Characterization of Strongly Equivalent Logic Programs in Intermediate Logics

Dick de Jongh and Lex Hendriks

May 2, 2002

Abstract

The non-classical, nonmonotonic inference relation associated with the answer set semantics for logic programs gives rise to a relationship of *strong equivalence* between logical programs that can be verified in 3-valued Gödel logic, **G3**, the strongest non-classical intermediate propositional logic (see [12]). In this paper we will show that **KC** (the logic obtained by adding axiom $\neg A \vee \neg\neg A$ to intuitionistic logic), is the weakest intermediate logic for which strongly equivalent logic programs in a language allowing negations are logically equivalent.

1 Introduction

In logic programming certain fragments of first-order logic are given a computational meaning. 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 [11]. 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 the substitution of *ground terms*, i.e. 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 Π^H in L. The rationale behind this is given by the following well-known fact (for a proof see for example [5]).

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 programming as well.

We will denote a fragment of the language of propositional logic by enumerating between square brackets the connectives and constants that are allowed in formulas of the fragment. So $[\wedge, \neg]$ will denote the set of formulas built from atomic formulas, using only conjunction and negation. And $[\wedge, \vee, \bot, \top]$ will be the fragment of formulas built with conjunction and disjunction from atomic formulas and the constants $\bot$ and $\top$.

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 $\langle X \rangle \models \Pi$ (i.e., Π is true in the valuation that makes exactly the atoms in X true). For programs with Horn clauses as rules this works fine as can be seen from the following fact.

Fact 2 *Let* $\Pi \subseteq \{A \rightarrow B \mid A, B \in [\wedge, \bot, \top]\}$ *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 $p \vee q$, shows 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 answer set (or 'stable model') semantics we use in this paper was introduced by Gelfond and Lifschitz in [6]. The main idea[1] is that X is an *answer set* of Π if $\langle X \rangle \models \Pi^X$ where Π^X is the program that arises if we replace all negations $\neg A$ in Π by either $\bot$ or $\top$ according to whether $\langle X \rangle \models A$ or not, and that X is minimal in this respect. The behaviour of negation in this semantics resembles that of the *negation as failure* (or *negation by default*) in many Prolog implementations.

We define a logic **L** to be *sound for stable models* or *sound for stable inference* if, whenever $\Pi \vdash_{\mathbf{L}} A$, then the answer sets for $\Pi \cup \{A\}$ are the same as for Π. Classical propositional logic, **CPL**, turns out to be too strong to have this property. For example, the program $\neg\neg p$ does not have answer sets at all, whereas $\neg\neg p \vdash_{\mathbf{CPL}} p$ and $\{\neg\neg p, p\}$ has $\{p\}$ as its answer set.

Logics weaker than classical logic **CPL** can give a solution to this problem, in particular *intermediate logics*, i.e. logics derived from intuitionistic propositional logic **IPL** by adding axioms that are valid in **CPL**, do provide a sound basis for stable inference. David Pearce used in [15] 3-valued Gödel logic **G3** to prove logic programs strongly equivalent. The notion of *strong equivalence* of logic programs was introduced in [12]. 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 answer sets. For programs in the language $\{A \rightarrow B \mid A, B \in [\wedge, \vee, \neg]\}$ it was shown in [12] that Π_1 and Π_2 are strongly equivalent precisely if they are equivalent in **G3**.

As pointed out in [12], 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 affecting 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 strong equivalence between logic programs. The logic **G3**, also known as the Smetanich logic of *here-and-there*, is the intermediate logic whose models are based on the partially ordered frame $\langle h, t \rangle$ with $h \leq t$ ([15], [3], and see section 2).

In this paper we will consider the problem of the 'weakest' intermediate logic **L** for which provable equivalence is the same as strong equivalence in the sense of stable models. In other words, which **L** has the property that logic programs $\Pi_1 \cup \Pi$ and $\Pi_2 \cup \Pi$ have the same answer sets for all

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

Π iff Π_1 and Π_2 are equivalent in **L**, but this property does not hold for any strictly weaker logic. Note that this will dependent on the language one allows for the programs! Our main result is that for programs in the language $\{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$ the weakest intermediate logic for which equivalence of programs equals strong equivalence on stable models is the logic **KC**, axiomatized by adding axiom $\neg A \vee \neg\neg A$ to **IPL**. This logic (also known as *Jankov's logic* or the *logic of the weak law of excluded middle*) was introduced in [10].

Our main result remains true if 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.

Let us note that **G3** is easier to implement than **KC**. This is witnessed by the fact that satisfiability in **G3** is NP and satisfiability in **KC** is $PSPACE$. However, in many particular cases it is easy to see that certain formulas are not derivable in **KC** whereas this is a complex matter for **G3**. This point also shows up when one wants to prove that the disjunctive rule $p \vee q$ is not strongly equivalent to any nondisjunctive rule. It is not clear how this could be done using the characterization of strong equivalence as provable equivalence in **G3** since $\vee$ is definable from the other connectives in **G3**. But with our characterization of strong equivalence as provable equivalence in **KC** it is a rather simple corollary which we will prove at the end of the paper.

2 Preliminaries

In the language of propositional logic formulas are built from atoms (plus possibly constants $\top$ and $\bot$) using $\wedge, \vee, \to, \neg$. Fragments of propositional logic are obtained by restricting the use of atoms, constants and/or the use of the connectives.

The Kripke semantics in this paper is fairly standard.

Definition 3 *A* Kripke frame $\langle W, \leq \rangle$ *is a set of worlds (or nodes)* W *with a partial ordering* $\leq$*. A* model M *will be such a frame together with a function* $atom(w)$ *mapping each world* $w \in W$ *to a set of atomic formulas, such that if* $w \leq v$ *then* $atom(w) \subseteq atom(v)$*.*

Note that Kripke models are not necessarily rooted. A *maximal world* $w \in W$ (i.e. such that for all $v \in W$, $w \leq v$ implies $v = w$) will be called a *terminal node* of W (or M).

For the language of propositional logic the interpretation in a world w of a model M (by which we mean $w \in W$ if $M = \langle W, \leq \rangle$) is given by the usual rules.

Definition 4 *$w \models_M A$ (A is true in M at w) is defined by recursion on the length of A.*

1. $w \models_M p \quad \Leftrightarrow \quad p \in atom(w)$,
2. $w \models_M A \wedge B \quad \Leftrightarrow \quad w \models_M A$ *and* $w \models_M B$,
3. $w \models_M A \vee B \quad \Leftrightarrow \quad w \models_M A$ *or* $w \models_M B$,
4. $w \models_M A \to B \quad \Leftrightarrow \quad \forall v \geq w\, (v \not\models_M A$ *or* $v \models_M B)$.
5. $w \models_M \top$,
6. $w \not\models_M \bot$,

If it is 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 is a set of formulas and w a world in a 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. Thus, $\vdash_{\mathbf{IPL}} A$ iff $M \models A$ for each finite M.

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 world w that may be identified with $atom(w)$, the set of atomic formulas valid in w. If $atom(w) = X$ we will denote $w \models A$ 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 $\langle w \rangle = \langle atom(w) \rangle$).

An *intermediate logic* is a logic obtained by adding formulas valid in **CPL**, to **IPL** as schemes.

3-valued Gödel logic **G3** can defined as the logic sound and complete for models based on the frame $\langle \{h, t\}, \leq \rangle$ with $h \leq t$ (in a short notation: $\langle h, t \rangle$). We will call these models *here-and-there models*. **G3** traditionally is introduced by giving the (3-valued) truth tables for the connectives. The three values correspond in the context of Kripke models of course to the three

sets of nodes that a formula can be true in: $\emptyset, \{t\}, \{h,t\}$. Alternatively **G3** may be obtained by adding e.g. 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 [14] seems to have been the first to axiomatize **G3**, using axiom 1. The second axiom is Gödel's [7] formula expressing that there are only three truth values. The third is a simplified version of Hosoi's axiom $A \vee \neg A \vee (A \to B) \vee (B \to C)$ [9]. The last axiom is a combination of the iterated Peirce formula (the substitution of the Peirce 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 $\neg\neg A$) or the other axioms. For more details see [3].

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

The intermediate logic **KC** is given by the rules and axioms of intuitionistic propositional logic **IPL** plus the axiom $\neg A \vee \neg\neg A$. **KC** is sound and complete with respect to the finite (rooted) Kripke models with a single terminal node ([10], see [3]).

The Kripke models of **G3** are a special kind of **KC**-Kripke models, hence by the soundness and completeness theorems for **G3** and **KC**, provability (from a set of formulas T) in **KC** implies provability (from T) in **G3**: $T \vdash_{\mathbf{KC}} A$ implies $T \vdash_{\mathbf{G3}} A$.

3 Answer sets and stable models

In this section we recall some of the definitions and results from [6] and [13] for programs in the language $\{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$. As in [12] our language allows more complex rules than the usual $A_1 \wedge \ldots \wedge A_m \wedge \neg A_{m+1} \ldots \wedge \neg A_n \to B_1 \vee \ldots \vee B_k$ (conjunctions, disjunctions and negations can be nested). Therefore, 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 5 *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 $\langle Y \rangle \models \Pi \quad \Leftrightarrow \quad Y = X$.*

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 6 *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 $\Pi_2 \cup \Pi$ have the same answer sets.*

Logic programs in $\{A \to B \mid A, B \in [\wedge, \vee, \bot, \top]\}$ are strongly equivalent if and only if (viewed as set of propositional formulas) they are equivalent in classical propositional logic.

Theorem 7 *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 they are equivalent in* **CPL**, *i.e. $\Pi_1 \equiv_{\mathbf{CPL}} \Pi_2$.*

Proof. First assume $\Pi_1 \equiv_{\mathbf{CPL}} \Pi_2$ and let X be an answer set for $\Pi_1 \cup \Pi$. Then for $Y \subseteq X$ 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 [6],[13].

Definition 8 *Let X be a set of atomic formulas and A a formula. A^X is defined recursively as:*

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

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 9 *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 for all $Y \subseteq X$ we have $\langle Y \rangle \models \Pi^X \quad \Leftrightarrow \quad Y = X$.*

If we restrict the language to $\{A \rightarrow B \mid A, B \in [\wedge, \vee, \bot, \top]\}$, we have $\Pi^X = \Pi$ and definition 9 coincides with definition 5.

To find a theorem similar to theorem 7 for strong equivalence in $\{A \rightarrow B \mid A, B \in [\wedge, \vee, \neg]\}$, we will use the characterization of answer sets in [15], 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 $\langle atom(w) \rangle$.

Lemma 10 *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 10 we have the following lemma for models of **G3**.

Lemma 11 *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 $\langle Y, X \rangle \models A \rightarrow B$. By lemma 10 we may conclude that $\langle X \rangle \models A \rightarrow B$ and $\langle Y \rangle \models A \rightarrow B$.

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 $\langle Y, X\rangle \models A \to B$. ⊣

The next lemma is true for all propositional formulas.

Lemma 12 *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 8, 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. ⊣

Theorem 13, theorem 14 and corollary 15 restate the main result of [12].

Theorem 13 *Let $\Pi \subseteq \{A \to B \mid A, B \in [\wedge, \vee, \neg]\}$ and X a set of atomic formulas. X is an an answer set of Π if and only if for all $Y \subseteq X$ it is true that $\langle Y, X\rangle \models \Pi \quad \Leftrightarrow \quad X = Y$.*

Theorem 14 *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. $\Pi_1 \equiv_{\mathbf{G3}} \Pi_2$.*

Corollary 15 *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 $\Pi \subseteq \{p \to q \mid p$ atomic or $p = \top, q$ atomic$\}$, $\Pi_1 \cup \Pi$ and $\Pi_2 \cup \Pi$ have the same answer sets.*

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 14, 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 determine for several fragments of propositional logic 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 16 *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 10 we may conclude $\langle w \rangle \models T$ and hence $\langle w \rangle \models C$. Again, use lemma 10 to prove $w \models C$. This proves that T implies C in Kripke models in general and hence $T \vdash_{\mathbf{IPL}} C$. ⊣

Corollary 17 *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. Is an immediate consequence of theorem 7 and lemma 16. ⊣

Strong equivalence between programs 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, will, in the following, turn out to be the weakest intermediate logic for which equivalence of programs is implied by strong equivalence in answer set semantics.

Lemma 18 *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 $T \nvdash_{\mathbf{G3}} A \to B$.

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 structural 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 $w \models C \rightarrow D$. Since $w \leq t$, $\langle t\rangle \models C \rightarrow D$. So, if $w \not\models C$, we have (by the above part of the proof) $\langle w, t\rangle \not\models C$ and hence $\langle w, t\rangle \models C \rightarrow D$. On the other hand, if $w \models C$, then also $w \models D$ and by the above part of the proof, also $\langle w, t\rangle \models C \rightarrow D$. Which proves that $w \models C \rightarrow D$ implies $\langle w, t\rangle \models C \rightarrow D$. $\dashv$

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

Corollary 20 **KC** *is the weakest intermediate logic* **L** *such that* $\Pi_1, \Pi_2 \subseteq \{A \rightarrow B \mid A, B \in [\wedge, \vee, \neg]\}$ *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 $\{A \rightarrow B \mid A, B \in [\wedge, \vee, \neg]\}$. $\dashv$

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

Proof. **KC** can alternatively be axiomatized as **IPL** plus $((\neg A \rightarrow B) \wedge (\neg\neg A \rightarrow B) \rightarrow B$. In one direction this is clear from the fact that this axiom immediately follows from $\neg A \vee \neg\neg A$, for the other direction substitute $\neg A \vee \neg\neg A$ for B in the axiom, and $\neg A \vee \neg\neg A$ follows. So, the programs $\{q\}$ and $\{\neg p \rightarrow q, \neg\neg p \rightarrow q\}$ are strongly equivalent and any logic making such programs equivalent will be as strong as **KC**. $\dashv$

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

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

Definition 22 *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 normal programs, is possible.

Lemma 23 $p \wedge r \to s, \neg p \to q, \neg r \to q \vdash_{\mathbf{KC}} \neg s \to q$.

Proof. Of course, this can be automatically checked in a tableau system as in [1], but let us do it from scratch. Assume the premises plus $\neg s$. We have to show q in **KC**. By **KC** we have $\neg p$ or $\neg\neg p$. If $\neg p$, q is immediate from $\neg p \to q$. So, we can assume $\neg\neg p$. From $\neg s$ and $p \wedge r \to s$ we obtain $\neg(p \wedge r)$. In **IPL** this gives $p \to \neg r$, but also $\neg\neg p \to \neg r$. With $\neg\neg p$ this gives us $\neg r$ and hence, from $\neg r \to q$, also q. ⊣

Let L be the set of formulas coding normal logic programs. As derivability in **KC** implies derivability in **G3**, we can use lemma 23 to prove that $\Pi_1 = \{p \wedge r \to s, \neg p \to q, \neg r \to q\}$ and $\Pi_2 = \{p \wedge r \to s, \neg p \to q, \neg r \to q, \neg s \to q\}$ 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 $p \wedge r \to s, \neg p \to q, \neg r \to q \vdash_{\mathbf{L}} \neg s \to q$. The following lemma shows that such an **L** has to contain **KC**.

Lemma 24 *If* **L** *is the intermediate logic with, apart from the axioms of* **IPL**, *the axiom* $(p \wedge r \to s) \wedge (\neg p \to q) \wedge (\neg r \to q) \to (\neg s \to q)$, *then* **L** *is equivalent with* **KC**.

Proof. That $\vdash_{\mathbf{L}} A$ implies $\vdash_{\mathbf{KC}} A$ is a simple consequence of lemma 23. For the other direction, let $q := \neg p \vee \neg\neg p$, $r := \neg p$ and $s := \bot$ in $(p \wedge r \to s) \wedge (\neg p \to q) \wedge (\neg r \to q) \to (\neg s \to q)$. All the antecedents as well as $\neg s$ are then derivable, so $\neg p \vee \neg\neg p$ follows. ⊣

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

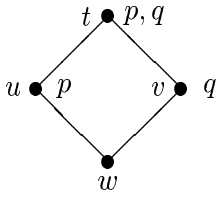
Corollary 25 *Let* Π_1 *and* Π_2 *be normal logical programs. Then* Π_1 *and* Π_2 *are strongly equivalent iff* $\Pi_1 \equiv_{\mathbf{KC}} \Pi_2$. *Moreover,* **KC** *is the weakest intermediate logic for which provable equivalence in the logic and strong equivalence of normal logic programs coincide.*

Proof. The first part immediately follows by corollary 19. Lemma 24 implies that **KC** is the weakest intermediate logic for which equivalent normal programs are strongly equivalent. ⊣

Corollary 26 *No program in the language* $\{A \to B \mid A, B \in [\wedge, \neg]\}$ *is strongly equivalent to the program* $\{p \vee q\}$.

Proof. Recall that **KC** is sound and complete with respect to the finite Kripke models with a single terminal node.

Let the model M be as pictured below, where $atom(t) = \{p, q\}$, $atom(u) = \{p\}$, $atom(v) = \{q\}$ and $atom(w) = \emptyset$.



Clearly $u \models p \vee q$ and $v \models p \vee q$, but $w \not\models p \vee q$. By induction on the complexity of formula $A \in [\wedge, \to, \neg]$ one easily proves that

$$w \models A \quad \Leftrightarrow \quad u \models A \text{ and } v \models A$$

Hence if $\Pi \subset \{A \to B \mid A, B \in [\wedge, \neg]\}$ and $\Pi \equiv_{\mathbf{KC}} p \vee q$, we would have $u \models \Pi$, $v \models \Pi$ and which would imply $w \models \Pi$, a contradiction. ⊣

Observe that the type of model we need for the proof above is not a **G3** model (not of the form $\langle h, t \rangle$). In fact, in the full language of **G3** we can define disjunction using $p \vee q = ((p \to q) \to q) \wedge ((q \to p) \to p)$. The simple proof that this is not possible (in **G3**) if one restricts the language of the programs to $\{A \to B \mid A, B \in [\wedge, \neg]\}$ indicates that the proof of certain properties of answer set programs may benefit from a detour in the logic **KC**.

References

[1] A. Avellone and M. Ferrari and P. Miglioli, Duplication-free tableau calculi and related cut-free sequent calculi for the interpolable propositional intermediate logics, *Logic Journal of the IGPL*, **7**, 447–480, 1999.

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

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

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

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

[6] M. Gelfond, V. Lifschitz, The stable model semantics for logic programs, in [2], 1070–1080, 1988.

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

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

[9] 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.

[10] V. Kowalski, The calculus of the weak "law of excluded middle", *Mathematics of the USSR*, vol. 8, pp. 648–658, 1968.

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

[12] V. Lifschitz, D. Pearce, A. Valverde, Strongly Equivalent Logic Programs, *ACM Transactions on Computational Logic*, **4**, no. 2, 526–541, 2001.

[13] V. Lifschitz, L.R. Tang, H. Turner, Nested expressions in logic programs, *Annals of Mathematics and Artificial Intelligence*, **25**, 369–389, 1999.

[14] 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.

[15] D. Pearce, A new logical characterization of stable models and answer sets, in [4], 57 – 70, 1997.