Updates Under Pstable

Fernando Zacarías Flores[1] and Mauricio Osorio Galindo[2] and Edgar Fernández Plascencia[2]

***Abstract*—Based on a recent view of Pstable models that allows talking about knowledge and beliefs of an agent, we propose an extension of the AGM postulates based on these notions. To this extent we introduce a new principle of Irrelevance of Syntax. We present a slight version of the updated operator as defined under Pstable models that satisfies this principle.**

***Index Terms*—Pstable, G_3 logic, G'_3 logic, Update programs, Classical negation, AGM postulates.**

I. Introduction

The A-Prolog (Stable Logic Programming [1] or Answer Set Programming) is the realization of much theoretical work on Non-monotonic Reasoning and AI applications of Logic Programming (LP) in the last 15 years. This is an important logic programming paradigm that has now great acceptance in the community. Efficient software to compute answer sets and a large list of applications to model real life problems justify this assertion. The two most well-known systems that compute Answer sets are DLV [2] and SMODELS [3]. It has been recently provided a characterization of answer sets by intuitionistic logic as follows: a literal is entailed by a program in the stable model semantics if and only if it belongs to every intuitionistically complete and consistent extension of the program formed by adding only negated literals [16]. The idea of these completions using in general intermediate logics is due to Pearce [18].

However, there is a family of problems that cannot be modeled with this paradigm, particularly some argument problems. Due to this inconvenience in [17] the authors introduce a new paradigm called "Pstable model semantics" which shares several properties with A-Prolog, but it is closer to classical logic than stable. Pstable model semantics has at least the same expressiveness of A-Prolog, this means we can model anything that is modeled in A-Prolog. Pstable can model things, hard or may be impossible of model with A-Prolog. In this paper we use the implementation of Pstable model semantics based on definition 2 and presented in [20].

This logical approach provides the foundations to define the notion of nonmonotonic inference of any propositional theory (using the standard connectives) in terms of a monotonic logic (namely G_3 logic), see [15, 16, 18]. The proposed interpretation would be the following: Given a theory T, its knowledge is understood as the formulas F such that F is derived in T using G_3 logic. This makes sense, since in G_3 logic according to Brouwer, F is identified with "I know F" (or perhaps some reader would prefer to understand the notion of "knowledge" as "justified belief"). An agent whose knowledge base is the theory T believes F if and only if F belongs to every G_3 complete and consistent extension of T by adding only negated literals (here "belief" could be better interpreted as "coherent" belief). Take for instance: $\neg a \rightarrow b$. The agent knows $\neg a \rightarrow b$, $\neg b \rightarrow \neg\neg a$ and so on. The agent does not know however a. Nevertheless, one believes more than one knows, but a cautious agent must have its beliefs consistent to its knowledge. This agent will then assume negated literals to be able to infer more information. Thus, in our example, our agent will believe $\neg a$ and so it can conclude b. It also makes sense that a cautious agent will believe $\neg a$ or $\neg\neg a$ rather than to believe a (recall that a is not equivalent to $\neg\neg a$ in G_3 logic). This view seems to agree with a point of view by Kowalski, namely "that Logic and LP need to be put into place: Logic within the thinking component of the observation-thought-action cycle of a single agent, and LP within the belief component of thought" [11].

We propose to reconsider the AGM postulates [1] under our new interpretation that considers "knowledge" and "belief". We propose the new postulate which we call Weak Irrelevance of Syntax as follows:

$$\text{(WIS): } T_1 \equiv_{G'_3} T_2 \text{ implies } Bel(K \nabla T_1) = Bel(K \nabla T_2).$$

We show that the proposal shown in [6] for update almost satisfies this principle and we propose a slight version of his operator that indeed satisfies WIS. Our ultimate goal is to question the current interpretations of the AGM postulates and motivate a better understanding of the update operators considering the logical framework that supports Pstable models via G_3 logic.

Our paper is structured as follows: In section 2 we give pstable model semantics. Immediately, we present G3 and G'3 logics to that they allow us to conserve our property WIS. Next, in section 4, we extend our programs with strong negation. In this order of ideas, in section 5, we present our analysis with respect to irrelevance of syntax. Continuing with this idea, in next section 6 we present our new proposal, giving our new proposal about update process under Pstable semantics. Finally, in 8 we give our conclusions and future work.

Paper submmited on February 26, 2007. Benemérita Universidad Autónoma de Puebla – Facultad de Ciencias de la Computación, Puebla, Pue., México, C.P. 72570, moderm_cruzades@hotmail.com[1], fzflores@yahoo.com.mx[2].

II. PSTABLE MODEL SEMANTICS

In this section, we introduce some concepts related with the semantics of Pstable and its interpretation.

Pstable Model Semantics

The definition of the pstable model semantics was shown in [17] and is presented here, and, in the same form as in [17] we show the next reduction.

Definition 1. [17] Let P be a normal program and *M* a set of atoms:

$$Red_M(P) = \{\ a \leftarrow \beta^+ \wedge \neg(\beta^- \cap M) \mid a :- \beta^+ \wedge \neg\beta^- \in P\}$$

and where β^+ and β^- are sets containing, respectively, the positive and negative atoms that occur in the body of the clauses of P.

In other words, with this reduction every atom that is not in the model *M* and is negated at the body of a rule is deleted.

Definition 2. [17] Let *P* be a normal program, and *M* a set of atoms. We say that *M* is a pstable model of *P* if *M* is a model of *P* and $Red_M(P) \vdash M$.

Textual Implication

This term is used when from the semantic of a text in natural language, is possible to infer, another text in natural language. More specific, if the true of one sentence implies the true of the other one, also call hypothesis [3].

III. G_3 AND G'_3 USING £$_3$

In this section we define G_3 and G'_3 using £$_3$. £$_3$ is defining by Lukasiewicz. He argues that if we wonder if future events are true or false, then we pretend that the future is so certain one as the past and it differs of this alone in that not yet has happened. Its way to escape from this bog deterministic was rejecting the law of the third excluded, i.e., that each proposition is true or false, a third value is added, which is read as possible. To build this logic £$_3$ it is considered a formal language (propositional) that it contains a numerable set of atomic formulas, L, the binary connectives, $\rightarrow_L$, $\vee_L$, $\wedge_L$, the unary connectives $\neg_L$, $\Diamond_L \Box_L$ (the last two well-known as modal operators) and the logical constant $\perp$ whose meaning is of falsehood and, therefore it is identified as zero (0). Considering to $\rightarrow_L$ and $\perp$ as basics the rest of the connectives can be seen as the following abbreviations:

$$\begin{aligned} \top &:= (p \rightarrow_L p) \\ \alpha \wedge_L \beta &:= \neg_L(\neg_L \alpha \vee_L \neg_L \beta) \\ \alpha \vee_L \beta &:= (\alpha \rightarrow_L \beta) \rightarrow_L \beta \\ \Diamond_L \alpha &:= \neg_L \alpha \rightarrow_L \alpha \\ \neg_L \alpha &:= \alpha \rightarrow_L \perp \\ \Box_L \alpha &:= \neg_L(\alpha \rightarrow_L \neg_L \alpha) \end{aligned}$$

This way the evaluation of the connectives $\neg_L$ and $\rightarrow_L$ is as it is shown in the figure 2.1, while that $a \vee_L b = \max\{a, b\}$ and $a \wedge_L b = \min\{a, b\}$.

a	$\neg_L$ a
0	2
1	1
2	0

$\rightarrow_L$	0	1	2
0	2	2	2
1	1	2	2
2	0	1	2

Figure 2.1 Evaluation of the connectives $\neg_L$ and $\rightarrow_L$

The only designated value is 2 therefore the tautologies £$_3$ in will be those formulas that evaluate at 2 for any possible interpretation. In [19] a syntactic characterization of the modal content of £$_3$, is given. Also, the behavior of the modal operators is studied and it is verified which modal principles satisfy.

We will build two logics of three values defining their basic connectives, negation and implication as abbreviations in the language £$_3$ according to figure 2.2.

$$\begin{aligned} \neg_{G3}\, a &:= \Box_L \neg_L a \\ a \rightarrow_{G3} b &:= (a \rightarrow_L b) \wedge_L \neg_{G3}\neg_{G3}(\neg_{G3}\neg_{G3}\, a \rightarrow_L b) \\ \neg_{G'3}\, a &:= \neg_L \Box_L a \\ a \rightarrow_{G'3} b &:= a \rightarrow_{G3} b \end{aligned}$$

Figure 2.2 Abbreviations in the language £$_3$

In to figure 2.3 we can observe the evaluations of the connectives: $\neg_{G3}$, $\neg_{G'3}$, y $\rightarrow_{G3}$.

We have two different negations, considering each one of them we build a different logic: G_3 o HT (are equivalents), with the connectives $\neg_{G3}$ and $\rightarrow_{G3}$ (you can see the relation between this logic and Stable in [16, 22]), and the logic G'_3 with the connectives $\neg_{G'3}$ and $\rightarrow_{G3}$. The disjunction and de conjunction for G_3 and G'_3 are the same ones that those of £$_3$.

a	$\neg_{G3}$ a	$\neg_{G'3}$ a
0	2	2
1	0	2
2	0	0

$\rightarrow_{G3}$	0	1	2
0	2	2	2
1	0	2	2
2	0	1	2

Figure 2.3 Evaluation of the connectives $\neg_{G3}$, $\neg_{G'3}$ and $\rightarrow_{G3}$

Definition 3. [17] Let be P a normal program, M a set of atoms. M is a G'_3 stable model of P if only if M is a model of P in classical logic and $\neg(L_P \setminus M) \cup P \vdash M$.

IV. Adding strong negation

We extend the language adding strong negation (denoted by "~"). Syntactically, the status of the strong negation operator $ "~" is different from the status of the operator "¬" the difference is the following:

"$\neg p$" can be denoted by "$p \rightarrow \perp$", i.e., we use "¬" when evidence doesn't exist about p. In the same form, we use "$\sim p$" when we know that p is false or it doesn't happen. We can say that pstable models are usually defined for logic programs possessing this second kind of negation that as we mentioned previously expresses the direct or explicit falsity of an atom.

A literal, L, is either an atom A (a positive literal) or a strongly negated atom $\sim A$ (a negative literal). For a literal L, the *complementary literal*, $\sim L$, is $\sim A$ if $L = A$, and A if $L = \sim A$, for some atom A. For a set S of literals, we define $\sim S = \{\sim L \mid L \in S\}$, and denote by Lit_A the set $A \cup \sim A\}$ for all literals over A. A literal preceded ¬ is called a *weakly negated literal.*

The concept of modeling ($I \models P$) is extended to include strong negation as explained in [6]. The concept of pstable model can be extended in a similar way, see [9].

V. Analysis of Irrelevance of Syntax

We show that the postulates (IS) and (WIS) fail for the update operator ⊕ defined in [6]. Firstly, let us analyze an example where given two equivalent programs P_1 and P_2 and P is another program. The update $P \oplus P_1 \equiv P \oplus P_2$ fails.

Example 1: Let P_1 and P_2 be two equivalent programs and let P be another program then $P \oplus P_1 \equiv P \oplus P_2$ is false. Let $P_1 = \{a \leftarrow b\}$, $P_2 = \{a \leftarrow a, b \leftarrow b\}$ and $P = \{b\}$
Both programs P_1 and P_2 have exactly one Pstable model, namely the empty set.

$U(P \oplus P_1) = \{\{a, b\}\}$ and $U(P \oplus P_2) = \{\{b\}\}$

therefore $\{\{a, b\}\} \neq \{\{b\}\}$

Hence, $P \oplus P_1 \equiv P \oplus P_2$ fails.

It is necessary to point out that in [6] update programs do not satisfy many of the properties defined in the literature. This is partly explained by the nonmonotonicity of logic programs and the causal rejection principle embodied in the semantics, which strongly depends on the syntax of rules. Now, we present an example where WIS also fails.

Example 2: Let $P = \{\sim d, d \leftarrow h\}$, let $P_1 = \{h, d \leftarrow d\}$ and let $P_2 = \{h\}$, clearly, $P_1 \equiv P_2$.

More specific, if the true of one sentence implies the true of the other one, also call hypothesis [3].

VI. NEW PROPOSAL

As we have mentioned, the interpretation given in [6] of the AGM postulates express a very demanding principle of irrelevance of syntax, because the AGM postulates were introduced for monotonic logics. We present our proposal about update process based on Pstable model semantics..

Definition 4: Giving an update of two programs $P_\otimes = (P_1, P_2)$ over a set of atoms A, we define the update program $P_\otimes = P_1 \otimes P_2$ = over A^* consisting of the following items:

(i) all constraints in $P_1 \cup P_2$;
(ii) for each $r \in P_1$, $L \leftarrow B(r), \neg\sim L$. if $H(r) = L$;
(iii) all rules $r \in P_2$.

Considering example given in [6] and applying our definition 4 we obtain:

$P_1 \otimes P_2$:
sleep ← ¬ tv-on, ¬~sleep.
night ← ¬~night.
tv-on ← ¬~tv-on.
watch-tv ← tv-on, ¬~watch-tv.
~tv-on ← power-failure, power-failure.

Observe that this program has the unique expected Pstable model, namely {power-failure, ~tv-on, sleep, night}.

Now, considering example 2 again and applying our proposal we obtain that:

$P \otimes P_1$:
~d ← ¬d.
d ← h, ¬~d, h.
d ← d.

notice that the rule ~d should be transformed to ~d ← ¬ ~~d but, ~~d is equivalent to d. Furthermore, this program has the Pstable models {h, ~d} and {h, d}.

On the other hand,
$P \otimes P_2$:
~d ← ¬d.
d ← h, ¬~d.
h.

this program has the same Pstable models {h, ~d} and {h, d} as
$P \otimes P_1$.

Example 3: We analyze another interesting example. Consider that you have an inconsistent program (say with a fact for both a and $\sim a$), and update it with an empty theory. Then, the program surprisingly becomes *consistent!*. We have two interpretations for this example. First, we are supposing that we begin with a consistent program. Second, our proposal allows to make a reflection process, showing us that indeed, or it happens *a* or *~a* happens, but not both, what is correct.

Definition 5: We say that *P* is *tau-comp* w.r.t. a signature $\mathcal{A}$ if every rule of the form l← l belongs to *P,* where l is a literal over $\mathcal{A}$.

The following lemma shows that our approach is closely related to [6]. The proof is based on results from [15, 16, 17].

Lemma 1: Let *P* and P_1 be programs, if P_1 is *tau-comp* then $P \oplus P_1 \equiv P \otimes P_1$.

Proof: The idea of the proof is to apply transformations to $P \oplus P_1$ with respect to Pstable models over the the $\mathcal{A}$ signature, to obtain $P \otimes P_1$.
Consider a particular literal *L*. By construction of $P \oplus P_1$ (defined in point iv of definition 2 given in [6]) the program include the formulas

$L_1 \leftarrow L_2$ *and* $L_1 \leftarrow L_1$. for the literal *L*.

Also, since P_1 is *tau-comp* then it includes the rules $L_2 \leftarrow L$ for the literal *L*.

Hence, $L_1 \leftrightarrow L_2 \leftrightarrow L$ is derived by $P \oplus P_1$.

So we can replace each literal L_1 by *L* and L_2 by *L* in the rest of the program. Then, we can eliminate rules in point iv of definition 2 given in [6] as well as rules of the form $L_2 \leftarrow L$ (recall that L_1 and L_2 are temporal literals).

So, the rules become
$L \leftarrow B(r), \neg rej(r)$ *and*
$rej(r) \leftarrow B(r), {\sim}L$ corresponding to program *P* *and*
$L \leftarrow B(r)$ corresponding to program P_1.

Now, observe that the Pstable models of the program are the same if we replace

$rej(r) \leftarrow B(r),\ {\sim}L$ *by* $rej(r) \leftrightarrow (B(r) \wedge {\sim}L)$

because *rej(r)* occurs only once as a head of a rule and since the original program is normal. Observe, that this transformation creates a non normal rule, however this situation is already considered in [16, 17]. Then we can replace *rej(r)* in $L \leftarrow B(r) \wedge \neg rej(r)$ to obtain

$$L \leftarrow B(r) \ \wedge\ \neg(B(r) \wedge {\sim}L) \qquad (*)$$

then we can delete the rule $rej(r) \leftrightarrow (B(r) \wedge {\sim}L)$ since *rej(r)* is a temporal literal. Finally, the rule of the form (*) can be replaced by $L \leftarrow B(r), \neg{\sim}L$. We can repeat this process iteratively for the rest of the literals, reaching program $P \otimes P_1$ as desired. The next theorem is a simple corollary of results in [12].

Theorem 1. For any P_1 and P_2 programs, $P_1 \equiv_{G_3'} P_2$ implies that for every P program, $P_1 \cup P$ and $P_2 \cup P$ have the same Pstable models.

Next, we present our main result with respect to our update postulate.

Theorem 2. Our update operator ⊗ satisfies the following postulate:

$$\text{If } P_1 \equiv_{G_3'} P_2 \text{ then } P * P1 \equiv P * P_2 \qquad \text{(WIS)}$$

Proof: This postulate follows straightforward by construction and theorem 1

We now show how the proposal in [6] almost satisfies WIS.

Corollary 1. Let *P* be a program and let P_1 and P_2 be *tau-comp* programs.

$$\text{If } P_1 \equiv_{G_3'} P_2 \text{ then } P \oplus P_1 \equiv P \oplus P_2.$$

Proof: We have that $P \oplus P_1 \equiv P \otimes P_1$ by lemma 1
now, $P \otimes P_1 \equiv P \otimes P_2$ by theorem 2
finally, $P \otimes P_2 \equiv P \oplus P_2$ by lemma 1

Finally we present the following example shown in [7].

Example 4: Next, we present our codification in ASP using our proposal.

Let *P* trap(a).
sweden(b).
swan(b).
white(X) ← swan(X), european(X).
swan(X) ← trap(X).
sweden(X) ← trap(X).
european(X) ← sweden(X)}

If we update *P* with next rule:
~white(a).

Applying our proposal we obtain:
trap(a) ← ¬~trap(a).
swan(b) ← ¬~swan(b).
sweden(b) ← ¬~sweden(b).
white(X) ← swan(X), european(X), ¬~white(X).
swan(X) ← trap(X), ¬~swan(X).
sweden(X) ← trap(X), ¬~sweden(X).
european(X) ← sweden(X), ¬~european(X).
~white(a).

Then, the Pstable model is, as desired:

{~white(a), trap(a), sweden(a), sweden(b), swan(a), swan(b), white(b), european(a), european(b)}.

As we can see this proposal satisfies that all European swans are white and the bird caught in the trap is a swan enunciated in [1]. Since, '*b*' is swan and Sweden, therefore is white. While 'a' is the exception, because 'a' is in trap, therefore is not white.

VII. Conclusions

In this paper, we considered a formalization of an update operator for Pstable model semantics. Also, we use in our examples the implementation of Pstable models presented in [20]. Different from other approaches we considered a view of Pstable models based on G_3 and G'_3 logic. This allowed us to reconsider the AGM postulates in a more solid framework. We also tried to stay as close as possible to the well-known proposal for updates given in [6]. Our main contribution is our postulate called WIS but, now in G_3 and G'_3 logics. Also, we extend Pstable with strong negation. This is supported by our theorem 2, and it opens new opportunities in the update and belief revision field.

As a future work, we consider to extend all our work related with updates and presented in [21] to Pstable model semantics.

References

[1] C.E. Alchourron, P. Gardenfors, and D. Makinson. On the logic of Theory Change,Partial Meet Functions for Contraction and Revision Functions. Journal of Symbolic Logic, vol. 50 pp. 510-530, 1985.

[2] J. Alferes, J. Leite, P. Pereira, H. Przymusinska, and T. Przymusinski. Dynamic Logic Programming. In A. Cohn and L. Schubert, editors. Proc. KR98, pp. 98-109. Morgan Kaufmann, 1998.

[3] G. Brewka. Declarative Representation of Revision Strategies. In Proc. Fourteenth European Conference on Artificial Intelligence (ECAI 2000), 2000.

[4] G. Brewka, J. Dix, and K. Knonolige. Nonmonotonic Reasoning: An overview. CSLI Publication Eds. Leland Stanford Junior University, 1997.

[5] A. Darwiche and J. Pearl. On the Logic of Iterated Belief Revision. Artificial Intelligence, vol.89(1-2): pp. 1-29, 1997.

[6] T. Eiter, M. Fink, G. Sabattini, and H. Thompits. Considerations on Updates of Logic Programs. In M.O. Aciego, L.P. de Guzmßn, G. Brewka, and L.M. Pereira, editors, Proc. Seventh European Workshop on Logic in Artificial Intelligence JELIA 2000, vol. 1919 in LNAI, Springer 2000.

[7] P. Gardenfors. Belief Revision : An Introduction. Cognitive Science, Department of Philosophy, Lund University, S-223, 50 Lund, Sweden, 1995.

[8] M. Gelfond and V. Lifschitz. The stable model semantics for logic programs. Proceedings of the Fifth International Conference on Logic Programming 2 MIT Press. Cambridge, Ma. pp.1070-1080.

[9] M. Gelfond and V. Lifschitz. Clasical negation in logic programs and Disjunctive databases, New Generation Computing.pp. 365-387, 1991.

[10] H. Katsumo and A.O. Mendelzon. Propositional knowledge base revision and minimal change, Artificial Intelligence vol. 52, pp. 263-294, Elsevier, 1991.

[11] R. Kowalski. Is logic really dead or just sleeping. In Proceedings of the 17th International Conference on Logic Programming, pages 2-3, 2001.

[12] V. Lifschitz, D. Pearce, and A. Valverde. Strongly equivalent logic programs. ACM Transactions on Computational Logic, 2:526-541, 2001.

[13] V. Lifschitz and T.Y.C. Woo. Answer sets in general nonmonotonic reasoning. In proc. Of IJCAI-91, 1991.

[14] W. Marek and V. Subrahmanian. The relationship Between Logic Program Semantics and Non-monotonic Reasoning, in G. Levi and M. Martelli (eds.), Proc. of the 6. Int. Conf. on Logic Programming. MIT, 600-617, 1989.

[15] M. Osorio, J.A. Navarro, and J. Arrazola. Equivalence in Answer Set Programming (extended version), Proceedings of LOPSTR 01, LNCS 2372, pp.57-75, Springer-Verlag, Paphos, Cyprus, November 2001.

[16] M. Osorio, J.A. Navarro, and J. Arrazola. Applications of Intuitionistic Logic in Answer Set Programming, Journal of TPLP , 2003.

[17] Mauricio Osorio Galindo and Juan Antonio Navarro Pérez and José R. Arrazola, Rodríguez and Verónica Borja Macías". Logics with common weak completions. Journal of Logic and Computation, 2006. URL doi: 10.1093/logcom/exl013

[18] D. Pearce. From Here to There: Stable negation in Logic Programming, in D. Gabbay, H. Wansing Eds. What is Negation? Kluwer Academic Publishers, Dordrecht, forthcoming. 1999.

[19] Minari, Pierluigi; A note on £ukasiewicz three-valued logic. Annali del Dipartimento di Filosofia dell'Università di Firenze. Florencia, 2003.

[20] Uceda P. Fernando and Zacarias F. Fernando. Pstable tool and its implementation. Submitted to special issue on Artificial Intelligence and Computer Science in Journal of Engineering letters, Spain, 2007.

[21] Zacarias Flores F. Ph. D thesis presented in UDLAP, 2004.

[22] Zakharyaschev, M.; Wolter, F.; Chagrov, A.; Advanced Modal Logic. In D. M. Gabbay and F. Guenthner, editors, Handbook of Philosophical Logic, Kluwer Academic Publishers, Dordrecht. 2001.