Approximating answer sets of unitary Lifschitz-Woo programs

Victor W. Marek[1], Inna Pivkina[2], and Mirosław Truszczyński[1]

[1] Department of Computer Science, University of Kentucky
Lexington, KY 40506-0046, USA
[2] Department of Computer Science, New Mexico State University
P.O. Box 30001, MSC CS, Las Cruces, NM 88003, USA

Abstract. We investigate several techniques for approximation of answer sets for a subclass of general logic programs of Lifschitz and Woo. The class we consider consists of programs that are unitary, i.e. allow for a single literal in the head (negation as failure is allowed in those literals). We compare three different classes of approximations and obtain results on the relationship between these schemes. Since unitary general logic programs are equivalent to revision programs we obtain results on approximations of justified revisions of databases by revision programs.

1 Introduction

General logic programs were introduced by Lifschitz and Woo [LW92]. Their syntax follows closely that of disjunctive logic programs but there is one essential difference. The operator $\mathbf{not}$, representing the *default negation* is no longer confined to the bodies of program rules but may appear in their heads, as well. Lifschitz and Woo [LW92] showed that the semantics of answer sets introduced for disjunctive logic programs in [GL91] can be lifted to the class of general logic programs.

In this paper, we study the class of those general programs that do not contain disjunctions in the heads of their rules. We call such programs *unitary*. Unitary general programs are of interest for two reasons. First, they go beyond the class of normal logic programs by allowing the default-negation operator in the rule heads. Second, in a certain precise sense, unitary general programs are equivalent to the class of revision programs [MT98,MPT02], which provide a formalism for describing and enforcing database revisions. Consequently, results for unitary general programs extend to the case of revision programs.

The problem we focus on in this paper is that of approximating answer sets of unitary general programs. The problem to decide whether a unitary logic program has an answer set is NP-complete[3]. Consequently, computing answer sets of unitary general programs is hard and it is important to establish efficient ways to approximate them. On one hand, such approximations can be sufficient for some reasoning tasks. On the other

[3] Without the restriction to unitary programs (and assuming that the polynomial hierarchy does not collapse) the problem is even harder — Σ_2^P-complete.

hand, they can be used by programs computing answer sets to prune the search space and can improve their performance significantly.

In the case of normal logic programs the well-founded model [VRS88] provides an effective approximation to all answer sets[4]. It can be computed in polynomial time and is known to provide an effective pruning mechanism for programs computing stable models [SNV95,SNS02]. An obvious approach to the problem at hand seems to be then to extend the well-founded model and its properties to the class of unitary programs. However, despite similarities between normal and unitary programs, no counterpart of the well-founded model has been proposed for the latter class so far, and whether it can be done remains unresolved.

Thus, we approach the problem not by attempting to generalize the well-founded semantics but by exploiting this semantics in some other, less direct ways. Namely, we introduce three operators for unitary general programs and use them to define the approximations. The first two operators are antimonotone and are closely related to operators behind the well-founded semantics of normal logic programs. Iterating them yields *alternating* sequences. We use the limits of these sequences to construct our first two approximations to answer sets of unitary general programs. The two approximations we obtain in this way are not comparable (neither is stronger than the other one). The third operator is not antimonotone in general. However, in the case of unitary general programs that have answer sets, iterating this operator results in an alternating sequence and the limit of this sequence yields yet another approximation to answer sets of unitary general programs. We show that this third approximation is stronger than the other two. We also show that all three approaches imply sufficient conditions for the *non-existence* of answer sets of unitary programs.

As we noted, unitary programs are related to revision programs [MT98,MPT99]. Having introduced approximations to answer sets of unitary general programs, we show that our results apply in a direct way to the case of revision programming.

All programs we consider in the paper are *finite*. That assumption simplifies arguments. However, all our results can be extended to the case of infinite programs.

2 Preliminaries

Atoms and literals. In the paper we consider a fixed set U of (propositional) atoms. Expressions of the form a and $\mathbf{not}(a)$, where $a \in U$, are *literals* (over U). We denote the set of all literals over U by $Lit(U)$. A set of literals $L \subseteq Lit(U)$ is *coherent* if there is no $a \in U$ such that both $a \in L$ and $\mathbf{not}(a) \in L$. A set of literals $L \subseteq Lit(U)$ is *complete* if for every $a \in U$, $a \in L$ or $\mathbf{not}(a) \in L$ (it is possible that for some a, both $a \in L$ and $\mathbf{not}(a) \in L$).

For a set M of atoms, $M \subseteq U$, we define

$$\mathbf{not}(M) = \{\mathbf{not}(a) \colon a \in M\} \ \text{ and } \ M^c = M \cup \mathbf{not}(U \setminus M).$$

The mapping $M \mapsto M^c$ is a bijection between subsets of U and coherent and complete sets of literals contained in $Lit(U)$.

[4] In the context of normal logic programming, answer sets are more commonly known as *stable models*.

Unitary general programs. A *unitary general logic program*, or *UG-program* is a collection of rules of the form:

$$\alpha \leftarrow \alpha_1, \ldots, \alpha_m \tag{1}$$

where α, $\alpha_1, \ldots, \alpha_m$ are literals from $Lit(U)$ The literal α is the *head* of the rule. The set of literals $\{\alpha_1, \ldots, \alpha_m\}$ is the *body* of the rule.

Let P be a UG-program. We write P^+ (respectively, P^-) to denote programs consisting of all rules in P that have an atom (respectively, a negated atom) as the head.

Satisfaction and models. A set of atoms $M \subseteq U$ *satisfies* (is a *model* of) an atom $a \in U$ (respectively, a literal $\mathbf{not}(a) \in Lit(U)$), if $a \in M$ (respectively, $a \notin M$). The concept of satisfaction (being a model of) extends in a standard way to rules and programs. As usual, we write $\models$ to denote the satisfaction relation.

Sets of literals closed under UG-programs. In addition to models, we also associate with a UG-program P sets of literals that are closed under rules in P. A set L of literals is *closed* under a UG-program P if for every rule $r = \alpha \leftarrow Body \in P$ such that $Body \subseteq L$, $\alpha \in L$. One can show that every UG-program P has a least set of literals closed under its rules[5]. We denote it by P^*. We observe that if P is a definite Horn program, P^* consists of atoms only and coincides with the least model of P.

Stable models of normal logic programs. Models are too weak for knowledge representation applications. In the case of normal logic programs, the appropriate semantic concept is that of a stable model. We recall that according to the original definition [GL88], a set of atoms M is a stable model of a normal logic program P if

$$[P^M]^* = M, \tag{2}$$

where P^M is the *Gelfond-Lifschitz* reduct of P with respect to M. The following characterization of stable models is well known [BTK93]: M is a stable model of a normal logic program P if and only if

$$[P \cup \mathbf{not}(U \setminus M)]^* \cap U = M. \tag{3}$$

Answer sets of UG-programs. Lifschitz and Woo [LW92] extended the concept of a stable model to the case of arbitrary general programs and called the resulting semantic object an *answer set*. Rather than to give the original definition from [LW92], we recall a basic characterization of answer sets of UG-programs that will be of use in the paper. Its proof can be found in [MPT99].

Proposition 1. *Let P be a UG-program. A set of atoms M is an answer set to P if and only if M is a stable model of P^+ and a model of P^-. In particular, if M is an answer set to P then M is a model of P.*

Alternating sequences. All approximations to answer sets of UG-programs we study in this paper are defined in terms of alternating sequences and their limits. A sequence (X_i) of sets of literals is *alternating* if

[5] If we treat literals $\mathbf{not}(a)$ as new atoms, P becomes a Horn program and its least model is the least set of literals closed under P.

1. $X_0 \subseteq X_2 \subseteq X_4 \subseteq \ldots$
2. $X_1 \supseteq X_3 \supseteq X_5 \supseteq \ldots$
3. $X_{2i} \subseteq X_{2i+1}$, for every non-negative integer i.

If (X_i) is an alternating sequence, we define $X^l = \bigcup_{i=0}^{\infty} X_{2i}$ and $X^u = \bigcap_{i=0}^{\infty} X_{2i+1}$. We call the pair (X^l, X^u) the *limit* of the alternating sequence (X_i). It follows directly from the definition that for every non-negative integers i and j,

$$X_{2i} \subseteq X^l \subseteq X^u \subseteq X_{2j+1}$$

Alternating sequences are often defined by means of operators that are antimonotone. An operator γ defined on $Lit(U)$ is *antimonotone* if for every two sets $X \subseteq Y \subseteq Lit(U)$, $\gamma(Y) \subseteq \gamma(X)$. Let γ be antimonotone. We define $X_0 = \emptyset$ and $X_{i+1} = \gamma(X_i)$. It is well known (and easy to show) that the sequence (X_i) is alternating. We call (X_i) the *alternating* sequence of γ.

We will consider in the paper the following two operators:

$$\gamma_{P,U}(X) = [P \cup \mathbf{not}(U \setminus X)]^* \cap U \ \text{ and } \ \gamma_P(X) = [P \cup \mathbf{not}(U \setminus X)]^*.$$

Both operators are antimonotone and give rise to alternating sequences, say (W_i) and (Y_i). Let (W^l, W^u) and (Y^l, Y^u) be the limits of these sequences, respectively. One can verify that these limits form *alternating pairs*. That is, we have

$$\gamma_{P,U}(W^l) = W^u \ \text{ and } \ \gamma_{P,U}(W^u) = W^l \tag{4}$$

and

$$\gamma_P(Y^l) = Y^u \ \text{ and } \ \gamma_P(Y^u) = Y^l. \tag{5}$$

One can show that if P is a normal logic program then the alternating sequence of $\gamma_{P,U}$ is precisely the alternating sequence defining the well-founded semantics of P [VRS88,Van93].

One can also show that the limit of the alternating sequence of γ_P is the well-founded model of the normal logic program P' obtained from P by replacing every literal $\mathbf{not}(a)$ with a *new* atom, say a', and adding rules of the form $a' \leftarrow \mathbf{not}(a)$ (the claim holds modulo the correspondence $a' \leftrightarrow \mathbf{not}(a)$). The mapping $P \mapsto P'$ was introduced and studied in [PT95] in the context of revision programs.

Approximating sets of atoms. Let M be a set of atoms. Every pair of sets (T, S) that *approximates* M, that is, such that $T \subseteq M \subseteq S$, implies a lower bound on the complete representation M^c of M:

$$T \cup \{\mathbf{not}(U \setminus S)\} \subseteq M^c.$$

Conversely, every set L of literals such that $L \subseteq M^c$ determines an *approximation* (T, S) of M, where $T = U \cap L$ and $S = \{a \in U \colon \mathbf{not}(a) \notin L\}$. Indeed,

$$U \cap L \subseteq M \subseteq \{a \in U \colon \mathbf{not}(a) \notin L\}.$$

In this way, we establish a bijection between approximations to a set of atoms M and subsets of M^c. It follows that approximations of answer sets can be represented as subsets of their complete representations. We have the following fact.

Proposition 2. *Let P be a UG-program and let T and S be two sets of atoms. For every answer set M of P, if $T \subseteq M \subseteq S$ then $[P \cup T \cup \mathbf{not}(U \setminus S)]^* \subseteq M^c$.*

Proof: We have $T \subseteq M \subseteq S$. Thus, $T \cup \mathbf{not}(U \setminus S) \subseteq M^c$. Let $r = \alpha \leftarrow Body$ be a rule in P such that $Body \subseteq M^c$. It follows that M satisfies the body of r. Since M is an answer set of P, M satisfies α and so, $\alpha \in M^c$. Thus, $T \cup \mathbf{not}(U \setminus S) \subseteq M^c$ and M^c is closed under P. Consequently, $[P \cup T \cup \mathbf{not}(U \setminus S)]^* \subseteq M^c$. □

In the case of normal logic programs, the well-founded model, that is, the limit (W^l, W^u) of the alternating sequence (W_i) of the operator $\gamma_{P,U}$, approximates every stable model (if they exist) and, in some cases determines the existence of a unique stable model.

Theorem 1 ([VRS88,Lif96]). *Let (W^l, W^u) be the well-founded model of a normal logic program P.*

1. *For every stable model M of P, $W^l \cup \mathbf{not}(U \setminus W^u) \subseteq M^c$*
2. *If $W^l = W^u$, then W^l is a unique stable model for P.*

In the remainder of the paper, we will propose approximations to answer sets of UG-programs generalizing Theorem 1.

3 Approximating answer sets using operators $\gamma_{P,U}$ and γ_P

Our first approach exploits the fact that every answer set of a UG-program P is a stable model of P^+ (Proposition 1). Let P be a UG-program and let (W^l, W^u) be the limit of the alternating sequence of the operator $\gamma_{P^+,U}$. As we observed, (W^l, W^u) is the well-founded model of P^+. We define

$$Appx_1(P) = [P \cup \mathbf{not}(U \setminus W^u)]^*.$$

By (4), $W^l = [P \cup \mathbf{not}(U \setminus W^u]^* \cap U$. Hence, $W^l \subseteq Appx_1(P)$ and so, $Appx_1(P)$ contains all literals that are true in the well-founded model (W^l, W^u).

Theorem 2. *Let P be a UG-program. For every answer set M of P, $Appx_1(P) \subseteq M^c$. In addition, if $Appx_1(P)$ is incoherent then P has no answer sets.*

Proof: Let M be an answer set of P. By Proposition 1, M is a stable model of P^+. Let (W^l, W^u) be the well-founded model of P^+. By Theorem 1, $\mathbf{not}(U \setminus W^u) \subseteq M^c$. Moreover, since M is an answer set of P, M is a model of P (Proposition 1, again) and so, M^c is closed under P. Since $Appx_1(P)$ is the least set of literals containing $\mathbf{not}(U \setminus W^u)$ and closed under P, $Appx_1(P) \subseteq M^c$, as claimed. The second part of the assertion follows from the first one. □

We will illustrate this approach with an example.

Example 1. Let us consider the following UG-program P:

$$\begin{aligned} a &\leftarrow \mathbf{not}(b), \mathbf{not}(c) \\ c &\leftarrow c, \mathbf{not}(b) \\ b &\leftarrow \mathbf{not}(d) \\ d &\leftarrow \mathbf{not}(b) \\ \mathbf{not}(b) &\leftarrow \end{aligned}$$

All but the last rule belong to P^+. The operator $\gamma_{P^+,U}$ determines the following alternating sequence (W_i) of sets:

$$\emptyset \mapsto \{a, b, d\} \mapsto \emptyset \ldots .$$

It follows that the well-founded model of P^+ is $(W^l, W^u) = (\emptyset, \{a, b, d\})$. Consequently,

$$Appx_1(P) = [P \cup \{\mathbf{not}(c)\}]^* = \{a, d, \mathbf{not}(b), \mathbf{not}(c)\}.$$

In this case, the well-founded model of P^+ alone provides a weak bound on answer sets of P. The improved bound $Appx_1(P)$, which closes the model under P, provides a much stronger approximation. In fact, only one set M is approximated by $\{a, d, \mathbf{not}(b), \mathbf{not}(c)\}$. This set is $\{a, d\}$ and it happens to be a unique answer set of P.

Let $Q = P \cup \{\mathbf{not}(a) \leftarrow d\}$. Since $Q^+ = P^+$, it follows that $Appx_1(Q) = [Q \cup \{\mathbf{not}(c)\}]^* = \{a, d, \mathbf{not}(a), \mathbf{not}(b), \mathbf{not}(c)\}$. Since $Appx_1(Q)$ is incoherent, Q has no answer sets, a fact that can be verified directly. $\square$

The approximation $Appx_1(P)$, where P is the first program from Example 1, is complete and coherent, and we noted that the unique set of atoms $Appx_1(P)$ approximates is a unique answer set of P. It is a general property extending Theorem 1(2).

Corollary 1. *Let P be a UG-program. If $Appx_1(P)$ is coherent and complete then $Appx_1(P) \cap U$ is a unique answer set of P.*

Proof: Since $Appx_1(P)$ is coherent and complete, Theorem 2 implies that P has at most one answer set. To prove the assertion it is then enough to show that $M = Appx_1(P) \cap U$ is an answer set of P.

Let (W^l, W^u) be the well-founded model of P^+. Since $Appx_1(P) = [P \cup \mathbf{not}(U \setminus W^u)]^*$, $[P \cup \mathbf{not}(U \setminus W^u)]^*$ is coherent and complete. Consequently,

$$M^c = [P \cup \mathbf{not}(U \setminus W^u)]^*.$$

It follows that $\mathbf{not}(U \setminus W^u) \subseteq \mathbf{not}(U \setminus M)$. Thus, $M^c \subseteq [P \cup \mathbf{not}(U \setminus M)]^*$. It also follows that M^c is closed under the rules in P. Since $\mathbf{not}(U \setminus M) \subseteq M^c$, $[P \cup \mathbf{not}(U \setminus M)]^* \subseteq M^c$. Thus,

$$M^c = [P \cup \mathbf{not}(U \setminus M)]^*.$$

It follows now that M is a model of P^-. Moreover, it also follows that $M = [(P^+ \cup \mathbf{not}(U \setminus M)]^*$ and so, M is a stable model of P^+. Thus, M is an answer set of P. $\square$

We will now introduce another approximation to answer sets of a UG-program P. This time, we will use the operator γ_P. Let Y_i be the alternating sequence of the operator γ_P and let (Y^l, Y^u) be the limit of (Y_i). We define

$$Appx_2(P) = Y^l.$$

Theorem 3. *Let P be a UP-program. If M is an answer-set for P then $Appx_2(P) \subseteq M^c$. In addition, if $Appx_2$ is incoherent, then P has no answer sets.*

Proof: Let M be an answer set of P and let (Y_i) be the alternating sequence for the operator γ_P. We will show by induction that for every $i \geq 0$, $Y_{2i} \cap U \subseteq M \subseteq Y_{2i+1}$.

Since $Y_0 = \emptyset$, $Y_0 \cap U \subseteq M$. We will now assume that $Y_{2i} \cap U \subseteq M$ and show that $M \subseteq Y_{2i+1}$. Our assumption implies that $\mathbf{not}(U \setminus M) \subseteq \mathbf{not}(U \setminus Y_{2i})$. Thus, since M is a stable model of P^+, it follows from (3) that

$$M = [P^+ \cup \mathbf{not}(U \setminus M)]^* \cap U \subseteq [P \cup \mathbf{not}(U \setminus M)]^* \subseteq [P \cup \mathbf{not}(U \setminus Y_{2i})]^* = Y_{2i+1}.$$

Next, we assume that $M \subseteq Y_{2i+1}$ and show that $Y_{2i+2} \cap U \subseteq M$. The assumption implies that $\mathbf{not}(U \setminus Y_{2i+1}) \subseteq \mathbf{not}(U \setminus M)$. Thus,

$$\begin{aligned} Y_{2i+2} \cap U &= [P \cup \mathbf{not}(U \setminus Y_{2i+1})]^* \cap U \subseteq [P \cup \mathbf{not}(U \setminus M)]^* \cap U \\ &= [P^+ \cup \mathbf{not}(U \setminus M)]^* \cap U = M. \end{aligned}$$

The last but one equality follows from the fact that M is a model of P^- and the last inequality follows from the fact that M is a stable model of P^+.

¿From the claim it follows that $M \subseteq Y^u$. Thus, $\mathbf{not}(U \setminus Y^u) \subseteq M^c$. Since M is a model of P, M^c is closed under P. Thus, $Y^l = [P \cup \mathbf{not}(U \setminus Y^u)]^* \subseteq M^c$. □

As before, if the approximation provided by $Appx_2(P)$ is complete and coherent, P has a unique answer set.

Corollary 2. *Let P be a UG-program such that $Appx_2(P)$ is complete and coherent. Then, $Appx_2(P) \cap U$ is a unique answer set of P.*

Proof: Let $M = Appx_2(P) \cap U$. By Theorem 3, it suffices to show that M is an answer set of P.

Let (Y_i) be the alternating sequence of γ_P and let (Y^l, Y^u) be its limit. Since $Appx_2(P) = Y^l$ is complete and coherent, $Y^l = M^c$. By (5), $M^c = Y^l = [P \cup \mathbf{not}(U \setminus Y^u)]^*$. Thus, M^c is closed under rules in P. Consequently, M is a model of P and of P^-, in particular.

We also have $M \subseteq M^c = Y^l \subseteq Y^u$. Thus, $\mathbf{not}(U \setminus Y^u) \subseteq \mathbf{not}(U \setminus M)$ and so, $M^c = [P \cup \mathbf{not}(U \setminus Y^u)]^* \subseteq [P \cup \mathbf{not}(U \setminus M)]^*$.

As we already observed, M^c is closed under P. Moreover, $\mathbf{not}(U \setminus M) \subseteq M^c$. Thus, $[P \cup \mathbf{not}(U \setminus M)]^* \subseteq M^c$.

It follows that $M^c = [P \cup \mathbf{not}(U \setminus M)]^*$, which implies that $M = [P \cup \mathbf{not}(U \setminus M)]^* \cap U$. Thus, M is a stable model of P^+. We already proved that M is a model of P^- and so, M is an answer set of P. □

The following example illustrates our second approach.

Example 2. Let P be a UG-program consisting of rules:

$$\begin{aligned} \mathbf{not}(a) &\leftarrow \mathbf{not}(b) \\ b &\leftarrow \mathbf{not}(a) \\ a &\leftarrow \end{aligned}$$

Iterating the operator γ_P results in the following alternating sequence:

$$\emptyset \mapsto \{a, b, \mathbf{not}(a), \mathbf{not}(b)\} \mapsto \{a\} \mapsto \{a, b, \mathbf{not}(a), \mathbf{not}(b)\} \mapsto \ldots \ .$$

Its limit is $(\{a\}, \{a, b, \mathbf{not}(a), \mathbf{not}(b)\})$ and so, $Appx_2(P) = \{a\}$. $\square$

We conclude this section by showing that the approximations $Appx_1$ and $Appx_2$ are, in general, not comparable.

The following example shows that there is a UG-program P such that $Appx_1(P)$ and $Appx_2(P)$ are coherent and $Appx_2(P)$ is a *proper* subset of $Appx_1(P)$.

Example 3. Let $U = \{a, b, c, d, e\}$ and let P be a UG-program consisting of the rules:

$$\begin{aligned} a &\leftarrow \mathbf{not}(a) \\ b &\leftarrow \mathbf{not}(a) \\ c &\leftarrow \mathbf{not}(d) \\ d &\leftarrow \mathbf{not}(c), \mathbf{not}(e) \\ e &\leftarrow \\ a &\leftarrow c, e \\ \mathbf{not}(e) &\leftarrow a, b \end{aligned}$$

Computing $Appx_1(P)$. The program P^+ consists of all rules of P except the last one. The alternating sequence of $\gamma_{P^+,U}$ starts as follows:

$$\emptyset \mapsto \left\{a, b, c, d, e\right\} \mapsto \{e\} \mapsto \left\{a, b, c, e\right\} \mapsto \left\{a, c, e\right\} \mapsto \left\{a, c, e\right\} \mapsto \ldots \ .$$

Thus, its limit is $(\{a, c, e\}, \{a, c, e\})$ and

$$Appx_1(P) = [P \cup \{a, c, e\} \cup \{\mathbf{not}(b), \mathbf{not}(d)\}]^* = \{a, c, e, \mathbf{not}(b), \mathbf{not}(d)\}.$$

Computing $Appx_2(P)$. Iterating the operator γ_P yields the following sequence:

$$\emptyset \mapsto Lit(U) \mapsto \{e\} \mapsto Lit(U) \mapsto \ldots \ .$$

Thus, the limit is $(\{e\}, Lit(U))$ and so, $Appx_2(P) = \{e\}$. $\square$

The next example shows that for some programs the opposite is true and the second approximation is strictly more precise.

Example 4. Let $U = \{a, b, c\}$ and let P be a UG-program consisting of the rules:

$$\begin{aligned} a &\leftarrow \mathbf{not}(b) \\ b &\leftarrow \mathbf{not}(a) \\ c &\leftarrow a, b \\ \mathbf{not}(a) &\leftarrow \end{aligned}$$

Computing $Appx_1(P)$**.** The alternating sequence of the operator $\gamma_{P^+,U}$ is

$$\emptyset \mapsto \{a, b, c\} \mapsto \emptyset \mapsto \ldots \,.$$

Thus,

$$Appx_1(P) = P^* = \{\mathbf{not}(a), b\}.$$

Computing $Appx_2(P)$**.** Iterating γ_P yields:

$$\emptyset \mapsto Lit(U) \mapsto \{\mathbf{not}(a), b\} \mapsto \{\mathbf{not}(a), b, \mathbf{not}(c)\} \mapsto \{\mathbf{not}(a), b, \mathbf{not}(c)\} \mapsto \ldots \,.$$

Thus, $Appx_2(P) = \{\mathbf{not}(a), b, \mathbf{not}(c)\}$. □

4 Strong approximation

Let P be a UG-program and $Z \subseteq Lit(U)$ a set of literals (not necessarily *coherent*). By the *weak reduct* of P with respect to Z we mean the program P^Z_w obtained from P by:

1. removing all rules that contain in the body a literal $\mathbf{not}(a)$ such that $a \in Z$ and $\mathbf{not}(a) \notin Z$;
2. removing from the bodies of the remaining rules all literals $\mathbf{not}(a)$ such that $a \notin Z$.

Let us note that if $a \in Z$ and $\mathbf{not}(a) \in Z$, $\mathbf{not}(a)$ will not be removed from the rules that remain after Step 1.

Let Z be a set of literals, $Z \subseteq Lit(U)$. We define

$$\gamma^w_P(Z) = [P^Z_w]^*.$$

In general, the operator γ^w_P is not antimonotone. Thus, the sequence (Z_i) obtained by iterating γ^w_P (starting with the empty set) in general is not alternating.

Example 5. Let P be a UG-program consisting of the rules:

$$\begin{aligned} a &\leftarrow \\ \mathbf{not}(a) &\leftarrow b \\ b &\leftarrow \mathbf{not}(c) \\ c &\leftarrow c \\ d &\leftarrow \mathbf{not}(a). \end{aligned}$$

By the definition, $Z_0 = \emptyset$. When computing P^{Z_0}, no rule is removed in Step 1 of the definition and every literal of the form $\mathbf{not}(a)$ is removed form the bodies of rules in P. Thus,

$$P_w^{Z_0} = \left\{ \begin{array}{c} a \leftarrow \\ \mathbf{not}(a) \leftarrow b \\ b \leftarrow \\ c \leftarrow c \\ d \leftarrow . \end{array} \right\}, \quad \text{and so,} \;\; Z_1 = \{a, b, d\},$$

Since Z_1 is coherent, the rule $d \leftarrow \mathbf{not}(a)$ is removed in Step 1 when computing $P_w^{Z_1}$. Thus,

$$P_w^{Z_1} = \left\{ \begin{array}{c} a \leftarrow \\ \mathbf{not}(a) \leftarrow b \\ b \leftarrow \\ c \leftarrow c \end{array} \right\}, \quad \text{and so,} \;\; Z_2 = \{a, b, \mathbf{not}(a)\},$$

When computing $P_w^{Z_2}$, the rule $d \leftarrow \mathbf{not}(a)$ is *not* removed in Step 1. Thus,

$$P_w^{Z_2} = \left\{ \begin{array}{c} a \leftarrow \\ \mathbf{not}(a) \leftarrow b \\ b \leftarrow \\ c \leftarrow c \\ d \leftarrow \mathbf{not}(a). \end{array} \right\}, \quad \text{and so,} \;\; Z_3 = \{a, b, d, \mathbf{not}(a)\}.$$

We note that neither Z_2 nor Z_3 are subsets of Z_1. Thus, for this program P, the sequence (Z_i) is not alternating. $\square$

In the remainder of this section we show that under some conditions the sequence (Z_i) is alternating and may be used to approximate answer sets of UG-programs.

Lemma 1. *Let P be a UG-program, X and X' be sets of literals such that $X \subseteq X'$. Moreover, let at least one of the following conditions hold:*

1. *X' is coherent*
2. *$X \subseteq [P_w^{X'}]^*$ and $[P_w^{X'}]^*$ is coherent.*
3. *$[P_w^{X'}]^* \subseteq X$*
4. *$X \subseteq [P_w^{X}]^*$ and $[P_w^{X}]^*$ is coherent.*

Then $[P_w^{X'}]^* \subseteq [P_w^{X}]^*$.

Proof: Let Q consist of those rules in $P_w^{X'}$ whose bodies are contained in $[P_w^{X'}]^*$. Then, $[P_w^{X'}]^* = Q^*$. Let $r = \alpha \leftarrow \mathit{Body}$ be a rule in Q such that $\mathit{Body} \subseteq [P_w^{X}]^*$. To prove the assertion, it suffices to show that $\alpha \in [P_w^{X}]^*$. Indeed, the fact that r is arbitrary, implies that $[P_w^{X}]^*$ is closed under rules in Q and, consequently, that

$$[P_w^{X'}]^* = Q^* \subseteq [P_w^{X}]^*.$$

By the definition of the reduct $P_w^{X'}$, there is a rule $r' = \alpha \leftarrow \mathit{Body}'$ in P such that $\mathit{Body} \subseteq \mathit{Body}'$ and for every literal $\beta \in \mathit{Body}' \setminus \mathit{Body}$, $\beta = \mathbf{not}(b)$, for some $b \notin X'$.

Let $\mathbf{not}(a) \in Body'$. We will show that either $a \notin X$ or $\mathbf{not}(a) \in X$. First, if $a \notin X'$ then $a \notin X$. Thus, let us assume that $a \in X'$. Then, $\mathbf{not}(a) \in Body$ and so, $\mathbf{not}(a) \in X'$ (otherwise, r would not belong to $P_w^{X'}$). Since $\mathbf{not}(a) \in Body$, we also have that $\mathbf{not}(a) \in [P_w^{X'}]^*$ and $\mathbf{not}(a) \in [P_w^X]^*$.

Under the condition (1), since $\mathbf{not}(a) \in X'$ and X' is coherent, $a \notin X'$, a contradiction. Under the condition (2), since $\mathbf{not}(a) \in [P_w^{X'}]^*$, the coherence of $[P_w^{X'}]^*$ implies that $a \notin [P_w^{X'}]^*$ and, consequently, that $a \notin X$. If the condition (3) holds, $\mathbf{not}(a) \in [P_w^X]^*$ implies that $\mathbf{not}(a) \in X$. Finally, if the condition (4) holds, since $\mathbf{not}(a) \in [P_w^X]^*$, the coherence of $[P_w^X]^*$ implies that $a \notin [P_w^X]^*$. Thus, $a \notin X$.

Let $Body''$ be obtained from $Body'$ by removing all literals of the form $\mathbf{not}(a)$, where $a \notin X$. By the observation we proved above, the rule $r'' = \alpha \leftarrow Body''$ is in P_w^X. Since $X \subseteq X'$, $Body'' \subseteq Body$. Thus, $Body'' \subseteq [P_w^X]^*$ and it follows that $\alpha \in [P_w^X]^*$. □

Lemma 2. *Let P be a UG-program and X a coherent set of literals, $X \subseteq Lit(U)$.*

1. $[P_w^X]^* = [P_w^{X \cap U}]^*$.
2. $[P_w^X]^* = [(P^+)_w^{X \cap U}]^* \cup \mathbf{not}(X') = [(P^+)^{X \cap U}]^* \cup \mathbf{not}(X')$, *where X' is the set of atoms such that $a \in X'$ if and only if there is a rule $\mathbf{not}(a) \leftarrow Body$ in P^- such that $[(P^+)_w^{X \cap U}]^* \models Body$.*

Proof: (1) For every atom $a \in U$, $a \in X$ if and only $a \in X \cap U$. Moreover, since X is coherent, if $a \in X$, then $\mathbf{not}(a) \notin X$. Therefore, $P_w^X = P_w^{X \cap U}$ and so, $[P_w^X]^* = [P_w^{X \cap U}]^*$.
(2) We observe that $P_w^X = (P^+)_w^X \cup (P^-)_w^X$. Moreover, since X is coherent, the bodies of rules in P_w^X consist of atoms only. Thus, it follows that

$$[P_w^X]^* = [(P^+)_w^{X \cap U}]^* \cup \mathbf{not}(X').$$

The second equality in (2) follows from the observation that, since P^+ is a normal logic program, $(P^+)_w^{X \cap U}$ and the standard Gelfond-Lifschitz reduct $(P^+)^{X \cap U}$ coincide. □

We have the following characterization of answer sets of UG-programs.

Lemma 3. *Let P be a UG-program, $M \subseteq U$ a set of atoms, and N a set of atoms consisting of all atoms $a \in U$ such that $a \notin M$ and there is a rule $\mathbf{not}(a) \leftarrow Body$ in P such that $M \models Body$. Then M is an answer set of P if and only if $[P_w^M]^* = M \cup \mathbf{not}(N)$.*

Proof: (⇒) By Proposition 1, M is a stable model of P^+ and a model of P^-. In particular, $[(P^+)^M]^* = M$. Let X' be the set specified in Lemma 2(2), defined for $X = M$. Since $[(P^+)^M]^* = M$ and M is a model of P^-, for every $a \in X'$, $a \notin M$. Thus, $X' = N$ and the assertion follows from Lemma 2(2).
(⇐) It follows from Lemma 2(2) that $M = [(P^+)^M]^*$. Thus, M is stable model of P^+. Let us consider a rule $\mathbf{not}(a) \leftarrow Body$ from P^- such that M satisfies $Body$. Let $Body'$ consist of all atoms in $Body$. It follows that $\mathbf{not}(a) \leftarrow Body'$ is a rule in P_w^M. Since M satisfies $Body$, $Body' \subseteq M = [(P^+)^M]^* = [(P^+)_w^M]^* \subseteq [P_w^M]^*$. Thus, $\mathbf{not}(a) \in [P_w^M]^*$ and, consequently, $a \notin M$. It follows that M is a model of P^- and so, an answer set of P. □

Lemma 4. *Let i be an integer such that $i \geq 1$ and Z_{2i} is coherent.*

1. *If $Z_{2i-2} \subseteq Z_{2i-1}$ and $Z_{2i-2} \subseteq Z_{2i}$, then $Z_{2i} \subseteq Z_{2i-1}$*
2. *If $Z_{2i-2} \subseteq Z_{2i}$, then $Z_{2i+1} \subseteq Z_{2i-1}$*
3. *If $Z_{2i} \subseteq Z_{2i-1}$, then $Z_{2i} \subseteq Z_{2i+1}$*
4. *If $Z_{2i+1} \subseteq Z_{2i-1}$ and $Z_{2i} \subseteq Z_{2i+1}$, then $Z_{2i} \subseteq Z_{2i+2}$.*

Proof: (1) Let us note that $Z_{2i} = [P_w^{Z_{2i-1}}]^*$. Thus, (1) follows from Lemma 1 applied to $X = Z_{2i-2}$ and $X' = Z_{2i-1}$, which satisfy the assumption (2) of the lemma.
(2) Let us assume that $Z_{2i-2} \subseteq Z_{2i}$. Since Z_{2i} is coherent, Lemma 1 applies (under the condition (1)) and implies that $[P_w^{Z_{2i}}]^* \subseteq [P_w^{Z_{2i-2}}]^*$. Consequently

$$Z_{2i+1} = [P_w^{Z_{2i}}]^* \subseteq [P_w^{Z_{2i-2}}]^* = Z_{2i-1}.$$

(3) Since $Z_{2i} = [P_w^{Z_{2i-1}}]^*$, $X = Z_{2i}$ and $X' = Z_{2i-1}$ satisfy the assumptions of Lemma 1 (in particular, the assumption (3)). Thus,

$$Z_{2i} = [P_w^{Z_{2i-1}}]^* \subseteq [P_w^{Z_{2i}}]^* = Z_{2i+1}.$$

(4) One can check that the assumptions of Lemma 1 are satisfied with $X = Z_{2i+1}$ and $X' = Z_{2i-1}$ (again, the assumption (3)). □

Corollary 3. *Let i be an integer, $i \geq 0$, such that Z_{2i} is coherent. Then*

1. $Z_0 \subseteq Z_2 \subseteq \ldots \subseteq Z_{2i}$
2. $Z_1 \supseteq Z_3 \supseteq \ldots \supseteq Z_{2i+1}$
3. $Z_{2i} \subseteq Z_{2i+1}$.

Proof: We have $Z_0 = \emptyset$. Thus, $Z_0 \subseteq Z_1$ and $Z_0 \subseteq Z_2$. Consequently, the assertion follows by induction from Lemma 4. □

Let us consider the sequence (Z_i). If for every i, Z_{2i} is coherent, Corollary 3 implies that the sequence (Z_i) is alternating. Let (Z^l, Z^u) be the limit of (Z_i). We define

$$Appx_3(P) = Z^l \cup \{\mathbf{not}(a)\colon a \in U \setminus Z^u\}.$$

Otherwise, there is i such that Z_{2i} is incoherent. In this case, we say that $Appx_3(P)$ is undefined.

Theorem 4. *Let P be a UG-program. If M is an answer set of P then $Appx_3(P)$ is defined and $Appx_3(P) \subseteq M^c$. If $Appx_3(P)$ is not defined, then P has no answer sets.*

Proof: The second part of the assertion follows from the first one. To prove the first part of the assertion, we will show that for every $i \geq 0$, $Z_{2i} \subseteq M^c$, and $M \subseteq Z_{2i+1}$.

We proceed by induction on i. If $i = 0$, then $Z_0 = \emptyset \subseteq M^c$. We now assume that $Z_{2i} \subseteq M^c$ and prove that $M \subseteq Z_{2i+1}$.

Since $Z_{2i} \subseteq M^c$ and M^c is coherent, Z_{2i} is coherent, too. By Lemma 1 (applied to $X = Z_{2i}$ and $X' = M^c$, under the assumption (4)), $[P_w^{M^c}]^* \subseteq [P_w^{Z_{2i}}]^*$. Thus, $[P_w^{M^c}]^* \subseteq Z_{2i+1}$. By Lemma 2(1), $[P_w^M]^* \subseteq Z_{2i+1}$. By Lemma 3, $M \subseteq [P_w^M]^*$. Therefore, $M \subseteq Z_{2i+1}$.

Next, we assume that $M \subseteq Z_{2i+1}$ and prove that $Z_{2i+2} \subseteq M^c$. Let us note that $Z_{2i+2} = [P_w^{Z_{2i+1}}]^*$ and that by Lemma 3, $[P_w^M]^* \subseteq M^c$. Thus, it will suffice to show that $[P_w^{Z_{2i+1}}]^* \subseteq [P_w^M]^*$. To this end, we note that by Lemma 3, $M \subseteq [P_w^M]^*$ and so Lemma 1 applies (under the condition (4)) to $X = M$ and $X' = Z_{2i+1}$, and implies the required inclusion.

It follows that $Z^l \subseteq M^c$ and that $M \subseteq Z^u$. If $a \notin Z^u$, then $a \notin M$ and so, $\mathbf{not}(a) \in M^c$. Thus, $Appx_3(P) = Z^l \cup \mathbf{not}(U \setminus Z^u) \subseteq M^c$. $\square$

Example 6. Let P be a UG-program consisting of the rules:

$$
\begin{aligned}
\mathbf{not}(a) &\leftarrow \\
a &\leftarrow \mathbf{not}(b) \\
b &\leftarrow \mathbf{not}(a) \\
c &\leftarrow a, b \\
\mathbf{not}(d) &\leftarrow \mathbf{not}(c) \\
d &\leftarrow \mathbf{not}(e) \\
e &\leftarrow \mathbf{not}(d) \\
f &\leftarrow d, e
\end{aligned}
$$

Iterating the operator γ_P^w results in the following sequence:

$$
\begin{aligned}
&\emptyset \mapsto \{a, b, c, d, e, f, \mathbf{not}(a), \mathbf{not}(d)\} \mapsto \{\mathbf{not}(a), b\} \mapsto \{b, d, e, f, \mathbf{not}(a), \mathbf{not}(d)\} \\
&\mapsto \{b, e, \mathbf{not}(a), \mathbf{not}(d)\} \mapsto \{b, e, \mathbf{not}(a), \mathbf{not}(d)\} \mapsto \ldots .
\end{aligned}
$$

Thus, the sequence (Z_i) is alternating. Its limit is (Z^l, Z^u), where $Z^l = Z^u = \{b, e, \mathbf{not}(a), \mathbf{not}(d)\}$. Thus,

$$Appx_3(P) = Z^l \cup \mathbf{not}(U \setminus Z^u) = \{b, e, \mathbf{not}(a), \mathbf{not}(c), \mathbf{not}(d), \mathbf{not}(f)\}.$$

Since $Appx_3(P)$ is coherent and complete, P has a unique answer set, $\{b, e\}$. This example also demonstrates that Z^u can improve on the bound provided by Z^l itself. $\square$

5 Properties of $Appx_3$

In this section we will show that if $Appx_3$ is defined then it is stronger than the other two approximations. We recall that if $Appx_3(P)$ is undefined, then P has no answer sets, that is, P is *inconsistent*. It follows that for all *consistent* UG-programs, $Appx_3$ is stronger than the the other two approximations.

Theorem 5. *Let$P$ be a UG-program. If $Appx_3(P)$ is defined then*

$$Appx_1(P) \cup Appx_2(P) \subseteq Appx_3(P)$$

Proof: Let (W_i) be the alternating sequence of the operator $\gamma_{P^+,U}$ and let (Z_i) be the alternating sequence of the operator γ_P^w. We observe that all sets W_i consist of atoms. Also, since $Appx_3(P)$ is defined, all sets Z_{2i} are coherent.

We will show that for every $i \geq 0$, $W_{2i} \subseteq Z_{2i}$ and $Z_{2i+1} \cap U \subseteq W_{2i+1}$.

We proceed by induction. The basis is evident as $W_0 = Z_0 = \emptyset$. We will now assume that $W_{2i} \subseteq Z_{2i}$ and prove that $Z_{2i+1} \cap U \subseteq W_{2i+1}$. We have

$$[P_w^{Z_{2i}}]^* = [(P^+)_w^{Z_{2i}} \cup (P^-)_w^{Z_{2i}}]^*.$$

Since Z_{2i} is coherent, no rule in $(P^+)_w^{Z_{2i}}$ contains a negated literal. Thus, there is a set $N \subseteq U$ such that

$$[P_w^{Z_{2i}}]* = [(P^+)_w^{Z_{2i}}]^* \cup \mathbf{not}(N).$$

It follows that $Z_{2i+1} \cap U = [(P^+)_w^{Z_{2i}}]^*$. Since Z_{2i} is coherent, $(P^+)_w^{Z_{2i}} = (P^+)_w^Z = (P^+)^Z$, where $Z = U \cap Z_{2i}$. We have $W_{2i} \subseteq U$ and $W_{2i} \subseteq Z_{2i}$. Thus, $W_{2i} \subseteq Z$ and $(P^+)^Z \subseteq (P^+)^{W_{2i}}$. Putting all these facts together, we obtain

$$Z_{2i+1} \cap U = [(P^+)_w^{Z_{2i}}]^* = [(P^+)^Z]^* \subseteq [(P^+)^{W_{2i}}]^* = W_{2i+1}.$$

Next, we assume that $Z_{2i+1} \cap U \subseteq W_{2i+1}$ and show that $W_{2i+2} \subseteq Z_{2i+2}$. Since $Z_{2i+1} \cap U \subseteq W_{2i+1}$ and since W_{2i+1} consists of atoms only, $(P^+)^{W_{2i+1}} \subseteq (P^+)_w^{Z_{2i+1}}$. Thus, $(P^+)^{W_{2i+1}} \subseteq P_w^{Z_{2i+1}}$ and so,

$$W_{2i+2} = [(P^+)^{W_{2i+1}}]^* \subseteq [P_w^{Z_{2i+1}}]^* = Z_{2i+2}.$$

Let (W^l, W^u) and (Z^l, Z^u) be the limits of the sequences (W_i) and (Z_i), respectively. The claim we proved above implies that $Z^u \cap U \subseteq W^u$. Thus,

$$\mathbf{not}(U \setminus W^u) \subseteq \mathbf{not}(U \setminus Z^u) \subseteq Appx_3(P).$$

Let $r = \alpha \leftarrow Body$ be a rule in P and let us assume that $Body \subseteq Appx_3(P)$. Let $\mathbf{not}(a)$ be a negated literal in $Body$. Then $\mathbf{not}(a) \in Appx_3(P)$ and so, $\mathbf{not}(a) \in Z^l$ or $a \notin Z^u$. Since $Z^l \subseteq Z^u$, r is not removed in Step 1 of the definition of the reduct $P_w^{Z^u}$. Let $r' = \alpha \leftarrow Body'$ be the rule obtained by removing from the body of r all literals $\mathbf{not}(a)$ such that $a \notin Z^u$. Then, $r' \in P_w^{Z^u}$ and $Body' \subseteq Z^l$. Since $Z^l = [P_w^{Z^u}]^*$, $\alpha \in Z^l$ and so, $\alpha \in Appx_3(P)$.

Thus, $Appx_3(P)$ is closed under rules in P and contains $\mathbf{not}(U \setminus W^u)$. It follows that

$$Appx_1(P) = [P \cup \mathbf{not}(U \setminus S)]^* \subseteq Appx_3(P).$$

Next, we will show that $Appx_2(P) \subseteq Appx_3(P)$. Let (Y_i) be the alternating sequence of γ_P. We will show that for every $i \geq 0$, $Y_{2i} \cap U \subseteq Z_{2i}$ and $Z_{2i+1} \subseteq Y_{2i+1}$.

We proceed by induction. The basis is evident as $Y_0 = Z_0 = \emptyset$. We now assume that $Y_{2i} \cap U \subseteq Z_{2i}$ and prove that $Z_{2i+1} \subseteq Y_{2i+1}$. Since Z_{2i} is coherent, Lemmas 1 and 2 imply that

$$Z_{2i+1} = [P_w^{Z_{2i}}]^* = [P_w^{Z_{2i} \cap U}]^* \subseteq [P_w^{Y_{2i} \cap U}]^* = [P_w^{Y_{2i}}]^*.$$

Let $\alpha \leftarrow Body$ be a rule in $P_w^{Y_{2i}}$ such that $Body \subseteq [P \cup \mathbf{not}(U \setminus Y_{2i})]^*$. Then, there is a rule $r' = \alpha \leftarrow Body'$ in P such that for every literal $\mathbf{not}(a)$ in $Body'$, $a \notin Y_{2i}$ or $\mathbf{not}(a) \in Y_{2i}$, and removing all literals $\mathbf{not}(a)$, where $a \notin Y_{2i}$, from $Body'$ results in

$Body$. It follows that $Body' \subseteq [P \cup \mathbf{not}(U \setminus Y_{2i})]^*$ and so, $\alpha \in [P \cup \mathbf{not}(U \setminus Y_{2i})]^*$. Thus, $[P \cup \mathbf{not}(U \setminus Y_{2i})]^*$ is closed under rules in $[P_w^{Y_{2i}}]^*$ and so,

$$[P_w^{Y_{2i}}]^* \subseteq [P \cup \mathbf{not}(U \setminus Y_{2i})]^* = Y_{2i+1}.$$

Consequently, $Z_{2i+1} \subseteq Y_{2i+1}$ follows.

Next, we assume that $Z_{2i+1} \subseteq Y_{2i+1}$ and prove that $Y_{2i+2} \cap U \subseteq Z_{2i+2}$.

Let P' be a program obtained from P by removing literals in $\mathbf{not}(U \setminus Y_{2i+1})$ from the bodies of rules in P. The program P' has the following property:

$$[P \cup \mathbf{not}(U \setminus Y_{2i+1})]^* \cap U \subseteq [P']^*.$$

Let $r = \alpha \leftarrow Body$ be a rule in P' such that $Body \in [P_w^{Z_{2i+1}}]^*$. Since (Z_i) is alternating (we recall that $Appx_3(P)$ is defined), $Z_{2i+2} \subseteq Z_{2i+1}$. Thus,

$$Body \subseteq [P_w^{Z_{2i+1}}]^* = Z_{2i+2} \subseteq Z_{2i+1}.$$

Since $r \in P'$, there is a rule $r' = \alpha \leftarrow Body'$ in P such that after removing from $Body'$ all literals of the form $\mathbf{not}(a)$, where $a \notin Y_{2i+1}$, we obtain $Body$. Let $\mathbf{not}(a)$ be a literal of $Body'$. If $a \in Z_{2i+1}$, then $a \in Y_{2i+1}$ and so, $\mathbf{not}(a) \in Body$. Thus, $\mathbf{not}(a) \in Z_{2i+1}$ and the rule r' is not removed in Step 1 of the construction of $P_w^{Z_{2i+1}}$.

Let $Body''$ be the result of removing from $Body'$ all literals $\mathbf{not}(a)$, where $a \notin Z_{2i+1}$. It follows that $r'' = \alpha \leftarrow Body''$ is in $P_w^{Y_{Z_{i+1}}}$. Let β be a literal in $Body''$. If β is an atom then, since $\beta \in Body'$ and only negated atoms are removed from $Body'$ when constructing r from r', $\beta \in Body$. If $\beta = \mathbf{not}(a)$, then $a \in Z_{2i+1}$ and, by an argument given above, $\mathbf{not}(a) \in Body$. Thus, in either case $\beta \in Body$ and so, $\beta \in [P_w^{Z_{2i+1}}]^*$, as well. It follows that $Body'' \subseteq [P_w^{Z_{2i+1}}]^*$. Consequently, $\alpha \in [P_w^{Z_{2i+1}}]^*$. Thus, $[P_w^{Z_{2i+1}}]^*$ is closed under P' and $[P']^* \subseteq [P_w^{Z_{2i+1}}]^*$. In this way, we get

$$Y_{2i+2} \cap U = [P \cup \mathbf{not}(U \setminus Y_{2i+1})]^* \cap U \subseteq [P']^* \subseteq [P_w^{Z_{2i+1}}]^* = Z_{2i+2}.$$

The claim we just proved implies that $Y^l \cap U \subseteq Z^l \cap U$ and $Z^u \subseteq Y^u$. Thus,

$$Appx_2(P) = Y^l \subseteq Z^l \cup \mathbf{not}(U \setminus Z^u) = Appx_3(P).$$

□

There are programs which show that$Appx_3$ is strictly stronger.

Example 7. Let P be the UG-program from Example 4. We recall that $Appx_1(P) = \{\mathbf{not}(a), b\}$. Let us compute $Appx_3(P)$. By iterating the operator γ_w^P, we obtain the following sequence:

$$Z_0 = \emptyset \mapsto Z_1 = \{a, b, c, \mathbf{not}(a)\} \mapsto Z_2 = \{\mathbf{not}(a), b\} \mapsto Z_3 = \{\mathbf{not}(a), b\} \ldots.$$

Hence, $Appx_3(P) = \{\mathbf{not}(a), b, \mathbf{not}(c)\}$ and $Appx_1(P)$ is a *proper* subset of $Appx_3(P)$. □

Example 8. Let P be the UG-program from Example 3. We recall that $Appx_2(P) = \{e\}$. To compute $Appx_3(P)$, we note that by iterating the operator γ_w^P we get the following sequence:

$$Z_0 = \emptyset \mapsto Z_1 = \{a, b, c, d, e, \mathbf{not}(e)\} \mapsto Z_2 = \{e\} \mapsto$$

$$Z_3 = \{a, b, c, e, \mathbf{not}(e)\} \mapsto Z_4 = \{a, c, e\} \mapsto Z_5 = \{a, c, e\} \ldots.$$

Hence, $Appx_3(P) = \{a, \mathbf{not}(b), c, \mathbf{not}(d), e\}$ and $Appx_2(P)$ is a *proper* subset of $Appx_3(P)$. $\Box$

Finally, we show that if $Appx_3(P)$ is defined and complete then P has a unique answer set.

Corollary 4. *Let P be a UG-program such that $Appx_3(P)$ is defined and complete. Then $Appx_3(P) \cap U$ is an answer set of P and P has no other answer sets.*

Proof: Let $M = Appx_3(P) \cap U$. By Theorem 4, it suffices to show that M is an answer set of P. To this end, we will show that M is a stable model of P^+ and a model of P^-.

We first observe that the definition of $Appx_3(P)$ implies that $M = Z^l \cap U$. We also note that since Z^l is coherent, $Z^u = [P_w^{Z^l}]^* = [(P^+)^M]^* \cup \mathbf{not}(N)$, for some set $N \subseteq U$. It follows that $M = Z^l \cap U \subseteq Z^u \cap U = [(P^+)^M]^*$. We will now show that $[(P^+)^M]^* \subseteq [P_w^{Z^u}]^*$.

Let $r = a \leftarrow Body$ be a rule in $(P^+)^M$. Let us assume that $Body \subseteq [P_w^{Z^u}]^*$. Let $r' = a \leftarrow Body'$ be a rule in P such that for every literal $\alpha \in Body' \setminus Body$, $\alpha = \mathbf{not}(a)$, for some $a \notin M$. Let $\mathbf{not}(a)$ be a literal in $Body'$ such that $a \in Z^u$. Then, $a \notin Z^l$. Thus, $\mathbf{not}(a) \in Z^l$ (by the completeness of Z^l). In particular, $\mathbf{not}(a) \in Z^u$ and so, r' is not removed in Step 1 of the construction of $P_w^{Z^u}$. Let $r'' = a \leftarrow Body''$ be the rule obtained from r' by removing all literals $\mathbf{not}(a)$ such that $a \notin Z^u$. It follows that $r'' \in P_w^{Z^u}$. For every atom $a \in Body''$, $a \in Body$ and so, $a \in [P_w^{Z^u}]^*$. For every literal $\mathbf{not}(a) \in Body''$, $a \in Z^u$ (otherwise, this literal would get removed). Thus, according to an earlier observation, $\mathbf{not}(a) \in Z^l = [P_w^{Z^u}]^*$. It follows that $Body'' \subseteq [P_w^{Z^u}]^*$. Thus, $a \in [P_w^{Z^u}]^*$ and $[P_w^{Z^u}]^*$ is closed under $(P^+)^M$. Thus, $[(P^+)^M]^* \subseteq [P_w^{Z^u}]^*$, as required.

Thus, $M = [(P^+)^M]^*$, that is, M is a stable model of P^+, and $Z^l \cap U = Z^u \cap U$. The latter identity implies $P_w^{Z_l} \subseteq P_w^{Z^u}$ and so, $Z^u = Z^l$. Let $r = \mathbf{not}(a) \leftarrow Body$ be a rule in P^- such that $M \models Body$. It follows that r is not removed in Step 1 of the construction of $P_w^{Z^u}$ and that $P_w^{Z^u}$ contains a rule $r' = a \leftarrow Body'$, where $Body' \subseteq Body$. It follows that $M \models Body'$. Moreover, since $Body'$ consists of atoms (Z^u is coherent), $Body' \subseteq M = Z^l \cap U \subseteq Z^l = [P_w^{Z^u}]^*$. Thus, $\mathbf{not}(a) \in [P_w^{Z^u}]^*$ and so, $\mathbf{not}(a) \in Z^l$. Consequently, $a \notin M$ and it follows that M is a model of P^-. Thus, M is an answer set of P. $\Box$

6 Corollaries for the case of revision programs.

Revision programming [MT98] is a formalism for describing and enforcing constraints on databases. The main concepts in the formalism are initial database, revision program, and justified revisions. Expressions of the form **in**(a) and **out**(a) $(a \in U)$ are called revision literals. Intuitively, **in**(a) (resp., **out**(a)) means that atom a is in (resp., is not in) a database. Revision program consists of rules of the type $\alpha \leftarrow \alpha_1, \ldots, \alpha_n$, where α, $\alpha_i, \ldots, \alpha_n$ are revision literals. Given a revision program P and an initial database I, P-justified revisions of I represent a set of revisions. Each revision satisfies the constraints and differs minimally from the initial database.

As we mentioned earlier, unitary general programs are equivalent to revision programs [MPT99].

The first two approximations, $Appx_1$ and $Appx_2$, directly correspond to two definitions of well-founded semantics for revision programs induced by embeddings of revision programs into logic programs proposed in [Piv01]. It was also shown that the definitions were in general not comparable.

Formal descriptions of $Appx_1$, $Appx_2$, and $Appx_3$ for revision programs can be found in [Piv05].

Theorem 5 implies that $Appx_3$ is stronger than $Appx_1$ and $Appx_2$ for revision programming.

Theorem 4 implies that if $Appx_3$ does not exist for a revision program P and initial database I, then there are no P-justified revisions of I.

Corollary 4 implies that if $Appx_3$ is defined and complete for a revision program P and a database I, then there exists exactly one P-justified revision of I and it is determined by the approximation.

Acknowledgements. Inna Pivkina was supported by the National Science Foundation-funded ADVANCE Institutional Transformation Program at New Mexico State University, fund # NSF0123690 and NMSU College of Arts and Sciences Research Center grant number 01-3-43891.

References

[BTK93] A. Bondarenko, F. Toni, and R.A. Kowalski. An assumption-based framework for non-monotonic reasoning. In A. Nerode and L. Pereira, editors, *Logic programming and non-monotonic reasoning (Lisbon, 1993)*, pages 171–189, Cambridge, MA, 1993. MIT Press.

[GL88] M. Gelfond and V. Lifschitz. The stable semantics for logic programs. In *Proceedings of the 5th International Conference on Logic Programming*, pages 1070–1080. MIT Press, 1988.

[GL91] M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9:365–385, 1991.

[Lif96] V. Lifschitz. Foundations of logic programming. In *Principles of Knowledge Representation*, pages 69–127. CSLI Publications, 1996.

[LW92] V. Lifschitz and T.Y.C. Woo. Answer sets in general nonmonotonic reasoning. In *Proceedings of the 3rd international conference on principles of knowledge representation and reasoning, KR '92*, pages 603–614, San Mateo, CA, 1992. Morgan Kaufmann.

[MPT99] V. W. Marek, I. Pivkina, and M. Truszczyński. Revision programming = logic programming + integrity constraints. In G. Gottlob, E. Grandjean, and K. Seyr, editors, *Computer Science Logic, 12th International Workshop, CSL'98*, volume 1584 of *Lecture Notes in Computer Science*, pages 73–89. Springer, 1999.

[MPT02] V. W. Marek, I. Pivkina, and M. Truszczyński. Annotated revision programs. *Artificial Intelligence Journal*, 138:149–180, 2002.

[MT98] W. Marek and M. Truszczyński. Revision programming. *Theoretical Computer Science*, 190(2):241–277, 1998.

[Piv01] I. Pivkina. Revision programming: a knowledge representation formalism. PhD dissertation, University of Kentucky, 2001.

[Piv05] I. Pivkina. Defining well-founded semantics for revision programming Technical Report NMSU-CS-2005-001, New Mexico State University, Computer Science Department, 2005.

[PT95] T.C. Przymusiński and H. Turner. Update by means of inference rules. In *Logic programming and nonmonotonic reasoning (Lexington, KY, 1995)*, volume 928 of *Lecture Notes in Computer Science*, pages 156–174, Berlin, 1995. Springer.

[SNS02] P. Simons, I. Niemelä, and T. Soininen. Extending and implementing the stable model semantics. *Artificial Intelligence*, 138:181–234, 2002.

[SNV95] V.S. Subrahmanian, D. Nau, and C. Vago. WFS + branch bound = stable models. *IEEE Transactions on Knowledge and Data Engineering*, 7:362–377, 1995.

[Van93] A. Van Gelder. The alternating fixpoint of logic programs with negation. *Journal of Computer and System Sciences*, 47(1):185–221, 1993.

[VRS88] A. Van Gelder, K.A. Ross, and J.S. Schlipf. Unfounded sets and well-founded semantics for general logic programs. In *ACM Symposium on Principles of Database Systems*, pages 221–230, 1988.