Belief Base Change Operations for Answer Set Programming

Patrick Krümpelmann and **Gabriele Kern-Isberner**
Information Engineering Group, Technische Universität Dortmund, Germany

Abstract

We present a principled approach to the problem of belief revision in (non-monotonic) logic programming under the answer set semantics. Unlike previous approaches we use a belief base approach. Belief bases are sets of sentences that are, in contrast to belief sets, not deductively closed. We show that many of the classic base revision postulates are applicable to the logic programming case. We discuss further postulates for logic program revision and show that many of them follow from classical base revision postulates. For those postulates that do not follow from base revision postulates we propose new postulates that may also be justified from the base revision perspective. Moreover we develop a new construction for prioritized multiple base revision based on a consolidation operation via remainder sets. This construction is applicable in the propositional and the logic programming case. We connect postulates and construction by proving a representation theorem showing that the construction is exactly characterized by the proposed set of postulates.

Introduction

Belief dynamics within propositional logics have been studied for over 25 years now (Fermé and Hansson 2011). The theory of classic belief revision has well formulated notions for theory change. The notions of the objects of change are well defined. In general these can be belief sets or belief bases. For both well defined sets of rationality postulates for the change operation have been defined.

For logic programs, most approaches to dynamics are very pragmatic and lack a formal representation of requirements of the process. Few works examined the formal properties of approaches to logic programming change. They tried to link it to the classical theory in propositional logic using different notions of the object of change as well as interpretations of the desiderata of the operation itself. All of these used the classic AGM postulates for change of belief sets to logic programs. Hereby the adequate definition of a belief set, a consequence relation and a notion of equality is crucial for the applicability of the classic Postulates. First attempts showed that "the majority of the adapted AGM and update postulates are violated ..." for a variety of approaches based on the causal rejection principle (Eiter et al. 2002). Hereby logic programs were interpreted as epistemic states and the set of rules satisfied by all answer sets the belief set. The first successful approach of a relation to the AGM theory was achieved by the use of monotonic SE-Models first presented in (Delgrande et al. 2008) and later on extended to merge operations (Delgrande et al. 2009; Hué, Papini, and Würbel 2009) and to update operations in the style of Katsuno and Mendelson by Slota and Leite (Slota and Leite 2010; Slota and Leite 2012). They made use of the semantic characterization of programs via SE-models and applied an adapted version of distance based revision operators from classic belief revision. This approach was shown to satisfy the majority of the adopted AGM postulates. AGM style change operations for answer set programming (ASP) have disadvantages and show undesired results from the ASP point of view as first noted in (Slota and Leite 2010). These undesired results for change operations in ASP are due to the application of a semantic approach to ASP.

In this work we approach change operations of ASP from the Belief Base perspective. The well developed classical base revision approach has not been considered in the light of ASP before. Moreover, we argue that the belief base approach is the natural one for ASP. AGM change operations on belief sets can be seen as operations on the knowledge level, abstractly describing how an ideal reasoner would change its beliefs. This underlies the assumption of an omnipotent reasoner while ASP's main features are efficient computation of finite programs with finite answer sets. Neither for programs nor for answer sets the deductive closure is defined. Belief bases are also more expressive, on the knowledge level one cannot distinguish between inferred beliefs and fundamental, or self-supporting, ones. While this abstraction from the fundamental beliefs and their syntactic representation have some advantages for the global picture of belief change we argue that ASP is an inherently syntax based approach. A key feature of ASP is that beliefs are formulated in form of easily understandable rules that allow for explicit exceptions. Inferences can be explained by use of the formulated rules. From the base revision perspective the result of a change operation for ASP should be founded, understandable and close to the original syntax.

The main contributions of this paper are the following. We present a general exploration of the application of classic base revision theory to change operations on ASP. We discuss ASP specific postulates from the literature in the light of a base revision approach, proofs of the relationships

among both and formulation of adapted postulates. Development of a base revision construction which is applicable to ASP and proof of a characterization theorem for it.

The remainder of this paper is structured as follows. In the next two sections we give some preliminaries on belief base revision and answer set programming. Following on this we develop base revision postulates for ASP, discuss them and relate them to other postulates for ASP belief change. After that we present our construction of multiple base revision and show it applicability to ASP and the correspondence to the postulates. Finally we discuss our approach and conclude.

Belief Base Revision

The classic theory of belief revision is formulated for a propositional language $\mathcal{L}_{prop}$. A *belief base* $\mathcal{B} \subseteq \mathcal{L}_{prop}$ is a set of propositional sentences and a *belief set* $\mathcal{BS}$ is a deductively closed set of sentences. The theory of belief base change operations has been intensively studied in (Hansson 1991; Hansson 2001). Postulates have been established as well as constructions and representation theorems connecting both. As we consider belief bases for knowledge representation we start with the corresponding postulates for belief base revision (Hansson 2001). Let $*$ be a base revision operator which given a belief base $\mathcal{B}$ and a sentence $\alpha \in \mathcal{L}_{prop}$ constitutes the revised belief base $\mathcal{B} * \alpha$. Moreover an expansion operator $+$ is defined for belief bases as the *non-closing expansion operator* defined as $\mathcal{B} + \alpha = \mathcal{B} \cup \{\alpha\}$. The basic set of postulates demanded for a belief base revision operator is:

Success $\alpha \in \mathcal{B} * \alpha$

Inclusion $\mathcal{B} * \alpha \subseteq \mathcal{B} + \alpha$

Vacuity If $\mathcal{B} \cup \alpha$ is consistent then $\mathcal{B} + \alpha \subseteq \mathcal{B} * \alpha$

Consistency If α is consistent then $\mathcal{B} * \alpha$ is consistent

Relevance If $\beta \in (\mathcal{B} \cup \alpha) \setminus (\mathcal{B} * \alpha)$ then there is a set H such that $\mathcal{B} * \alpha \subseteq H \subseteq \mathcal{B} \cup \alpha$ and H is consistent but $H \cup \{\beta\}$ is inconsistent

Uniformity If for all $\mathcal{B}' \subseteq \mathcal{B}, \mathcal{B}' \cup \alpha$ is inconsistent if and only if $\mathcal{B}' \cup \beta$ is inconsistent, then $\mathcal{B} \cap (\mathcal{B} * \alpha) = \mathcal{B} \cap (\mathcal{B} * \beta)$

The success postulate states that the new information should be part of the revision result. Inclusion demands that revision by some information should not introduce more information than the expansion. Vacuity demands that if the new information is consistent with the belief base then no information should be discarded. Consistency postulates that if the new information is consistent in itself then the result of the revision is consistent as well. Relevance states that any discarded piece of information would have lead to inconsistency in a super set of the revision result. Any operator that satisfies Relevance and Inclusion satisfies Vacuity, (Hansson 2001). Uniformity demands for the independence of exact input if the set of inconsistent sets are equal.

The standard construction in belief base revision is via the Levi-Identity by use of an appropriate contraction $-$ and expansion $+$ operator:

$$\mathcal{B} * \alpha = (\mathcal{B} - \neg\alpha) + \alpha$$

Hereby $-$ is an appropriate contraction operator characterized by a set of rationality postulates. Via the Levi-Identity connections between properties of the contraction and the revision operator can be drawn. A contraction operator $\mathcal{B} - \alpha$ can be defined as the intersection of a selection of maximal sets not containing α. Formally for a given belief base $\mathcal{B}$ and a sentence α the set of *remainder sets* denoted by $\mathcal{B} \perp \alpha$ is such that for each $X \in \mathcal{B} \perp \alpha$: $X \subseteq \mathcal{B}$, $\alpha \notin Cn(X)$ and there is no X' such that $X \subset X' \subseteq \mathcal{B}$ and $\alpha \notin Cn(X')$. Let $\mathcal{B}$ be a set of sentences. A function γ is a *selection function* for $\mathcal{B}$ if and only if for all sentences α if $\mathcal{B} \perp \alpha \neq \emptyset$ then $\emptyset \neq \gamma(\mathcal{B} \perp \alpha) \subseteq \mathcal{B} \perp \alpha$ and if $\mathcal{B} \perp \alpha = \emptyset$ then $\gamma(\mathcal{B} \perp \alpha) = \{\mathcal{B}\}$. The *partial meet contraction* operator is then defined as

$$\mathcal{B} -_\gamma \alpha = \bigcap \gamma(\mathcal{B} \perp \alpha).$$

It has been shown that a revision operator satisfies all of the above postulates if and only if it is constructible via the Levi-identity and a partial meet contraction operator.

Partial meet contraction can be applied to belief sets in the same way as for belief bases. The difference between the revision of belief sets and belief bases results from the difference of the expansion operator. For belief bases the *non-closing expansion operator* defined as $\mathcal{B} + \alpha = \mathcal{B} \cup \alpha$ is used and for belief sets the closing expansion operator $\mathcal{B} \dot{+} \alpha = Cn(\mathcal{B} \cup \alpha)$. This slight difference has implications for the resulting revision operator. These implications are reflected by the corresponding postulates for the revision of belief sets (Alchourron, Gärdenfors, and Makinson 1985). The postulates of Success and Consistency are identical and the postulates Inclusion and Vacuity only differ in the expansion operator. In addition to these four postulates the basic six AGM postulates contain the postulate *Closure:* $\mathcal{K} * \alpha = Cn(\mathcal{K} * \alpha)$ and the postulate *Extensionality:* If $\alpha \Leftrightarrow \beta \in Cn(\emptyset)$ then $\mathcal{K} * \alpha = \mathcal{K} * \beta$. The first postulate states that all logical consequences of the belief set should be contained in the belief set which makes it infinite in general. The second postulate demands for irrelevance of syntax, that is, all logically equivalent formulas shall lead to the same change. While Closure is an inherent features of the knowledge level approach and is definitely not appropriate in a base change framework, the idea of Extensionality as a postulate to control semantic differences caused by slight syntactic variations should also be kept in mind for belief base changes.

Answer Set Programming

In this work we focus on extended logic programs under the answer set semantics (Gelfond and Lifschitz 1988), but most results also hold for generalized disjunctive logic programs. Extended logic programs consist of rules over a set of propositional atoms $\mathcal{A}$ using strong negation $\neg$ and default negation not. A literal L can be an atom A or a negated atom $\neg A$. The complement of a literal L is denoted by $\neg L$ and is A if $L = \neg A$ and $\neg A$ if $L = A$. Let $\mathcal{A}$ be the set of all atoms and Lit the set of all literals $Lit = \mathcal{A} \cup \{\neg A \mid A \in \mathcal{A}\}$. $\mathcal{D} = \{\text{not } L \mid L \in Lit\}$ denotes the set of all default negated literals. And $\mathfrak{L} = Lit \cup \mathcal{D}$ represents the set of all literals

and default negated literals. Throughout this paper we also call default negated literals assumptions. A rule r is written as

$$L \leftarrow L_0, \ldots, L_m, \text{not } L_{m+1}, \ldots, \text{not } L_n.$$

where the head of the rule $H(r) = L$ is either empty or consists of a single literal and the body $\mathcal{B}(r) = \{L_0, \ldots, L_m, \text{not } L_{m+1}, \ldots, \text{not } L_n\}$ is a subset of $\mathfrak{L}$. The body consists of a set of literals $\mathcal{B}(r)^+ = \{L_0, \ldots, L_m\}$ and a set of default negated literals denoted by $\mathcal{B}(r)^- = \{L_{m+1}, \ldots, L_n\}$. Given this we can write a rule as

$$H(r) \leftarrow \mathcal{B}(r)^+, \mathcal{B}(r)^-.$$

If $\mathcal{B}(r) = \emptyset$ we call r a fact and if $H(r) = \emptyset$ we call r a constraint. We call a program without default negation a strict program. A set of literals which is consistent, i. e. it does not contain complementary literals L and $\neg L$, is called a state I. A literal L is true in I iff $L \in I$ and false otherwise. The body $\mathcal{B}(r)$ of a given rule r is true in I iff each $L \in \mathcal{B}(r)^+$ is true in I and each $L \in \mathcal{B}(r)^-$ is false in I. A rule r is true in I iff $H(r)$ is true in I whenever $\mathcal{B}(r)$ is true in I. A state is a model of a program P if r is true in I for all $r \in P$. The reduct P^S of a program P relative to a set S of literals is defined as:

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

An answer set of a program P is a state I which is a minimal model of P^I. The set of answer sets of a program P is denoted by $AS(P)$ and the set of all constructible programs by $\mathcal{P}$.

Postulates for ASP Base Revision

In contrast to postulates for revision of belief sets, the postulates for belief base revision do not demand many properties of the objects of change and are general for sets of propositional formulas and, in this case, sets of rules. That is, we consider a revision operator $* : \mathcal{P} \times \mathcal{P} \rightarrow \mathcal{P}$. The only notion to be defined is the one of consistency of a set. Classically the consistency bases on a notion of models of a given theory. A belief base is consistent if and only if it has a model. For logic programs the notion of a model is not uniquely defined and dependent on the used semantics. Here we consider logic programs under the answer set semantics and use the following definition.

Definition 1. *Let P be a program. P is called consistent if and only if $AS(P) \neq \emptyset$.*

Note that we defined answer sets of a program based on a state which does not contain complementary literals. There are other notions of answer sets in the literature in which answer sets can contain complementary literals but usually are defined to equal the whole set of literals in this case. It would be easy to adapt the notion of consistency to such cases.

A notable property of the notion of consistency of logic programs is that the state of inconsistency of a program is non-monotonic. While in the propositional case any subset of a consistent set of sentences is consistent, for a consistent logic program Q there can be a subset $P \subset Q$ such that P is inconsistent.

Example 1. *The program $P = \{a., \neg a \leftarrow not\ b.\}$ is obviously inconsistent. $P' = P \cup \{b \leftarrow not\ c.\}$ has $\{a, b\}$ as its only answer set. $P'' = P' \cup \{c.\}$ is inconsistent again.*

The base revision postulates given above can be directly translated to the logic programming case with $+$ being the non-closing expansion $P + Q = P \cup Q$. We start with the direct translation of the base revision postulates to logic programs. Let $*$ be a base revision operator for logic programs.

Success $Q \subseteq P * Q$

Inclusion $P * Q \subseteq P + Q$

Vacuity If $P + Q$ is consistent then $P + Q \subseteq P * Q$

Consistency If Q is consistent then $P * Q$ is consistent.

Relevance If $r \in (P \cup Q) \setminus (P * Q)$ then there is a program H such that $P * Q \subseteq H \subseteq P \cup Q$ and H is consistent but $H \cup \{r\}$ is inconsistent.

Fullness If $r \in (P \cup Q) \setminus (P * Q)$, then $P * Q$ is consistent and $(P * Q) \cup \{r\}$ is inconsistent.

Uniformity If for all $P' \subseteq P, P' \cup \{Q\}$ is inconsistent if and only if $P' \cup \{R\}$ is inconsistent, then $P \cap (P * Q) = P \cap (P * R)$

The postulate of Fullness is a stronger version of Relevance and resembles a stronger requirement to the minimality of change. For the propositional case Fullness is arguably too strong (Hansson 2001), however for weaker logics it has also been proven to be useful, cf. (Delgrande 2008). For logic programing it also does not have the same undesirable implications as we shall see later on.

Due to the non-monotonicity of inconsistency we can strengthen the Consistency postulate for the logic programming case such that they capture the same idea as for the propositional case. The Consistency postulate expresses in the propositional case that the outcome of the revision shall be consistent whenever possible. For propositional logic this is possible if and only if the input is consistent. That is, due to the Success postulate and the monotonicity of inconsistency any revision with an inconsistent input will be inconsistent. If the input is consistent, if necessary, rejecting all previous information leads to a consistent belief base. For logic programming the input can be inconsistent and a subset of the revision outcome and still the revision outcome can be consistent.

Example 2. *Let $P = \{b., \neg a.\}$ and $Q = \{a., \neg a \leftarrow not\ b.\}$. The program Q is inconsistent but the revision $P * Q = \{b., a., \neg a \leftarrow not\ b.\}$ is consistent and satisfies all other postulates.*

The following formulation of NM-Consistency considers non-monotonic inconsistency by use of an appropriate premise for the possibility of consistency.

NM-Consistency If there exists some consistent X, $Q \subseteq X \subseteq P \cup Q$ then $P * Q$ is consistent.

Proposition 1. *Let $*$ be a revision operator. If $*$ satisfies Consistency then it satisfies NM-Consistency.*

Proof. If Q is consistent then $X = Q$ is consistent such that the premise of NM-Consistency is satisfied. □

The base revision postulates have been accepted as characterizations of desirable change operations for propositional belief bases. The postulates can be applied to belief bases represented as logic programs as just shown. However, it has to be shown that the defined postulates for belief base change lead to desirable inference behavior given the non-monotonicity of the answer set semantics. Further postulates for the connection of the change on the program level and the resulting answer sets might have to be formulated. In previous works such properties have been formulated which we consider in the following.

The ASP specific revision postulates have been proposed in (Eiter et al. 2002) and adopted by several authors for the evaluation of their approaches afterwards (Delgrande, Schaub, and Tompits 2007; Delgrande et al. 2008; Delgrande 2010; Osorio and Cuevas 2007). The postulates base on a notion of equivalence of programs which is defined through the identity of the sets of answer sets in (Eiter et al. 2002)[1], through the equivalence of SE-Models in (Delgrande et al. 2008) and partially by means of uniform equivalence (Eiter and Fink 2003) in (Delgrande 2010).

Here, we generalize the postulates by considering different notions of equivalence. Therefore we parameterize them with a notion of equivalence based on $\circ$. We obtain the original postulates with $\circ = AS$ but consider a family of equivalences $\circ \in \{AS, UE, SE, P\}$ with AS being equivalence of sets of answer sets, UE being uniform equivalence, SE being strong equivalence and P the syntactic identity of the programs. The equivalences are increasingly stronger with the order given above. More precisely, it holds that for any two programs P and P' that if $P \equiv_P P'$ then $P \equiv_{SE} P'$, if $P \equiv_{SE} P'$ then $P \equiv_{UE} P'$ and if $P \equiv_{UE} P'$ then $P \equiv_{AS} P'$. Thus apart from the original postulates we also consider stronger versions of these. It should be noted that this family of notions of equivalence could be extended by all intermediate notions as formalized in (Woltran 2008).

The Tautology postulate uses the notion of tautological programs which are denoted by $P_\top$. For propositional logic tautologies are defined as being true in all interpretations. Formally a sentence $\alpha_\top$ is a tautology if and only if $\alpha_\top \in Cn(\emptyset)$, due to $Mod(\emptyset) \subseteq Mod(\alpha_\top)$. We can give a definition of tautologies and set of tautologies, tautological programs.

Definition 2. *Given a program P over the set of literals Lit. P is* tautological *if and only if for each rule $r \in P$, r is true in all states $I \subseteq Lit$. A tautological program is denoted by $P_\top$.*

There is also a syntactic characterization of tautologies for ASP. If $P_\top$ is a tautological program then it holds for all rules $r \in P_\top$ that $H(r) \in \mathcal{B}^+(r)$.

The generalized postulates for ASP revision from (Eiter et al. 2002) are given here:

Initialisation$_\circ$: $\emptyset * P \equiv_\circ P$

Idempotence$_\circ$: $P * P \equiv_\circ P$

Absorption$_\circ$: $(P * Q) * Q \equiv_\circ P * Q$

[1]For finite alphabets.

Tautology$_\circ$: $P * P_\top \equiv_\circ P$

Disjointness$_\circ$: If $P = P_1 \cup P_2$ and P_1 and P_2 have disjoint sets of literals then $P * Q \equiv_\circ (P_1 * Q) \cup (P_2 * Q)$.

Parallelism$_\circ$: If Q_1 and Q_2 have disjoint sets of literals then $P * (Q_1 \cup Q_2) \equiv_\circ (P * Q_1) \cup (P * Q_2)$.

The implications on the equivalences lead to implications of the resulting postulates. Most importantly we get that, if an operator $*$ satisfies Initialisation$_P$ then it also satisfies Initialisation$_{SE}$, Initialisation$_{UE}$ and Initialisation$_{AS}$. On the other hand, if $*$ is shown to violate Initialisation$_{AS}$, then it also violates Initialisation$_{UE}$, Initialisation$_{SE}$ and Initialisation$_P$. In the following we only show results for the strongest version of a postulate and omit the implications of them.

We obtain the following results for the connection of base revision postulates and ASP change postulates:

Proposition 2. *Let $*$ be a change operator on logic programs. If $*$ satisfies*

1. *Success and Inclusion then it satisfies Initialisation$_P$*
2. *Success and Inclusion then it satisfies Idempotence$_P$*
3. *Success, Consistency, Inclusion and Vacuity then it satisfies Absorption$_P$*

Proof. From Success follows $P \subseteq \emptyset * P$ and from Inclusion follows $\emptyset * P \subseteq P$. From Success follows $P \subseteq P * P$ and from Inclusion follows $P * P \subseteq P$. From Success follows $Q \subseteq P * Q$ and together with Consistency that $(P * Q) \cup P$ is consistent. Then it follows from Vacuity and Inclusion that $(P * Q) * Q = P * Q$. □

Hence, we have just shown, that the first three ASP postulates follow, for all considered notions of equivalence, from very basic base revision postulates. This verifies the adequateness of the base revision approach for ASP.

However, the remaining four ASP postulates do not only not follow from the base revision postulates, they are even inconsistent with them. This is not entirely surprising since they were formulated for a very different approach for handling update sequences of logic programs. However, the underlying ideas can be adapted to the base revision setting.

In the following we have a closer look at the ASP postulates that are in conflict with the base revision postulates. The Tautology$_{AS}$ postulate is violated if the belief base is inconsistent and consistent after revising by a tautology.

Example 3. *Let $P = \{a., \neg a.\}$ and $Q = \{b \leftarrow b.\}$ with $AS(P) = \emptyset$ and $AS(Q) = \{\emptyset\}$. For any revision operator $*$ satisfying Tautology$_{AS}$ we get $AS(P * Q) = \emptyset$, e. g. $P * Q = \{a., \neg a., b \leftarrow b.\}$. Consistency on the other hand allows for changes to make the belief base consistent. For any revision operator $*'$ satisfying Consistency we get $AS(P *' Q) \neq \emptyset$, e. g. $P *' Q = \{a., b \leftarrow b.\}$.*

Thus tautology cannot be satisfied by any operator satisfying consistency because of the case in which an inconsistent belief base is made consistent by the revision by a tautology.

Proposition 3. *Let $*$ be a change operator on logic programs. If $*$ satisfies*

1. *Consistency then it violates Tautology$_{AS}$*
2. *Success then it violates Tautology$_P$*

Proof. See Appendix. □

However, Tautology$_{AS}$ in general addresses an important issue in ASP revision, namely that if the belief base is consistent, the revision by a tautology should not lead to any changes. This is not satisfied by many approaches to dynamics in ASP.

To adapt the Tautology$_{AS}$ postulate to the base revision setting it is desirable that Tautology$_{AS}$ is satisfied, except for the case in which the belief base is inconsistent. To this end we introduce a new postulate as a weakening of Tautology$_{AS}$.

Consistent Tautology$_{AS}$ If P is consistent, then $P * P_\top \equiv_{AS} P$.

We consider Consistent Tautology in its weakest form, based on $\equiv_{AS}$, here. Clearly, demanding Consistent Tautology$_P$ to be satisfied conflicts with Vacuity. For Consistent Tautology$_{AS}$ we can show that it follows from basic base revision postulates. Before that we give a stronger version of the postulate.

The problem of the approaches not satisfying the tautology postulate is, that they make unnecessary changes not only for tautological revisions. Any revision by program that does not influence the generation of the answer sets of the belief base should not add any answer sets. This idea has been discussed in (Alferes et al. 2005) and (Sefránek and Sika 2006) and formalized for dynamic logic programming as the *refined extension principle*. Here we formalize this idea in the following postulate:

Irrelevance$_{AS}$ If $\emptyset \neq AS(P) = AS(P \cup Q)$ then $AS(P * Q) \subseteq AS(P)$.

Irrelevance is a postulate formulating some form of minimal change on the answer set level. The two basic postulates for minimal change of the base revision postulates, Inclusion and Vacuity, are sufficient to guarantee Irrelevance$_{AS}$.

Proposition 4. *If $*$ satisfies Inclusion and Vacuity then it satisfies Irrelevance$_{AS}$.*

Proof. If $\emptyset \neq AS(P) = AS(P \cup Q)$ then $P+Q$ is consistent and from Vacuity follows $P + Q \subseteq P * Q$. Together with Inclusion follows $P + Q = P * Q$ and from the premise follows that $AS(P * Q) = AS(P)$ such that Irrelevance$_{AS}$ is satisfied. □

We consider Parallelism and Disjointness together since the underlying idea is similar. The problem with Parallelism and Disjointness is that the respective third program can contain rules connecting the disjoint sets of literals such that inconsistencies arise in combination of both sets of literals but not based on a single one.

Proposition 5. *Let $*$ be a change operator on logic programs. If $*$ satisfies*

1. *Success, Consistency and Vacuity then it violates Parallelism$_{AS}$*
2. *Vacuity and Consistency then it violates Disjointness$_{AS}$*

Proof. See Appendix. □

Consider the following example which demonstrates that Disjointness$_{AS}$ is in conflict with the principle of minimal change in base revision.

Example 4. *Let $P = \{a., b.\}$ such that $P = P_1 \cup P_2$ with $P_1 = \{a.\}$ and $P_2 = \{b.\}$, and $Q = \{\neg a \leftarrow b.\}$. The revision of $P_1 * Q$ we can note that there is no conflict and $AS(P_1 \cup Q) = \{a.\}$ such that there is no need to change anything, such that $P_1 * Q = P_1 \cup Q$ seems to be a reasonable revision. The same holds for $P_2 * Q$ with $AS(P_2 \cup Q) = \{b.\}$. The Disjointness$_{AS}$ postulate demands that $AS(P * Q) = AS((P_1 * Q) \cup (P_2 * Q))$. In this case $AS((P_1 * Q) \cup (P_2 * Q)) = \emptyset$ which is clearly not a desirable outcome for $AS(P * Q)$.*

Disjointness$_{AS}$ is in conflict with the minimal change of the revisions of $P_1 * Q$ and $P_2 * Q$. In order to satisfy Disjointness$_{AS}$ the the revisions of $P_1 * Q$ and $P_2 * Q$ have to be cautious enough to anticipate possible inconsistencies with additional input. Here we consider that inconsistencies with some input should be handled by a revision operator applied to this input and not by a previous revision.

Hence these postulates are too strong to be satisfiable by a base revision approach. Therefore we present weakened versions of the postulates that allow for minimal change operations.

Weak Disjointness$_\circ$: If $P = P_1 \cup P_2$ and P_1 and P_2 have disjoint sets of literals $\mathcal{A}_1$ and $\mathcal{A}_2$ and for each set of literals $\mathcal{A}_r$ of a rule $r \in Q$ it holds $\mathcal{A}_r \cap \mathcal{A}_1 = \emptyset$ or $\mathcal{A}_r \cap \mathcal{A}_2 = \emptyset$ then $P * Q \equiv_\circ (P_1 * Q) \cup (P_2 * Q)$.

Weak Parallelism$_\circ$: If Q_1 and Q_2 have disjoint sets of literals $\mathcal{A}_1$ and $\mathcal{A}_2$, and for each set of literals $\mathcal{A}_r$ of a rule $r \in P$ it holds $\mathcal{A}_r \cap \mathcal{A}_1 = \emptyset$ or $\mathcal{A}_r \cap \mathcal{A}_2 = \emptyset$ then $P * (Q_1 \cup Q_2) \equiv_\circ (P * Q_1) \cup (P * Q_2)$.

These weakened versions of disjointness and parallelism are not in conflict with the proposed set of base revision postulates, but are still strong enough such that they do not follow from the base revision postulate set. Thus we add them to the set of desirable postulates.

To sum up, we have shown that all postulates for ASP revision that are not in conflict with the base revision setting follow from the base revision postulates. For those ASP revision postulates that are in conflict with the base revision postulates we gave adequately weakened versions. Therefore we are looking for an operator which satisfies Success, Inclusion, Consistency, Relevance, Uniformity, Weak Disjointness$_P$ and Weak Parallelism$_P$.

Construction of ASP Base Revision

The direct transfer of the construction of belief base operators via the Levi identity to logic programs does not work, because neither the negation nor the inference of a rule is defined and inconsistency cannot be reduced to complementary literals. Even in the very restricted case of $Q = \{L.\}$ consisting of a single fact, it would not be sufficient to contract such that $\neg L \notin \cap AS(P)$.

So we have to look for different constructions which implement the idea of the Levi-Identity while being adequate

for the ASP case. The idea of the Levi-Identity is, that after contracting by $\neg\alpha$ the belief base is consistent with α. That is the belief base is made consistent with the information to be added, before adding it. This does not work in general for the logic programming case because the dependencies within a program are complex and cannot be anticipated without including the input program. The inconsistency of a program P with a new program Q can only be determined by considering $P \cup Q$, as the interaction of rules of both programs generates the inconsistency. Base revision constructions of this type are called *external revision* since a sub-operation takes place outside of the original set. External revisions do not make sense for belief sets since all inconsistent belief sets are equal.

Hence the base revision construction for logic programs has to consider $P \cup Q$ to determine inconsistency. From $P \cup Q$ rules are removed such that the resulting program is consistent with Q. This idea amounts to the consolidation of $P \cup Q$ under certain constraints. In the base revision literature the unary operator ! is called a consolidation operator and results in a consistent subset of the input. A consolidation operator has been used in (Hansson 1997) to define *semi-revision*, which is defined as $\mathcal{B} *_? \alpha = (\mathcal{B} \cup \alpha)!$. The problem with semi-revision for our means is that it is non-prioritized, i. e. the success postulate is not satisfied in general.

We extend the idea of the semi-revision construction to be able to define a prioritized revision operator for logic programs. To this end we define a screened consolidation operator $!_R$ with R being a set of core sentences that are immune to change like in screened revision (Makinson 1997).

We propose the following set of postulates for a screened consolidation:

C-Screen $R \subseteq P!_R$.

C-Screen-Consistency If there exists some consistent X, $R \subseteq X \subseteq P$ then $P\,!_R$ is consistent.

C-Inclusion $P\,!_R \subseteq P$

C-Relevance If $r \in P$ and $r \notin P!_R$, then there is a set P' such that $P\,!_R \subseteq P' \subseteq P$. and that P' is consistent but $P' \cup \{r\}$ is inconsistent.

C-Fullness If $r \in P$ and $r \notin P!_R$, then $P!_R$ is consistent and $P!_R \cup \{r\}$ is inconsistent.

C-Screen-Uniformity Let R, R' and P be sets of rules and R and R' be consistent. If for all $X \subseteq P$, $R \cup X$ is consistent iff $R' \cup X$ is consistent then $P \cap (P \cup R)!_R = P \cap (P \cup R')!_{R'}$.

Note that as in the propositional case satisfaction of $C - Fullness$ implies satisfaction of $C - Relevance$.

Proposition 6. *Let $!_R$ be a screened consolidation operator. If $!_R$ satisfies $C - Fullness$ then it satisfies $C - Relevance$.*

Proof. Let $r \in P$ and $r \notin P!_R$. We have $P!_R \subseteq P!_R \subseteq P$ and from $C - Fullness$ it follows that $P!_R$ is consistent and $P!_R \cup \{r\}$ is inconsistent such that $C - Relevance$ is satisfied. □

We consider this as a basic set of postulates for a screened consolidation operation.

Definition 3 (Screened Consolidation)**.** *An operator $!_R$ is an operator of screened consolidation if and only if it satisfies C-Screen, C-Screen-Consistency, C-Inclusion, C-Relevance, C-Fullness and C-Screen-Uniformity.*

As stated earlier, the new postulates of Weak-Disjointness and Weak-Parallelism do not follow from the basic set of belief base postulates. This is also true for the postulate set for screened consolidation presented here. For their satisfaction we introduce the notion of topic independence as defined in (Hansson 1997), which bases on topicalizations.

Definition 4. *(Hansson 1997) Given a set P, a set $\mathfrak{B} \subseteq \mathcal{P}(P)$ is a* topicalization *of P if and only if:*

1. $P = \bigcup \mathfrak{B}$
2. *If $R \subseteq P$, then R is consistent if $R \cap B$ is consistent for all $B \in \mathfrak{B}$*

A topicalization $\mathcal{B}$ of P is hence a cover of P for which it holds that the consistency of each subset of P is only dependent on the consistency of its projection for each topic $B \in \mathcal{B}$.

C-Topic-Independence (Hansson 1997) If $\mathfrak{B}$ is a topicalization of P, then $P \setminus P!_R = \bigcup_{B \in \mathcal{B}} (B \setminus B!_R)$

The postulate of $C - Topic - Independence$ demands that given a topicalization of P each rule that has been removed from P by the consolidation operation $!_R$ is also removed from each topic $B \in \mathcal{B}$ with $r \in B$ by the respective consolidation of the topic and each rule removed by all consolidations of topics of P containing it, is removed from P.

Definition 5. *Given a screened consolidation operator $!_R$ we define a* multiple ASP base revision operator *by setting:*

$$P * Q = (P \cup Q)!_Q$$

Proposition 7. *Let $*$ be a multiple base revision operator defined as $P * Q = (P \cup Q)!_Q$. If $!_R$ is a screened consolidation operator that satisfies C-Topic-Independence then $*$ satisfies Success, Inclusion, Vacuity, Consistency, NM-Consistency, Relevance, Fullness, Uniformity, Weak-Disjointness and Weak-Parallelism.*

Proof. See Appendix for partial proof. □

We reduced the revision operation to a consolidation operation and characterized the latter by a set of postulates. We need a construction of a screened consolidation operator that is suitable for the application to logic programs. Two constructions of consolidation operations are available from belief base theory, partial meet and kernel consolidation (Hansson 1997). We use a partial meet construction here.

We start by defining a screened version of remainder sets for logic programs in which all remainder sets contain the screened set of rules.

Definition 6 (Screened Remainder Sets)**.** *For sets of sentences P and R with $R \subseteq P$ the set of* screened consistent remainder sets *of P, denoted by $P \perp_! R$ is such that for each $X \in P \perp_! R$:*

1. $R \subseteq X \subseteq P$
2. X *is consistent*
3. *There is no* X' *such that* $X \subset X' \subseteq P$ *and* X' *is consistent*

The definition of screened remainder sets differs from the original formulation only in the first condition which is $X \subseteq P$ originally.

In contrast to the propositional case, the intersection of remainders of logic programs is not necessarily consistent due to the non-monotonicity of logic programs.

Example 5. *Let* $P = \{a \leftarrow b., \neg a., b., \leftarrow not\ \neg a, not\ b.\}$. *The set of remainder sets with the empty screen are*

$$P \perp_! \emptyset = \quad \{ \quad \{a \leftarrow b.,\ b.,\ \leftarrow not\ \neg a, not\ b.\},$$
$$\{\neg a.,\ b.,\ \leftarrow not\ \neg a, not\ b.\},$$
$$\{a \leftarrow b.,\ \neg a.,\ \leftarrow not\ \neg a, not\ b.\}\}$$

The intersection of the first two remainders, $\{b., \leftarrow not\ \neg a, not\ b.\}$ *is consistent. The intersection of the last two remainders,* $\{a \leftarrow b., \leftarrow not\ \neg a, not\ b.\}$ *is inconsistent.*

This leaves us with the option of selecting exactly one of the remainders which also has the advantage of performing less change.

A selection function is specific to a certain belief base since it shall evaluate to the belief base if no remainder sets exits. We obtain a global selection function by making use of a two place selection function (Gärdenfors and Rott 1995) with the belief base a parameter.

Definition 7 (Global maxichoice selection function)**.** *Let* P *be a set of sentences.* γ_P *is a* selection function *for* P *if and only if for all sets of sentences* R

1. *If* $P \perp_! R \neq \emptyset$ *then* $\gamma_P(P \perp_! R) = X$ *for some* $X \in P \perp_! R$.
2. *If* $P \perp_! R = \emptyset$ *then* $\gamma_P(P \perp_! R) = P$.

A global maxichoice selection function *is a function* γ *such that for each* $P \subseteq \mathcal{L}$, $\gamma(P, \cdot) = \gamma_P(\cdot)$ *is a selection function for P. We drop the index* P *if it is obvious.*

With a global maxichoice selection function we can, in contrast to one place selection functions, we obtain operator that are not specific to one belief base is globally defined for all belief bases. Properties of consolidation operators which connect the consolidation results of several dependent belief bases can only be expressed for global operators.

We define a screened maxichoice consolidation operation based on a global maxichoice selection function.

Definition 8. *Let* P *and* R *be sets of sentences and* γ *a maxichoice selection function for* P. *The operation* $P!_R$ *such that*

$$P!_R = \gamma(P \perp_! R)$$

is a screened maxichoice consolidation *based on* γ.

The following representation theorem shows that any screened maxichoice consolidation satisfies the set of postulates for screened consolidation and, moreover, that any operation satisfying these postulates can be constructed as a screened maxichoice consolidation.

Proposition 8. *An operation* $!_R$ *is an operation of screened maxichoice consolidation if and only if it satisfies C-Inclusion, C-Screen-Consistency, C-Screen, C-Fullness and C-Uniformity.*

Proof. See Appendix. □

Definition 9. *A global selection function* γ *is* monotone *if and only if for all* $P, P' \subseteq \mathcal{L}$, *if for each* $X \in P \perp_! \emptyset$ *there exists some* $X' \in P' \perp_! \emptyset$ *such that* $P \setminus X = P \setminus X'$ *then* $P \setminus \gamma(P \perp_! \emptyset) = P \setminus \gamma(P' \perp_! \emptyset)$.

An operator of screened consolidation is monotone *if and only if it is based on a monoton selection function.*

Proposition 9. *If a screened maxichoice consolidation operator is based on a monotone maxichoice selection function then it satisfies topic-independence.*

Discussion

We have developed a base revision description and construction for belief bases represented by extended logic programs under the answer set semantics. As already discussed in the introduction the majority of work on the dynamics in logic programming has focused on the implementation of inconsisency handling in sequences of logic programs via logic programs, e. g. (Alferes et al. 2005; Delgrande, Schaub, and Tompits 2007; Eiter et al. 2002; Zhang 2006; Krümpelmann and Kern-Isberner 2008; Krümpelmann 2012). The few principle based approaches used the classic AGM postulates as reference and developed alternative postulates as discussed previously, e. g. (Eiter et al. 2002; Alferes et al. 2005). The state transition system approach presented in (Kudo and Murai 2004) was meant to satisfy base revision postulates for the revision by a fact, but no formal results are shown. The only construction adapted from classic belief revision is the distance based construction via SE-Models used in (Delgrande et al. 2008). Approaches to belief revision in other non-monotonic formalisms are often not based on principles and very specific to the underlying logic (Witteveen and van der Hoek 1997; Billington et al. 1999). A principle based approach to contraction operations in logic programs has been considered in (Krümpelmann and Kern-Isberner 2010).

The results presented in this paper show that the base revision approach to belief revision is applicable to revision of logic programs and formalizes adequate properties such an operation should satisfy and constructions satisfying them. New postulates are presented. This resembles an important foundation for base revision approaches to answer set programming. Future work involves the further investigation of desirable postulates for ASP base revision, comparison to other syntax base approaches and implementation.
Acknowledgements: This work has been supported by the DFG, Collaborative Research Center SFB876, Project A5. (`http://sfb876.tu-dortmund.de`)

References

[Alchourron, Gärdenfors, and Makinson 1985] Alchourron, C. E.; Gärdenfors, P.; and Makinson, D. 1985. On the logic of theory change: Partial meet contraction and revision

functions. *The Journal of Symbolic Logic* Vol. 50, No. 2 (Jun., 1985):510–530.

[Alferes et al. 2005] Alferes, J. J.; Banti, F.; Brogi, A.; and Leite, J. A. 2005. The refined extension principle for semantics of dynamic logic programming. *Studia Logica: An International Journal for Symbolic Logic* 79(1):pp. 7–32.

[Billington et al. 1999] Billington, D.; Antoniou, G.; Governatori, G.; and Maher, M. 1999. Revising nonmonotonic theories: The case of defeasible logic. In Burgard, W.; Cremers, A.; and Cristaller, T., eds., *KI-99: Advances in Artificial Intelligence*, volume 1701. Springer. 695–695.

[Delgrande et al. 2008] Delgrande, J.; Schaub, T.; Tompits, H.; and Woltran, S. 2008. Belief revision of logic programs under answer set semantics. In Brewka, G., and Lang, J., eds., *Proceedings of the Eleventh International Conference on Principles of Knowledge Representation and Reasoning (KR'08)*, 411–421. AAAI Press.

[Delgrande et al. 2009] Delgrande, J.; Schaub, T.; Tompits, H.; and Woltran, S. 2009. Merging logic programs under answer set semantics. In *Proceedings of the 25th International Conference on Logic Programming*, ICLP '09, 160–174. Berlin, Heidelberg: Springer-Verlag.

[Delgrande, Schaub, and Tompits 2007] Delgrande, J. P.; Schaub, T.; and Tompits, H. 2007. A preference-based framework for updating logic programs. In Baral, C.; Brewka, G.; and Schlipf, J. S., eds., *Proc. of the 9th Int'l Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'07)*, volume 4483, 71–83. Springer.

[Delgrande 2008] Delgrande, J. P. 2008. Horn clause belief change: Contraction functions. In Brewka, G., and Lang, J., eds., *Principles of Knowledge Representation and Reasoning: Proceedings of the Eleventh International Conference, KR 2008, Sydney, Australia, September 16-19, 2008*, 156–165. AAAI Press.

[Delgrande 2010] Delgrande, J. P. 2010. An approach to revising logic programs under the answer set semantics. *Proceedings of the 13th International Workshop on Non-Monotonic Reasoning (NMR)*.

[Eiter and Fink 2003] Eiter, T., and Fink, M. 2003. Uniform equivalence of logic programs under the stable model semantics. In Palamidessi, C., ed., *Logic Programming*, volume 2916. Springer Berlin / Heidelberg. 224–238.

[Eiter et al. 2002] Eiter, T.; Fink, M.; Sabbatini, G.; and Tompits, H. 2002. On properties of update sequences based on causal rejection. *Theory and Practice of Logic Programming* 2(6):711–767.

[Fermé and Hansson 2011] Fermé, E., and Hansson, S. 2011. Agm 25 years. *Journal of Philosophical Logic* 40:295–331. 10.1007/s10992-011-9171-9.

[Gärdenfors and Rott 1995] Gärdenfors, P., and Rott, H. 1995. *Belief revision*. Oxford, UK: Oxford University Press.

[Gelfond and Lifschitz 1988] Gelfond, M., and Lifschitz, V. 1988. The stable model semantics for logic programming. In *ICLP/SLP*, 1070–1080. MIT Press.

[Hansson 1991] Hansson, S. O. 1991. *Belief base dynamics*. Ph.D. Dissertation, Uppsala University, Faculty of Arts.

[Hansson 1997] Hansson, S. O. 1997. Semi-revision. *Journal of Applied Non-Classical Logics* 7(2).

[Hansson 2001] Hansson, S. O. 2001. *A Textbook of Belief Dynamics*. Norwell, MA, USA: Kluwer Academic Publishers.

[Hué, Papini, and Würbel 2009] Hué, J.; Papini, O.; and Würbel, E. 2009. Merging belief bases represented by logic programs. In Sossai, C., and Chemello, G., eds., *Proc. of the 10th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty, (ECSQARU09)*, 371–382. Springer.

[Krümpelmann and Kern-Isberner 2008] Krümpelmann, P., and Kern-Isberner, G. 2008. Propagating credibility in answer set programs. In Schwarz, S., ed., *Proc. of the 22nd Workshop on (Constraint) Logic Programming (WLP08), Dresden, Germany*.

[Krümpelmann and Kern-Isberner 2010] Krümpelmann, P., and Kern-Isberner, G. 2010. On belief dynamics of dependency relations for extended logic programs. In *Proceedings of the 13th International Workshop on Non-Monotonic Reasoning (NMR10)*.

[Krümpelmann 2012] Krümpelmann, P. 2012. Dependency semantics for sequences of extended logic programs. *Logic Journal of the IGPL* doi: 10.1093/jigpal/jzs012.

[Kudo and Murai 2004] Kudo, Y., and Murai, T. 2004. A method of belief base revision for extended logic programs based on state transition diagrams. In Negoita, M.; Howlett, R.; and Jain, L., eds., *Knowledge-Based Intelligent Information and Engineering Systems*. Springer.

[Makinson 1997] Makinson, D. 1997. Screened revision. *Theoria* 63(1-2):14–23.

[Osorio and Cuevas 2007] Osorio, M., and Cuevas, V. 2007. Updates in answer set programming: An approach based on basic structural properties. *Theory Pract. Log. Program.* 7(4):451–479.

[Sefránek and Siika 2006] Sefránek, J., and Siika, J. 2006. Irrelevant updates of nonmonotonic knowledge bases. In *ECAI 2006, 17th European Conference on Artificial Intelligence*, 771–772.

[Slota and Leite 2010] Slota, M., and Leite, J. 2010. On semantic update operators for answer-set programs. In Coelho, H., and Wooldridge, M., eds., *Proceedings of the 19th European Conference on Artificial Intelligence, ECAI 2010*.

[Slota and Leite 2012] Slota, M., and Leite, J. a. 2012. Robust equivalence models for semantic updates of answer-set programs. In *13th International Conference on Principles of Knowledge Representation and Reasoning*. to appear.

[Witteveen and van der Hoek 1997] Witteveen, G., and van der Hoek, W. 1997. A general framework for revising nonmonotonic theories. In *Logic Programming And Nonmonotonic Reasoning*. Springer.

[Woltran 2008] Woltran, S. 2008. A common view on strong, uniform, and other notions of equivalence in answer-set programming. *Theory Pract. Log. Program.* 8(2):217–234.

[Zhang 2006] Zhang, Y. 2006. Logic program-based updates. *ACM Transactions on Computational Logic (TOCL)*.

Appendix: Proofs

Proposition 3. 1. Given a program P with $AS(P) = \emptyset$ and a tautologic program $P_\top$ then consistency demands that $AS(P * P_\top) \neq \emptyset$, in contradiction to Tautology$_{AS}$

2. Given a program P and a tautologic program $P_\top$ such that $P \cap P_\top = \emptyset$. It follows from Success that $P * P_\top \neq P$, in contradiction to Tautology$_P$. □

Proposition 5. 1. Given Q_1 and Q_2 with disjoint sets of literals and some program P. Assume that $Q_1 \cup P$ is inconsistent and $Q_2 \cup P$ is consistent and that P, Q_1 and Q_2 are strict programs. From Consistency it follows that $AS(P * (Q_1 \cup Q_2)) \neq \emptyset$. It follows from Vacuity that $P \subseteq P * Q_2$ and from Success that $Q_1 \subseteq P * Q_1$. Parallelism$_{AS}$ demands that $AS(P*(Q_1 \cup Q_2)) = AS((P*Q_1) \cup (P*Q_2))$. It holds that $P \cup Q_1 \subseteq (P * Q_1) \cup (P * Q_2)$. Since $P \cup Q_1$ is inconsistent and P and Q_1 are strict programs it holds that $AS((P * Q_1) \cup (P * Q_2)) = \emptyset$, in contradiction to Consistency. Given $P = P_1 \cup P_2$ and P_1 and P_2 have disjoint sets of literals and some program Q such that $P_1 \cup Q$ and $P_2 \cup Q$ are consistent and $P \cup Q$ is inconsistent. Assume P and Q are strict programs. From Consistency it follows that $AS(P * Q) \neq \emptyset$. From Vacuity it follows that $P_1 \cup Q \subseteq P_1 * Q$ and $P_2 \cup Q \subseteq P_2 * Q$. Disjointness$_{AS}$ demands that $AS(P * Q) = AS((P_1 * Q) \cup (P_2 * Q))$. It holds that $P \cup Q \subseteq (P_1 * Q) \cup (P_2 * Q)$. Since $P \cup Q$ is inconsistent and P and Q are strict programs we have that $AS((P_1 * Q) \cup (P_2 * Q)) = \emptyset$ in contradiction to Consistency. □

Proposition 7. C-Screen implies that $Q \subseteq (P \cup Q)!_Q = P * Q$, i. e. *Success*. From C-Inclusion follows that $(P \cup Q)!_Q = P * Q \subseteq P \cup Q$, i. e. *Inclusion*. From C-Screen-Consistency follows that if Q is consistent then $(P \cup Q)!_Q = P * Q$ is consistent, i. e. *Consistency*. The satisfaction of C-Fullness implies that if $r \in (P \cup Q) \setminus P * Q$ then $P * Q$ is consistent and $(P * Q) \cup \{r\}$ is inconsistent, i. e. *Fullness* is satisfied. Note that Relevance follows from Fullness and Vacuity follows from Relevance and Inclusion.

The proofs of Weak Disjointness$_P$ and Weak Parallelism$_P$ are left out due to the page limitation. □

Proposition 8. *Construction to postulates: C-Screen* We need to show that for all $R \subseteq P$, $R \subseteq \gamma_P(P \perp_! R)$. If $P \perp_! R = \emptyset$ then $\gamma_P(P \perp_! R) = P$ and $R \subseteq P$ by definition. If $P \perp_! R \neq \emptyset$ then by definition of screened remainder sets $R \subseteq X$ for all $X \in (P \perp_! R)$ it directly follows that $R \subseteq \gamma_P(P \perp_! R)$.

C-Screen-Consistency We need to show that for all $R \subseteq P$, if there exists some consistent X, $R \subseteq X \subseteq P$ then $\gamma_P(P \perp_! R)$ is consistent. In the case of $P \perp_! R = \emptyset$ there does not exits a consistent set X, $R \subseteq X \subseteq P$. In the case of $P \perp_! R \neq \emptyset$ by definition of screened remainder sets each $X \in (P \perp_! R)$ is consistent and by definition $\gamma(P \perp_! R)$ is consistent.

C-Inclusion We need to show that for all $R \subseteq P$, $\gamma_P(P \perp_! R) \subseteq P$. If $P \perp_! R = \emptyset$ then $\gamma_P(P \perp_! R) = P$. If $P \perp_! R \neq \emptyset$ then by definition of screened remainder sets for all $X \in (P \perp_! R)$, $X \subseteq P$ and thus by definition $\gamma(P \perp_! R) \subseteq P$.

C-Fullness We need to show that for all R, if $r \in P$ and $r \notin \gamma(P \perp_! R)$ then $\gamma(P \perp_! R)$ is consistent and $\gamma(P \perp_! R) \cup \{r\}$ is inconsistent. If $P \perp_! R = \emptyset$ then $\gamma_P(P \perp_! R) = P$ and there is no $r \in R$ and $r \notin \gamma(P \perp_! R)$ such that C-Fullness is satisfied vacuously. If $P \perp_! R \neq \emptyset$ and $r \notin \gamma(P \perp_! R)$ then from $\gamma(P \perp_! R) = X$ for some $X \in P \perp_! R$ it follows that X is consistent and from condition 3 of Definition 6 and $X \subset X \cup \{r\} \subseteq P$ it follows that $X \cup \{r\}$ is inconsistent such that C-Fullness is satisfied.

C-Screen-Uniformity We need to show that all R, R' and P, and a selection function for P it holds that if for all $X \subseteq P$, $R \cup X$ is consistent iff $R' \cup X$ is consistent then $\gamma(P \perp_! R) \cap P = \gamma(P \perp_! R') \cap P$. If for all $X \subseteq P$, $R \cup X$ is consistent iff $R' \cup X$ is consistent then it follows that $P \perp_! R = P \perp_! R'$ and by definition of a selection function it holds that $\gamma(P \perp_! R) = \gamma(P \perp_! R')$.

Postulates to construction: Let $!_R$ be an operation for P that satisfies C-Screen, C-Screen-Consistency, C-Inclusion, C-Fullness and C-Uniformity. Let γ be such that: $\gamma(P \perp_! R) = P!_R$ We need to show that (1) γ is a well-defined function, that (2) γ is a maxichoice selection function, and that (3) for all R, $\gamma(P \perp_! R) = P!_R$. Part (1): γ is a well-defined function if for all $P \perp_! R = P \perp_! R'$ it holds that $\gamma(P \perp_! R) = \gamma(P \perp_! R')$. Suppose $P \perp_! R = P \perp_! R'$ then it follows from C-Screen-Uniformity that $P!_R = P!_{R'}$. By definition of γ it follows that $\gamma(P \perp_! R) = \gamma(P \perp_! R')$.

Part (2): For γ to be a maxichoice selection function we have to show that if $P \perp_! R \neq \emptyset$ then $\gamma(P \perp_! R) = X$ for some $X \in P \perp_! R$ and that if $P \perp_! R = \emptyset$ then $\gamma(P \perp R_!) = P$.

If $P \perp_! R \neq \emptyset$ there exits some consistent X, $R \subseteq X \subseteq P$ and it follows from C-Screen Consistency that $P!_R$ is consistent. From C-Inclusion follows that $P!_R \subseteq P$ and from C-Screen that $R \subseteq P!_R$. To show that $P!_R \in P \perp_! R$ we show that if $P!_R \subset H \subseteq P$ then H is inconsistent. Let H be such that $P!_R \subset H \subseteq P$. Then there exists some $r \in H$ and $r \notin P!_R$ and from C-Fullness it follows that $P!_R \cup \{r\}$ is inconsistent. It holds that $P!_R \cup \{r\} \subseteq H$ and $P!_R$ is due to the satisfaction of C-Fullness a maximal consistent set, consequently H is inconsistent.

If $P \perp_! R = \emptyset$ then there does not exits a consistent set X, $R \subseteq X \subseteq P$. Suppose to the contrary that $P \not\subseteq P!_R$ and let $r \in P \setminus P!_R$. Then it follows from C-Screen-Fullness that $P!_R$ is consistent and $P!_R \cup \{r\}$ is inconsistent. This is not possible since there does not exits a consistent set X, $R \subseteq X \subseteq P$ and $R \subseteq P!_R$ by C-Screen and $P!_R \subseteq P$ by C-Inclusion. Therefore we have shown that $P \subseteq P!_R$ and since it follows from inclusion that $P!_R \subseteq P$ we have $P = P!_R$ such that by definition of γ we have $\gamma(P \perp_! R) = P$ which was to show.

Part (3): That for all R, $\gamma(P \perp_! R) = P!_R$ follows directly from the construction. □