Characterization of Strongly Equivalent Logic Programs in Intermediate Logics

Dick de Jongh and Lex Hendriks

May 28, 2001

Abstract

The non-classical, nonmonotonic inference relation associated with the stable model semantics for logic programs gives rise to a relationship of *strong equivalence* between logical programs that can be verified in the 3-valued Gödel logic, **G3**, the strongest non-classical intermediate propositional logic (see [10]). In this paper we will show that **KC** (the logic of $\neg p \vee \neg\neg p$), is the weakest intermediate logic for which strongly equivalent logic programs in a language allowing negations are logically equivalent.

1 Introduction

In logic programming fragments of first-order logic are given a computational interpretation. The first and best known example of such a fragment is that of the *Horn clauses*, quantifier-free formulas of the form B or $\bigwedge A_i \rightarrow B$, where the A_i and B are atomic formulas, the basis for the programming language Prolog [9]. A logic program is a finite set of such formulas (called *rules*).

A logic program Π in the language L can be interpreted in first-order logic as a set of sentences in L, by taking the universal closure $\forall\vec{x}R$ of each of the rules R in Π. An alternative method of eliminating the free variables in the rules, is substitution of *ground terms*, terms built from the constants and functions in L. The set of ground terms of L, the *Herbrand Universe* of L, may be used as the domain for models of theories in L, the *Herbrand models*. Replacing each R by the set of all possible substitutions with ground terms, Π is now replaced by a set of quantifier-free sentences in L, Π^H. The rationale behind this is given by the following well known fact (for a proof see for example [4]).

Fact 1 *Let* Π *be a set of universal sentences. The following are equivalent:*

1. Π *has a model*

2. Π *has a Herbrand model*

3. Π^H *is satisfiable in propositional logic*

In this paper we are interested in logic programs as (possibly infinite) sets of propositional formulas. In general, our program rules may include negations and disjunctions and hence the results in this paper extend to disjunctive logic programing as well.

As long as we remain in classical propositional logic, a model w for a program Π can be identified by the set of atoms X valid in w, in our notation $w = \langle X \rangle$.

In logic programming one is interested in constructing a most 'general' model for a program Π (in the language L based on the Herbrand Universe of L). Such a 'most general' model should not identify terms, for example, unless such an identity is implied by Π. For the propositional analogue of programs with Horn clauses as rules, the *minimal Herbrand model* is such a 'most general' model.

In propositional logic an obvious candidate for the most general model of Π is the intersection of all sets of atoms X such that $\langle X \rangle \models \Pi$. For programs with Horn clauses as rules this works fine as can be seen from the following fact.

Fact 2 *Let* $\Pi \subseteq \{A \to B \mid A, B \in [\wedge, \bot, \top]\}$ *and* $X = \bigcap\{Y \mid \langle Y \rangle \models \Pi\}$. *Then* $\langle X \rangle \models \Pi$.

Although the above fact introduces a fragment slightly richer than the language of Horn clauses, it is still easy to prove.

A simple example, like $p \vee q$, proves that there may not be a unique minimal model for Π if disjunctions are allowed in the rules of Π. And even more serious problems arise for the notion of *most general model*, when negations in the head or body of rules of Π are allowed.

Several solutions have been proposed for the semantics of logic programs with disjunctions and negations. The stable model semantics we use in this paper was introduced by Gelfond and Lifschitz in [5]. The main idea[1] is that X is an *answer set* of Π if $\langle X \rangle \models \Pi$ and that if we replace all negations $\neg A$

[1] A more precise definition will be given in the sequel.

in Π by either $\bot$ or $\top$ according to whether $\langle X\rangle \models A$ or not, X is minimal for the resulting program Π^X. The behaviour of negation in this semantics resembles that of the *negation as failure* (or *negation by default*) in many Prolog implementations.

Classical propositional logic, **CPL**, turns out to be too strong for inference sound for stable models. For example the programs p and $\neg\neg p$ are different according to the stable model semantics. But weaker *intermediate logics*, i.e. logics derived from the intuitionistic propositional logic **IPL** by adding axioms that are valid in **CPL**, provide a sound basis for *stable inference.* David Pearce showed in [12] that that the logic **G3**, the 3-valued Gödel logic can be used to prove logic programs *strongly equivalent.* The notion of *strong equivalence* of logic programs was introduced in [10]. Logic programs Π_1 and Π_2 are said to be *strongly equivalent* in the sense of stable model semantics if for every logic program Π, $\Pi_1 \cup \Pi$ and $\Pi_2 \cup \Pi$ have the same stable models. For programs in the language $\{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$ it was shown in [10] that Π_1 and Π_2 are strongly equivalent precisely if they are equivalent in **G3**.

As pointed out in [10], the notion of strong equivalence may be of interest in showing that a part of a program can be replaced by a simpler equivalent part, without effecting the behavior of the whole program or its extensions. Replacing nonclassical, nonmonotonic stable inference by a well-understood monotonic intermediate logic like **G3**, will simplify the verification of such a strong equivalence between logic programs.

The logic **G3**, also known as the Smetanich logic of *here-and-there*, is the intermediate logic of models based on the partially ordered frame $\langle h, t\rangle$ with $h \leq t$ ([12]).

In this paper we will consider the problem of the 'weakest' intermediate logic for which strong equivalence of programs is the sense of stable models. Our main result is that for programs in $\{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$ the weakest intermediate logic for which equivalence of programs equals strong equivalence on stable models is the logic **KC**, axiomatized by $\neg A \vee \neg\neg A$. This remains true is we restrict the language of programs to $\{A \to B \mid A, B \in [\vee, \neg]\}$ or $\{A \to B \mid A, B \in [\wedge, \neg]\}$. But in the language $\{A \to B \mid A, B \in [\wedge, \vee, \bot, \top]\}$ strong equivalence of programs coincides with equivalence in **IPL** itself.

2 Preliminaries

In the language of propositional logic formulas are built from atoms using $\wedge, \vee, \rightarrow$ and $\neg$. Fragments of propositional logic are obtained by restricting the use of atoms and/or the use of the connectives. We will often denote such a fragment by its signature, enumerating the connectives that are allowed in the formulas between square brackets. So $[\wedge, \neg]$ will denote the fragment of formulas built from atomic formulas using only conjunction and negation, and $[\wedge, \vee, \bot, \top]$ the fragment of formulas built with conjunction, disjunction from the atomic formulas and the constants $\top$ and $\bot$.

The Kripke semantics in this paper is fairly standard. Kripke frames will be sets of worlds (or nodes) W with a partial ordering $\leq$ and a *model* M will be such a *frame* $\langle W, \leq \rangle$ with a function $\mathrm{atom}(w)$ mapping each world $w \in W$ to a set of atomic formula, such that if $w \leq v$ then $\mathrm{atom}(w) \subseteq \mathrm{atom}(v)$. Note that Kripke models are not necessarily rooted. A *maximal world* $w \in M$ (i.e. such that for all $v \in M$, $w \leq v$ implies $v = w$) will be called a *terminal node* of M.

For the language of propositional logic the interpretation in a world w of a model M is given by the usual rules, like $w \models_M A \rightarrow B \quad \Leftrightarrow \quad \forall v \geq w(v \not\models_M A \vee v \models_M B)$. If clear from the context which model is meant in $w \models_M A$, we will omit the subscript (and simply write $w \models A$). If T a set of formulas and w a world in Kripke model M, then $w \models T$ iff $w \models A$ for all $A \in T$. We will write $M \models A$ (or $M \models T$) if for all w in M it is true that $w \models A$ (or $w \models T$). A well-known fact about Kripke models is that if $w \models A$ and $w \leq v$ then $v \models A$, which is true for atomic formulas A by the monotony of the function atom but extends to all formulas A.

Intuitionistic propositional logic (**IPL**) is sound and complete for the set of finite Kripke models.

Classical propositional logic (**CPL**) is sound and complete for the set of Kripke models where the partial ordering is identity. Hence a classical model consists of a set of worlds w that may be identified with $\mathrm{atom}(w)$, the set of atomic formulas valid in w. If $\mathrm{atom}(w) = X$ we will denote $m \models A$ often by $\langle X \rangle \models A$. In the case where w is a node in a Kripke model M, $\langle w \rangle$ will denote the classical world with the same set of atoms as the node w (so $\langle w \rangle = \langle \mathrm{atom}(w) \rangle$).

The *intermediate logics* in this paper are logics obtained by adding formulas valid in the classical propositional logic, **CPL**, to the intuitionistic propositional logic, **IPL**, as schemes.

The logic **G3** is defined here simply as the logic sound and complete for models based on the frame $\langle h, t\rangle$ with $h \leq t$. **G3** traditionally is introduced by giving the (3-valued) truth tables for the connectives and alternatively may be obtained by adding one of the following axioms to **IPL**:

1. $(\neg A \to B) \to (((B \to A) \to B) \to B)$
2. $(A \leftrightarrow B) \vee (A \leftrightarrow C) \vee (A \leftrightarrow D) \vee (B \leftrightarrow C) \vee (B \leftrightarrow D) \vee (C \leftrightarrow D)$
3. $A \vee (A \to B) \vee \neg B$
4. $(A \to (((B \to C) \to B) \to B)) \to A) \to A) \wedge (\neg A \vee \neg\neg A)$

Łukasiewicz [11] seems to have been the first to axiomatize **G3**, using axiom 1. The second axiom is Gödel's [6] formula expressing that there are only three truth values. The third is a simplified version of Hosoi's $A \vee \neg A \vee (A \to B) \vee (B \to C)$ [8]. The last axiom is a combination of the iterated Peirce formula (the substitution of the Pierce formula $((B \to C) \to B) \to B$ for B in $((A \to B) \to A) \to A)$) and the axiom for **KC** (see below), together expressing that the logic will be complete with respect to frames of maximal depth 2 and a single terminal node. Clearly $\neg A \vee \neg\neg A$ can also easily be derived from 3 (take $B = \neg A$ and use that $A \to \neg A$ and $\neg A$ are equivalent and $A \vee \neg\neg A$ is equivalent to $\neg\neg A$). For more details see [2].

We will use the notation $\langle Y, X\rangle$ for the Kripke model $\langle h, t\rangle$, with $X = \text{atom}(t)$ and $Y = \text{atom}(h)$.

The intermediate logic **KC** is given by the rules and axioms of the intuitionistic propositional logic **IPL** plus the axiom $\neg A \vee \neg\neg A$. A proof that **KC** is sound and complete with respect to the finite (rooted) Kripke models with a single terminal node can be found in [2].

The **G3** Kripke models are a special kind of **KC** Kripke models, hence by the completeness theorems for **G3** and **KC**, $T \vdash_{\mathbf{KC}} A$ implies $T \vdash_{\mathbf{G3}} A$.

3 Answer sets and stable models

In this section we recall some of the definitions and results from [5] and [10] for programs in the language $\{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$. As our language does allow more complex rules than the usual $A_1 \wedge \ldots \wedge A_m \wedge \neg A_{m+1} \ldots \wedge \neg A_n \to B_1 \vee \ldots \vee B_k$ (as for example in [10]), we will include the proofs to show the results still hold for the generalized rules. In each case we try to state and prove the results for as large a class of formulas as possible.

We will start with some results for programs in the language $\{A \to B \mid A, B \in [\wedge, \vee, \bot, \top]\}$. Examples of rules in this language are: p (i.e. $\top \to p$), $p \wedge q \vee r \to p \wedge r$ and $p \to \bot$. Note that also negations of formulas A in $[\wedge, \vee]$ are allowed if $\neg A$ is written as $A \to \bot$.

Definition 3 *Let Π be a program in $\{A \to B \mid A, B \in [\wedge, \vee, \bot, \top]\}$. A set of atoms X is an* answer set *of Π if for all $Y \subseteq X$ it is true that $\langle Y \rangle \models \Pi \quad \Leftrightarrow \quad Y = X$. If Π is inconsistent its answer set is the set of all atoms.*

A program in $\{A \to B \mid A, B \in [\wedge, \vee, \bot, \top]\}$ may have several answer sets (like for example the program $p \vee q$) and (logically) different programs may have the same answer sets (for example $p \to q$ and $q \to p$ both have the empty set as their only answer set).

Definition 4 *Programs Π_1 and Π_2 in L are called* strongly equivalent *(in L) if for every program Π in L the programs $\Pi_1 \cup \Pi$ and $\Pi_1 \cup \Pi$ have the same answer sets.*

Logical programs in $\{A \to B \mid A, B \in [\wedge, \vee, \bot, \top]\}$ are strongly equivalent if and only if they are equivalent in classical propositional logic.

Theorem 5 *Let $L = \{A \to B \mid A, B \in [\wedge, \vee, \bot, \top]\}$ and let Π_1 and Π_2 be programs in L. Π_1 and Π_2 are strongly equivalent if $\Pi_1 \equiv_{\mathbf{CPL}} \Pi_2$.*

Proof. First assume $\Pi_1 \equiv_{\mathbf{CPL}} \Pi_2$ and let X be an answer set for $\Pi_1 \cup \Pi$. Then for $Y \subset X$ with $\langle Y \rangle \models \Pi_2 \cup \Pi$ we may infer that $\langle Y \rangle \models \Pi_1 \cup \Pi$ and hence $Y = X$. Which proves X is also an answer set for $\Pi_2 \cup \Pi$. Likewise, every answer set for $\Pi_2 \cup \Pi$ can be proven to be an answer set for $\Pi_1 \cup \Pi$ and hence Π_1 and Π_2 are strongly equivalent.

For the other direction, let $\langle X \rangle \models \Pi_1$ and let $\Pi = X$. Observe that X is an answer set for $\Pi_1 \cup \Pi$ and, as Π_2 is strongly equivalent to Π_1, X is also an answer set for $\Pi_2 \cup \Pi$. Which proves $\langle X \rangle \models \Pi_2$. Likewise, every model of Π_2 will be a model of Π_1, which proves $\Pi_1 \equiv_{\mathbf{CPL}} \Pi_2$. $\dashv$

For a more general treatment of negations in logic programs the following *reduction* of a program was introduced in [5].

Definition 6 *Let X be a set of atomic formulas and A a formula. A^X is defined recursively as:*

$$
\begin{array}{lll}
p^X & = p & \textit{if } p \textit{ is atomic} \\
(A \circ B)^X & = A^X \circ B^X & \textit{for } \circ \in \{\wedge, \vee, \rightarrow\} \\
(\neg A)^X & = \left\{ \begin{array}{ll} \bot & \textit{if } \langle X \rangle \models A \\ \top & \textit{otherwise} \end{array} \right. &
\end{array}
$$

For a program $\Pi \subseteq \{A \rightarrow B \mid A, B \in \{A \rightarrow B \mid [\wedge, \vee, \neg]\}$ the reduction Π^X will be a program in $\{A \rightarrow B \mid A, B \in [\wedge, \vee, \bot, \top]\}$.

Definition 7 *Let $\Pi \subseteq \{A \rightarrow B \mid A, B \in [\wedge, \vee, \neg]\}$. A set X of atomic formulas is called an* answer set *for Π if X a minimal model for Π^X, i.e. for all $Y \subseteq X$ we have $\langle Y \rangle \models \Pi^X \quad \Leftrightarrow \quad Y = X$. In case Π is not consistent, its answer set will be taken to be the set of all atomic formulas.*

If we restrict the language to $\{A \rightarrow B \mid A, B \in [\wedge, \vee, \bot, \top]\}$, we have $\Pi^X = \Pi$ and definition 7 coincides with definition 3.

To find a theorem similar to theorem 5 for strong equivalence in $\{A \rightarrow B \mid A, B \in [\wedge, \vee, \neg]\}$, we will use the characterization of answer sets in [12], based on Kripke models for the intermediate logic **G3**.

The following lemma is not only useful in this case but also will have applications in the next section. Recall that for a world w in a Kripke model M, $\langle w \rangle$ denotes the classical model $\langle \text{atom}(w) \rangle$.

Lemma 8 *Let w be a node in a Kripke model M. For $A, B \in [\wedge, \vee, \bot, \top]$, $w \models A$ iff $\langle w \rangle \models A$ and, if $w \models A \rightarrow B$ then $\langle w \rangle \models A \rightarrow B$.*

Proof. First we prove for $A \in [\wedge, \vee, \bot, \top]$ that $w \models A$ iff $\langle w \rangle \models A$. If A is atomic, $\bot$ or $\top$, this is obvious. By induction on the complexity of A, the proof for the cases of conjunction and disjunction is straightforward.

For the second part of the proof, let both A and B be in $[\wedge, \vee, \bot, \top]$. If $\langle w \rangle \not\models A \rightarrow B$ then $\langle w \rangle \models A$ and $\langle w \rangle \not\models B$. By the first part of the lemma then $w \not\models A \rightarrow B$. $\dashv$

As an immediate consequence of lemma 8 we have the following lemma for models of **G3**.

Lemma 9 *For $A, B \in [\wedge, \vee, \bot, \top]$, $\langle Y, X \rangle \models A \rightarrow B$ iff $\langle X \rangle \models A \rightarrow B$ and $\langle Y \rangle \models A \rightarrow B$*

Proof. Assume $\langle Y, X\rangle \models A \to B$. By lemma 8 we may conclude that $\langle X\rangle \models A \to B$ and $\langle Y\rangle \models A \to B$.

For the other direction, assume $\langle X\rangle \models A \to B$ and $\langle Y\rangle \models A \to B$. If $\langle Y, X\rangle \models A$, then $\langle Y\rangle \models A$ and hence $\langle Y\rangle \models B$, which implies $\langle Y, X\rangle \models B$, so $\langle Y, X\rangle \models A \to B$. On the other hand if $\langle Y, X\rangle \not\models A$ then $\langle X\rangle \models A \to B$ immediately implies $\langle Y, X\rangle \models A \to B$. $\dashv$

The next lemma is true for all propositional formulas.

Lemma 10 *For all sets of atoms X and Y such that $Y \subseteq X$ it is true that* $\langle Y, X\rangle \models A \quad \Leftrightarrow \quad \langle Y, X\rangle \models A^X$.

Proof. Observe that $\langle Y, X\rangle \models \neg A \quad \Leftrightarrow \quad \langle X\rangle \not\models A$. As a consequence we have $\langle Y, X\rangle \models \neg A \quad \Leftrightarrow \quad (\neg A)^X = \top \quad \Leftrightarrow \quad \langle Y, X\rangle \models (\neg A)^X$. Hence for all A it is true that $\langle Y, X\rangle \models \neg A \leftrightarrow (\neg A)^X$. This implies, using the definition 6, that for all $A \in [\wedge, \vee, \to, \neg]$ it is true that $\langle Y, X\rangle \models A \leftrightarrow A^X$, from which the lemma immediately follows. $\dashv$

Theorem 11, theorem 12 and corollary 13 restate the main result of [10]. The proofs of these results are repeated here, to show that they remain true for the slightly more general language for logical programs we use in this paper.

Theorem 11 *Let $\Pi \subseteq \{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$ and X a set of atom formulas. X is an an answer set of Π if and only if for all $Y \subseteq X$ it is true that $\langle Y, X\rangle \models \Pi \quad \Leftrightarrow \quad X = Y$.*

Proof. Let X be an answer set for $\Pi \subseteq \{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$. Note that by definition $\langle X\rangle \models \Pi$ and hence also $\langle X, X\rangle \models \Pi$. Let $Y \subseteq X$ and $\langle Y, X\rangle \models \Pi$. Use lemma 10 to conclude that $\langle Y, X\rangle \models \Pi^X$ and lemma 9 to infer $\langle Y\rangle \models \Pi^X$. As X is an answer set of Π this implies by definition that $X = Y$.

For the other direction, assume for all $Y \subseteq X$ it is true that $\langle Y, X\rangle \models \Pi \quad \Leftrightarrow \quad X = Y$. That $\langle X, X\rangle \models \Pi$ implies $\langle X\rangle \models \Pi$. For $Y \subseteq X$ and $\langle Y\rangle \models \Pi^X$ we may infer that $\langle Y, X\rangle \models \Pi^X$, using lemma 9. Again with lemma 10 we conclude $\langle Y, X\rangle \models \Pi$, which by assumption implies that $X = Y$. Which proves that X is an answer set indeed. $\dashv$

Theorem 12 *Let Π_1 and Π_2 be programs in $\{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$. Π_1 and Π_2 are strongly equivalent if and only if they are equivalent in* **G3***, i.e. $\Pi_1 \equiv_{\mathbf{G3}} \Pi_2$.*

Proof. In one direction the theorem is a direct consequence of theorem 11. Let Π_1 and Π_2 be equivalent in **G3**. Assume X is an answer set for $\Pi_1 \cup \Pi$. Let $\langle Y, X\rangle \models \Pi_2 \cup \Pi$. Then, by the completeness theorem for **G3** with respect to the *here-and-there models*, $\langle Y, X\rangle \models \Pi_1 \cup \Pi$ and hence $X = Y$. Which proves X also to be an answer set of $\Pi_2 \cup \Pi$.

For the other direction assume Π_1 and Π_2 are not equivalent in **G3**. So we may assume there is a $\langle Y, X\rangle \models \Pi_1$ such that $\langle Y, X\rangle \not\models \Pi_2$.

If $\langle X\rangle \not\models \Pi_2$, then let $\Pi = X$. Now X is an answer set for $\Pi_1 \cup X$, but not for $\Pi_2 \cup X$, proving Π_1 and Π_2 not to be strongly equivalent.

If on the other hand $\langle X\rangle \models \Pi_2$, apparently $Y \neq X$. Now define $\Pi = \{p \to q \mid p, q \in X \setminus Y\} \cup Y$. Obviously $\langle X\rangle \models \Pi_2 \cup \Pi$ and for all $Z \subseteq X$, if $\langle Z, X\rangle \models \Pi_2 \cup \Pi$ then $Y \subseteq Z$ and $Y \neq X$. By the definition of Π we conclude that $Z = X$. Which proves X to be an answer set for $\Pi_2 \cup \Pi$. As $\langle Y, X\rangle \models \Pi_1 \cup \Pi$ and Y is a proper subset of X, X is not an answer set of $\Pi_1 \cup \Pi$ and Π_1 and Π_2 are not strongly equivalent. $\dashv$

Corollary 13 *Let Π_1 and Π_2 be programs in $\{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$. Π_1 and Π_2 are strongly equivalent if and only if for all $\Pi \subseteq \{p \to q \mid p, q$ atomic$\}$, $\Pi_1 \cup \Pi$ and $\Pi_2 \cup \Pi$ have the same answer sets.*

Proof. By inspection of the proof of theorem 12. $\dashv$

According to the corollary above, the notion of strong equivalence of logic programs may depend on the language for the programs Π_1 and Π_2, but in all sublanguages L of $\{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$, we may use theorem 12, as long as rules of the form $p \to q$ (with p and q atomic) are in L.

4 Stable inference in intermediate logics

The previous section linked strong equivalence of logic programs in stable inference with equivalence in **CPL** (for programs without negations in the head or the body of the rules) or in **G3**. In this section we will for several fragments of propositional logic determine the weakest intermediate logic for which equivalence of programs is implied by strong equivalence in stable inference.

For the fragment $\{A \to B \mid A, B \in [\wedge, \vee, \bot, \top]\}$ we have the following lemma.

Lemma 14 *Let* $L = \{A \to B \mid A, B \in [\wedge, \vee, \bot, \top]\}$ *and* $T \subseteq L$, $C \in L$. *Then* $T \vdash_{\mathbf{CPL}} C \quad \Leftrightarrow \quad T \vdash_{\mathbf{IPL}} C$.

Proof. The direction from **IPL** to **CPL** is trivial. So let us assume $T \vdash_{\mathbf{CPL}} C$. Let M be a Kripke model and w a node in M such that $w \models T$. Using lemma 8 we may conclude $\langle w \rangle \models T$ and hence $\langle w \rangle \models C$. Again, use lemma 8 to prove $w \models C$. This proves that T implies C in Kripke models in general and hence $T \vdash_{\mathbf{IPL}} C$. $\dashv$

Corollary 15 *Let* $L = \{A \to B \mid A, B \in [\wedge, \vee, \bot, \top]\}$ *and* $\Pi_1, \Pi_2 \subseteq L$. *Then* Π_1 *and* Π_2 *are strongly equivalent in* L *iff* $\Pi_1 \equiv_{\mathbf{IPL}} \Pi_2$.

Proof. The corollary is an immediate consequence of theorem 5 and lemma 14. $\dashv$

Strong equivalence in the language $\{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$ will not be the same as equivalence in **IPL**, as for example $\neg p \vee \neg\neg p$ is strongly equivalent to $\top$ (it is a derivable formula in **G3**) and is not derivable in **IPL**. The intermediate logic **KC**, which has $\neg A \vee \neg\neg A$ as its axiom, turns out to be the weakest intermediate logic for which equivalence of programs is implied by strong equivalence in answer set semantics.

Lemma 16 *Let* $L = \{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$, $T \subseteq L$ *and* $C \in L$. *Then* $T \vdash_{\mathbf{KC}} C \quad \Leftrightarrow \quad T \vdash_{\mathbf{G3}} C$.

Proof. Again the direction from **KC** to **G3** is trivial. For the other direction, let $T \nvdash_{\mathbf{KC}} A \to B$ (where $A, B \in [\wedge, \vee, \neg]$). Then for some Kripke model M with a single terminal (i.e. maximal) node t, there is a $w \in M$ such that $w \models T, w \models A$ and $w \not\models B$. We will prove that for the **G3**-model $\langle w, t \rangle$ we have for all formulas $C \in [\wedge, \vee, \neg]$ that $w \models C \quad \Leftrightarrow \quad \langle w, t \rangle \models C$ and for $C, D \in [\wedge, \vee, \neg]$ that $w \models C \to D \quad \Rightarrow \quad \langle w, t \rangle \models C \to D$. As a consequence, $\langle w, t \rangle \models T, \langle w, t \rangle \models A$ and $\langle w, t \rangle \not\models B$, which proves $T \nvdash A \to B$.

The proof that for $C \in [\wedge, \vee, \neg]$ we have $w \models C \quad \Leftrightarrow \quad \langle w, t \rangle \models C$ is by induction. For atomic formulas it is obvious and the cases for conjunctions and disjunctions are trivial. For the case of negation, observe that $w \models \neg C \quad \Leftrightarrow \quad t \not\models C$, and $t \not\models C \quad \Leftrightarrow \quad \langle t \rangle \not\models C \quad \Leftrightarrow \quad \langle w, t \rangle \models \neg C$.

Now let $C, D \in [\wedge, \vee, \neg]$ and $w \models C \to D$. As a consequence $\langle t \rangle \models C \to D$. So if $w \not\models C$ we have (by the above part of the proof) $\langle w, t \rangle \not\models C$ and hence $\langle w, t \rangle \models C \to D$. On the other hand, if $w \models C$, then also $w \models D$ and by the above part of the proof, also $\langle w, t \rangle \models C \to D$. Which proves that $w \models C \to D$ implies $\langle w, t \rangle \models C \to D$. ⊣

Corollary 17 *Let* $L = \{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$ *and* $\Pi_1, \Pi_2 \subseteq L$. *Then* Π_1 *and* Π_2 *strongly equivalent in* L *iff* $\Pi_1 \equiv_{\mathbf{KC}} \Pi_2$.

Corollary 18 **KC** *is the weakest intermediate logic* **L** *such that* $\Pi_1, \Pi_2 \subseteq \{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$ *are strongly equivalent iff* $\Pi_1 \equiv_{\mathbf{L}} \Pi_2$.

Proof. Note that the **KC** axiom $\neg p \vee \neg\neg p$ can be expressed in the language $\{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$. ⊣

That **KC** is the weakest intermediate logic for strong equivalence in almost any language with negation (where negation is taken to be a *negation by default* and strong equivalence defined according to the answer set semantics) can be seen from the following corollary.

Corollary 19 *Let* $L = \{A \to B \mid A, B \in [\neg]\}$ *and* $\Pi_1, \Pi_2 \subseteq L$. *Then* Π_1 *and* Π_2 *strongly equivalent in* L *iff* $\Pi_1 \equiv_{\mathbf{KC}} \Pi_2$.

Proof. **KC** can alternatively be axiomatized as **IPL** plus $\neg A \to B, \neg\neg A \to B \vdash B$. So in the programs $\{q\}$ and $\{\neg p \to q, \neg\neg p \to q\}$ are strongly equivalent and any logic making such programs equivalent will be as strong as **KC**. ⊣

As a consequence also strong equivalence in for example $\{A \to B \mid A, B \in [\wedge, \neg]\}$ and $\{A \to B \mid A, B \in [\vee, \neg]\}$ will coincide with equivalence in **KC**.

Even if we restrict the language further, allowing in the body only atoms or negated atoms and in the head only atoms (apart from simple statements of atoms and negation of atoms), **KC** is still the weakest intermediate logic **L** such that equivalence of programs in **L** is equivalent with strong equivalence.

In logic programming the programs in this restricted language are known as *normal* programs and have historically been most important.

Definition 20 *A* normal *logic program is a finite set of rules* $\bigwedge l_i \to p$*, where the* l_i *are literals (so either atomic or a negation of an atomic formula) and* p *is atomic.*

First we will prove that an alternative axiomatization of **KC**, in the language of the normal programs, is possible.

Lemma 21 $p \wedge r \to s, \neg p \to q, \neg r \to q, \neg s \vdash_{\mathbf{KC}} q$*.*

Proof. Clearly, $\neg p, p \wedge r \to s, \neg p \to q, \neg r \to q, \neg s \vdash q$ (in **IPL**). Reasoning in **IPL**, from the assumptions $\neg\neg p, p \wedge r \to s, \neg p \to q, \neg r \to q, \neg s$ we conclude that r implies $\neg p$ (as p would imply s and we have $\neg s$). This is a contradiction (as we have $\neg\neg p$) and hence we may infer $\neg r$. By $\neg r \to q$ we conclude $\neg\neg p, p \wedge r \to s, \neg p \to q, \neg r \to q, \neg s \vdash q$. As in **KC** we have $\neg p \vee \neg\neg p$, this proves the lemma. $\dashv$

Let $L = \{A \to B \mid A \in \{p \mid p \text{ atomic or } p = \top\} \cup \{\neg p \mid p \text{ atomic}\}, B \in \{p \mid p \text{ atomic or } p = \bot\}$. As derivability in **KC** implies derivability in **G3** we can use lemma 21 to prove that $\Pi_1 = \{p \wedge r \to s, \neg p \to q, \neg r \to q, \neg s\}$ and $\Pi_2 = \{p \wedge r \to s, q, \neg s\}$ are strongly equivalent programs in L.

On the other hand it is easily seen that for each intermediate logic **L** that proves Π_1 and Π_2 equivalent, we have $p \wedge r \to s, \neg p \to q, \neg r \to q, \neg s \vdash_{\mathbf{L}} q$. The following lemma shows that such an **L** has to be **KC**.

Lemma 22 *If* **L** *is the intermediate logic with, apart from the axioms of* **IPL***, the axiom* $(p \wedge r \to s) \wedge (\neg p \to q) \wedge (\neg r \to q) \wedge \neg s \to q$*, then* **L** *is equivalent with* **KC***.*

Proof. That $\vdash_{\mathbf{L}} A$ implies $\vdash_{\mathbf{KC}} A$ is a simple consequence of lemma 21. For the other direction, let $q := \neg p \vee \neg\neg p$, $r := \neg p$ and $s := \bot$ in $(p \wedge r \to s) \wedge (\neg p \to q) \wedge (\neg r \to q) \wedge \neg s \to q$. Note that in **IPL**, $p \wedge \neg p \to \bot, \neg p \to \neg p \vee \neg\neg p, \neg\neg p \to \neg p \vee \neg\neg p$ and obviously also $\neg\bot$ are derivable. Hence the logic **L** has as an axiom $\neg p \vee \neg\neg p$. $\dashv$

The result of the above discussion is summarized in the next corollary.

Corollary 23 *Let* Π_1 *and* Π_2 *be normal logical programs. Then* Π_1 *and* Π_2 *are strongly equivalent iff* $\Pi_1 \equiv_{\mathbf{KC}} \Pi_2$*. Moreover,* **KC** *is the weakest intermediate logic for which equivalency and strong equivalency of normal logic programs coincide.*

Proof. The first part of the corollary is a simple consequence of corollary 17. Lemma 22 implies that **KC** is the weakest intermediate logic for which equivalent normal programs are strongly equivalent, and visa versa.⊣

References

[1] K. Bowen, R. Kowalski (Eds.), *Proceedings of the Fifth International Conference on Logic Programming 2*, MIT Press, Cambridge, MA., 1988.

[2] A. Chagrov and A. M. Zakharyaschev, Modal Logic, Clarendon Press, Oxford, 1997.

[3] J. Dix, L. Pereira, T. Przymusinski (eds), *Non-Monotoneic Extensions of Logic Programming (Lecture Notes in Artificial Intelligence 1216)*, Springer-Verlag, 1997.

[4] K. Doets, From Logic to Logic Programming, The MIT Press, Massachusetts, 1994.

[5] M. Gelfond, V. Lifschitz, The stable model semantics for logic programs, in [1], 1070–1080, 1988.

[6] K. Gödel, Zum intuitionistischen Aussagenkalkül, *Anzeiger der Akademie der Wissenschaften in Wien*, **69**, 65–66, 1932.

[7] A. Hendriks, *Computations in Propositional Logic*, PHD Thesis University of Amsterdam, 1996.

[8] T. Hosoi, The axiomatization of the intermediate propositional systems S_n of Gödel, *Journal of the Faculty of Science of the University of Tokio*, **13**, 183–187, 1966.

[9] R.A. Kowalski, Predicate logic as a programming language, *Proceedings IFIP'74*, North-Holland, 569–574, 1974.

[10] V. Lifschitz, D. Pearce, A. Valverde, Strongly Equivalent Logic Programs, to appear in *ACM Transactions on Computational Logic.*

[11] J. Łukasiewicz, Die Logik und das Grundlagenproblem, *Les Entretiens de Zürich sur les Fondaments et la Méthode des Sciences Mathématiques 6–9*, **12**, Zürich, 82 –100, 1938.

[12] D. Pearce,A new logical characterization of stable models and answer sets, in [3], 57 – 70, 1997.