Characterizing Notions of Strong Equivalence for Logic Programs with Ordered Disjunctions*

Wolfgang Faber
Department of Mathematics
University of Calabria
Via P. Bucci, cubo 30B
87036 Rende (CS), Italy
wf@wfaber.com

Hans Tompits
Institute for Information Systems 184/3
Technische Universität Wien
Favoritenstrasse 9-11
A-1040 Vienna, Austria
tompits@kr.tuwien.ac.at

Stefan Woltran
Institute for Information Systems 184/2
Technische Universität Wien
Favoritenstrasse 9-11
A-1040 Vienna, Austria
woltran@dbai.tuwien.ac.at

ABSTRACT

Ordered disjunctions have recently been introduced as a simple, yet expressive approach for representing preferential knowledge by means of logic programs. The semantics for the resulting language is based on the answer-set semantics, but comes in different flavors, depending on the particular notion of preference associated to the disjunction connective. While in standard answer-set programming, the question of when a program is to be considered equivalent to another received increasing attention in recent years, this problem has not been addressed for programs with ordered disjunctions so far. In this paper, we discuss the concept of strong equivalence in the latter setting. We introduce different versions of strong equivalence for programs with ordered disjunctions and provide model-theoretic characterizations, extending well-known ones for strong equivalence between ordinary logic programs. Furthermore, we discuss interesting relations between the proposed notions.

1. INTRODUCTION

During the last decade, *answer-set programming* (ASP) has become an increasingly acknowledged tool for declarative knowledge representation and reasoning [6, 8, 9, 1]. A main advantage of ASP is that it is based on solid theoretical foundations while being able to model commonsense reasoning in an arguably satisfactory way. The availability of efficient solvers has furthermore stimulated its use in practical applications in recent years. This development had quite some implications on ASP research. For example, increasingly large applications require features for modular programming. Another requirement is the fact that in applications, ASP code is often generated automatically by so-called *frontends*, calling for optimization methods which remove redundancies, as also found in database query optimizers. For these purposes, the fairly recently suggested notion of *strong equivalence* for ASP [7, 10] can be used. Intuitively, two programs P and Q are strongly equivalent iff, for any program R, $P \cup R$ and $Q \cup R$ have the same answer sets. To put it another way, two ASP programs are strongly equivalent if they can be used interchangeably in any context (accordingly, the program R above is also referred to as the *context program*). This gives a handle on showing the equivalence of ASP modules. Moreover, if a program is strongly equivalent to a subprogram of itself, then one can always use the subprogram instead of the original program, yielding potential for optimization.

Among the different lines of ASP research, many extensions of the basic formalism have been proposed—an important one is the modelling of preferences in ASP [4]. Strongly rooted in the research of nonmonotonic formalisms, the ability to specify preferences is acknowledged to be particularly beneficial to ASP, since they constitute a very natural and effective way of resolving indeterminate solutions. A fairly recent means of representing preferences is ASP with *ordered disjunctions* [2, 3]. The basic idea is to augment the syntax by a designated operator "$\times$" to form ordered disjunctions. Programs of this form (called *logic programs with ordered disjunctions*, or LPODs) can be evaluated in a standard way or with respect to different preferential semantics which take the occurrences of this new operator into account.

In this paper, we examine how the inclusion of preferences in the form of ordered disjunctions affects equivalence, and in particular strong equivalence of ASP. To this end, we introduce different notions of strong equivalence for LPODs. The distinguishing aspects of these notions are (i) whether the context programs are arbitrary LPODs or just ordinary programs, i.e., whether the context may change preferential information, and (ii) whether the semantics is taken in terms of standard answer sets or preferred answer sets. Following Brewka et al. [3], we study three different preference strategies, viz. Pareto-, inclusion-, and cardinality-based relations (identified using the letters p, i, and c, respectively). More formally, we introduce the following relations: for all LPODs P and Q,

- $P \equiv_s Q$ iff the standard answer sets of P and Q coincide under any extension by normal programs;
- $P \equiv_{s,\times} Q$ iff the standard answer sets of P and Q

*This work was partially supported by the Austrian Science Fund (FWF) under grant P18019 and by M.I.U.R. within project "Sistemi basati sulla logica per la rappresentazione di conoscenza: estensioni e tecniche di ottimizzazione.".

VLDB '07, September 23-28, 2007, Vienna, Austria.

coincide under any extension by LPODs;

- $P \equiv_s^\kappa Q$ iff the κ-preferred answer sets of P and Q coincide under any extension by normal programs (for $\kappa \in \{p, i, c\}$); and
- $P \equiv_{s,\times}^\kappa Q$ iff the κ-preferred answer sets of P and Q coincide under any extension by LPODs (for $\kappa \in \{p, i, c\}$).

For all of these notions we provide novel model-theoretic characterizations and we discuss relations among them. Furthermore, the notions coincide for ordinary programs, so they properly generalize the usual concept of strong equivalence. Interestingly, the two relations $\equiv_s$ and $\equiv_{s,\times}$ coincide, and they can be characterized in a similar fashion as strong equivalence for ordinary programs. That is to say, this characterization uses a generalization of the concept of an *SE-model* [10], based on a novel notion of a reduct, extending the usual reduct as introduced by Gelfond and Lifschitz [6].

Taking a preferential point of view, the arguably most interesting relation among the ones we study is $\equiv_{s,\times}^\kappa$ (for $\kappa \in \{p, i, c\}$) but the others have their importance too. Indeed, the model-theoretic characterization for $\equiv_{s,\times}^\kappa$ involves, besides specific conditions on the models of the involved programs, the relation $\equiv_s^\kappa$ as well.

2. PRELIMINARIES

A *logic program with ordered disjunction* (LPOD) is a finite set of rules of the form

$$p_1 \times \cdots \times p_k \leftarrow p_{k+1}, \ldots, p_m, \mathit{not}\ p_{m+1}, \ldots, \mathit{not}\ p_n, \quad (1)$$

where $1 \leq k \leq m \leq n$, and each p_i $(1 \leq i \leq n)$ is an *atom*.[1] A rule of above form is *normal* if $k = 1$ and *basic* if $k = 1$ and $m = n$. We use $\mathit{head}(r) = \{p_1, \ldots, p_k\}$ to denote the *head* of r and $\mathit{body}(r) = \{p_{k+1}, \ldots, p_m,\ \mathit{not}\ p_{m+1}, \ldots, \mathit{not}\ p_n\}$ to denote the the *body* of r. As well, $\mathit{body}^+(r)$ stands for the set $\{p_{k+1}, \ldots, p_m\}$, and $\mathit{body}^-(r)$ for $\{p_{m+1}, \ldots, p_n\}$, We will also write a rule r of form (1) as $p_1 \times \cdots \times p_k \leftarrow \mathit{body}(r)$ whenever convenient. The *j-th option* $(1 \leq j \leq k)$ of a rule of the form (1) is defined as

$$r[j] \ = \ p_j \leftarrow p_{k+1}, \ldots, p_m, \mathit{not}\ p_{m+1}, \ldots, \mathit{not}\ p_n, \mathit{not}\ p_1, \ldots, \mathit{not}\ p_{j-1}.$$

For a program P, we define $\mathit{atoms}(P) = \bigcup_{r \in P} \mathit{head}(r) \cup \mathit{body}^+(r) \cup \mathit{body}^-(r)$ and we say that a program P is *over* V if $\mathit{atoms}(P) \subseteq V$. A program is called *normal*, or simply a logic program (LP), if each rule in it is normal. A program is called *basic* if each rule in it is basic. A *split program* of an LPOD is obtained by replacing each rule by one of its options. Clearly any split program is normal, and a normal program is its unique split program. The set of all split programs of an LPOD P is denoted as $SP(P)$.

Towards the semantics of LPODs, we first define a notion of satisfaction for LPODs. Then, we recall the notion of an answer set of a normal program. The answer sets of an LPOD will be defined over the answer sets of its split programs. Finally, we introduce different types of preferences between answer sets of LPODs.

An interpretation I, i.e., a set of atoms, *satisfies* a rule r, in symbols $I \models r$, iff $I \cap \mathit{head}(r) \neq \emptyset$ whenever $\mathit{body}^+(r) \subseteq I$ and $I \cap \mathit{body}^-(r) = \emptyset$ jointly hold. An interpretation I satisfies an LPOD P, in symbols $I \models P$, iff $I \models r$, for each $r \in P$. I is then also called a (classical) model of P. Note that the concept of satisfaction is the same for programs with (unordered) disjunction.

The *reduct* [6] of a normal program P *relative to* an interpretation I is defined by $P^I = \{\mathit{head}(r) \leftarrow \mathit{body}^+(r) \mid r \in P, \mathit{body}^-(r) \cap I = \emptyset\}$. The smallest interpretation satisfying a basic program P is denoted by $Cn(P)$. Then, an interpretation I is an *answer set* of a normal program P if $Cn(P^I) = I$. The answer sets of an LPOD P are defined as the collection of all answer sets of its split programs.[2] We use $AS(P)$ for denoting the set of all answer sets of P. Hence, $AS(P) = \{I \mid \exists P' \in SP(P) : I \in AS(P')\}$.

Towards the definition of the preference relations, we introduce a more fine-grained concept of satisfaction: Let I be an interpretation and r a rule of form (1). Then, I *satisfies* r *to degree* j, in symbols $I \models_j r$, if $I \models r[j]$ and for all $1 \leq i < j$, $I \not\models r[i]$. Note that $I \models_1 r$ also holds if the body of r is not satisfied by I. Intuitively, satisfying a rule to degree 1 means that there is "no better way" to satisfy it. We also use $d_I(r)$ to denote the degree to which r is satisfied under I. Based on this concept we can identify certain parts of a program with respect to the degree they are satisfied by $P_I[k] = \{r \in P \mid I \models_k r\}$.

We now define three types of preference relations, which are defined on models of programs. Let I, J be (classical) models of a program P. Then,[3]

- $I >_P^c J$ iff there is a k such that $|P_I[k]| > |P_J[k]|$, and for all $j < k$, $|P_I[j]| = |P_J[j]|$,
- $I >_P^i J$ iff there is a k such that $P_I[k] \supset P_J[k]$, and for all $j < k$, $P_I[j] = P_J[j]$,
- $I >_P^p J$ iff there is a rule $r \in P$ such that $d_I(r) < d_J(r)$, and for no $r \in P$, $d_I(r) > d_J(r)$.

Finally, an interpretation I is a *κ-preferred answer set* of an LPOD P (for $\kappa \in \{p, i, c\}$) iff $I \in AS(P)$ and there is no $J \in AS(P)$ such that $J >_P^\kappa I$. The set of κ-preferred answer sets of an LPOD P is denoted by $AS^\kappa(P)$.

The relation $>_P^c$ is the *cardinality-based* preference relation, $>_P^i$ is the *inclusion-based* preference relation, and $>_P^p$ is the *Pareto-based* preference relation. We have, for any program P, that $I >_P^p J$ implies $I >_P^i J$, and $I >_P^i J$ implies $I >_P^c J$. Hence, $AS^c(P) \subseteq AS^i(P) \subseteq AS^p(P) \subseteq AS(P)$. There are programs for which all subset-relations are proper.

EXAMPLE 1. *Consider the following program P (we label rules with ordered disjunctions for easier reference):*

$$\begin{array}{lll} a \leftarrow \mathit{not}\ b, \mathit{not}\ c; & b \leftarrow \mathit{not}\ a, \mathit{not}\ c; & c \leftarrow \mathit{not}\ a, \mathit{not}\ b; \\ r_a : b \times d \leftarrow a; & r_b : c \times e \leftarrow a; & z \leftarrow b; \quad z \leftarrow c; \\ r_c : a \times b \times c : -z; & r_d : c \times b : -z. & \end{array}$$

The answer sets of P are $A_1 = \{a, d, e\}$, $A_2 = \{b, z\}$, and

[1] In contrast to Brewka et al. [2, 3], we do not consider strong negation for reasons of simplicity.

[2] Brewka et al. [3] provide an alternative definition of answer sets for LPODs based on a reduct. We shall introduce a different notion of a reduct for our purposes later, however.

[3] Brewka et al. [3] define these relations on answer sets rather than on models; however, this is too restrictive in our context where compared programs are inherently incomplete.

$A_3 = \{c, z\}$, *with the following rule-satisfaction degrees:*

	1	2	3
A_1	$P \setminus \{r_a, r_b\}$	$\{r_a, r_b\}$	
A_2	$P \setminus \{r_c, r_d\}$	$\{r_c, r_d\}$	
A_3	$P \setminus \{r_c\}$		$\{r_c\}$

We have that $A_3 >_P^c A_2$ *and* $A_3 >_P^c A_1$*, and therefore* $AS^c(P) = \{A_3\}$*. We also have* $A_3 >_P^i A_2$*, but* $A_3 \not>_P^i A_1$*, hence* $AS^i(P) = \{A_3, A_1\}$*. But then we have* $A_3 \not>_P^p A_2$ $(d_{A_3}(r_d) < d_{A_2}(r_d)$*, but* $d_{A_3}(r_c) > d_{A_2}(r_c))$ *and also* $A_3 \not>_P^p A_1$*, therefore* $AS^p(P) = \{A_3, A_1, A_2\}$*.*

We conclude this section by reviewing strong equivalence between normal programs. Two LPs P_1 and P_2 are *strongly equivalent* [7], denoted $P_1 \equiv_s P_2$, iff, for any LP P, $AS(P_1 \cup P) = AS(P_2 \cup P)$. As shown by Lifschitz, Pearce, and Valverde [7], strong equivalence actually amounts to equivalence in the non-classical logic of here-and-there; this characterization was adapted to logic-programming terms by Turner [10] as follows: Let P be an LP and let X, Y be sets of atoms such that $X \subseteq Y$. The pair (X, Y) is an *SE-model* of P if $Y \models P$ and $X \models P^Y$. By $SE(P)$ we denote the set of all SE-models of P. Then, for any LPs P_1 and P_2, $P_1 \equiv_s P_2$ iff $SE(P_1) = SE(P_2)$ [10].

Our goal is to define suitable extensions of strong equivalence for LPODs, and to develop similar model-theoretic characterizations. This will be done in terms of a novel extension of the notion of a reduct as defined for ordinary LPs, which is discussed next.

3. DEFINING A REDUCT FOR LPODS

As noted above, we now provide a definition of a reduct which properly extends the usual one due to Gelfond and Lifschitz [6], and which allows us to characterize answer sets of LPODS just in the same way as answer sets of LPs.

DEFINITION 1. *Let P be an LPOD and I an interpretation. Then,*

$$P^I = \{p_j \leftarrow body^+(r) \mid r \in P, I \models_j r, I \cap body^-(r) = \emptyset\} \cup \{p_k \leftarrow body^+(r) \mid r \in P, I \not\models r, head(r) = \{p_1, \ldots, p_k\}\}.$$

In other words, for a rule $r = p_1 \times \cdots \times p_k \leftarrow body(r)$, we take the positive part, $p_j \leftarrow body^+(r)$, of the j-th option of r to build the reduct P^I, in case $I \cap body^-(r) = \emptyset$ and r is satisfied to degree j by I, and if r is not satisfied by I (note that $I \cap body^-(r) = \emptyset$ thus holds as well), we take the positive part of the last option, i.e., $p_k \leftarrow body^+(r)$. Now, for a normal program P this definition coincides with the usual notion of a reduct, since, for any normal rule r, we have that $r = r[1]$, and thus $I \models r$ iff $I \models_1 r$, hence $r \in P^I$ iff $I \cap body^-(r) \cap I = \emptyset$. Thus, P^I as defined in the background for LPs is properly generalized to LPODs by Definition 1. The difference to the reduct $P_\times^I$ as defined by Brewka et al. [3] is that rules r from P with $I \not\models r$, are not necessarily present in $P_\times^I$.

We need two further technical lemmas.

LEMMA 1. *For each LPOD P and each interpretation I, $I \models P$ iff $I \models P^I$.*

PROOF. ($\Rightarrow$) From $I \models P$, we have, for each $r \in P$, $I \models r$, and thus $I \models_{r_j} r$, for some degree r_j. Thus, for each $r \in P$, we have either $I \cap body^-(r) \neq \emptyset$ or $I \models p_j \leftarrow body^+(r)$, and $I \models P^I$ follows.

($\Leftarrow$) If $I \not\models P$, then there exists an $r \in P$ such that $body^+(r) \subseteq I$ and $I \cap (head(r) \cup body^-(r)) = \emptyset$. This implies $body^+(r) \subseteq I$ and $I \cap head(r) = \emptyset$. Hence, for any $p_i \in head(r)$, $I \not\models p_i \leftarrow body^+(r)$. This holds, in particular, for the last head-element p_k. Since $p_k \leftarrow body^+(r)$ is contained in P^I, we get $I \not\models P^I$. $\square$

LEMMA 2. *Let P be an LPOD, $S \in SP(P)$, I an interpretation. Then, $I \models S$ implies $I \models P$.*

PROOF. Suppose $I \not\models P$. Hence, there exists a rule $r \in P$ such that $body^+(r) \subseteq I$ and $I \cap (head(r) \cup body^-(r)) = \emptyset$. Since $S \in SP(P)$, the j-th option of r, $r[j]$, is contained in S, for some j. Since $body^+(r[j]) = body^+(r)$, and $(head(r[j]) \cup body^-(r[j])) \subseteq (head(r) \cup body^-(r))$, we get $body^+(r[j]) \subseteq I$ and $I \cap (head(r[j]) \cup body^-(r[j])) = \emptyset$, thus $I \not\models S$. $\square$

THEOREM 3. *Let P be an LPOD and I an interpretation. Then, $I \in AS(P)$ iff $I = Cn(P^I)$.*

PROOF. ($\Leftarrow$) Assume $I = Cn(P^I)$, and consider the program S which contains, for each rule $r \in P$ of the form $p_1 \times \cdots \times p_k \leftarrow body(r)$, the j-th option, $r[j]$, of r if $I \models_j r$ for some $1 \leq j \leq k$, and the k-th option of r otherwise. By construction, $S \in SP(P)$ and $P^I = S^I$. Since $I = Cn(P^I)$ by hypothesis, we get $I = Cn(S^I)$, and thus $S \in AS(P)$.

($\Rightarrow$) From $I \in AS(P)$ we get that there exists a split program $S \in SP(P)$ such that $I = Cn(S^I)$. We show that $I = Cn(P^I)$. From $I \models S^I$, and since S is an LP, we know $I \models S$. By Lemma 2, $I \models P$, and thus, by Lemma 1, we get $I \models P^I$. It remains to show that for each $J \subset I$, $J \not\models P^I$. So, fix some $J \subset I$. We know $J \not\models S^I$, i.e. $J \not\models \{r[j]\}^I$, for some j-th option of a rule $r \in P$. Let r be of form $p_1 \times \cdots \times p_k \leftarrow body(r)$. From $J \not\models \{r[j]\}^I$, we get $I \cap body^-(r) = \emptyset$, $I \cap \{p_1, \ldots, p_{j-1}\} = \emptyset$, $body^+(r) \subseteq J$, and $p_j \notin J$. But $p_j \in I$ has to hold, otherwise $I \not\models S$. But then, we have $body^+(r) \subseteq I$, since $J \subset I$, $I \cap body^-(r) = \emptyset$, $p_j \in I$ and $I \cap \{p_1, \ldots, p_{j-1}\} = \emptyset$. Therefore, by definition, $I \models_j r$, and since $I \cap body^-(r) = \emptyset$, $p_j \leftarrow body^+(r)$ is contained in P^I. Hence, $J \not\models P^I$. $\square$

4. STRONG EQUIVALENCE FOR LPODS

The different semantics for LPODs motivate the definition of various notions for strong equivalence. First, let us introduce the straightforward generalizations of strong equivalence, which refer to (standard) answer sets of LPODs. However, we distinguish between the actual form of the context which either allows to contain all LPODs or disallows programs with ordered disjunction.

To start with, let us call two LPODs, P_1 and P_2, *equivalent*, denoted $P_1 \equiv P_2$, iff $AS(P_1) = AS(P_2)$. Strong pendants of this notion are as follows:

DEFINITION 2. *Let P and Q be two LPODs. Then, P and Q are* LPOD-strongly equivalent *(resp.,* LP-strongly equivalent*), symbolically $P \equiv_{s,\times} Q$ (resp,. $P \equiv_s Q$), iff, for any LPOD (resp., LP) R, $AS(P \cup R) = AS(Q \cup R)$.*

In order to characterize these strong-equivalence notions between LPODs, we define the notion of an SE-model for LPODs in the same way as done for LPs, but using our new notion of a reduct.

DEFINITION 3. *A pair (X, Y) of interpretations with $X \subseteq Y$ is an* SE-model *of an LPOD P iff $X \models P^Y$ and $Y \models P$. The set of all SE-models of an LPOD P is denoted by $SE(P)$.*

The next result shows that the extended concept of an SE-model characterizes both LPOD- and LP-strong equivalence. Thus, the latter two notions coincide.

THEOREM 4. *The following statements are equivalent, for all LPODs P, Q:* (1) $P \equiv_{s,\times} Q$; (2) $P \equiv_s Q$; (3) $SE(P) = SE(Q)$.

PROOF. The proof proceeds basically along the lines of the corresponding proof by Turner [10]. Recall that following Theorem 3, for any LPOD P, $I \in AS(P)$ iff $I \models P$ and no $J \subset I$ satisfies $J \models P^I$.

(1) $\Rightarrow$ (2): Follows by definition.

(2) $\Rightarrow$ (3): Suppose, without loss of generality, $(X, Y) \in SE(P) \setminus SE(Q)$.

Case 1: $X = Y$. Then, $Y \models P$ and $Y \not\models Q$. Clearly, $Y \in AS(P \cup \{y \leftarrow \mid y \in Y\})$ but $Y \notin AS(Q \cup \{y \leftarrow \mid y \in Y\})$.

Case 2: $X \subset Y$ and $(Y, Y) \in SE(P) \cap SE(Q)$. Take $R = \{x \leftarrow \mid x \in X\} \cup \{p \leftarrow q \mid p, q \in Y \setminus X\}$. Then, $Y \models Q \cup R$, and, for each $Z \subset Y$ with $X \neq Z$, $Z \not\models R^Y = R$. Since $X \not\models Q^Y$, by hypothesis $(X, Y) \notin SE(Q)$, we obtain that no $U \subset Y$ satisfies $U \models (Q \cup R)^Y$. Consequently, $Y \in AS(Q \cup R)$. On the other hand, $X \models P^Y$ by hypothesis, and $X \models R^Y$ is easily checked, since $R = R^Y$. But then, $X \models P^Y \cup R^Y = (P \cup R)^Y$. Hence, $Y \notin AS(P \cup R)$. In both cases we used an LP R to show $AS(P \cup R) \neq AS(Q \cup R)$, hence $P \not\equiv_s Q$.

(3) $\Rightarrow$ (1). Suppose there exists an LPOD R such that $AS(P \cup R) \neq AS(Q \cup R)$. Without loss of generality, assume that $Y \in AS(P \cup R) \setminus AS(Q \cup R)$. We get $Y \models P$ and $Y \models R$, and thus have two cases for $Y \notin AS(Q \cup R)$. First, $Y \not\models Q$. We immediately get $(Y, Y) \notin SE(Q)$ and are done, since $(Y, Y) \in SE(P)$ holds in view of $Y \models P$. So suppose $Y \models Q$ but some $X \subset Y$ satisfies $(Q \cup R)^Y$. Then, $X \models Q^Y$ and we obtain $(X, Y) \in SE(Q)$. On the other hand, since $X \models R^Y$, we have $X \not\models P^Y$, otherwise $X \models P^Y \cup R^Y = (P \cup R)^Y$, which contradicts the assumption $Y \in AS(P \cup R)$. From $X \not\models P^Y$, we get $(X, Y) \notin SE(P)$. $\square$

5. STRONG EQUIVALENCE FOR PREFERRED ANSWER SETS

We continue with strong-equivalence notions which take care about preferences among answer sets. Once again, we provide two different notions depending whether the context allows for ordered disjunctions or restricts to normal programs.

Like in the previous section, let us first call two LPODs P_1 and P_2 *κ-equivalent*, denoted $P_1 \equiv^\kappa P_2$, iff $AS^\kappa(P_1) = AS^\kappa(P_2)$.

DEFINITION 4. *Let P and Q be LPODs and $\kappa \in \{p, i, c\}$. Then, P and Q are* κ-LPOD-strongly equivalent *(resp., κ-LP-strongly equivalent), in symbols $P \equiv^\kappa_{s,\times} Q$ (resp,. $P \equiv^\kappa_s Q$), iff, for any LPOD (resp., LP) R, $AS^\kappa(P \cup R) = AS^\kappa(Q \cup R)$.*

5.1 Characterizations for LPs

We first address κ-LP-strong equivalence, for $\kappa \in \{p, i, c\}$. Before turning to the characterizations, we need two technical lemmas.

LEMMA 5. *Let Y, Z be models of an LPOD P over atoms V and let*

$$\begin{aligned} R^\dagger_{Y,Z} \;=\; & \{a \leftarrow not\ b; \quad b \leftarrow not\ a\} \cup \\ & \{y \leftarrow a \mid y \in Y\} \cup \{z \leftarrow b \mid z \in Z\} \cup \\ & \{w \leftarrow a, y', not\ w \mid y' \in V \setminus Y\} \cup \\ & \{w \leftarrow b, z', not\ w \mid z' \in V \setminus Z\}, \end{aligned}$$

where a, b, w are new atoms. Then, $AS(P \cup R^\dagger_{Y,Z}) = \{Y \cup \{a\}, Z \cup \{b\}\}$.

The proof is straightforward. Note, however, that the constraint-like rules using w are necessary, since the preferred answer sets of LPODs do not satisfy the anti-chain property, in general.

LEMMA 6. *Let P be an LPOD, R an LP, $\kappa \in \{p, i, c\}$, and Y, Z models of $P \cup R$. Then, $Y >^\kappa_P Z$ iff $Y >^\kappa_{P \cup R} Z$.*

PROOF. Since R is an LP, for every $r \in R$, we obtain $d_Y(r) = d_Z(r) = 1$ and $R_Y[1] = R_Z[1]$. Moreover, for any $k > 1$, it holds that $R_Y[k] = R_Z[k] = 0$, and hence $(P \cup R)_Y[k] = P_Y[k]$ and $(P \cup R)_Z[k] = P_Z[k]$.

It follows that

$$|(P \cup R)_Y[1]| = |P_Y[1]| + |R_Y[1]|$$

and

$$|(P \cup R)_Z[1]| = |P_Z[1]| + |R_Z[1]|,$$

and since $R_Y[1] = R_Z[1]$,

$$|(P \cup R)_Y[1]| > |(P \cup R)_Z[1]| \text{ iff } |P_Y[1]| > |P_Z[1]|$$

and

$$|(P \cup R)_Y[1]| = |(P \cup R)_Z[1]| \text{ iff } |P_Y[1]| = |P_Z[1]|.$$

Consequently, $Y >^c_P Z$ iff $Y >^c_{P \cup R} Z$.

Since $R_Y[1] = R_Z[1]$, also $(P \cup R)_Y[1] = P_Y[1] \cup R_Y[1] \supset (P \cup R)_Z[1] = P_Z[1] \cup R_Z[1]$ iff $P_Y[1] \supset P_Z[1]$, and $(P \cup R)_Y[1] = (P \cup R)_Z[1]$ iff $P_Y[1] = P_Z[1]$. Consequently, $Y >^i_P Z$ iff $Y >^i_{P \cup R} Z$.

Since, for any $r \in R$, $d_Y(r) = d_Z(r)$ if there is some rule $r' \in P \cup R$ such that $d_Y(r') < d_Z(r')$, we obtain $r' \in P$. Moreover, for no rule $r'' \in R$, $d_Y(r'') > d_Z(r'')$ can hold. Consequently, $Y >^p_P Z$ iff $Y >^p_{P \cup R} Z$. $\square$

We are now prepared to characterize $\equiv^\kappa_s$, for $\kappa \in \{p, i, c\}$.

DEFINITION 5. *Let P, Q LPODs and $\kappa \in \{p, i, c\}$. We say that the models of P, Q are $>^\kappa$-*equivalent *iff, for all interpretations Y, Z satisfying both P and Q, it holds that $Z >^\kappa_P Y$ iff $Z >^\kappa_Q Y$.*

THEOREM 7. *For all LPODs P, Q and $\kappa \in \{p, i, c\}$, $P \equiv^\kappa_s Q$ iff both $P \equiv_s Q$ and the models of P, Q are $>^\kappa$-equivalent.*

PROOF. ($\Leftarrow$) First, suppose $P \not\equiv_s Q$. Hence, there exist an LP R, such that, without loss of generality, $I \in AS(P \cup R) \setminus AS(Q \cup R)$. Let $U = atoms(P \cup Q \cup R)$ and consider the program

$$R' = R \cup \{w \leftarrow I \cup \{not\ a \mid a \in U \setminus I\}\} \cup \{w \leftarrow \ not\ w\},$$

where $w \notin U$. Then, $I \cup \{w\}$ is the only answer set of $P \cup R'$ and thus also κ-preferred, while $Q \cup R'$ possesses no answer set and thus, in particular, no κ-preferred answer-set. But then, since R' is an LP, $P \not\equiv^\kappa_s Q$.

Second, suppose there exist interpretations Y, Z satisfying $P \cup Q$ but, without loss of generality, $Z >^{\kappa}_{P} Y$ and $Z \not>^{\kappa}_{Q} Y$. Moreover, let $V = atoms(P \cup Q)$. By Lemma 5,

$$\{Y \cup \{a\}, Z \cup \{b\}\} = AS(P \cup R^{\dagger}_{Y,Z}) = AS(Q \cup R^{\dagger}_{Y,Z}).$$

But $Y \cup \{a\} \in AS^{\kappa}(Q \cup R^{\dagger}_{Y,Z}) \setminus AS^{\kappa}(P \cup R^{\dagger}_{Y,Z})$. In particular, $Y \cup \{a\} \notin AS^{\kappa}(P \cup R^{\dagger}_{Y,Z})$ since we have $Z >^{\kappa}_{P} Y$, and using $a \notin P$, $Z \cup \{a\} >^{\kappa}_{P} Y \cup \{a\}$. Lemma 6 then yields $Z >^{\kappa}_{P \cup R^{\dagger}_{Y,Z}} Y$. This shows $P \not\equiv^{\kappa}_{s} Q$.

($\Rightarrow$) Assume $P \not\equiv^{\kappa}_{s} Q$ and, without loss of generality, $Y \in AS^{\kappa}(P \cup R) \setminus AS^{\kappa}(Q \cup R)$ for some LP R. First, if $Y \notin AS(Q \cup R)$ we are done, since then $P \not\equiv_{s} Q$. So suppose $Y \in AS(Q \cup R)$. Then, there exists a $Z >^{\kappa}_{Q \cup R} Y$ such that $Z \in AS(Q \cup R)$. If $Z \notin AS(P \cup R)$ we are done again since this yields $P \not\equiv_{s} Q$. Hence, Y and Z are answer sets of both $P \cup R$ and $Q \cup R$. Thus, Y and Z have to satisfy both P and Q. Hence, we have that (i) $Z \not>^{\kappa}_{P \cup R} Y$ (as $Y \in AS^{\kappa}(P \cup R)$) and (ii) $Z >^{\kappa}_{Q \cup R} Y$ (as $Y \notin AS^{\kappa}(Q \cup R)$). Z and Y are models of R, and Lemma 6 tells us that (i) implies $Z \not>^{\kappa}_{P} Y$ while (ii) implies $Z >^{\kappa}_{Q} Y$. This shows that the models of P and Q are not $>^{\kappa}$-equivalent. □

5.2 Pareto Preferred LPOD-Strong Equivalence

In terms of comparison of Pareto-preferred answer sets, we only need a slight strengthening of the semantical conditions for $\equiv^{p}_{s}$ in order to capture $\equiv^{p}_{s,\times}$. The following relation is used to capture this aim.

Definition 6. *Given a program P and two models Y, Z of P, $Y \geq^{p}_{P} Z$ denotes that for all $r \in P$, $d_Y(r) \leq d_Z(r)$.*

The relation between $>^{p}$ and $\geq^{p}$ is as follows:

Lemma 8. *Given a program P and two models Y, Z of P, we have that $Y >^{p}_{P} Z$ iff jointly $Y \geq^{p}_{P} Z$ and $d_Y(r) < d_Z(r)$, for at least one $r \in P$.*

Proof. Recall that $Y >^{p}_{P} Z$ is defined as follows: There is a rule $r \in P$ such that $d_Y(r) < d_Z(r)$ and for no $r \in P$, $d_Y(r) > d_Z(r)$. We furthermore note that, for any $r \in P$, $d_Y(r) > d_Z(r)$ does not hold iff $d_Y(r) \leq d_Z(r)$ holds. The result thus follows. □

We now define $\geq^{p}$-equivalence between models in the same manner as $>^{p}$-equivalence was defined in Definition 5.

Definition 7. *We say that the models of two LPODs P and Q are* $\geq^{p}$-equivalent *iff, for all interpretations Y, Z satisfying both P and Q, it holds that $Z \geq^{p}_{P} Y$ iff $Z \geq^{p}_{Q} Y$.*

Lemma 9. *For all LPODs P and Q, if $P \equiv^{p}_{s,\times} Q$, then the models of P, Q are $\geq^{p}$-equivalent models.*

Proof. Suppose Y, Z are models of $P \cup Q$ such that, without loss of generality, $Y \geq^{p}_{P} Z$ but $Y \not\geq^{p}_{Q} Z$. Moreover, let P and Q be given over atoms V, and let $R = R^{\dagger}_{Y,Z} \cup \{a \times b\}$, with $R^{\dagger}$ as defined in Lemma 5. Then, $Y' = Y \cup \{a\}$ and $Z' = Z \cup \{b\}$ are the only answer sets of $P \cup R$ and $Q \cup R$. However, Y' is an Pareto-preferred answer set for $P \cup R$ while Z' is not, as $Y' >^{p}_{P \cup R} Z'$ is easily seen from $Y \geq^{p}_{P} Z$ and the presence of rule $a \times b$ in R. On the other hand, $Y \geq^{p}_{Q} Z$ does not hold, so there is a rule r in Q such that $d_Y(r) > d_Z(r)$. Therefore, $Y' \not\geq^{p}_{Q \cup R} Z'$ which implies $Y' \not>^{p}_{Q \cup R} Z'$, in view of Lemma 8. Thus, $Z' \in AS^{p}(Q \cup R)$, and consequently $P \not\equiv^{p}_{s,\times} Q$ does not hold. □

Lemma 10. *For all LPODs P and Q, if $P \equiv^{p}_{s} Q$ and the models of P, Q are $\geq^{p}$-equivalent, then $P \equiv^{p}_{s,\times} Q$.*

Proof. Assume $P \equiv^{p}_{s} Q$ and let R be an LPOD. From $P \equiv^{p}_{s} Q$ we obtain by Theorem 7 that $P \equiv_{s} Q$ holds, and thus $AS(P \cup R) = AS(Q \cup R)$. Now consider arbitrary $X, Y \in AS(P \cup R)$. We show $X >^{p}_{P \cup R} Y$ iff $X >^{p}_{Q \cup R} Y$. $P \equiv^{p}_{s,\times} Q$ then follows.

Assume $X >^{p}_{P \cup R} Y$. That is, there is an $r \in P \cup R$ such that $d_X(r) < d_Y(r)$ and for all $r' \in P \cup R$ it holds that $d_X(r') \leq d_Y(r')$. Since the models of P, Q are $\geq^{p}$-equivalent, for all $r' \in Q \cup R$ it holds that $d_X(r') \leq d_Y(r')$. If there is an $r \in R$ such that $d_X(r) < d_Y(r)$, we immediately obtain $X >^{p}_{Q \cup R} Y$. Otherwise, we have $X \geq^{p}_{R} Y$ and an $r \in P$ such that $d_X(r) < d_Y(r)$, and so $X >^{p}_{P} Y$. Since $P \equiv^{p}_{s} Q$ holds by hypothesis, we get from Theorem 7 that the models of P, Q are $>^{p}$-equivalent, thus $X >^{p}_{Q} Y$. Together with $X \geq^{p}_{R} Y$ we obtain $X >^{p}_{Q \cup R} Y$ as well. Using a symmetric argument, we can also show that $X >^{p}_{Q \cup R} Y$ implies $X >^{p}_{P \cup R} Y$. □

Lemma 9 and 10, together with the fact that $\equiv^{p}_{s,\times}$ implies $\equiv^{p}_{s}$, provides us with the following characterization.

Theorem 11. *For all LPODs P, Q, $P \equiv^{p}_{s,\times} Q$ iff jointly $P \equiv^{p}_{s} Q$ and the models of P, Q are $\geq^{p}$-equivalent.*

5.3 Inclusion Preferred LPOD-Strong Equivalence

We now adapt our strategy for $\equiv^{i}_{s,\times}$.

Definition 8. *We say that the models of two LPODs P and Q are* $\geq^{i}$-equivalent *iff, for all interpretations Y, Z satisfying both P and Q, it holds that $\delta_P(Y, Z) = \delta_Q(Y, Z)$, where, for every LPOD S,*

$$\delta_S(Y, Z) = \begin{cases} k & \text{if } S_Y[k] \neq S_Z[k] \text{ and} \\ & \forall j < k : S_Y[j] = S_Z[j]; \\ \infty & \text{otherwise.} \end{cases}$$

Lemma 12. *For all LPODs P and Q, if $P \equiv^{i}_{s,\times} Q$, then the models of P, Q are $\geq^{i}$-equivalent.*

Proof. Suppose the models of P and Q are not $\geq^{i}$-equivalent, witnessed by Y, Z such that, without loss of generality, $k = \delta_P(Y, Z) < \delta_Q(Y, Z)$. Moreover, let P and Q be given over atoms V and let $R = R^{\dagger}_{Y,Z} \cup R^{k}$, where

$$\begin{aligned} R^{k} = \{&v \leftarrow c_i, \mathit{not}\ v \mid 1 \leq i < k\} \cup \\ &\{r^{+} : c_1 \times \cdots \times c_{k-1} \times a \times b \mid P_Y[k] \subset P_Z[k]\} \cup \\ &\{r^{-} : c_1 \times \cdots \times c_{k-1} \times b \times a \mid P_Y[k] \not\subset P_Z[k]\}, \end{aligned}$$

where $R^{\dagger}_{Y,Z}$ is defined as in Lemma 5 and $c_1, \ldots, c_{k-1}, v$ are fresh symbols.

We first observe that $AS(P \cup R) = AS(Q \cup R) = \{Y', Z'\}$, where $Y' = Y \cup \{a\}$ and $Z' = Z \cup \{b\}$.

Suppose that $P_Y[k] \subset P_Z[k]$. Then, $d_{Y'}(r^{+}) = k$ and $d_{Z'}(r^{+}) = k + 1$, therefore neither $(P \cup R)_{Y'}[k] \subseteq (P \cup R)_{Z'}[k]$ nor $(P \cup R)_{Y'}[k] \supseteq (P \cup R)_{Z'}[k]$. Note also that $(P \cup R)_{Y'}[j] = (P \cup R)_{Z'}[j]$ for all $j < k$, and hence $AS^{i}(P \cup R) = \{Y', Z'\}$, since neither $Y' >^{i}_{P \cup R} Z'$ nor $Z' >^{i}_{P \cup R} Y'$ holds. Recall that by hypothesis $\delta_Q(Y, Z) > k$, thus $Q_Y[k] = Q_Z[k]$ but now due to r^{+}, $(Q \cup R)_{Y'}[k] \supset (Q \cup R)_{Z'}[k]$, i.e., we have $Y >^{i}_{Q \cup R} Z'$. Hence, $AS^{i}(Q \cup R) = \{Y'\}$, and thus $P \not\equiv^{i}_{s,\times} Q$.

For $P_Y[k] \supset P_Z[k]$, a symmetric argument shows that $AS^i(P \cup R) = \{Y', Z'\}$ and $AS^i(Q \cup R) = \{Z'\}$. For the remaining case, we have neither $P_Y[k] \supseteq P_Z[k]$ nor $P_Y[k] \subseteq P_Z[k]$ (since $k = \delta_P(Y, Z)$). We obtain $AS^i(P \cup R) = \{Y', Z'\}$ and $AS^i(Q \cup R) = \{Y'\}$ as in the first case. □

LEMMA 13. *For all LPODs P and Q, if $P \equiv^i_s Q$ and the models of P, Q are $\geq^i$-equivalent, then $P \equiv^i_{s,\times} Q$.*

PROOF. Assume $P \equiv^i_s Q$ and let R be an LPOD. From $P \equiv^i_s Q$, we obtain by Theorem 7 $P \equiv_s Q$, and thus $AS(P \cup R) = AS(Q \cup R)$. Now consider arbitrary $X, Y \in AS(P \cup R)$. We show $X >^i_{P \cup R} Y$ iff $X >^i_{Q \cup R} Y$. $P \equiv^i_{s,\times} Q$ then follows.

Assume $X >^i_{P \cup R} Y$. Hence, there exists a k such that $(P \cup R)_X[k] \supset (P \cup R)_Y[k]$ and, for each $j < k$, $(P \cup R)_X[j] = (P \cup R)_Y[j]$, i.e., $P_X[j] = P_Y[j]$ and $R_X[j] = R_Y[j]$. First, suppose $P_X[k] \supset P_Y[k]$ and $R_X[k] \supseteq R_Y[k]$. Since the models of P and Q are $\geq^i$-equivalent, $Q_X[k] \neq Q_Y[k]$, and for each $j < k$, $Q_X[j] = Q_Y[j]$. Thus, for each $j < k$, $(Q \cup R)_X[j] = (Q \cup R)_Y[j]$. It remains to show $(Q \cup R)_X[k] \supset (Q \cup R)_Y[k]$ to obtain $X >^i_{Q \cup R} Y$. Since $P \equiv^i_s Q$ holds by hypothesis, we get by Theorem 7 that the models of P and Q are $>^i$-equivalent. Thus, $Q_X[k] \supset Q_Y[k]$ has to hold. Together with $R_X[k] \supseteq R_Y[k]$, we obtain $(Q \cup R)_X[k] \supset (Q \cup R)_Y[k]$, as desired. Second, suppose $R_X[k] \supset R_Y[k]$ and $P_X[k] = P_Y[k]$. Hence, $\delta_P(X, Y) > k$. Since the models of P and Q are $\geq^i$-equivalent, $\delta_Q(X, Y) > k$ holds. Hence, for each $j \leq k$, $Q_X[j] = Q_Y[j]$. It follows that $(Q \cup R)_X[k] \supset (Q \cup R)_Y[k]$, as well as $(Q \cup R)_X[j] = (Q \cup R)_Y[j]$, for each $j < k$. Hence, $X >^i_{Q \cup R} Y$.

Again, using a symmetric argument shows that $X >^i_{Q \cup R} Y$ implies $X >^i_{P \cup R} Y$. □

Lemma 12 and 13, together with the fact that $\equiv^i_{s,\times}$ implies $\equiv^i_s$, provides us with the following characterization.

THEOREM 14. *For all LPODs P and Q, $P \equiv^i_{s,\times} Q$ iff jointly $P \equiv^i_s Q$ and the models of P, Q are $\geq^i$-equivalent.*

5.4 Cardinality Preferred LPOD-Strong Equivalence

The remaining equivalence notion to consider is $\equiv^c_{s,\times}$. Here, proofs turn out to be more cumbersome. This is due to the fact that also the number of rules appearing in the context program is of relevance to make an interpretation preferred over another one. Nevertheless, we can make use of similar concepts as before.

DEFINITION 9. *We say that the models of two LPODs P and Q are $\geq^c$-*equivalent *iff, for all interpretations Y, Z satisfying both P and Q, $\Delta^k_P(Y, Z) = \Delta^k_Q(Y, Z)$, for all $k > 0$, where $\Delta^k_S(Y, Z) = |S_Y[k]| - |S_Z[k]|$, for every LPOD S.*

LEMMA 15. *For all LPODs P and Q, if $P \equiv^c_{s,\times} Q$, then the models of P, Q are $\geq^c$-equivalent.*

PROOF. (SKETCH) Suppose the models of P and Q are not $\geq^c$-equivalent. So, we have two interpretations X and Y which satisfy P and Q and, without loss of generality, $\Delta^{k_0}_P(X, Y) < \Delta^{k_0}_Q(X, Y)$ for the minimal k_0 among all $k > 0$ such that $\Delta^k_P(X, Y) \neq \Delta^k_Q(X, Y)$. Let $m = max\{k \mid |P_X[k]| + |P_Y[k]| + |Q_X[k]| + |Q_Y[k]| > 0\}$ be the maximum satisfaction degree of rules in P or Q with respect to X or Y. We then construct the following program R with the intention that $AS^c(P \cup R) \supset AS^c(Q \cup R)$. Let $R = R^\dagger_{X,Y} \cup R^\ddagger_{X,Y}$, where $R^\dagger_{X,Y}$ is defined as in the proof for Theorem 7, and $R^\ddagger_{X,Y} = R^+_{X,Y} \cup R^-_{X,Y} \cup$

$$\begin{aligned}
&\{d^{i_j}_j \leftarrow a \mid 1 \leq j \leq m, 1 \leq i_j \leq |\Delta^j_P(X, Y)|\} \cup \\
&\{v \leftarrow d^{i_j}_j, b, not\ v \mid 1 \leq j \leq m, 1 \leq i_j \leq |\Delta^j_P(X, Y)|\} \cup \\
&\{e^{i_j}_j \leftarrow b \mid 1 \leq j \leq m, 1 \leq i_j \leq |\Delta^j_P(X, Y)|\} \cup \\
&\{v \leftarrow e^{i_j}_j, a, not\ v \mid 1 \leq j \leq m, 1 \leq i_j \leq |\Delta^j_P(X, Y)|\} \cup \\
&\{v \leftarrow c_j, not\ v \mid 1 \leq j \leq m\},
\end{aligned}$$

where

$$\begin{aligned}
R^+_{X,Y} = \{&c_1 \times \cdots \times c_{j-1} \times d^{i_j}_j \times c_{j+1} \times \cdots \times c_m \times b \mid \\
&1 \leq j \leq m, |P_X[j]| < |P_Y[j]|, 1 \leq i_j \leq \Delta^j_P(Y, X)\}, \\
R^-_{X,Y} = \{&c_1 \times \cdots \times c_{j-1} \times e^{i_j}_j \times c_{j+1} \times \cdots \times c_m \times a \mid \\
&1 \leq j \leq m, |P_X[j]| > |P_Y[j]|, 1 \leq i_j \leq \Delta^j_P(X, Y)\},
\end{aligned}$$

and all c_j, $d^{i_j}_j$, $e^{i_j}_j$ and v are new symbols. The reason why the rules in $R^+_{X,Y}$ and $R^-_{X,Y}$ include $d^{i_j}_j$ and $e^{i_j}_j$ rather than a and b is to guarantee that the rules differ also if $j = m = 1$.

Let $X' = X \cup \{a\} \cup \{d^{i_j}_j \mid 1 \leq j \leq m, 1 \leq i_j \leq |\Delta^j_P(X, Y)|\}$ and $Y' = Y \cup \{b\} \cup \{e^{i_j}_j \mid 1 \leq j \leq m, 1 \leq i_j \leq |\Delta^j_P(X, Y)|\}$. It is easy to see that X' and Y' satisfy both $P \cup R$ and $Q \cup R$, and that these are the only interpretations (over variables in P, Q, and R) doing so.

Intuitively, for each j ($1 \leq j \leq m$) such that X satisfies fewer rules in P to degree j than does Y, $R^+_{X,Y}$ gives rise to $|P_Y[j]| - |P_X[j]|$ rules being satisfied to degree j with respect to X'. These rules are satisfied to degree $m+1$ with respect to Y'. In this way, X' and Y' satisfy the same number of rules to degree j in $P \cup R$. Likewise, for those j such that X satisfies more rules in P to degree j than Y, $R^-_{X,Y}$ gives rise to $\Delta^j_P(X, Y)$ rules being satisfied to degree j with respect to Y', which are satisfied to degree $m + 1$ with respect to X'. So, X' and Y' satisfy the same number of rules to degree j in $P \cup R$ for all j up to (and including) m. Due to a more involved argument, X' and Y' also satisfy the same number of rules to degree $m+1$ in $P \cup R$. Concerning $Q \cup R$, however, X' and Y' cannot satisfy the same number of rules to degree k_0. This means that both X' and Y' are c-preferred answer sets of $P \cup R$, but only one of these can be a c-preferred answer set of $Q \cup R$. □

LEMMA 16. *For all LPODs P and Q, if $P \equiv^c_s Q$ and the models of P, Q are $\geq^c$-equivalent, then $P \equiv^c_{s,\times} Q$.*

PROOF. Assume $P \equiv^c_s Q$ and let R be an LPOD. From $P \equiv^c_s Q$, we obtain by Theorem 7 $P \equiv_s Q$, and thus $AS(P \cup R) = AS(Q \cup R)$. Now consider arbitrary $X, Y \in AS(P \cup R)$. We show $X >^c_{P \cup R} Y$ iff $X >^c_{Q \cup R} Y$. $P \equiv^c_{s,\times} Q$ then follows.

Assume $X >^c_{P \cup R} Y$. Hence, there exists a k such that $|(P \cup R)_X[k]| < |(P \cup R)_Y[k]|$ and, for each $j < k$, $|(P \cup R)_X[j]| = |(P \cup R)_Y[j]|$. First, observe that there may exist a $j < k$ such that $|P_X[j]| \neq |P_Y[j]|$. Then, $|P_X[j]| - |P_Y[j]| = |R_Y[j]| - |R_X[j]|$ has to hold. However, since the models of P and Q are $\geq^c$-equivalent, we obtain $|Q_X[j]| - |Q_Y[j]| = |P_X[j]| - |P_Y[j]|$ and thus $|(Q \cup R)_X[j]| = |(Q \cup R)_Y[j]|$.

To show $X >^c_{Q \cup R} Y$, it remains to derive $|(Q \cup R)_X[k]| < |(Q \cup R)_Y[k]|$, which can be shown by a similar argumentation.

Also, the other direction proceeds in a symmetric manner. □

Once more, Lemma 15 and 16, together with the fact that $\equiv^c_{s,\times}$ implies $\equiv^c_s$, shows our final main result.

THEOREM 17. *For all LPODs P and Q, we have that $P \equiv^c_{s,\times} Q$ iff jointly $P \equiv^c_s Q$ and the models of P, Q are $\geq^c$-equivalent.*

We note that in Lemma 16 and Theorem 17 it would be sufficient to use $P \equiv_s Q$ rather than $P \equiv^c_s Q$ (because $\geq^c$-equivalence implies $>^c$-equivalence), but we have used the latter for uniformity with Lemmata 10 and 13 and Theorems 11 and 14.

6. RELATIONSHIPS

We briefly discuss relationships among the various notions of equivalence. To this end, we first recall that Theorem 4 shows that $\equiv_s$ and $\equiv_{s,\times}$ coincide. Moreover, a somewhat surprising result, which is a consequence of Theorems 4 and 7 is as follows:

THEOREM 18. *For every $\kappa \in \{p, i, c\}$ and every LPOD P, Q, $P \equiv^\kappa_{s,\times} Q$ implies $P \equiv_{s,\times} Q$ (or likewise, $P \equiv_s Q$).*

PROOF. By definition, $P \equiv^\kappa_{s,\times} Q$ implies $P \equiv^\kappa_s Q$. Furthermore, by Theorem 7, $P \equiv^\kappa_s Q$ implies $P \equiv_s Q$, which, by Theorem 4, is equivalent to $P \equiv_{s,\times} Q$. □

In particular, $P \equiv^\kappa_s Q$ implies $P \equiv_s Q$, for each $\kappa \in \{p, i, c\}$. However, the converse direction is not true. The following example shows that $P \equiv_s Q$ does not even imply $P \equiv^\kappa Q$.

EXAMPLE 2. *Consider the programs*

$$\begin{aligned} P \;=\; & \{c \times a \times b; \\ & \quad a \leftarrow c;\ b \leftarrow c;\ c \leftarrow a, b;\ d \leftarrow c, not\ d\}, \\ Q \;=\; & \{c \times b \times a; \\ & \quad a \leftarrow c;\ b \leftarrow c;\ c \leftarrow a, b;\ d \leftarrow c, not\ d\}. \end{aligned}$$

The SE-models of P and Q coincide and are given by $(\{a\}, \{a\})$, $(\{b\}, \{b\})$, $(\{a\}, \{a, d\})$, $(\{a, d\}, \{a, d\})$, $(\{b\}, \{b, d\})$, $(\{b, d\}, \{b, d\}), (\{a, b, c\}, \{a, b, c, d\})$, and $(\{a, b, c, d\}, \{a, b, c, d\})$. Thus, we have $P \equiv_s Q$. Moreover, $AS(P) = AS(Q) = \{\{a\}, \{b\}\}$. But, for each $\kappa \in \{p, i, c\}$, $\{a\} >^\kappa_P \{b\}$, and, on the other hand, $\{b\} >^\kappa_Q \{a\}$, yielding $AS^p(P) = AS^i(P) = AS^c(P) = \{\{a\}\}$ and $AS^p(Q) = AS^i(Q) = AS^c(Q) = \{\{b\}\}$. Hence, $P \not\equiv^\kappa Q$.

As well, for any preference criterion $\kappa \in \{p, i, c\}$, we have that $P \equiv^\kappa_{s,\times} Q$ implies $P \equiv^\kappa_s Q$, but in neither case the converse holds. The following three examples demonstrate this.

EXAMPLE 3. *Consider the programs $P = \{r_1 : a \times b\}$ and $Q = \{r_2 : a \times a \times b\}$ (rules are labeled for easier reference). We obtain $SE(P) = SE(Q) = \{(\{a\}, \{a\}), (\{b\}, \{b\}), (\{a\}, \{a, b\}), (\{a, b\}, \{a, b\})\}$. Therefore, $P \equiv_s Q$. The satisfaction degrees of the models of P and Q are as follows:*

P	1	2
$\{a\}$	P	$\emptyset$
$\{b\}$	$\emptyset$	P
$\{a, b\}$	P	$\emptyset$

Q	1	2	3
$\{a\}$	Q	$\emptyset$	$\emptyset$
$\{b\}$	$\emptyset$	$\emptyset$	Q
$\{a, b\}$	Q	$\emptyset$	$\emptyset$

Note that $\{a\} >^c_P \{b\}$ and $\{a, b\} >^c_P \{b\}$, while no other pairs of models of P are in the relation $>^c_P$. As well, $\{a\} >^c_Q \{b\}$ and $\{a, b\} >^c_Q \{b\}$, while no other pairs of models of Q are in the relation $>^c_Q$. Therefore, P and Q are $>^c$-equivalent. By virtue of Theorem 7 we know that $P \equiv^c_s Q$.

However, P and Q are not $\geq^c$-equivalent, since

$$\Delta^2_P(\{a\}, \{b\}) = -1 \neq \Delta^2_Q(\{a\}, \{b\}) = 0.$$

Therefore, due to Theorem 17, $P \not\equiv^c_{s,\times} Q$. Indeed, consider $r_3 = b \times a$ and let $P' = P \cup \{r_3\}$ and $Q' = Q \cup \{r_3\}$. We have $AS(P') = AS(Q') = \{\{a\}, \{b\}\}$. Neither $\{a\} >^c_{P'} \{b\}$ nor $\{b\} >^c_{P'} \{a\}$ holds, and therefore $AS^c(P') = \{\{a\}, \{b\}\}$. However, $\{a\} >^c_{Q'} \{b\}$ (because $|Q'_{\{a\}}[2]| = 1 > |Q'_{\{b\}}[2]| = 0$), and therefore $AS^c(Q') = \{\{a\}\}$. $P \not\equiv^c_{s,\times} Q$ follows.

EXAMPLE 4. *Now consider programs*

$$\begin{aligned} P \;=\; & \{r_1 : c \times c \times a \times b;\ a \leftarrow c;\ b \leftarrow c;\ c \leftarrow a, b\}, \\ Q \;=\; & \{r_2 : c \times a \times b;\ a \leftarrow c;\ b \leftarrow c;\ c \leftarrow a, b\}. \end{aligned}$$

Note that we have $SE(P) = SE(Q) = \{(\{a\}, \{a\}), (\{b\}, \{b\}), (\{a, b, c\}, \{a, b, c\})\}$. Therefore, $P \equiv_s Q$. The rule satisfaction degrees of the models of P and Q are as follows:

P	1	2	3	4
$\{a\}$	$P \setminus \{r_1\}$	$\emptyset$	$\{r_1\}$	$\emptyset$
$\{b\}$	$P \setminus \{r_1\}$	$\emptyset$	$\emptyset$	$\{r_1\}$
$\{a, b, c\}$	P	$\emptyset$	$\emptyset$	$\emptyset$

Q	1	2	3
$\{a\}$	$Q \setminus \{r_2\}$	$\{r_2\}$	$\emptyset$
$\{b\}$	$Q \setminus \{r_2\}$	$\emptyset$	$\{r_2\}$
$\{a, b, c\}$	Q	$\emptyset$	$\emptyset$

We have $\{a\} >^i_P \{b\}$, $\{a, b, c\} >^i_P \{a\}$, $\{a, b, c\} >^i_P \{b\}$ and also, $\{a\} >^i_Q \{b\}$, $\{a, b, c\} >^i_Q \{a\}$, $\{a, b, c\} >^i_Q \{b\}$. Moreover, no other pairs of models are in the relations $>^i_P$, resp., $>^i_Q$. Therefore, the models of P and Q are $>^i$-equivalent. By virtue of Theorem 7, $P \equiv^i_s Q$ holds.

However, $\delta_P(a, b) = 3 \neq \delta_Q(a, b) = 2$, and so the models of P and Q are not $\geq^i$-equivalent. Therefore, due to Theorem 14, $P \not\equiv^i_{s,\times} Q$. Indeed, consider $r_3 = c \times c \times b \times a$ and let $P' = P \cup \{r_3\}$ and $Q' = Q \cup \{r_3\}$. It can be checked that that $AS^i(P') = \{\{a\}, \{b\}\}$, as neither $\{a\} >^i_{P'} \{b\}$ nor $\{b\} >^i_{P'} \{a\}$. However, $\{a\} >^i_{Q'} \{b\}$ (because $Q'_{\{a\}}[2] \supset Q'_{\{b\}}[2]$), and therefore $AS^i(Q') = \{\{a\}\}$.

EXAMPLE 5. *Our final example shows that $P \equiv^p_s Q$ does not imply $P \equiv^p_{s,\times} Q$. Consider*

$$\begin{aligned} P = \{& r_1 : d \times c \times a;\ r_2 : d \times c \times b;\ a \leftarrow not\ b; \\ & b \leftarrow not\ a;\ c \leftarrow a;\ c \leftarrow b;\ d \leftarrow a, b;\ a \leftarrow d;\ b \leftarrow d\}, \\ Q = \{& r_3 : d \times a \times c;\ r_4 : d \times b \times c;\ a \leftarrow not\ b; \\ & b \leftarrow not\ a;\ c \leftarrow a;\ c \leftarrow b;\ d \leftarrow a, b;\ a \leftarrow d;\ b \leftarrow d\}. \end{aligned}$$

We obtain $SE(P) = SE(Q) = \{(\{a, c\}, \{a, c\}), (\{b, c\}, \{b, c\}), (\{a, b, c, d\}, \{a, b, c, d\})\}$ and the following rule satisfaction degrees for the models:

P	1	2
$\{a, c\}$	$P \setminus \{r_1, r_2\}$	$\{r_1, r_2\}$
$\{b, c\}$	$P \setminus \{r_1, r_2\}$	$\{r_1, r_2\}$
$\{a, b, c, d\}$	P	$\emptyset$

Q	1	2	3
$\{a, c\}$	$Q \setminus \{r_3, r_4\}$	$\{r_3\}$	$\{r_4\}$
$\{b, c\}$	$Q \setminus \{r_3, r_4\}$	$\{r_4\}$	$\{r_3\}$
$\{a, b, c, d\}$	Q	$\emptyset$	$\emptyset$

We have $\{a,b,c,d\} >_P^p \{a,c\}$ *as well as* $\{a,b,c,d\} >_P^p \{b,c\}$, *but neither* $\{a,c\} >_P^p \{b,c\}$ *nor* $\{b,c\} >_P^p \{a,c\}$. *The same holds for* Q. *Hence,* $P \equiv_s^p Q$ *holds. However, we have* $\{a,b,c,d\} \geq_P^p \{a,c\}$ *as well as* $\{a,b,c,d\} \geq_P^p \{b,c\}$, *but now also* $\{a,c\} \geq_P^p \{b,c\}$ *and* $\{b,c\} \geq_P^p \{a,c\}$, *while neither* $\{a,c\} \geq_Q^p \{b,c\}$ *nor* $\{b,c\} \geq_Q^p \{a,c\}$ *holds. A witness for* $P \not\equiv_{s,\times}^p Q$ *is the LPOD* $R = \{e \leftarrow d, \mathit{not}\ e;\ r_5 : a \times b\}$, *where* $AS^p(P \cup R) = \{\{a,c\}\}$ *and* $AS^p(Q \cup R) = \{\{a,c\}, \{b,c\}\}$.

Finally, we state that (not surprisingly) if we compare only LPs, all strong-equivalence notions introduced in our work collapse to standard strong equivalence between normal logic programs, although preference information may be added in the context. This indeed shows that each notion is a generalization of standard strong-equivalence due to Lifschitz, Pearce and Valverde [7].

PROPOSITION 19. *For all normal programs* P, Q *and every* $\kappa \in \{p,i,c\}$, *the following statements are equivalent:* (1) $P \equiv_s Q$, (2) $P \equiv_{s,\times} Q$, (3) $P \equiv_s^\kappa Q$, (4) $P \equiv_{s,\times}^\kappa Q$.

PROOF. (1) $\Leftrightarrow$ (2): This follows by Theorem 4.

(1) $\Leftrightarrow$ (3): Note that in our case, for any model X of P and Q, we have that $P_X[1] = P$ and $Q_X[1] = Q$, and for all $k > 1$, $P_X[k] = Q_X[k] = \emptyset$. So for any two models X, Y of P and Q, $X \not>_P^c Y$, $X \not>_Q^c Y$, $X \not>_P^i Y$, and $X \not>_Q^i Y$, as $P_X[1] = P = P_Y[1]$ and $Q_X[1] = Q = Q_Y[1]$, and $Q_X[k] = P_X[k] = \emptyset = P_Y[k] = Q_Y[k]$ for all $k > 0$. Moreover, $X \not>_P^p Y$ and $X \not>_Q^p Y$ hold for any two models X, Y of P and Q, as for any rule $r \in P \cup Q$, $d_X(r) = 1 = d_Y(r)$. So, P, Q have $>^\kappa$-equivalent models ($\kappa \in \{p,i,c\}$), and thus by Theorem 7 we obtain $P \equiv_s Q$ iff $P \equiv_s^\kappa Q$.

(3) $\Leftrightarrow$ (4): For any two models X, Y of P and Q, we have $|P_X[k]| - |P_Y[k]| = 0 = |Q_X[k]| - |Q_Y[k]|$ for all $k > 0$, hence the models of P and Q are $\geq^c$-equivalent. Additionally, we obtain $\delta_P(X,Y) = \infty = \delta_Q(X,Y)$, and so the models of P and Q are $\geq^i$-equivalent. Moreover, since X and Y satisfy all rules to degree 1, $X \geq_P^p Y$ and $X \geq_Q^p Y$ both hold, so the models of P and Q are also $\geq^p$-equivalent. Therefore, by Theorems 11, 14, and 17, we get $P \equiv_s^\kappa Q$ iff $P \equiv_{s,\times}^\kappa Q$. □

7. DISCUSSION

In this paper, we discussed different notions of strong equivalence for logic programs with ordered disjunctions, extending the usual one for normal logic programs. Following Brewka et al. [2, 3], we studied Pareto, inclusion, and cardinality based preference relations and introduced corresponding equivalence notions based on these strategies. We provided model-theoretic characterizations and introduced to that end a novel notion of a reduct for LPODs, leading to a direct generalization of the well-known characterization of strong equivalence for LPs by Turner [10].

Although $\equiv_{s,\times}^\kappa$, for $\kappa \in \{p,i,c\}$, is arguably the most direct generalization of strong equivalence for normal programs, in the sense that it tests whether two LPODs have the same preferred answer sets in any context, the other notions are nonetheless relevant—in fact, our characterizations for $\equiv_{s,\times}^\kappa$ make use of these notions.

Concerning related work, to the best of our knowledge, strong equivalence with respect to programs allowing for a representation of preferences has been studied only by Faber and Konczak [5] (called "strong order equivalence"). However, the formalism studied there differs considerably from LPODs. Already syntactically, preferences are specified among rules using a construct different from rules. In LPODs, preferences are specified among atoms using an extended rule syntax. For this reason, also the semantics of the formalisms are hardly comparable. Indeed, also the characterizations of strong order equivalence obtained by Faber and Konczak [5] are quite different: For instance, the preferences expressed in two strongly order equivalent programs have to be exactly equal. This also implies that the rules upon which preferences are defined must occur in both strongly order equivalent programs. Therefore, one can never substitute a rule upon which a preference is expressed by another one without losing strong order equivalence.

Interesting issues for future work include the consideration of notions for *uniform equivalence* between LPODs, as well as analyzing the computational complexity of our notions.

8. REFERENCES

[1] C. Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, 2002.

[2] G. Brewka. Logic programming with ordered disjunction. In *Proc. AAAI'02*, pages 100–105. AAAI Press, 2002.

[3] G. Brewka, I. Niemelä, and T. Syrjänen. Logic programs with ordered disjunctions. *Computational Intelligence*, 20(2):335–357, 2004.

[4] J. P. Delgrande, T. Schaub, H. Tompits, and K. Wang. A classification and survey of preference handling approaches in nonmonotonic reasoning. *Computational Intelligence*, 20(2):308–334, 2004.

[5] W. Faber and K. Konczak. Strong order equivalence. *Annals of Mathematics and Artificial Intelligence*, 47(1–2):43–78, 2006.

[6] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In *Proc. ICLP'88*, pages 1070–1080. MIT Press, 1988.

[7] V. Lifschitz, D. Pearce, and A. Valverde. Strongly equivalent logic programs. *ACM Transactions on Computational Logic*, 2(4):526–541, 2001.

[8] V. Marek and M. Truszczyński. Stable models and an alternative logic programming paradigm. In *The Logic Programming Paradigm: A 25-Year Perspective*, pages 375–398. Springer Verlag, 1999.

[9] I. Niemelä. Logic programs with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence*, 25:241–273, 1999.

[10] H. Turner. Strong equivalence made easy: Nested expressions and weight constraints. *Theory and Practice of Logic Programming*, 3(4–5):609–622, 2003.