Equivalence in Abductive Logic

Katsumi Inoue
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430
Japan
ki@nii.ac.jp

Chiaki Sakama
Wakayama University
Sakaedani, Wakayama 640-8510
Japan
sakama@sys.wakayama-u.ac.jp

Abstract

We consider the problem of identifying equivalence of two knowledge bases which are capable of abductive reasoning. Here, a knowledge base is written in either first-order logic or nonmonotonic logic programming. In this work, we will give two definitions of abductive equivalence. The first one, *explainable equivalence*, requires that two abductive programs have the same explainability for any observation. Another one, *explanatory equivalence*, guarantees that any observation has exactly the same explanations in each abductive framework. Explanatory equivalence is a stronger notion than explainable equivalence. In first-order abduction, explainable equivalence can be verified by the notion of extensional equivalence in default theories. In nonmonotonic logic programs, explanatory equivalence can be checked by means of the notion of relative strong equivalence. We also show the complexity results for abductive equivalence.

1 Introduction

Nowadays, abduction is used in many AI applications, including diagnosis, design, updates, and discovery. Abduction is an important paradigm for problem solving, and is incorporated in programming technologies, i.e., *abductive logic programming* (ALP) [12]. Automated abduction is also studied in the literature as an extension of deductive methods or a part of inductive systems, and its computational properties have also been studied [22; 2; 3; 4].

In this work, we are concerned with such computational issues on abductive reasoning. Despite being a problem-solving paradigm, ALP has a lot of issues which have not been fully understood yet. In particular, there are no concrete methods for (a) evaluation of abductive power in ALP, (b) measurement of efficiency in abductive reasoning, (c) semantically correct simplification and optimization, (d) debugging and verification in ALP, and (e) standardization in ALP. Since all these topics are important for any programming paradigm, the lack of them is a serious drawback of ALP. Then, it can be recognized that all the above issues are related to different notions of identification or *equivalence* in ALP. In particular, the item (c) is related to understanding the semantics of ALP with respect to modularity and *contexts*.

Abduction can be formalized in various logics [13]. Then, we can consider several notions of equivalence in several logics for abduction. In this paper, we will give two definitions of abductive equivalence in two logical frameworks for abduction. Two logics we consider here are *first-order logic* (FOL) and *abductive logic programming* (ALP). The first abductive equivalence, called *explainable equivalence*, requires that two abductive programs have the same explainability for any observation. Another one, *explanatory equivalence*, guarantees that any observation has exactly the same explanations in each abductive framework. Explanatory equivalence is stronger than explainable equivalence.

In this paper, we characterize these two notions of abductive equivalence in terms of other well-known concepts in AI and logic programming. In abduction in first-order logic, we will see that explainable equivalence can be verified by the notion of *equivalence* in *default logic* [18], which is defined for the families of *extensions* of two default theories. On the other hand, abductive equivalence in ALP is more complicated than in the case of FOL due to the nonmonotonicity in logic programs. In fact, equivalence between two abductive logic programs has little been discussed in the literature except that effects of partial deduction in ALP are analyzed in [20]. Here, we will see that explanatory equivalence in ALP can be characterized by the notion of *(relative) strong equivalence* [14; 10; 24].

The rest of this paper is organized as follows. Section 2 presents two definitions for abductive equivalence. Section 3 considers first-order logic as the representation language, while Section 4 considers nonmonotonic logic programming for ALP. Section 5 gives concluding remarks.

2 Abductive Equivalence

We start with the question as to when two abductive frameworks are equivalent. As far as the authors know, there is no answer for such a question in the literature of ALP. There are many parameters which should be considered important in defining equivalence notions in abductive frameworks. In the *world*, both background knowledge and observations are surely essential. In an *agent* who performs abduction, on the other hand, her abductive power must depend on her *logic* (language, syntax, semantics) of background knowledge, observations and hypotheses. Moreover, the quality of abduction is relevant to other parameters such as axioms, inference procedures, logics of explanations, and criteria of best explanations. If we would take all such parameters into account, the task of defining the equivalence notion might become combinatorial and too complex.

In the following, we thus consider a rather simple framework for our problem while we try to hold the essence of equivalence notions as much as possible. First, logic, background knowledge and hypotheses are put as input parameters in each abductive framework. Secondly, a logic of explanations is taken into account in a definition, but its diversity is reflected in different notions of abductive equivalence.

The following definition of abductive frameworks is a standard one [13; 22; 2; 3]. As a notation, $\Sigma \models_L F$ means that a formula F is derived from a set Σ of formulas in a logic L.

Definition 2.1 Let B and H be sets of formulas in some underlying logic L. An *abductive framework* is defined as a triple (L, B, H), where B is called *background knowledge* and each element of H is called a *candidate hypothesis*.

Definition 2.2 Let (L, B, H) be an abductive framework, and O a formula in L, and E a formula belonging to H. We define that E is an *explanation* of an *observation* O in (L, B, H) if $B \cup E \models_L O$ and $B \cup E$ is consistent in L. We say that O is *explainable* in (L, B, H) if it has an explanation in (L, B, H).

We now give two definitions for abductive equivalence. We assume that the underlying logic L is common when two abductive theories are compared.

Definition 2.3 Two abductive frameworks (L, B_1, H_1) and (L, B_2, H_2) are *explainably equivalent* if, for any observation O, there is an explanation of O in (L, B_1, H_1) iff there is an explanation of O in (L, B_2, H_2).

Explainable equivalence requires that two abductive frameworks have the same *explainability* for any observation. Explainable equivalence may reflect a situation that two programs have different knowledge to derive the same goals.

Definition 2.4 Two abductive frameworks (L, B_1, H_1) and (L, B_2, H_2) are *explanatorily equivalent* if, for any observation O, E is an explanation of O in (L, B_1, H_1) iff E is an explanation of O in (L, B_2, H_2).

Explanatory equivalence assures that two abductive frameworks have the same *explanation power* for any observation. Explanatory equivalence is stronger than explainable equivalence as follows.

Proposition 2.1 *If abductive frameworks (L, B_1, H_1) and (L, B_2, H_2) are explanatorily equivalent, then they are explainably equivalent.*

For explanatory equivalence, we can assume that the hypotheses H are common in two abductive frameworks in Definition 2.4, as the following property holds.

Proposition 2.2 *If $A_1 = (L, B_1, H_1)$ and $A_2 = (L, B_2, H_2)$ are explanatorily equivalent, then $H'_1 = H'_2$, where $H'_i = \{h \in H_i \mid B_i \cup \{h\}$ is consistent in $L\}$ for $i = 1, 2$.*

Proof. Assume that $H'_1 \setminus H'_2 \neq \emptyset$. Then, for a formula $\varphi \in H'_1 \setminus H'_2$, $\{\varphi\}$ is an explanation of φ in A_1 because $B_1 \cup \{\varphi\}$ is consistent in L. However, $\{\varphi\}$ is not an explanation of φ in A_2. Hence, A_1 and A_2 are not explanatorily equivalent. □

Note in Proposition 2.2 that any hypothesis h in $H_i \setminus H'_i$ cannot be added without violating the consistency of $B_i \cup \{h\}$ in L. Thus, H'_i is the set of hypotheses that can be actually used in explanations of some formulas.

Example 2.1 Suppose the abductive frameworks $A_1 = (\text{FOL}, \{a \supset p\}, \{a, b\})$ and $A_2 = (\text{FOL}, \{b \supset p\}, \{a, b\})$. Then, A_1 and A_2 are explainably equivalent, but are not explanatorily equivalent. On the other hand, $A_3 = (\text{FOL}, \{a \supset p\}, \{b\})$ and $A_4 = (\text{FOL}, \{b \supset p\}, \{b\})$ are neither explainably equivalent nor explanatorily equivalent.

3 Abduction in First-order Logic

Abduction is used in many AI applications, and classical first-order logic is most often used as the underlying logic for abduction [17; 13; 2; 22]. When the underlying logic L is FOL, the relation $\models_L$ becomes the usual entailment relation $\models$. In first-order abduction, explanations are usually defined as a set of ground instances from hypotheses [17]. That is, a set E of ground instances of elements of H is an *explanation* of O in (FOL, B, H) if $\Sigma \cup E \models O$ and $\Sigma \cup E$ is consistent.

In the following, $Th(\Sigma)$ denotes the set of logical consequences of a set Σ of first-order formulas. That is, $Th(\Sigma) = \{F \mid \Sigma \models F\}$. The next definition is originally given for *default logic* by Reiter [18].

Definition 3.1 [19; 17] Let B and H be sets of first-order formulas. An *extension of B with respect to H* is $Th(B \cup S)$ where S is a maximal subset of ground instances of elements from H such that $B \cup S$ is consistent.

Using the notion of extensions, explainable equivalence can be characterized in first-order abduction.

Theorem 3.1 *Two abductive frameworks (FOL, B_1, H_1) and (FOL, B_2, H_2) are explainably equivalent iff the extensions of B_1 with respect to H_1 coincide with the extensions of B_2 with respect to H_2.*

Proof. First, we claim that the union of the extensions of B with respect to H are exactly the set of formulas explainable in (FOL, B, H). To see this, we can use a well-known theorem [17; 22] that a formula O can be explained in (FOL, B, H) iff there is a consistent extension X of B with respect to H such that X contains O. Thus, the set of all explainable formulas are precisely those formulas contained in at least one extension of B with respect to H.

Now, let $A_1 = (\text{FOL}, B_1, H_1)$ and $A_2 = (\text{FOL}, B_2, H_2)$ be two abductive frameworks. Suppose that the extensions of B_1 with respect to H_1 coincide with those of B_2 with respect to H_2. By the above claim, the set of formulas explainable in A_1 is equal to the set of formulas explainable in A_2. This means that A_1 and A_2 are explainably equivalent.

Conversely, assume that there is an extension X_2 of B_2 with respect to H_2 which is not an extension of B_1 with respect to H_1. Let F_{X_2} be a first-order formula which is logically equivalent to X_2. Such a formula actually exists because $X_2 = Th(B_2 \cup S)$ holds for some maximally consistent subset S of H_2, and hence X_2 is logically equivalent to $\bigwedge_{f \in B_2} f \wedge \bigwedge_{g \in S} g$. Since X_2 is consistent, F_{X_2} is consistent too. Then, F_{X_2} is explainable in A_2 because S is an explanation of F_{X_2}.

Now, if F_{X_2} is not explainable in A_1, then obviously A_1 and A_2 are not explainably equivalent. Hence, there is an explanation of F_{X_2} in A_1. Then, there is an extension X_1 of B_1 with respect to H_1 which contains F_{X_2}. Since X_2 is not an extension of B_1 with respect to H_1, $X_1 \neq X_2$ holds. Then, $X_2 \subset X_1$. Let F_{X_1} be a formula which is logically equivalent to X_1. By the same argument as above, F_{X_1} is explainable in A_1. However, this F_{X_1} cannot be explained in A_2. This is because, if F_{X_1} were explained in A_2, there must be an extension X_2' of B_2 with respect to H_2 such that $X_2 \subset X_2'$, which is impossible because any extension is *orthogonal* to another extension in a default theory [18]. In any case, A_1 and A_2 are not explainably equivalent. □

In [15], Reiter's default theories $\Delta_1 = (D_1, B_1)$ and $\Delta_2 = (D_2, B_2)$ are said to be *equivalent* if the *extensions* of Δ_1 are the same as the extensions of Δ_2. When an abductive framework (FOL, B, H) is given, we can associate a default theory $\Delta = (D, B)$ where D is the set of *prerequisite-free normal defaults* $\{\frac{:d}{d} \mid d \in H\}$ such that there is a one-to-one correspondence between the extensions of Δ and the extensions of B with respect to H [17].

Corollary 3.2 *Two abductive frameworks* (FOL, B_1, H_1) *and* (FOL, B_2, H_2) *are explainably equivalent iff the default theories* (D_1, B_1) *and* (D_2, B_2) *are equivalent where* $D_i = \{\frac{:d}{d} \mid d \in H_i\}$ *for* $i = 1, 2$.

Example 3.1 Suppose two abductive frameworks, $A_1 = (\text{FOL}, B_1, H_1)$ and $A_2 = (\text{FOL}, B_2, H_2)$, where

$$\begin{aligned} B_1 &= \{a \supset p,\ \ b \supset \neg p\}, \\ H_1 &= \{a,\ b,\ a \equiv c,\ b \equiv d,\ p \equiv q\}, \\ B_2 &= \{c \supset q,\ \ d \supset \neg q\}, \text{ and} \\ H_2 &= \{c,\ d,\ a \equiv c,\ b \equiv d,\ p \equiv q\}. \end{aligned}$$

Then, A_1 and A_2 are explainably equivalent. In fact, the two extensions of B_1 with respect to H_1 are $Th(B_1 \cup (H_1 \setminus \{b\})) = Th(\{a, \neg b, c, \neg d, p, q\})$ and $Th(B_1 \cup (H_1 \setminus \{a\})) = Th(\{\neg a, b, \neg c, d, \neg p, \neg q\})$, which are respectively equivalent to the two extensions of B_2 with respect to H_2, $Th(B_2 \cup (H_2 \setminus \{d\}))$ and $Th(B_2 \cup (H_2 \setminus \{c\}))$.

It is interesting to see that we can transform any abductive framework to an explainably equivalent abductive framework whose background theory is empty. The next property is also derived by the representation theory for default logic [15].

Corollary 3.3 *For any abductive framework* (FOL, B, H)*, there is an abductive framework* $(\text{FOL}, \emptyset, H')$ *which is explainably equivalent to* (FOL, B, H).

Proof. Put $H' = \{h \wedge \varphi \mid h \in H\} \cup \{\varphi\}$, where $\varphi = \bigwedge_{f \in B} f$. Then, it holds for any O that, $B \cup E \models O$ iff $E' \models O$ where $E \subseteq H$ and $E' = \{h \wedge \varphi \mid h \in E\} \cup \{\varphi\} \subseteq H'$. □

An abductive framework (L, B, H) is called *compatible* if $B \cup H$ is consistent. Explainable equivalence can be easily verified for compatible frameworks.

Corollary 3.4 *Two compatible abductive frameworks* (FOL, B_1, H_1) *and* (FOL, B_2, H_2) *are explainably equivalent iff* $B_1 \cup H_1 \equiv B_2 \cup H_2$.

An abductive framework $(\text{FOL}, B, \mathcal{L})$ is called *assumption-free* where $\mathcal{L}$ is the set of all literals in the underlying language. It is known that the complexity of finding explanations in assumption-free abductive frameworks is not harder than that in assumption-based frameworks [22; 4]. Explainable equivalence in the assumption-free case can also be simply characterized as follows.

Corollary 3.5 *Two abductive frameworks* $(\text{FOL}, B_1, \mathcal{L})$ *and* $(\text{FOL}, B_2, \mathcal{L})$ *are explainably equivalent iff* $B_1 \equiv B_2$.

Proof. For an assumption-free abductive framework $(\text{FOL}, B, \mathcal{L})$, each extension of B with respect to $\mathcal{L}$ is logically equivalent to a model of B. Hence, explainable equivalence implies that the models of B_1 coincide with the models of B_2, and vice versa. □

For explanatory equivalence in first-order abduction, logical equivalence of background theories is necessary and sufficient.

Theorem 3.6 *Two abductive frameworks* (FOL, B_1, H) *and* (FOL, B_2, H) *are explanatorily equivalent iff* $B_1 \equiv B_2$.

Proof. If $B_1 \equiv B_2$, then for any E and any O, it holds that, $B_1 \cup E \models O$ iff $B_2 \cup E \models O$, and that, $B_1 \cup E$ is consistent iff $B_2 \cup E$ is consistent. Hence, (FOL, B_1, H) and (FOL, B_2, H) are explanatorily equivalent.

Conversely, suppose that (FOL, B_1, H) and (FOL, B_2, H) are explanatorily equivalent. Then, for any formula O and any E from H, it holds that $B_1 \cup E \models O$ iff $B_2 \cup E \models O$. Then, for any E, we have $Th(B_1 \cup E) = Th(B_2 \cup E)$. That is, $B_1 \cup E \equiv B_2 \cup E$ holds for any E. This implies $B_1 \equiv B_2$ when $E = \emptyset$. □

The complexity of abductive equivalence in the propositional case can be given as follows.

Lemma 3.7 [1] *Let* $\Delta = (D, W)$ *be a prerequisite-free normal default theory. Then, a formula* φ *is true in all extensions of* Δ *iff* $\Delta' = (D, W \cup \{\varphi\})$ *and* Δ *are equivalent.*

Theorem 3.8 *Deciding explainable equivalence in propositional abduction is* Π_2^P*-complete.*

Proof. By Corollary 3.2 and Lemma 3.7, cautious reasoning in default logic can be transformed to explainable equivalence via equivalence of default theories. This transformation is obviously feasible in polynomial time. Because cautious reasoning from prerequisite-free normal default theories is Π_2^P-complete [6], the decision problem of explainable equivalence is Π_2^P-hard.

We now prove membership in Π_2^P. Two abductive frameworks $A_1 = (\text{FOL}, B_1, H_1)$ and $A_2 = (\text{FOL}, B_2, H_2)$ are *not* explainably equivalent iff there is an extension of B_1 with respect to H_1 which is not an extension of B_2 with respect to H_2. Given a subset $S \subseteq H_1$ as a guess, deciding if $X = Th(B_1 \cup S)$ is an extension of B_1 with respect to H_1 can be checked by computing the "reduct" S' of H_1 by X and then checking if $B_1 \cup S \equiv B_1 \cup S'$. Here, computing the reduct requires satisfiability tests, and this computation as well as testing logical equivalence can be done in polynomial time with an NP-oracle. Once we know that X is an extension of B_1 with respect to H_1, we need to check if X is not an extension of B_2 with respect to H_2, which can also be done in the same way as the former test. Thus, we can construct a polynomial-time nondeterministic Turing machine with an NP-oracle to decide if A_1 and A_2 are not explainably equivalent. Hence, the original problem is the complement of this, and belongs to $\text{coNP}^{\text{NP}} = \Pi_2^P$. □

Theorem 3.9 *The following decision problems in propositional abduction are coNP-complete.*

(1) *Explainable equivalence of two compatible abductive frameworks.*

(2) *Explainable equivalence of two assumption-free abductive frameworks.*

(3) *Explanatory equivalence of two abductive frameworks.*

Proof. By Corollaries 3.4 and 3.5 and Theorem 3.6, these problems can be solved by checking logical equivalence of two propositional theories, which is coNP-complete. □

4 Abductive Logic Programming

Abductive logic programming (ALP) is another popular formalization of abduction in AI [12; 3]. Background knowledge in ALP is called a *logic program*, and the candidate hypotheses are given as literals called *abducibles*. The most significant difference between abduction in FOL and ALP is that ALP allows the nonmonotonic *negation-as-failure* operator not in background knowledge. In abduction, addition of hypotheses may invalidate explanations of some observations if the background theory is nonmonotonic.

Recall that a *(logic) program* is a set of rules of the form

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

where each L_i is a literal ($n \geq m \geq l \geq k \geq 0$), and not is *negation as failure* (NAF). The symbol ; represents a disjunction. The left-hand side of the rule is the *head*, and the right-hand side is the *body*. A rule with variables stands for the set of its ground instances. In this paper, the semantics of a logic program is given by its *answer sets* [5; 9], while another semantics can be considered as well in ALP [12; 3]. Problem solving by representing knowledge as a logic program and then computing its answer sets is called *answer set programming* (ASP).

Definition 4.1 An *abductive (logic) program* is defined as a pair $\langle P, \mathcal{A} \rangle$, where P is a logic program and $\mathcal{A}$ is a set of literals called *abducibles*. Instead of using the notation $(\text{ALP}, P, \mathcal{A})$, we also use $\langle P, \mathcal{A} \rangle$ to represent an abductive framework whose underlying logic is ALP.

Definition 4.2 Let $\langle P, \mathcal{A} \rangle$ be an abductive program, and G a conjunction of ground literals called *observations*. A set $E \subseteq \mathcal{A}$ is a *(credulous) explanation* of G in $\langle P, \mathcal{A} \rangle$[1] if every ground literal in G is true in a consistent answer set of $P \cup E$.

[1] Another, *skeptical* notion for explanations is defined as $E \subseteq \mathcal{A}$ such that G is true in all consistent answer sets of $P \cup E$.

Note that both abducibles and observations are restricted to ground literals in ALP. However, it is known for this framework that rules can be allowed in abducibles and that observations can contain NAF formulas as well as literals [8]. Definition 4.2 can also be represented in a different way as follows [8]. A *belief set (with respect to E)* of an abductive program $\langle P, \mathcal{A} \rangle$ is a consistent answer set of a logic program $P \cup E$ where $E \subseteq \mathcal{A}$. Then, $E \subseteq \mathcal{A}$ is an explanation of G if G is true in a belief set of $\langle P, \mathcal{A} \rangle$ with respect to E.

Definition 4.3 Let $A_1 = \langle P_1, \mathcal{A}_1 \rangle$ and $A_2 = \langle P_2, \mathcal{A}_2 \rangle$ be abductive programs. A_1 and A_2 are *explainably equivalent* if, for any ground literal G, G is explainable in A_1 iff G is explainable in A_2. A_1 and A_2 are *explanatorily equivalent* if, for any conjunction of ground literals G, E is an explanation of G in A_1 iff E is an explanation of G in A_2.

Explainable equivalence in ALP guarantees the same explainability for any ground literal as a *single observation*, but it does not matter how each observation is explained. Hence, we do not have to care about whether multiple observations can be *jointly* explained by a common explanation. On the other hand, explanatory equivalence in ALP guarantees that, any *conjunction* (or *set*) *of observations* [2] has exactly the same credulous explanations. Hence, explanatory equivalence implies that any set of abducibles $E \subseteq \mathcal{A}$ should explain the same set of observations in each abductive program.

We now show that explainable equivalence in ALP can be checked by comparing the belief sets of two abductive programs. Because there exist several methods to compute belief sets using ASP [21; 8; 9], checking explainable equivalence is also possible using such methods. In the following, we denote the set of all belief sets of $\langle P, \mathcal{A} \rangle$ as $BS(P, \mathcal{A})$.

Theorem 4.1 *Abductive programs* $\langle P_1, \mathcal{A}_1 \rangle$ *and* $\langle P_2, \mathcal{A}_2 \rangle$ *are explainably equivalent iff*

$$\bigcup_{S \in BS(P_1, \mathcal{A}_1)} S = \bigcup_{S \in BS(P_2, \mathcal{A}_2)} S.$$

Proof. Recall that $E \subseteq \mathcal{A}$ is an explanation of a ground literal G iff G is true in a belief set of $\langle P, \mathcal{A} \rangle$ with respect to E. Then, the set of all explainable literals are precisely those literals contained in some belief sets of $\langle P, \mathcal{A} \rangle$ with respect to some E. Hence, the union of the belief sets of $\langle P, \mathcal{A} \rangle$ are exactly the set of literals explainable in $\langle P, \mathcal{A} \rangle$. Therefore, two abductive programs are explainably equivalent iff the unions of the belief sets of two abductive programs coincide. □

In some case of compatible problems, explanatory equivalence can be easily verified. Here, a logic program is *definite* if every its rule is NAF-free and has exactly one atom in the head and only atoms in the body. A definite program has a unique answer set that is equivalent to its *least model*. An abductive program $\langle P, \mathcal{A} \rangle$ is called *definite* if P is a definite logic program and $\mathcal{A}$ is a set of atoms.

Corollary 4.2 *Two definite abductive programs* $\langle P_1, \mathcal{A}_1 \rangle$ *and* $\langle P_2, \mathcal{A}_2 \rangle$ *are explainably equivalent if the least model of* $P_1 \cup \mathcal{A}_1$ *coincides with that of* $P_2 \cup \mathcal{A}_2$.

[2] We assume that the set of observations includes the special atom $\top$, which represents the empty conjunction of observations. Note that $\top$ is always true in any consistent set of ground literals.

Example 4.1 Given the common set of abducibles $\mathcal{A} = \{a, b\}$ and three logic programs:

$$\begin{array}{rcl} P_1 & = & \{\, p \leftarrow a,\;\; q \leftarrow b \,\}, \\ P_2 & = & \{\, p \leftarrow b,\;\; q \leftarrow a \,\}, \\ P_3 & = & \{\, p \leftarrow\, ,\;\; q \leftarrow a,\;\; \leftarrow a, b \}, \end{array}$$

the three abductive programs $\langle P_i, \mathcal{A} \rangle$ (for $i = 1, 2, 3$) are all explainably equivalent, but none of them are explanatorily equivalent. In particular, the least model of $P_1 \cup \mathcal{A}$ is $\{p, q, a, b\}$, which is identical to that of $P_2 \cup \mathcal{A}$. P_3 is not definite because of the third rule, but $\langle P_3, \mathcal{A} \rangle$ has three belief sets: $\{p\}, \{p, q, a\}, \{p, b\}$, the union of which is equal to that of $\langle P_i, \mathcal{A} \rangle$ for $i = 1, 2$.

Explanatory equivalence in ALP, on the other hand, requires a more semantical notion of logic programming.[3] For this purpose, we need to utilize the concept of equivalence in logic programming and ASP.

There are several notions for equivalence in logic programming, and *weak equivalence* and *strong equivalence* are most well known. We say that two programs are *weakly equivalent* if they simply agree with their answer sets. The notion of weak equivalence is similar to that of *logical equivalence* in FOL and other classical logics. Given two abductive programs $\langle P_1, \mathcal{A} \rangle$ and $\langle P_2, \mathcal{A} \rangle$, however, weak equivalence of P_1 and P_2 is not a sufficient condition for explanatory equivalence of them, and is not even a sufficient condition for explainable equivalence. Weak equivalence is meaningful only when the abducibles are empty.

Proposition 4.3 *Abductive programs $\langle P_1, \emptyset \rangle$ and $\langle P_2, \emptyset \rangle$ are explanatorily equivalent iff P_1 and P_2 are weakly equivalent.*

On the other hand, *strong equivalence* [14] is a more context-sensitive notion for equivalence of logic programs. Two logic programs P_1 and P_2 are said to be *strongly equivalent* if for any additional logic program R, $P_1 \cup R$ and $P_2 \cup R$ have the same answer sets. When we allow NAF in logic programs, weak equivalence is too fragile as a criterion. For example, $\{\, p \leftarrow not\, a \}$ and $\{\, p \leftarrow \;\}$ are weakly equivalent with the same unique answer set $\{p\}$, but adding a to both results in the withdrawal of p in the former. Often, we can restrict the language for additional programs R to some subset $\mathcal{R}$ of the whole language of programs. Then, two programs P_1 and P_2 are said to be *strongly equivalent with respect to* $\mathcal{R}$ if $P_1 \cup R$ and $P_2 \cup R$ have the same answer sets for any program $R \subseteq \mathcal{R}$ [10]. The equivalence notion with such restriction is called *relative strong equivalence* [10; 24]. Using this notion, explanatory equivalence can be characterized as follows.

Theorem 4.4 *Two abductive programs $\langle P_1, \mathcal{A} \rangle$ and $\langle P_2, \mathcal{A} \rangle$ are explanatorily equivalent iff P_1 and P_2 are strongly equivalent with respect to $\mathcal{A}$.*

Proof. Suppose that $\langle P_1, \mathcal{A} \rangle$ and $\langle P_2, \mathcal{A} \rangle$ are explanatorily equivalent. Then, for any conjunction G of literals and any $E \subseteq \mathcal{A}$, it holds that, E is an explanation of G in $\langle P_1, \mathcal{A} \rangle$ iff E is an explanation of G in $\langle P_2, \mathcal{A} \rangle$. The latter equivalence then implies that, for any G and any E, we have that, G is true in a belief set of $\langle P_1, \mathcal{A} \rangle$ with respect to E iff G is true in a belief set of $\langle P_2, \mathcal{A} \rangle$ with respect to E. Then, for any G and any E, G is true in an answer set of $P_1 \cup E$ iff G is true in an answer set of $P_2 \cup E$. That is, for any E and any set S of literals, S is an answer set of $P_1 \cup E$ iff S is an answer set of $P_2 \cup E$. Hence, P_1 and P_2 are strongly equivalent with respect to $\mathcal{A}$. The converse direction can also be proved by tracing the above proof backward. □

[3] Explanatory equivalence of $\langle P_1, \mathcal{A} \rangle$ and $\langle P_2, \mathcal{A} \rangle$ implies $BS(P_1, \mathcal{A}) = BS(P_2, \mathcal{A})$, but the converse does not hold. For example, when $P_1 = \{a \leftarrow\, ,\; p \leftarrow a\}$, $P_2 = \{a \leftarrow not\, a,\; p \leftarrow a\}$ and $\mathcal{A} = \{a\}$, $BS(P_1, \mathcal{A}) = BS(P_2, \mathcal{A}) = \{\{a, p\}\}$. However, $\emptyset$ is an explanation of p, a and $\top$ in $\langle P_1, \mathcal{A} \rangle$, but is not in $\langle P_2, \mathcal{A} \rangle$.

Example 4.2 Given the common set of abducibles $\mathcal{A} = \{a, b\}$, consider three programs

$$\begin{array}{rcl} P_1 & = & \{\, p \leftarrow a,\;\; a \leftarrow b \,\}, \\ P_2 & = & \{\, p \leftarrow a,\;\; p \leftarrow b,\;\; a \leftarrow b \,\}, \\ P_3 & = & \{\, p \leftarrow b,\;\; a \leftarrow b \,\}. \end{array}$$

Then, the three abductive programs $\langle P_i, \mathcal{A} \rangle$ (for $i = 1, 2, 3$) are explainably equivalent. Although $\langle P_1, \mathcal{A} \rangle$ is explanatorily equivalent to $\langle P_2, \mathcal{A} \rangle$, it is not to $\langle P_3, \mathcal{A} \rangle$ [20]. In fact, P_1 and P_2 are strongly equivalent with respect to $\mathcal{A}$, while P_1 and P_3 are not because the addition of a derives p in P_1 but this is not the case in P_3. This example shows that unfold/fold transformation [23] does not preserve explanatory equivalence in ALP [20] even when P_1 and P_2 are definite.

The complexity results of abductive equivalence in ALP are given as follows.

Theorem 4.5 *Deciding explainable equivalence in propositional ALP is in Δ_3^P.*

Proof. Explanation-finding, i.e., deciding if each literal has an explanation in an ALP, is a Σ_2^P-complete problem [3]. To decide explainable equivalence, we need to check if explainability agrees in two abductive frameworks for each literal. Thus we can construct a polynomial-time deterministic Turing machine with an oracle for the explanation-finding problem in order to decide explainable equivalence. Hence, the problem is in $P^{\Sigma_2^P} = \Delta_3^P$. □

Theorem 4.6 *Deciding explainable equivalence in propositional ALP is Π_2^P-hard.*

Proof. The problem contains the case that the abducibles are empty. In this case, explainable equivalence and explanatory equivalence coincide. Then, by Proposition 4.3, the problem reduces to deciding weak equivalence of two logic programs, which is known to be Π_2^P-complete [16]. □

Corollary 4.7 *Explainable equivalence of two definite abductive programs can be decided in polynomial time.*

Theorem 4.8 *Deciding explanatory equivalence in propositional ALP is Π_2^P-complete.*

Proof. From a set $\mathcal{A}$ of literals, we construct a logic program

$$\mu(\mathcal{A}) = \{\delta_l; not\, \delta_l \leftarrow\, ,\; l \leftarrow \delta_l \mid l \in \mathcal{A}\},$$

where δ_l is a new atom which is uniquely associated with l. Then, it can be shown that, P_1 and P_2 are strongly equivalent with respect to $\mathcal{A}$ iff $P_1' = P_1 \cup \mu(\mathcal{A})$ and $P_2' = P_2 \cup \mu(\mathcal{A})$ are weakly equivalent. By Theorem 4.4, explanatory equivalence of $\langle P_1, \mathcal{A} \rangle$ and $\langle P_2, \mathcal{A} \rangle$ reduces to weak equivalence of P_1' and P_2', which is Π_2^P-complete [16]. □

5 Conclusion

We have introduced the notion of abductive equivalence in this paper. We have considered two definitions of abductive equivalence in two logics. Two important differences between FOL and ALP as the underlying logics are that (1) explainability is considered for all formulas in FOL while only literals are considered as observations in ALP, and that (2) nonmonotonicity by NAF appears in ALP while this is not the case in FOL. Intuitively, the restriction of observations to literals in ALP gives more chances for two abductive programs to be equivalent, but the existence of nonmonotonicity in ALP makes comparison of abductive programs more complicated. In each case, we can observe that explanatory equivalence is not computationally harder than explainable equivalence.

In [11], the notion of abductive equivalence in this paper is further applied to *extended abduction* [7], where hypotheses can not only be added to a program but also be removed from the program to explain an observation. In extended abduction, explanatory equivalence can be characterized by the notion of *update equivalence* [10].

We have observed that logical equivalence of background theories in FOL or weak equivalence of logic programs does not simply imply abductive equivalence except for some very simple cases. That is why we need to characterize abductive equivalence in terms of other known concepts in classical or nonmonotonic logics. Having such characterizations in this paper, the next target is to develop transformation techniques which preserve abductive equivalence. In another future work, we can consider other underlying logics for background theories, hypotheses and observations as well as the criteria of best explanations for abductive equivalence.

Acknowledgment

We thank Kazuhisa Makino for his valuable comments.

References

[1] Jürgen Dix. Default theories of Poole-type and a method for constructing cumulative version of default logic. In: *Proceedings of ECAI-92*, pages 289–293, 1992.

[2] Thomas Eiter and George Gottlob. The complexity of logic-based abduction. *J. ACM*, 42(1):3–42, 1995.

[3] Thomas Eiter, George Gottlob and Nicola Leone. Abduction from logic programs: semantics and complexity. *Theoretical Computer Science*, 189:129–177, 1998.

[4] Thomas Eiter and Kazuhisa Makino. Abduction and the dualization problem. In: *Proceedings of the 6th International Conference on Discovery Science*, LNAI 2843, pages 1–20, Springer, 2003.

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

[6] George Gottlob. Complexity results for nonmonotonic logics. *J. Logic and Computation*, 2:397–425, 1992.

[7] Katsumi Inoue and Chiaki Sakama. Abductive framework for nonmonotonic theory change. In: *Proceedings of IJCAI-95*, pages 204–210, Morgan Kaufmann, 1995.

[8] Katsumi Inoue and Chiaki Sakama. A fixpoint characterization of abductive logic programs. *J. Logic Programming*, 27:107–136, 1996.

[9] Katsumi Inoue and Chiaki Sakama. Negation as failure in the head. *J. Logic Programming*, 35(1):39–78, 1998.

[10] Katsumi Inoue and Chiaki Sakama. Equivalence of logic programs under updates. In: *Proceedings of the 9th European Conference on Logics in Artificial Intelligence*, LNAI 3229, pages 174–186, Springer, 2004.

[11] Katsumi Inoue and Chiaki Sakama. On abductive equivalence. In: Lorenzo Magnani, editor, *Model-based Reasoning in Science and Engineering*, submitted, 2005.

[12] A. C. Kakas, R. A. Kowalski and F. Toni. The role of abduction in logic programming. In: D. M. Gabbay, C. J. Hogger and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 5*, pages 235–324, Oxford University Press, 1998.

[13] Hector J. Levesque. A knowledge-level account of abduction (preliminary version). In: *Proceedings of IJCAI-89*, pages 1061–1067, Morgan Kaufmann, 1989.

[14] Vladimir Lifschitz, David Pearce and Agustín Valverde. Strongly equivalent logic programs. *ACM Transactions on Computational Logic*, 2:526–541, 2001.

[15] V. Wiktor Marek, Jan Treur and Mirosław Truszczyński. Representation theory for default logic. *Annals of Mathematics and Artificial Intelligence*, 21:343–358, 1997.

[16] Emilia Oikarinen and Tomi Janhunen. Verifying the equivalence of logic programs in the disjunctive case. In: *Proceedings of the 7th International Conference on Logic Programming and Nonmonotonic Reasoning*, LNAI 2923, pages 180–193, Springer, 2004.

[17] David Poole. A logical framework for default reasoning. *Artificial Intelligence*, 36:27–47, 1988.

[18] Raymond Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

[19] Raymond Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:571–95, 1987.

[20] Chiaki Sakama and Katsumi Inoue. The effect of partial deduction in abductive reasoning. In: *Proceedings of the 12th International Conference on Logic Programming*, pages 383–397, MIT Press, 1995.

[21] Ken Satoh and Noboru Iwayama. Computing abduction by using the TMS. In: *Proceedings of the 8th International Conference on Logic Programming*, pages 505–518, MIT Press, 1991.

[22] Bart Selman and Hector J. Levesque. Support set selection for abductive and default reasoning. *Artificial Intelligence*, 82(1–2):259–272, 1996.

[23] Hisao Tamaki and Taisuke Sato. Unfold/fold transformation of logic programs. In: *Proceedings of the 2nd International Conference on Logic Programming*, pages 127–138, Uppsala University, Uppsala, Sweden, 1984.

[24] Stefan Woltran. Characterizations for relativized notions of equivalence in answer set programming. In: *Proceedings of the 9th European Conference on Logics in Artificial Intelligence*, LNAI 3229, pages 161–173, Springer, 2004.