ELSEVIER
The Journal of Logic Programming 44 (2000) 75–100
THE JOURNAL OF LOGIC PROGRAMMING
www.elsevier.com/locate/jlpr

Abductive logic programming and disjunctive logic programming: their relationship and transferability

Chiaki Sakama[a,*], Katsumi Inoue[b]

[a] *Department of Computer and Communication Sciences, Wakayama University, Sakaedani, Wakayama 640 8510, Japan*

[b] *Department of Electrical and Electronics Engineering, Kobe University, Rokkodai, Nada-ku, Kobe 657 8501, Japan*

Received 1 May 1998; received in revised form 1 April 1999; accepted 23 September 1999

Abstract

Abductive logic programming (ALP) and disjunctive logic programming (DLP) are two different extensions of logic programming. This paper investigates the relationship between ALP and DLP from the program transformation viewpoint. It is shown that the belief set semantics of an abductive program is expressed by the answer set semantics and the possible model semantics of a disjunctive program. In converse, the possible model semantics of a disjunctive program is equivalently expressed by the belief set semantics of an abductive program, while such a transformation is generally impossible for the answer set semantics. Moreover, it is shown that abductive disjunctive programs are always reducible to disjunctive programs both under the answer set semantics and the possible model semantics. These transformations are verified from the complexity viewpoint. The results of this paper turn out that ALP and DLP are just different ways of looking at the same problem if we choose an appropriate semantics.

Keywords: Abductive logic programming; Disjunctive logic programming; Program transformation

1. Introduction

1.1. Background

Abduction is a form of commonsense reasoning in artificial intelligence (AI), and early AI systems realize abduction in first-order logic or default reasoning systems

* Corresponding author. Tel.: +81-73-457-8128; fax: +81-73-457-8128.
E-mail addresses: sakama@sys.wakayama-u.ac.jp (C. Sakama), inoue@eedept.kobe-u.ac.jp (K. Inoue).

0743-1066/00/$ - see front matter
PII: S 0 7 4 3 - 1 0 6 6 (9 9) 0 0 0 7 3 - 4

[5,31–33]. In the context of logic programming, Eshghi and Kowalski [12] introduced an *abductive framework* and provided an abductive interpretation of negation as failure. Kakas and Mancarella [22] developed the framework and introduced the notion of *generalized stable models*, which is an extension of the stable model semantics of Ref. [14], as a declarative semantics of abductive normal programs. A similar idea was independently developed by Gelfond [16]. In the paper, Gelfond introduced the *belief set semantics* which characterizes explanation-based reasoning in a knowledge system having disjunctions, default and explicit negations. The belief set semantics was lately applied by Inoue and Sakama [21] to the semantics of abductive (extended) disjunctive programs. The framework of *abductive logic programming* (ALP) was established and made popular by the survey paper [23]. Background and recent studies in the area are also reviewed in Refs. [24,25].

Logic programs were originally defined as a set of (definite) Horn clauses. It is known that Horn logic programs provide a powerful computational language. However, they can represent only definite information in the world and provide no inference mechanism for reasoning with *indefinite* information. A *disjunctive program* is a logic program possibly containing indefinite or disjunctive information in a program. A theory of disjunctive programs was firstly studied by Minker [29] in which he introduced the *minimal model semantics* and a theory of negation in *positive disjunctive programs*. Later, the minimal model semantics has been extended in various ways to disjunctive programs containing negation. Such extensions include the *disjunctive stable model semantics* [34] and the *possible model semantics* [37] of *normal disjunctive programs*, and the *answer set semantics* [15] of *extended disjunctive programs*. An intensive study on *disjunctive logic programming* (DLP) is done in the monograph [27]. Historical background and recent techniques in the area are described in Refs. [3,30].

1.2. ALP vs DLP

ALP supplies the ability to perform reasoning with hypotheses. It is known that abduction is useful for various AI problems including diagnosis, planning, and theory revision [24]. ALP enables us to use logic programming as an inference engine for solving these abductive problems. On the other hand, DLP provides us with a method of reasoning with indefinite information. DLP extends Horn logic to a larger subset of first-order logic. It has close relation to first-order theorem proving [27], and also provides a powerful database query language [11]. DLP is useful in representing knowledge having non-deterministic and multiple possible interpretations. Applications include legal rules, biological inheritance, natural language understanding, and so on [6].

Thus, ALP and DLP supply apparently different forms of reasoning and have been independently studied in logic programming. Indeed, ALP and DLP handle incomplete information in their own ways, and have different syntax and semantics from each other. Comparing these two languages, however, it is observed that there are some similarities between hypothetical knowledge in ALP and indefinite knowledge in DLP. In ALP, a set of hypotheses are distinguished as *abducibles*, then in the process of abduction each candidate hypothesis is examined whether it is adopted or not to explain the observation. For instance, consider an abduction problem such that a background knowledge has the rules:

$$wet\text{-}grass \leftarrow rained,$$
$$wet\text{-}grass \leftarrow sprinkler\text{-}on,$$

with the abducibles *rained* and *sprinkler-on*. Given the observation *wet-grass*, abduction produces either *rained* or *sprinkler-on* (or both) as a possible explanation. Here, each abducible can take alternative truth values, and the situation can be specified as meta-level disjunctive knowledge that either an abducible is true or not.

On the other hand, in DLP each disjunction is considered to represent knowledge about possible alternative beliefs and such beliefs can also be regarded as a kind of hypotheses. For instance, consider the disjunctive program:

$$\neg lh\text{-}usable \leftarrow lh\text{-}broken,$$
$$\neg rh\text{-}usable \leftarrow rh\text{-}broken,$$
$$lh\text{-}broken \mid rh\text{-}broken \leftarrow .$$

We remember a person with a broken hand, but we are not sure which one (left or right) is broken. We also know that the broken hand is not usable. Here, the disjunction represents our uncertain knowledge and each disjunct represents a possible assumption. Selecting one disjunct, we reach a different conclusion.

Thus, the two formalisms appear to deal with very similar problems from different viewpoints. Then the question naturally arises whether there is any formal correspondence between ALP and DLP. Relating different frameworks to each other helps one to better understand the semantic nature of each language. Moreover, linking different types of reasoning contributes to a unified theory of commonsense reasoning in AI. On the practical side, a transformation between different languages enables one to use a proof procedure of one language for the other. It also extends and enriches applications in both areas.

1.3. Related work

There are some studies which relate ALP and DLP. Inoue and Sakama [21] present a transformation from abductive programs to disjunctive programs and use a bottom-up model generation procedure of DLP for computing abduction. They also introduce another program transformation from abductive programs to disjunctive programs which contain negation as failure in the heads of rules [20]. Dung [7] presents a transformation from acyclic disjunctive programs to normal programs under the stable model semantics and uses Eshghi and Kowalski's abductive proof procedure [12] for such programs. You et al. [41] extend the work and generalize the EK-procedure to (propositional) DLP. Aravindan [1] characterizes negative inference in disjunctive programs through abduction. Lifschitz and Turner [26] provide a transformation from disjunctive programs to abductive programs in a particular action theory. These works provide one-way transformations or characterizations for particular purposes, and do not consider the bi-directional relationships between the two frameworks in general.

1.4. Outline of the paper

This paper is intended to investigate a general correspondence between ALP and DLP under the answer set semantics [15], the belief set semantics [16,21], and the possible model semantics [35,37]. For the part from abductive programs to disjunctive programs, we show that the belief sets of an abductive program are expressed by the answer sets and the possible models of the transformed disjunctive program. Conversely, from disjunctive programs to abductive programs, we show that the possible models of a disjunctive program are exactly the belief sets of the transformed abductive program. In contrast, the answer sets of a disjunctive program are unlikely to be expressed by the belief sets of an abductive program in general. Moreover, abductive disjunctive programs can be reduced to disjunctive programs under the answer set semantics and the possible model semantics. It is also shown that abductive disjunctive programs are reducible to abductive programs under the possible model semantics.

This paper is an extended form of Ref. [36]. The rest of this paper is organized as follows. Section 2 reviews the frameworks of ALP and DLP. Section 3 introduces program transformations from abductive programs to disjunctive programs. Section 4 presents a reverse transformation from disjunctive programs to abductive programs. In Section 5, it is shown that abductive disjunctive programs are reducible to disjunctive programs. Section 6 provides a connection between the possible models and the answer sets of a disjunctive program. Section 7 discusses the relation between ALP and DLP from the complexity viewpoint. Section 8 concludes the paper.

2. Preliminaries

2.1. Disjunctive logic programming

A program considered in this paper is an *extended disjunctive program* (EDP) [15] which is a set of rules of the form:

$$L_1 \mid \cdots \mid L_l \leftarrow L_{l+1}, \ldots, L_m, not\, L_{m+1}, \ldots, not\, L_n \quad (n \geqslant m \geqslant l \geqslant 0), \tag{1}$$

where each L_i is a positive or negative literal. '|' represents a disjunction and *not* means negation as failure. The disjunction to the left of $\leftarrow$ is the *head* and the conjunction to the right of $\leftarrow$ is the *body* of the rule. In this paper, we often use the Greek letter Γ to denote the conjunction in the body. A rule is called *disjunctive* (resp. *normal*) if its head contains more than one literal (resp. exactly one literal). A rule with the empty head is called an *integrity constraint*. A *not-free* EDP is an EDP in which each rule contains no *not*, i.e., for each rule $m = n$. An EDP is called (i) an *extended logic program* (ELP) if $l \leqslant 1$ for each rule (1); and (ii) a *normal disjunctive program* (NDP) if every L_i is an atom. An NDP is called (i) a *normal logic program* (NLP) if $l \leqslant 1$ for each rule (1); and (ii) a *positive disjunctive program* (PDP) if it contains no *not*. EDPs, NDPs, and PDPs are simply called *disjunctive programs* when such distinction is not important in the context. A program (rule, literal) is *ground* if it contains no variable. A program P is semantically identified with its ground instantiation, i.e., the set of ground rules obtained from P by substituting variables in P by elements of its Herbrand universe in every possible way. Throughout the paper, a program containing variables is considered as a shorthand of its ground instantiation.

As the semantics of EDPs, we consider the *answer set semantics* [15] and the *possible model semantics* [35,37] in this paper.

Let *Lit* be the set of all ground literals from the language of a program P. First, let P be a *not*-free EDP and $S \subset Lit$. Then, S is a *(consistent) answer set* of P if S is a minimal set satisfying the conditions:

(i) S satisfies every ground rule from P, i.e., for each ground rule

$$L_1 \mid \cdots \mid L_l \leftarrow L_{l+1}, \ldots, L_m$$

in the ground instantiation of P, $\{L_{l+1}, \ldots, L_m\} \subseteq S$ implies $\{L_1, \ldots, L_l\} \cap S \neq \emptyset$. In particular, for each integrity constraint $\leftarrow L_1, \ldots, L_m$ from P, $\{L_1, \ldots, L_m\} \not\subseteq S$;

(ii) S does not contain a pair of complementary literals L and $\neg L$.[1]

Secondly, let P be any EDP and $S \subset Lit$. The *reduct* P^S of P by S is a *not*-free EDP obtained as follows: a ground rule

$$L_1 \mid \cdots \mid L_l \leftarrow L_{l+1}, \ldots, L_m$$

is in P^S iff there is a ground rule

$$L_1 \mid \cdots \mid L_l \leftarrow L_{l+1}, \ldots, L_m, \; not\ L_{m+1}, \ldots, not\ L_n$$

from P such that $\{L_{m+1}, \ldots, L_n\} \cap S = \emptyset$. For programs of the form P^S, their answer sets have already been defined. Then, S is an *answer set* of P if S is an answer set of P^S.

An EDP has none, one or multiple answer sets in general. In NDPs answer sets coincide with (*disjunctive*) *stable models* [34], and in PDPs answer sets coincide with *minimal models*.

Next, given an EDP P, a *split program* is defined as a ground ELP obtained from P by replacing every ground disjunctive rule

$$r: \; L_1 \mid \cdots \mid L_l \leftarrow \Gamma$$

from P with the rules in $R \subseteq Split_r$, where

$$Split_r = \{ L_i \leftarrow \Gamma \;\mid\; i = 1, \ldots, l \}$$

and R is a non-empty subset of $Split_r$. Each rule in $Split_r$ is called a *split rule* of r. By the definition, P has multiple split programs in general. Then, a *possible model* of P is defined as an answer set of a split program of P. A possible model S of P is *minimal* if there is no possible model T of P such that $T \subset S$. Any answer set is a possible model but not vice versa.

Example 2.1. Let P be the program:

$$a \mid \neg b \leftarrow not\, c,$$
$$d \leftarrow a, \neg b.$$

Then P has the following three split programs:

$$P_1: a \leftarrow not\ c, \quad d \leftarrow a, \neg b, \qquad P_2: \neg b \leftarrow not\, c, \quad d \leftarrow a, \neg b, \qquad P_3: a \leftarrow not\, c, \quad \neg b \leftarrow not\, c, \quad d \leftarrow a, \neg b,$$

[1] In this paper we are interested in consistent programs and do not consider the contradictory answer set *Lit*.

and $\{a\}$, $\{\neg b\}$, $\{a, \neg b, d\}$ are the possible models of P. Here, $\{a\}$ and $\{\neg b\}$ are also answer sets of P, while $\{a, \neg b, d\}$ is not an answer set of P.

In the example, the answer sets provide exclusive interpretations of the disjunction, while the possible models provide both exclusive and inclusive interpretations. [2] Note that if one prefers exclusive interpretations under the possible model semantics, it is done by inserting the additional constraint $\leftarrow a, \neg b$ to the program. Thus, under the possible model semantics we can freely specify both inclusive and exclusive interpretations of disjunctions.

The above definition of possible models is a direct extension of those introduced in PDPs [35] and NDPs [37]. The following properties immediately follow from the results in Ref. [37].

Proposition 2.1 (Properties of possible models).
(i) *In an EDP, the answer sets are minimal possible models, but not vice versa.*
(ii) *In a not-free EDP, the answer sets coincide with the minimal possible models.*
(iii) *In an ELP, the possible models coincide with the answer sets.*

The converse of Proposition 2.1(i) does not hold in general. For a counter-example, consider the program P:

$$a \mid b \leftarrow, \quad b \leftarrow a, \quad \leftarrow not\ a.$$

Then, P has the minimal possible model $\{a, b\}$, while P has no answer set. In the program, the first disjunctive rule asserts that either a or b is true. As the third integrity constraint presents that a should be true, a is selected from the disjunction. On the other hand, a implies b by the second rule, hence it is natural to conclude both a and b to be true. This example illustrates that the answer set semantics (or equivalently the stable model semantics) often fails to provide a natural meaning of a program. The possible model semantics has some nice properties for inferring negation [35,4,37] and also has theoretical relation to autoepistemic logic [20]. Recent study [40] presents that the possible model semantics is useful for representing product configuration applications.

2.2. Abductive logic programming

An *abductive program* is defined as a pair $\Pi = \langle P, \mathscr{A} \rangle$ where P is a program and $\mathscr{A}$ is a set of literals from the language of P called *abducibles*. The set $\mathscr{A}$ is identified with the set of ground instances from $\mathscr{A}$, and any instance of an element from $\mathscr{A}$ is also called an abducible. An abductive program is called an abductive EDP (resp. ELP) when P is an EDP (resp. ELP). An abductive program is also called an abductive NDP (resp. NLP) when P is an NDP (resp. NLP) and $\mathscr{A}$ is a set of atoms. An abductive EDP (resp. ELP, NDP, NLP) is abbreviated as AEDP (resp. AELP,

[2] Note that an answer set also provides an inclusive interpretation of disjunctions in the program $\{a \mid b \leftarrow, a \leftarrow b, b \leftarrow a\}$. However, minimality-based semantics interprets disjunctions *as exclusive as possible* [41].

ANDP, ANLP). AEDPs, AELPs, ANDPs and ANLPs are simply called abductive programs when such distinction is not important in the context.

A set of literals $S(\subset Lit)$ is a *belief set* of Π (wrt E) if S is an answer set of $P \cup E$ where $E \subseteq \mathscr{A}$. A belief set S is called *minimal* if there is no belief set T such that $T \subset S$. A belief set S is called $\mathscr{A}$*-minimal* if there is no belief set T such that $T \cap \mathscr{A} \subset S \cap \mathscr{A}$. Each belief set reduces to an answer set of P when $\mathscr{A} = \emptyset$. Belief sets are called *belief models* [21] when P is either an NDP or an NLP. Belief models are also called *generalized stable models* [22] when P is an NLP.

Let $\Pi = \langle P, \mathscr{A} \rangle$ be an abductive program and O a ground literal which represents an *observation.* [3] A set E $(\subseteq \mathscr{A})$ is an *explanation* of O (wrt Π) if O is true in a belief set S of Π such that $E = S \cap \mathscr{A}$. An explanation E of O is *minimal* if no $E' \subset E$ is an explanation of O. With these settings, the problem of finding an explanation is essentially equivalent to the problem of finding a belief set. That is, E is a (minimal) explanation of O with respect to $\langle P, \mathscr{A} \rangle$ iff S is an ($\mathscr{A}$-minimal) belief set of $\langle P \cup \{\leftarrow not\, O\}, \mathscr{A} \rangle$ such that $S \cap \mathscr{A} = E$ [21].

Remark.
1. The above definition provides a *credulous* explanation. In contrast, a *skeptical explanation* is defined as a set $E \subseteq \mathscr{A}$ such that O is true in *every* belief set S of Π such that $E = S \cap \mathscr{A}$. In this paper, an explanation means a credulous explanation.
2. The condition $E = S \cap \mathscr{A}$ identifies the abducibles included in a belief set in which an observation holds. This condition is useful for computing explanations via belief sets, while the existence of this condition is not important in the transformations between ALP and DLP considered in this paper.

Example 2.2. Let $\Pi = \langle P, \mathscr{A} \rangle$ be an abductive program such that

$$\begin{aligned} P:\ & p(x) \leftarrow q(x), not\, r(x), \\ & q(x) \leftarrow s(x), \\ & q(x) \leftarrow t(x). \\ \mathscr{A}:\ & s(x),\ t(b). \end{aligned}$$

Given the observation $O = p(a)$, O is true in the ($\mathscr{A}$-minimal) belief set $S = \{p(a), q(a), s(a)\}$ of Π and $E = S \cap \mathscr{A} = \{s(a)\}$. Hence, O has the (minimal) explanation E. Here, S is also the unique ($\mathscr{A}$-minimal) belief set of $\langle P \cup \{\leftarrow not\, p(a)\}, \mathscr{A} \rangle$.

The next proposition presents that an $\mathscr{A}$-minimal belief set is always a minimal belief set.

Proposition 2.2 ($\mathscr{A}$-minimal belief sets are minimal). *Let $\Pi = \langle P, \mathscr{A} \rangle$ be an AEDP. If S is an $\mathscr{A}$-minimal belief set of Π, then S is a minimal belief set of Π.*

[3] Without loss of generality an observation is assumed to be a (non-abducible) ground literal [21].

Proof. If S is an $\mathscr{A}$-minimal belief set of Π, there is no belief set T of Π such that $T \cap \mathscr{A} \subset S \cap \mathscr{A}$. To see that S is also minimal, suppose that there is a belief set S' of Π such that $S' \subset S$. As $S' \cap \mathscr{A} \not\subset S \cap \mathscr{A}$, it holds that $S' \cap \mathscr{A} = S \cap \mathscr{A}$ and $S' \cap \overline{\mathscr{A}} \subset S \cap \overline{\mathscr{A}}$, where $\overline{\mathscr{A}} = Lit \setminus \mathscr{A}$. Put $F = S' \cap \mathscr{A} = S \cap \mathscr{A}$. Then, S and S' are answer sets of $P \cup F$. Since any answer set is minimal, S cannot be an answer set of $P \cup F$ by $S' \subset S$. Contradiction. □

The converse also holds when P is an ELP and $P \setminus I$ contains no *not*, where I is the set of integrity constraints in P.

Proposition 2.3 (Minimal belief sets are $\mathscr{A}$-minimal (sometimes)). *Let $\Pi = \langle P, \mathscr{A} \rangle$ be an AELP such that $P \setminus I$ contains no not. If S is a minimal belief set of Π, then S is an $\mathscr{A}$-minimal belief set of Π.*

Proof. Suppose that a minimal belief set S is not $\mathscr{A}$-minimal. Then, there is a belief set T of Π such that $T \cap \mathscr{A} \subset S \cap \mathscr{A}$. Let S be an answer set of $P \cup E$ where $E \subseteq \mathscr{A}$. Then, T is an answer set of $P \cup F$ such that $F \subset E$. When $P \setminus I$ contains no *not*, a belief set monotonically grows by the introduction of abducibles to the ELP P, as far as the set satisfies integrity constrains I. Since S and T satisfy I and $S \neq T$ by $T \cap \mathscr{A} \subset S \cap \mathscr{A}$, $F \subset E$ implies $T \subset S$. This contradicts the assumption that S is minimal. □

When a program contains disjunctions or negation as failure, the above proposition does not hold in general.

Example 2.3. Let $\Pi_1 = \langle P_1, \mathscr{A} \rangle$ be an abductive program such that

$$P_1 : p \leftarrow not\, a.$$
$$\mathscr{A} : a.$$

Then, $\{p\}$ and $\{a\}$ are the minimal belief sets wrt $E_1 = \emptyset$ and $E_2 = \{a\}$, respectively. But $\{a\}$ is not an $\mathscr{A}$-minimal belief set.

Next, let $\Pi_2 = \langle P_2, \mathscr{A} \rangle$ be an abductive program such that

$$P_2 : p \mid q \leftarrow,$$
$$a \leftarrow p.$$
$$\mathscr{A} : a.$$

Then, both $\{a, p\}$ and $\{q\}$ are the minimal belief sets wrt $E = \emptyset$, but $\{a, p\}$ is not an $\mathscr{A}$-minimal belief set.

3. Transforming abductive LP to disjunctive LP

3.1. From belief sets to answer sets

We first present a program transformation from abductive programs to disjunctive programs, which converts the belief sets of an AELP into the semantically equivalent answer sets of an EDP.

Given an abductive program $\Pi = \langle P, \mathscr{A} \rangle$, each abducible in $\mathscr{A}$ is either true or not in constructing a belief set. The situation is declaratively specified by a disjunctive program.

Definition 3.1 (*dlp-transformation*). Let $\Pi = \langle P, \mathscr{A} \rangle$ be an AELP. Then the *dlp-transformation* transforms Π to the EDP $dlp(\Pi)$, which is obtained from P by introducing the following disjunctive rules for each abducible $A \in \mathscr{A}$:

$$A \mid \varepsilon_A \leftarrow \tag{2}$$

where ε_A is an atom appearing nowhere in P and is uniquely associated with each A.[4]

The intuitive meaning of the *dlp*-transformation is as follows. When an abducible A is assumed in an abductive program Π, the corresponding disjunct A is chosen from (2) in the transformed disjunctive program $dlp(\Pi)$. Else when A is not assumed, the newly introduced atom ε_A is chosen from (2). Thus the *dlp*-transformation encodes meta-level knowledge representing whether each abducible is assumed or not.

Formally, this transformation converts the belief sets of an AELP Π into the semantically equivalent answer sets of an EDP $dlp(\Pi)$. For an answer set S of an EDP P, we say that S is *$\mathscr{A}$-minimal* if there is no answer set T of P such that $T \cap \mathscr{A} \subset S \cap \mathscr{A}$.

Theorem 3.1 (Belief sets of an AELP $\rightarrow$ answer sets of an EDP). *Let $\Pi = \langle P, \mathscr{A} \rangle$ be an AELP.*

(i) *If S is a belief set of Π, there is an answer set T of $dlp(\Pi)$ such that $T \cap Lit = S$. Conversely, if T is an answer set of $dlp(\Pi)$, there is a belief set S of Π such that $S = T \cap Lit$.*

(ii) *If S is an $\mathscr{A}$-minimal belief set of Π, there is an $\mathscr{A}$-minimal answer set T of $dlp(\Pi)$ such that $T \cap Lit = S$. Conversely, if T is an $\mathscr{A}$-minimal answer set of $dlp(\Pi)$, there is an $\mathscr{A}$-minimal belief set S of Π such that $S = T \cap Lit$.*

Proof. (i) When S is a belief set of Π, each abducible A in $\mathscr{A}$ is either true or not in S. Then there is an answer set T of $dlp(\Pi)$ such that $T \cap \overline{\mathscr{A}} = S \cap \overline{\mathscr{A}}$ with $\overline{\mathscr{A}} = Lit \setminus \mathscr{A}$, and for each ground disjunctive rule $A \mid \varepsilon_A \leftarrow$ from $dlp(\Pi)$, $A \in T$ iff $A \in S$, and $\varepsilon_A \in T$ iff $A \notin S$. Hence the result follows. Conversely, when T is an answer set of $dlp(\Pi)$, T contains either A or ε_A for each ground disjunctive rule $A \mid \varepsilon_A \leftarrow$ from $dlp(\Pi)$. Put $E = \{A : A \in T$ and there is a ground rule $A \mid \varepsilon_A \leftarrow$ from $dlp(\Pi)\}$. Then, $T \cap Lit$ is an answer set of $P \cup E$ where $E \subseteq \mathscr{A}$, hence $T \cap Lit = S$ is a belief set of Π.

(ii) The result follows from (i) and the definitions of $\mathscr{A}$-minimal belief sets and $\mathscr{A}$-minimal answer sets. $\square$

Using the *dlp*-transformation, abductive explanations are computed in the transformed disjunctive programs.

[4] A similar transformation is introduced in Ref. [20] using negation as failure in the head.

Corollary 3.1 (Characterization of explanations via answer sets). *Let $\Pi = \langle P, \mathscr{A} \rangle$ be an AELP. Then, an observation O has a (minimal) explanation E iff there is an ($\mathscr{A}$-minimal) answer set S of $dlp(\Pi)$ such that $O \in S$ and $S \cap \mathscr{A} = E$.*

Example 3.1. Let $\Pi = \langle P, \mathscr{A} \rangle$ be an abductive program such that

$$\begin{aligned} P : \ & \textit{wet-shoes} \leftarrow \textit{wet-grass}, \textit{not driving-car}, \\ & \textit{wet-grass} \leftarrow \textit{rained}, \\ & \textit{wet-grass} \leftarrow \textit{sprinkler-on}. \\ \mathscr{A} : \ & \textit{rained}, \textit{sprinkler-on}. \end{aligned}$$

Then Π has four belief sets:

$$\begin{aligned} & \emptyset, \\ & \{\textit{rained}, \textit{wet-grass}, \textit{wet-shoes}\}, \\ & \{\textit{sprinkler-on}, \textit{wet-grass}, \textit{wet-shoes}\}, \\ & \{\textit{rained}, \textit{sprinkler-on}, \textit{wet-grass}, \textit{wet-shoes}\}. \end{aligned}$$

On the other hand, $dlp(\Pi)$ is the program:

$$\begin{aligned} & \textit{wet-shoes} \leftarrow \textit{wet-grass}, \textit{not driving-car}, \\ & \textit{wet-grass} \leftarrow \textit{rained}, \\ & \textit{wet-grass} \leftarrow \textit{sprinkler-on}, \\ & \textit{rained} \mid \varepsilon_r \leftarrow, \\ & \textit{sprinkler-on} \mid \varepsilon_s \leftarrow . \end{aligned}$$

Then, $dlp(\Pi)$ has the answer sets:

$$\begin{aligned} & \{\varepsilon_r, \varepsilon_s\}, \\ & \{\varepsilon_s, \textit{rained}, \textit{wet-grass}, \textit{wet-shoes}\}, \\ & \{\varepsilon_r, \textit{sprinkler-on}, \textit{wet-grass}, \textit{wet-shoes}\}, \\ & \{\textit{rained}, \textit{sprinkler-on}, \textit{wet-grass}, \textit{wet-shoes}\}. \end{aligned}$$

Thus, the belief sets of Π and the answer sets of $dlp(\Pi)$ coincide up to the literals in P. In particular, $\{\varepsilon_r, \varepsilon_s\}$ is the $\mathscr{A}$-minimal answer set which corresponds to the $\mathscr{A}$-minimal belief set $\emptyset$ of Π.

3.2. From belief sets to possible models

Next we consider converting the belief sets of an AELP into the possible models of an EDP.

For an AELP Π every disjunctive rule in the program $dlp(\Pi)$ is of the form (2), then the following result holds.

Lemma 3.1 (Answer sets vs possible models in $dlp(\Pi)$). *Let $\Pi = \langle P, \mathscr{A} \rangle$ be an AELP.*

(i) *If S is an answer set of $dlp(\Pi)$, then S is a possible model of $dlp(\Pi)$.*

(ii) *For any possible model T of $dlp(\Pi)$, there is an answer set S of $dlp(\Pi)$ such that $S \cap Lit = T \cap Lit$.*

Proof. (i) Since an answer set is a possible model, the result immediately follows.

(ii) The difference between an answer set and a possible model is that a possible model may contain both A and ε_A for each ground disjunctive rule $A \mid \varepsilon_A \leftarrow$ from $dlp(\Pi)$. When a possible model T contains both A and ε_A, only A is included in $T \cap Lit$. In this case, there is an answer set S of $dlp(\Pi)$ which contains A. Hence the result holds. $\square$

Using the above Lemma, the results of Theorem 3.1 are directly applied to the possible models. In the following, S is called an *$\mathscr{A}$-minimal* possible model if it is a possible model such that $S \cap \mathscr{A}$ is minimal.

Theorem 3.2 (Belief sets of an AELP $\rightarrow$ possible models of an EDP). *Let $\Pi = \langle P, \mathscr{A} \rangle$ be an AELP. If S is an ($\mathscr{A}$-minimal) belief set of Π, there is an ($\mathscr{A}$-minimal) possible model T of $dlp(\Pi)$ such that $T \cap Lit = S$. Conversely, if T is an ($\mathscr{A}$-minimal) possible model of $dlp(\Pi)$, there is an ($\mathscr{A}$-minimal) belief set S of Π such that $S = T \cap Lit$.*

Under the possible model semantics, the *dlp*-transformation is further simplified as follows.

Let $\Pi = \langle P, \mathscr{A} \rangle$ be an AELP. Then, the *dlp_{pm}-transformation* transforms Π to the EDP $dlp_{pm}(\Pi)$ which is obtained from P by introducing the following disjunctive rules for each abducible $A \in \mathscr{A}$:

$$A \mid \varepsilon \leftarrow \tag{3}$$

where ε is an atom appearing nowhere in P. Comparing (2) and (3), the dlp_{pm}-transformation does not distinguish each ε_A from ε_B for any other abducible B. Then the following result holds.

Theorem 3.3 (Belief sets of an AELP $\rightarrow$ possible models of an EDP – simplified transformation). *Let $\Pi = \langle P, \mathscr{A} \rangle$ be an AELP. If S is an ($\mathscr{A}$-minimal) belief set of Π, there is an ($\mathscr{A}$-minimal) possible model T of $dlp_{pm}(\Pi)$ such that $T \setminus \{\varepsilon\} = S$. Conversely, if T is an ($\mathscr{A}$-minimal) possible model of $dlp(\Pi)$, there is an ($\mathscr{A}$-minimal) belief set S of Π such that $S = T \setminus \{\varepsilon\}$.*

Proof. Let T be a belief set of Π. Then T is an answer set of $P \cup E$ where $E \subseteq \mathscr{A}$. Then there is a split program P' of $dlp_{pm}(\Pi)$ such that for each ground disjunctive rule $A \mid \varepsilon \leftarrow$ from $dlp(\Pi)$, $(A \leftarrow) \in P'$ if $A \in E$; and $(\varepsilon \leftarrow) \in P'$ if $A \notin E$. When $\varepsilon \leftarrow$ is in P', $T \cup \{\varepsilon\}$ is an answer set of P' and also a possible model of $dlp_{pm}(\Pi)$. Else when $\varepsilon \leftarrow$ is not in P', T is an answer set of P' and also a possible model of $dlp_{pm}(\Pi)$. Hence the result of the only-if part follows.

In converse, when S is a possible model of $dlp_{pm}(\Pi)$, it is an answer set of some split program P' of $dlp_{pm}(\Pi)$. Let E be the set of all split rules included in P'. Then S is an answer set of $P \cup E$. Since $E \setminus \{\varepsilon \leftarrow\}$ consists of instances from $\mathscr{A}$, $S \setminus \{\varepsilon\}$ is a belief set of Π.

The correspondence between the $\mathscr{A}$-minimal belief sets and the $\mathscr{A}$-minimal possible models also follows from the above result and the definitions. $\square$

Example 3.2. In the program of Example 3.1, $dlp_{pm}(\Pi)$ has the rules in P plus the disjunctive rules:

$$rained \mid \varepsilon \leftarrow,$$
$$sprinkler\text{-}on \mid \varepsilon \leftarrow .$$

Then, $dlp_{pm}(\Pi)$ has five possible models:

$$S_1 = \{\varepsilon\},$$
$$S_2 = \{\varepsilon,\ rained,\ wet\text{-}grass,\ wet\text{-}shoes\},$$
$$S_3 = \{\varepsilon,\ sprinkler\text{-}on,\ wet\text{-}grass,\ wet\text{-}shoes\},$$
$$S_4 = \{rained,\ sprinkler\text{-}on,\ wet\text{-}grass,\ wet\text{-}shoes\},$$
$$S_5 = \{\varepsilon,\ rained,\ sprinkler\text{-}on,\ wet\text{-}grass,\ wet\text{-}shoes\}.$$

Thus, the belief sets of Π and the answer sets of $dlp_{pm}(\Pi)$ coincide up to the literals in P. In particular, S_1 is the $\mathscr{A}$-minimal possible model which corresponds to the $\mathscr{A}$-minimal belief set $\emptyset$ of Π.

Note here that the sets S_2, S_3 and S_5 do not become answer sets of $dlp_{pm}(\Pi)$, since $S_1 \subset S_2$, $S_1 \subset S_3$, and $S_1 \subset S_5$. Therefore, the simplified transformation dlp_{pm} cannot be used for relating AELPs to EDPs under the answer set semantics.

4. Transforming disjunctive LP to abductive LP

4.1. From possible models to belief sets

Indefinite information in disjunctive programs is considered to represent alternative beliefs. This suggests the possibility of converting disjunctive knowledge into abducibles in an abductive program. However, disjunctive rules generally have conditions in their bodies, while abductive programs introduced in Section 2 lack the ability of expressing assumptions with preconditions. To fill the gap, we first consider the *knowledge system* [19] which has a mechanism of hypothesizing rules in an ELP.

A knowledge system (KS) is a pair $K = \langle T, H \rangle$, where both T and H are ELPs. The rules in H are hypothetical rules that are used for abducing an observation together with the background theory T. Abductive programs introduced in Section 2 are considered as a special case of a knowledge system in which each hypothetical rule has the empty precondition. The belief sets of a knowledge system is defined as follows. Given a KS $K = \langle T, H \rangle$, a set of literals S ($\subset Lit$) is a *belief set* of K (wrt E) if S is an answer set of $T \cup E$ where $E \subseteq H$. Clearly, the belief sets introduced above reduce to those presented in Section 2 when H consists of abducible literals.

We first provide a program transformation which transforms an EDP to a KS introduced above. For an EDP P, we define $P = disj(P) \cup \overline{disj}(P)$ where $disj(P)$ is the set of all disjunctive rules from P and $\overline{disj}(P)$ is the set of all non-disjunctive rules (i.e., normal rules and integrity constraints) from P.

Definition 4.1 (*ks*-transformation). Given an EDP P, let us consider the set of normal rules

$$H = \{L_i \leftarrow \Gamma \ : (L_1 \mid \cdots \mid L_l \leftarrow \Gamma) \in disj(P) \text{ and } 1 \leqslant i \leqslant l\} \tag{4}$$

and the integrity constraints

$$IC = \{\leftarrow \Gamma, not\, L_1, \ldots, not\, L_l : (L_1 \mid \cdots \mid L_l \leftarrow \Gamma) \in disj(P)\}. \tag{5}$$

Then the *ks-transformation* transforms P to the knowledge system $ks(P) = \langle T, H\rangle$ where $T = \overline{disj}(P) \cup IC$.

The *ks*-transformation replaces each disjunctive rule in a program with a set of hypothetical rules (4) in H. The integrity constraints (5) in IC impose the condition that at least one disjunct is chosen as a hypothesis whenever the body of the disjunctive rule is true.

It is shown that the possible models of an EDP P are equivalent to the belief sets of the transformed KS $ks(P)$.

Theorem 4.1 (Possible models of an EDP $\rightarrow$ belief sets of a KS). *Let P be an EDP. Then S is a possible model of P iff S is a belief set of $ks(P)$.*

Proof. Let S be a possible model of P. Then there is a split program P' of P such that S is an answer set of P'. Suppose that each ground disjunctive rule $r_k : L_1 \mid \cdots \mid L_{l_k} \leftarrow \Gamma_k$ from P is replaced by split rules R_k in P' where R_k is a non-empty subset of $Split_{r_k} = \{L_i \leftarrow \Gamma_k \mid i = 1, \ldots, l_k\}$. Then S is an answer set of $\overline{disj}(P) \cup \bigcup_k R_k$, where $\bigcup_k R_k$ is a collection of split rules from each r_k. Since $\bigcup_k R_k$ consists of instances from H and S satisfies the integrity constraints IC, S is also a belief set of $ks(P)$.

Conversely, let S be a belief set of $ks(P)$. Then S is an answer set of $T \cup E$ where $T = \overline{disj}(P) \cup IC$ and $E \subseteq H$. For each normal rule $L_i \leftarrow \Gamma$ in E, there is a corresponding disjunctive rule $r : L_1 \mid \cdots \mid L_l \leftarrow \Gamma$ in $disj(P)$ such that $1 \leqslant i \leqslant l$. As S satisfies the integrity constraints IC, when S satisfies Γ, at least one normal rule $L_i \leftarrow \Gamma$ is included in E for each r. In this case, there is a split program P' of P in which each ground instance of the rule r is split into a ground instance of the rule $L_i \leftarrow \Gamma$ in E. Thus S is also an answer set of P', hence a possible model of P. $\square$

Example 4.1 [4]. Let P be the program:

$$violent \mid psychopath \leftarrow suspect,$$
$$dangerous \leftarrow violent, psychopath,$$
$$suspect \leftarrow .$$

Then it becomes $ks(P) = \langle T, H\rangle$ where

$$\begin{aligned} T :\ & dangerous \leftarrow violent, psychopath,\\ & suspect \leftarrow,\\ & \leftarrow suspect, not\ violent, not\ psychopath.\\ H :\ & violent \leftarrow suspect,\\ & psychopath \leftarrow suspect. \end{aligned}$$

The knowledge system $ks(P)$ has three belief sets:

$$\{suspect,\ violent\},$$
$$\{suspect,\ psychopath\},$$
$$\{suspect,\ violent,\ psychopath,\ dangerous\},$$

which coincide with the possible models of P.

Note that in the above example P has no answer set containing *dangerous*. By contrast, $ks(P)$ has a belief set in which *dangerous* is true, which corresponds to a possible model in which the disjunction is inclusively true.

A knowledge system $\langle T, H\rangle$ can be transformed to a semantically equivalent abductive program $\langle P, \mathscr{A}\rangle$ as follows [19].

Given a KS $\langle T, H\rangle$, consider a program P and abducibles $\mathscr{A}$ such that

$$P = T \cup \{L \leftarrow \delta_r, \Gamma \mid r = (L \leftarrow \Gamma) \in H\},$$
$$\mathscr{A} = \{\delta_r \mid r \in H\}.$$

Here, δ_r is a newly introduced atom appearing nowhere in T and is uniquely associated with each hypothetical rule r in H. This transformation is called *naming* of hypothetical rules. [5] Then, there is a one-to-one correspondence between the belief sets of $\langle T, H\rangle$ and the belief sets of $\langle P, \mathscr{A}\rangle$.

Proposition 4.1 (Knowledge system vs abductive program [19]). *Let $\langle T, H\rangle$ be a knowledge system and $\langle P, \mathscr{A}\rangle$ an abductive program as presented above. If S is a belief set of $\langle T, H\rangle$, there is a belief set S' of $\langle P, \mathscr{A}\rangle$ such that $S' \cap Lit = S$. Conversely, if S' is a belief set of $\langle P, \mathscr{A}\rangle$, there is a belief set S of $\langle T, H\rangle$ such that $S = S' \cap Lit$.*

For instance, the knowledge system $ks(P)$ of Example 4.1 is transformed to the abductive program $\langle P', \mathscr{A}\rangle$ such that

$$\begin{aligned} P':\ & dangerous \leftarrow violent, psychopath,\\ & suspect \leftarrow,\\ & \leftarrow suspect, not\ violent, not\ psychopath,\\ & violent \leftarrow \delta_1, suspect,\\ & psychopath \leftarrow \delta_2, suspect.\\ \mathscr{A}:\ & \delta_1,\ \delta_2. \end{aligned}$$

Here, δ_1 and δ_2 are newly introduced abducibles associated with the hypothetical rules $violent \leftarrow suspect$ and $psychopath \leftarrow suspect$, respectively. Then, $\langle P', \mathscr{A}\rangle$ has three belief sets:

$$\{suspect,\ violent,\ \delta_1\},$$
$$\{suspect,\ psychopath,\ \delta_2\},$$
$$\{suspect,\ violent,\ psychopath,\ dangerous,\ \delta_1,\ \delta_2\},$$

which correspond to the belief sets of $ks(P)$.

[5] A similar technique is introduced in Ref. [31].

Since the possible models of an EDP are converted into the belief sets of a KS (Theorem 4.1), the above fact implies that the possible models of an EDP are also expressed by the belief sets of an AELP. We define the *alp-transformation* which transforms an EDP P to an AELP $alp(P)$ by combining the *ks*-transformation and naming as presented above. Then, we have the following result.

Theorem 4.2 (Possible models of an EDP $\rightarrow$ belief sets of an AELP). *Let P be an EDP. If S is a possible model of P, there is a belief set T of an AELP $alp(P)$ such that $T \cap Lit = S$. Conversely, if T is a belief set of $alp(P)$, there is a possible model S of P such that $S = T \cap Lit$.*

Proof. The result follows from Theorem 4.1 and Proposition 4.1. $\square$

4.2. From answer sets to belief sets

Next we consider converting the answer sets of a disjunctive program into the belief sets of an abductive program. In not-free EDPs, the following result holds.

Theorem 4.3 (Answer sets of a not-free EDP $\rightarrow$ $\mathscr{A}$-minimal belief sets of an AELP). *Let P be a not-free EDP. If S is an answer set of P, there is an $\mathscr{A}$-minimal belief set T of $alp(P)$ such that $T \cap Lit = S$. Conversely, if T is an $\mathscr{A}$-minimal belief set of $alp(P)$, there is an answer set S of P such that $S = T \cap Lit$.*

Proof. In a not-free EDP, the answer sets coincide with the minimal possible models (Proposition 2.1). Let $alp(P) = \langle P', \mathscr{A} \rangle$ and I be the set of integrity constraints in P'. Then, $P' \setminus I$ contains no *not*, so that the minimal belief sets of $alp(P)$ coincide with the $\mathscr{A}$-minimal belief sets of $alp(P)$ (Propositions 2.2 and 2.3). By Theorem 4.2, there is a one-to-one correspondence between the (minimal) possible models of P and the (minimal) belief sets of $alp(P)$. Therefore, there is also a one-to-one correspondence between the answer sets of P and the $\mathscr{A}$-minimal belief sets of $alp(P)$. Hence, the result follows. $\square$

The above result does not hold for EDPs containing negation as failure.

Example 4.2. Let P_1 be the program:

$$a \mid b \leftarrow,$$

$$a \leftarrow b, not\, c,$$

$$b \leftarrow a, not\, d,$$

$$c \leftarrow not\, a,$$

$$d \leftarrow not\, b,$$

which has three answer sets: $\{a, d\}$, $\{b, c\}$, $\{a, b\}$. Then, $alp(P_1) = \langle P_1', \mathscr{A} \rangle$ becomes

$$
\begin{aligned}
P_1' :\ & a \leftarrow \delta_a,\\
& b \leftarrow \delta_b,\\
& \leftarrow not\, a,\ not\, b,\\
& a \leftarrow b,\ not\, c,\\
& b \leftarrow a,\ not\, d,\\
& c \leftarrow not\, a,\\
& d \leftarrow not\, b.\\
\mathscr{A} :\ & \delta_a, \delta_b.
\end{aligned}
$$

Here, $alp(P_1)$ has the belief sets: $T_1 = \{a, d, \delta_a\}$, $T_2 = \{b, c, \delta_b\}$, $T_3 = \{a, b, \delta_a, \delta_b\}$, of which T_3 is not $\mathscr{A}$-minimal. Thus, for an answer set S of an EDP P, there does not necessarily exist an $\mathscr{A}$-minimal belief set T of $alp(P)$ such that $T \cap Lit = S$.

Next let P_2 be the program:

$$
\begin{aligned}
& a \mid b \leftarrow,\\
& b \mid c \leftarrow,\\
& b \leftarrow a,\\
& c \leftarrow not\, a,
\end{aligned}
$$

which has the answer set $\{b, c\}$. Then, $alp(P_2) = \langle P_2', \mathscr{A} \rangle$ becomes

$$
\begin{aligned}
P_2' :\ & a \leftarrow \delta_a,\\
& b \leftarrow \delta_b,\\
& c \leftarrow \delta_c,\\
& \leftarrow not\, a,\ not\, b,\\
& \leftarrow not\, b,\ not\, c,\\
& b \leftarrow a,\\
& c \leftarrow not\, a.\\
\mathscr{A} :\ & \delta_a,\ \delta_b,\ \delta_c.
\end{aligned}
$$

Here, $alp(P_2)$ has two $\mathscr{A}$-minimal belief sets: $T_1 = \{b, c, \delta_b\}$ and $T_2 = \{a, b, \delta_a\}$, but there is no answer set of P_2 which corresponds to T_2. Thus, for an $\mathscr{A}$-minimal belief set T of $alp(P)$, there does not necessarily exist an answer set S of an EDP P such that $S = T \cap Lit$.

From the result of Section 4.1, the *alp*-transformation converts the possible models of an EDP into the belief sets of an AELP. In contrast, the answer sets of an EDP are not expressed by the belief sets of an AELP, while they are characterized by the $\mathscr{A}$-minimal belief sets if an EDP is *not*-free (Theorem 4.3). It is left open whether there is another transformation which converts the answer sets of an EDP into the $\mathscr{A}$-minimal belief sets of an AELP in general.

5. Transforming abductive disjunctive LP to disjunctive LP

Abductive disjunctive programs are disjunctive programs with abducibles, and their semantics are defined as the belief set semantics in Section 2. The possible model semantics are also extended to abductive disjunctive programs as follows.

Given an AEDP $\Pi = \langle P, \mathscr{A} \rangle$, S is a *possible belief set* of Π if S is a possible model of the EDP $P \cup E$ where $E \subseteq \mathscr{A}$. A possible belief set S is $\mathscr{A}$*-minimal* if there is no possible belief set T such that $T \cap \mathscr{A} \subset S \cap \mathscr{A}$. By the definition, the possible belief sets coincide with the belief sets in AELPs, and reduce to the possible models in EDPs.

A difference between belief sets and possible belief sets is illustrated below.

Example 5.1. Let $\Pi = \langle P, \mathscr{A} \rangle$ be an abductive program such that

$$
\begin{aligned}
P :\ & p \leftarrow q, r, \\
& q \mid r \leftarrow a. \\
\mathscr{A} :\ & a.
\end{aligned}
$$

Then, $\emptyset, \{q, a\}, \{r, a\}, \{p, q, r, a\}$ are the possible belief sets of Π, while $\{p, q, r, a\}$ is not a belief set. Given the observation $O = p$, it has the explanation a under the possible belief set semantics, while no explanation is available under the belief set semantics.

Thus, the possible belief set semantics can provide explanations which come from inclusive disjunctions, while such explanations do not come out under the belief set semantics in general.

The belief sets of an abductive disjunctive program are converted into the answer sets of a disjunctive program using the *dlp*-transformation in Section 3. For an AEDP $\Pi = \langle P, \mathscr{A} \rangle$, its *dlp-transformation* $dlp(\Pi)$ is defined as Definition 3.1. Then the following result holds.

Theorem 5.1 (Belief sets of an AEDP $\rightarrow$ answer sets of an EDP). *Let $\Pi = \langle P, \mathscr{A} \rangle$ be an AEDP. If S is an ($\mathscr{A}$-minimal) belief set of Π, there is an ($\mathscr{A}$-minimal) answer set T of $dlp(\Pi)$ such that $T \cap Lit = S$. Conversely, if T is an ($\mathscr{A}$-minimal) answer set of $dlp(\Pi)$, there is an ($\mathscr{A}$-minimal) belief set S of Π such that $S = T \cap Lit$.*

Proof. Similar to the proof of Theorem 3.1. □

The above theorem presents that the belief sets of an AEDP are expressed by the answer sets of the transformed EDP. This implies the significant fact that *abducibles do not introduce additional expressive power to disjunctive programs*. Such reduction is also done using the possible belief sets.

Theorem 5.2 (Possible belief sets of an AEDP $\rightarrow$ possible models of an EDP). *Let $\Pi = \langle P, \mathscr{A} \rangle$ be an AEDP. If S is an ($\mathscr{A}$-minimal) possible belief set of Π, there is an ($\mathscr{A}$-minimal) possible model T of $dlp_{pm}(\Pi)$ such that $T \setminus \{\varepsilon\} = S$. Conversely, if T is an ($\mathscr{A}$-minimal) possible model of $dlp(\Pi)$, there is an ($\mathscr{A}$-minimal) possible belief set S of Π such that $S = T \setminus \{\varepsilon\}$.*

Proof. Similar to the proof of Theorem 3.3. □

Theorem 5.2 presents that AEDPs are reducible to EDPs also under the possible model semantics. Moreover, in Section 4.1 it was shown that the possible models of an EDP P are converted into the belief sets of an AELP $alp(P)$ (Theorem 4.2). Combining these results, we have the following result.

Corollary 5.1 (Possible belief sets of an AEDP → belief sets of an AELP). *Let $\Pi = \langle P, \mathscr{A} \rangle$ be an AEDP. If S is an ($\mathscr{A}$-minimal) possible belief set of Π, there is an ($\mathscr{A}$-minimal) belief set T of an AELP $alp(dlp_{pm}(\Pi))$ such that $T \cap Lit = S$. Conversely, if T is an ($\mathscr{A}$-minimal) belief set of $alp(dlp_{pm}(\Pi))$, there is an ($\mathscr{A}$-minimal) possible belief set S of Π such that $S = T \cap Lit$.*

Proof. The result follows from Theorems 5.2 and 4.2. □

The above corollary presents that AELPs are as expressive as AEDPs under the possible belief set semantics. That is, *disjunctions do not introduce additional expressibility to abductive programs under the possible belief set semantics.*

6. Relating possible models to answer sets

The *alp*-transformation converts the possible models of an EDP into the belief sets of an AELP. On the other hand, the belief sets of an AELP are converted into the answer sets of an EDP by the *dlp*-transformation. Composing these two transformations, the possible models of an EDP are transformed to the answer sets of an EDP.

Theorem 6.1 (Possible models of an EDP → answer sets of an EDP). *Let P be an EDP. If S is a possible model of P, there is an answer set T of $dlp(alp(P))$ such that $T \cap Lit = S$. Conversely, if T is an answer set of $dlp(alp(P))$, there is a possible model S of P such that $S = T \cap Lit$.*

Proof. The result follows from Theorems 4.2 and 3.1. □

Example 6.1. Let P be the program:

$$a \mid \neg b \leftarrow c,$$
$$c \leftarrow not\, d,$$

which has three possible models: $S_1 = \{a, c\}$, $S_2 = \{\neg b, c\}$, $S_3 = \{a, \neg b, c\}$.

First, $alp(P)$ becomes $\langle P', \mathscr{A} \rangle$ where

$$\begin{aligned} P' :\ & a \leftarrow \delta_1, c,\\ & \neg b \leftarrow \delta_2, c,\\ & \leftarrow c, not\ a, not\ \neg b,\\ & c \leftarrow not\ d.\\ \mathscr{A} :\ & \delta_1,\ \delta_2. \end{aligned}$$

Then, $dlp(alp(P))$ becomes

$$a \leftarrow \delta_1, c,$$
$$\neg b \leftarrow \delta_2, c,$$
$$\leftarrow c,\ not\,a,\ not\,\neg b,$$
$$c \leftarrow not\,d,$$
$$\delta_1 \mid \varepsilon_{\delta_1} \leftarrow,$$
$$\delta_2 \mid \varepsilon_{\delta_2} \leftarrow,$$

which has three answer sets: $T_1 = \{a, c, \delta_1, \varepsilon_{\delta_2}\}$, $T_2 = \{\neg b, c, \varepsilon_{\delta_1}, \delta_2\}$, $T_3 = \{a, \neg b, c, \delta_1, \delta_2\}$. Thus, $S_i = T_i \cap Lit$ $(i = 1, 2, 3)$ holds.

The above theorem presents that possible models are expressed by answer sets in general. However, this fact does not depreciate the value of the possible model semantics. Indeed, the transformation from possible models to answer sets requires the introduction of new symbols, and the resulting program is less natural and harder to understand than the original program. [6] Moreover, the possible model semantics has a nice computational property as presented in the next section.

7. Discussion

7.1. Computational complexity

In this section, we discuss the relationship between ALP and DLP from the computational complexity viewpoint. Throughout this section, programs mean ground programs.

While there are various complexity measures in abduction problems [8,10], we consider in this section the decision problem of finding a (credulous) explanation for a given observation. Recall that the problem of finding a (credulous) explanation for an observation is identical to the problem of finding a belief set satisfying the observation (Section 2.2). Then, the above decision problem is rephrased as the *set-membership problem* in an abductive program, i.e., *deciding whether there is a* (*possible*) *belief set satisfying an observation*. When abductive programs do not contain negation as failure, Eiter and Gottlob [8] and Selman and Levesque [39] show that the decision problem of finding an explanation for an observation in an abductive Horn program is NP-complete. In other words, in an abductive Horn program deciding whether there is a belief model satisfying an observation is NP-complete.

Inoue [19] shows that an abductive ELP can be transformed to a semantically equivalent ELP under the answer set semantics. For an AELP $\Pi = \langle P, \mathscr{A} \rangle$, consider an ELP obtained from P by adding the following rules for each abducible L in $\mathscr{A}$:

$$L \leftarrow not\,L',$$
$$L' \leftarrow not\,L$$

[6] Ref. [20] provides another simpler transformation, but it still needs the introduction of extra symbols.

where L' is a literal appearing nowhere in P and is uniquely associated with each L. Then it is shown that there is a one-to-one correspondence between the belief sets of Π and the answer sets of the transformed ELP. Satoh and Iwayama [38] introduce a similar transformation from ANLP to NLP. These results imply the following fact.

Theorem 7.1 (Set-membership problem for the belief sets of an AELP). *Deciding whether there is a belief set of an AELP satisfying an observation is NP-complete.*

Proof. It is known that deciding whether a literal is true in an answer set of an ELP is NP-complete [2,28]. Belief sets include answer sets as a special case, then the set-membership problem in an AELP is NP-hard. As the polynomial-time transformation from an AELP to an ELP presented above translates the decision problem for the belief sets of an AELP into the corresponding problem for the answer sets of an ELP which is in NP, the membership in NP also follows. [7] □

Sakama and Inoue [37] show that the possible models of an NDP can be expressed by the stable models of an NLP. Given an NDP P, consider an NLP obtained from P by replacing each disjunctive rule:

$$A_1 \mid \cdots \mid A_l \leftarrow \Gamma$$

in P with the following normal rules and the integrity constraint:

$$\begin{aligned}
&A_i \leftarrow \Gamma, \mathit{not}A'_i \quad \text{for } i = 1, \ldots, l,\\
&A'_i \leftarrow \Gamma, \mathit{not}A_i \quad \text{for } i = 1, \ldots, l,\\
&\leftarrow \Gamma,\ A'_1, \ldots, A'_l
\end{aligned}$$

where each A'_i is an atom appearing nowhere in P and is uniquely introduced for each A_i in the Herbrand base. Then they show that there is a one-to-one correspondence between the possible models of P and the stable models of the transformed NLP. Using the transformation, they prove that the set-membership problem in an NDP under the possible model semantics is NP-complete. This result is immediately extended to EDPs. As shown in Section 5, the possible belief sets of an AEDP are converted into the possible models of an EDP. Hence, the set-membership problem for the possible belief sets of an AEDP is efficiently translated into the corresponding problem for the possible models of an EDP. This fact implies that:

Theorem 7.2 (Set-membership problem for the possible belief sets of an AEDP). *Deciding whether there is a possible belief set of an AEDP satisfying an observation is NP-complete.*

On the other hand, the set-membership problem in an EDP under the answer set semantics is known to be Σ_2^P-complete [9]. Then we have the following result.

[7] Recall that if a problem L' is polynomial-time reducible to an NP problem L, L' is also in NP [17, Lemma 13.1].

Table 1
Comparison of computational complexity

Program	Semantics	Complexity
Abductive LP	Belief model (Horn)	NP-complete [8,39]
	Belief set (AELP)	NP-complete (Theorem 7.1)
Disjunctive LP	Answer set	Σ_2^P-complete [9]
	Possible model	NP-complete [37]
Abductive DLP	Belief set	Σ_2^P-complete (Theorem 7.3)
	Possible belief set	NP-complete (Theorem 7.2)

Theorem 7.3 (Set-membership problem for the belief sets of an AEDP). *Deciding whether there is a belief set of an AEDP satisfying an observation is Σ_2^P-complete.*

Proof. As belief sets include answer sets as a special case, the set-membership problem in an AEDP is Σ_2^P-hard. To see that it is in Σ_2^P, the *dlp*-transformation in Section 5 efficiently translates the decision problem for the belief sets of an AEDP into the corresponding problem for the answer sets of an EDP which is in Σ_2^P. Then the membership in Σ_2^P follows. □

These results are summarized in Table 1.

The above complexity analyses verify the results of this paper that there are bidirectional polynomial-time transformations between the belief sets of an AELP and the possible models of an EDP. Moreover, we can observe that *there is no efficient way to express the answer sets of an EDP in terms of the belief sets of an AELP in general unless the polynomial hierarchy collapses*. This observation extends the well-known fact that there is no efficient way to express the answer sets of an EDP in terms of the answer sets of an ELP in general [9]. [8] On the other hand, Theorem 4.3 presents that the answer sets of a not-free EDP are converted into the $\mathscr{A}$-minimal belief sets of an AELP. Since the set-membership problem for the answer sets of a not-free EDP is Σ_2^P-complete, the corresponding decision problem for the $\mathscr{A}$-minimal belief sets of an AELP is Σ_2^P-hard. The fact presents that checking the $\mathscr{A}$-minimality introduces an additional source of complexity to abduction. This observation coincides with the result in [10], which indicates that in nonmonotonic theories checking the minimality of an explanation causes an increase in complexity by one level of the polynomial hierarchy.

We can also observe that when considering an extension from AELPs to AEDPs, the possible belief set semantics extends the framework without increasing computational complexity, while this is not the case for the belief set semantics of AEDPs. The fact that the complexity of computing answer sets of EDPs (or belief sets of AEDPs) is higher than the complexity of computing possible models of EDPs (or possible belief sets of AEDPs) is explained as follows. Computation of answer sets or belief sets introduces an additional source of complexity for its minimality-checking, while this is not the case for computation of possible models or possible belief sets due to its 'non-minimal' feature. From the complexity viewpoints, the answer set semantics is strictly more expressive than the possible model semantics. Neverthe-

[8] Such reduction is possible if an EDP is *head-cycle-free* [2].

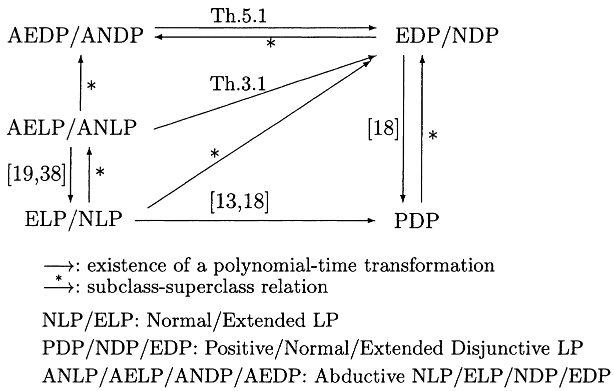

Fig. 1. Relationship between various extensions of logic programming under the answer set semantics.

less, as shown in this paper, the possible model semantics is useful for knowledge representation and reasoning in both abductive and disjunctive programs.

7.2. Relations between extensions of LPs

AELPs are expressed by EDPs under the answer set semantics (Theorem 3.1), and AEDPs are reducible to EDPs (Theorem 5.1). In other words, under the answer set semantics abducibles are equivalently specified by disjunctions, but not vice versa. Also, as presented in Section 7.1, AELPs/ANLPs are reducible to ELPs/NLPs under the answer set/stable model semantics [19,38]. That is, abducibles are expressed using negation as failure. [9] The authors in Refs. [13,18] introduce program transformations from the stable models of NLPs and NDPs to the minimal models of PDPs. This fact implies that negation as failure is equivalently specified by disjunctions (plus integrity constraints). However, specifying disjunctions in terms of negation as failure is generally impossible. This is because NDPs are not efficiently reducible to NLPs under the stable model semantics in general [9]. Summarizing the results, the relationship between the extensions of logic programming under the answer set semantics is illustrated in Fig. 1. Recall that (A)EDPs and (A)ELPs are respectively reducible to (A)NDPs and (A)NLPs, by replacing each negative literal $\neg A$ with a new atom A' and inserting the integrity constraint $\leftarrow A, A'$ for every such new atom [15]. Thus (abductive) extended (disjunctive) programs are identified with (abductive) normal (disjunctive) programs in Fig. 1.

In Fig. 1, an arrow $A \rightarrow B$ presents the existence of a polynomial-time transformation from a program in the class of A to a semantically equivalent program in the class of B. An arrow $A \overset{*}{\rightarrow} B$ presents that A is a subclass of B (hence a transformation from A to B is trivial). Every program transformation in Fig. 1 is *modular*, i.e., a transformation of a part of a program is done independently from the other part of the program.

[9] In converse, Refs. [12,22] simulate negation as failure using abducibles, together with (meta-level) integrity constraints in disjunctive formulas.

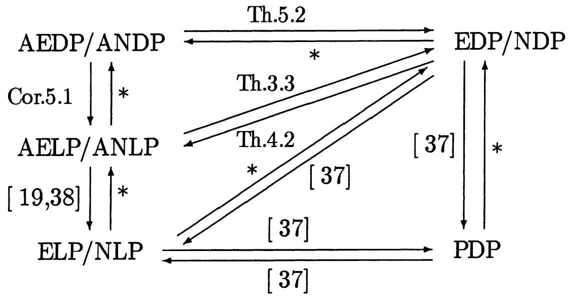

Fig. 2. Relationship between various extensions of logic programming under the possible model semantics.

Under the possible model semantics, on the other hand, AELPs are expressed by EDPs and vice versa (Theorems 3.3 and 4.2). In other words, abducibles are equivalently specified by disjunctions, and vice versa. Also, AEDPs are reducible to EDPs (Theorem 5.2) and to AELPs (Corollary 5.1). In Ref. [37] it is shown that under the possible model semantics NDPs and NLPs are transferable to semantically equivalent PDPs. Furthermore, Ref. [37] show that NDPs/PDPs are reducible to NLPs using the program transformation presented in Section 7.1. The program transformation from AELPs/ANLPs to ELPs/NLPs of Refs. [19,38] is also applicable under the possible model semantics, since possible belief sets (resp. possible models) are equivalent to belief sets (resp. answer sets) in AELPs (resp. ELPs). These relations are summarized in Fig. 2. Every program transformation in Fig. 2 is also modular and performed in polynomial time.

From Fig. 2, it is observed that the difference between the answer set semantics and the possible model semantics is the existence of the links: AEDP/ANDP → AELP/ANLP, PDP → ELP/NLP, and EDP/NDP → (A)ELP/ (A)NLP. That is, disjunctions can also be replaced by negation as failure or abducibles under the possible model semantics. As a result, there is a cycle ELP/NLP → AELP/ANLP → AEDP/ANDP → EDP/NDP → PDP → ELP/NLP. This implies a rather surprising fact that *all 'extensions' of logic programming, i.e., normal and extended programs, disjunctive programs, and abductive programs, are essentially equivalent under the possible model semantics. That is, negation as failure, disjunctions, and abducibles can be used interchangeably under the possible model semantics.*

The possible model semantics is originally introduced as a semantics for disjunctive programs. However, the above observation indicates that the possible model emantics can provide a unified framework for various extensions of logic programming.

8. Conclusion

Abductive hypotheses and disjunctive knowledge appear to deal with very similar problems from different viewpoints. This paper verified this conjecture and revealed the close relationship between abductive logic programming and disjunctive logic programming. The main results of this paper are summarized as follows.

- Abductive programs can be transformed to disjunctive programs both under the answer set semantics and the possible model semantics.
- Disjunctive programs can be transformed to abductive programs under the possible model semantics, while such a transformation is most unlikely possible in general under the answer set semantics.
- Abductive disjunctive programs can be reduced to disjunctive programs both under the answer set semantics and the possible model semantics. Furthermore, abductive disjunctive programs are reducible to abductive (non-disjunctive) programs under the possible model semantics.

By the transformation from abductive programs to disjunctive programs, it is concluded that abducibles are equivalently specified by disjunctions. In particular, abductive disjunctive programs are reducible to disjunctive programs, hence no expressive power is gained by introducing abducibles to disjunctive programs. On the other hand, disjunctive programs are efficiently transformed to abductive programs under the possible model semantics. From these facts, we can conclude that abducibles and disjunctions are essentially equivalent under the possible model semantics. In other words, abductive programs and disjunctive programs are just different ways of looking at the same problem under the possible model semantics.

The results of this paper verify the usefulness of the possible model semantics as a semantics not only for disjunctive programs but also for abductive programs. Moreover, the possible model semantics can provide a unifying framework for various extensions of logic programming without introducing additional computational complexity more than the classical propositional satisfiability.

Acknowledgements

We thank Thomas Eiter for useful discussion on the subject of the paper.

References

[1] C. Aravindan, An abductive framework for negation in disjunctive logic programming, in: Proceedings of the European Workshop on Logics in AI, Lecture Notes in Artificial Intelligence, vol. 1126, Springer, Berlin, 1996, pp. 252–267.

[2] R. Ben-Eliyahu, R. Dechter, Propositional semantics for disjunctive logic programs, Annals of Mathematics and Artificial Intelligence 12 (1&2) (1994) 53–87.

[3] G. Brewka, J. Dix, Knowledge representation with logic programs, in: D.M. Gabbay (Ed.), Handbook of Philosophical Logic, second edition, vol. 6, Oxford University Press, Oxford, 1999 (Chapter 6).

[4] E.P.F. Chan, A possible world semantics for disjunctive databases, IEEE Transactions on Data and Knowledge Engineering 5 (2) (1993) 282–292.

[5] P.T. Cox, T. Pietrzykowski, Causes for events: their computation and applications, in: Proceedings of the Eighth International Conference Automated Deduction, Lecture Notes in Computer Science, vol. 230, Springer, Berlin, 1986, pp. 608–621.

[6] J. Dix, L. Pereira, T. Przymusinski, Prolegomena to logic programming for nonmonotonic reasoning, in: Nonmonotonic Extensions of Logic Programming, Lecture Notes in Artificial Intelligence, vol. 1216, Springer, Berlin, 1997, pp. 1–36.

[7] P.M. Dung, Acyclic disjunctive logic programs with abductive procedure as proof procedure, in: Proceedings of the International Conference on Fifth Generation Computer Systems, Ohm-sha, Tokyo, 1992, pp. 555–561.

[8] T. Eiter, G. Gottlob, The complexity of logic-based abduction, Journal of ACM 42 (1) (1995) 3–42.
[9] T. Eiter, G. Gottlob, On the computational cost of disjunctive logic programming: propositional case, Annals of Mathematics and Artificial Intelligence 15 (3&4) (1995) 289–323.
[10] T. Eiter, G. Gottlob, N. Leone, Abduction from logic programs: semantics and complexity, Theoretical Computer Science 189 (1&2) (1997) 129–177.
[11] T. Eiter, G. Gottlob, H. Mannila, Disjunctive datalog, ACM Transactions on Database Systems 22 (3) (1997) 364–413.
[12] K. Eshghi, R.A. Kowalski, Abduction compared with negation by failure, in: Proceedings of the Sixth International Conference on Logic Programming, MIT Press, Cambridge, MA, 1989, pp. 234–254.
[13] J.A. Fernandez, J. Lobo, J. Minker, V.S. Subrahmanian, Disjunctive LP+integrity constraints = stable model semantics, Annals of Mathematics and Artificial Intelligence 8 (3&4) (1993) 449–474.
[14] M. Gelfond, V. Lifschitz, The stable model semantics for logic programming, in: Proceedings of the Fifth International Conference and Symposium on Logic Programming, MIT Press, Cambridge, MA, 1988, pp. 1070–1080.
[15] M. Gelfond, V. Lifschitz, Classical negation in logic programs and disjunctive databases, New Generation Computing 9 (3&4) (1991) 365–385.
[16] M. Gelfond, Epistemic approach to formalization of commonsense reasoning, Technical Report TR-91-2, University of Texas at El Paso, 1991.
[17] J.E. Hopcroft, J.D. Ullman, Introduction to Automata Theory, Languages and Computation, Addison-Wesley, Reading, MA, 1979.
[18] K. Inoue, M. Koshimura, R. Hasegawa, Embedding negation as failure into a model generation theorem prover, in: Proceedings of the 11th International Conference on Automated Deduction, Lecture Notes in Artificial Intelligence, vol. 607, Springer, Berlin, 1992, pp. 400–415.
[19] K. Inoue, Hypothetical reasoning in logic programs, Journal of Logic Programming 18 (3) (1994) 191–227.
[20] K. Inoue, C. Sakama, On positive occurrences of negation as failure, in: Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning, Morgan Kaufmann, Los Altos, CA, 1994, pp. 293–304 (extended version: Negation as failure in the head, Journal of Logic Programming 35 (1) (1998) 39–78).
[21] K. Inoue, C. Sakama, Transforming abductive logic programs to disjunctive programs, in: Proceedings of the 10th International Conference on Logic Programming, MIT Press, Cambridge, MA, 1993, pp. 335–353 (extended version: A fixpoint characterization of abductive logic programs, Journal of Logic Programming 27 (2) (1996) 107–136).
[22] A.C. Kakas, P. Mancarella, Generalized stable models: a semantics for abduction, in: Proceedings of the Ninth European Conference on Artificial Intelligence, Pitman, 1990, pp. 385–391.
[23] A.C. Kakas, R.A. Kowalski, F. Toni, Abductive logic programming, Journal of Logic and Computation 2 (6) (1992) 719–770.
[24] A.C. Kakas, R.A. Kowalski, F. Toni, The role of abduction in logic programming, in: D.M. Gabbay, C.J. Hogger, J.A. Robinson (Eds.), Handbook of Logic in Artificial Intelligence and Logic Programming, vol. 5, Oxford University Press, Oxford, 1998, pp. 235–324.
[25] K. Konolige, Abductive theories in artificial intelligence, in: G. Brewka (Ed.), Principles of Knowledge Representation CSLI Publications, California, 1994, pp. 129–152.
[26] V. Lifschitz, H. Turner, From disjunctive programs to abduction, in: Non-Monotonic Extensions of Logic Programming, Lecture Notes in Artificial Intelligence, vol. 927, Springer, Berlin, 1995, pp. 23–42.
[27] J. Lobo, J. Minker, A. Rajasekar, Foundations of disjunctive logic programming, MIT Press, Cambridge, MA, 1992.
[28] W. Marek, M. Truszczynski, Autoepistemic logic, Journal of ACM 38 (3) (1991) 588–619.
[29] J. Minker, On indefinite data bases and the closed world assumption, in: Proceedings of the Sixth International Conference on Automated Deduction, Lecture Notes in Computer Science, vol. 138, Springer, Berlin, 1982, pp. 292–308.
[30] J. Minker, Overview of disjunctive logic programming, Annals of Mathematics and Artificial Intelligence 12 (1&2) (1994) 1–24.
[31] D. Poole, A logical framework for default reasoning, Artificial Intelligence 36 (1) (1988) 27–47.
[32] D. Poole, Explanation and prediction: an architecture for default and abductive reasoning, Computational Intelligence 5 (2) (1989) 97–110.

[33] H.E. Pople, Jr, On the mechanization of abductive logic, in: Proceedings of the Third International Joint Conference on Artificial Intelligence, Morgan Kaufmann, Los Altos, CA, 1973, pp. 147–152.
[34] T.C. Przymusinski, Stable semantics for disjunctive programs, New Generation Computing 9 (3&4) (1991) 401–424.
[35] C. Sakama, Possible model semantics for disjunctive databases, in: Proceedings of the First International Conference on Deductive and Object-Oriented Databases, North-Holland, New York, 1989, pp. 369–383.
[36] C. Sakama, K. Inoue, On the equivalence between disjunctive and abductive logic programs, in: Proceedings of the 11th International Conference on Logic Programming, MIT Press, Cambridge, MA, 1994, pp. 489–503.
[37] C. Sakama, K. Inoue, An alternative approach to the semantics of disjunctive logic programs and deductive databases, Journal of Automated Reasoning 13 (1) (1994) 145–172.
[38] K. Satoh, K. Iwayama, Computing abduction by using the TMS, in: Proceedings of the Eighth International Conference on Logic Programming, MIT Press, Cambridge, MA, 1991, pp. 505–518.
[39] B. Selman, H.J. Levesque, Support set selection for abductive and default reasoning, Artificial Intelligence 82 (1&2) (1996) 259–272.
[40] T. Soininen, I. Niemelä, Developing a declarative rule language for applications in product configuration, in: Proceedings of the First International Workshop on Practical Aspects of Declarative Language, Lecture Notes in Computer Science, vol. 1551, Springer, Berlin, 1999, pp. 305–319.
[41] J.-H. You, L.-Y. Yuan, R. Goebel, Regular extension semantics and disjunctive Eshghi-Kowalski procedure, in: Proceedings of the 1998 Joint International Conference and Symposium on Logic Programming, MIT Press, Cambridge, MA, 1998, pp. 175–189.