Forgetting literals with varying propositional symbols

Yves Moinard
IRISA, Campus de Beaulieu, 35042 RENNES-Cedex FRANCE
moinard@irisa.fr

Abstract

Recently, the old logical notion of forgetting propositional symbols (or reducing the logical vocabulary) has been generalized to a new notion: forgetting literals. The aim was to help the automatic computation of various formalisms which are currently used in knowledge representation. We extend here this notion, by allowing propositional symbols to vary while forgetting literals. The definitions are not really more complex than for literal forgetting without variation. We describe the new notion, on the syntactical and the semantical side. Then, we show how to apply it to the computation of circumscription. This computation has been done before with standard literal forgetting, but here we show how introducing varying propositional symbols simplifies significantly the computation. We revisit a fifteen years old result about computing circumscription, showing that it can be improved in the same way. We provide hints in order to apply this forgetting method also to other logical formalisms.

1 Introduction

The well-known notion of forgetting propositional symbols has been used for a long time in mathematical logic and in its applications for knowledge representation. Recently, [Lang *et al.*, 2003] has extended this notion in a significant manner, by allowing the forgetting of *literals*. The main purpose of the authors was the introduction of a new way for helping the automatic computation of various formalisms.

The example that they provided concerns circumscription.

We extend the notion by allowing some propositional symbols to *vary* when forgetting literals. The new definitions are a simple and natural extension of the original ones, and they have the same kind of behavior.

We show that with this simple extension, the result given in [Lang *et al.*, 2003] for circumscription, which involves a two stages operation, can be reduced into a single stage. We extend the correlation (given in [Lang *et al.*, 2003]) of the method with an earlier proposal for computing circumscription [Przymusinski, 1989]. Precisely, we show that Przymunsinski's method, which is a two stages method also, can similarly be reduced into a single stage method.

Meanwhile, we show that this way of using the notion of literal forgetting is a particular case of a powerful formalism, introduced in [Siegel and Forget, 1996] in order to facilitate the computation for nonmonotonic reasoning.

2 Technical preliminaries

We work in a propositional language $\mathbf{PL}$. As usual, $\mathbf{PL}$ also denotes the set of all the formulas, and the *vocabulary of* $\mathbf{PL}$ is a set of *propositional symbols* denoted by $\mathcal{V}(\mathbf{PL})$. We restrict our attention to finite sets $\mathcal{V}(\mathbf{PL})$ in this text.

Letters φ, ψ denote formulas in $\mathbf{PL}$. Two logical constants $\top$ and $\bot$ denote respectively the true and the false formulas.

Letters ω, μ, ν denote *interpretations for* $\mathbf{PL}$, identified with subsets of $\mathcal{V}(\mathbf{PL})$. The notations $\omega \models \varphi$ and $\omega \models X$ for a set X of formulas are defined classically.

For a set E, $\mathcal{P}(E)$ denotes the set of the subsets of E. The set $\mathcal{P}(\mathcal{V}(\mathbf{PL}))$ of the interpretations for $\mathbf{PL}$ is denoted by **Mod**. A *model* of X is an interpretation ω such that $\omega \models X$, **Mod**(φ) and **Mod**(X) denote respectively the sets of the models of $\{\varphi\}$ and X.

A *literal* l is either a symbol p in $\mathcal{V}(\mathbf{PL})$ (*positive literal*) or its negation $\neg p$ (*negative literal*). A *clause* (respectively a *term*) is a disjunction (respectively a conjunction) of literals. Subsets of $\mathcal{V}(\mathbf{PL})$ are denoted by P, Q, V.

P^+ (respectively P^-) denotes the set of the positive (respectively negative) literals built on P, and $P^\pm$ denotes the set $P^+ \cup P^-$ of all the literals built on P (P and P^+ can often be assimilated).

For any (finite) set X of formulas, $\bigwedge X$ (respectively $\bigvee X$) denotes the conjunction (respectively disjunction) of all the formulas in X. We get: $\bigwedge X \equiv X$, $\bigwedge \emptyset \equiv \top$ and $\bigvee \emptyset \equiv \bot$.

$\mathcal{V}(X)$ denotes the set of the propositional symbols appearing in X.

A *disjunctive normal form* or DNF (respectively *conjunctive normal form* or CNF) of φ is a disjunction of consistent terms (respectively a conjunction of non trivial clauses) which is equivalent to φ.

A *prime CNF* is a CNF which involves only *prime implicates*: for any clauses c, c_i, if $\varphi \models c$, and $c \models c_i$ for some c_i in the CNF, then $c_i \models c$.

A set L of literals in $V^{\pm}$ (and the term $\bigwedge L$) is *consistent and complete in* V if each propositional symbol of V appears once and only once in L; the clause $\bigvee L$ is then *non trivial and complete in* V. For any set L of literals, $\neg L$ denotes the set of the literals complementary to those in L.

We need the following notions and notations, many of them coming from [Lang *et al.*, 2003]:

If φ is some formula and p is a propositional symbol in $\mathbf{PL}$, then $\varphi_{p \leftarrow 1}$ (respectively $\varphi_{p \leftarrow 0}$) is the formula obtained from φ by replacing each occurrence of p by $\top$ (respectively $\bot$). If l is the positive literal p (respectively the negative literal $\neg p$), then $\varphi_{l \leftarrow i}$ denotes the formula $\varphi_{p \leftarrow i}$ (respectively $\varphi_{p \leftarrow (1-i)}$), for $i \in \{0, 1\}$.

Notations 2.1

1. *If $v_1, \cdots, v_n$ are propositional symbols, $\varphi_{(v_1 \leftarrow i_1, \cdots, v_n \leftarrow i_n)}$ with each $i_j \in \{0,1\}$ denotes the formula $(\cdots((\varphi_{v_1 \leftarrow i_1})_{v_2 \leftarrow i_2})\cdots)_{v_n \leftarrow i_n}$.*

   If the v_j's in the list are all distinct, then the order of the v_j's is without consequence for the final result. Thus, if V_1 and V_2 are disjoint subsets of V, we may define $\varphi_{[V_1 \leftarrow 1, V_2 \leftarrow 0]}$ as $\varphi_{[v_1 \leftarrow 1, \cdots, v_n \leftarrow 1, v_{n+1} \leftarrow 0, \cdots, v_{n+m} \leftarrow 0]}$, where $(v_1, \cdots, v_n)$ and $(v_{n+1}, \cdots, v_{n+m})$ are two orderings of all the elements of V_1 and V_2 respectively.

2. *If $L = (l_1, \cdots, l_n)$ is a list of literals, then $\varphi_{(l_1 \leftarrow i_1 \cdots l_n \leftarrow i_n)}$ denotes the formula $(\cdots((\varphi_{l_1 \leftarrow i_1})_{l_2 \leftarrow i_2})\cdots)_{l_n \leftarrow i_n}$.*

3. *Let $\mathcal{V}(\mathbf{PL})^{\pm}$ be ordered in some arbitrary way. If $L_1, \cdots, L_n$ are disjoint sets of literals, $\varphi_{\langle L_1 \leftarrow i_1, \cdots, L_n \leftarrow i_n \rangle}$ denotes the formula $\varphi_{(l_1 \leftarrow k_1, \cdots, l_n \leftarrow k_n)}$ where $(l_1, \cdots, l_n)$ is the enumeration of the set $L_1 \cup \cdots \cup L_n$ which respects the order chosen for the set of all the literals, and where, for each l_j, k_j is equal to i_r where $r \in \{1, \cdots, n\}$ is such that $l_j \in L_r$.*

Let us remind a well known notion, *variable forgetting*:

Definition 2.2 *If $V \subseteq \mathcal{V}(\mathbf{PL})$ and $\varphi \in \mathbf{PL}$, $ForgetV(\varphi, V)$ denotes a formula, in the propositional language $\mathbf{PL}_{\overline{V}}$ built on the vocabulary $\overline{V} = \mathcal{V}(\mathbf{PL}) - V$, which is equivalent to φ in this restricted language: $ForgetV(\varphi, V) \equiv Th(\varphi) \cap \mathbf{PL}_{\overline{V}}$ where $Th(\varphi) = \{\varphi' \in \mathbf{PL} / \varphi \models \varphi'\}$.*

For any $\psi \in \mathbf{PL}_{\overline{V}}$, $\varphi \models \psi$ iff $ForgetV(\varphi, V) \models \psi$.

Various ways are known in order to get $ForgetV(\varphi, V)$, the two easiest (to describe) ways being the following ones:

1. In a DNF form of φ, suppress all the literals in $V^{\pm}$.
2. In a prime CNF form of φ, suppress any clause containing a symbol in V.

Remark 2.3 *When considering a formula equivalent to a set $Th(\varphi) \cap X$, the set of formulas X can be replaced by any set Y having the same $\wedge$-closure: $\{\bigwedge X' / X' \subseteq X\} = \{\bigwedge X' / X' \subseteq Y\}$. Indeed, we have:*

- *If X and Y have the same $\wedge$-closure, then $Th(\varphi) \cap X \equiv Th(\varphi) \cap Y$.*
- *The converse is true,* provided that we assimilate equivalent formulas*: if $Th(\varphi) \cap X \equiv Th(\varphi) \cap Y$ for any $\varphi \in \mathbf{PL}$, then X and Y have the same $\wedge$-closure.*

*Since we work in finite propositional languages, there exists a unique smallest (for set inclusion, and up to logical equivalence) possible set, the $\wedge$-*reduct of X*, equal to the set $X - \{\varphi \in X / \varphi$ is in the $\wedge$-closure of $X - \{\varphi\}\}$.*

Thus, X can be replaced by any set which contains the $\wedge$-reduct of X and is included in the $\wedge$-closure of X.

Thus, instead of considering the whole set $\mathbf{PL}_{\overline{V}}$ in $ForgetV(\varphi, V) \equiv Th(\varphi) \cap \mathbf{PL}_{\overline{V}}$ (Definition 2.2), we can consider the set of all the clauses built on $\overline{V}$, the smallest (for $\subseteq$) set that can be considered here being the set of these clauses which are non trivial and complete in $\overline{V}$.

From a semantical side, the set of the models of $ForgetV(\varphi, V)$ is the set of all the interpretations for $\mathbf{PL}$ which coincide with a model of φ for all the propositional symbols not in V.

3 Literal forgetting

Variable forgetting as been generalized as detailed now, beginning by the semantical side.

Definition 3.1 *[Lang* et al.*, 2003, pp. 396–397] Let ω be an interpretation for $\mathbf{PL}$, p be a propositional symbol in $\mathbf{PL}$ and L be a consistent set of literals in $\mathbf{PL}$.*

We define the interpretations
$Force(\omega, p) = \omega \cup \{p\}$ and $Force(\omega, \neg p) = \omega - \{p\}$
and more generally,
$Force(\omega, L) =$
$\omega \cup \{p / p \in \mathcal{V}(\mathbf{PL}), p \in L\} - \{p / p \in \mathcal{V}(\mathbf{PL}), \neg p \in L\}$.

Thus, $Force(\omega, L)$ is the interpretation for $\mathbf{PL}$ which is equal to ω for all the propositional symbols in $\mathcal{V}(\mathbf{PL}) - \mathcal{V}(L)$ and which satisfies all the literals of L.

Definition 3.2 (Literal forgetting) *[Lang* et al.*, 2003, Prop. 15] If φ is a formula and L a set of literals in $\mathbf{PL}$, $ForgetLit(\varphi, L)$ is a formula having for models the set of all the interpretations for $\mathbf{PL}$ which can be turned into a model of φ when forced by a consistent subset of L:*

Mod$(ForgetLit(\varphi, L)) = \{\omega /$
$Force(\omega, L_1) \models \varphi$ and L_1 is a consistent subset of $L\}$.

This amounts to say that the models of $ForgetLit(\varphi, L)$ are built from the models of φ by allowing to negate an arbitrary number of values of literals in L:

$$\mathbf{Mod}(ForgetLit(\varphi, L)) = \{Force(\omega', L'_1) \ / \ \omega' \models \varphi \quad \textit{and} \quad L'_1 \textit{ is a consistent subset of } \neg L\}.$$

Let us consider the syntactical side now. One way is to start from a DNF formulation of φ:

Proposition 3.3 *[Lang* et al.*, 2003] If $\varphi = t_1 \vee \cdots \vee t_n$ is a DNF, then $ForgetLit(\varphi, L)$ is equivalent to the formula $t'_1 \vee \cdots \vee t'_n$ where t'_i is the term t_i without the literals in L.*

The similar method for obtaining $ForgetV(\varphi, V)$ when φ is a DNF has been reminded in point 1 following Definition 2.2. Remind also that, for any φ, $ForgetV(\varphi, V)$ can be defined as follows (cf Notations 2.1-1):

$$ForgetV(\varphi, V) = \bigvee_{V' \subseteq V} \varphi_{[V' \leftarrow 1, \ (V - V') \leftarrow 0]}.$$

Similarly, the following syntactical definition can be given:

Definition 3.4 *If L is a set of literals in $\mathbf{PL}$, then*

$$ForgetLit(\varphi, L) = \bigvee_{L' \subseteq L} \left(\left(\bigwedge \neg L' \right) \wedge \varphi_{\langle (L - L') \leftarrow 1 \rangle} \right).$$

This is a paraphrase of [Lang *et al.*, 2003, Definition 7]. We refer the reader to this text which shows

1. the adequacy with Definition 3.2 and Proposition 3.3,
2. that choosing any order of the literals does not modify the meaning of the final formula (cf Notations 2.1-3), and
3. that we get also

$$ForgetLit(\varphi, L) \equiv \bigvee_{L' \subseteq L} \left(\left(\bigwedge \neg L' \right) \wedge \varphi_{\langle (L - L') \leftarrow 1, L' \leftarrow 0 \rangle} \right).$$

The presence of $\left(\bigwedge_{l' \in L'} \neg l'\right)$, which is what differentiates $ForgetLit(\varphi, ...)$ from $ForgetV(\varphi, ...)$, comes from the fact that here we want to forget $l \in L$, but not $l' \in \neg L$.

A proof in [Lang *et al.*, 2003], using Proposition 3.3, shows that we get $ForgetLit(\varphi, V^{\pm}) \equiv ForgetV(\varphi, V)$.

This proof is easily extended to get:

Remark 3.5 *Since any set of literals can be written as a disjoint union between a consistent set L' and a set $V^{\pm}$ of complementary literals, here is a useful formulation:*

$$ForgetLit(\varphi, L' \cup V^{\pm}) \equiv ForgetLit(ForgetV(\varphi, V), L').$$

(Notice that L' can even be inconsistent in this equivalence.)

This remark has the advantage of separating clearly the *literals* which are forgotten and the *propositional symbols* which are forgotten. Let V' denote the set $\mathcal{V}(L')$ of the propositional symbols in L', and $V'' = \mathcal{V}(\mathbf{PL}) - V - V'$ be the set of the remaining symbols. Then we get:

1. The propositional symbols in V are forgotten;
2. the literals in L' are forgotten, but their symbols (in V') may remain, since the literals in $\neg L'$ are not forgotten;
3. the literals in $V''^{\pm}$ are not forgotten, thus the propositional symbols in V'' are *fixed*.

Thus, $ForgetLit(\varphi, L_1)$ can be defined as: *forgetting literals with some propositional symbols fixed*. It is tempting to generalize the notion, by allowing some propositional symbols to *vary* in the *forgetting* process.

4 Literal forgetting with varying symbols

As done with the original notion, it is convenient to introduce the semantical definitions first.

Definition 4.1 *Let φ be a formula, V a set of propositional symbols, and L a consistent set of literals, in $\mathbf{PL}$, with V and $\mathcal{V}(L)$ disjoint in $\mathcal{V}(\mathbf{PL})$. $ForgetLitVar(\varphi, L, V)$ is a formula having the following set of models:*

$$\mathbf{Mod}(ForgetLitVar(\varphi, L, V)) = \{\omega \ / \ Force(\omega, L_1 \cup L_2) \models \varphi \textit{ where } L_1 \subseteq L, L_2 \subseteq V^{\pm}, \ L_2 \textit{ consistent, and } (\omega \not\models L_1 \textit{ or } L_2 = \emptyset)\}.$$

This is equivalent to:

$$\mathbf{Mod}(ForgetLitVar(\varphi, L, V)) = \{Force(\omega, L_1 \cup L_2) \ / \ \omega \models \varphi \textit{ where } L_1 \subseteq \neg L, \ L_2 \subseteq V^{\pm}, \ L_2 \textit{ consistent and } (\omega \not\models L_1 \textit{ or } L_2 = \emptyset)\}.$$

Since $\omega \models L_2$ iff $Force(\omega, L_2) = \omega$, in these two formulations the condition "*$(\omega \not\models L_1$ or $L_2 = \emptyset)$*" can be replaced by "*$(\omega \not\models L_1$ or $\omega \models L_2)$*", and then we can replace "L_2 *consistent*" by "L_2 *consistent and complete in* V*"*.

We could be more general, by allowing to forget some propositional symbols, which amounts to allow non consistent sets L. This generalization does not present difficulties, however, since we have not found any application for it till now, we leave it for future work.

With respect to Definition 3.2, what happens here is that the non consistent part of the set of literals, which allowed to forget some set V of propositional symbols altogether, has been replaced by a set of varying propositional symbols.

Remark 4.2 *Since $ForgetLit(L_1, \varphi) \models ForgetLit(L_1 \cup L_2, \varphi)$ holds from [Lang* et al.*, 2003], we get:*

$$\varphi \models ForgetV(\varphi, V) \models ForgetLit(\varphi, L \cup V^{\pm}).$$

It is immediate to show that we get also:

$$\varphi \models ForgetLitVar(\varphi, L, V) \models ForgetLit(\varphi, L \cup V^{\pm}).$$

Here are the motivations for introducing $ForgetLitVar$: we want to "forget" the literals in L, even at the price of modifying the literals in $V^{\pm}$. This is what explains that, if we effectively forget at least one literal in L, then, we allow

any modification for the literals in $V^{\pm}$. However, we do not want to modify the literals in $V^{\pm}$ "for nothing": our aim is to forget as many literals in L as possible. This justifies the condition "*($\omega \not\models L_1$ or $L_2 = \emptyset$)*" in the definition.

The syntactical aspect is slightly more tricky, but it remains rather simple and it allows to revisit and improve already known results. As with the original notion (see Proposition 3.3), the simplest way is to start from a DNF.

Without loss of generality we can consider that L is a set of negative literals (otherwise, replace any $p \in \mathcal{V}(L)$ such that $p \in L$ by $\neg p'$, p' being a new propositional symbol, then after the computations, replace p' by $\neg p$). Thus, till now, we will consider two disjoint subsets P and V of $\mathcal{V}(\mathbf{PL})$, and $L = \neg P$ with $Q = \mathcal{V}(\mathbf{PL}) - V - P$ denoting the set of the remaining propositional symbols.

Proposition 4.3 *Let $\varphi = t_1 \vee \cdots \vee t_n$ be a DNF, with*

$$t_i = (\bigwedge P_{i,1}) \wedge (\bigwedge \neg(P_{i,2})) \wedge (\bigwedge V_{i,l}) \wedge (\bigwedge Q_{i,l}),$$

where $P_{i,1} \subseteq P$, $P_{i,2} \subseteq P - P_{i,1}$, and where $V_{i,l}$ and $Q_{i,l}$ are consistent sets of literals in $V^{\pm}$ and $Q^{\pm}$ respectively. Then $ForgetLitVar(\varphi, P^-, V) \equiv t'_1 \vee \cdots \vee t'_n$ where

$$t'_i = (\bigwedge P_{i,1}) \wedge (\bigwedge Q_{i,l}) \wedge ((\bigvee (P - P_{i,1})) \vee (\bigwedge V_{i,l})), \text{i.e.}$$

$$t'_i = (\bigwedge P_{i,1}) \wedge (\bigwedge Q_{i,l}) \wedge \bigwedge_{l \in V_{i,l}} (l \vee (\bigvee (P - P_{i,1}))).$$

Thus, t'_i is t_i except that the literals in $\neg P$ are suppressed while each literal in $V^{\pm}$ must be disjuncted with the clause $\bigvee(P - P_1)$, this clause denoting the disjunction of all the literals in P^+ which do not appear (positively) in t_i.

Proof: Let us consider complete terms first, such as

$$t_i = t = (\bigwedge P_1) \wedge (\bigwedge \neg(P - P_1)) \wedge (\bigwedge V_l) \wedge (\bigwedge Q_l),$$

where $P_1 \subseteq P$, V_l and Q_l being consistent and complete sets of literals in V and Q respectively. t corresponds to an interpretation ω. The set $F(\omega) =$
$\{Force(\omega, L_1 \cup L_2) \ / \ L_1 \subseteq P,\ L_2 \subseteq V^{\pm},$
L_2 consistent and complete in V, and $\omega \not\models L_1$ or $\omega \models L_2\}$
is the set of the models of the formula $t^1 \wedge t^2$ where
$t^1 = (\bigwedge P_1) \wedge (\bigwedge Q_l)$ and $t^2 = (\bigvee (P - P_1)) \vee (\bigwedge V_l))$.

Indeed, for each $\omega' \in F(\omega)$, t^1 holds since it holds in ω, and the symbols in $P - P_1$ and V can take any value satisfying the condition $\omega \not\models L_1$ or $\omega \models L_2$. Since $\omega \models t$, this means $L_1 \cap (P - P_1) \neq \emptyset$ or $L_2 \subseteq V_l$, which is equivalent to $\omega' \models t_2$. Conversely, any model ω'' of $t_1 \wedge t_2$ is easily seen to be in $F(\omega)$.

The same result holds for any (consistent) term $t = t_i = (\bigwedge P_1) \wedge (\bigwedge \neg(P_2)) \wedge (\bigwedge V_l) \wedge (\bigwedge Q_l)$, where $P_1 \subseteq P$, $P_2 \subseteq P - P_1$, V_l and Q_l being consistent subsets of $V^{\pm}$ and $Q^{\pm}$ respectively: Let us first consider separately the cases where some symbols in P are missing, then symbols in V, then symbols in Q.

(1) If $p \in P$ is missing in t (p and $\neg p$ are missing), for any model ω' of t, $\omega'' = Force(\omega', \{\neg p\})$ and $Force(\omega'', \{p\})$ are two models of t (one of these is ω'). By considering all the missing p's, we get that the set
$\{Force(\omega', L_1 \cup L_2) \ / \ \omega' \models t, \quad L_1 \subseteq \neg P, L_2$ consistent $\subseteq V^{\pm}, \omega' \not\models L_1$ or $L_2 = \emptyset\}$ is included in the set
$\{Force(\omega'', L_1 \cup L_2) \ / \ \omega'' \models t \wedge \bigwedge \neg(P - P_1),\ L_1 \subseteq \neg P, L_2$ consistent $\subseteq V^{\pm}, \omega'' \not\models L_1$ or $L_2 = \emptyset\}$.
Thus any missing p in t behaves as if the negative literal $\neg p$ was present: we get a term "completed in P" satisfying
$ForgetLitVar(t, P^-, V) \equiv$
$ForgetLitVar(t \wedge \neg(P - P_1), P^-, V)$.

(2) The reasoning for a missing q in t ($q \in Q$) is simpler yet: if some $q \in Q$ does not appear in t, it can be interpreted as false or true for any model of $ForgetLitVar(t, L, Q)$, which means that we keep the part $\bigwedge Q_l$ unmodified, exactly as in the case where Q_l is complete in Q.

(3) The case for V is similar (the disjunction of all the formulas with all the possibilities for the missing symbols gives the formula where these symbols are missing): If some $v \in V$ is missing in t, then any model ω' of t has its counterpart where the value for v is modified. Let us call V_m the set of the symbols in V which are absent in t. By considering the disjunctions of all the possibilities, we get the formula $\bigvee_{V'_l \in \mathcal{L}_m} ((\bigwedge P_1) \wedge (\bigwedge Q_l) \wedge ((\bigvee (P - P_1)) \vee (\bigwedge V_l \wedge \bigwedge V'_l)))$, where $\mathcal{L}_m$ is the set of all the sets of literals consistent and complete in V_m. This is equivalent to the formula $(\bigwedge P_1) \wedge (\bigwedge Q_l) \wedge ((\bigvee (P - P_1)) \vee (\bigwedge V_l))$.

Combining "the three incompleteness" (1)–(3) gives:
$ForgetLitVar(t_i, P^-, V) \equiv$
$(\bigwedge P_1) \wedge (\bigwedge Q_l) \wedge ((\bigvee (P - P_1)) \vee (\bigwedge V_l))$.
The disjunction for all the t_i's gives the result. □

We have provided the semantical definition (in the lines of Definition 3.2) and a characterization from a DNF formulation (in the lines of Proposition 3.3). Let us provide now other characterizations, and a comparison with $ForgetLit$.

Proposition 4.4 *Let φ be a formula in $\mathbf{PL}$, and P, Q and V be three pairwise disjoint sets of propositional symbols such that $P \cup Q \cup V = \mathcal{V}(\mathbf{PL})$.*

1. *$ForgetLit(\varphi, P^- \cup V^{\pm})$ is equivalent to the set $Th(\varphi) \cap X$ where X is the set of all the formulas in $\mathbf{PL}$ which are disjunctions of terms of the kind $(\bigwedge P_1) \wedge (\bigwedge Q_l)$ with $P_1 \subseteq P$ and $Q_l \subseteq Q^{\pm}$ (we can clearly consider consistent sets Q_l only).*
2. *$ForgetLitVar(\varphi, P^-, V)$ is equivalent to the set $Th(\varphi) \cap X$ where X is the set of all the formulas in $\mathbf{PL}$ which are disjunctions of terms of the kind $(\bigwedge P_1) \wedge (\bigwedge Q_l) \wedge \bigwedge_{l \in V_l} (l \vee (\bigvee (P - P_1)))$, where $P_1 \subseteq P$, V_l and Q_l being consistent sets of literals in $V^{\pm}$ and $Q^{\pm}$ respectively.*

These two results are immediate consequences of Propositions 3.3 and 4.3 respectively. We get the following alternative possibilities for the sets X's, firstly by duality from the preceding results, then by considering some set

having the same $\wedge$-closure as X (Remark 2.3):

Proposition 4.4 (following)

1. (a) *For $ForgetLit(\varphi, P^- \cup V^\pm)$, X is the set of all the conjunctions of the clauses of the kind $(\bigvee P_1) \vee (\bigvee Q_l)$ with $P_1 \subseteq P$ and $Q_l \subseteq Q^\pm$ (we can clearly consider consistent sets Q_l only).*
   (b) *We can also consider the set X of all the clauses $(\bigvee P_1) \vee (\bigvee Q_l)$ with $P_1 \subseteq P$ and $Q_l \subseteq Q^\pm$.*
   (c) *The smallest set X possible is the set of all the clauses $(\bigvee P_1) \vee (\bigvee Q_l)$ with $P_1 \subseteq P$, $Q_l \subseteq Q^\pm$, Q_l consistent and complete in Q.*
2. (a) *For $ForgetLitVar(\varphi, P^-, V)$, X is the set of all the conjunctions of the formulas $flv(P_1, Q_l, V_l) = (\bigvee P_1) \vee (\bigvee Q_l) \vee \bigvee_{l \in V_l} (l \wedge (\bigwedge (P - P_1)))$, where $P_1 \subseteq P$, V_l and Q_l being consistent sets of literals in $V^\pm$ and $Q^\pm$ respectively.*
   (b) *We can also consider the set X of all the formulas $flv(P_1, Q_l, V_l)$ of this kind.*
   (c) *The smallest set X possible is the set of all the formulas $flv(P_1, Q_l, V_l)$ with $P_1 \subseteq P$, Q_l and V_l being sets of literals consistent and complete in Q and V respectively.*

These results provide the analogous, for $ForgetLit$ and $ForgetLitVar$, of the results for $ForgetV$ reminded in Definition 2.2, and in Remark 2.3.

The next definition, analogous to Definition 3.4, will be our last general result about $ForgetLitVar$:

Definition 4.5 *If φ is a formula and P and V are two disjoint subsets of $\mathcal{V}(\mathbf{PL})$, then $ForgetLitVar(\varphi, P^-, V)$ is the formula*

$$\bigvee_{P_1 \subseteq P} \Big(\bigwedge P_1 \wedge (\varphi_{[P_1 \leftarrow 1, (P-P_1) \leftarrow 0]} \vee ForgetV(\varphi_{[P_1 \leftarrow 1, (P-P_1) \leftarrow 0]}, V) \wedge (\bigvee (P - P_1)))) \Big).$$

Proof of the adequacy with Definition 4.1: Each model ω of φ gives rise to the following models of $ForgetLitVar(\varphi, P^-, V)$:

- ω itself, model of $\psi_1 = \bigwedge P_1 \wedge \bigwedge \neg(P - P_1) \wedge \varphi_{[P_1 \leftarrow 1,\ (P-P_1) \leftarrow 0]}$ where $P_1 = \omega \cap P$, together with
- all the interpretations differing from ω in that they have at least one more $p \in P$, and no constraint holds for the symbols in V; this set of interpretations being the set of models of the formula $\psi_2 = \bigwedge P_1 \wedge ForgetV(\varphi_{[P_1 \leftarrow 1,\ (P-P_1) \leftarrow 0]}, V) \wedge \bigvee (P - P_1)$.

Since $\varphi_{[P_1 \leftarrow 1, (P-P_1) \leftarrow 0]} \models ForgetV(\varphi_{[P_1 \leftarrow 1, (P-P_1) \leftarrow 0]}, V)$ and $\bigwedge \neg(P - P_1) \equiv \neg(\bigvee (P - P_1))$, when considering the disjunction $\psi_1 \vee \psi_2$, we can suppress $\wedge \bigwedge \neg(P - P_1)$ in ψ_1. The disjunction of all these formulas $\psi_1 \vee \psi_2$ for each model ω of φ, gives the formula as written in this definition. □

Example 1 *Here $P = \{a, b\}, V = \{c\}, Q = \{d\}$, with $\varphi = (\neg a \wedge b \wedge c) \vee (a \wedge \neg b \wedge \neg c \wedge \neg d)$.*

Since φ is a DNF, the rules from a DNF in Section 2, Propositions 3.3 and 4.3 respectively for $ForgetV$, $ForgetLit$ and $ForgetLitVar$, give the three results:

- $ForgetV(\varphi, V) \equiv (\neg a \wedge b) \vee (a \wedge \neg b \wedge \neg d)$.
- $ForgetLit(\varphi, P^- \cup V^\pm) \equiv b \vee (a \wedge \neg d)$.
- $ForgetLitVar(\varphi, P^-, V) \equiv (a \wedge b) \vee (a \wedge \neg c \wedge \neg d) \vee (b \wedge c)$. $\quad FLV1$

Definitions 3.4 and 4.5 can be used also, as shown now for Definition 4.5 where, in each case, $\psi = \varphi_{[P_1 \leftarrow 1, (P-P_1) \leftarrow 0]}$:

$$P_1 = \emptyset : \psi \vee (ForgetV(\psi, c) \wedge (a \vee b)) \equiv \bot \vee (\bot \wedge (a \vee b)) \equiv \bot. \quad (\varphi_1)$$

$$P_1 = \{a\} : a \wedge (\psi \vee (ForgetV(\psi, c) \wedge b)) \equiv a \wedge ((\neg c \wedge \neg d) \vee (\neg d \wedge b)). \quad (\varphi_2)$$

$$P_1 = \{b\} : b \wedge (\psi \vee (ForgetV(\psi, c) \wedge a)) \equiv b \wedge (c \vee (\top \wedge a)). \quad (\varphi_3)$$

$$P_1 = \{a, b\} : a \wedge b \wedge (\psi \vee (ForgetV(\psi, c) \wedge \bot)) \equiv a \wedge b \wedge (\bot \vee \bot) \equiv \bot. \quad (\varphi_4)$$

The disjunction $\bigvee_{i=1}^4 \varphi_i$ is equivalent to $FLV1$.

Let us examine the semantical side now, starting with **Mod**$(\varphi) = \{\mu_1, \mu_2, \mu_3\}$ *with* $\mu_1 = \{a\}, \mu_2 = \{b, c\}, \mu_3 = \{b, c, d\}$.

- *The six models of $ForgetV(\varphi, V)$ are obtained by adding the three interpretations differing from the three models of φ by the value attributed to c: $\mu'_1 = \{a, c\}, \mu'_2 = \{b\}, \mu'_3 = \{b, d\}$.*
- *The ten models of $ForgetLit(\varphi, P^- \cup V^\pm)$ are obtained by adding to the models of φ the seven interpretations differing from these models by adding any subset of $\{a, b\}$ and by either do nothing else or modify the value of c (adding c if it is not present and removing c if it is present). This gives the models of $ForgetV(\varphi, V)$ plus the four interpretations including $\{a, b\}$.*
- *The seven models of $ForgetLitVar(\varphi, P^-, V)$ are obtained by adding to the models of φ the four interpretations differing from these models by adding a* non empty *subset of $\{a, b\}$ and by either do nothing else or modify the value of c, which gives here the four interpretations including $\{a, b\}$.*

Notice that on this example we get (cf Remark 4.2):

- $ForgetV(\varphi, V) \vee ForgetLitVar(\varphi, P^-, V) \equiv ForgetLit(\varphi, P^- \cup V^\pm)$
- $ForgetV(\varphi, V) \wedge ForgetLitVar(\varphi, P^-, V) \equiv \varphi$.

It is immediate (from the semantical characterizations) to show that these two equivalences are general.

Notice also that, once we have all the models of φ, the complexity of the construction of all the models of $ForgetLitVar(\varphi, P^-, V)$ is not greater than the complexity of the construction of all the models of $ForgetLit(\varphi, P^- \cup V^\pm)$.

Let us (re-)examine an application of literal forgetting.

5 Circumscription and literal forgetting

Literal forgetting has connexions with circumscription, a well known formalism in knowledge representation [McCarthy, 1986]. Circumscription minimizes "exceptions", and it is used in action languages and other formalizations of common sense reasoning. A key issue is the efficient computation, so that any step towards an amelioration can be helpful.

Definition 5.1 $\mathcal{V}(\mathbf{PL}) = P \cup V \cup Q$ *(disjoint sets). The propositional symbols in* P, V, Q *are respectively* circumscribed, varying, *and* fixed.

We define the relation $\prec_{(P,Q,V)}$ *in* **Mod** *by*
$\mu \prec_{(P,Q,V)} \nu$ *if* $P \cap \mu \subset P \cap \nu$ *and* $Q \cap \mu = Q \cap \nu$
(strict $\subset$*, no condition for* V*).*

A model μ *of* φ *is* minimal for $\prec_{(P,Q,V)}$ *if for no model* ν *of* φ *we have* $\nu \prec \mu$*.*

The circumscription $CIRC(P, Q, V)$ *is the mapping* $\mathbf{PL} \hookrightarrow \mathcal{P}(\mathbf{PL})$ *defined as follows:*

for any φ, ψ *in* $\mathbf{PL}$*,* $\psi \in CIRC(P, Q, V)(\varphi)$ *iff any model of* φ *minimal for* $\prec_{(P,Q,V)}$ *is a model of* ψ*.*

This is the semantical definition of circumscription [Lifschitz, 1994], in the propositional case as given in [Moinard and Rolland, 1998b]. The set $CIRC(P, Q, V)(\varphi)$ is equivalent to the formula having for models the models of φ minimal for $\prec_{(P,Q,V)}$, and $CIRC(P, Q, V)(\varphi)$ will sometimes be identified with this formula.

We get the following result:

Theorem 5.2 $Circ(P, Q, V)(\varphi) \models \psi$ *iff* $\varphi \models ForgetLitVar(\varphi \wedge \psi, P^-, V)$.

This is in fact a rewriting of an already known result. Some technical indications are useful. in order to relate this result with the literature. First, this result improves the following result:

Proposition 5.3 *[Lang* et al.*, 2003, Proposition 22]*

1. *If* φ *does not contain the propositional symbols in* V*, then* $Circ(P, Q, V)(\varphi) \models \psi$ *iff* $\varphi \models ForgetLit(\varphi \wedge \psi, P^- \cup V^{\pm})$.

2. *In the general case,* $Circ(P, Q, V)(\varphi) \models \psi$ *iff*

$\varphi \models ForgetLit(\varphi \wedge \neg ForgetLit(\varphi \wedge \neg\psi, P^- \cup V^{\pm}), P^- \cup V^{\pm})$.

Point 1 concerns circumscription without varying proposition since, if ψ has no symbol in V, $Circ(P, Q, V)(\varphi) \models \psi$ iff $Circ(P, Q, \emptyset)(ForgetV(\varphi, V)) \models \psi$.

As shown in [Lang *et al.*, 2003], the two points of this result are respectively consequences of the two points of an older result:

Proposition 5.4 *[Przymusinski, 1989, Theorems 2.5 – 2.6]*

1. *[Theorem 2.5] If* ψ *is without symbol in* V*, then* $Circ(P, Q, V)(\varphi) \models \psi$ *iff for any clause* c *made of literals not in* $P^- \cup V^{\pm}$*,* $\varphi \wedge \psi \models c$ *implies* $\varphi \models c$*.*

2. *[Theorem 2.6]* $Circ(P, Q, V)(\varphi) \models \psi$ *iff there exists* $\gamma = ForgetLit(\gamma', P^- \cup V^{\pm})$ *for some formula* γ'*, such that* $CIRC(P, Q, V)(\varphi) \models \neg\gamma$ *and* $\varphi \models \gamma \vee \psi$*.*

The second point is also a two stages method, which greatly increases its complexity: we must examine all the formulas γ in a rather large set and check, by using the first point, whether $\neg\gamma$ is entailed by the circumscription or not. The first point is again in fact restricted to the circumscriptions without varying proposition.

Let us introduce here another nonmonotonic formalism, which encompasses the formalisms implicitly used in the first points of Propositions 5.3 and 5.4.

Definition 5.5 *[Siegel and Forget, 1996] An* X-mapping *is a mapping* $f_X : \mathbf{PL} \hookrightarrow \mathcal{P}(\mathbf{PL})$ *defined from a set* X *of formulas in* $\mathbf{PL}$ *as follows:*

$$\varphi \in f_X(\psi) \quad \textit{iff} \quad Th(\psi \wedge \varphi) \cap X \subseteq Th(\psi).$$

This notion has been introduced in order to facilitate the computation of a nonmonotonic formalism. In fact, this notion appeared before the referenced paper, e.g. in [Suchenek, 1993] and even in earlier papers about the computation of minimization formalisms. The novelty in [Siegel and Forget, 1996] was a clear identification of the kind of formalism at hand, in particular it is shown there that, contrarily to circumscription and other formalisms, we do not get a theory: the set $f_X(\varphi)$ is generally not closed for $\wedge$, thus $f_X(\varphi)$ in the general case cannot be assimilated to a formula.

We can choose for X any set having the same $\wedge$-closure (cf Remark 2.3).

Let f be a mapping: $\mathbf{PL} \hookrightarrow \mathcal{P}(\mathbf{PL})$. The set of the formulas *inaccessible for* f is the set of all the formulas ψ for which there exist no formula φ such that $\psi \in f(\varphi)$ and $\varphi \not\models \psi$. It is known (and easy to check) that for any X-mapping f_X, the set of all the inaccessible formulas is (up to logical equivalence) the $\wedge$-closure of X.

Points 1 in Propositions 5.3 and 5.4 can be written respectively as follows, where $\equiv_{\overline{V}}$ means equivalent on the set of the formulas without symbol in V:

- $Circ(P, Q, V)(\varphi) \equiv_{\overline{V}} f_X(\varphi)$ where X is the set given in Proposition 4.4-1 [or alternatively any of the sets indicated in Proposition 4.4 (following)-1].
- $Circ(P, Q, V)(\varphi) \equiv_{\overline{V}} f_X(\varphi)$ where X is the set of all the clauses with literals in $P^+ \cup Q^{\pm}$.

It is clear from Remark 2.3 and Proposition 4.4 that these two "Points 1" are equivalent, as shown in [Lang *et al.*, 2003].

We will show now that, in Proposition 5.4 (the same considerations hold in Proposition 5.3) Point 2 is (almost) a direct consequence of Point 1, when using an old result about the elimination of varying propositions in circumscription thanks to variable forgetting [Lifschitz, 1994, Proposition

3.2.1] (first published in 1985):

$$Circ(P,Q,V)(\varphi) \models \psi \quad \text{iff} \quad Circ(P,Q,\emptyset)(ForgetV(\varphi,V)) \cup \{\varphi\} \models \psi \qquad (\text{Eq}_L)$$

Indeed, from (Eq$_L$), we get $Circ(P,Q,V)(\varphi) \models \psi$ iff there exists a formula $\neg\gamma$ without symbol in V such that $Circ(P,Q,\emptyset)(ForgetV(\varphi,V)) \models \neg\gamma$ and $\varphi \wedge \neg\gamma \models \psi$. Since $\neg\gamma$ is without symbol in V, this is equivalent to: $Circ(P,Q,V)(\varphi) \models \neg\gamma$ and $\varphi \models \gamma \vee \psi$. The only difference is that Przymusinski has improved this result a bit by restricting the set of the available formulas for γ, since P^- can be suppressed ("forgotten") also, and not only V.

The connexions between X-logic and circumscription are now well known [Suchenek, 1993; Siegel and Forget, 1996; Moinard and Rolland, 1998a]. It is convenient here to introduce a few technical notions.

Definition 5.6 *Let us call* positive in $\prec_{(P,Q,V)}$ *any formula φ whose set of models is finishing for $\prec_{(P,Q,V)}$: if $\mu \models \varphi$ and $\mu \prec_{(P,Q,V)} \nu$, then $\nu \models \varphi$.*

Directly from Definition 4.1 we get that $ForgetLitVar(\varphi,P,V)$ is positive in $\prec_{(P,Q,V)}$.

Notice that the "converse" also holds since each φ positive in $\prec_{(P,Q,V)}$ is equal to $ForgetLitVar(\varphi,P,V)$.

Example 2 (Example 1 following) *With φ, P, V, Q as in Example 1, we get clearly* **Mod**$(ForgetLitVar(\varphi,P^-,V)) = \{\nu \in$ **Mod** $/\ \mu \prec_{(P,Q,V)} \nu$ *or* $\mu = \nu$ *for some* $\mu \in$ **Mod**$(\varphi)\}$. *This set is clearly finishing for $\prec_{(P,Q,V)}$.*

With these technical indications, it would be easy to complete a direct proof of Theorem 5.2. However, a new proof is not necessary since Theorem 5.2 is an immediate consequence of Proposition 4.4 together with the following already known result:

Proposition 5.7 *[Moinard and Rolland, 1998b, Theorem 6.40] $Circ(P,Q,V) = f_X$ for any set X which has (up to logical equivalence) for its $\wedge$-closure the set of the formulas positive in $\prec_{(P,Q,V)}$.*

Let us provide an example in order to show how Theorem 5.2 works in practice.

Example 3 *P, Q, V are as in Example 1.*
$\varphi' = (b \wedge c) \vee (a \wedge \neg b \wedge \neg c \wedge \neg d)$ and
$\psi' = (\neg a \wedge b) \vee (a \wedge \neg b \wedge \neg c \wedge \neg d) \vee (\neg a \wedge c \wedge d)$.
We get that $\varphi' \wedge \psi'$ is equivalent to φ of Example 1.
Since we have $\varphi' \models (ForgetLitVar(\varphi,P^-,V))$, we conclude that we have $Circ(P,Q,V)(\varphi') \models \psi'$.

The $ForgetLitVar$ approach provides new insights into the effective methods for computing circumscriptions: In particular, it provides single stage process, instead of the two stages process of Propositions 5.3 and 5.4. It provides also a concrete way for computing circumscription thanks to Proposition 5.7, giving some flesh to the method for computing circumscription from their inaccessible formulas which has been described in [Moinard and Rolland, 1998a; 1998b].

6 Conclusion and perspectives

We have provided the semantical and several syntactical characterizations for a new notion, extending the notion of literal forgetting introduced in [Lang *et al.*, 2003] to the cases where some propositional symbols are allowed to vary. These results show that the new notion is not significantly harder than literal forgetting without varying symbols. The various characterizations provide various effective ways for computing the results, depending on the form in which the formulas appear.

Then, we have developed an application for computing circumscriptions. Doing this, we have reinforced the connexions (already found in [Lang *et al.*, 2003]) between the literal forgetting method and an older method from [Przymusinski, 1989]. Our work has allowed to transform the two stages methods which were known, either by explicit literal forgetting or by Przymunsinski's way, into simpler one stage methods. Doing so, we have shown that the literal forgetting method (thus also Przymunsinski's method) is strongly related to the method of X-mappings. These connexions provide a simpler presentation of various results, including the simplification (evoked above) of Przymunsinski's method for computing circumscription.

Notice that the "inaccessible formulas" of some non classical inference mechanism (built upon classical logic) have a clear meaning from the knowledge representation point of view. These formulas are the formulas which cannot be obtained as a new result (meaning not already obtained by classical inference) by using the inference mechanism. Thus, the description, provided here, of the way of computing circumscription from its inaccessible formulas, in terms of literal forgetting, provides new insights from the common sense reasoning point of view.
Also, the method described here could pave the way for more applications of the "forget literal method" towards already known formalisms. As written in [Lang *et al.*, 2003], results similar to the ones given here could be given for other closed world formalisms. Also, some of the various formalisms evoked in [Lang *et al.*, 2003, p. 413] could take profit from the introduction of varying propositional symbols. Moreover, since X-mappings can falsify the stability for $\wedge$ (in the default terminology, several extensions are possible), further generalizations of the notion of literal forgetting could possibly deal with some nonmonotonic formalisms falsifying the stability for $\wedge$.

An interest of the method of literal forgetting, since it consists in small and easy manipulations of propositional formulas, is that it can help the effective computation. As shown in [Lang *et al.*, 2003], we cannot hope that this will solve all the problems, but it should help in providing significant practical

improvements.

References

[Lang *et al.*, 2003] Jerome Lang, Paolo Liberatore, and Pierre Marquis. Propositional Independence - Formula-Variable Independence and Forgetting. *(Electronic) Journal of Artificial Intelligence Research*, 18:391–443, 2003. `http://WWW.JAIR.ORG/`.

[Lifschitz, 1994] Vladimir Lifschitz. Circumscription. In Dov M. Gabbay, C.J. Hogger, and J.A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 3: Non-Monotonic and Uncertainty Reasoning*, pages 297–352. Oxford University Press, 1994.

[McCarthy, 1986] John McCarthy. Application of circumscription to formalizing common sense knowledge. *Artificial Intelligence*, 28(1):89–116, February 1986.

[Moinard and Rolland, 1998a] Yves Moinard and Raymond Rolland. Circumscriptions from what they cannot do (Preliminary report). In *Working papers of Common Sense'98*, pages 20–41, London, January 1998.
`http://www.ida.liu.se/ext/etai/nj/fcs-98/`.

[Moinard and Rolland, 1998b] Yves Moinard and Raymond Rolland. Propositional circumscriptions. Technical report, INRIA, Research Report RR-3538, Rennes, France, October 1998.

[Przymusinski, 1989] Teodor C. Przymusinski. An Algorithm to Compute Circumscription. *Artificial Intelligence*, 38(1):49–73, February 1989.

[Siegel and Forget, 1996] Pierre Siegel and Lionel Forget. A representation theorem for preferential logics. In Luigia Carlucci Aiello, Jon Doyle, and Stuart Shapiro, editors, *KR'96*, pages 453–460, Cambridge, November 1996. Morgan Kaufmann.

[Suchenek, 1993] Marek A. Suchenek. First-Order Syntactic Characterizations of Minimal Entailment, Domain-Minimal Entailment, and Herbrand Entailment. *Journal of Automated reasoning*, 10:237–263, 1993.