A semantic framework for preference handling in answer set programming

Torsten Schaub∗
Institut für Informatik, Universität Potsdam
Postfach 60 15 53, D–14415 Potsdam, Germany
torsten@cs.uni-potsdam.de

Kewen Wang†
School of Computing and Information Technology
Griffith University, Brisbane 4111, Australia
K.Wang@cit.gu.edu.au

Abstract
We provide a semantic framework for preference handling in answer set programming. To this end, we introduce preference preserving consequence operators. The resulting fixpoint characterizations provide us with a uniform semantic framework for characterizing preference handling in existing approaches. Although our approach is extensible to other semantics by means of an alternating fixpoint theory, we focus here on the elaboration of preferences under answer set semantics. Alternatively, we show how these approaches can be characterized by the concept of order preservation. These uniform semantic characterizations provide us with new insights about interrelationships and moreover about ways of implementation.

1 Introduction
Preferences constitute a very natural and effective way of resolving indeterminate situations. For example, in scheduling not all deadlines may be simultaneously satisfiable, and in configuration various goals may not be simultaneously met. In legal reasoning, laws may apply in different situations, but laws may also conflict with each other. In fact, while logical preference handling constitutes already an indispensable means for legal reasoning systems (cf. Gordon 1993; Prakken 1997), it is also advancing in other application areas such as intelligent agents and e-commerce (Grosof 1999) and the resolution of grammatical ambiguities (Cui and Swift 2002). The growing interest in preferences is also reflected by the large number of proposals in logic programming (Sakama and Inoue 1996; Brewka 1996; Gelfond and Son 1997; Zhang and Foo 1997; Grosof 1997; Brewka and Eiter 1999; Delgrande et al. 2000b; Wang et al. 2000). A common approach is to employ meta-formalisms for characterizing “preferred answer sets”. This has led to a diversity of approaches that are hardly comparable due to considerably different ways of formal characterization. Hence, there is no homogeneous account of preference.

∗ Affiliated with the School of Computing Science at Simon Fraser University, Burnaby, Canada.
† This work was done while the second author was with the University of Potsdam.
We address this shortcoming by proposing a uniform semantical framework for extended logic programming with preferences. To be precise, we develop an (alternating) fixpoint theory for so-called ordered logic programs (also, prioritized logic programs). An ordered logic program is an extended logic program whose rules are subject to a strict partial order. In analogy to standard logic programming, such a program is then interpreted by means of an associated fixpoint operator. We start by elaborating upon a specific approach to preference handling that avoids some problems of related approaches. We also show how the approaches of Brewka and Eiter (2000) and Delgrande et al. (2000b) can be captured within our framework. As a result, we obtain that the investigated approaches yield an increasing number of answer sets depending on how “tight” they integrate preferences. For obtaining a complementary perspective, we also provide characterizations in terms of the property of order preservation, originally defined in (Delgrande et al. 2000b) for distinguishing “preferred” from “non-preferred” answer sets. Moreover, we show how these approaches can be implemented by the compilation techniques developed in (Delgrande et al. 2000b). As well, we show that all these different preferred answer set semantics correspond to the perfect model semantics on stratified programs. We deal with approaches whose preferred answer sets semantics amounts to a selection function on the standard answer sets of an ordered logic program. In view of our interest in compiling these approaches into ordinary logic programs, we moreover limit our investigation to those guaranteeing polynomial translations. This excludes approaches like the ones in (Rintanen 1995; Zhang and Foo 1997) that step outside the complexity class of the underlying logic programming framework. This applies also to the approach in (Sakama and Inoue 1996), where preferences on literals are investigated. While the approach of (Gelfond and Son 1997) remains within NP, it advocates strategies that are non-selective (as discussed in Section 5). Approaches that can be addressed within this framework include those in (Baader and Hollunder 1993; Brewka 1994) that were originally proposed for default logic.

The paper is organized as follows. Once Section 2 has provided formal preliminaries, we begin in Section 3 by elaborating upon our initial semantics for ordered logic programs. Afterwards, we show in Section 4 how this semantics has to be modified in order to account for the two other aforementioned approaches.

2 Definitions and notation

We assume a basic familiarity with alternative semantics of logic programming (Lifschitz 1996). An extended logic program is a finite set of rules of the form

\[ L_0 \leftarrow L_1, \ldots, L_m, \text{not } L_{m+1}, \ldots, \text{not } L_n, \]

where \( n \geq m \geq 0 \), and each \( L_i (0 \leq i \leq n) \) is a literal, i.e. either an atom \( A \) or the negation \( \neg A \) of \( A \). The set of all literals is denoted by \( \text{Lit} \). Given a rule \( r \) as in (1), we let \( \text{head}(r) \) denote the head, \( L_0 \), of \( r \) and \( \text{body}(r) \) the body, \( \{L_1, \ldots, L_m, \text{not } L_{m+1}, \ldots, \text{not } L_n\} \), of \( r \). Further, let \( \text{body}^+(r) = \{L_1, \ldots, L_m\} \) and \( \text{body}^-(r) = \{L_{m+1}, \ldots, L_n\} \). A program is called basic if \( \text{body}^-(r) = \emptyset \) for all its rules; it is called normal if it contains no classical negation symbol \( \neg \). The reduct of a rule \( r \) is defined as \( r^+ = \text{head}(r) \leftarrow \text{body}^+(r) \); the reduct, \( \Pi^X \), of a program \( \Pi \) relative to a set \( X \) of literals is defined by

\[ \Pi^X = \{r^+ \mid r \in \Pi \text{ and } \text{body}^-(r) \cap X = \emptyset\}. \]
A semantic framework for preference handling in answer set programming

A set of literals \( X \) is closed under a basic program \( \Pi \) iff for any \( r \in \Pi \), \( head(r) \in X \) whenever \( body^+(r) \subseteq X \). We say that \( X \) is logically closed iff it is either consistent (ie. it does not contain both a literal \( A \) and its negation \( \neg A \)) or equals \( Lit \). The smallest set of literals which is both logically closed and closed under a basic program \( \Pi \) is denoted by \( Cn(\Pi) \). With these formalities at hand, we can define answer set semantics for extended logic programs: A set \( X \) of literals is an answer set of a program \( \Pi \) iff \( Cn(\Pi^X) = X \). For the rest of this paper, we concentrate on consistent answer sets. For capturing other semantics, \( Cn(\Pi^X) \) is sometimes regarded as an operator \( Cn(\Pi) \). The anti-monotonicity of \( Cn(\Pi) \) implies that \( Cn(\Pi) \) is monotonic. As shown in (van Gelder 1993), different semantics are obtained by appeal to immediate consequence operators \( \text{Lloyd} \). Let \( \Pi \) be a basic program and \( X \) a set of literals. The immediate consequence operator \( \text{T}_\Pi \) is defined as follows:

\[
\text{T}_\Pi X = \{ head(r) \mid r \in \Pi \text{ and } body(r) \subseteq X \}
\]

if \( X \) is consistent, and \( \text{T}_\Pi X = \text{Lit} \) otherwise. Iterated applications of \( \text{T}_\Pi \) are written as \( \text{T}^j_\Pi \) for \( j \geq 0 \), where \( \text{T}^0_\Pi X = X \) and \( \text{T}^i_\Pi X = \text{T}_\Pi \text{T}^{i-1}_\Pi X \) for \( i \geq 1 \). It is well-known that \( Cn(\Pi) = \bigcup_{i \geq 0} \text{T}^i_\Pi \emptyset \), for any basic program \( \Pi \). Also, for any answer set \( X \) of program \( \Pi \), it holds that \( \bar{X} = \bigcup_{i \geq 0} \text{T}^i_\Pi X \emptyset \). A reduction from extended to basic programs is avoidable with an extended operator: Let \( \Pi \Pi \) be an extended program and \( X \) \( Y \) be sets of literals. The extended immediate consequence operator \( \text{T}_{\Pi,Y} \) is defined as follows:

\[
\text{T}_{\Pi,Y} X = \{ head(r) \mid r \in \Pi \text{ and } body^+(r) \subseteq X \text{, and } body^-(r) \cap Y = \emptyset \}
\]

if \( X \) is consistent, and \( \text{T}_{\Pi,Y} X = \text{Lit} \) otherwise. Iterated applications of \( \text{T}_{\Pi,Y} \) are written as those of \( \text{T}_{\Pi} \). Clearly, we have \( \text{T}_{\Pi,\emptyset} X = \text{T}_\Pi X \) for any basic program \( \Pi \) and \( \text{T}_{\Pi,Y} X = \text{T}_\Pi \Pi \text{ for any extended program } \Pi \). Accordingly, we have for any answer set \( X \) of program \( \Pi \) that \( X = \bigcup_{i \geq 0} \text{T}^i_{\Pi,Y} \emptyset \). Finally, for dealing with the individual rules in (2), we rely on the notion of activeness: Let \( X, Y \subseteq \text{Lit} \) be two sets of literals in a program \( \Pi \). A rule \( r \) in \( \Pi \) is active wrt the pair \((X, Y)\), if \( body^+(r) \subseteq X \) and \( body^-(r) \cap Y = \emptyset \). Alternatively, we thus have that \( \text{T}_{\Pi,Y} X = \{ head(r) \mid r \in \Pi \text{ is active wrt } (X, Y) \} \).

Lastly, an ordered logic program is simply a pair \((\Pi, <)\), where \( \Pi \) is an extended logic program and \( < \subseteq \Pi \times \Pi \) is an irreflexive and transitive relation. Given, \( r_1, r_2 \in \Pi \), the relation \( r_1 < r_2 \) is meant to express that \( r_2 \) has higher priority than \( r_1 \). Programs associated with such an external ordering are also referred to as statically ordered programs, as opposed to dynamically ordered programs whose order relation is expressed through a special-purpose predicate within the program.

### 3 Preferred fixpoints

We elaborate upon a semantics for ordered logic program that allows us to distinguish the “preferred” answer sets of a program \((\Pi, <)\) by means of fixpoint equations. That is, a set of literals \( X \) is a preferred answer set of \((\Pi, <)\), if it satisfies the equation \( C(\Pi, <)(X) = X \) for some operator \( C(\Pi, <) \). In view of the classical approach described above, this makes us investigate semantics that interpret preferences as inducing selection functions on the set of standard answer sets of the underlying non-ordered program \( \Pi \).
Answer sets are defined via a reduction of extended logic programs to basic programs. Controlling such a reduction by means of preferences is difficult since all conflicts are simultaneously resolved when turning $\Pi$ into $\Pi^X$. Furthermore, we argue that conflict resolution must be addressed among the original rules in order to account for blockage between rules. In fact, once the negative body $\text{body}^- (r)$ is eliminated there is no way to detect whether $\text{head}(r') \in \text{body}^- (r)$ holds in case of $r < r'$. Our idea is thus to characterize preferred answer sets by an inductive development that agrees with the given ordering. In terms of a standard answer set $X$, this means that we favor its formal characterization as $X = \bigcup_{i \geq 0} \mathcal{T}_{\Pi_i, X} \emptyset$ over $X = Cn(\Pi^X)$. This leads us to the following definition.

**Definition 1**

Let $(\Pi, <)$ be an ordered logic program and let $X$ and $Y$ be sets of literals.

We define the set of immediate consequences of $X$ with respect to $(\Pi, <)$ and $Y$ as

$$
\mathcal{T}_{(\Pi, <), Y} X = \begin{cases}
\text{I. } r \in \Pi \text{ is active wrt } (X, Y) \text{ and } \\
\text{II. there is no rule } r' \in \Pi \text{ with } r < r' \text{ such that } \\
\quad (a) \ r' \text{ is active wrt } (Y, X) \text{ and } \\
\quad (b) \ \text{head}(r') \not\in X
\end{cases}
$$

if $X$ is consistent, and $\mathcal{T}_{(\Pi, <), Y} X = \text{Lit}$ otherwise.

Note that $\mathcal{T}_{(\Pi, <), Y}$ is a refinement of its classical counterpart $T_{\Pi_i, Y}$ in (4). The idea behind Condition II is to apply a rule $r$ only if the “question of applicability” has been settled for all higher-ranked rules $r'$. Let us illustrate this in terms of iterated applications of $\mathcal{T}_{(\Pi, <), Y}$.

In this case, $X$ accumulates conclusions, while $\mathcal{Y}$ comprises the putative answer set. Then, the “question of applicability” is considered to be settled for a higher ranked rule $r'$

- if the prerequisites of $r'$ will never be derivable, viz. $\text{body}^+(r') \not\subseteq Y$, or
- if $r'$ is defeated by what has been derived so far, viz. $\text{body}^- (r) \cap X \neq \emptyset$, or
- if $r'$ or another rule with the same head have already applied, viz. $\text{head}(r') \in X$.

The first two conditions show why activeness of $r'$ is stipulated wrt $(Y, X)$, as opposed to $(X, Y)$ in Condition I. The last condition serves two purposes: First, it detects whether the higher ranked rule $r'$ has applied and, second, it suspends the preference $r < r'$ whenever the head of the higher ranked has already been derived by another rule. This suspension of preference constitutes a distinguishing feature of the approach at hand.

As with $T_{\Pi_i, Y}$, iterated applications of $\mathcal{T}_{(\Pi, <), Y}$ are written as $\mathcal{T}^j_{(\Pi, <), Y}$ for $j \geq 0$, where $\mathcal{T}^0_{(\Pi, <), Y} X = X = \mathcal{T}_{(\Pi, <), Y} X$ and $\mathcal{T}^i_{(\Pi, <), Y} X = \mathcal{T}_{(\Pi, <), Y} \mathcal{T}^{i-1}_{(\Pi, <), Y} X$ for $i \geq 1$. The counterpart of operator $C_{\Pi}$ for ordered programs is then defined as follows.

**Definition 2**

Let $(\Pi, <)$ be an ordered logic program and let $X$ be a set of literals.

We define $C_{(\Pi, <)}(X) = \bigcup_{i \geq 0} \mathcal{T}^i_{(\Pi, <), X} \emptyset$.

Clearly, $C_{(\Pi, <)}$ is a refinement of $C_{\Pi}$. The difference is that $C_{(\Pi, <)}$ obtains consequences directly from $\Pi$ and $Y$, while $C_{\Pi}$ (normally) draws them by appeal to $Cn$ after reducing $\Pi$ to $\Pi^Y$. All this allows us to define preferred answer sets as fixpoints of $C_{(\Pi, <)}$. 
Definition 3
Let \((\Pi, \prec)\) be an ordered logic program and let \(X\) be a set of literals.
We define \(X\) as a preferred answer set of \((\Pi, \prec)\) iff \(C((\Pi, \prec))(X) = X\).

For illustration, consider the following ordered logic program \((\Pi, \prec)\):
\[
\begin{align*}
r_1 & : \neg f \leftarrow p, \neg f & r_4 & : b \leftarrow p & r_2 < r_1 \tag{3} \\
r_2 & : w \leftarrow b, \neg w & r_5 & : p \leftarrow \\
r_3 & : f \leftarrow w, \neg f
\end{align*}
\]
Observe that \(\Pi\) admits two answer sets: \(X = \{p, b, \neg f, w\}\) and \(X' = \{p, b, f, w\}\). As argued in [Baader and Hollunder 1993], \(X\) is preferred to \(X'\). To see this, observe that
\[
\begin{align*}
\mathcal{T}_0(\Pi, \prec)_X \theta &= \emptyset & \mathcal{T}_0(\Pi, \prec)_X \emptyset &= \emptyset \\
\mathcal{T}_1(\Pi, \prec)_X \emptyset &= \{p\} & \mathcal{T}_{(\Pi, \prec)}(X, \emptyset) &= \{p\} \tag{4} \\
\mathcal{T}_2(\Pi, \prec)_X \emptyset &= \{p, b, \neg f\} & \mathcal{T}_2(\Pi, \prec)_X \emptyset &= \{p, b\} \\
\mathcal{T}_3(\Pi, \prec)_X \emptyset &= \{p, b, \neg f, w\} & \mathcal{T}_3(\Pi, \prec)_X \emptyset & = \mathcal{T}_3(\Pi, \prec)_X \emptyset \\
\mathcal{T}_4(\Pi, \prec)_X \emptyset &= \mathcal{T}_4(\Pi, \prec)_X \emptyset = X
\end{align*}
\]
We thus get \(C((\Pi, \prec))(X) = X\), while \(C((\Pi, \prec))(X') = \{p, b\} \neq X'\). Note that \(w\) cannot be included into \(\mathcal{T}_3(\Pi, \prec)_X \emptyset\) since \(r_1\) is active wrt \((X', \mathcal{T}_3(\Pi, \prec)_X)\) and \(r_1\) is preferred to \(r_2\).

It is important to see that preferences may sometimes be too strong and deny the existence of preferred answer sets although standard ones exist. This is because preferences impose additional dependencies among rules that must be respected by the resulting answer sets. This is nicely illustrated by programs \(\Pi_1 = \{r_1, r_2\}\) and \(\Pi_2 = \{r_1', r_2'\}\), respectively:
\[
\begin{align*}
r_1 &= a \leftarrow b & r_1' &= a \leftarrow \neg b \\
r_2 &= b \leftarrow & r_2' &= b \leftarrow
\end{align*}
\]
Observe that in \(\Pi_1\), rule \(r_1\) depends \(r_2\), while in \(\Pi_2\), rule \(r_1'\) is defeated by \(r_2'\). But despite the fact that \(\Pi_1\) has answer set \(X = \{a, b\}\) and \(\Pi_2\) has answer set \(X' = \{b\}\), we obtain no preferred answer set after imposing preferences \(r_2 < r_1\) and \(r_2' < r_1'\), respectively. To see this, observe that \(\mathcal{T}_1(\Pi, \prec)_X \emptyset = \mathcal{T}_1(\Pi, \prec)_X \emptyset = \emptyset \neq X\) and \(\mathcal{T}_1(\Pi, \prec)_X \emptyset = \mathcal{T}_1(\Pi, \prec)_X \emptyset = \emptyset \neq X'\). In both cases, the preferred rules \(r_1\) and \(r_1'\), respectively, are (initially) inapplicable: \(a \leftarrow b\) is not active wrt \(\emptyset, \emptyset\) and \(a \leftarrow \neg b\) is not active wrt \(\emptyset, \emptyset\). And the application of the second rule \(b \leftarrow \) is inhibited by Condition II: In the case of \(\mathcal{T}_1(\Pi, \prec)_X \emptyset\), rule \(a \leftarrow b\) is active wrt \(\emptyset, \emptyset\); informally, \(X\) puts the construction on the false front that \(b\) will eventually be derivable. In the case of \(\mathcal{T}_1(\Pi, \prec)_X \emptyset\), rule \(a \leftarrow \neg b\) is active wrt \(\emptyset, \emptyset\). This is due to the conception that a higher-ranked rule can never be defeated by a lower-ranked one.

Formal elaboration. We start with the basic properties of our consequence operator:

Theorem 1
Let \((\Pi, \prec)\) be an ordered program and let \(X\) and \(Y\) be sets of literals. Then, we have:
For $i = 1, 2$, let $X_i$ and $Y_i$ be sets of literals and $<_i \subseteq \Pi \times \Pi$ be strict partial orders.

3. If $X_1 \subseteq X_2$, then $T_{(\Pi, <_1), Y} X_1 \subseteq T_{(\Pi, <_2), Y} X_2$.
4. If $Y_1 \subseteq Y_2$, then $T_{(\Pi, <_2), Y_2} X \subseteq T_{(\Pi, <_1), Y_1} X$.
5. If $<_1 \subseteq <_2$, then $T_{(\Pi, <_1), Y} X \subseteq T_{(\Pi, <_2), Y} X$.

The next results show how our fixpoint operator relates to its classical counterpart.

**Theorem 2**

Let $(\Pi, <)$ be an ordered program and let $X$ be a set of literals. Then, we have:

1. $C_{(\Pi, <)}(X) \subseteq C_{\Pi}(X)$.
2. $C_{(\Pi, <)}(X) = C_{\Pi}(X)$, if $X \subseteq C_{(\Pi, <)}(X)$.
3. $C_{(\Pi, \emptyset)}(X) = C_{\Pi}(X)$.

We obtain the following two corollaries.

**Corollary 3**

Let $(\Pi, <)$ be an ordered logic program and $X$ a set of literals.

If $X$ is a preferred answer set of $(\Pi, <)$, then $X$ is an answer set of $\Pi$.

Our strategy thus implements a selection function among the standard answer sets of the underlying program. This selection is neutral in the absence of preferences, as shown next.

**Corollary 4**

Let $\Pi$ be a logic program and $X$ a set of literals.

Then, $X$ is a preferred answer set of $(\Pi, \emptyset)$ iff $X$ is an answer set of $\Pi$.

Of interest in view of an alternating fixpoint theory is that $C_{(\Pi, <)}$ enjoys anti-monotonicity:

**Theorem 5**

Let $(\Pi, <)$ be an ordered logic program and $X_1, X_2$ sets of literals.

If $X_1 \subseteq X_2$, then $C_{(\Pi, <)}(X_2) \subseteq C_{(\Pi, <)}(X_1)$.

We next show that for any answer set $X$ of a program $\Pi$, there is an ordering $<$ on the rules of $\Pi$ such that $X$ is the unique preferred answer set of $(\Pi, <)$.

**Theorem 6**

Let $\Pi$ be a logic program and $X$ an answer set of $\Pi$. Then, there is a strict partial order $<$ such that $X$ is the unique preferred answer set of the ordered program $(\Pi, <)$.

Our last result shows that a total order selects at most one standard answer set.

**Theorem 7**

Let $(\Pi, \ll)$ be an ordered logic program and $\ll$ be a total order.

Then, $(\Pi, \ll)$ has zero or one preferred answer set.
Relationship to perfect model semantics. Any sensible semantics for logic programming should yield, in one fashion or other, the smallest Herbrand model $Cn(\Pi)$ whenever $\Pi$ is a basic program. A similar consensus seems to exist regarding the perfect model semantics of stratified normal programs [Apt et al. 1987, Przymusinski 1988]. Interestingly, stratified programs can be associated with a rule ordering in a canonical way. We now show that our semantics corresponds to the perfect model semantics on stratified normal programs.

A normal logic program $\Pi$ is stratified, if $\Pi$ has a partition, called stratification, $\Pi = \Pi_1 \cup \ldots \cup \Pi_n$ such that the following conditions are satisfied for $i,j \in \{1,\ldots,n\}$:

1. $\Pi_i \cap \Pi_j = \emptyset$ for $i \neq j$;
2. $\text{body}^+(r) \cap (\bigcup_{k=1}^i \text{head}(\Pi_k)) = \emptyset$ and $\text{body}^-(r) \cap (\bigcup_{k=i+1}^n \text{head}(\Pi_k)) = \emptyset$ for all $r \in \Pi_i$.

That is, whenever a rule $r$ belongs to $\Pi_i$, the atoms in $\text{body}^+(r)$ can only appear in the heads of $\bigcup_{k=1}^i \Pi_k$, while the atoms in $\text{body}^-(r)$ can only appear in the heads of $\bigcup_{k=i+1}^n \Pi_k$.

A stratification somehow reflects an intrinsic order among the rules of a program. In a certain sense, rules in lower levels are preferred over rules in higher levels, insofar as rules in lower levels should be considered before rules in higher levels. Accordingly, the intuition behind the perfect model of a stratified program is to gradually derive atoms, starting from the most preferred rules. Specifically, one first applies the rules in $\Pi_1$, resulting in a set of atoms $X_1$; then one applies the rules in $\Pi_2$ relative to the atoms in $X_1$; and so on.

Formally, the perfect model semantics of a stratified logic program $\Pi = \Pi_1 \cup \ldots \cup \Pi_n$ is recursively defined for $0 < i < n$ as follows [Apt et al. 1987, Przymusinski 1988].

1. $X_0 = \emptyset$
2. $X_{i+1} = \bigcup_{j \geq 0} T_{\Pi_{i+1}, X_i}^j X_i$

The perfect model $X^*$ of $\Pi$ is then defined as $X^* = X_n$.

Let $\Pi$ be a stratified logic program and $\Pi = \Pi_1 \cup \ldots \cup \Pi_n$ be a stratification of $\Pi$. A natural priority relation $<_s$ on $\Pi$ can be defined as follows:

For any $r_1, r_2 \in \Pi$, we define $r_1 <_s r_2$ iff $r_1 \in \Pi_i$ and $r_2 \in \Pi_j$ such that $j < i$.

That is, $r_2$ is preferred to $r_1$ if the level of $r_2$ is lower than that of $r_1$. We obtain thus an ordered logic program $(\Pi, <_s)$ for any stratified logic program $\Pi$ with a fixed stratification.

**Theorem 8**

Let $X^*$ be the perfect model of stratified logic program $\Pi$ and let $<_s$ be an order induced by some stratification of $\Pi$. Then, we have

1. $X^* = C_{(\Pi, <_s)}(X^*)$,
2. If $X \subseteq C_{(\Pi, <_s)}(X)$, then $X^* = X$.

These results imply the following theorem.

**Corollary 9**

Let $X^*$ be the perfect model of stratified logic program $\Pi$ and let $<_s$ be an order induced by some stratification of $\Pi$. Then, $(\Pi, <_s)$ has the unique preferred answer set $X^*$.

Interestingly, both programs $\Pi_3$ as well as $\Pi_5$ are stratifiable. None of the induced orderings, however, contains the respective preference ordering imposed in $\Pi_5$. In fact, this provides an easy criterion for the existence of (unique) preferred answer sets.
Corollary 10

Let $X^\star$ be the perfect model of stratified logic program $\Pi$ and let $<_s$ be an order induced by some stratification of $\Pi$. Let $(\Pi, <)$ be an ordered logic program such that $< \subseteq <_s$.

Then, $(\Pi, <)$ has the unique preferred answer set $X^\star$.

Implementation through compilation. A translation of ordered logic programs to standard programs is developed in (Delgrande et al. 2000b). Although the employed strategy (cf. Section 4) differs from the one put forward in the previous section, it turns out that the computation of preferred answer sets can be accomplished by means of this translation technique in a rather straightforward way. In the framework of (Delgrande et al. 2000b), preferences are expressed within the program via a predicate symbol $\prec$. A logic program over a propositional language $L$ is said to be dynamically ordered iff $L$ contains the following pairwise disjoint categories: (i) a set $N$ of terms serving as names for rules; (ii) a set $At$ of atoms; and (iii) a set $At_\prec$ of preference atoms $s \prec t$, where $s, t \in N$ are names.

For a program $\Pi$, we need a bijective function $n(\cdot)$ assigning a name $n(r) \in N$ to each rule $r \in \Pi$. We sometimes write $n_r$ instead of $n(r)$. An atom $n_r \prec n_{r'} \in At_\prec$ amounts to asserting that $r < r'$ holds. A (statically) ordered program $(\Pi, <)$ can thus be captured by programs containing preference atoms only among their facts; it is then expressed by the program $\Pi \cup \{(n_r \prec n_{r'}) \leftarrow \mid r < r'\}$.

Given $r < r'$, one wants to ensure that $r'$ is considered before $r$ (cf. Condition II in Definition 2). For this purpose, one needs to be able to detect when a rule has been applied or when a rule is defeated. For detecting blockage, a new atom $bl(n_r)$ is introduced for each $r$ in $\Pi$. Similarly, an atom $ap(n_r)$ is introduced to indicate that a rule has been applied. For controlling application of rule $r$ the atom $ok(n_r)$ is introduced. Informally, one concludes that it is ok to apply a rule just if it is ok with respect to every $\prec$-greater rule; for such a $\prec$-greater rule $r'$, this will be the case just when $r'$ is known to be blocked or applied.

More formally, given a dynamically ordered program $\Pi$ over $L$, let $L^+$ be the language obtained from $L$ by adding, for each $r, r' \in \Pi$, new pairwise distinct propositional atoms $ap(n_r), bl(n_r), ok(n_r), and rdy(n_r, n_{r'})$. Then, the translation $T$ maps an ordered program $\Pi$ over $L$ into a standard program $T(\Pi)$ over $L^+$ in the following way.

Definition 4

Let $\Pi = \{r_1, \ldots, r_k\}$ be a dynamically ordered logic program over $L$.

Then, the logic program $T(\Pi)$ over $L^+$ is defined as $T(\Pi) = \bigcup_{r \in \Pi} \tau(r)$, where $\tau(r)$
consists of the following rules, for \( L^+ \in \text{body}^+(r), L^- \in \text{body}^-(r) \), and \( r', r'' \in \Pi \):

\[
\begin{align*}
a_1(r) &= \text{head}(r) \leftarrow \text{ap}(n_r) \\
a_2(r) &= \text{ap}(n_r) \leftarrow \text{ok}(n_r), \text{body}(r) \\
b_1(r, L^+) &= \text{bl}(n_r) \leftarrow \text{ok}(n_r), \text{not } L^+ \\
b_2(r, L^-) &= \text{bl}(n_r) \leftarrow \text{ok}(n_r), L^- \\
c_1(r) &= \text{ok}(n_r) \leftarrow \text{rdy}(n_r, n_{r_1}), \ldots, \text{rdy}(n_r, n_{r_k}) \\
c_2(r, r') &= \text{rdy}(n_r, n_{r'}) \leftarrow \text{not } (n_r \prec n_{r'}) \\
c_3(r, r') &= \text{rdy}(n_r, n_{r'}) \leftarrow (n_r \prec n_{r'}, \text{ap}(n_{r'})) \\
c_4(r, r') &= \text{rdy}(n_r, n_{r'}) \leftarrow (n_r \prec n_{r'}, \text{bl}(n_{r'})) \\
c_5(r, r') &= \text{rdy}(n_r, n_{r'}) \leftarrow (n_r \prec n_{r'}, \text{head}(r')) \\
t(r, r', r'') &= n_r \prec n_{r''} \leftarrow n_r \prec n_{r'}, n_{r'} \prec n_{r''} \\
as(r, r') &= \neg(n_r \prec n_{r'}) \leftarrow n_r \prec n_{r'}
\end{align*}
\]

We write \( \mathbb{T}(\Pi, <) \) rather than \( \mathbb{T}(\Pi') \), whenever \( \Pi' \) is the dynamically ordered program capturing \( \Pi, < \). The first four rules of \( \tau(r) \) express applicability and blocking conditions of the original rules. For each rule \( r \in \Pi \), we obtain two rules, \( a_1(r) \) and \( a_2(r) \), along with \( n \) rules of the form \( b_1(r, L^+) \) and \( m \) rules of the form \( b_2(r, L^-) \), where \( n \) and \( m \) are the numbers of the literals in \( \text{body}^+(r) \) and \( \text{body}^-(r) \), respectively. The second group of rules encodes the strategy for handling preferences. The first of these rules, \( c_1(r) \), “quantifies” over the rules in \( \Pi \). This is necessary when dealing with dynamic preferences since preferences may vary depending on the corresponding answer set. The four rules \( c_i(r, r') \) for \( i = 2, 5 \) specify the pairwise dependency of rules in view of the given preference ordering: For any pair of rules \( r, r' \), we derive \( \text{rdy}(n_r, n_{r'}) \) whenever \( n_r \prec n_{r'} \) fails to hold, or otherwise whenever either \( \text{ap}(n_{r'}) \) or \( \text{bl}(n_{r'}) \) is true, or whenever \( \text{head}(r') \) has already been derived. This allows us to derive \( \text{ok}(n_r) \), indicating that \( r \) may potentially be applied whenever we have for all \( r' \) with \( n_r \prec n_{r'} \) that \( r' \) has been applied or cannot be applied.

It is instructive to observe how close this specification of \( \text{ok}(\cdot) \) and \( \text{rdy}(\cdot, \cdot) \) is to Condition II in Definition \[1\]. In fact, given a fixed \( r \in \Pi \), Condition II can be read as follows.

\[
\begin{align*}
\text{II.} & \quad \text{for every } r' \in \Pi \text{ with } r < r' \text{ either} \\
& \quad (a) \ r' \text{ is not active wrt } (Y, X) \text{ or} \\
& \quad (b) \ \text{head}(r') \in X
\end{align*}
\]

The quantification over all rules \( r' \in \Pi \) with \( r < r' \) is accomplished by means of \( c_1(r) \) (along with \( c_2(r, r') \)). By definition, \( r' \) is not active wrt \( (Y, X) \) \[1\] if either \( \text{body}^+(r) \not\subseteq Y \) or \( \text{body}^-(r) \cap X \neq \emptyset \), both of which are detected by rule \( c_4(r, r') \). The condition \( \text{head}(r') \in X \) is reflected by \( c_3(r, r') \) and \( c_5(r, r') \). While the former captures the case where \( \text{head}(r') \) was supplied by \( r' \) itself, \[2\] the latter accounts additionally for the case where \( \text{head}(r') \) was supplied by another rule than \( r' \).

The next result shows that translation \( \mathbb{T} \) is a realization of operator \( C \).

---

1 Recall that \( X \) is supposed to contain the set of conclusions that have been derived so far, while \( Y \) provides the putative answer set.
2 Strictly speaking rule \( c_3(r, r') \) is subsumed by \( c_4(r, r') \); nonetheless we keep both for conceptual clarity in view of similar translations presented in Section \[4\].
Theorem 11
Let $(\Pi, <)$ be an ordered logic program over $\mathcal{L}$ and let $X \subseteq \{\text{head}(r) \mid r \in \Pi\}$ be a consistent set of literals. Then, there is some set of literals $Y$ over $\mathcal{L}^+$ where $X = Y \cap \mathcal{L}$ such that $C_{(\Pi, <)}(X) = C_{T(\Pi, <)}(Y) \cap \mathcal{L}$.

Note that the fixpoints of $C_{(\Pi, <)}$ constitute a special case the previous theorem.

Theorem 12
Let $(\Pi, <)$ be an ordered logic program over $\mathcal{L}$ and let $X$ and $Y$ be consistent sets of literals. Then, we have that
1. if $C_{(\Pi, <)}(X) = X$, then there is an answer set $Y$ of $\mathcal{T}(\Pi, <)$ such that $X = Y \cap \mathcal{L}$;
2. if $Y$ is an answer set of $\mathcal{T}(\Pi, <)$, then $C_{(\Pi, <)}(Y \cap \mathcal{L}) = Y \cap \mathcal{L}$.

4 Other strategies (and characterizations)
We now show how the approaches of Delgrande et al. (2000b) and Brewka/Eiter (1999; 2000) can be captured within our framework. Also, we take up a complementary characterization provided in (Delgrande et al. 2000b) in order to obtain another insightful perspective on the three approaches. For clarity, we add the letter “W” to all concepts from Section 3. Accordingly we add “D” and “B”, respectively, when dealing with the two aforementioned approaches.

Characterizing D-preference. In (Delgrande et al. 2000b), the selection of preferred answer sets is characterized in terms of the underlying set of generating rules: The set $\Gamma_{\Pi,X}$ of all generating rules of an answer set $X$ of literals from program $\Pi$ is given by
\[ \Gamma_{\Pi,X} = \{r \in \Pi \mid \text{body}^+(r) \subseteq X \text{ and } \text{body}^-(r) \cap X = \emptyset\}. \]

The property distinguishing preferred answer sets from ordinary ones is referred to as order preservation and defined in the following way.

Definition 5
Let $(\Pi, <)$ be an ordered program and let $X$ be an answer set of $\Pi$.
Then, $X$ is called $<^\mathcal{D}$-preserving, if there exists an enumeration $(r_i)_{i \in I}$ of $\Gamma_{\Pi,X}$ such that for every $i, j \in I$ we have that:
1. $\text{body}^+(r_i) \subseteq \{\text{head}(r_j) \mid j < i\}$; and
2. if $r_i < r_j$, then $j < i$; and
3. if $r_i < r'$ and $r' \in \Pi \setminus \Gamma_{\Pi,X}$, then
   \begin{enumerate}
   \item $\text{body}^+(r') \not\subseteq X$ or
   \item $\text{body}^-(r') \cap \{\text{head}(r_j) \mid j < i\} \neq \emptyset$.
   \end{enumerate}

We often refer to $<^\mathcal{D}$-preserving answer sets as D-preferred answer sets.

Condition 1 makes the property of groundedness\(^3\) explicit. Although any standard answer set enjoys this property, we will see that its interaction with preferences varies with

---

\(^3\) This term is borrowed from the literature on default logic (cf. Konolige 1988; Schwind 1990).
the strategy. Condition 2 stipulates that \( (r_i)_{i \in I} \) is compatible with \(<\), a property invariant to all of the considered approaches. Lastly, Condition 3 is comparable with Condition II in Definition 11; it guarantees that rules can never be blocked by lower-ranked rules.

Roughly speaking, an order preserving enumeration of the set of generating rules reflects the sequence of successive rule applications leading to some preferred answer set. For instance, the preferred answer set \( X = \{ p, b, \neg f, w \} \) of Example 4 can be generated by the two order preserving sequences \( \langle r_5, r_4, r_2, r_1 \rangle \) and \( \langle r_5, r_1, r_4, r_2 \rangle \). Intuitively, both enumerations are order preserving since they reflect the fact that \( r_1 \) is treated before \( r_2 \). Although there is another grounded enumeration generating \( X \), namely \( \langle r_5, r_4, r_2, r_1 \rangle \), it is not order preserving since it violates Condition 2. The same applies to the only grounded enumeration \( \langle r_5, r_4, r_2, r_3 \rangle \) that allows to generate the second standard answer set of \( \Pi \). It violates Condition 3b. Consequently, \( X \) is the only \(<^n\)-preserving answer set of \( (\Pi, <^n) \).

We are now ready to provide a fixpoint definition for \( D\)-preference. For this purpose, we assume a bijective mapping \( \text{rule}(\cdot) \) among rule heads and rules, that is, \( \text{rule}(\text{head}(r)) = r \); accordingly, \( \text{rule}(\{ \text{head}(r) \mid r \in R \}) = R \). Such mappings can be defined in a bijective way by distinguishing different occurrences of literals.

**Definition 6**

Let \( (\Pi, <) \) be an ordered logic program and let \( X \) and \( Y \) be sets of literals.

We define the set of immediate \( D\)-consequences of \( X \) with respect to \( (\Pi, <) \) and \( Y \) as

\[
T^0_{(\Pi, <), Y} X = \begin{cases} \text{head}(r) & \text{I. } r \in \Pi \text{ is active wrt } (X, Y) \text{ and } \\
 & \text{II. there is no rule } r' \in \Pi \text{ with } r < r' \\
 & \text{such that } \\
 & \quad (a) \ r' \text{ is active wrt } (Y, X) \text{ and } \\
 & \quad (b) \ r' \notin \text{rule}(X) 
\end{cases}
\]

if \( X \) is consistent, and \( T^0_{(\Pi, <), Y} X = \text{Lit} \) otherwise.

The distinguishing feature between this definition and Definition 11 manifests itself in IIb. While \( D\)-preference requires that a higher-ranked rule has effectively applied, \( W\)-preference contents itself with the presence of the head of the rule, no matter whether this was supplied by the rule itself.

Defining iterated applications of \( T^0_{(\Pi, <), Y} \) in analogy to those of \( T_{(\Pi, <), Y} \), we may capture \( D\)-preference by means of a fixpoint operator in the following way.

**Definition 7**

Let \( (\Pi, <) \) be an ordered logic program and let \( X \) be a set of literals.

We define \( C^0_{(\Pi, <)}(X) = \bigcup_{i \geq 0} T^0_{(\Pi, <), Y} X^i \).

A similar elaboration of \( C^0_{(\Pi, <)} \) as done with \( C^W_{(\Pi, <)} \) in Section 5 yields identical formal properties; in particular, \( C^0_{(\Pi, <)} \) also enjoys anti-monotonicity.

The aforementioned difference is nicely illustrated by extending the programs in (5) by

\footnote{Note that both enumerations are compatible with the iteration through \( T_{(\Pi, <), Y} X^i \) for \( i = 0, 1 \).}
rule \( a \leftarrow \), yielding \((\Pi)_5^\star <\) and \((\Pi)_5 <')\), respectively:

\[
\begin{align*}
  r_1 &= a \leftarrow b & r'_1 &= a \leftarrow \neg b \\
  r_2 &= b \leftarrow & r'_2 &= b \leftarrow \\
  r_3 &= a \leftarrow & r'_3 &= a \leftarrow \\
  r_2 &< r_1 & r'_2 &< r'_1
\end{align*}
\]

While in both cases the single standard answer set is \(W\)-preferred, neither of them is \(D\)-preferred. Let us illustrate this in terms of the iterated applications of \(T^w_{(\Pi),X}\) and \(T^w_{(\Pi),X'}\), where \(X = \{a, b\}\) is the standard answer set of \((\Pi)\). At first, both operators allow for applying rule \(a \leftarrow\), resulting in \(\{a\}\). As with \(T^w_{(\Pi),X}\) in (5), however, operator \(T^w_{(\Pi),X'}\) does not allow for applying \(r_2\) at the next stage, unless \(r_1\) is inactive. This requirement is now dropped by \(T^w_{(\Pi),X}\), since the head of \(r_1\) has already been derived through \(r_3\). In such a case, the original preference is ignored, which enables the application of \(r_2\). In this way, we obtain the \(W\)-preferred answer set \(X = \{a, b\}\). The analogous behavior is observed on \((\Pi)_6^\star <'\).

As \(W\)-preferred answer sets, \(D\)-preferred ones coincide with the perfect model on stratified programs.

**Theorem 13**

Let \(X^*\) be the perfect model of stratified logic program \(\Pi\) and let \(<_s\) be an order induced by some stratification of \(\Pi\). Then, \((\Pi, <_s)\) has the unique \(D\)-preferred answer set \(X^*\).

The subtle difference between \(D\)- and \(W\)-preference is also reflected in the resulting compilation. Given the same prerequisites as in Definition 9, the logic program \(T^0(\Pi)\) over \(L^+\) is defined as \(T^0(\Pi) = T^w(\Pi) \setminus \{c_5(r, r') \mid r, r' \in \Pi\}\). Hence, in terms of this compilation technique, the distinguishing feature between \(D\)- and \(W\)-preference manifests itself in the usage of rule \(c_5(r, r') : \text{rdy}(n_r, n_{r'}) \leftarrow (n_r < n_{r'}), \text{head}(r')\). While \(W\)-preference allows for suspending a preference whenever the head of the preferred rule was derived, \(D\)-preference stipulates the application of the preferred rule itself. This is reflected by the fact that the translation \(T^0\) merely uses rule \(c_5(r, r') : \text{rdy}(n_r, n_{r'}) \leftarrow (n_r < n_{r'}), \text{ap}(n_{r'})\) to enforce that the preferred rule itself has been applied. This demonstrates once more how closely the compilation technique follows the specification given in the fixpoint operation.

As shown in (Delgrande et al. 2000b), a set of literals \(X\) is a \(<^D\)-preserving answer set of a program \(\Pi\) iff \(X = Y \cap L\) for some answer set \(Y\) of \(T^0(\Pi, <)\). This result naturally extends to the fixpoint operator \(C^D_{(\Pi, <)}\), as shown in the following result.

**Theorem 14**

Let \((\Pi, <)\) be an ordered logic program over \(L\) and let \(X\) be a consistent set of literals. Then, the following propositions are equivalent.

1. \(C^D_{(\Pi, <)}(X) = X\);
2. \(X = Y \cap L\) for some answer set \(Y\) of \(T^0(\Pi, <)\);
3. \(X\) is a \(<^D\)-preserving answer set of \(\Pi\).

While the last result dealt with effective answer sets, the next one shows that applying \(C^D_{(\Pi, <)}\) to the translated program \(\Pi' = T^0(\Pi, <)\).
Theorem 15
Let \((\Pi, <)\) be an ordered logic program over \(L\) and let \(X \subseteq \{\text{head}(r) \mid r \in \Pi\}\) be a consistent set of literals. Then, there is some set of literals \(Y\) over \(L^+\) where \(X = Y \cap L\) such that \(C_{(\Pi, <)}^0(X) = C_{(\Pi, <)}^\omega(Y) \cap L\).

Characterizing \(W\)-preference (alternatively). We now briefly elaborate upon a characterization of \(W\)-preference in terms of order preservation. This is interesting because order preservation provides an alternative perspective on the formation of answer sets. In contrast to the previous fixpoint characterizations, order preservation furnishes an account of preferred answer sets in terms of the underlying generating rules. While an immediate consequence operator provides a rather rule-centered and thus local characterization, order preservation gives a more global and less procedural view on an entire construction. In particular, the underlying sequence nicely reflects the interaction of its properties. In fact, we see below that different approaches distinguish themselves by a different degree of interaction between groundedness and preferences.

Definition 8
Let \((\Pi, <)\) be an ordered program and let \(X\) be an answer set of \(\Pi\). Then, \(X\) is called \(<_W\)-preserving, if there exists an enumeration \(\langle r_i \rangle_{i \in I}\) of \(\Gamma_{\Pi} X\) such that for every \(i, j \in I\) we have that:
1. (a) \(\text{body}^+(r_i) \subseteq \{\text{head}(r_j) \mid j < i\}\) or (b) \(\text{head}(r_i) \in \{\text{head}(r_j) \mid j < i\}\); and
2. if \(r_i < r_j\), then \(j < i\); and
3. if \(r_i < r'\) and \(r' \in \Pi \setminus \Gamma_{\Pi} X\), then (a) \(\text{body}^+(r') \notin X\) or (b) \(\text{body}^-(r') \cap \{\text{head}(r_j) \mid j < i\} \neq \emptyset\) or (c) \(\text{head}(r') \in \{\text{head}(r_j) \mid j < i\}\).

The primary difference between this concept of order preservation and the one for \(D\)-preference is clearly the weaker notion of groundedness. While \(D\)-preference makes no compromise when enforcing rule dependencies induced by preference, \(W\)-preference “smoothes” their integration with those induced by groundedness and defeat relationships: First, regarding rules in \(\Gamma_{\Pi} X\) (via Condition 1b) and second concerning rules in \(\Pi \setminus \Gamma_{\Pi} X\) (via Condition 3c). The rest of the definition is identical to Definition 5.

This “smoothed” integration of preferences with groundedness and defeat dependencies is nicely illustrated by programs \((\Pi, <)\) and \((\Pi', <)\). Regarding \(\Pi\), we observe that there is no enumeration of \(\Gamma_{\Pi} X\) satisfying both Condition 1a and 2. Rather it is Condition 1b that weakens the interaction between both conditions by tolerating enumeration \(\langle r_3, r_2, r_1 \rangle\). A similar observation can be made regarding \(\Pi'\) where, in contrast to \(\Pi\), the preferred rule \(r_1'\) does not belong to \(\Gamma_{\Pi} X\). We observe that there is no enumeration of \(\Gamma_{\Pi} X\) satisfying both Condition 2 and 3a/b. Now, it is Condition 3c that weakens the interaction between both conditions by tolerating enumeration \(\langle r_3', r_2' \rangle\). In fact, the two examples show that both Condition 1b as well as 3c function as exceptions to conditions 1a and 3a/b, respectively. In this way, \(W\)-preference imposes the same requirements as \(D\)-preference, unless the head of the rule in focus has already been derived by other means.

Finally, we have the following summarizing result.
**Theorem 16**
Let \((\Pi, <)\) be an ordered logic program over \(\mathcal{L}\) and let \(X\) be a consistent set of literals. Then, the following propositions are equivalent.

1. \(C_{\Pi, <}^{\text{w}}(X) = X\);
2. \(X = Y \cap \mathcal{L}\) for some answer set \(Y\) of \(T^{\text{w}}(\Pi, <)\);
3. \(X\) is a \(<^{\text{w}}\)-preserving answer set of \(\Pi\).

**Characterizing B-preference.** Another approach to preference is proposed in (Brewka and Eiter 1999). This approach differs in two ways from the previous ones. First, the construction of answer sets is separated from verifying preferences. Interestingly, this verification is done on the basis of the prerequisite-free program obtained from the original one by “evaluating” \(\text{body}^{+}(r)\) for each rule \(r\) wrt the separately constructed (standard) answer set. Second, rules that may lead to counter-intuitive results are explicitly removed. This is spelled out in (Brewka and Eiter 2000), where the following filter is defined:

\[
\mathcal{E}_X(\Pi) = \Pi \setminus \{ r \in \Pi \mid \text{head}(r) \in X, \text{body}^{-}(r) \cap X \neq \emptyset \} \quad (7)
\]

Accordingly, we define \(\mathcal{E}_X(\Pi, <) = (\mathcal{E}_X(\Pi), < \cap (\mathcal{E}_X(\Pi) \times \mathcal{E}_X(\Pi)))\).

We begin with a formal account of \(B\)-preferred answer sets. In this approach, partially ordered programs are reduced to totally ordered ones: A **fully ordered logic program** is an ordered logic program \((\Pi, <)\) where \(<\) is a total ordering. The case of arbitrarily ordered programs is reduced to this restricted case: Let \((\Pi, <)\) be an ordered logic program and let \(X\) be a set of literals. Then, \(X\) is a \(B\)-preferred answer set of \((\Pi, <)\) iff \(X\) is a \(B\)-preferred answer set of some fully ordered logic program \((\Pi, \triangleleft)\) such that \(< \triangleleft \triangleleft \).

The construction of \(B\)-preferred answer sets relies on an operator, defined for prerequisite-free programs, comprising only rules \(r\) with \(\text{body}^{+}(r) = \emptyset\).

**Definition 9**
Let \((\Pi, \triangleleft)\) be a fully ordered prerequisite-free logic program, let \(\langle r_i \rangle_{i \in I}\) be an enumeration of \(\Pi\) according to \(\triangleleft\), and let \(X\) be a set of literals. Then, \(B_{\Pi, \triangleleft}(X)\) is the smallest logically closed set of literals containing \(\bigcup_{i \in I} X_i\), where \(X_j = \emptyset\) for \(j \notin I\) and

\[
X_i = \begin{cases} 
X_{i-1} \bigcup \{ \text{head}(r_i) \} & \text{if } \text{body}^{-}(r_i) \cap X_{i-1} \neq \emptyset \\
X_{i-1} & \text{otherwise.}
\end{cases}
\]

This construction is unique insofar that for any such program \((\Pi, \triangleleft)\), there is at most one standard answer set \(X\) of \(\Pi\) such that \(B_{\Pi, \triangleleft}(X) = X\). Accordingly, this set is used for defining the **\(B\)-preferred answer set** of a prerequisite-free logic program:

**Definition 10**
Let \((\Pi, \triangleleft)\) be a fully ordered prerequisite-free logic program and let \(X\) be a set of literals. Then, \(X\) is the \(B\)-preferred answer set of \((\Pi, \triangleleft)\) iff \(B_{\Pi, \triangleleft}(X) = X\).

The reduction of \((\Pi, \triangleleft)\) to \(\mathcal{E}_X(\Pi, \triangleleft)\) removes from the above construction all rules whose heads are in \(X\) but which are defeated by \(X\). This is illustrated in (Brewka and Eiter 2000) through the following example:

\[
\begin{align*}
r_1 &= a \leftarrow \text{not } b, & r_3 &= a \leftarrow \text{not } \neg a, & \{ r_j < r_i \mid i < j \} . \\
r_2 &= \neg a \leftarrow \text{not } a, & r_4 &= b \leftarrow \text{not } \neg b,
\end{align*}
\]
Program $\Pi_9 = \{r_1, \ldots, r_4\}$ has two answer sets, $\{a, b\}$ and $\{-a, b\}$. The application of operator $B$ relies on sequence $(r_1, r_2, r_3, r_4)$. Now, consider the processes induced by $B_{\mathcal{E}_X}(\Pi_{\leq <}(X))$ and $B_{\mathcal{E}_X}(\Pi_{\geq <}(X))$ for $X = \{a, b\}$, respectively:

$$\begin{align*}
B_{\mathcal{E}_X}(\Pi_{\leq <}(X)) : & \quad X_1 = \{\} \quad X_2 = \{-a\} \quad X_3 = \{-a\} \quad X_4 = \{-a, b\} \\
B_{\mathcal{E}_X}(\Pi_{\geq <}(X)) : & \quad X'_1 = \{a\} \quad X'_2 = \{a\} \quad X'_3 = \{a\} \quad X'_4 = \{a, b\}
\end{align*}$$

Thus, without filtering by $\mathcal{E}_X$, we get $\{a, b\}$ as a $B$-preferred answer set. As argued in (Brewka and Eiter 2000), such an answer set does not preserve priorities because $r_2$ is defeated in $\{a, b\}$ by applying a rule which is less preferred than $r_2$, namely $r_3$. The above program has therefore no $B$-preferred answer set.

The next definition accounts for the general case by reducing it to the prerequisite-free one. For checking whether an answer set $X$ is $B$-preferred, the prerequisites of the rules are evaluated wrt $X$. For this purpose, we define $r^- = head(r) \leftarrow body^-(r)$ for a rule $r$.

**Definition 11**
Let $(\Pi, \ll)$ be a fully ordered logic program and $X$ a set of literals.

The logic program $(\Pi_X, \ll_X)$ is obtained from $(\Pi, \ll)$ as follows:

1. $\Pi_X = \{r^- \mid r \in \Pi$ and $body^+(r) \subseteq X\}$;
2. for any $r_1', r_2' \in \Pi_X$, $r_1' \ll_X r_2'$ iff $r_1 \ll r_2$ where $r_i = \max_{\ll} \{r \in \Pi \mid r^- = r_i\}$.

In other words, $\Pi_X$ is obtained from $\Pi$ by first eliminating every rule $r \in \Pi$ such that $body^+(r) \not\subseteq X$, and then substituting all remaining rules $r$ by $r^-$. In general, $B$-preferred answer sets are then defined as follows.

**Definition 12**
Let $(\Pi, \ll)$ be a fully ordered logic program and $X$ a set of literals.

Then, $X$ is a $B$-preferred answer set of $(\Pi, \ll)$, if

1. $X$ is a (standard) answer set of $\Pi$, and
2. $X$ is a $B$-preferred answer set of $(\Pi_X, \ll_X)$.

The distinguishing example of this approach is given by program $(\Pi_{\leq <})$:

$$\begin{align*}
& r_1 = \quad b \leftarrow a, not \neg b & \text{with } \{r_j \ll r_i \mid i < j\} \\
& r_2 = \quad \neg b \leftarrow not b \\
& r_3 = \quad a \leftarrow not \neg a
\end{align*}$$

Program $\Pi_{\leq <} = \{r_1, r_2, r_3\}$ has two standard answer sets: $X_1 = \{a, b\}$ and $X_2 = \{a, \neg b\}$. Both $(\Pi_{\leq <})_X_1$ as well as $(\Pi_{\leq <})_X_2$ turn $r_1$ into $b \leftarrow not \neg b$ while leaving $r_2$ and $r_3$ unaffected. Clearly, $\mathcal{E}_X(\Pi_{\leq <}) = (\Pi_{\leq <})$ for $i = 1, 2$. Also, we obtain that $B_{\mathcal{E}_X}(\Pi_{\leq <})(X_1) = X_1$, that is, $X_1$ is a $B$-preferred answer set. In contrast to this, $X_2$ is not $B$-preferred. To see this, observe that $B_{\mathcal{E}_X}(\Pi_{\leq <})(X_2) = X_1 \neq X_2$. That is, $B_{\mathcal{E}_X}(\Pi_{\leq <})(X_2)$ reproduces $X_1$ rather than $X_2$. In fact, while $X_1$ is the only $B$-preferred set, neither $X_1$ nor $X_2$ is $w$- or $d$-preferred (see below).

We note that $B$-preference disagrees with $w$- and $d$-preference on Example $\models$. In fact, both answer sets of program $(\Pi_{\leq <})$ are $B$-preferred, while only $\{p, b, \neg f, w\}$ is $w$- and $d$-preferred. In order to shed some light on these differences, we start by providing a fixpoint characterization of $B$-preference:
Definition 13
Let \((\Pi, <)\) be an ordered logic program and let \(X\) and \(Y\) be sets of literals.

We define the set of immediate consequences of \(X\) with respect to \((\Pi, <)\) and \(Y\) as

\[
T^{n}_{(\Pi, <), Y} X = \begin{cases} 
\text{head}(r) & \text{I. } r \in \Pi \text{ is active wrt } (Y, Y) \text{ and} \\
\text{II. } \text{there is no rule } r' \in \Pi \text{ with } r < r' \text{ such that } \\
(a) r' \text{ is active wrt } (Y, X) \text{ and} \\
(b) \text{head}(r') \notin X
\end{cases}
\]

if \(X\) is consistent, and \(T^{n}_{(\Pi, <), Y} X = \text{Lit}\) otherwise.

The difference between this operator and its predecessors manifests itself in Condition I, where activeness is tested wrt \((Y, Y)\) instead of \((X, Y)\), as in Definition 1 and 4. In fact, in Example 9 it is the (unprovability of the) prerequisite \(a\) of the highest-ranked rule \(r_1\) that makes the construction of \(W\)- or \(D\)-preferred answer sets break down (cf. Definition 1 and 4). This is avoided with \(B\)-preference because once answer set \(\{a, b\}\) is provided, preferences are enforced wrt the program obtained by replacing \(r_1\) with \(b \leftarrow \text{not } \neg b\).

With an analogous definition of iterated applications of \(T^{n}_{(\Pi, <), Y} X\) as above, we obtain the following characterization of \(B\)-preference:

Definition 14
Let \((\Pi, <)\) be an ordered logic program and let \(X\) be a set of literals.

We define \(C^{n}_{(\Pi, <)}(X) = \bigcup_{i \geq 0} (T^{n})_{(\Pi, <), X} \emptyset\).

Unlike above, \(C^{n}_{(\Pi, <)}\) is not anti-monotonic. This is related to the fact that the “answer set property” of a set is verified separately (cf. Definition 12). We have the following result.

Theorem 17
Let \((\Pi, <)\) be an ordered logic program over \(L\) and let \(X\) be an answer set of \(\Pi\).

Then, we have that \(X\) is \(B\)-preferred iff \(C^{n}_{X}(\Pi, <)(X) = X\).

As with \(D\)- and \(W\)-preference, \(B\)-preference gives the perfect model on stratified programs.

Theorem 18
Let \(X^*\) be the perfect model of stratified logic program \(\Pi\) and let \(<_s\) be an order induced by some stratification of \(\Pi\). Then, \((\Pi, <_s)\) has the unique \(B\)-preferred answer set \(X^*\).

Alternatively, \(B\)-preference can also be captured by appeal to order preservation:

Definition 15
Let \((\Pi, <)\) be an ordered program and let \(X\) be an answer set of \(\Pi\).

Then, \(X\) is called \(<_B\)-preserving, if there exists an enumeration \(\langle r_i \rangle_{i \in I}\) of \(\Gamma_{\Pi} X\) such that, for every \(i, j \in I\), we have that:

1. if \(r_i < r_j\), then \(j < i\); and
2. if \(r_i < r'\) and \(r' \in \Pi \setminus \Gamma_{\Pi} X\), then

\(^5\) We have refrained from integrating \(\Pi\) in order to keep the fixpoint operator comparable to its predecessors. This is taken care of in Theorem 19. We note however that an integration of \(\Pi\) would only affect Condition II.
A semantic framework for preference handling in answer set programming

(a) \( \text{body}^+ (r') \not\subseteq X \) or 
(b) \( \text{body}^- (r') \cap \{ \text{head}(r_j) \mid j < i \} \neq \emptyset \) or 
(c) \( \text{head}(r') \in X \).

This definition differs in two ways from its predecessors. First, it drops any requirement on groundedness. This corresponds to using \((Y, Y)\) instead of \((X, Y)\) in Definition 13. Hence, groundedness is fully disconnected from order preservation. For example, the B-preferred answer set \( \{a, b\} \) of \((\Pi_9, <)\) is associated with the \(<B\)-preserving sequence \( \langle r_1, r_2 \rangle \), while the standard answer set is generated by the grounded sequence \( \langle r_2, r_1 \rangle \). Second, Condition 2c is more relaxed than in Definition 8. That is, any rule \( r' \) whose head is in \( X \) (as opposed to \( \{ \text{head}(r_j) \mid j < i \} \)) is taken as “applied.” Also, Condition 2c integrates the filter in (7).

For illustration, consider Example (6) extended by \( r_3 < r_2 \):

\[
\begin{align*}
r_1 &= a \leftarrow \text{not } b \\
r_2 &= b \leftarrow \\
r_3 &= a \leftarrow \\
r_3 < r_2 < r_1
\end{align*}
\]

While this program has no D- or W-preferred answer set, it has a B-preferred one: \( \{a, b\} \) generated by \( \langle r_2, r_3 \rangle \). The critical rule \( r_1 \) is handled by 2c. As a net result, Condition 2 is weaker than its counterpart in Definition 8. We have the following summarizing result.

**Theorem 19**

Let \((\Pi, <)\) be an ordered logic program over \( \mathcal{L} \) and let \( X \) be a consistent answer set of \( \Pi \). Then, the following propositions are equivalent.

1. \( X \) is B-preferred;
2. \( C^{<B}_X (\Pi, <) (X) = X \);
3. \( X \) is a \(<B\)-preserving answer set of \( \Pi \);
4. \( X = Y \cap \mathcal{L} \) for some answer set \( Y \) of \( \mathcal{T}^B(\Pi, <) \)
   (where \( \mathcal{T}^B \) is defined in (Delgrande et al. 2000a)).

Unlike theorems 14 and 16, the last result stipulates that \( X \) must be an answer set of \( \Pi \). This requirement can only be dropped in case 4, while all other cases rely on this property.

**Relationships.** First of all, we observe that all three approaches treat the blockage of (higher-ranked) rules in the same way. That is, a rule \( r' \) is found to be blocked if either its prerequisites in \( \text{body}^+ (r') \) are never derivable or if some member of \( \text{body}^- (r') \) has been derived by higher-ranked or unrelated rules. This is reflected by the identity of conditions IIa and 2a/b in all three approaches, respectively. Although this is arguably a sensible strategy, it leads to the loss of preferred answer sets on programs like \((\Pi_9', <')\).

The difference between D- and W-preference can be directly read off Definition II and IV, it manifests itself in Condition IIb and leads to the following relationships.

**Theorem 20**

Let \((\Pi, <)\) be an ordered logic program such that for \( r, r' \in \Pi \) we have that \( r \neq r' \) implies \( \text{head}(r) \neq \text{head}(r') \). Let \( X \) be a set of literals. Then, \( X \) is a D-preferred answer set of \((\Pi, <)\) iff \( X \) is a W-preferred answer set of \((\Pi, <)\).

---

6 Condition \( \text{body}^- (r') \cap X \neq \emptyset \) in (7) is obsolete because \( r' \not\in \Gamma_{\Pi} X \).
The considered programs deny the suspension of preferences under \( W \)-preference, because all rule heads are derivable in a unique way. We have the following general result.

**Theorem 21**
Every \( D \)-preferred answer set is \( W \)-preferred.

Example 6 shows that the converse does not hold.

Interestingly, a similar relationship is obtained between \( W \)- and \( B \)-preference. In fact, Definition 15 can be interpreted as a weakening of Definition 8 by dropping the condition on groundedness and weakening Condition 2 (via 2c). We thus obtain the following result.

**Theorem 22**
Every \( W \)-preferred answer set is \( B \)-preferred.

Example 9 shows that the converse does not hold.

Let \( \text{AS}(\Pi) = \{ X \mid C_\Pi(X) = X \} \) and \( \text{AS}_P(\Pi, <) = \{ X \in \text{AS}(\Pi) \mid X \text{ is } P \text{-preferred} \} \) for \( P = w, d, b \). Then, we obtain the following summarizing result.

**Theorem 23**
Let \((\Pi, <)\) be an ordered logic program. Then, we have

\[
\text{AS}_D(\Pi, <) \subseteq \text{AS}_W(\Pi, <) \subseteq \text{AS}_B(\Pi, <) \subseteq \text{AS}(\Pi)
\]

This hierarchy is primarily induced by a decreasing interaction between groundedness and preference. While \( D \)-preference requires the full compatibility of both concepts, this interaction is already weakened in \( W \)-preference, before it is fully abandoned in \( B \)-preference. This is nicely reflected by the evolution of the condition on groundedness in definitions 5, 8, and 15. Notably, groundedness as such is not the ultimate distinguishing factor, as demonstrated by the fact that prerequisite-free programs do not necessarily lead to the same preferred answer sets, as witnessed in 6 and 10. Rather it is the degree of integration of preferences within the standard reasoning process that makes the difference.

Taking together theorems 9, 13, and 18, we obtain the following result.

**Theorem 24**
Let \( \times \) be the perfect model of stratified logic program \( \Pi \) and let \( <_s \) be an order induced by some stratification of \( \Pi \). Let \((\Pi, <)\) be an ordered logic program such that \( < \subseteq <_s \).

Then, we have \( \text{AS}_D(\Pi, <) = \text{AS}_W(\Pi, <) = \text{AS}_B(\Pi, <) = \text{AS}(\Pi) = \{ \times \} \).

### 5 Discussion and related work

Up to now, we have been dealing with static preferences only. In fact, all fixpoint characterizations are also amenable to dynamically ordered programs, as introduced in Section 4. To see this, consider Definition 1 along with a dynamically ordered program \( \Pi \) and sets of literals \( X, Y \) over a language extended by preference atoms \( At_\prec \). Then, the corresponding preferred answer sets are definable by substituting \( "r \prec r'" \) by \( "(r \prec r') \in Y" \) in definitions 1, 6 and 13 respectively. That is, instead of drawing preference information from...
the external order $<$, we simply consult the initial context, expressed by $Y$. In this way, the preferred answer sets of $\Pi$ can be given by the fixpoints of an operator $C_\Pi$.

Also, we have concentrated so far on preferred answer sets semantics that amount to selection functions on the standard answer sets of the underlying program. Another strategy is advocated in (Gelfond and Son 1997), where the preference $d_1 < d_2$ "stops the application of default $d_2$ if defaults $d_1$ and $d_2$ are in conflict with each other and the default $d_1$ is applicable" (Gelfond and Son 1997). In contrast to $B$-, $D$-, and $W$-preference this allows for exclusively concluding $\neg p$ from program $\{r_1, r_2\}$, $<$:

$$r_1 = p \leftarrow r_2 = \neg p \leftarrow r_1 < r_2$$

This approach amounts to $B$-preference on certain "hierarchically" structured programs (Gelfond and Son 1997). A modification of the previous compilation techniques for this strategy is discussed in (Delgrande and Schaub 2000). Although conceptually different, one finds similar strategies when dealing with inheritance, update and/or dynamic logic programs (Buccafurri et al. 1999; Eiter et al. 2000; Alferes et al. 1998), respectively.

While all of the aforementioned approaches remain within the same complexity class, other approaches step up in the polynomial hierarchy (Rintanen 1995; Sakama and Inoue 1996; Zhang and Foo 1997). Among them, preferences on literals are investigated in (Sakama and Inoue 1996). In contrast to these approaches, so-called courteous logic programs (Grosof 1997) step down the polynomial hierarchy into $P$. Due to the restriction to acyclic positive logic programs a courteous answer set can be computed in $O(n^2)$ time. Other preference-based approaches that exclude negation as failure include (Dimopoulos and Kakas 1995; Pradhan and Minker 1996; You et al. 2001) as well as the framework of defeasible logics (Nute 1987; Nute 1994). A comparision of the latter with preferred well-founded semantics (as defined in (Brewka 1996)) can be found in (Brewka 2001).

In a companion paper, we exploit our fixpoint operators for defining regular and well-founded semantics for ordered logic programs within an alternating fixpoint theory. This yields a surprising yet negative result insofar as these operators turn out to be too weak in the setting of well-founded semantics. We address this by defining a parameterizable framework for preferred well-founded semantics, summarized in (Schaub and Wang 2002).

6 Conclusion

The notion of preference seems to be pervasive in logic programming when it comes to knowledge representation. This is reflected by numerous approaches that aim at enhancing logic programming with preferences in order to improve knowledge representation capacities. Despite the large variety of approaches, however, only very little attention has been paid to their structural differences and sameness, finally leading to solid semantical underpinnings. In particular, there were up to now only few attempts to characterize one approach in terms of another one. The lack of this kind of investigation is clearly due to the high diversity of existing approaches.

\[7\] This material was removed from this paper due to space restrictions.
This work is a first step towards a systematic account to logic programming with preferences. To this end, we employ fixpoint operators following the tradition of logic programming. We elaborated upon three different approaches that were originally defined in rather heterogeneous ways. We obtained three alternative yet uniform ways of characterizing preferred answer sets (in terms of fixpoints, order preservation, and an axiomatic account). The underlying uniformity provided us with a deeper understanding of how and which answer sets are preferred in each approach. This has led to a clarification of their relationships and subtle differences. On the one hand, we revealed that the investigated approaches yield an increasing number of answer sets depending on how tight they connect preference to groundedness. On the other hand, we demonstrated how closely the compilation technique follows the specification given in the fixpoint operation. Also, we have shown that all considered answer sets semantics correspond to the perfect models semantics whenever the underlying ordering stratifies the program.

We started by formally developing a specific approach to preferred answer sets semantics that is situated “between” the approaches of Delgrande et al. (2000b) and that of Brewka and Eiter (1999). This approach can be seen as a refinement of the former approach in that it allows to suspend preferences whenever the result of applying a preferred rule has already been derived. This feature avoids the overly strict prescriptive approach to preferences pursued in (Delgrande et al. 2000b), which may lead to the loss of answer sets.

Acknowledgements. This work was supported by the German Science Foundation (DFG) under grant FOR 375/1-1, TP C. We are grateful to the anonymous referees, although we were unable to follow all suggestions due to severe space restrictions.

7 Proofs

Proof[1] It can be directly verified from the definition of $T_{\Pi,\cdot}$.  

Proof[2]  
1. $C_{\Pi,\cdot}(X) \subseteq C_P(X)$: Since $C_{\Pi,\cdot}(X) = \bigcup_{i \geq 0} T_{\Pi,\cdot}^i \cdot X \emptyset$ and $C_P(X) = T_{\Pi,X}^0 \emptyset$, we need only to prove that $T_{\Pi,\cdot}^i \cdot X \emptyset \subseteq T_{\Pi,X}^0 \emptyset$ for $i \geq 0$ by using induction on $i$.

**Base** For $i = 0$, it is obvious that $T_{\Pi,\cdot}^0 \cdot X \emptyset = \emptyset \subseteq T_{\Pi,X}^0 \emptyset$.

**Step** Assume that $T_{\Pi,\cdot}^i \cdot X \emptyset \subseteq T_{\Pi,X}^i \emptyset$, we want to show that $T_{\Pi,\cdot}^{i+1} \cdot X \emptyset \subseteq T_{\Pi,X}^{i+1} \emptyset$.

In fact, if $L \in T_{\Pi,\cdot}^{i+1} \cdot X \emptyset$, then, by Definition 1 there is a rule $r$ in $\Pi$ such that $L = \text{head}(r)$, $\text{body}_r^+(r) \subseteq T_{\Pi,\cdot}^i \cdot X \emptyset$ and $\text{body}_r^-(r) \cap X = \emptyset$. By induction assumption, $\text{body}_r^+(r) \subseteq T_{\Pi,X}^i \emptyset$. Since the rule $L \leftarrow \text{body}_r^+(r)$ is in the reduct program $P_X$, $L \in T_{\Pi,X}^i \emptyset$.

2. $C_P(X) \subseteq C_{\Pi,\cdot}(X)$ if $X \subseteq C_{\Pi,\cdot}(X)$: For simplicity, we denote $T_i = T_{\Pi,X}^i \emptyset$ and $X_i = T_{\Pi,\cdot}^i \cdot X \emptyset$ for $i \geq 0$. It suffices to prove $T_{\Pi,X}^i \emptyset \subseteq C_{\Pi,\cdot}(X)$ for $k \geq 0$ by using induction on $k$. That is, for each $i \geq 0$, there is $n_i \geq 0$ such that $T_i \subseteq X_{n_i}$.

**Base** If $k = 1$, it is obvious that $T_{\Pi,X}^0 \emptyset \subseteq X_0$.

**Step** Assume that $T_i \subseteq X_{n_i}$. We want to show $T_{i+1} \subseteq X_{n_{i+1}}$. Let $a \in T_{i+1}$, then there is a rule $r \in \Gamma$ with $\text{head}(r) = a$, $\text{body}_r^+(r) \subseteq T_i$ and $\text{body}_r^-(r) \cap X = \emptyset$. 


By the induction assumption, \( r \) is active wrt \((X_n, X)\). We claim that there will be no rule \( r' \) such that both of Condition I and II hold wrt \((X_n, X)\). Otherwise, suppose that there is a rule \( r' \) such that head\((r') \not\subseteq X_n, r < r' \) and \( r' \) is active wrt \((X, X_n)\). Without loss of generality, there is no rule \( r'' \) such that head\((r'') \not\subseteq X_n, r < r'' < r' \) and \( r'' \) is active wrt \((X, X_n)\). Since \( X \subseteq \mathcal{C}_{(\Pi, \prec)}(X) \), there be a number \( n \geq n_i \) such that \( r' \) is active wrt \((X_n, X)\). By the assumption of \( r'' \), it should be that head\((r'') \in X_n\). A contradiction. Therefore, head\((r) \in X_{n+1}\).

3. If \( \prec \) is empty, then the condition II in Definition | is automatically satisfied because, for any rule \( r \in \Pi \), there is no rule \( r' \) that is preferred to \( r \). This implies that \( T_{(\Pi, \prec)}^i(\emptyset) = T_{\Pi, X}^i(\emptyset) \) for any \( i \geq 0 \). Therefore, \( \mathcal{C}_{(\Pi, \prec)}(X) = \mathcal{C}_P(X) \).

**Proof** If \( X \subseteq X' \), it is a direct induction on \( i \) to show that \( T_{(\Pi, \prec)}^i(\emptyset) \subseteq T_{(\Pi, \prec)}^i(\emptyset) \).

**Proof** If \( \Pi \) has no consistent answer set, the conclusion is obvious. Thus, we assume that \( X \) is consistent. First, we can easily generalize the notion of generating rules as follows: For any two sets \( Y_1 \) and \( Y_2 \) of literals, set \( \Gamma(Y_1, Y_2) = \{ \text{head}(r) \leftarrow \text{body}^+(r) \mid \text{body}^+(r) \subseteq Y_1, \text{body}^-(r) \cap Y_2 = \emptyset \} \).

Since \( X \) is an answer set of \( \Pi \), we have \( X = \mathcal{C}_\Pi(X) = \bigcup_{i \geq 0} T_{\Pi, X}^i(\emptyset) \). Let \( \Gamma_0 = \Gamma(T_{\Pi, X}^0(X), X) \) and \( \Gamma_{k+1} = \Gamma(T_{\Pi, X}^k(X) - \Gamma_k \bigcup X \) for \( k \geq 1 \). Define a total order \( \ll_X \) on \( \Pi \) such that the following requirements are satisfied:

1. \( r' \ll_X r \) for any \( r \in \Gamma_k \) and \( r' \in \Gamma_{k+1}, k = 0, 1, \ldots \).
2. If \( r \in \cup_{n \geq 0} \Gamma_n \) and \( r' \not\in \cup_{n \geq 0} \Gamma_n \), then \( r' \ll_X r \).

Since \( \Gamma_k \cap \Gamma_{k'} \not\subseteq \emptyset \) for \( n \neq n' \), such an ordering exists. Denote \( X_i = T_{(\Pi, \ll_X)}^i(\emptyset) \). We need only to prove the following two propositions P1 and P2:

**P1** \( X \) is a prioritized answer set of \((\Pi, \ll_X)\): Since \( \mathcal{C}_P(X) = X \), it suffices to prove that \( \mathcal{C}_{(\Pi, \ll_X)}(X) = \mathcal{C}_P(X) \). Firstly, by Theorem | \( \mathcal{C}_{(\Pi, \ll_X)}(X) \subseteq \mathcal{C}_P(X) \). For the opposite inclusion, we note that \( \mathcal{C}_P(X) = \text{head}(\cup_{n \geq 0} \Gamma_k) \), where \( \text{head}(\cup_{k \geq 0} \Gamma_k) = \{ \text{head}(r) \mid r \in \cup_{k \geq 0} \Gamma_k \} \). Hence, we need only to prove that \( \text{head}(\Gamma_k) \subseteq \mathcal{C}_{(\Pi, \ll_X)}(X) \) for any \( k \geq 0 \) by using induction on \( k \).

**Base** For \( k = 0 \), without loss of generality, suppose that \( \Gamma_0 = \{ r_1, \ldots, r_l \} \) and \( r_t \ll_X \cdots \ll_X r_1 \). We use second induction to show that head\((r_i) \in \mathcal{C}_P(X) \) for \( 1 \leq i \leq t \).

**Base** For \( i = 1 \), since there is no rule \( r' \) with \( r_1 \ll_X r' \), head\((r_1) \in X_1 \).

**Step** Assume that head\((r_i) \in X_i \), then head\((r_{i+1}) \in X_{i+1} \). Thus head\((\Gamma) \subseteq X_i \).

**Step** Assume that head\((\Gamma_k) \subseteq \mathcal{C}_{(\Pi, \ll_X)}(X) \). Then head\((\Gamma_k) \in X_{m_k} \) for some \( m_k > 0 \).

Let \( \Gamma_{k+1} = \{ r_1, \ldots, r_u \} \) and \( r_u \ll_X \cdots \ll_X r_1 \). Then, similar to the case of \( k = 0 \), we have that head\((r_i) \in X_{m_{k+1}} \) for \( i = 1, \ldots, u \).

Thus, head\((\Gamma_k) \subseteq \mathcal{C}_{(\Pi, \ll_X)}(X) \) for any \( k \geq 0 \).

This implies that \( \mathcal{C}_P(X) \subseteq \mathcal{C}_{(\Pi, \ll_X)}(X) \). Therefore, \( \mathcal{C}_{(\Pi, \ll_X)}(X) = X \).

**P2** If \( X' \) is an answer set of \( \Pi \) such that \( X' \neq X \), then \( X' \) is not a prioritized answer set of \((\Pi, \ll_X)\): First note that \( X \setminus X' \not\subseteq \emptyset \) and \( X' \setminus X \not\subseteq \emptyset \). We assert that there is literal \( l \in X \setminus X' \) such that \( l \not\in \mathcal{C}_{(\Pi, \ll_X)}(X') \); otherwise, \( X \setminus X' \subseteq \mathcal{C}_{(\Pi, \ll_X)}(X') \). We can choose \( t \geq 0 \) and a literal \( l_0 \in X \setminus X' \) such that \( X'_t \subseteq X \cap X' \) and \( l_0 \in X_{t+1} \). Then
there is a rule \( r \) such that head\( (r) = l_0 \), body\( ^+ (r) \subseteq X'_{ij} \) and body\( ^− (r) \cap X' = \emptyset \). This will implies that \( l \in C_{\Pi ^X \setminus \emptyset } \), i.e. \( l \in X' \), contradiction. Therefore, we have shown that there is a rule \( r \in \Pi \) such that head\( (r) \in X \) and head\( (r) \notin C_{(\Pi, \prec X)} \). For each \( l' \in X' \setminus X \) and each rule \( r' \) such that head\( (r') = l' \), we have \( r' \prec r \). Thus, we know that \( l' \notin C_{\Pi ^X \setminus \emptyset } \). This means that \( X' \notin C_{(\Pi, \prec X)} \) and thus, \( X' \) is not a prioritized answer set of \( (\Pi, \prec X) \).

\[\square\]

**Proof**

On the contrary, suppose that \( (\Pi, \prec) \) has two distinct prioritized answer sets \( X \) and \( X' \). Since \( X \setminus X' \neq \emptyset \) and \( X' \setminus X \neq \emptyset \), there are literals \( l \) and \( l' \) such that \( l \in X \setminus X' \) and \( l' \in X' \setminus X \). Without loss of generality, assume that \( T^l_{\Pi, \prec X} \emptyset = T^l_{\Pi, \prec X} \emptyset \) for \( l \leq n \) but \( l \in T^l_{\Pi, \prec X} \emptyset \) and \( l' \notin T^{l'}_{\Pi, \prec X} \emptyset \). This means that there are two rules \( r \) and \( r' \) such that head\( (r) = l \), head\( (r') = l' \) and \( r \) and \( r' \) satisfy the two conditions \( l \) and \( X' \) in Definition \( \Pi \) at stage \( n \) with respect to \( X \) and \( X' \), respectively. We observe two obvious facts: F1. \( r' \) is active wrt \( (X, T^l_{\Pi, \prec X} \emptyset) \); and F2. \( r \) is active wrt \( (X', T^{l'}_{\Pi, \prec X} \emptyset) \). By F1, we have \( r' \prec r \). Similarly, by F2, it should be \( r \prec r' \), contradiction. Therefore, \( (\Pi, \prec X) \) has the unique prioritized answer sets.

\[\square\]

**Proof**

1. \( X^* = M_t \) is a prioritized answer set of \( (\Pi, \prec_s): X^* = C_{(\Pi, \prec_s)} \). 
   a. \( C_{(\Pi, \prec_s)} \subseteq X^* \): we show that \( T^i_{\Pi, \prec_s} X \setminus \emptyset \subseteq X^* \) by using induction on \( i \).

   - **Base** For \( i = 0 \), \( T^0_{\Pi, \prec_s} \emptyset = X^* \) is obvious.

   - **Step** Assume that \( T^i_{\Pi, \prec_s} X \setminus \emptyset \subseteq X^* \). If \( p \in T^{i+1}_{\Pi, \prec_s} X \setminus \emptyset \), then there is a rule \( r \in \Pi \) such that head\( (r) \subseteq T^i_{\Pi, \prec_s} X \setminus \emptyset \) and body\( ^+ (r) \cap X^* = \emptyset \). By induction assumption, \( body^+(r) \subseteq X^* \). If \( r \in \Pi^j \), then \( body^+(r) \subseteq M_j \) and \( body^-(r) \cap M_{j-1} = \emptyset \). Therefore, \( p = X^* \). That is, \( T^{i+1}_{\Pi, \prec_s} X \setminus \emptyset \subseteq X^* \).

   b. \( X^* \subseteq C_{(\Pi, \prec_s)} \): we show that \( M_i \subseteq C_{(\Pi, \prec_s)} \) for \( 0 \leq i \leq t \).

   - **Base** For \( i = 1 \), it is obvious since \( M_0 = \emptyset \).

   - **Step** We show that \( M_i \subseteq C_{(\Pi, \prec_s)} \). We again use second induction on \( k \) to prove that if \( p \in T^k_{\Pi, M_i} X \), then \( p \in C_{(\Pi, \prec_s)} \).

   - **Base** For \( k = 1 \), i.e. \( p \in T^1_{\Pi, M_i} X \), if \( p \notin M_i \), then there is a rule \( r \in \Pi \) such that head\( (r) \), body\( ^+(r) = \emptyset \) and body\( ^− (r) \cap M_i = \emptyset \). Then body\( ^-(r) \cap X^* = \emptyset \). By the first induction assumption, \( M_i \subseteq T^j_{\Pi, \prec_s} X \setminus \emptyset \) for some \( j_0 \). If there are \( j > 0 \) and a rule \( r' \) such that \( r < s \) and \( r' \) is active with respect to \( (X^*, T^j_{\Pi, \prec_s} X \setminus \emptyset) \) and head\( (r') \notin T^j_{\Pi, \prec_s} X \setminus \emptyset \). Then, body\( ^+(r') \subseteq X^* \) and body\( ^− (r') \cap T^j_{\Pi, \prec_s} X \setminus \emptyset = \emptyset \). We assert that \( j < j_0 \). Otherwise, if \( j > j_0 \), body\( ^− (r') \cap T^j_{\Pi, \prec_s} X \setminus \emptyset = \emptyset \) then body\( ^− (r') \cap X^* = \emptyset \). Therefore, head\( (r') \in M_i \subseteq T^j_{\Pi, \prec_s} X \setminus \emptyset \), a contradiction. Thus, when \( j > j_0 \), there will be no rule in \( \Pi \) that prevents \( r \) to be included in \( T^j_{\Pi, \prec_s} X \setminus \emptyset \). Thus, \( p \in C_{(\Pi, \prec_s)} \).
Clearly, we have $X_i > 0 \subseteq \text{be precise, we show for every } r \in \Pi_{i+1} \text{ such that } p = \text{head}(r), \text{body}^+(r) \subseteq T_{\Pi_{i+1}, M_i}^k 0 \text{ and } \text{body}^-(r) \cap M_i = \emptyset. \text{Then } \text{body}^+(r) \subseteq M_i \subseteq T_{(\Pi, <, \bot), X}^j \emptyset \text{ for some } j_0 \text{ and } \text{body}^-(r) \cap X^* = \emptyset. \text{Similar to the proof of the case } k = 1, \text{we can also prove that } p \in C_{(\Pi, <, \bot)}(X^*)$.

2. If $X = C_{(\Pi, <, \bot)}(X)$, then $X$ is a preferred answer set of $(\Pi, <, \bot)$. By Corollary $X$ is also an answer set of $\Pi$. However, $\Pi$ has the unique answer set $X^*$ and thus $X = X^*$. □

Proof$[6]$

By Theorem$\text{8}(1)$, the perfect model $X^*$ is a preferred answer set. On the other hand, since each preferred answer set $X$ is also a standard answer set. In particular, for the stratified program $\Pi$, it has the unique answer set $X^*$. Therefore, $X = X^*$. □

Proof$[7]$

Let $(\Pi, <)$ be an ordered logic program over $\mathcal{L}$ and $X$ a consistent set of literals over $\mathcal{L}$.

“$\subseteq$”-part Define$^8$

$$Y = \{\text{head}(r) \mid r \in \text{rule}(C_2(\Pi, <))(Y))\}$$
$$\cup \{\text{ap}(n_r) \mid r \in \text{rule}(C_{T(\Pi, <)}(Y))\} \cup \{\text{bl}(n_r) \mid r \notin \text{rule}(C_{T(\Pi, <)}(Y))\}$$
$$\cup \{\text{ok}(n_r) \mid r \in \Pi\} \cup \{\text{rdy}(n_r, n_{r'}) \mid r, r' \in \Pi\}$$

Clearly, we have $X = Y \cap \mathcal{L}$. By definition, we have $C_{(\Pi, <)}(X) = \bigcup_{i \geq 0} T_{(\Pi, <), X}^i \emptyset$ and $C_{T(\Pi, <)}(Y) = Cn(T(\Pi, <))^Y$.

In view of this, we show by induction that $T_{(\Pi, <), X}^i \emptyset \subseteq Cn(T(\Pi, <))^Y$ for $i \geq 0$. To be precise, we show for every $r \in \Pi$ by nested induction that $\text{head}(r) \in T_{(\Pi, <), X}^i \emptyset$ implies $\text{head}(r) \in Cn(T(\Pi, <))^Y$ and moreover, for every $r' \in \Pi$, that if $r < r'$ then $\text{bl}(n_r) \in Cn(T(\Pi, <))^Y$ or $\text{ap}(n_r) \in Cn(T(\Pi, <))^Y$ or $\text{head}(n_r) \in Cn(T(\Pi, <))^Y$.

$i = 0$ By definition, $T_{(\Pi, <), X}^0 \emptyset = \emptyset \subseteq Cn(T(\Pi, <))^Y$.

$i > 0$ Consider $r \in \Pi$ such that $\text{head}(r) \in T_{(\Pi, <), X}^{i+1} \emptyset$. By definition, we have that $r$ is active wrt $T_{(\Pi, <), X}^i \emptyset, X)$. That is,

1. $\text{body}^+(r) \subseteq T_{(\Pi, <), X}^i \emptyset$. By the induction hypothesis, we get $\text{body}^+(r) \subseteq Cn(T(\Pi, <))^Y$.
2. $\text{body}^-(r) \cap X = \emptyset$. By definition of $Y$, this implies $\text{body}^-(r) \cap Y = \emptyset$.

Furthermore, this implies that $a_2(r)^+ = \text{ap}(n_r) \leftarrow \text{ok}(n_r), \text{body}^+(r) \in T(\Pi, <)^Y$.

We proceed by induction on $\lt$. 

Base Suppose $r$ is maximal with respect to $\lt$. We can show the following lemma.

$^8$ As defined in Section$[7]$, $\text{rule}(\cdot)$ is a bijective mapping between rule heads and rules.
Lemma 7.1
If \( r \in \Pi \) is maximal with respect to \(<\), then \( \text{ok}(n_r) \in Cn(\mathbb{T}(\Pi, <)^Y) \).

Given that we have just shown in \[\text{Lemma 7.1}\] and \[\text{Lemma 7.2}\] that body\(^+(r) \subseteq Cn(\mathbb{T}(\Pi, <)^Y) \) and \( a_2(r)^+ \in \mathbb{T}(\Pi, <)^Y \), Lemma \[\text{Lemma 7.2}\] and the fact that \( Cn(\mathbb{T}(\Pi, <)^Y) \) is closed under \( \mathbb{T}(\Pi, <)^Y \) imply that \( \text{ap}(n_r) \in Cn(\mathbb{T}(\Pi, <)^Y) \). Analogously, we get head\((r) \in Cn(\mathbb{T}(\Pi, <)^Y) \) due to \( a_1(r)^+ \in \mathbb{T}(\Pi, <)^Y \). We have thus shown that \( \{\text{head}(r), \text{ap}(n_r)\} \subseteq Cn(\mathbb{T}(\Pi, <)^Y) \).

**Step** We start by showing the following auxiliary result.

Lemma 7.2
Given the induction hypothesis, we have \( \text{ok}(n_{r^\prime}) \in Cn(\mathbb{T}(\Pi, <)^Y) \).

**Proof**
Consider \( r'' \in \Pi \) such that \( r' < r'' \). By the induction hypothesis, we have either \( \text{bl}(n_{r''}) \in Cn(\mathbb{T}(\Pi, <)^Y) \) or \( \text{ap}(n_{r''}) \in Cn(\mathbb{T}(\Pi, <)^Y) \) or \( \text{head}(n_{r''}) \in Cn(\mathbb{T}(\Pi, <)^Y) \). Clearly, we have \( n_{r'} \prec n_{r''} \in Cn(\mathbb{T}(\Pi, <)^Y) \) iff \( r' < r'' \).

Hence, whenever \( r' < r'' \), we obtain \( \text{rdy}(n_{r'}, n_{r''}) \in Cn(\mathbb{T}(\Pi, <)^Y) \) by means of \( c_3(r', r'')^+, c_4(r', r'')^+ \), or \( c_5(r', r'')^+ \) (all of which belong to \( \mathbb{T}(\Pi, <)^Y \)). Similarly, we get \( \text{rdy}(n_{r'}, n_{r''}) \in Cn(\mathbb{T}(\Pi, <)^Y) \), whenever \( r' < r'' \) from \( c_3(r', r'')^+ \). Lastly, we obtain \( \text{ok}(n_{r'}) \in Cn(\mathbb{T}(\Pi, <)^Y) \) via \( c_1(r')^+ \in \mathbb{T}(\Pi, <)^Y \).

For all rules \( r'' \) with \( r < r'' \), we have that either

1. \( r' \) is not active wrt \( (X, T^i) \). That is, we have that either
   
   (a) \( \text{body}^+(r) \nsubseteq X \). By definition of \( Y \), this implies \( \text{body}^+(r) \nsubseteq Y \).
      
      By definition, \( b_1(r', L')^+ = \text{bl}(n_{r'}) \leftarrow \text{ok}(n_{r'}) \in \mathbb{T}(\Pi, <)^Y \) for some \( L^+ \in \text{body}^+(r) \) such that \( L^+ \notin Y \). By Lemma \[\text{Lemma 7.2}\], we have \( \text{ok}(n_{r'}) \in Cn(\mathbb{T}(\Pi, <)^Y) \). Given that \( Cn(\mathbb{T}(\Pi, <)^Y) \) is closed under \( \mathbb{T}(\Pi, <)^Y \), we get that \( \text{bl}(n_{r'}) \in Cn(\mathbb{T}(\Pi, <)^Y) \).

   (b) \( \text{body}^-(r) \cap T^i \neq \emptyset \). By the induction hypothesis, this implies that \( \text{body}^-(r) \cap Cn(\mathbb{T}(\Pi, <)^Y) \neq \emptyset \).
      
      Therefore, \( b_2(r, L)^+ = \text{bl}(n_{r'}) \leftarrow \text{ok}(n_{r'}) \in \mathbb{T}(\Pi, <)^Y \) for some \( L \in \text{body}^-(r) \cap Cn(\mathbb{T}(\Pi, <)^Y) \). In analogy to \[\text{Lemma 7.2}\] this allows us to conclude that \( \text{bl}(n_{r'}) \in Cn(\mathbb{T}(\Pi, <)^Y) \).

In both cases, we conclude \( \text{bl}(n_{r'}) \in Cn(\mathbb{T}(\Pi, <)^Y) \). By the induction assumption, \( \text{head}(r') \in Cn(\mathbb{T}(\Pi, <)^Y) \).

We have thus shown that either \( \text{bl}(n_{r'}) \in Cn(\mathbb{T}(\Pi, <)^Y) \) or \( \text{head}(r') \in Cn(\mathbb{T}(\Pi, <)^Y) \) for all \( r' \) such that \( r < r' \).

In analogy to what we have shown in the proof of Lemma \[\text{Lemma 7.2}\], we can now show that \( \text{ok}(n_{r'}) \in Cn(\mathbb{T}(\Pi, <)^Y) \).

In analogy to the base case, we may then conclude \( \{\text{head}(r), \text{ap}(n_{r'})\} \subseteq Cn(\mathbb{T}(\Pi, <)^Y) \).

**“⊇”-part** We have \( X = Y \cap \mathcal{L} \). By definition, we have \( Cn(T^i_{(\Pi, <)^Y} = Cn(\mathbb{T}(\Pi, <)^Y) \) and moreover that \( Cn(\mathbb{T}(\Pi, <)^Y) = \bigcup_{i \geq 0} T^i_{(\Pi, <)^Y} \emptyset \). Given this, we show by induction that \( (T^i_{(\Pi, <)^Y} \cap \mathcal{L}) \subseteq C(\Pi, <)(X) \) for \( i \geq 0 \).
A semantic framework for preference handling in answer set programming

i = 0 By definition, \( T_0^{\mathbb{T}(\Pi, <)} X_i = \emptyset \subset C_{(\Pi, <)}(X) \).

i > 0 Consider \( r \in \Pi \) such that \( \text{head}(r) \in (T_{T_{\mathbb{T}(\Pi, <)}} X_i) \). In view of \( \mathbb{T}(\Pi, <) Y \), this implies that \( \text{ap}(nr_r) \in (T_{T_{\mathbb{T}(\Pi, <)}} X_i) \). By the induction hypothesis, we obtain from 2 that \( \text{body}^+(T_{T_{\mathbb{T}(\Pi, <)}} X_i) \cap \emptyset \). By the induction hypothesis, we obtain that \( \text{body}^+(r) \subseteq C_{(\Pi, <)}(X) \). Consequently, \( r \) is active wrt \( (C_{(\Pi, <)}(X), X) \).

Suppose there is some \( r' \in \Pi \) with \( r < r' \) such that

1. \( r' \) is active wrt \( (X, C_{(\Pi, <)}(X)) \). That is,
   (a) \( \text{body}^+(r') \subseteq X \) and
   (b) \( \text{body}^-(r') \cap C_{(\Pi, <)}(X) = \emptyset \).
2. \( \text{head}(r') \notin C_{(\Pi, <)}(X) \).

By the induction hypothesis, we obtain from 2 that \( \text{head}(r') \notin T_{T_{\mathbb{T}(\Pi, <)}} Y \) for \( j \leq i \).

Clearly, we have \( (n_r, n_{r'}) \in T_{T_{\mathbb{T}(\Pi, <)}} \emptyset \) for \( i \geq 1 \) iff \( r < r'' \). Moreover, \( \text{ok}(nr_r) \in T_{T_{\mathbb{T}(\Pi, <)}} Y \) implies (see above) \( \text{ok}(n_{r'}, n_{r''}) \in T_{T_{\mathbb{T}(\Pi, <)}} Y \) for all \( r'' \in \Pi \). This and the fact that \( \text{head}(r') \notin T_{T_{\mathbb{T}(\Pi, <)}} Y \) for \( j \leq i \) implies that \( \text{bl}(n_r) \in T_{T_{\mathbb{T}(\Pi, <)}} Y \).

This makes us distinguish the following two cases.

1. If \( \text{bl}(n_r) \) is provided by \( b_1(r', L^+) \), then there is some \( L^+ \in \text{body}^+(r') \) such that \( L^+ \notin Y \). Given that \( X = Y \cap \mathcal{L} \), this contradicts 1a.
2. If \( \text{bl}(n_r) \) is provided by \( b_2(r', L^-) \), then there is some \( L^- \in \text{body}^-(r') \) such that \( L^- \notin T_{T_{\mathbb{T}(\Pi, <)}} Y \). By the induction hypothesis, we obtain that \( L^- \notin C_{(\Pi, <)}(X) \). A contradiction to 1b.

So, given that \( r \) is active wrt \( (C_{(\Pi, <)}(X), X) \) and that there is no \( r' \in \Pi \) such that \( r < r' \) satisfying 1a 1b and 2 we have that \( \text{head}(r) \in T_{\mathbb{T}(\Pi, <), X} (C_{(\Pi, <)}(X)) \). That is, \( \text{head}(r) \in C_{(\Pi, <)}(X) \).

Proof 12 It follows from Lemma 7.7 and Lemma 7.8

Proof 13 Similar to the proof of Theorem 8

By Theorem 4.8 in [Delgrande et al. 2003], it suffices to show the following Lemma 7.3.

Before doing this, we first present a definition. Given a statically ordered logic program \( (\Pi, <) \) and a set \( X \) of literals, set \( X_i = (T^i)_{\Pi, X} \emptyset \) for \( i \geq 0 \).

Definition 16

Let \( (\Pi, <) \) be a statically ordered logic program and \( r \) be a rule in \( \Pi \). \( X_i (i \geq 1) \) are as above. We say another rule \( r' \) is a \text{-preventer of} \( r \) in the context \( (X, X_i) \) if (1) \( r < r' \) and (2) \( r' \) is active wrt \( (X, X_i) \) and \( r' \notin \text{rule}(X_i) \).

Lemma 7.3

Let \( (\Pi, <) \) be a statically ordered logic program and \( X \) a set of literals with \( C_{(\Pi, <)}(X) = X \). Then, for any \( r \in \Gamma_{\Pi, X} \), there exists a number \( i \) such that \( r \in \text{rule}(X_i) \).

The intuition behind this lemma is that each \text{-preventer of} \( r \) in \( \Gamma_{\Pi, X} \) is a “temporary” one if \( C_{(\Pi, <)}(X) = X \).
Proof

On the contrary, suppose that there is a rule \( r \in \Gamma_1 X \) such that \( r \not\in \text{rule}(X_i) \) for any \( i \).
Without loss of generality, assume that there is no such rule that is preferred than \( r \).

Since \( r \in \Gamma_1 X \) and \( X = \bigcup_{t=1}^{\infty} X_t \), \( r \) will become active \( \text{wrt} (X_t, X) \) at some stage \( t \geq 0 \). Therefore, it must be the case that there is a \( \delta \)-preventer \( r' \) satisfying \( r' \in \Gamma_1 X \).

This implies that \( r < r' \) and \( r' \in \Gamma_1 X \) but \( r' \not\in \text{rule}(X_i) \) for any \( i \), contradiction to our assumption on \( r \). Thus, the lemma is proven. \( \square \)

Lemma 7.4

Let \( (\Pi, \prec) \) be a statically ordered logic program and \( X \) a set of literals. Then \( X \) is a \( <^\Pi \)-preserving answer set of \( \Pi \) if and only if \( X \) is a set of literals with \( C^\Pi_\prec(X) = X \).

Proof

Without loss of generality, assume that \( \text{rule}(X_i) = \{r_{i1}, \ldots, r_{in_i}\} \) for \( i \geq 1 \).

if part Let \( C^\Pi_\prec(X) = X \). By Lemma 7.3, \( \Gamma_1 X = \bigcup_{i=1}^{\infty} \text{rule}(X_i) \). This means that the sequence \( \Delta; \langle r_{i1}, \ldots, r_{in_1}, r_{12}, \ldots, r_{2n_2}, \ldots \rangle \) is an enumeration of \( \Gamma_1 X \).

It suffices to prove that this sequence of rules in \( \Delta \) is \( <^\Pi \)-preserving with respect to \( X \).

We need to justify the two conditions of \( <^\Pi \)-sequence are satisfied by \( \Delta \):

C1 For each \( r_i \in \text{rule}(X_i) \) where \( t > 0 \), then \( r_i \) is active \( \text{wrt} (X_{t-1}, X) \). This implies that \( \text{body}^+(r_i) \subseteq \{\text{head}(r_j) \mid j < i\} \).

C2 if \( r < r' \), then \( r' \) is prior to \( r \) in \( \Delta \): notice that, since \( X = \bigcup_{t=1}^{\infty} X_t \), if a rule is active \( \text{wrt} (X_t, X) \) then it is also active \( \text{wrt} (X, X_t) \). Thus, by Definition 7.1 \( r \) and \( r' \) can not be in the same section \( \text{rule}(X_t) \). If C2 is not satisfied by \( \Delta \), then there are two rules, say \( r \) and \( r' \) such that \( r < r' \) but \( r \) is prior to \( r' \) in \( \Delta \). Without loss of generality, assume that \( r \in \text{rule}(X_i) \) and \( r' \in \text{rule}(X_j) \) but \( i < j \). Then \( r' \) should prevent \( r \) to be included in \( \text{rule}(X_i) \), which means \( r \not\in \text{rule}(X_i) \), contradiction. Therefore, C2 holds.

C3 if \( r_i < r' \) and \( r' \in \Pi \setminus \Gamma_1 X \), then \( \text{body}^+(r') \not\subseteq X \) or \( r' \) is defeated by the set \( \{\text{head}(r_j) \mid j < i\} \): Assume that \( r_i \in \text{rule}(X_i) \), then \( r' \not\in \text{rule}(X_i) \). On the contrary, assume that \( \text{body}^+(r') \subseteq X \) and \( r' \) is not defeated by the set \( \{\text{head}(r_j) \mid j < i\} \), then \( r' \) is not defeated by \( X_{i-1} \) because \( X_{i-1} \subseteq \{\text{head}(r_j) \mid j < i\} \). Thus, \( r' \) is active \( \text{wrt} (X, X_{i-1}) \) and \( r' \not\in \text{rule}(X_{i-1}) \). This means that \( r' \) is a \( \delta \)-preventer of \( r_i \) in the context \( (X, X_{i-1}) \) and thus, \( r_i \not\in \text{rule}(X_i) \), contradiction. That is, \( \text{body}^+(r') \not\subseteq X \) or \( r' \) is defeated by the set \( \{\text{head}(r_j) \mid j < i\} \).

only-if part Assume that \( X \) is a \( <^\Pi \)-preserving answer set of \( \Pi \), then there is a grounded enumeration \( \langle r_i \rangle_{i \in I} \) of \( \Gamma_1 X \) such that, for every \( i, j \in I \), we have that:

1. if \( r_i < r_j \), then \( j < i \); and
2. if \( r_i < r' \) and \( r' \in \Pi \setminus \Gamma_1 X \), then either (a) \( \text{body}^+(r') \not\subseteq X \) or (b) \( \text{body}^-(r') \cap \{\text{head}(r_j) \mid j < i\} \neq \emptyset \).

A set \( \Delta \) of rules is discrete if there is no pair of rules \( r \) and \( r' \) in \( \Delta \) s. t. \( r < r' \).
We define recursively a sequence of sets of rules as follows.
Define \( \Delta_1 \) as the largest section of \( \langle r_i \rangle_{i \in I} \) satisfying the following conditions:
We prove the two propositions by parallel induction on ordering.

Proof 7.5
Let \( r_1 \in \Delta_1 \);
\( \text{body}(r) = \emptyset \) for any \( r \in \Delta_1 \).

Suppose that \( \Delta_1 \) is well-defined and \( r_{m_1} \) is the last rule of \( \Delta_1 \), we define \( \Delta_{i+1} \) as the largest section of \( \langle r_i \rangle \subseteq I \) satisfying the following conditions:

1. \( \Delta_{i+1} \) is discrete;
2. \( r_{m_{i+1}} \in \Delta_{i+1} \);
3. \( \text{body}(r) \subseteq \{ \text{head}(r') \mid r' \in \bigcup_{j=0}^{i} \Delta_j \} \) for any \( r \in \Delta_{i+1} \).
4. disjoint with \( \bigcup_{j=0}^{i} \Delta_j \).

Denote \( \bar{X}_i = \{ \text{head}(r) \mid r \in \bigcup_{j=0}^{i} \Delta_j \} \). Then we have the following fact:

if \( r \in \Delta_{i+1} \) such that \( \bar{X}_{i-1} = \text{body}(r) \) and no rule \( r' \in \Delta_i \) with \( r < r' \), then we can move \( r \) from \( \Delta_{i+1} \) to \( \Delta_i \), the resulting sequence of rules still is \(<^0\)-preserving.

Without loss of generality, assume that our sequence \( \langle \Delta_i \rangle \) is fully transformed by the above transformation. Since \( \bigcup_{i=0}^{\infty} \bar{X}_i = X \), we can prove \( \bigcup_{i=0}^{\infty} X_i = X \) by proving \( \bar{X}_i = X_i \) for every \( i \in I \). Thus, it suffices to prove \( \Delta_i = \text{rule}(X_i) \) for every \( i \in I \). We use induction on \( i \):

**Base** \( \Delta_0 = \text{rule}(X_0) = \emptyset \).

**Step** Assume that \( \Delta_i = \text{rule}(X_i) \), we want to prove \( \Delta_{i+1} = \text{rule}(X_{i+1}) \).

\( \Delta_{i+1} \subseteq \text{rule}(X_{i+1}) \): For any \( r \in \Delta_{i+1} \), by the condition 3 in the construction of \( \Delta_{i+1} \), \( r \) is active wrt \( (X_i, X) \). And for any \( r' \) such that \( r < r' \) and \( r' \) is active wrt \( (X, X_i) \) then \( \text{body}^+(r') \subseteq X \) and \( r' \) is not defeated by \( X_i \). By induction, \( \bigcup_{k<i} \text{head}(r_k) \subseteq X_i = \bar{X}_i \), thus \( r' \) is not defeated by \( \bigcup_{k<i} \text{head}(r_k) \). By Definition 5 it should be the case that \( r' \in \Pi X \), which implies that \( r' \in \Delta_i = \text{rule}(X_i) \). Therefore, \( r' \) is not a \( D \)-preventer of \( r_i \).

That is, \( r_{i+1} \in \text{rule}(X_{i+1}) \).

\( \text{rule}(X_{i+1}) \subseteq \Delta_{i+1} \): For \( r \in \text{rule}(X_{i+1}) \), we claim that \( r \in \Delta_{i+1} \). Otherwise, there would exist \( t > i + 1 \) such that \( r \in \Delta_t \). Notice that, by induction assumption, \( \text{body}^+(r) \subseteq X_i \). Thus, it must be the case that there is at least one rule \( r' \in \bigcup_{j=i+1}^{t} \text{rule}(X_j) \) such that \( r' \in \Delta_t \). But \( r' \) is active wrt \( (X, X_{i+1}) \), which contradicts to \( r \in \text{rule}(X_{i+1}) \). Therefore, \( \text{rule}(X_{i+1}) \subseteq \Delta_{i+1} \).

**Proof** 7.5 Similar to the proof of Theorem 11.

**Proof** 7.6 It follows from the following Lemma 7.7 and Theorem 12.

**Lemma 7.5**
Let \( (\Pi, <) \) be an ordered logic program over \( L \) and let \( Y \) be a consistent answer set of \( T^w_{(\Pi, <)} \). Denote \( X = Y \cap L \). Then, we have for any \( r \in \Pi \):

1. \( \text{ok}(n_r) \in Y \); and
2. \( \text{ap}(n_r) \in Y \) iff \( \text{bl}(n_r) \not\in Y \).

**Proof** 7.8
We prove the two propositions by parallel induction on ordering \( < \).
The intuition behind this lemma is that each \( W_r \in X \). Without loss of generality, assume that there is no such rule that is preferred than \( r < r' \). Suppose that there is a rule \( r \), one if \( C \) are as above. We say another rule \( W_{r'} \) is the set of the generating rules that are used at stage \( i \).

Definition 17
Let \( (\Pi, <) \) be a statically ordered logic program and \( r \) be a rule in \( \Pi \). \( X_i \) and \( X_i^\prime \) are as above. We say another rule \( r^\prime \) is a w-preventer of \( r \) in the context \( (X, X_i) \) if the following conditions are satisfied:

1. \( r < r^\prime \)
2. \( r^\prime \) is active wrt \( (X, X_i) \) and \( \text{head}(r') \notin X_i \).

Lemma 7.6
Let \( (\Pi, <) \) be a statically ordered logic program and \( X \) a set of literals with \( C^w_{(\Pi, <)}(X) = X \). Then, for any \( r \in \Gamma_\Pi X \), there exists a number \( i \) such that \( r \in ugr(X_i) \).

The intuition behind this lemma is that each w-preventer of a rule in \( \Gamma_\Pi X \) is a “temporary” one if \( C^w_{(\Pi, <)}(X) = X \).

Proof
On the contrary, suppose that there is a rule \( r \in \Gamma_\Pi X \) such that \( r \notin ugr(X_i) \) for any \( i \). Without loss of generality, assume that there is no such rule that is preferred than \( r \). Since \( r \in \Gamma_\Pi X \) and \( X = \bigcup_{t \geq 0} X_t \), \( r \) will become active wrt \( (X_t, X) \) at some stage \( t \geq 0 \). Therefore, it must be the case that there is a w-preventer \( r^\prime \) satisfying \( r^\prime \in \Gamma_\Pi X \). This implies that \( r < r^\prime \) and \( r^\prime \in \Gamma_\Pi X \) but \( r^\prime \notin ugr(X_i) \) for any \( i \), contradiction to our assumption on \( r \). Thus, the lemma is proven.
Lemma 7.7
Let $(\Pi, \triangleleft)$ be a statically ordered logic program and $X$ a set of literals. Then $X$ is a $<^w$-preserving answer set of $\Pi$ if and only if $X$ is a set of literals with $C_{\Pi, \triangleleft}^w(X) = X$.

Proof
Without loss of generality, assume that $ugr(X_i) = \{r_{i1}, \ldots, r_{in_i}\}$ for $i \geq 1$.

If part Let $C_{\Pi, \triangleleft}^w(X) = X$. By Lemma 7.6, $\Gamma_\Pi X = \bigcup_{i=1}^\infty ugr(X_i)$. This means that the sequence $\Delta: (r_{11}, \ldots, r_{i1}, r_{21}, \ldots, r_{j1}, \ldots)$ is an enumeration of $\Gamma_\Pi X$. It suffices to prove that this sequence is $<^w$-preserving with respect to $X$.

We need to justify that the three conditions of $<^w$-sequence are satisfied by $\Delta$:

C1 For each $r_i \in \Delta$, either $r_i$ is active wrt $(X_t, X)$ or $\text{head}(r_i) \in X_t$ for some $t > 0$.
Thus, Condition 1 in Definition 3 is satisfied.

C2 If $r < r'$, then $r'$ is prior to $r$ in $\Delta$: Notice that $X = \bigcup_{i=1}^\infty X_i$, if a rule is active wrt $(X_t, X)$ then it is also active wrt $(X_t, X_i)$. Thus, by Definition 4, $r$ and $r'$ cannot be in the same section $ugr(X_t)$.

If C2 is not satisfied by $\Delta$, then there are two rules, say $r$ and $r'$, such that $r < r'$ but $r$ is prior to $r'$ in $\Delta$. Without loss of generality, assume that $r \in ugr(X_i)$ and $r' \in ugr(X_j)$ but $i < j$. Then $r'$ should prevent $r$ to be included in $ugr(X_i)$, which means $r \notin ugr(X_i)$, contradiction. Therefore, C2 holds.

C3 On the contrary, suppose that Condition 3 in Definition 3 is not satisfied. That is, there are two rules $r_i$ and $r'$ such that $r_i < r'$, $r' \in \Pi \setminus \Gamma_\Pi X$ and the following items hold:

1. $\text{body}^+(r') \subseteq X$.
2. $r'$ is not defeated by the set $\{\text{head}(r_j) \mid j < i\}$.
3. $\text{head}(r') \notin \{\text{head}(r_j) \mid j < i\}$.

Without loss of generality, assume that $r_i \in ugr(X_i)$, then $r'$ is not defeated by $X_{t-1}$ because $X_{t-1} \subseteq \{\text{head}(r_j) \mid j < i\}$. Thus, $r'$ is active wrt $(X, X_{t-1})$ and $r' \notin ugr(X_{t-1})$. This means that $r'$ is a $w$-preventer of $r_i$ in the context $(X, X_{t-1})$ and thus, $r_i \notin ugr(X_{t-1})$, contradiction.

Only-if part Assume that $X$ is a $<^w$-preserving answer set of $\Pi$, then there is a grounded enumeration $(r_{i})_{i \in I}$ of $\Gamma_\Pi X$ such that the three conditions in Definition 3 are all satisfied.

A set $\Delta$ of rules is discrete if there is no pair of rules $r$ and $r'$ in $\Delta$ such that $r < r'$. We define recursively a sequence of sets of rules as follows.

Define $\Delta_1$ as the largest set of $\langle r_i \rangle_{i \in I}$ satisfying the following conditions:

1. $\Delta_1$ is discrete;
2. $r_1 \in \Delta_1$;
3. $\text{body}(r) = \emptyset$ for any $r \in \Delta_1$.

Suppose that $\Delta_1$ is well-defined and $r_m$, is the last rule of $\Delta_1$, we define $\Delta_{i+1}$ as the largest section of $\langle r_i \rangle_{i \in I}$ satisfying the following conditions:

1. $\Delta_{i+1}$ is discrete;
2. $r_{m+i+1} \in \Delta_{i+1}$;
3. Either $\text{body}(r) \subseteq \{\text{head}(r') \mid r' \in \bigcup_{j=0}^{i} \Delta_j\}$ or $\text{head}(r) \in \{\text{head}(r') \mid r' \in \bigcup_{j=0}^{i} \Delta_j\}$ for any $r \in \Delta_{i+1}$.

4. Disjoint with $\bigcup_{j=0}^{i} \Delta_j$.

Denote $\bar{X}_i = \{\text{head}(r) \mid r \in \bigcup_{j=0}^{i} \Delta_j\}$. Then we observe the following fact:

If $r \in \Delta_{i+1}$ such that $\text{body}^+(r)$ is satisfied by $\bar{X}_{i-1}$ and no rule $r' \in \Delta_i$ with $r < r'$, then we can move $r$ from $\Delta_{i+1}$ to $\bar{\Delta}_i$, the resulting sequence of rules is still $<^w$-preserving.

Without loss of generality, assume that our sequence $\langle \Delta_i \rangle$ is fully transformed by the above transformation. Since $\bigcup_{i=0}^{\infty} X_i = X$, we can prove $\cup_{i=0}^{\infty} X_i = X$ by proving $\bar{X}_i = X_i$ for every $i \in I$. Thus, it suffices to prove $\Delta_i = ugr(X_i)$ for every $i \in I$. We use induction on $i$.

**Base** \( \Delta_0 = ugr(X_0) = \emptyset \).

**Step** Assume that $\Delta_i = ugr(X_i)$, we want to prove $\Delta_{i+1} = ugr(X_{i+1})$.

1. $\Delta_{i+1} \subseteq ugr(X_{i+1})$: For any $r_i \in \Delta_{i+1}$, by the condition 3 in the construction of $\Delta_{i+1}$, either $r_i$ is active wrt $X_i$ or $\text{head}(r_i) \in X_i$. If $\text{head}(r_i) \in X_i$, it is obvious that $r \in ugr(X_{i+1})$. Thus, we assume that $r_i$ is active wrt $X_i(X)$. For any $r'$ such that $r_i < r'$ and $r'$ is active wrt $(X,X_i)$ then $\text{body}^+(r') \subseteq X_i$ and $r'$ is not defeated by $X_i$. By induction, $\cup_{r \in \text{head}(r_i)} \subseteq X_i$ is not defeated by $\cup_{r \in \text{head}(r_i)}$. By Definition 8, it should be the case that $r' \in \Pi X$, which implies that $r' \in \Delta_i = ugr(X_i)$. Therefore, $r'$ is not a $w$-prevenient wrt $X_i$ in the context of $(X_i, X)$. That is, $r_i \in ugr(X_{i+1})$.

2. $ugr(X_{i+1}) \subseteq \Delta_{i+1}$: For $r \in ugr(X_{i+1})$, we claim that $r \in \Delta_{i+1}$. Otherwise, there would exist $t > i + 1$ such that $r \in \Delta_t$. Notice that, by induction assumption, $\text{body}^+(r) \subseteq X_i$ (Note that $\text{head}(r) \in X_i$ is impossible because we assume that $r \in \Delta_t$ and $t > i + 1$). Thus, it must be the case that there is at least one rule $r' \in \cup_{j=i+1}^{\infty} ugr(X_j)$ such that $r' < r$. But $r'$ is active wrt $(X, X_{i+1})$, which contradicts to $r \in ugr(X_{i+1})$. Therefore, $ugr(X_{i+1}) \subseteq \Delta_{i+1}$. \( \square \)

**Lemma 7.8**

Let $\langle \Pi, < \rangle$ be an ordered logic program over $L$ and let $X$ and $Y$ be consistent sets of literals. Then, we have that

1. if $X$ is a $<^w$-preserving answer set of $\Pi$, then there is some answer set $Y$ of $\text{T}^w(\Pi, <)$ such that $X = Y \cap L$;

2. if $Y$ is an answer set of $\text{T}^w(\Pi, <)$, then $X$ is a $<^w$-preserving.

**Proof.**

1. Let $X$ be a $<^w$-preserving answer set of $\Pi$. Define

\[
Y = \{\text{head}(r) \mid r \in \Gamma_\Pi X\} \\
\cup \{\text{ap}(n_r) \mid r \in \Gamma_\Pi X\} \cup \{\text{bl}(n_r) \mid r \notin \Gamma_\Pi X\} \\
\cup \{\text{ok}(n_r) \mid r \in \Pi\} \cup \{\text{rdy}(n_r, n_r) \mid r, r' \in \Pi\} \\
\cup \{n_r < n_r' \mid r < r'\} \cup \{\neg(n_r < n_r') \mid r \neq r'\}
\]

Notice that $L \in X$ iff $L \in Y$. We want to show that $Y = Cn(\text{T}(\Pi, <))^Y$ by two steps:
"\(\geq\)-part" For any \(s \in T^w(\Pi, \prec)\), if \(s^+ \in T(\Pi, \prec)^Y\) and \(\text{body}^+(s) \subseteq Y\), we need to prove \(\text{head}(s) \in Y\) by cases.

**Case 1** \(a_1(r) : \text{head}(r) \leftarrow \text{ap}(n_r)\). Since \(a_1(r) = a_1(r)^+\), \(a_1(r) \in T(\Pi, \prec)^Y\). If \(\text{ap}(n_r) \in Y\), then \(r \in \Gamma_\Pi X\). This implies \(\text{head}(r) \in Y\).

**Case 2** \(a_2(r) : \text{ap}(n_r) \leftarrow \text{ok}(n_r), \text{body}(r)\). If \(\text{ok}(n_r) \in Y\), \(\text{body}^+(r) \subseteq Y\) and \(\text{body}^-(r) \cap Y = \emptyset\), then \(\text{body}^+(r) \subseteq X\) and \(\text{body}^-(r) \cap X = \emptyset\). This implies that \(r \in \Gamma_\Pi X\) and thus \(\text{ap}(n_r) \in Y\).

**Case 3** \(b_1(r, L^+) : \text{bl}(n_r) \leftarrow \text{ok}(n_r), \text{not } L^+\). If \(\text{ok}(n_r) \in Y\) and \(L^+ \not\subseteq Y\), then \(L^+ \not\subseteq X\). That is, \(r \not\in \Gamma_\Pi X\) and thus \(\text{bl}(r) \in Y\).

**Case 4** \(b_2(r, L^-) : \text{bl}(n_r) \leftarrow \text{ok}(n_r), L^-\). If \(\text{ok}(n_r) \in Y\) and \(L^- \not\subseteq Y\), then \(L^- \not\subseteq X\). That is, \(r \not\in \Gamma_\Pi X\) and thus \(\text{bl}(r) \in Y\).

**Case 5** For the rest of rules in \(T^w(\Pi, \prec)\), we trivially have that \(\text{head}(s) \in Y\) whenever \(s^+ \in T(\Pi, \prec)^Y\) and \(\text{body}^+(s) \subseteq Y\).

"\(\leq\)-part" Since \(X\) is a \(\prec\)-preserving answer set of \(\Pi\), there is an enumeration \(\langle r_i \rangle_{i \in I}\) of \(\Gamma_\Pi X\) satisfying all conditions in Definition. This enumeration can be extended to an enumeration of \(\Pi\) as follows:

For any \(r \not\in \Gamma_\Pi X\), let \(r_i\) be the first rule that blocks \(r\) and \(r_j\) be the last rule s. t. \(r \prec r_j\). Then we insert \(r\) immediately after \(r_{\max(i, j)}\). For simplicity, the extended enumeration is still denoted \(\langle r_i \rangle_{i \in I}\). Obviously, this enumeration has the following property by Definition.

**Lemma 7.9**
Let \(\langle r_i \rangle_{i \in I}\) be the enumeration for \(\Pi\) defined as above. If \(r_i \prec r_j\), then \(j < i\).

For each \(r_i \in \Pi\), we define \(Y_i\) as follows:

\[
\begin{align*}
\{\text{head}(r_i), \text{ap}(n_{r_i}) | r_i \in \Gamma_\Pi X, i \in I\} & \cup \{\text{bl}(n_{r_i}) | r_i \not\in \Gamma_\Pi X, i \in I\} \\
\{\text{ok}(n_{r_i}) | i \in I\} & \cup \{\text{rdy}(n_{r_i}, n_{r_j}) | i, j \in I\} \\
\{n_r \prec n_{r_i} | r < r_i\} & \cup \{\neg(n_r \prec n_{r_i}) | r \not< r_i\}.
\end{align*}
\]

We prove \(Y_i \subseteq Cn(T(\Pi)^Y)\) by using induction on \(i\).

**Base** Consider \(r_0 \in \Pi\). Given that \(X\) is consistent, we have \(r_0 \not\prec r\) for all \(r \in \Pi\) by Definition. Thus, \(\neg(n_{r_0} \prec n_r) \in Y\) for all \(r \in \Pi\). Consequently,

\[
c_2(r_0, r)^+ : \text{rdy}(n_{r_0}, n_r) \leftarrow \in T(\Pi, \prec)^Y \text{ for all } r \in \Pi.
\]

This implies \(\text{rdy}(n_{r_0}, n_r) \in Cn(T(\Pi, \prec)^Y)\) for all \(r \in \Pi\).

Let \(\Pi = \{r_0, r_1, \ldots, r_k\}\). Since

\[
c_1(r_0) = c_1(r_0)^+ : \text{ok}(n_{r_0}), \ldots, \text{rdy}(n_{r_0}, n_{r_1}), \ldots, \text{rdy}(n_{r_0}, n_{r_k}) \in T(\Pi, \prec)^Y,
\]

thus \(\text{ok}(n_{r_0}) \in Cn(T(\Pi, \prec)^Y)\). We distinguish two cases.

**Case 1** If \(r_0 \in \Gamma_\Pi X\), we have \(\text{body}^+(r_0) = \emptyset\) by Definition, and \(\text{body}^-(r_0) \cap X = \emptyset\) which also implies \(\text{body}^-(r_0) \cap Y = \emptyset\). Thus

\[
a_2(r_0) = a_2(r_0)^+ : \text{ap}(n_{r_0}) \leftarrow \text{ok}(n_{r_0}) \in T(\Pi, \prec)^Y.
\]

Accordingly, we obtain \(\text{ap}(n_{r_0}) \in Cn(T(\Pi, \prec)^Y)\) by \(\text{ok}(n_{r_0}) \in Cn(T(\Pi, \prec)^Y)\).
Furthermore, from

\[ a_1(r_0) = a_1(r_0)^+ : \text{head}(n_{r_0}) \leftarrow \text{ap}(n_{r_0}) \in \text{T}(\Pi, <)^Y, \]  

we obtain \( \text{head}(r_0) \in Cn(\text{T}(\Pi, <)^Y) \).

**Case 2** If \( r_0 \in \Pi \setminus \Gamma_\Pi X \), we must have \( \text{body}^+(r_0) \not\subseteq X \) by Definition 8. That is, \( \text{body}^+(r_0) \not\subseteq Y \). Then, there is some \( L^+ \in \text{body}^+(r_0) \) with \( L^+ \not\subseteq X \). We also have \( L^+ \not\subseteq Y \). Therefore,

\[ b_1(r_0, L^+) = b_1(r_0, L^+_1) : \text{bl}(n_{r_0}) \leftarrow \text{ok}(n_{r_0}) \in \text{T}(\Pi, <)^Y. \]  

Since we have shown above that \( \text{ok}(n_{r_0}) \in Cn(\text{T}(\Pi, <)^Y) \), we obtain

\[ \text{bl}(n_{r_0}) \in Cn(\text{T}(\Pi, <)^Y). \]

**Step** Assume that \( Y_j \subseteq \text{T}(\Pi, <)^Y \) for all \( j < i \), we show \( Y_i \subseteq \text{T}(\Pi, <)^Y \) by cases.

- \( \text{rdy}(n_{r_i}, n_{r_j}) \in Cn(\text{T}(\Pi, <)^Y) \):
  
  If \( r_i < r_j \), then \( n_{r_i} \prec n_{r_j} \in Y \) and \( j < i \) by Lemma 2.29.
  
  By the induction assumption, either \( \text{ap}(n_{r_j}) \in Cn(\text{T}(\Pi, <)^Y) \) or \( \text{bl}(n_{r_j}) \in Cn(\text{T}(\Pi, <)^Y) \). Since \( c_3(r_i, r_j) \), \( c_4(r_i, r_j) \) are in \( \text{T}(\Pi, <)^Y \), we have
  \[ \text{rdy}(n_{r_i}, n_{r_j}) \in Cn(\text{T}(\Pi, <)^Y) \]  
  whenever \( r_i < r_j \).

  If \( r_i \not< r_j \), then \( n_{r_i} \prec n_{r_j} \in Y \) and thus
  \[ c_2(r_i, r_j)^+ : \text{rdy}(n_{r_i}, n_{r_j}) \leftarrow \text{T}(\Pi, <)^Y. \]

  Consequently, for all \( j \in I \), \( \text{rdy}(n_{r_i}, n_{r_j}) \in Cn(\text{T}(\Pi, <)^Y) \).

- \( \text{ok}(n_{r_i}) \in Cn(\text{T}(\Pi, <)^Y) \):
  
  It is obtained directly by \( c_1(r_i)^+ = c_1(r_i) \in Cn(\text{T}(\Pi, <)^Y) \).

- If \( r_i \in \Gamma_\Pi X \), then \( \{ \text{ap}(r_i), \text{head}(r_i) \} \not\subseteq Cn(\text{T}(\Pi, <)^Y) \).
  
  By Definition 8, \( \text{body}^+(r_i) \not\subseteq \{ \text{head}(r_j) \mid r_j \in \Gamma_\Pi X, j \prec i \} \) or \( \text{head}(r_i) \in \{ \text{head}(r_j) \mid r_j \in \Gamma_\Pi X, j < i \} \). By the induction assumption, \( \text{body}^+(r_i) \subseteq Cn(\text{T}(\Pi, <)^Y) \). Also, \( r_i \in \Gamma_\Pi X \) implies \( \text{body}^- (r_i) \cap X = \emptyset \). Thus \( \text{body}^- (r_i) \cap Y = \emptyset \).

  This means that
  \[ a_2(r_i) = a_2(r_i)^+ : \text{ap}(n_{r_i}) \leftarrow \text{ok}(n_{r_i}), \text{body}^+(r_i) \in \text{T}(\Pi, <)^Y. \]  

  As shown above, \( \text{ok}(n_{r_i}) \in Cn(\text{T}(\Pi, <)^Y) \). Therefore, \( \text{ap}(n_{r_i}) \in Cn(\text{T}(\Pi, <)^Y) \).

  Accordingly, we obtain \( \text{head}(r_i) \in Cn(\text{T}(\Pi, <)^Y) \) due to \( a_1(r_i)^+ \in \text{T}(\Pi, <)^Y \).

- If \( r_i \in \Pi \setminus \Gamma_\Pi X \), \( \text{bl}(n_{r_i}) \in Cn(\text{T}(\Pi, <)^Y) \): We consider three possibilities.

  1. \( \text{body}^+(r_i) \not\subseteq X \); then there is some \( L^+ \in \text{body}^+(r_i) \) with \( L^+ \not\subseteq X \).
     
     Also, \( L^+ \not\subseteq Y \). Thus,
     \[ b_1(r_i, L^+) = b_1(r_i, L^+_1) : \text{bl}(n_{r_i}) \leftarrow \text{ok}(n_{r_i}) \in \text{T}(\Pi, <)^Y. \]  

     By \( \text{ok}(n_{r_i}) \in Cn(\text{T}(\Pi, <)^Y) \), we have \( \text{bl}(n_{r_i}) \in Cn(\text{T}(\Pi, <)^Y) \).

  2. \( \text{body}^- (r_i) \cap \{ \text{head}(r_j) \mid r_j \in \Gamma_\Pi X, j < i \} \neq \emptyset \); then there is some \( L^- \in \text{body}^- (r_i) \) with \( L^- \in \{ \text{head}(r_j) \mid r_j \in \Gamma_\Pi X, j < i \} \). That is, \( L^- = \text{head}(r_j) \) for some \( r_j \in \Gamma_\Pi X \) with \( j < i \). With the induction hypothesis, we then obtain
Let \( X \), \( Y \), and \( Z \) be consistent answer sets of \( \Pi \). By induction on \( \Pi \), we have \( X \subseteq Y \subseteq Z \). Without loss of generality, let \( X = \emptyset \). Define \( r \) as a grounded \( \Pi \) such that \( X = r \), \( X \subseteq Y \), and \( Y \subseteq Z \). Define \( I \) as the set of variables \( \Pi \) and \( J \) as the set of constants \( \Pi \). For each \( r \in I \), we have \( \Pi \subseteq r \subseteq \Pi \). By the induction assumption, \( \Pi \subseteq r \subseteq \Pi \). This means \( \Pi \subseteq r \subseteq \Pi \). Therefore, \( \Pi \subseteq r \subseteq \Pi \).

**P1** \( X \) is an answer set of \( \Pi \): that is, \( Cn(\Pi^X) = X \).

1. \( Cn(\Pi^X) \subseteq X \): Let \( x \in X \) s. t. \( body^+(x) \subseteq X \) and \( body^-(x) \cap X = \emptyset \). Then \( body^+(x) \subseteq Y \) and \( \Pi \subseteq body^+(x) \). Thus, \( a_1(x) \in T(\Pi, <)^Y \) and \( \Pi \subseteq body^+(x) \). Since \( Y \) is closed under \( T(\Pi, <)^Y \), \( \Pi \subseteq body^+(x) \). This means \( \Pi \subseteq body^+(x) \). Therefore, \( \Pi \subseteq X \).

2. \( X \subseteq Cn(\Pi^X) \): Since \( X = Y \cap \mathcal{L} = (\cup_{i=0}^\infty T(\Pi, <)^Y \cap \mathcal{L} \), we need only to show by induction on \( i \) that, for \( i \geq 0 \),

\[
T(\Pi, <)^Y \cap \mathcal{L} \subseteq Cn(\Pi^X). \tag{17}
\]

**Base** It is obvious that \( T(\Pi, <)^Y \emptyset = \emptyset \).

**Step** Assume that \( T(\Pi, <)^Y \emptyset \) holds for \( i \), we want to prove \( T(\Pi, <)^Y \emptyset \) holds for \( i + 1 \).

If \( L \in T(\Pi, <)^Y \emptyset \), then there is a rule \( r \in \Pi \) s. t. \( head(r) = L, a_1(r)^+, a_2(r)^+ \in T(\Pi, <)^Y \) and \( \Pi \subseteq body^+(r) \subseteq T(\Pi, <)^Y \). This means \( \Pi \subseteq body^+(r) \). By the induction assumption, \( body^+(r) \subseteq Cn(\Pi^X) \). Together with \( body^+(r) \subseteq \Pi \subseteq \Pi \), we have \( r \in \Pi^X \) and \( \Pi \subseteq body^+(r) \subseteq Cn(\Pi^X) \). Therefore, \( X = Cn(\Pi^X) \).

**P2** \( X \) is \( \Pi \)-preserving: Since \( Y \) is a standard answer set of \( T(\Pi, <) \), there is a grounded enumeration \( \langle s_k \rangle_{k \in K} \). Induction of \( \Pi \) \( Y \). Define \( \langle r_i \rangle_{i \in I} \) as the enumeration obtained from \( \langle s_k \rangle_{k \in K} \) by

- deleting all rules apart from those of form \( a_2(r), b_1(r, L^+), b_2(r, L^-) \);
- replacing each rule of form \( a_2(r), b_1(r, L^+), b_2(r, L^-) \) by \( r \);
- removing duplicates\(^9\) by increasing \( i \).

for \( r \in \Pi \) and \( L^+ \in body^+(r), L^- \in body^-(r) \).

We justify that the sequence \( \langle r_i \rangle_{i \in I} \) satisfies the conditions in Definition\(^8\)

1. Since \( \langle s_k \rangle_{k \in K} \) is grounded, Condition 1 is satisfied.
2. If \( r_i < r_j \), we want to show \( j < i \). Since \( \Pi \) \( Y \), at least one of \( a_2(r_j), b_1(r_j, L^+), b_2(r_j, L^-) \) appears before any of \( a_2(r_i), b_1(r_i, L^+), b_2(r_i, L^-) \). Thus, \( j < i \).
3. Let \( r_i < r' \) and \( r' \in \Pi \cap \Pi X \). Suppose that \( body^+(r) \subseteq X \) and \( head(r) \notin \{head(r_k) \mid k < i\} \). Since \( body^-(r) \cap X \neq \emptyset \), there is some \( L^- \in body^-(r) \) s. t. \( L^- \in X \). Then \( L^- \in Y \). Without loss of generality, let \( L^- \) is included in \( Y \) through rule \( s_k \). Furthermore, we can assume that there is no \( k' < k_0 \) such that \( s_k' \)

\(^9\) Duplicates can only occur if a rule is blocked in multiple ways.
To prove Lemma 7.10, some preparations are in order.

Proof 19

Throughout the proofs for Theorem 19, the set $X_i$ for any $i \geq 0$ is defined as in Definition 7.10. By the definition of $E_X(\Pi, <)$, we observe the following facts:

**F1** $X$ is a standard answer set of $(\Pi, <)$ iff $X$ is a standard answer set of $E_X(\Pi, <)$.

**F2** $\bar{X}$ is a $<^b$-preserving answer set of $(\Pi, <)$ iff $X$ is a $<^b$-preserving answer set of $E_X(\Pi, <)$.

**F3** $X$ is a standard answer set of $T^b(\Pi, <)$ iff $X$ is a standard answer set of $T(E_X(\Pi, <))$.

Having the above facts, we can assume that $(\Pi, <) = E_X(\Pi, <)$. Thus, we need only to prove the following Lemma 7.10 and Lemma 7.14.

Given a statically ordered logic program $(\Pi, <)$ and a set $X$ of literals, set $X_i = (T^b)^{(\Pi, i)}(X)\emptyset$ for $i \geq 0$.

**Lemma 7.10**

Let $(\Pi, <)$ be a statically ordered logic program over $L$ and let $X$ be an answer set of $\Pi$. Then, the following propositions are equivalent.

1. $X$ is a $b$-preferred answer set of $(\Pi, <)$.
2. $C^{(\Pi, <)}_{\bar{X}}(X) = X$.

To prove Lemma 7.10 some preparations are in order.

**Definition 18**

Let $(\Pi, <) = (r_1, r_2, \ldots, r_n)$ be a totally ordered logic program, where $r_i < r_j$ for each $i$, and let $X$ be a set of literals.

We define

\[
\begin{align*}
\bar{X}_0 & = \emptyset \\
\bar{X}_{i+1} & = \bar{X}_i \cup \{\text{head}(r_i)\} \quad \text{and for } i \geq 0
\end{align*}
\]

\[
\begin{align*}
&\begin{cases}
(1) \ r_i+1 \text{ is active wrt } (X, X) \text{ and there is no rule } r' \in \Pi \text{ with } r_{i+1} < r' \\
(b) \ \text{head}(r') \not\in \bar{X}
\end{cases} \quad \text{such that}
\end{align*}
\]

Then, $D_{(\Pi, <)}(X) = \bigcup_{i \geq 0} \bar{X}_i$ if $\bigcup_{i \geq 0} \bar{X}_i$ is consistent. Otherwise, $D_{(\Pi, <)}(X) = \text{Lit}$. If we want to stress that $\bar{X}_i$ is for ordering $<$, we will also write it as $\bar{X}_i^<$. We assume the same notation for $X_i$.

**Lemma 7.11**

Let $(\Pi, <)$ be an ordered logic program. $\bar{X}_i$ for $i \geq 0$ is given as above and $\Pi$ is prerequisite-free. Then $X_i = \bar{X}_{k_i}$ for some non-decreasing sequence $\{k_i\}_{i \geq 0}$ with $0 \leq k_1 \leq \cdots \leq k_i \leq \cdots$. 

Proof 18

Similar to Proof 8.

Proof 17

See the proof of Theorem 19.
Proof 7.11
Without loss of generality, assume that \( \bar{X}_0 = \cdots = \bar{X}_{k_1}, \bar{X}_{k_1+1} = \cdots = \bar{X}_{k_2}, \ldots \). Then by a simple induction on \( i \), we can directly prove that
\[
X_0 = \bar{X}_0, X_1 = \bar{X}_{k_1+1}, \ldots, X_i = \bar{X}_{k_i+1}, \ldots
\]
\( \square \)

Lemma 7.12
The conclusion of Lemma 7.11 is correct for ordered logic program \( (\Pi, <) \) if \( \Pi \) is prerequisite-free and \( < \) is total.

Proof 7.12
Since \( \Pi \) is prerequisite-free, we have that \( \Pi_X = \Pi \). By Lemma 7.11 it is enough to prove that \( X = \cup \bar{X}_i \) iff \( X = \cup X_i \) (see Definition 9). For simplicity, we say a rule \( r \) is applicable wrt \( (X, \bar{X}) \) (only in this proof) if \( r \) satisfies the conditions in the definition of \( \bar{X}_{i+1} \).

if part  If \( X = \cup \bar{X}_i \), we want to prove that \( X = \cup X_i \). It suffices to show that \( \bar{X}_i = X_i \) hold for all \( i \geq 0 \). We use induction on \( i \geq 0 \):

Base  \( \bar{X}_0 = X_0 = \emptyset \).

Step  Assume that \( \bar{X}_{i-1} = X_{i-1} \), we need to show that \( \bar{X}_i = X_i \).

1. \( \bar{X}_i \subseteq X_i \):
   - If \( \bar{X}_i = X_{i-1} \), the inclusion follows from the induction assumption;
   - If \( \bar{X}_i \neq X_{i-1} \), then \( r_i \) is applicable wrt \( (X, \bar{X}_{i-1}) \).
   - Thus, \( r_i \) is not defeated by \( X \) by Definition 13.

2. \( X_i \subseteq \bar{X}_i \):
   - If \( X_i = X_{i-1} \), the inclusion follows from the induction assumption;
   - Let \( X_i \neq X_{i-1} \), that is, \( \text{head}(r_i) \notin X_i \). Then we can assert that \( \text{head}(r_i) \notin \bar{X}_i \).
   - Otherwise, if \( \text{head}(r_i) \notin \bar{X}_i \), there will be two possible cases because \( \Pi \) is prerequisite-free:
     - \( r_i \) is not active wrt \( (X, X) \): then there exists a literal \( l \in \text{body}^{-}(r_i) \) such that \( l \notin X \). On the other hand, since \( \text{head}(r_i) \notin X_i \), \( r_i \) is not defeated by \( X_{i-1} = \bar{X}_{i-1} \), so we have \( l \notin \bar{X}_{i-1} \). This implies that there exists \( t \leq i \) such that \( l \in \bar{X}_t \setminus \bar{X}_{t-1} \). Thus, \( l = \text{head}(r_t) \) and \( r_t < r_i \). Notice that \( r_t \) is active wrt \( (X, \bar{X}_{t-1}) = (X, \bar{X}_{i-1}) \) and \( \text{head}(r_t) \notin X \), thus \( r_t \) is active wrt \( (X, \bar{X}_{t-1}) \) and \( \text{head}(r_t) \notin \bar{X}_{t-1} \). This implies that \( r_t \) is a preventer of \( r_i \). Therefore, \( \text{head}(r_t) \notin \bar{X}_t \) and so by \( X = \cup \bar{X}_i \), \( \text{head}(r_t) \notin X \), contradiction.
     - There is a rule \( r' \in \Pi \) with \( r_t < r' \) such that \( r' \) is active wrt \( (X, \bar{X}_{i-1}) \) and \( \text{head}(r') \notin X \). Since there are only a finite number of rules in \( \Pi \) which are preferred over \( r_t \), so this case is impossible.

Combining the two cases, we have \( X_{i+1} \subseteq \bar{X}_{i+1} \). Thus, \( X_i = \bar{X}_i \) for all \( i \geq 0 \).

only-if part  Suppose that \( X = \cup X_i \) and \( X \) is an answer set of \( \Pi \), we want to prove that
\[
X = \cup \bar{X}_i
\]

1. We prove \( \bar{X}_i \subseteq X \) by using induction on \( i \).
   - Base  \( \bar{X}_0 = \emptyset \subseteq X \).
Step Assume that $X_i \subseteq X$. If $\text{head}(r_{i+1}) \in X_{i+1}$, then $r_{i+1}$ is not defeated by $X$ and thus not defeated by $X_i$. Thus $\text{head}(r_{i+1}) \in X_{i+1}$.

2. $X \subseteq \cup \check{X}_i$: it is sufficient to show that $X_i \subseteq \check{X}_i$ by using induction on $i$.

Base $X_0 = \emptyset = \check{X}_0$.

Step Assume that $X_k \subseteq \check{X}_k$ for $k \leq i$, then we claim that $\check{X}_i = X_i$. On the contrary, assume that $\text{head}(r_{i+1}) \in X_{i+1} \setminus \check{X}_{i+1}$. From $X = \cup X_i$, we have $\text{head}(r_{i+1}) \in X$. Notice that $X$ is an answer set of $\Pi$, so we can further assume that $r_{i+1}$ is active wrt $(X, X)$. Therefore, $\text{head}(r_{i+1}) \notin X_{i+1}$ implies that there is a number $t \leq i$ such that $r_t$ is active wrt $(X, X_t)$ but $\text{head}(r_t) \notin \check{X}_i$. Thus, $r_t$ is active wrt $(X, X_{i-1})$ by induction. This forces $\text{head}(r_t) \in X$ and $r_t$ is not active wrt $(X, X)$, contradiction.

Lemma 7.13
The conclusion of Lemma [A4] is correct for ordered logic program $(\Pi, <)$ if $\Pi$ is prerequisite-free and $<$ is a partial ordering.

Proof [A3]

if part Suppose that $X$ is an answer set of $\Pi$ and $X = \cup \check{X}_i$. Let $<_t$ be any total ordering on $\Pi$ satisfying the following three conditions:

1. If $r < r'$ then $r <_t r'$; and
2. If $r$ and $r'$ are unrelated wrt $<$ two rules and they are applied in producing $\check{X}_i$ and $\check{X}_j$ respectively ($i < j$), then $r' <_t r$.
3. If
   - $r$ is active wrt $(X, X)$ and
   - $r'$ is active wrt $(X, \check{X}_i)$ with $\text{head}(r') \notin \check{X}_i$ for some $i$ and
   - $r$ and $r'$ are unrelated wrt $<$,
   
   then $r' <_t r$.

Notice that the above total ordering $<_t$ exists. We want to prove that $X = \cup \check{X}_i <_t$. By the condition (3) above, there will be no new preventer in $(\Pi, <_t)$ for any rule $r$ though there may be more rules that are preferred than $r$. Thus, $\cup \check{X}_i <_t = \cup \check{X}_i <$. That is, $X = \cup \check{X}_i <$. Since $<_t$ is a total ordering, $X = \cup X_i <$. Therefore, $X$ is a BE-preferred answer set of $(\Pi, <)$.

only-if part Suppose that $X$ is a BE-preferred answer set of $(\Pi, <)$, then there is a total ordering $<_t$ such that $X = \cup X_i <_t$. By Lemma [A5] $X = \cup X_i <$. We want to prove that $\cup \check{X}_i <_t = \cup \check{X}_i <$. On the contrary, assume that this is not true. Then $\cup \check{X}_i <_t \subseteq \cup \check{X}_i <$. That is, there is a rule $r \in \Pi$ such that $\text{head}(r) \notin \cup \check{X}_i <_t = X$ but $r$ is active wrt $(X, X)$. On the other hand, since $X$ is an answer set of $\Pi$, $\text{head}(r) \in X$, contradiction. Therefore, $X = \cup \check{X}_i <$. $\Box$
Proof 7.10
If \( \Pi \) is transformed into \( E_X(\Pi) \), then \( \Pi \) may be performed two kinds of transformations:

1. Deleting every rule having prerequisite \( l \) such that \( l \in X \): this kind of rule can be neither active wrt \((X, X)\) nor a preventer of another rule because it is not active wrt \((X, X_i)\) for any \( i \geq 0 \).

2. Removing from each remaining rule \( r \) all prerequisites.

Suppose that \( r \) is changed into \( r' \) by this transformation. Then
- \( r \) is active wrt \((X, X)\) iff \( r' \) is active wrt \((X, X)\);
- \( r \) is a preventer in \((\Pi, \prec)\) iff \( r' \) is a preventer in \( E_X(\Pi), \prec)\).

By Lemma 7.13, Lemma 7.10 is proven. □

Lemma 7.14
Let \((\Pi, \prec)\) be a statically ordered logic program over \( L \) and let \( X \) be an answer set of \( \Pi \). Then \( X \) satisfies the Brewka/Eiter criterion for \( \Pi \) (or equivalently for \( E_X(\Pi) \)) according to [Brewka and Eiter 1999] if and only if \( X \) is a \( \prec_\mathbf{B} \)-preserving answer set of \( \Pi \).

To prove this theorem, the following result given in [Brewka and Eiter 1999] is required.

Lemma 7.15
Let \((\Pi, \prec)\) be a statically ordered logic program over \( L \) and let \( X \) be an answer set of \( \Pi \). Then \( X \) is a \( \mathbf{B} \)-preferred answer set if and only if, for each rule \( r \in \Pi \) with \( \text{body}^+(r) \subseteq X \) and \( \text{head}(r) \notin X \), there is a rule \( r' \in \Gamma_{\Pi}X \) such that \( r \prec r' \) and \( \text{head}(r') \notin \text{body}^-(r) \).

Proof 7.14

if part Let \( X \) be a \( \prec_\mathbf{B} \)-preserving answer set of \( \Pi \).

Assume that \( X \) is not a \( \mathbf{B} \)-preferred answer set, by Lemma 7.15 then there is a rule \( r \in \Pi \) such that the followings hold:

1. \( \text{body}^+(r) \subseteq X \);
2. \( \text{head}(r) \notin X \) and
3. For any rule \( r' \in \Gamma_{\Pi}X \) with \( r \prec r' \), \( \text{head}(r') \) does not defeat \( r \).

Then, \( \text{head}(r') \notin \text{body}^-(r) \). Thus \( r' \in \Pi \setminus \Gamma_{\Pi}X \). This contradict to the Condition 2 in Definition 15. Therefore, \( X \) is a \( \mathbf{B} \)-preferred answer set of \( \Pi \).

only-if part Suppose that \( X \) is a \( \mathbf{B} \)-preferred answer set of \( \Pi \). Then \( X \) is also a \( \mathbf{B} \)-preferred answer set of \((\Pi, \prec')\) where \( \prec' \) is a total ordering and compatible with \( \prec \). Notice that the ordering \( \prec' \) actually determines an enumeration \( \langle r_i \rangle_{i \in I} \) of \( \Gamma_{\Pi}X \) such that \( r_i \prec' r_j \) if \( j < i \). Thus, this enumeration of \( \Gamma_{\Pi}X \) obviously satisfies the condition 1 in Definition 15.

We prove the Condition 2 is also be satisfied. Let \( r_i \prec r' \) and \( r' \notin \Pi \setminus \Gamma_{\Pi}X \). Suppose that \( \text{body}^+(r') \subseteq X \) and \( \text{head}(r') \notin X \). By Lemma 7.15 there is a rule \( r_j \in \Gamma_{\Pi}X \) such that \( r' \prec r_j \) and \( \text{head}(r_j) \in \text{body}^-(r') \). Thus, the Condition 2 is satisfied. □
Proof 20

Under the assumption of the theorem, we can see that $T^D_\Pi, <, Y \cdot X = T^W_\Pi, <, Y \cdot X$ for any sets $X$ and $Y$ of literals, which implies $C^D_\Pi(X) = C^W_\Pi(X)$ for any set $X$ of literals. Thus, the conclusion is obtained by Theorem 14. □

Proof 21

By comparing Condition II(b) in Definition 1 and 6, we get $T^D_\Pi, <, Y \cdot X \subseteq (T^B_\Pi, <, Y \cdot X$. This means $C^D_\Pi(X) \subseteq C^B_\Pi(X)$. If $X$ is a $D$-preferred answer set of $(\Pi, <)$, it follows from Theorem 14 that $C^D_\Pi(X) = X$. Thus, $X \subseteq C^B_\Pi(X)$. On the other hand, since a $D$-preferred answer set is also a standard answer set, we have $C^B_\Pi(X) \subseteq C_\Pi X = X$. Therefore, $X = C^B_\Pi(X)$. □

Proof 22

By comparing Condition I in Definition 1 and 13, we get $(T^B_\Pi, <, Y \cdot X \subseteq T^D_\Pi, <, Y \cdot X$. This means $C^B_\Pi(X) \subseteq C^D_\Pi(X)$. If $X$ is a $W$-preferred answer set of $(\Pi, <)$, then $X = C^W_\Pi(X)$. Thus, $X \subseteq C^B_\Pi(X)$. On the other hand, since $X$ is also a standard answer set, we have $C^B_\Pi(X) \subseteq C_\Pi X = X$. Therefore, $X = C^B_\Pi(X)$. □

Proof 23

It follows directly from Theorem 21 and 22. □

Proof 24

By Theorem 13, the ordered program $(\Pi, <, s)$ has the unique $D$-preferred answer set $X^\star$. Since $s \subseteq <, X^\star$ is also a $D$-preferred answer set of $(\Pi, <)$. On the other hand, each stratified logic program has the unique answer set (the perfect model), i.e., $\mathcal{AS}(\Pi) = \{X^\star\}$. By Theorem 23 we arrive at the conclusion of Theorem 24. □

References


