AGM-Style Belief Revision of Logic Programs under Answer Set Semantics*

James Delgrande[1], Pavlos Peppas[2], and Stefan Woltran[3]

[1] School of Computing Science
Simon Fraser University
Burnaby, B.C.
Canada V5A 1S6

[2] Dept of Business Administration
University of Patras
Patras 265 00, Greece

[3] Institut für Informationssysteme
Technische Universität Wien
Favoritenstraße 9–11
A–1040 Vienna, Austria

Abstract. In the past few years, several approaches for revision (and update) of logic programs have been studied. None of these however matched the generality and elegance of the original AGM approach to revision in classical logic. One particular obstacle is the underlying nonmonotonicity of the semantics of logic programs. Recently however, specific revision operators based on the monotonic concept of SE-models (which underlies the answer-set semantics of logic programs) have been proposed. Basing revision of logic programs on sets of SE-models has the drawback that arbitrary sets of SE-models may not necessarily be expressed via a logic program. This situation is similar to the emerging topic of revision in fragments of classical logic. In this paper we show how nonetheless classical AGM-style revision can be extended to various classes of logic programs using the concept of SE-models. That is, we rephrase the AGM postulates in terms of logic programs, provide a semantic construction for revision operators, and then in a representation result show that these approaches coincide. This work is interesting because, on the one hand it shows how the AGM approach can be extended to a seemingly nonmonotonic framework, while on the other hand the formal characterization may provide guiding principles for the development of specific revision operators.

1 Introduction

Answer set programming [5] is an appealing approach for representing problems in knowledge representation and reasoning. It has a conceptually simple theoretical foundation, while at the same time being applicable to a wide range of practical problems. As well, efficient ASP systems have become available. However, a logic program is not

* This work was partially supported by a Canadian NSERC Discovery Grant and by the Austrian Science Fund (FWF) under grant P25521-N23.

a static object in general, but rather it will evolve and be subject to change, whether as a result of correcting information in the program, adding to the information already present, or in some other fashion modifying the knowledge represented in the program.

In classical logic, the problem of handling knowledge *in flux* has been thoroughly investigated (see [24] for an overview). The seminal AGM approach [1, 17], provides a general, elegant, and widely accepted framework for this purpose. Central to this approach are powerful representation theorems, which aim to characterize *all* rational operators satisfying certain postulates. Its generality, emphasizing logical formalization, syntax independence, and minimal change, has made this approach a standard for investigating problems concerned with revision or update of knowledge bases, regardless of the underlying semantics (see, for instance, [16] for AGM-style revision in terms of description logics).

Although there has been very active research in revision (and update) of logic programs [31, 2, 20, 25, 14], the generality of the original AGM approach has not been matched yet in any of these approaches. One obstacle is the underlying nonmonotonicity of the semantics of logic programs, which has led to the study of postulates different from the ones in the AGM approach, see e.g. [14, 23]. Recently however, specific revision operators based on the monotonic concept of SE-models [28] (which underlies the answer-set semantics of logic programs [21]) have been proposed [12] together with a suitable variant of the AGM postulates; see also [27] for a variant thereof. In recent work [19], the notion of SE-models (and similar concepts) has been put in connection to postulates for belief base change different from AGM.

However, representation theorems are still lacking. One problem is that arbitrary sets of SE-models may not necessarily be expressed via a logic program. (While such a requirement is crucial for a representation theorem, it is not problematic in classical logic, at least in the finite case, since for any set of interpretations there is a set of formulas having these interpretations as its models.) A similar challenge arises in another emerging topic in the area of belief change, namely revision in fragments of classical logic, see e.g. [10, 7]. To be more specific, consider the problem of revision in the Horn fragment of classical logic. Since the models of Horn formulas satisfy a certain closure property, the result of a revision requires that this property is obtained in order to be represented in the Horn fragment, too. For representation theorems, it turned out that one needs to suitably integrate this property to the concept of faithful assignments [18] and to add an additional postulate [10].

In this paper we show how classical AGM-style revision can be extended to various classes of logic programs using the concept of SE-models. We give representation theorems for the AGM-style postulates proposed in [12] by exploiting, first, the recent techniques for Horn revision due to [10] and, second, the properties that program classes enjoy in terms of SE-models [13]. This allows us to give representation theorems for the important classes of disjunctive (generalized and ordinary), normal, positive, and Horn logic programs. This work is interesting because, on the one hand it shows how the AGM approach can be extended to a seemingly nonmonotonic framework, while on the other hand the formal characterization may provide guiding principles for the development of specific revision operators beyond the ones suggested in [12].

The remainder of the paper is organized as follows. The next section reviews answer set programming and belief revision, and surveys previous work in belief change in logic programs. The following section examines the problems that arise in a direct application of the AGM approach to answer set programming. Section 4 provides the main formal results, comprising representation theorems for each of the major classes of logic programs. The next section shows that the approach is compatible with iterated revision, while the last section is a summary. Proofs of theorems are omitted due to space considerations but are available on request.

2 Background and Formal Preliminaries

2.1 Answer Set Programming

Let $\mathcal{A}$ be a finite *alphabet* or set of propositional variables. A (*generalised*) *logic program* (GLP) over an alphabet $\mathcal{A}$ is a finite set of rules of the form

$$a_1; \ldots ; a_m;\ \sim b_1; \ldots ; \sim b_n \ \leftarrow\ c_1, \ldots , c_j,\ \sim d_1, \ldots , \sim d_k \qquad (1)$$

where $a_p, b_q, c_r, d_s \in \mathcal{A}$ and $p, q, r, s \geq 0$. Operators ';' and ',' express disjunctive and conjunctive connectives. A *(default) literal* is an atom a or its (default) negation $\sim a$. A rule r as in (1) is a *fact* if $m = 1$ and $n = j = k = 0$, and an *integrity constraint* if $m = n = 0$, yielding an empty disjunction denoted by $\perp$. $\mathcal{LP}$ will denote the set of generalised logic programs. Unless stated otherwise, *logic program* will refer to a GLP.

A rule r as in (1) is called *disjunctive* if $n = 0$; *normal* if $m \leq 1$ and $n = 0$; or *positive* if $n = k = 0$. A program is a *disjunctive logic program* (DLP) if it consists of disjunctive rules only. A program is a *normal logic program* (NLP) if it consists of normal rules only. For completeness, we also consider *positive logic programs* (PLP), consisting of positive rules, and *Horn logic programs* (HLP), consisting of rules that are both positive and normal.

We define $H(r) = \{a_1, \ldots , a_m, \sim b_1, \ldots , \sim b_n\}$ as the *head* of r and $B(r) = \{c_1, \ldots , c_j, \sim d_1, \ldots , \sim d_k\}$ as the *body* of r. Given a set X of literals, $X^+ = \{a \in \mathcal{A} \mid a \in X\}$, $X^- = \{a \in \mathcal{A} \mid \sim a \in X\}$, and $\sim X = \{\sim a \mid a \in X\}$. For simplicity, we sometimes use a set-based notation, expressing a rule as in (1) as $H(r)^+; \sim H(r)^- \leftarrow B(r)^+, \sim B(r)^-$.

An interpretation is represented by the subset of atoms in $\mathcal{A}$ that are true in the interpretation. A (*classical*) *model* of a program P is an interpretation in which all of the rules in P are true according to the standard definition of truth in propositional logic, and where default negation is treated as classical negation. $Mod(P)$ denotes the set of classical models of P. The *reduct* of a program P with respect to a set of atoms Y, denoted P^Y, is the set of rules:

$$\{H(r)^+ \leftarrow B(r)^+ \mid r \in P,\ H(r)^- \subseteq Y,\ B(r)^- \cap Y = \emptyset\}.$$

Note that the reduct consists of negation-free rules only. An *answer set* Y of a program P is a subset-minimal model of P^Y. The set of all answer sets of a program P is denoted by $AS(P)$. For example, the program $P = \{a \leftarrow,\quad c; d \leftarrow a, \sim b\}$ has answer sets $AS(P) = \{\{a, c\}, \{a, d\}\}$.

An *SE interpretation* [28] is a pair (X, Y) of interpretations such that $X \subseteq Y \subseteq \mathcal{A}$. The set of all SE interpretations (over $\mathcal{A}$) is denoted by $\mathscr{SE}$. For simplicity, we often drop set-notation within SE interpretations and simply write, e.g., (a, ab) instead of $(\{a\}, \{a, b\})$. An SE interpretation is an *SE model* of a program P if $Y \models P$ and $X \models P^Y$. The set of all SE models of a program P is denoted by $SE(P)$. Note that Y is an answer set of P iff $(Y, Y) \in SE(P)$ and for every $X \subset Y$, $(X, Y) \notin SE(P)$. Also, we have $(Y, Y) \in SE(P)$ iff $Y \in Mod(P)$.

A program P is *satisfiable* just if $SE(P) \neq \emptyset$. Two programs P and Q are *strongly equivalent*, symbolically $P \equiv_s Q$, iff $SE(P) = SE(Q)$. Alternatively, $P \equiv_s Q$ holds iff $AS(P \cup R) = AS(Q \cup R)$, for every program R [21]. We write $P \models_s Q$ iff $SE(P) \subseteq SE(Q)$. This means that P is satisfiable iff $P \not\models_s \bot$.

One feature of SE models is that they contain "more information" than answer sets, which makes them an appealing candidate for problems where programs are examined with respect to further extension (in fact, this is what strong equivalence is about). We illustrate this issue with the following well-known example, involving programs

$$P = \{p; q \leftarrow\} \quad \text{and} \quad Q = \left\{ \begin{array}{l} p \leftarrow {\sim}q \\ q \leftarrow {\sim}p \end{array} \right\}.$$

Here, we have $AS(P) = AS(Q) = \{\{p\}, \{q\}\}$. However, the SE models (we list them for $\mathcal{A} = \{p, q\}$) differ:

$$\begin{array}{l} SE(P) = \{(p, p), (q, q), (p, pq), (q, pq), (pq, pq)\}; \\ SE(Q) = \{(p, p), (q, q), (p, pq), (q, pq), (pq, pq), (\emptyset, pq)\}. \end{array}$$

This is to be expected, since P and Q behave differently with respect to program extension, and thus are not strongly equivalent. Consider $R = \{p \leftarrow q, q \leftarrow p\}$. Then $AS(P \cup R) = \{\{p, q\}\}$, while $AS(Q \cup R)$ has no answer set.

Next, we recall several properties the set of SE-models satisfies for certain program classes. These properties when suitably combined characterize a logic program class $\mathcal{C}$ in a necessary (for any program $P \in \mathcal{C}$, $SE(P)$ satisfies certain properties) and sufficient (for each S satisfying these properties, there exists a $P \in \mathcal{C}$, such that $SE(P) = S$) way; see [15, 6] and the overview [13]. The properties we require are as follows: A set S of SE interpretations is *well-defined* if, for each $(X, Y) \in S$, also $(Y, Y) \in S$. A well-defined set S of SE interpretations is *complete* if, for each $(X, Y) \in S$, also $(X, Z) \in S$, for any $Y \subseteq Z$ with $(Z, Z) \in S$. A complete set S of SE interpretations is *closed under here-intersection* if, for each $(X, Z), (Y, Z) \in S$ also $(X \cap Y, Z) \in S$. A complete set S of SE interpretations is *positive definable* if, for each $(X, Y) \in S$, also $(X, X) \in S$. Last, a positive definable set S of SE interpretations is *Horn definable* iff $(X_1, Y_1), (X_2, Y_2) \in S$ implies that $(X_1 \cap X_2, Y_1 \cap Y_2) \in S$. Intuitively, these properties capture inherent features that the reducts of program classes enjoy. For instance, for any positive program and any interpretation Y, it holds that $P^Y = P$, as mirrored by the concept of positive definable. We have the following results, c.f. [13].

– For each GLP P, $SE(P)$ is well defined.
– For each DLP P, $SE(P)$ is complete.

– For each NLP P, $SE(P)$ is closed under here-intersection.
– For each PLP P, $SE(P)$ is positive definable.
– For each HLP P, $SE(P)$ is Horn definable.

Moreover, for a set S of SE interpretations:

– if S is well defined, there exists a GLP P such that $SE(P) = S$;
– if S is complete, there exists a DLP P such that $SE(P) = S$;
– if S is closed under here-intersection, there exists a NLP P such that $SE(P) = S$;
– if S is positive definable, there exists a PLP P such that $SE(P) = S$; and
– if S is Horn definable, there exists a HLP P such that $SE(P) = S$.

Consequently, for a set of SE models S, we define $t(S)$ to be a least (with respect to SE models) logic program whose SE models contain S. Note that such a program is unique, up to strong equivalence. We also overload notation and in the case that W is a set of classical interpretations, we define $t(W)$ to be a formula of propositional logic whose models are exactly W. In both cases, since the alphabet is finite, $t(\cdot)$ is guaranteed to exist.

2.2 Belief Revision

The best known work in belief revision is the *AGM approach* [1, 17], in which standards for belief *revision* and *contraction* functions are given. In belief revision, a formula is added to a knowledge base such that the resulting knowledge base is consistent (unless the formula to be added is inconsistent). In the AGM approach it is assumed that a knowledge base receives information concerning a static domain. Belief states are modeled by logically closed sets of sentences, called *belief sets*. Thus, a belief set is a set K of sentences which satisfies the constraint

$$\text{if } K \text{ logically entails } \beta, \text{ then } \beta \in K.$$

K can be seen as comprising a partial theory of the world. For belief set K and formula α, $K + \alpha$ is the deductive closure of $K \cup \{\alpha\}$, called the *expansion* of K by α. $K_\perp$ is the inconsistent belief set (i.e., $K_\perp$ is the set of all formulas).

Subsequently, Katsuno and Mendelzon [18] reformulated the AGM approach so that a knowledge base was represented by a formula in some language $\mathcal{L}$. The following postulates comprise Katsuno and Mendelzon's reformulation of the AGM revision postulates, where $*$ is a function from $\mathcal{L} \times \mathcal{L}$ to $\mathcal{L}$:

(R1) $\psi * \mu \vdash \mu$.
(R2) If $\psi \wedge \mu$ is satisfiable, then $\psi * \mu \leftrightarrow \psi \wedge \mu$.
(R3) If μ is satisfiable, then $\psi * \mu$ is also satisfiable.
(R4) If $\psi_1 \leftrightarrow \psi_2$ and $\mu_1 \leftrightarrow \mu_2$, then $\psi_1 * \mu_1 \leftrightarrow \psi_2 * \mu_2$.
(R5) $(\psi * \mu) \wedge \phi \vdash \psi * (\mu \wedge \phi)$.
(R6) If $(\psi * \mu) \wedge \phi$ is satisfiable, then $\psi * (\mu \wedge \phi) \vdash (\psi * \mu) \wedge \phi$.

Katsuno and Mendelzon also show that a necessary and sufficient condition for constructing an AGM revision operator is that there is a function that associates a total preorder on the set of interpretations with any formula ψ, as follows:

Definition 1. *A* faithful assignment *is a function that maps each formula ψ to a total preorder $\preceq_\psi$ on the set of interpretations $\mathcal{M}$ such that for any interpretations m_1, m_2:*

1. *If m_1, $m_2 \in Mod(\psi)$ then $m_1 \approx_\psi m_2$*
2. *If $m_1 \in Mod(\psi)$ and $m_2 \notin Mod(\psi)$, then $m_1 \prec_\psi m_2$.*
3. *If $\psi \leftrightarrow \mu$ then $\preceq_\psi = \preceq_\mu$.*

The resulting preorder is referred to as a *faithful ranking* associated with ψ. Intuitively, $m_1 \preceq_\psi m_2$ if m_1 is at least as plausible as m_2 with respect to ψ. Katsuno and Mendelzon then provide the following representation result.

Theorem 1 ([18]). *A revision operator * satisfies postulates (R1)–(R6) iff there exists a faithful assignment that maps each formula ψ to a total preorder $\preceq_\psi$ such that*

$$\psi * \mu \;=\; t(\min(Mod(\mu), \preceq_\psi)).$$

Thus the revision of ψ by μ is characterized by those models of μ that are most plausible according to the agent.

More recently there has been work in belief revision with respect to subsets of propositional logic. [10] extends the AGM approach to Horn clause knowledge bases while [7] addresses revision in other syntactic restrictions of propositional logic.

2.3 Belief Change in Logic Programming

Most previous work on belief change for logic programs is referred to as *update*. Representative work includes [31, 2, 20, 14, 26, 30, 11]. Strictly speaking, however, such approaches generally do not address "update," at least insofar as the term is understood in the belief revision community, but rather general change to a logic program.

A typical approach (e.g. [14], [30], and [11]) for such updates is to consider a sequence of logic programs P_1, P_2, $\ldots$, P_n, where for P_i, P_j, and $i > j$, the intuition is that P_i has higher priority or precedence over P_j. Given such a sequence, a set of answer sets is determined that in some sense respects the ordering. This may be done by translating the sequence into a single logic program that contains an encoding of the priorities, or by treating the sequence as a prioritized logic program, or by some other appropriate method. The net result, one way or another, is that one obtains a set of answer sets from such a program sequence. In particular, one does not obtain a new program expressed in the language of the original logic programs. Hence, these approaches fall outside the general AGM belief revision paradigm. Such approaches are also clearly syntactic in nature, and fall into the *belief base* category, rather than the *belief set* category.

Several principles have nonetheless been proposed for logic program update. In particular, [14] considers the question of what principles the update of logic programs should satisfy. This is done by re-interpreting different AGM-style postulates for revising or updating classic knowledge bases, as well as introducing new principles. Among the latter, we note the following:

Initialization $\emptyset * P \equiv P$.

Idempotency $(P * P) \equiv P$.
Tautology If Q is tautologous, then $P * Q \equiv P$.
Absorption If $Q = R$, then $((P * Q) * R) \equiv (P * Q)$.
Augmentation If $Q \subseteq R$, then $((P * Q) * R) \equiv (P * R)$.

It can be noted that if $\subseteq$ and $\equiv$ are interpreted in terms of strong equivalence, the first four postulates are implied by the AGM postulates, while the last corresponds to the first of the Darwiche and Pearl iteration postulates [8]. [23] also suggest the following postulate, which is also implied by the AGM approach:

WIS If $Q \equiv_s R$, then $(P * Q) \equiv (P * R)$.

Some work has focussed specifically on *revision* of logic programs. Early work in this direction includes a series of investigations dealing with restoring consistency for programs possessing no answer sets (e.g., [29]). Other work uses logic programs under a variant of the stable semantics to specify database revision, i.e., the revision of knowledge bases given as sets of atomic facts [22]. [12] addresses specific revision (and belief merging) operators based on distances defined in terms of the SE models of the underlying programs. As well, [9] considers the extent to which logic programs per se, are compatible with the AGM approach to revision.

3 Recasting Belief Revision in Terms of Answer Set Programs

The postulates and semantic construction of Section 2 are easily adapted to logic programs; for this, we draw on material from [12, 10]. To begin, the *expansion* of logic programs P and Q, $P + Q$, can be defined as a logic program R where $SE(P) \cap SE(Q) = SE(R)$. It can be observed that logic program expansion is unproblematic, since for any programs P, Q of a particular class, $SE(P) \cap SE(Q)$ satisfies the semantic conditions for that class; for example if $SE(P)$ and $SE(Q)$ are complete then so is $SE(P) \cap SE(Q)$.

For the postulates, we have the following, expressed in terms of logic programs. An *(AGM logic program) revision* function $*$ is a function from $\mathcal{LP} \times \mathcal{LP}$ to $\mathcal{LP}$ satisfying the following postulates.

(L0) $P * Q$ is a GLP.
(L1) $P * Q \models_s Q$.
(L2) If $P + Q$ is satisfiable, then $P + Q \equiv_s P * Q$.
(L3) If Q is satisfiable, then $P * Q$ is satisfiable.
(L4) If $P_1 \equiv_s P_2$ and $Q_1 \equiv_s Q_2$, then $P_1 * Q_1 \equiv_s P_2 * Q_2$.
(L5) $(P * Q) + R \models_s P * (Q + R)$.
(L6) If $(P * Q) + R$ is satisfiable, then $P * (Q + R) \models_s (P * Q) + R$.

For later reference we also give a postulate adapted from a similarly-named postulate from Horn revision [10].

(Acyc) If, for $0 \leq i < n$, we have $(P * Q_{i+1}) + Q_i$ is satisfiable and $(P * Q_0) + Q_n$ is satisfiable, then $(P * Q_n) + Q_0$ is satisfiable.

As well, faithful assignments can be defined for logic programs and SE models, basically by changing notation:

Definition 2. *A* faithful assignment *is a function that maps each logic program P to a total preorder $\preceq_P$ on $\mathcal{SE}$ such that for $m_1, m_2 \in \mathcal{SE}$:*

1. *If $m_1, m_2 \in SE(P)$ then $m_1 \approx_P m_2$*
2. *If $m_1 \in SE(P)$ and $m_2 \notin SE(P)$, then $m_1 \prec_P m_2$.*
3. *If $P \equiv_s Q$ then $\preceq_P = \preceq_Q$.*

The resulting preorder is referred to as the *faithful ranking* associated with P. Finally, one can define a function $*$ in terms of a faithful ranking by:

$$P * Q \;=\; t(\min(SE(Q), \preceq_P)). \tag{2}$$

The use of $*$ in (2) is suggestive; ideally one would next establish a correspondence between functions that satisfy the postulates and those that can be specified via Definition 2. However, there are two difficulties that arise with the naïve application of AGM revision to logic programs:

1. Some postulates may not be satisfied in a faithful ranking.
2. Necessary logical consequences of (L*0)-(L*6) may not hold in some classes of logic programs.

For the first problem, consider the following example involving normal logic programs:

$$\begin{aligned}
&P = \{\bot \leftarrow p,\ \bot \leftarrow q,\ \bot \leftarrow r.\}\\
&Q = \{\bot \leftarrow {\sim}p,\ \bot \leftarrow {\sim}q,\ \bot \leftarrow {\sim}r,\ r \leftarrow p,\ r \leftarrow q.\}\\
&R = \{\bot \leftarrow {\sim}p,\ \bot \leftarrow {\sim}q,\ \bot \leftarrow {\sim}r,\ r \leftarrow p, q,\ p \leftarrow q,\ q \leftarrow p.\}
\end{aligned}$$

We have the corresponding SE models:

$$\begin{aligned}
&SE(P) = \{(\emptyset, \emptyset)\}\\
&SE(Q) = \{(pqr, pqr), (pr, pqr), (qr, pqr), (r, pqr), (\emptyset, pqr)\}\\
&SE(R) = \{(pqr, pqr), (r, pqr), (\emptyset, pqr)\}
\end{aligned}$$

Now consider the total preorder over these SE models:

$$(\emptyset, \emptyset) < [(pqr, pqr), (pr, pqr), (qr, pqr)] < (\emptyset, pqr) < (r, pqr) < \langle \text{rest} \rangle$$

It can be verified that $SE((P*Q)+R) = \{(pqr, pqr), (r, pqr)\}$ and $SE(P*(Q+R)) = \{(pqr, prq)\}$. However, this violates (L5).

The second problem is analogous to one that cropped up in [10] with respect to Horn theories: due to the inferential weakness of Horn theories, an operator that satisfied the Horn revision postulates was not strong enough to guarantee the existence of a corresponding faithful ranking; instead the postulate (Acyc) (which is redundant in classical AGM revision) was required. Informally, the problem with respect to Horn theories was that, for two Horn formulas ϕ and ψ, $\phi \vee \phi$ is generally not Horn.

In terms of logic programs, one would require that, for programs P and Q, there is a program with SE models given exactly by $SE(P) \cup SE(Q)$. This is the case for GLPs, but not for any of the other classes of logic programs that we consider. Hence we obtain:

Theorem 2. *For generalised logic programs, (Acyc) is a logical consequence of the postulates (L*0) - (L*6).*

This result does not obtain for other classes of logic programs, and so for these (Acyc) is necessary.

4 Belief Revision of Answer Set Programs

To begin, we need to restrict candidate faithful rankings over SE models to just those that are sensible with respect to a given class of logic programs. The next two definitions serve to eliminate orderings which are incoherent with respect to a class of programs.

Definition 3. *A set of SE models S is* GLP (DLP, NLP, PLP, HLP) elementary *iff there exists a GLP (DLP, NLP, PLP, HLP) P such that $S = SE(P)$.*

Definition 4. *A faithful ranking on SE models $\preceq_P$ is* GLP (DLP, NLP, PLP, HLP) compliant *iff for every GLP (DLP, NLP, PLP, HLP) Q, we have that $\min(SE(Q), \preceq_P)$ is GLP (DLP, NLP, PLP, HLP) elementary.*

We have the following conditions on faithful rankings that provide counterparts for the notions of compliance, and that make it easier to work with a given ranking.

Definition 5. *Let $\preceq$ be a faithful ranking on $\mathcal{SE}$ and let $X, Y, Z \subseteq \mathcal{A}$.*
Then $\preceq$ satisfies:
(G$\preceq$) iff: if $X \subseteq Y$ then $(Y, Y) \preceq (X, Y)$.
(D$\preceq$) iff: $\preceq$ satisfies (G$\preceq$) and
if $X \subseteq Y \subseteq Z$ and $(X, Y) \approx (Z, Z)$ then $(X, Z) \preceq (Z, Z)$.
(N$\preceq$) iff: $\preceq$ satisfies (D$\preceq$) and
if $X, Y \subseteq Z$ and $(X, Z) \approx (Y, Z)$ then $(X \cap Y, Z) \preceq (X, Z)$.
(P$\preceq$) iff: $\preceq$ satisfies (D$\preceq$) and
if $X \subseteq Y$ then $(X, X) \preceq (X, Y)$.
(H$\preceq$) iff: $\preceq$ satisfies (P$\preceq$) and
if $X_1 \subseteq Y_1$ and $X_2 \subseteq Y_2$ then $(X_1 \cap X_1, Y_1 \cap Y_2 \preceq (X_1, Y_1)$.

Theorem 3. *Let $\preceq$ be a faithful ranking on $\mathcal{SE}$.*

1. *$\preceq$ is GLP compliant iff $\preceq$ satisfies (G$\preceq$).*
2. *$\preceq$ is DLP compliant iff $\preceq$ satisfies (D$\preceq$).*
3. *$\preceq$ is NLP compliant iff $\preceq$ satisfies (N$\preceq$).*
4. *$\preceq$ is PLP compliant iff $\preceq$ satisfies (P$\preceq$).*
5. *$\preceq$ is HLP compliant iff $\preceq$ satisfies (H$\preceq$).*

These notions of compliance on the one hand, and the postulate (Acyc) on the other, prove to be sufficient to extend the AGM approach to capture revision in logic programs. These results are described next.

The next two results (actually, two sets of results) constitute the two parts of a representation theorem. For each, x is a class of logic programs, where x is one of GLP, DLP, NLP, PLP, HLP, and (L0x) is postulate (L0) adjusted for class x. Then we have:

Theorem 4. *Let P be a logic program of class x and $\preceq$ an x-compliant faithful ranking associated with P. Define an operator $* : \mathcal{LP} \times \mathcal{LP} \mapsto \mathcal{LP}$ by $P * Q = t(\min(SE(Q), \preceq))$. Then $*$ satisfies postulates (L0x) - (L6) and (Acyc).*

Theorem 5. *Let $* : \mathcal{LP} \times \mathcal{LP} \mapsto \mathcal{LP}$ be a function satisfying postulates (L0x) - (L6) and (Acyc). Then for fixed program P of class x, there is a faithful ranking $\preceq$ on $\mathcal{SE}$ such that $\preceq$ is x-compliant and for every program Q of class x, $P * Q = t(\min(SE(Q), \preceq))$.*

Proof Outline. Let P be a logic program of type x. For any two SE models m_1 and m_2, $t(\{m_1, m_2\})$ was defined to be a least (with respect to characterizing SE models) logic program of type x containing m_1 and m_2. A binary relation $\preceq'$ over SE models is defined by: $m_1 \preceq' m_2$ iff $m_1 \in SE(P * t(\{m_1, m_2\}))$. (Note that a point of difficulty, and in contrast with the corresponding Katsuno-Mendelzon proof for AGM revision, is that it is possible to have both $m_1 \notin SE(P * t(\{m_1, m_2\}))$ and $m_2 \notin SE(P * t(\{m_1, m_2\}))$.)
The relation $\preceq'$ is in general not transitive; its transitive closure $\preceq^*$ is, of course, transitive, and moreover for arbitrary logic program Q of type x, the minimal Q SE models in $\preceq^*$ are shown to be the same as the SE models of $P * Q$. Finally, $\preceq^*$ is in general not total. The last step is to show that there is a total preorder on SE models $\preceq$ such that for any program Q of type x, the minimal Q SE models in $\preceq^*$ and $\preceq$ coincide. □

5 Iteration and GLP Belief Revision

The results of the previous section show that the classical AGM postulates can be recast in a logic programming framework. In this section we show that this is also the case for the Darwiche and Pearl postulates for iterated revision [8]. For simplicity and space reasons we just treat GLPs.

The four postulates for iterated revision proposed by Darwiche and Pearl, call them the *DP postulates*, have been characterized by corresponding restrictions on faithful rankings. We express these conditions in terms of logic programs. Let P be a logic program and $\preceq$ a faithful ranking with respect to P, and let us denote by $\preceq_Q$ the total preorder assigned to the logic program $P * Q$ resulting from the revision of P by Q. To save writing two sets of postulates, in the case of a logic program, $SE(\neg Q)$ is understood to mean $\mathcal{SE} \setminus SE(Q)$. In [8] it was shown that the conditions (IL1) - (IL4) below characterize (respectively) the four DP postulates (where, of course P and Q would be formulas of propositional logic):

(IL1) If $w, w' \in SE(Q)$ then $w \prec_Q w'$ iff $w \prec w'$.

(IL2) If $w, w' \in SE(\neg Q)$ then $w \prec_Q w'$ iff $w \prec w'$.
(IL3) If $w \in SE(Q)$ and $w' \in SE(\neg Q)$ then $w \prec w'$ entails $w \prec_Q w'$.
(IL4) If $w \in SE(Q)$ and $w' \in SE(\neg Q)$ then $w \preceq w'$ entails $w \preceq_Q w'$.

The first two postulates assert that following revision by Q, (SE)-models of Q retain their same relative ranking, as do non-models. The next two postulates assert roughly that a non-(SE)-model of Q never becomes more plausible with respect to a model of Q.

Thus to show that the DP postulates are consistent with (L0) - (L6) and (Acyc), it suffices to prove the following result:

Theorem 6. *Let P be a GLP, and $\preceq$ a GLP compliant, faithful ranking with respect to P. Moreover, let $*$ be the GLP revision function induced from $\preceq$ via Definition 2. For every GLP Q, there exists a GLP compliant, total preorder $\preceq_Q$, that is faithful with respect to $P * Q$, and such that (IL1) - (IL4) are satisfied.*

Proof. Let Q be any GLP. If Q is inconsistent, define $\preceq_Q$ to be equal to $\preceq$. Clearly, in this case $\preceq_Q$ satisfies (IL1) - (IL4). Moreover, since $\preceq$ is GLP compliant, so is $\preceq_Q$. Finally for faithfulness, since Q is inconsistent, by (L1), $SE(P * Q) = \emptyset$ and therefore $\preceq_Q$ is trivially faithful with respect to $P * Q$. Hence the theorem is true when Q is inconsistent.

Assume now that Q is consistent. We define $\preceq_Q$ as follows:

$$w \preceq_Q w' \text{ iff } w \in \min(SE(Q), \preceq) \text{ or } w \preceq w' \text{ and } w' \notin \min(SE(Q), \preceq). \quad (3)$$

According to (3), to construct $\preceq_Q$, one starts with $\preceq$ and simply places the minimal Q SE models (with respect to $\preceq$) at the beginning of the ranking; everything else remains the same. This construction is not new. In the propositional setting it was proposed and explored by Boutilier [3, 4] in his treatment of iterated revision; it is known to satisfy (IL1) - (IL4).

The ranking $\preceq_Q$ is clearly faithful with respect to $P * Q$. For GLP compliance, let w be a SE model (X, Y) where $X \subset Y$, and let w' be (Y, Y). Since (G$\preceq$) entails GLP compliance, it suffices to show that $w' \preceq_Q w$. We distinguish two cases. First assume that $w' \in \min(SE(Q), \preceq)$. Then $w' \in \min(\mathcal{SE}, \preceq_Q)$, and therefore $w' \preceq_Q w$ as desired. Second assume that $w' \notin \min(SE(Q), \preceq)$. Since $\preceq$ is GLP compliant, by (G$\preceq$) it follows that $w' \preceq w$. Consequently, since $w' \notin \min(SE(Q), \preceq)$, by (3) we obtain $w' \preceq_Q w$. $\Box$

6 Conclusion

In this paper we have shown how classical AGM-style revision may be expressed with respect to the major classes of extended logic programs under the answer-set semantics. That is, on the one hand we rephrased the AGM postulates in terms of logic programs and on the other hand we provided a semantic construction for revision operators analogous to faithful rankings, but with respect to SE models. Except for generalised logic programs, the postulate set had to be augmented by an "acyclicity" postulate; for the ranking on SE models, rankings also have to satisfy a "compliance" condition, specific to the class of logic programs being considered. Since both the new postulate and the compliance conditions are redundant in belief revision with respect to classical logic (as we have shown the additional postulate remains redundant for the most general class of programs, GLPs), our approach in fact extends the AGM approach to logic programs. Given these (postulational and semantic) characterizations, in a representation result we then show that these characterizations capture the same set of revision functions for each class of logic programs.

This work is interesting for several reasons. It shows how the AGM approach can be extended to a seemingly nonmonotonic (and certainly nonclassical) framework. As well, most previous work in logic program change was at the syntax level, in that the results of belief change depended on how a program was expressed. In contract, the approach at hand deals with the semantic level, in which arbitrary syntactic commitments don't play a role. Presumably also, the formal characterization may provide guiding principles for the development of specific revision operators.

References

1. C. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change: Partial meet functions for contraction and revision. *Journal of Symbolic Logic*, 50(2):510–530, 1985.
2. J. Alferes, J. Leite, L. Pereira, H. Przymusinska, and T. Przymusinski. Dynamic updates of non-monotonic knowledge bases. *Journal of Logic Programming*, 45(1–3):43–70, 2000.
3. C. Boutilier. Revision sequences and nested conditionals. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 519–531, 1993.
4. C. Boutilier. Iterated revision and minimal change of conditional beliefs. *Journal of Logic and Computation*, 25:262–305, 1996.
5. G. Brewka, T. Eiter, and M. Truszczynski. Answer set programming at a glance. *Commun. ACM*, 54(12):92–103, 2011.
6. P. Cabalar and P. Ferraris. Propositional theories are strongly equivalent to logic programs. *Theory and Practice of Logic Programming*, 7(6):745–759, 2007.
7. N. Creignou, O. Papini, R. Pichler, and S. Woltran. Belief revision within fragments of propositional logic. In G. Brewka, T. Eiter, and S. A. McIlraith, editors, *Proceedings of the Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*. AAAI Press, 2012.
8. A. Darwiche and J. Pearl. On the logic of iterated belief revision. *Artificial Intelligence*, 89:1–29, 1997.
9. J. Delgrande. A program-level approach to revising logic programs under the answer set semantics. *Theory and Practice of Logic Programming, 26th Int'l. Conference on Logic Programming (ICLP'10) Special Issue*, 10(4–6):681–696, July 2010.
10. J. Delgrande and P. Peppas. Revising Horn theories. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 839–844, Barcelona, Spain, 2011.
11. J. Delgrande, T. Schaub, and H. Tompits. A preference-based framework for updating logic programs. In *Ninth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'07)*, pages 71–83, Phoenix, AZ, 2007. Springer Verlag.
12. J. Delgrande, T. Schaub, H. Tompits, and S. Woltran. A model-theoretic approach to belief change in answer set programming. *ACM Transactions on Computational Logic*, 14(2), 2013.

13. T. Eiter, M. Fink, J. Pührer, H. Tompits, and S. Woltran. Model-based recasting in answer-set programming. Technical Report DBAI-TR-2013-83, Institute of Information Systems 184/2, Vienna University of Technology, Austria, 2013.
14. T. Eiter, M. Fink, G. Sabbatini, and H. Tompits. On properties of update sequences based on causal rejection. *Theory and Practice of Logic Programming*, 2(6):711–767, 2002.
15. T. Eiter, H. Tompits, and S. Woltran. On solution correspondences in answer set programming. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence* (*IJCAI 2005*), pages 97–102, 2005.
16. G. Flouris, D. Plexousakis, and G. Antoniou. On applying the AGM theory to DLs and OWL. In *Proc. ISWC*, volume 3729 of *LNCS*, pages 216–231. Springer, 2005.
17. P. Gärdenfors. *Knowledge in Flux: Modelling the Dynamics of Epistemic States*. The MIT Press, Cambridge, MA, 1988.
18. H. Katsuno and A. Mendelzon. Propositional knowledge base revision and minimal change. *Artificial Intelligence*, 52(3):263–294, 1991.
19. P. Krümpelmann and G. Kern-Isberner. Belief base change operations for answer set programming. In L. F. del Cerro, A. Herzig, and J. Mengin, editors, *Logics in Artificial Intelligence - 13th European Conference, JELIA 2012, Toulouse, France, September 26-28, 2012. Proceedings*, volume 7519 of *Lecture Notes in Computer Science*, pages 294–306. Springer, 2012.
20. J. Leite. *Evolving Knowledge Bases: Specification and Semantics*. IOS Press, Amsterdam, 2003.
21. V. Lifschitz, D. Pearce, and A. Valverde. Strongly equivalent logic programs. *ACM Transactions on Computational Logic*, 2(4):526–541, 2001.
22. V. W. Marek and M. Truszczyński. Revision programming. *Theoretical Computer Science*, 190:241–277, 1998.
23. M. Osorio and V. Cuevas. Updates in answer set programming: An approach based on basic structural properties. *Theory and Practice of Logic Programming*, 7(4):451–479, 2007.
24. P. Peppas. Belief revision. In F. van Harmelen, V. Lifschitz, and B. Porter, editors, *Handbook of Knowledge Representation*, pages 317–359. Elsevier Science, San Diego, USA, 2008.
25. C. Sakama and K. Inoue. Updating extended logic programs through abduction. In *Proceedings of the Fifth International Conference on Logic Programming and Nonmonotonic Reasoning* (*LPNMR'99*), volume 1730 of *Lecture Notes in Artificial Intelligence*, pages 147–161. Springer, 1999.
26. C. Sakama and K. Inoue. An abductive framework for computing knowledge base updates. *Theory and Practice of Logic Programming*, 3(6):671–713, 2003.
27. M. Slota and J. Leite. Robust equivalence models for semantic updates of answer-set programs. In G. Brewka, T. Eiter, and S. A. McIlraith, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference, KR 2012, Rome, Italy, June 10-14, 2012*. AAAI Press, 2012.
28. H. Turner. Strong equivalence made easy: nested expressions and weight constraints. *Theory and Practice of Logic Programming*, 3(4):609–622, 2003.
29. C. Witteveen, W. van der Hoek, and H. de Nivelle. Revision of non-monotonic theories: Some postulates and an application to logic programming. In *Proceedings of the Fifth European Workshop on Logics in Artificial Intelligence* (*JELIA'94*), volume 838 of *Lecture Notes in Artificial Intelligence*, pages 137–151, Berlin, 1994. Springer Verlag.
30. F. Zacarías, M. Osorio, J. C. Acosta Guadarrama, and J. Dix. Updates in Answer Set Programming based on structural properties. In S. McIlraith, P. Peppas, and M. Thielscher, editors, *Proceedings of the 7th International Symposium on Logical Formalizations of Commonsense Reasoning*, pages 213–219. Fakultät für Informatik, ISSN 1430-211X, May 2005.
31. Y. Zhang and N. Y. Foo. Updating logic programs. In *Proceedings of the Thirteenth European Conference on Artificial Intelligence* (*ECAI'98*), pages 403–407, 1998.