)}.\varphi_2) \wedge \mu_2) \\
&\equiv (\varphi_1 \diamond_{Dep_v} \mu_1) \wedge (\varphi_2 \diamond_{Dep_v} \mu_2)
\end{aligned}$$

□

Since the dependence functions *Lit* and *RelLit* are both confined and decomposable, the corresponding update operators satisfy **(D)**.

Here are now some general negative results.

PROPOSITION 4.10.

(1) No operator $\diamond^{\top}_{Dep}$ *based on a standard formula/literal dependence function satisfies* **(U2)**.

(2) No operator $\diamond^{\top}_{Dep}$ *based on a standard and confined formula/literal dependence function satisfies* **(U6)**.

PROOF.

(1) Let $\psi = a \wedge b$ and $\mu = a \vee \neg b$. We have $\psi \models \mu$. As *Dep* is standard, both a and $\neg b$ are contained in $Dep(\mu)$. Therefore the set of literals $\overline{Dep(\mu)}$ to be forgotten in ψ contains both $\neg a$ and b. Hence $\psi \diamond^{\top}_{Dep} \mu \not\models b$.

(2) Let $\psi = a \wedge b$, $\mu_1 = a \vee \neg b$ and $\mu_2 = a$. We have $RelLit(\mu_1) = Lit(\mu_1) = \{a, \neg b\}$. If *Dep* is standard and confined, then $\overline{Dep(\mu_1)} = \{\neg a, b\}$ has to be forgotten in ψ. Hence we have $\psi \diamond^{\top}_{Dep} \mu_1 \equiv a$. Thus, $\psi \diamond^{\top}_{Dep} \mu_1 \models \mu_2$. Furthermore, we have

[4] A weaker requirement is the following postulate that was introduced for belief revision by Parikh [1999] and further studied by other authors [Bienvenu et al. 2008; Kourousias and Makinson 2007; Makinson 2007]:

(D′) if $Var(\varphi_1) \cap Var(\varphi_2) = \emptyset$ and $Var(\mu) \subseteq Var(\varphi_2)$ then $(\varphi_1 \wedge \varphi_2) \diamond \mu \equiv \varphi_1 \wedge (\varphi_2 \diamond \mu)$.

$RelLit(\mu_2) = Lit(\mu_2) = \{a\}$. $\overline{Dep(\mu_2)} = \{\neg a\}$ has to be forgotten in ψ. Hence, we have $\psi \diamond^{\top}_{Dep} \mu_2 \equiv a \wedge b$, which implies μ_1. However, $\psi \diamond^{\top}_{Dep} \mu_2 \not\equiv \psi \diamond^{\top}_{Dep} \mu_1$,and the conclusion follows. □

Whether an update operator should satisfy **(U2)** or not depends on what update is meant to represent. Viewing an update by μ as the action of making μ true, **(U2)** should be satisfied whenever the action of making μ true should leave the world unchanged as soon as μ is already true (see, for instance, Lang [2007]). There are many situations where this should not be satisfied (see for instance in Herzig and Rifi [1999], Example 39, Remark 40, and the discussion after Theorem 31, which argue why **(U2)** should not be required in the general case). However, when **(U2)** is desirable, it is always possible to enforce it without hampering the other desirable properties (see [Delgrande et al. 2008]) just by defining the update operator $\diamond'$ associated with $\diamond$ by:

$$\varphi \diamond' \mu = \begin{cases} \varphi & \text{if } \varphi \models \mu \\ \varphi \diamond \mu & \text{otherwise.} \end{cases}$$

Explaining why **(U5)**, **(U6)**, and **(U7)** are not satisfied in the general case is more difficult, due to the technical nature of these postulates, whose role is to guarantee the existence of a collection of faithful preorders on the minimization of which the operator is defined. Therefore, it is not easy to go beyond the explanation that these postulates are not satisfied in the general case because dependence-based operators are not based on minimization.

5. COMPUTATIONAL ASPECTS

We consider now a number of complexity issues for update operators based on formula/literal dependence. Some results hold for the whole family of operators, and others for some subsets that are obtained by imposing restrictions on *Dep*. We also consider some compactability issues.

5.1. The Model Checking Problem

We first investigate the complexity of the model checking problem from an updated base. The model checking problem for $\diamond_{Dep}$ is as follows: given three formulas φ, ψ, μ and a world ω, how difficult is it to determine whether $\omega \models \psi \diamond^{\varphi}_{Dep} \mu$ holds? Here and in the next subsection we suppose that determining whether $l \in Dep(\mu)$ is in NP. This is clearly the case for $Dep = Lit$ and $Dep = RelLit$.

PROPOSITION 5.1. *Let Dep be a formula/literal dependence function such that for any $l \in Lit_{Var}$ and $\mu \in FML_{Var}$, determining whether $l \in Dep(\mu)$ is in* NP. *The model checking problem for $\diamond_{Dep}$ is in* NP. *It is* NP*-complete when Dep is a standard function, even in the restricted case when no integrity constraints are considered (i.e., when φ equals $\top$).*

PROOF. *Membership.* If determining whether $l \in Dep(\mu)$ is in NP, then determining whether any given set $L \subseteq Lit_{Var}$ of literals is included in $Dep(\mu)$ is in NP as well. Hence for every positive instance $\langle L, \mu \rangle$ of this decision problem, there exists a certificate c of size polynomial in the input size such that verifying that $\langle L, \mu \rangle$ is a positive instance can be achieved in (deterministic) polynomial time when c is given. Let $V = Var(\psi) \cup Var(\varphi) \cup Var(\mu)$. The problem can be solved in nondeterministic polynomial time, thanks to the following algorithm: guess a world ω' over V, a subset L of $V \cup \overline{V}$ and a string c (both of them being of polynomial size in the input size), then check in polynomial time that $L \subseteq Dep(\mu)$ using c, and check that $\omega \models \psi$, $\omega' \models \mu \wedge \varphi$ and $\omega = (\omega' \setminus L) \cup \overline{L}$.

Hardness. The reduction from SAT is as follows: to $\psi \in FML_{Var}$, we associate in polynomial time the formulas $\varphi = \top$ and

$$\mu = \Big(\bigvee_{l \in Lit(\psi) \mid l \text{ is a positive literal}} \bar{l} \vee y\Big) \wedge \Big(\bigvee_{l \in Lit(\psi) \mid l \text{ is a negative literal}} \bar{l} \vee z\Big)$$

where y and z are fresh variables not occurring in ψ. Let $\omega = \bigwedge Var$. By construction, we have $RelLit(\psi) \subseteq \overline{RelLit(\mu)}$. We have that ψ is consistent if and only if $\omega \models \psi \diamond_{Dep}^{\varphi} \mu$ holds; indeed, $\exists_l \overline{RelLit(\mu)}.\psi$ is valid if ψ is consistent, and $\exists_l \overline{RelLit(\mu)}.\psi$ is inconsistent otherwise. □

There are special cases where NP-hardness does not hold. Suppose, for instance, *Dep* is both monotone and antimonotone. Then $Dep(\mu) = \emptyset$, and $\psi \diamond_{Dep}^{\varphi} \mu = \mu \wedge \varphi \wedge \exists_l \overline{Dep(\mu)}.\psi$, which is equivalent to $\mu \wedge \varphi \wedge \psi$—, and $\omega \models \psi \diamond_{Dep}^{\varphi} \mu$ iff $\omega \models \mu \wedge \psi \wedge \varphi$, which is the model checking problem in propositional logic (which is well-known to be decidable in deterministic polynomial time).

Moreover, it could be the case that hardness is due only to the fact that the knowledge base ψ may be inconsistent. If the knowledge based is required to be consistent, which is a reasonable assumption, then this hardness result no longer holds. Consider for instance the full dependence function defined by $Dep(\mu) = Lit_{Var}$ for any μ. As we have seen in Proposition 3.15, if ψ is consistent then $\psi \diamond_{Dep} \mu \equiv \mu$ and $\psi \diamond_{Dep} \mu \equiv \bot$ otherwise. Note that *Dep* is standard. Clearly, if ψ is required to be consistent then $\psi \diamond_{Dep_v} \mu \equiv \mu$ and model checking is obviously polynomial in this case.

Tractable cases can be obtained by restricting the input so that $\exists_l \overline{Dep(\mu)}.\psi$ can be computed in time polynomial in the input size. Such restrictions are discussed in Section 5.4.

5.2. The Consistency Problem

We now investigate the complexity of the consistency problem for an updated base when operators based on formula/literal dependence are used. The problem is as follows: given three formulas φ, ψ, μ how difficult is it to determine whether $\psi \diamond_{Dep}^{\varphi} \mu$ is consistent?

PROPOSITION 5.2. *Let Dep be a formula/literal dependence function such that for any $l \in Lit_{Var}$ and $\mu \in FML_{Var}$, determining whether $l \in Dep(\mu)$ is in* NP. *Given three formulas φ, ψ, and μ, determining whether $\psi \diamond_{Dep}^{\varphi} \mu$ is consistent is* NP*-complete, even if φ is valid (no integrity constraints) and the input formula μ is an atom.*

PROOF. *Membership.* It is close to the membership proof of Proposition 5.1. The consistency problem can be solved in nondeterministic polynomial time, thanks to the following algorithm: let $V = Var(\psi) \cup Var(\mu) \cup Var(\varphi)$; guess two worlds ω, ω' over V, a set of literals L of $V \cup \overline{V}$ and a string c (both of them being of polynomial size if the input size), then check in polynomial time that $L \subseteq Dep(\mu)$ using c, and check that $\omega \models \psi$, $\omega' \models \varphi \wedge \mu$ and $\omega' = (\omega' \setminus L) \cup \overline{L}$.

Hardness. The reduction from SAT is as follows: to $\psi \in FML_{Var}$, we associate in polynomial time the formulas $\varphi = \top$ and $\mu = y$ where y is a fresh variable not occurring in ψ. Then $\psi \diamond_{Dep}^{\varphi} \mu \equiv y \wedge \varphi \wedge \exists_l \overline{Dep(\mu)}.\psi$ is consistent iff $\exists_l \overline{Dep(\mu)}.\psi$ is consistent (the right-to-left direction holds just because y is fresh) iff ψ is consistent. (Observe that for any $L \subseteq Lit_{Var}$, $\exists_l L.\psi$ is consistent iff ψ is consistent.) □

Once again, NP-hardness is due to the fact that ψ may be inconsistent. Indeed, consider the update operator $\diamond_{Dep}$ that is induced by the full dependence function ($Dep_f(\mu) = Lit_{Var}$ for every μ), which is both standard and constant. As we have seen in Proposition 3.15, $\psi \diamond_{Dep} \mu$ is equivalent to μ if ψ is consistent, and $\bot$ otherwise. Hence if ψ is consistent then $\psi \diamond_{Dep_v} \mu$ is equivalent to μ, which is always consistent when μ is a variable.

5.3. The Inference Problem

In this section, we investigate the complexity of the inference problem (or query entailment problem) from an updated base when operators based on formula/literal dependence are used. The problem is as follows: given four formulas φ, ψ, μ and γ, how difficult is it to determine whether $\psi \diamond^{\varphi}_{Dep} \mu \models \gamma$ holds? We first identify the complexity in the general case, then focus on some specific cases obtained by imposing further restrictions on some parts of the input.

Lang et al. [2003a] showed that determining whether ψ is Lit-dependent of l is an NP-complete task. On this ground, one could suspect that the query entailment problem for operators based on *RelLit* is beyond coNP. That is not the case: under reasonable assumptions on *Dep*, inference from an updated base when operators based on formula/literal dependence are used remains in coNP (this result slightly extends Theorem 15 from Liberatore [2000a], which focuses on the MPMA case).

PROPOSITION 5.3. *Let Dep be a formula/literal dependence function such that for any $l \in Lit_{Var}$ and $\mu \in FML_{Var}$, determining whether $l \in Dep(\mu)$ is in* NP. *Given four formulas φ, ψ, μ and γ, determining whether $\psi \diamond^{\varphi}_{Dep} \mu \models \gamma$ holds is* coNP*-complete. Hardness is still the case when φ is valid (no integrity constraints) and the input formula μ and the query γ are atoms.*

PROOF. *Membership.* If determining whether $l \in Dep(\mu)$ is in NP, then determining whether any given set $L \subseteq Lit_{Var}$ of literals is included in $Dep(\mu)$ is in NP as well. Hence for every positive instance $\langle L, \mu \rangle$ of this decision problem there exists a certificate c of size polynomial in the input size such that verifying that $\langle L, \mu \rangle$ is a positive instance can be achieved in (deterministic) polynomial when c is given. Now, the complementary problem to query entailment can be solved in nondeterministic polynomial time, thanks to the following algorithm. Let $V = Var(\varphi) \cup Var(\psi) \cup Var(\mu) \cup Var(\gamma)$; guess two worlds ω and ω' over V, a subset L of $V \cup \overline{V}$ and a string c (both of them being of polynomial size if the input size), then check in polynomial time that $L \subseteq Dep(\mu)$ using c, and check that $\omega \models \psi$, $\omega' \models \varphi \wedge \mu \wedge \neg\gamma$ and $\omega = (\omega' \setminus L) \cup \overline{L}$.

Hardness. The reduction from UNSAT is as follows: to $\psi \in FML_{Var}$ we associate in polynomial time the formulas $\varphi = \top$ and $\mu = y$, $\gamma = z$ where y and z are two fresh variables not occurring in ψ. Whatever *Dep* is, ψ is inconsistent if and only if $\psi \diamond^{\varphi}_{Dep} \mu \models \gamma$; this is because for every $L \subseteq Lit_{Var}$, $\exists_l L.\psi$ is inconsistent if and only if ψ is inconsistent. □

Lemma 4.7 is a valuable property from a computational point of view when CNF queries are considered since it shows that Lit-separability is a sufficient condition for separability in Levesque's sense. Indeed, thanks to Lemma 4.7, if there are no integrity constraints (i.e., $\varphi \equiv \top$) and if the dependence function is standard then inference from an updated base can be achieved by replacing one global entailment test by two simpler entailment tests, and exponential savings may be achieved this way in practice.

Now, clausal entailment from $\exists_l \overline{Dep(\mu)}.\psi$ can be reduced in linear time to clausal entailment from a classical formula; this is achieved without turning $\exists_l \overline{Dep(\mu)}.\psi$ first into such a formula (by means of the definition of $\exists_l$).

PROPOSITION 5.4. *Let $L \subseteq Lit_{Var}$, ψ a formula and γ a non-tautological clause. $\exists_l L.\psi \models \gamma$ if and only if $\psi \models \bigvee_{l \in Lit(\gamma) \setminus L} l$.*

PROOF. ($\Rightarrow$) If $\exists_l L.\psi \models \gamma$ holds then since $\exists_l L.\psi$ is Lit-independent of every literal of L, we have $\exists_l L.\psi \models \bigvee_{l \in Lit(\gamma) \setminus L} l$. Since $\psi \models \exists_l L.\psi$ also holds, we get that $\psi \models \bigvee_{l \in Lit(\gamma) \setminus L} l$.

($\Leftarrow$) Assume that $\psi \models \bigvee_{l \in Lit(\gamma) \setminus L} l$. Since the clause $\bigvee_{l \in Lit(\gamma) \setminus L} l$ is Lit-independent of every literal of L, Corollary 4 from Lang et al. [2003a] tells us that $\psi \models \bigvee_{l \in Lit(\gamma) \setminus L} l$ iff $\exists_l L.\psi \models \bigvee_{l \in Lit(\gamma) \setminus L} l$. Since $\bigvee_{l \in Lit(\gamma) \setminus L} l \models \gamma$ holds, we obtain the expected conclusion. □

Altogether, the two previous propositions give us some tractable restrictions for clausal query entailment from an updated belief base. Let us say that a subset $\mathcal{C}$ of FML_{Var} is a *tractable fragment for clausal entailment* if and only if there exists a polytime algorithm for deciding whether $\psi \models \gamma$ holds for any formula ψ from $\mathcal{C}$ and any CNF formula γ (see Darwiche and Marquis [2002]). Many such fragments exist in the literature: the DNNF fragment (and its subsets DNF and $\text{OBDD}_<$), the Horn CNF fragment, the Krom fragment (conjunctions of binary clauses), the Blake fragment (formulas in prime implicates form), etc. We have the following corollary to the two previous propositions.

COROLLARY 5.5. *Let Dep be a standard dependence function. Let μ be a CNF formula from a tractable fragment for clausal entailment. Suppose also that determining whether any given literal l belongs to $Dep(\mu)$ can be achieved in polynomial time. Let ψ belong to a tractable fragment for clausal entailment. Then determining whether $\psi \diamond^{\top}_{Dep} \mu \models \gamma$ can be achieved in polynomial time for every CNF formula γ.*

The conditions provided in this corollary hold for instance when $Dep = RelLit$, μ is a CNF formula from a tractable fragment for clausal entailment, which is stable for variable instantiation (see Proposition 11 in Lang et al. [2003a]), and ψ belongs to a tractable fragment for clausal entailment. To make things more concrete, say that ψ is a Horn CNF formula and μ is a Krom formula (a conjunction of binary clauses), or vice-versa. It is interesting to observe that in such a case, clausal entailment from the updated base $\psi \diamond^{\top}_{Dep} \mu$ is in P, thus computationally easier than clausal entailment from the corresponding expanded base $\psi \wedge \mu$, under the assumption $\mathsf{P} \neq \mathsf{NP}$. Indeed, determining whether $\psi \wedge \mu \models \gamma$ is coNP-complete in this situation – tractable classes are known not to mix well.

5.4. Compactability Issues

We finally consider the compactability problem for updated bases when operators based on formula/literal dependence are used. In a nutshell, the problem is as follows: given three formulas φ, ψ, and μ, does there exist a propositional formula equivalent to $\psi \diamond^{\varphi}_{Dep} \mu$ whose size is polynomial in the input size? Stated otherwise, is the propositional language FML_{Var} enriched with $\diamond_{Dep}$ (viewed as a connective) as succinct as FML_{Var}, or is it strictly more succinct? Clearly enough, the naive rewriting of

$\mu \wedge \varphi \wedge \exists_l \overline{Dep(\mu)}.\psi$ into a propositional formula according to the definition of $\exists_l$, does not lead to a formula of size polynomial in the input size in the general case.

Formally, two versions of the compactability problem can be taken into account, depending on which notion of equivalence we choose Cadoli et al. [1999].

— *Logical compactability*. $\diamond_{Dep}$ is logically compactable if and only if there exists a polynomial p such that for every triple of propositional formulas φ, μ, ψ there exists a propositional formula Σ such that:
  (1) $|\Sigma| \leq p(|\varphi| + |\mu| + |\psi|)$, and
  (2) Σ is logically equivalent to $\psi \diamond_{Dep}^{\varphi} \mu$.

— *Query compactability*. $\diamond_{Dep}$ is query-compactable if and only if there exists a polynomial p such that for every triple of propositional formulas φ, μ, ψ there exists a propositional formula Σ such that:
  (1) $|\Sigma| \leq p(|\varphi| + |\mu| + |\psi|)$, and
  (2) Σ is query-equivalent to $\psi \diamond_{Dep}^{\varphi} \mu$, i.e., Σ and $\psi \diamond_{Dep}^{\varphi} \mu$ have the same logical consequences over $Var(\varphi) \cup Var(\mu) \cup Var(\psi)$.

Let us first consider the query-compactability issue.

PROPOSITION 5.6. *Let Dep be a formula/literal dependence function. Then $\diamond_{Dep}$ is query-compactable.*

PROOF. For any $L \subseteq Lit_{Var}$ and any formula ψ, let ψ'_L be the formula obtained by replacing in $NNF(\psi)$ every occurrence of $l \in L$ by an occurrence of $l' \in L'$, where symbols from $L' \subseteq Lit_{Var}$ are fresh, i.e., $\{x' \mid x' \in L' \text{ or } \neg x' \in L'\} \cap Var(\psi) = \emptyset$. (For example, if $\psi = a \vee (\neg a \wedge \neg b)$ and $L = \{a, \neg a, b\}$ then $\psi'_L = a' \vee (\neg a' \wedge \neg b)$.) Let us consider the formula $\Sigma = \varphi \wedge \mu \wedge \psi'_L$ where $L = \overline{Dep(\mu)} \cap Lit(\psi)$. By construction, this formula is of size polynomial in $|\varphi| + |\mu| + |\psi|$. Let us show that $\psi \diamond_{Dep}^{\varphi} \mu$ is query-equivalent to Σ. Let $V = Var(\varphi) \cup Var(\mu) \cup Var(\psi)$. We have to show that $\exists \overline{V}.(\psi \diamond_{Dep}^{\varphi} \mu) \equiv \exists \overline{V}.\Sigma$. The formula $\exists \overline{V}.(\psi \diamond_{Dep}^{\varphi} \mu) = \exists \overline{V}.(\mu \wedge \varphi \wedge \exists_l \overline{Dep(\mu) \cap Lit(\psi)}.\psi)$ is equivalent to $\mu \wedge \varphi \wedge \exists_l \overline{Dep(\mu) \cap Lit(\psi)}.\psi$ since every variable from $Var(\mu \wedge \varphi \wedge \exists_l \overline{Dep(\mu) \cap Lit(\psi)}.\psi)$ is in V. Now, since every variable from $\mu \wedge \varphi$ also is in V, we have that $\exists \overline{V}.(\mu \wedge \varphi \wedge \psi'_L)$ is equivalent to $\mu \wedge \varphi \wedge \exists \overline{V}.\psi'_L$, which is also equivalent to $\mu \wedge \varphi \wedge \exists_l \overline{V \cup \overline{V}}.\psi'_L$. Since every literal of $Lit(\psi'_L)$ is in $V \cup \overline{V}$, except the literals from L', $\exists \overline{V}.\Sigma$ is equivalent to $\mu \wedge \varphi \wedge \exists_l L'.\psi'_L$. But $\exists_l L'.\psi'_L$ is equivalent to $\exists_l L.\psi$ (quantified literals are 'dummy literals') and the substitution metatheorem for propositional logic concludes the proof. □

There is no such a general result for $\diamond_{Dep}$ operators w.r.t. logical compactability. Intuitively, some $\diamond_{Dep}$ operators are logically compactable because they lead to forget no literal (except a preset number of them) or to forget every literal (except a preset number of them); for instance, the update operators induced by $Dep(\mu) = \emptyset$ and $Dep(\mu) = Lit_{Var}$ for every μ are logically compactable. Some other operators are not logically compactable. We have the following result, which slightly extends Theorem 10 from Liberatore and Schaerf [2004].

PROPOSITION 5.7. *Let Dep be any standard and confined formula/literal dependence function. Then $\diamond_{Dep}$ is not logically compactable unless the polynomial hierarchy collapses.*

PROOF. First of all, it is obvious that the (non-standard) third parameter φ (the integrity constraints) of $\diamond_{Dep}$ operators has no influence at all on the logical

compactability issue of the operator: $\diamond_{Dep}$ is logically compactable if and only if the standard (i.e., with two parameters) update operator $\diamond^{\top}_{Dep}$ is logically compactable. Accordingly, we now assume that $\varphi = \top$. Let $3\mathrm{CNF}_n$ be the set of all propositional CNF formulas ψ, such that each clause of ψ contains at most 3 literals, and $Var(\psi)$ contains n elements. We take advantage of Theorem 2.3 from [Cadoli et al. 1999]. This theorem says that if $\diamond$ is a belief change operator and if there exists a polynomial p such that for every integer $n > 0$, there exists a pair of formulas ψ_n and μ_n such that

— $|\psi_n| + |\mu_n| \leq p(n)$, and
— for every formula $\pi \in 3\mathrm{CNF}_n$, there exists a world ω_π over $Var(\psi_n) \cup Var(\mu_n)$ that can be computed from π in polynomial time and which is such that π is consistent if and only if $\omega_\pi \models \psi_n \diamond \mu_n$.

Then if $\diamond$ is query-compactable, then NP $\subseteq$ P/poly. Let us consider the set $X_n = \{x_1, \ldots, x_n\}$ of variables from Var. Let π_n^{max} be the conjunction of all clauses containing at most three literals built up from X_n. π_n^{max} has a size polynomial in n since it contains $\mathcal{O}(n^3)$ clauses. Furthermore, viewing CNF formulas as sets of clauses, each $\pi \in 3\mathrm{CNF}_n$ is (up to a variable renaming) a subset of π_n^{max}. To every clause γ_i of π_n^{max} we associate a new symbol c_i not occurring in X_n. We now define ψ_n, μ_n and ω_π as follows:

— $\psi_n = (\bigwedge_{\gamma_i \in \pi_n^{max}} (\neg c_i \vee \gamma_i)) \vee \neg t$ where t is a new symbol (not in X_n and different of any c_i);
— $\mu_n = ((\bigvee_{x_i \in X_n} \neg x_i) \vee y) \wedge ((\bigvee_{x_i \in X_n} x_i) \vee z)$ where y and z are two new symbols (not in X_n and different of any c_i);
— ω_π is such that for every $x \in Var(\psi_n) \cup Var(\mu_n)$, $\omega_\pi(x) = 0$ if $x = c_i$ and $\gamma_i \notin \pi$, and $\omega_\pi(x) = 1$ otherwise.

If π is consistent, then it has a model ω' over X_n; let us extend ω' to $Var(\psi_n) \cup Var(\mu_n)$, by imposing that $x \notin \omega'$ if $x = c_i$ and $\gamma_i \notin \pi$, and $x \in \omega'$ otherwise. By construction, we have $\omega' \models \psi$. Now, let $L = \{x_i \in X_n \mid x_i \in \omega'\} \cup \{\neg x_i \mid x_i \in X_n \text{ and } x_i \notin \omega'\}$. We have $L \subseteq X_n \cup \overline{X_n}$. Now, we also have $X_n \cup \overline{X_n} \subseteq \overline{Dep(\mu)}$ since $RelLit(\mu) = X_n \cup \overline{X_n} \cup \{y, z\}$. As a consequence, we get $L \subseteq \overline{Dep(\mu)}$. Since $\omega' = (\omega_\pi \setminus \overline{L}) \cup L$, we get that $\omega_\pi \models \psi_n \diamond^{\varphi}_{Dep} \mu_n$.

If π is inconsistent, then assume that $\omega_\pi \models \psi_n \diamond^{\varphi}_{Dep} \mu_n$. Then there exists a world ω' over $Var(\psi_n) \cup Var(\mu_n)$ and a subset L of $\overline{Dep(\mu)}$ such that $\omega' = (\omega_\pi \setminus \overline{L}) \cup L$. Since $c_i \notin \overline{Dep(\mu)}$ for every i, we have $c_i \in \omega'$ iff $c_i \in \omega_\pi$. Since π is inconsistent, we get that $\omega' \not\models \bigwedge_{\gamma_i \in \pi_n^{max}} (\neg c_i \vee \gamma_i)$. Therefore, we have $t \notin \omega'$. But $\neg t \notin \overline{Dep(\mu)}$, so we should also have $t \notin \omega_\pi$, contradiction. □

Nevertheless, tractable cases can be achieved by restricting the input so that a formula logically equivalent to $\exists_l \overline{Dep(\mu)}.\psi$ and of size polynomial in the input size can be computed.

PROPOSITION 5.8. *If ψ is a Blake formula or a DNNF formula, then $\psi \diamond^{\varphi}_{Dep} \mu$ can be turned into an equivalent propositional formula of polynomial size. Furthermore, if Dep is a polytime function, then such a formula can be computed in polynomial time.*

PROOF. Clearly enough, since *Dep* is a computable function, $\overline{Dep(\mu)}$ can be computed in finite time (resp. in polynomial time if *Dep* is a polytime function).

Blake. It is then enough to take advantage of Proposition 19 from Lang et al. [2003a] showing that a prime implicate formula equivalent to $\exists_l \overline{Dep(\mu)}.\psi$ can be computed in

polynomial time as the conjunction of all prime implicates of ψ that does not contain any literal from $\overline{Dep(\mu)}$.

DNNF. It is necessary to generalize Theorem 9 from [Darwiche 2001] to literal forgetting. Let us note $DNNF(\exists_l L.\psi)$ where $L \subseteq Lit_{Var}$ and ψ is a DNNF formula the propositional formula defined by structural induction as follows:

— if ψ is a Boolean constant or a literal not belonging to L, then $DNNF(\exists_l L.\psi) = \psi$;
— if ψ is a literal from L, then $DNNF(\exists_l L.\psi) = \top$;
— if ψ is a conjunction $\bigwedge_{i=1}^n \alpha_i$, then $DNNF(\exists_l L.\psi) = \bigwedge_{i=1}^n DNNF(\exists_l L.\alpha_i)$;
— if ψ is a disjunction $\bigvee_{i=1}^n \alpha_i$, then $DNNF(\exists_l L.\psi) = \bigvee_{i=1}^n DNNF(\exists_l L.\alpha_i)$.

Clearly enough, $DNNF(\exists_l L.\psi)$ can be computed in linear time. It remains to show that $\exists_l.\psi \equiv DNNF(\exists_l L.\psi)$. We do so by structural induction on ψ.

Base. If ψ is a Boolean constant or a literal not belonging to L, then $DNNF(\exists_l L.\psi) = \psi$ is equivalent to $\exists_l L.\psi$ since in such a case ψ is Lit-independent of L. If ψ is a literal from L, then $DNNF(\exists_l L.\psi) = \top$ is equivalent to $\exists_l L.\psi$ (just consider the definition of $\exists_l L.\psi$).

Inductive step. If ψ is a conjunction $\bigwedge_{i=1}^n \alpha_i$, then

$$DNNF(\exists_l L.\psi) = \bigwedge_{i=1}^{n} DNNF(\exists_l L.\alpha_i).$$

By induction hypothesis each $DNNF(\exists_l L.\alpha_i)$ is equivalent to the corresponding $\exists_l L.\alpha_i$. From Item 2. of Corollary 2 in Lang et al. [2003a] we know that for each $i \in \{1,\ldots,n\}$, $\exists_l L.\bigwedge_{i=1}^n \alpha_i \models \exists_l L.\alpha_i$ holds. Hence we have $\exists_l L.\bigwedge_{i=1}^n \alpha_i \models \bigwedge_{i=1}^n (\exists_l L.\alpha_i)$. It remains to show that $\bigwedge_{i=1}^n (\exists_l L.\alpha_i) \models \exists_l L.\bigwedge_{i=1}^n \alpha_i$. Let ω be a model of $\bigwedge_{i=1}^n (\exists_l L.\alpha_i)$ on $Var(\psi)$. Then ω is a model of $\exists_l L.\alpha_i$ on $Var(\psi)$ for each $i \in \{1,\ldots,n\}$. By Proposition 15 from Lang et al. [2003a], for each $i \in 1\ldots n$, there exists a subset L_i of $Var(\alpha_i) \cup \overline{Var(\alpha_i)}$ and a model ω_i of α_i on $Var(\alpha_i)$ such that $\omega_i = (\omega \setminus \overline{L_i}) \cup L_i$. Now, since $\psi = \bigwedge_{i=1}^n \alpha_i$ is a DNNF formula, the sets $Var(\alpha_i)$ ($i \in \{1,\ldots,n\}$) are pairwise disjoint. Hence the world ω' on $Var(\psi)$ defined by: for all $x \in Var$, if $x \in Var(\alpha_i)$ then $x \in \omega'$ iff $x \in \omega$, is a model of ψ. Furthermore, the L_i's are also pairwise disjoint. Hence, we have $\omega' = (\omega \setminus \overline{\bigcup_{i=1}^n L_i)) \cup \bigcup_{i=1}^n L_i}$. Since $\bigcup_{i=1}^n L_i \subseteq L$, we get that ω is a model of $\exists_l L.\psi$ and the result follows. Finally, if ψ is a disjunction $\bigvee_{i=1}^n \alpha_i$, then $DNNF(\exists_l L.\psi) = \bigvee_{i=1}^n DNNF(\exists_l L.\alpha_i)$ is equivalent to $\exists_l L.\psi$: this follows directly from Proposition 17 from Lang et al. [2003a] and the induction assumptions stating that each $DNNF(\exists_l L.\alpha_i)$ is equivalent to the corresponding $\exists_l L.\alpha_i$. □

6. DEPENDENCE-BASED UPDATE AND ACTION PROGRESSION

It is well-known that belief update and action progression are highly related (see an extended discussion in Lang [2007]). Especially, since belief update can be used as a language for specifying action effects, it can be used for reasoning about action and planning. We start by reviewing some recent works that go in this direction.

Liberatore [2000b] presents a framework for reasoning about actions, whose semantics can be seen as a proper restriction of the one pointed out in Baral et al. [1997], but which remains expressive enough to incorporate many update settings, including Winslett's PMA. In this framework, minimization of change is achieved through the minimization of `happens` statements (a primitive allowing to declare that a given action occurs at a given time point). Only deterministic actions are formalized in this

framework, making it inappropriate for modeling nondeterministic updates (like the one happening when tossing a coin). This strongly contrasts with our approach.

Amir and Russell [2003] define a *logical filtering* operation that takes a belief state represented compactly by a propositional formula and consider actions with conditional effects of the form `if` α `then` a `causes` φ, where α and φ are propositional formulas. This filtering operator corresponds to WSS [Winslett 1990] enriched with conditional updates. Then they go further and address the practical computation of this conditional update operator, depending on the syntax of the input belief base. Most of the results of Amir and Russell [2003] can be easily generalized to our update operators based on formula/literal dependence, thus completing nicely our results in Section 5. A first-order extension of logical filtering has also been proposed by Shirazi and Amir [2011]. In such a general setting, the update issue is much more difficult (especially, progressing a belief state is known to be not first-order definable [Lin and Reiter 1997]). In [Shirazi and Amir 2011], the authors present some restrictions on the belief state representations and on the action representations enabling some tractable algorithms for logical filtering to be designed.

Lang [2007] characterizes the class of feedback-free actions for which progression corresponds to a belief update, and shows that two well-known forms of action regression correspond in a natural way to two forms of "reverse update".

Now, let us investigate deeper the respective position of belief update and propositional action languages [Baral and Gelfond 2005; Giunchiglia and Lifschitz 1998; Giunchiglia et al. 2004; McCain and Turner 1997]. Both frameworks are based on propositional logic and aim at expressing compactly action effects and reasoning about them. How do they compare in terms of expressivity? What is the position of dependency-based update in this zoo of propositional action representation languages?

Two remarks come immediately to mind.

— "Update" actions are specific actions with only one effect – in other terms, updating by α corresponds to progressing by the action "make α true" – and this effect α can be any propositional formula; on the other hand, action languages consider only 'literal' effects of the form a `causes` l, where l is a literal.
— Action languages contain many constructs for "combining" action effects: conditional effects (`if` γ `then` a `causes` l), concurrent actions, as well as static causal rules (for ramifications). They can also express some forms of nondeterminism, using their nonmonotonic semantics in a clever way, or by the `release` construct, which nondeterministically reassigns the truth value of a variable.

A first conclusion emerging from these observations is that the expressivity of belief update comes from its handling of complex effects (expressed by arbitrary propositional formulas) while the expressivity of action languages comes from its ability to combine action effects. On the latter point, it is not difficult to enrich belief update so as to integrate these constructs; the second half of [Herzig et al. 2001] shows how update operators can be extended so as to deal with more complex change formulas (so-called "extended inputs" Φ).

— Conditional effects. If γ then update ψ by α else update ψ by β, denoted by $\psi \diamond \Phi$ with $\Phi =$ `if` γ `then` α `else` β.
— Nondeterminism. Update ψ by α or update ψ by β, denoted by $\psi \diamond \Phi$ with $\Phi = \alpha \cup \beta$.
— Concurrency. Update ψ concurrently by α and by β, denoted by $\psi \diamond \Phi$ with $\Phi = \alpha \mid\mid \beta$.

An interesting feature of such extended updates is that they do not imply any complexity shift with respect to the corresponding basic updates for the inference problem (especially, when dependence-based updates operators are considered, the inference problem remains coNP-complete).

We focus first on conditional and concurrent updates. If we consider the restriction of $\diamond_{Dep}$ to the case when the input formula is a literal, then when $Dep = RelLit$, we have $Dep(l) = \{l\}$, and therefore, $\omega \diamond_{Dep} l = (\omega \setminus \{\bar{l}\}) \cup \{l\}$, i.e., $\omega \diamond_{Dep} l$ is the model obtained from ω by forcing l to be true, which gives the same truth value as ω to all variables except $Var(l)$. Consider now an update by the extended input $\Phi = (\text{if } C_1 \text{ then } l_1) \,||\, \ldots \,||\, (\text{if } C_1 \text{ then } l_n)$, and let $\Gamma^+(l) = \bigvee\{C_i | l_i = l\}$. Then we have

$$\forall x \in VAR,\ \omega \diamond_{Dep} \Phi \models x \text{ iff } \omega \models (x \wedge \neg\Gamma^+(\neg x)) \vee \Gamma^+(x).$$

We recover here a well-known concept in the literature on reasoning about action: a *successor state axiom*. This result shows how $\diamond_{Dep}$ can be positioned with respect to propositional action languages: both conditional update and progression in action languages coincide when conditional update is restricted to literal inputs (but allows for conditional and concurrent updates), and actions languages are restricted to dynamic effects (no static causal effects).

We focus now on nondeterminism. We see from our discussion above that there are two ways of expressing nondeterministic effects: either by using disjunctive effects (make $a \vee b$ true) or by using nondeterministic effect trigerring (make a true or make b true). This dichotomy in the expression of nondeterminism has been discussed first in Brewka and Hertzberg [1993]. Some nondeterministic actions, such as tossing a coin, can be expressed easily in both ways (`release p`, equivalent to a nondeterministic update by $p \cup \neg p$; and a dependency-based update by $p \vee \neg p$ with $Dep(p \vee \neg p) = \{p, \neg p\}$). However, some actions are more succinctly expressed using one of these two ways; for instance, making an equivalence true ("update by $(p \Rightarrow q) \wedge (q \Rightarrow p)$") can be expressed in action languages but in a less intuitive and less compact way (for instance, by the four static causal rules p `causes` q; q `causes` p; $\neg p$ `causes` $\neg q$; $\neg q$ `causes` $\neg p$), which demonstrates that allowing for complex effects gives much more flexibility than allowing just literals.

Moreover, identifying the literals on which the direct effect of an action may have a negative influence (including the negations of those on which it depends as in our update operators) appears also as a key to discriminate a disjunctive effect from a nondeterministic effect. Indeed, frame axioms should not be considered for literals occurring in nondeterministic actions like "tossing a coin": $head \diamond_{Dep} head \cup \neg head$ is expected to be equivalent to $\top$, which can be achieved only if $head$ is released. Such a distinction is salient in the completions of the generalized action theories proposed by Lang et al. [2003b], which are based on the concept of causal explanation [Giunchiglia et al. 2004; McCain and Turner 1997], just as many action languages proposed so far, especially $\mathcal{C}$ [Giunchiglia and Lifschitz 1998].

7. RELATED WORK

Herzig and Rifi [1999] gave a comprehensive study of belief update, which exhaustively surveyed all works on belief update up to that date. To our knowledge, no survey of this kind has been published since then, and we believe that it is the right time to do so. We now briefly review a number of papers about belief update, published after Herzig and Rifi [1999] and, when this is relevant, discuss their connections to our dependence-based operators.

7.1. Dependence-Based Update

We start by listing a series of works which address various forms of belief update based on formula/variable dependence. Early works [Doherty et al. 1998; Herzig 1996; Herzig and Rifi 1999; Winslett 1990] have been already discussed in Section 3 and in [Herzig and Rifi 1999]. Since then, many works have studied, under various points of

view, belief update operators based on formula/variable or, more rarely, formula/literal dependence. We review them below.

Vo et al. [2006] introduce a belief update operator where the syntax of the input matters, but the syntax of the initial knowledge base (a propositional formula) does not. More precisely, it consists in applying the following operations in sequence:

(1) rewrite the knowledge base φ to a set of clauses C;
(2) saturate C by resolution on the variables of $Var(\mu)$;
(3) eliminate from C the clauses containing variables from $Var(\mu)$;
(4) expand C by μ.

For instance, if $\varphi = (p \vee q) \wedge (\neg p \vee r)$ and $\mu = p$ then $C = \{p \vee q, \neg p \vee r\}$; after step 2 we have $C = \{p \vee q, \neg p \vee r, q \vee r\}$; after step 3 we have $C = \{q \vee r\}$; and finally, the result of the update is equivalent to $(q \vee r) \wedge p$. It appears that, although defined in a different manner, this operator is identical to WSS; it provides thus an alternative, syntactic, characterization of WSS.

Lang et al. [2001] extend dependence-based update operators to epistemic states, allowing to express various degrees of beliefs in the initial base as well as action effects with various degrees of plausibility. In this article, epistemic states are modeled as ordinal conditional functions and represented as propositional stratified belief bases. While the authors focus on formula/variable dependence functions, it would be easy to extend their work to formula/literal dependence functions.

Madalińska-Bugaj and Łukaszewicz [2009] propose a first-order version of the MPMA, where the knowledge base is a closed first-order formula and the input formula is a Boolean combination of ground atoms. Allowing first-order knowledge bases and input formulas gives much more expressivity to the language—and, on the other hand, leads to the computational difficulties inherent to first-order logic. Clearly, since dependence-based update generalizes the MPMA, we could build a general framework for first-order dependence-based update, which would of course generalize the approach given in Madalinska-Bugaj and Łukaszewicz [2009].

7.2. General Families of Update Operators

We now review some works that define general families of update operators which include some dependence-based operators.

Delgrande et al. [2005] define *Gricean update operators* as follows. Given $U \subseteq Var$, a U-world is defined as a partial interpretation (also viewed as a term) associating each variable of U with a truth value. Given a formula μ, let $Mod_{RelVar(\mu)}(\mu)$ be the set of all $RelVar(\mu)$-worlds implying μ. Let $O = \{\preceq_\omega |\ \omega \in 2^{Var}\}$ be a collection of so-called strongly centered partial preorders over the worlds (called faithful partial preorders in [Katsuno and Mendelzon 1991]), that is, partial preorders such that $\omega \prec_\omega \omega'$ for all ω and $\omega' \neq \omega$. Then the Gricean update operator induced from O is defined by:

$$Mod(\varphi \mathbin{\hat{\diamond}_O} \mu) = \bigcup_{\omega' \in Mod(\varphi)} \bigcup_{\sigma \in Mod_{RelVar(\mu)}(\mu)} \min_{\preceq_{\omega'}}(Mod(\sigma)).$$

The very essence of Gricean updates is that after updating a consistent φ by a consistent μ, what we know about $RelVar(\mu)$ is exactly μ [Delgrande et al. 2005, Theorem 4.11]. Let us call this key condition "the Gricean property" **(GR)**. Using propositional quantification the Gricean property can be written more formally as:

(GR) $\exists (Var \backslash RelVar(\mu)).(\varphi \mathbin{\hat{\diamond}_O} \mu) \equiv \mu$.

From this we can conclude that WSS$\downarrow$ satisfies **(GR)** while WSS does not. In the general case, update operators based on formula/literal dependence are not Gricean

either, since, for instance, updating $a \wedge b$ by $\mu = a \vee b$ using $\diamond_{Dep}$ with $Dep = RelLit$ leads to a formula equivalent to $a \wedge b$, which tells more about $\{a, b\}$ than μ. The following result characterizes those update operators based on formula/literal dependence that satisfy the Gricean property:

PROPOSITION 7.1. *$\diamond_{Dep}$ satisfies the Gricean property* **(GR)** *if and only if Dep satisfies the following property.*

(DG) *for every consistent formula μ, $Dep(\mu)$ contains both $RelVar(\mu)$ and $\overline{RelVar(\mu)}$.*

PROOF. Assume *Dep* satisfies **(DG)**. Then for any consistent μ and φ, $RelVar(\exists_l \overline{Dep(\mu)}.\varphi) \cap RelVar(\mu) = \emptyset$, therefore $\varphi \diamond_{Dep} \mu \equiv \psi \wedge \mu$ for some ψ such that $RelVar(\psi) \cap RelVar(\mu) = \emptyset$, which implies that $\diamond_{Dep}$ satisfies the Gricean property.

Conversely, assume that $\diamond_{Dep}$ satisfies the Gricean property and let μ be a consistent formula and l a literal such that $Var(l) \in RelVar(\mu)$ and $l \notin Dep(\mu)$. We have $\neg l \diamond_{Dep} \mu = (\exists_l \overline{Dep(\mu)}.\neg l) \wedge \mu \equiv \neg l \wedge \mu$. Because $\diamond_{Dep}$ satisfies the Gricean property we must have $\exists Var \setminus RelVar(\mu).(\neg l \diamond_{Dep} \mu) \equiv \mu$. This, together with $Var(l) \in RelVar(\mu)$, implies that $\neg l \wedge \mu \equiv \mu$, hence $\mu \models \neg l$. Therefore, any literal l such that $Var(l) \in Var(\mu)$ and $\mu \not\models \neg l$ must be in $Dep(\mu)$. It remains to be shown that if $Var(l) \in RelVar(\mu)$ and $\mu \models \neg l$ then l must be in $Dep(\mu)$ as well. Towards a contradiction, suppose it is not. Let $\varphi = \neg l$. Since by assumption $l \notin Dep(\mu)$, we have $\exists_l \overline{Dep(\mu)}.\varphi \equiv \neg l$, hence $\varphi \diamond_{Dep} \mu = \neg l \wedge \mu \equiv \bot$, contradicting the assumption that $\diamond_{Dep}$ satisfies the Gricean property. □

Delgrande et al. [2008] define a *compositional update operator* $\diamond_C$ whose most distinctive feature is that update distributes over disjunction, that is, $\varphi \diamond_C (\mu_1 \vee \mu_2) \equiv (\varphi \diamond_C \mu_1) \vee (\varphi \diamond_C \mu_2)$. They also define a more general family of update operators, where compositional update is preceded by a syntactic transformation μ^* of the input formula μ, giving rise to a family of generalized compositional update operators $\diamond_C^*$; for instance, μ^* can be $NNF(\mu)$, or the conjunction of its prime implicates, or the disjunction of all conjunctions of literals corresponding to its models, or the disjunctions of all conjunctions of literals corresponding to its models restricted to the set of variables occurring in μ. In the latter case, $\diamond_C^*$ is shown to be equivalent to WSS. This shows that the class of generalized compositional updates and the class of dependence-based updates overlap. Characterizing their intersection precisely is an interesting open issue.

7.3. Update and Observations

Hunter and Delgrande [2011] propose a general framework for reasoning about action and change, where ontic actions are expressed by updates while observations are expressed by revisions. Especially, they define a new class of belief change operators, called belief evolution operators. A belief evolution operator takes two arguments: a set of states and an alternating sequence of actions and observations. Each belief evolution operator is defined with respect to a fixed update operator and a fixed AGM revision operator. The authors show that the interaction between update and revision can be complex, and specify how an agent should consider the history of actions when incorporating a new observation. In contrast, existing formalisms for reasoning about epistemic action effects either ignore the interaction between revision and update or they deal with it implicitly.

In the same vein, Shapiro et al. [2011] define a very expressive framework for reasoning about action and change, integrating revision and update, within the situation calculus. Generalizing belief update so as to take observations into account was also the key point of *generalized update* [Boutilier 1998]. Finally, *belief extrapolation* [Dupin de Saint-Cyr and Lang 2011] focuses on reasoning about time-stamped observations by minimizing change, but without any action progression component.

7.4. Belief Update Beyond Classical Logic

There have been also a few recent extensions of belief update beyond classical logic, that we review briefly now.

Liu et al. [2006] consider the update problem in description logics. One of their main results is that there are cases where, for a fixed logic used to express the knowledge base, the result of an instance level evolution operation, i.e., when only the Abox is modified, is not expressible in the language of the logic under consideration. In particular, they show that this is the case for the description logic $\mathcal{ALC}$. Contrastingly, de Giacomo et al. [2006] show how Winslett's PMA can be extended to the description logic DL-Lite in a way such that the updated base is still expressible in DL-Lite. In de Giacomo et al. [2007], the authors propose to address the problem of limited expressivity by resorting to a notion of maximal approximation of update and illustrate it by focusing on a restriction of the logic considered in the previous paper. Obviously enough, such expressiveness issues do not arise in the simple setting of classical propositional logic that we consider in this article. However, compactability issues (i.e., the ability of representing the updated base as a propositional formula using a polynomial amount of space) make sense for our update operators and we investigated them.

Eiter et al. [2005, 2010] address the problem of updating action descriptions represented in a fragment of the action language $\mathcal{C}$. The framework they design is quite general: the updated action description can be required to include some causal laws and to imply some given integrity constraints; moreover, a preference relation over action descriptions is also taken into account. Their approach has flavors of both belief revision and belief update.

Baral and Zhang [2005] consider the problem of updating a belief base expressed in the language of epistemic logic S5. They introduce an update operator based on the principle of minimal change and show that it satisfies the KM postulates (interpreted under the context of S5). In this setting, forgetting μ in φ consists in updating the latter with $\neg K\mu \wedge \neg K\neg\mu$, asserting that neither μ nor its negation is known to hold.

Zhang and Zhou [2009] is closer to our work: in that paper, the authors study a notion of knowledge forgetting in S5 and consider, among other things, knowledge update as a possible application of knowledge forgetting. Their notion of knowledge forgetting is similar to the one at work in classical propositional logic (and different from the one considered in Baral and Zhang [2005]); indeed, forgetting a set of propositional variables in a formula φ amounts to computing the logically strongest consequence of φ that is independent from the set (i.e., there exists a equivalent formula in which no atom from the given set occurs). Zhang and Zhou define two update operators, the second being the model-wise version of the first. A significant difference with our dependence-based update operators is that the set of variables to be forgotten depends not only on the change formula μ but also on the knowledge base φ itself. Roughly, the models of the updated base are obtained by considering in all possible ways minimal sets of atoms (w.r.t. $\subseteq$) such that forgetting them in φ leads to a formula consistent with μ. This is a way of implementing the principle of minimal change, and accordingly, the corresponding update operator satisfies **(U2)**.

Bienvenu et al. [2010] show how an S5 belief base can be progressed in polynomial time when it has been first compiled into a specific DNF form and the action description (which can refer to an ontic action or to an epistemic action) has been normalized as well. The notion of progression by an ontic action that is considered in their paper corresponds to an update operator following the "forget-then-conjoin" scheme that we follow in the present article.

7.5. Other Related Works

Nayak et al. [2006] investigate an interesting relationship between literal forgetting and update: they show that Forbus' update operator [Forbus 1989] and literal forgetting are inter-definable. This establishes an interesting connection between minimization-based and dependence-based approaches to belief update. Note that although our work also talks about literal forgetting and belief update, the nature of their work differs from ours, which aims at defining new update operators, based on formula/literal dependence, that can be computed via literal forgetting.

Georgatos [2008] introduces a form of update based on the minimization of the geodesic distance on a graph. The corresponding update operators correspond bijectively to geodesic metrics. Since such operators are based on minimization of change (and not on a "forget-then-conjoin" scheme), they are quite different from our update operators. Especially, Georgatos's operators satisfy **(U2)** while this is not the case for our operators in general.

From a more applicative side, Hoffmann et al. [2009] use belief update for planning and web service composition and, following previous proposals in that direction, show that applying a web service is essentially a belief update operation. The authors introduce the decomposability postulate **(D)** that we considered previously (they call it (IV)). As in our approach, some integrity constraints are considered in their work. However, their update operator is based on change minimization, following Winslett's PMA.

8. CONCLUSION

In this article, we have proposed a new family of belief update operators that are based on dependence functions associating to every change formula μ the set of literals on which μ has a negative influence. We have shown that the family of KM update operators and the family of update operators based on formula/literal dependence are disjoint. We have identified the KM postulates our operators satisfy (and under which conditions), and we have studied their computational properties: complexity of model checking, satisfiability and inference, as well as compactability.

We have shown that update operators based on formula/literal dependence generalize those based on formula/variable dependence as defined in [Doherty et al. 1998, 2000; Herzig 1996; Herzig and Rifi 1999]. We have also shown by means of examples that natural definitions of formula/literal dependence such as $Dep = RelLit$ induce smaller change of the belief base than operators based on formula/variable dependence; for example $(a \wedge b) \diamond^{\top}_{Dep} (a \vee b) \models a \wedge b$, while $(a \wedge b) \diamond^{\top}_{Dep_v} (a \vee b) \models a \vee b$.

Interestingly, this is achieved without requiring a complexity shift w.r.t. operators based on formula/variable dependence. In particular, the inference problem for update operators based on formula/literal dependence is coNP-complete, just as the inference problem for Winslett's standard semantics (WSS), Hegner's operator (which is identical to Doherty et al.'s MPMA operator and to Herzig and Rifi's WSS$\downarrow$), Herzig and Rifi's WSS$^{\delta_v}$and WSS$\downarrow^{\delta_v}$. Observe also that the other update operators of the literature that are not dependence-based all have higher complexity than ours; for instance, the inference problem for Winslett's PMA operator, Forbus' operator, Zhang and Foo's MCD operator ("minimal change with maximal disjunctive inclusion") and Zhang and Foo's MCE operator ("minimal change with exceptions", the update version of Weber's revision operator) is Π^p_2-complete [Eiter and Gottlob 1992; Liberatore 2000a].

As to the update postulates, Herzig and Rifi [1999] criticized the KM rationality postulates for update; they show that **(U1)**, **(U3)** and **(U8)** are uncontroversial, while **(U4)** is desirable, **(U7)** is neutral and **(U2)**, **(U5)**, and **(U6)** are undesirable. Notably, the most interesting of our update operators, those based on standard formula/literal

dependence functions, conform to the desiderata: they satisfy **(U1)**, **(U3)**, and **(U8)**, and those who are syntax-independent moreover satisfy **(U4)**; none of them satisfies **(U2)**, and **(U5)** and **(U6)** are not always satisfied.

This article calls for some perspectives for further research. First of all, it would be interesting to investigate other logical settings where update operators based on the "forget-then-conjoin" scheme could be defined. A key property for it is the ability of expressing in the language of the logic the result of forgetting some atoms in a formula. Indeed, while this is feasible in many logical settings, it is not always possible (for instance, this cannot be achieved in modal logic S4). On the other hand, in this article we have supposed that dependence functions are given. In future work we plan to investigate how they can be built automatically.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for some helpful comments.

REFERENCES

Alchourrón, C. E., Gärdenfors, P., and Makinson, D. 1985. On the logic of theory change: Partial meet contraction and revision functions. *J. Symb. Logic 50,* 2, 510–530.

Amir, E. and Russell, S. J. 2003. Logical filtering. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'03)*. 75–82.

Baral, C. and Gelfond, M. 2005. *Logic Programming and Reasoning about Actions*. Elsevier, 389–428.

Baral, C. and Zhang, Y. 2005. Knowledge updates: Semantics and complexity issues. *Artif. Intell. 164,* 1–2, 209–243.

Baral, C., Gelfond, M., and Provetti, A. 1997. Representing actions: Laws, observations and hypotheses. *J. Logic Program. 31,* 1–3, 201–243.

Bienvenu, M., Fargier, H., and Marquis, P. 2010. Knowledge compilation in the modal logic S5. In *Proceedings of the National Conference on Artificial Intelligence (AAAI'10)*. 261–266.

Bienvenu, M., Herzig, A., and Qi, G. 2008. Prime implicate-based belief revision operators. In *Proceedings of the European Conference on Artificial Intelligence (ECAI'08)*. 741–742.

Boutilier, C. 1998. A unified model of qualitative belief change: A dynamical systems perspective. *Artif. Intell. 98,* 1–2, 281–316.

Brewka, G. and Hertzberg, J. 1993. How to do things with worlds: On formalizing actions and plans. *J. Logic Comput. 3,* 5, 517–532.

Cadoli, M., Donini, F., Liberatore, P., and Schaerf, M. 1999. The size of a revised knowledge base. *Artif. Intell. 115,* 1, 25–64.

Darwiche, A. 2001. Decomposable negation normal form. *J. ACM 48,* 4, 608–647.

Darwiche, A. and Marquis, P. 2002. A knowledge compilation map. *J. Artif. Intell. Res. 17*, 229–264.

de Giacomo, G., Lenzerini, M., Poggi, A., and Rosati, R. 2006. On the update of description logic ontologies at the instance level. In *Proceedings of the National Conference on Artificial Intelligence (AAAI'06)*.

de Giacomo, G., Lenzerini, M., Poggi, A., and Rosati, R. 2007. On the approximation of instance level update and erasure in description logics. In *Proceedings of the National Conference on Artificial Intelligence (AAAI'07)*. 403–408.

Delgrande, J., Nayak, A., and Pagnucco, M. 2005. Gricean belief change. *Studia Logica 79*, 97–113.

Delgrande, J., Jin, Y., and Pelletier, J. 2008. Compositional belief update. *J. Artif. Intell. Res. 32*, 757–791.

Doherty, P., Łukasziewicz, W., and Madalińska-Bugaj, E. 1998. The PMA and relativizing change for action update. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR'98)*. 258–269.

Doherty, P., Łukasziewicz, W., and Madalińska-Bugaj, E. 2000. The PMA and relativizing change for action update. *Fundamenta Informaticae 44,* 1–2, 95–131.

Dupin de Saint-Cyr, F. and Lang, J. 2011. Belief extrapolation (or how to reason about observations and unpredicted change). *Artif. Intell. 175,* 2, 760–790.

Eiter, T. and Gottlob, G. 1992. On the complexity of propositional knowledge base revision, updates, and counterfactuals. *Artif. Intell. 57,* 2–3, 227–270.

Eiter, T., Erdem, E., Fink, M., and Senko, J. 2005. Updating action domain descriptions. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'05)*. 418–423.

Eiter, T., Erdem, E., Fink, M., and Senko, J. 2010. Updating action domain descriptions. *Artif. Intell. 174,* 15, 1172–1221.

Forbus, K. D. 1989. Introducing actions into qualitative simulation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'89)*. 1273–1278.

Georgatos, K. 2008. Belief update using graphs. In *Proceedings of the International Florida Artificial Intelligence Research Society Conference (FLAIRS'08)*. 649–654.

Giunchiglia, E. and Lifschitz, V. 1998. An action language based on causal explanation: Preliminary report. In *Proceedings of the National Conference on Artificial Intelligence (AAAI'98)*. 623–630.

Giunchiglia, E., Lee, J., Lifschitz, V., McCain, N., and Turner, H. 2004. Nonmonotonic causal theories. *Artif. Intell. 153,* 1–2, 49–104.

Herzig, A. 1996. The PMA revisited. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR'96)*. 40–50.

Herzig, A. and Rifi, O. 1998. Update operations: A review. In *Proceedings of the European Conference on Artificial Intelligence (ECAI'98)*. 13–17.

Herzig, A. and Rifi, O. 1999. Propositional belief update and minimal change. *Artif. Intell. 115*, 107–138.

Herzig, A., Lang, J., Marquis, P., and Polacsek, T. 2001. Updates, actions, and planning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'01)*. 119–124.

Hoffmann, J., Bertoli, P., Helmert, M., and Pistore, M. 2009. Message-based web service composition, integrity constraints, and planning under uncertainty: A new connection. *J. Artif. Intell. Res. 35*, 49–117.

Hunter, A. and Delgrande, J. 2011. Iterated belief change due to actions and observations. *J. Artif. Intell. Res. 40*, 269–304.

Karp, R. M. and Lipton, R. J. 1980. Some connections between non-uniform and uniform complexity classes. In *Proceedings of the Annual ACM Symposium on Theory of Computing (STOC'80)*. 302–309.

Katsuno, H. and Mendelzon, A. 1991. On the difference between updating a knowledge base and revising it. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*. 387–394.

Katsuno, H. and Mendelzon, A. 1992. Propositional knowledge base revision and minimal change. *Artif. Intell. 52*, 263–294.

Kourousias, G. and Makinson, D. 2007. Parallel interpolation, splitting, and relevance in belief change. *J. Symb. Logic 72,* 3, 994–1002.

Lang, J. 2007. Belief update revisited. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'07)*. 2517–2522.

Lang, J. and Marquis, P. 1998. Two forms of dependence in propositional logic: Controllability and definability. In *Proceedings of the National Conference on Artificial Intelligence (AAAI'98)*. 268–273.

Lang, J., Marquis, P., and Williams, M.-A. 2001. Updating epistemic states. In *Proceedings of the 14th Australian Joint Conference on Artificial Intelligence (AI'01)*. 297–308.

Lang, J., Liberatore, P., and Marquis, P. 2002. Conditional independence in propositional logic. *Artif. Intell. 141,* 1/2, 79–121.

Lang, J., Liberatore, P., and Marquis, P. 2003a. Propositional independence: Formula-variable independence and forgetting. *J. Artif. Intell. Res. 18*, 391–443.

Lang, J., Lin, F., and Marquis, P. 2003b. Causal theories of action: A computational core. In *Proceedings of the International Joint Conference Artificial Intelligence (IJCAI'03)*. 1073–1078.

Levesque, H. 1998. A completeness result for reasoning with incomplete first-order knowledge bases. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR'98)*. 14–23.

Liberatore, P. 2000a. The complexity of belief update. *Artif. Intell. 119*, 141–190.

Liberatore, P. 2000b. A framework for belief update. In *Proceedings of the European Conference on Logics in Artificial Intelligence (JELIA'00)*. 361–375.

Liberatore, P. and Schaerf, M. 2004. The compactness of belief revision and update operators. *Fundamenta Informaticae 62,* 3–4, 377–393.

Lin, F. and Reiter, R. 1994. How to progress a database (and why) I: Logical foundations. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR'94)*. 425–436.

Lin, F. and Reiter, R. 1997. How to progress a database. *Artif. Intell. 92,* 1–2, 131–167.

Liu, H., Lutz, C., Milicic, M., and Wolter, F. 2006. Updating description logic A-boxes. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR'06)*. 46–56.

Madalinska-Bugaj, E. and Łukaszewicz, W. 2009. First-order generalization of the MPMA belief update operator. *Fundamenta Informaticae 94*, 49–61.

Makinson, D. 2007. Propositional relevance through letter-sharing: Review and contribution. In *Formal Models of Belief Change in Rational Agents*. Dagstuhl Seminar Proceedings.

Marquis, P. 2000. Consequence finding algorithms. In *Handbook on Defeasible Reasoning and Uncertainty Management Systems*, vol. 5, Kluwer Academic Publisher, Chapter 2, 41–145.

McCain, N. and Turner, H. 1997. Causal theories of action and change. In *Proceedings of the National Conference on Artificial Intelligence (AAAI'97)*. 460–465.

Nayak, A. C., Chen, Y., and Lin, F. 2006. Forgetting and knowledge update. In *Proceedings of the 19th Australian Joint Conference on Artificial Intelligence (AI'06)*. 131–140.

Papadimitriou, C. 1994. *Computational Complexity*. Addison–Wesley.

Parikh, R. 1999. Beliefs, belief revision, and splitting languages. In *Logic, Language, and Computation*, vol. 2.

Shapiro, S., Pagnucco, M., Espérance, Y., and Levesque, H. J. 2011. Iterated belief change in the situation calculus. *Artif. Intell. 175,* 1, 165–192.

Shirazi, A. and Amir, E. 2011. First-order logical filtering. *Artif. Intell. 175,* 1, 193–219.

Vo, Q., Nayak, A., and Foo, N. 2006. A syntax-based approach to reasoning about action and belief update. *J. Logic Comput. 16,* 3, 315–338.

Winslett, M. 1990. *Updating Logical Databases*. Cambridge University Press.

Zhang, Y. and Zhou, Y. 2009. Knowledge forgetting: Properties and applications. *Artif. Intell. 173,* 16–17, 1525–1537.