Abducing through Negation as Failure: Stable models within the independent choice logic

David Poole
Department of Computer Science
University of British Columbia
2366 Main Mall
Vancouver, B.C., Canada V6T 1Z4
poole@cs.ubc.ca
http://www.cs.ubc.ca/spider/poole

May 21, 1998

Abstract

The independent choice logic (ICL) is part of a project to combine logic and decision/game theory into a coherent framework. The ICL has a simple possible-worlds semantics characterised by independent choices and an acyclic logic program that specifies the consequences of these choices. This paper gives an abductive characterization of the ICL. The ICL is defined model-theoretically, but we show that it is naturally abductive: the set of explanations of a proposition g is a concise description of the worlds in which g is true. We give an algorithm for computing explanations and show it is sound and complete with respect to the possible-worlds semantics. What is unique about this approach is that the explanations of the negation of g can be derived from the explanations of g. The use of probabilities over choices in this framework and going beyond acyclic logic programs are also discussed.

1 Introduction

This paper is part of a project aimed at combining logic and decision or game theory into a coherent framework [23]. This project follows from the work on probabilistic Horn abduction [19], which showed how a combination of independent probabilistic hypotheses and a restricted acyclic logic program (without negation as failure) that gives the consequences of the choices, can represent any probability distribution in much the same way as do Bayesian networks [17].

The **independent choice logic (ICL)** [23] builds on probabilistic Horn abduction to extend the logic to include arbitrary acyclic logic programs (that can include negation as failure) and allow different agents to make choices. The general idea is to have a structured hypothesis space, and an acyclic logic program to give the consequences of hypotheses. The hypotheses are partitioned into *alternatives*. The set of all alternatives is a *choice space*. There is a possible world for each selection of one element from each alternative; the logic program specifies what is true in that world. The semantics of negation as failure is given in terms of stable models. This framework is defined model-theoretically but, as we show in this paper, it is naturally abductive: the set of explanations of a proposition g is a concise description of the worlds in which g is true. We give an algorithm for computing explanations and show it is sound and complete. What is unique about this approach is that the explanations of the negation of g can be derived from the explanations of g; the algorithm is based on Reiter's [27] hitting set algorithm.

In other work [23], we consider how this framework can be used for modelling multiple agents under uncertainty in a way that extends decision and game theory. By allowing different agents to make independent choices, the semantic framework extends the notion of the strategic or normal form of a game [29]by allowing for a logic program to model the dynamics of the world and the capabilities of agents. One of the agents can be nature; in this case we have probability distributions over alternatives, with the alternatives corresponding to independent random variables [19].

The goal of abduction [25, 12] is to explain why some observed proposition is true; we want a description of what the (real) world may be like to account for the observation. The input is a set of assumables (possible hypotheses), a logical theory that axiomatises what follows from the assumables, and an observation to be explained. Given an observation we want the explanations to be descriptions of how the world could be to produce the observation: formally we want a consistent set of assumables that logically implies the observations. Thus abduction is inherently about partial information; before we make an observation we don't know which assumptions we will make.

In contrast, negation as failure [5] is about complete knowledge. If some atom cannot be proved, its negation is inferred. In combining negation as failure with abduction, we have complete knowledge about some predicates even though we may only have partial knowledge about others [6, 20, 9]. There are many areas where we want negation as failure to mean that all of the cases for a predicate have been covered even though we may not have complete knowledge about all of the atoms that make up the definition. For example, there is a neat solution to the frame problem using logic programming and negation as failure [14, 1, 28], which can still be used even if we don't have complete knowledge about all of the atoms in the bodies. Consider the following example:

Example 1.1 Consider a simple domain where there is a robot and a key, and the robot can pick up or put down the key, and move to different locations. We can write rules such as, the robot is carrying the key after it has (successfully) picked it up[1]:

$$\begin{aligned}&\mathit{carrying}(\mathit{key}, s(T)) \leftarrow\\&\quad \mathit{do}(\mathit{pickup}(\mathit{key}), T) \wedge\\&\quad \mathit{at}(\mathit{robot}, \mathit{Pos}, T) \wedge\\&\quad \mathit{at}(\mathit{key}, \mathit{Pos}, T) \wedge\\&\quad \mathit{pickup\_succeeds}(T).\end{aligned}$$

Together with this rule that specifies when *carrying* commences, we need a frame rule to specify when carrying persists. The general form of a frame axiom specifies that a fluent is true after a situation if it were true before, and the action were not one that undid the fluent, and there was no other mechanism that undid the fluent. For example, an agent is carrying the key as long as the action was not to put down the key or pick up the key, and the agent did not accidentally drop the key while carrying out another action:[2]

$$\begin{aligned}&\mathit{carrying}(\mathit{key}, s(T)) \leftarrow\\&\quad \mathit{carrying}(\mathit{key}, T) \wedge\\&\quad {\sim}\mathit{do}(\mathit{putdown}(\mathit{key}), T) \wedge\\&\quad {\sim}\mathit{do}(\mathit{pickup}(\mathit{key}), T) \wedge\\&\quad {\sim}\mathit{drops}(\mathit{key}, T).\end{aligned}$$

[1] We are using Prolog's convention with variables in upper case, but with negation written as "$\sim$", and conjunction as "$\wedge$". This axiomatisation is similar to a situation calculus definition, but what whether action is attempted at any time is a proposition. This is closer to the event calculus[15], where we are explicitly interested in narratives [28].

[2] Note that ${\sim}\mathit{do}(\mathit{pickup}(\mathit{key}), T)$ is an element of the body of this rule because we don't want both rules to be applicable when the agent is carrying the key and tries to pick it up. In that case, for the sake of making a choice, we assume the action is like the case when it just picks up the key.

These two rules cover all of the cases when the robot is carrying the key. By these rules, we really mean the completion: the robot is carrying the key if and only if one of these two cases occurs. However, we don't want to globally assume complete knowledge. For example, we may not know whether pickup succeeds or whether the robot drops the key (the ICL [23] allows uncertainty expressed as probabilities for these atoms; see Section 5).

Suppose we have some explanations for why the robot may drop the key[3]. Each such explanation will serve to describe conditions in which the agent drops the key. In all other situations, the agent won't drop the key, and so, assuming the agent is carrying the key and doesn't put down or pickup the key, should serve for explanations for the agent carrying the key in the next state. What distinguished this work is the interaction of abduction and negation as failure. A set of explanations for $drops(key, T)$ will induce another set of explanations for $\sim drops(key, T)$, which then can be used to find explanations for $carrying(key, s(T))$. These, in turn, could be used to derive explanations for $\sim carrying(key, s(T))$. In the independent choice logic, this abductive characterization is a consequence of an independently defined model-theoretic semantics for the logic.

We first overview the ICL and give a model theoretic semantics [23]. The main contribution of this paper is to provide an abductive characterisation of the logic in terms of operations on sets of assumptions (in terms of composite choices) and how the assumptions interact with the logic programs. We show how the abductive machinery can be used for probabilistic reasoning and for Bayesian decision theory, and finally discuss going beyond acyclic logic programs.

2 Background: Acyclic Logic Programs

We use the Prolog conventions with **variables** starting an upper case letter and **constants**, **function symbols**, and **predicate symbols** starting with lower case letters. A **term** is either a variable, a constant, of is of the form $f(t_1, \ldots, t_m)$ where f is a function symbol and $t_1, \ldots, t_m$ are terms. An **atomic formula** (atom) is either a predicate symbol or is of the form $p(t_1, \ldots, t_m)$ where p is a predicate symbol and $t_1, \ldots, t_m$ are terms. A **formula** is either an atom or is of the form $\sim f$,

[3]To contrast this with other work on abductive logic programming [12, 11], consider the case where $drop(key, T)$ isn't a logical consequence of the clauses and no explanations for $drop(key, T)$ involve proofs that use negation as failure. In standard abductive logic programming, no hypotheses need to be added to explain $\sim drops(key, T)$. Any explanations for $drop(key, T)$ serve only to disallow other combinations of assumptions.

$f \wedge g$ or $f \vee g$ where f and g are formulae. A **clause** is either an atom or is a **rule** of the form $a \leftarrow f$ where a is an atom and f is a formula (the **body** of the clause). A **logic program** is a set of clauses.

A **ground** term is a term that does not contain any variables. A ground instance of a term/atom/clause c is a term/atom/clause obtained by uniformly replacing ground terms for the variables in c. The **Herbrand base** is the set of ground instances of the atoms in the language (inventing a new constant if the language does not contain any constants). A **Herbrand interpretation** is an assignment of **true** or **false** to each element of the Herbrand base. If P is a program, let $gr(P)$ be the set of ground instances of elements of P.

Definition 2.1 ([10]) Interpretation $\mathbf{M}$ is a **stable model**[4] of logic program $\mathbf{F}$ if for every ground atom h, h is true in $\mathbf{M}$ if and only if either $h \in gr(\mathbf{F})$ or there is a rule $h \leftarrow b$ in $gr(\mathbf{F})$ such that b is true in $\mathbf{M}$. Conjunction $f \wedge g$ is true in $\mathbf{M}$ if both f and g are true in $\mathbf{M}$. Disjunction $f \vee g$ is true in $\mathbf{M}$ if either f or g (or both) are true in $\mathbf{M}$. Negation $\sim f$ is true in $\mathbf{M}$ if and only if f is not true in $\mathbf{M}$.

Definition 2.2 ([1]) A logic program F is **acyclic** if there is an assignment of a natural number (non-negative integer) to each element of the Herbrand base of F such that, for every rule in $gr(F)$ the number assigned to the atom in the head of the rule is greater than the number assigned to each atom that appears in the body.

Acyclic programs are surprisingly general. Note that acyclicity does not preclude recursive definitions. It just means that all such definitions have to be well founded. They have very nice semantic properties, including the following that are used in this paper:

Theorem 2.3 ([1]) Acyclic logic programs have the following properties:

1. There is a unique stable model.
2. Clark's completion [5] characterises what is true in this model.

Section 7 discusses where we may want to go beyond acyclic programs.

[4] This is a slight generalization of the normal definition of a stable model to include more general bodies in clauses. This is done here because it is easier to describe the abductive operations in terms of the standard logical operators. Note that under this definition $b \leftarrow \sim\sim a$ is the same as $b \leftarrow a$.

3 Independent Choice Logic

In this section we define the semantics of the independent choice logic (ICL) where the base logic is the set of acyclic logic programs under the stable model semantics. Section 4 gives an abductive characterisation of the logic.

Definition 3.1 An **independent choice logic theory** is a pair $\langle \mathbf{C}, \mathbf{F} \rangle$, where

$\mathbf{C}$, the **choice space**, is a set of non-empty sets of ground atomic formulae, such that if $\chi_1 \in \mathbf{C}$, $\chi_2 \in \mathbf{C}$ and $\chi_1 \neq \chi_2$ then $\chi_1 \cap \chi_2 = \{\}$. An element of $\mathbf{C}$ is called an **alternative**. An element of an alternative is called an **atomic choice**.

$\mathbf{F}$, the **facts**, is an acyclic logic program such that no atomic choice unifies with the head of any rule.

In this paper we assume that each alternative is finite, and that there are countably many alternatives.

The semantics is defined in terms of possible worlds. There is a possible world for each selection of one element from each alternative. The atoms which follow from these atoms together with $\mathbf{F}$ are true in this possible world. This is formalised in the next two definitions.

Definition 3.2 Given independent choice logic theory $\langle \mathbf{C}, \mathbf{F} \rangle$, a **selector function** is a mapping $\tau : \mathbf{C} \rightarrow \cup \mathbf{C}$ such that $\tau(\chi) \in \chi$ for all $\chi \in \mathbf{C}$. The **range** of selector function τ, written $\mathbf{R}(\tau)$ is the set $\{\tau(\chi) : \chi \in \mathbf{C}\}$. The range of a selector function is called a **total choice**.

Definition 3.3 Suppose we are given ICL theory $\langle \mathbf{C}, \mathbf{F} \rangle$. For each selector function τ, we construct a **possible world** w_τ. If f is a closed formula, and w_τ is a possible world, f is **true in world** w_τ based on $\langle \mathbf{C}, \mathbf{F} \rangle$, written $w_\tau \models_{\langle \mathbf{C}, \mathbf{F} \rangle} f$, if f is true in the (unique) stable model of $\mathbf{F} \cup \mathbf{R}(\tau)$. f is **false** in world w_τ otherwise.

When understood from context, the $\langle \mathbf{C}, \mathbf{F} \rangle$ is omitted as a subscript of $\models$. The uniqueness of the model follows from the acyclicity of $\mathbf{F} \cup \mathbf{R}(\tau)$.

Note that, for each alternative $\chi \in \mathbf{C}$ and for each world w_τ, there is exactly one element of χ that is true in w_τ. In particular, $w_\tau \models \tau(\chi)$, and $w_\tau \models \sim\alpha$ for all $\alpha \in \chi - \{\tau(\chi)\}$.

4 Abductive Characterisation of the ICL

The semantic framework here, with probabilities on the choices[5] and acyclic logic programs without negation as failure (with some other restrictions that are relaxed here; see Section 8.3) was the basis for probabilistic Horn abduction [19]. One of the main results of [19] (proven in the appendix of that paper) was that the set of minimal explanations of g is a concise description of the possible worlds in which g is true.

In this paper we give an abductive characterisation of the above semantic definition of consequence that allows for negation as failure. In particular, the explanations of a proposition will be a description of the set of possible worlds in which the proposition is true. This can be related to previous work on abductive logic programming [12], but there is a different interaction between abduction and negation as failure. If g has some set of explanations, then $\sim g$ also has a set of explanations which are the *dual* of the explanations of g (Section 4.2). We interpret negation quite differently from the interpretation as "failure to prove" [5] that is appropriate when there is complete knowledge of all propositions. Negation is interpreted with respect to each world; $\sim g$ is true in a world if g is false in the stable model defining that world. This is in contrast to the view that $\sim g$ means that g cannot be proved—there may be many proofs for g (each relying on different assumptions) but this doesn't mean we can't also explain $\sim g$. Section 8.2 discusses the relationship with abductive logic programming in more detail.

4.1 Composite Choices

The notion of a *composite choice* is defined to allow us to partition the worlds according to which atomic choices are true. This forms the basis for the abductive characterisation.

Definition 4.1 A set κ of atomic choices is **consistent** with respect to choice space $\mathbf{C}$ if there is no alternative which contains more than one element of κ. Where $\mathbf{C}$ is understood from context, we just say that κ is consistent. A consistent set of atomic choices is called a **composite choice**. If A is a set of alternatives, a **composite choice on** A is a set of atomic choices that contains exactly one member of each element of A (and no other members).

[5] There is a probability distribution on each alternative (a function $P_0 : \cup\mathbf{C} \rightarrow [0, 1]$ such that for all $\chi \in \mathbf{C}$, $\sum_{\alpha \in \chi} P_0(\alpha) = 1$), where the different alternatives are probabilistically unconditionally independent (see Section 5).

A consistent set of atomic choices is satisfiable: it can always be extended to a total choice, which is true in a possible world.

The following lemma shows the relationship between composite choices and total choices.

Lemma 4.2 A set of atomic choices is a total choice if and only if it is a maximal composite choice.

This is because each total choice is a composite choice, and there is no composite choice that is a (strict) superset. Each maximal composite choice is the range of a selector function and so is a total choice.

The elements of a composite choice are implicitly conjoined: composite choice χ is true in world w_τ, written $w_\tau \models \chi$, if $\chi \subseteq \mathbf{R}(\tau)$. A set of composite choices is implicitly disjoined: a set of composite choices is true in a world w_τ if one of the elements is true in w_τ. Thus a set of composite choices can be seen as a DNF formula made up of atomic choices.

Definition 4.3 Two composite choices are **compatible** if their union is consistent. A set $\mathbf{K}$ of composite choices is **mutually incompatible** if for all $\kappa_1 \in \mathbf{K}, \kappa_2 \in \mathbf{K}$, $\kappa_1 \neq \kappa_2$ implies $\kappa_1 \cup \kappa_2$ is inconsistent.

Given the syntactic definitions of incompatible and mutually incompatible, we can give an equivalent semantic characterisation:

Lemma 4.4 Two composite choices are compatible if and only if there is a world in which they are both true. A set $\mathbf{K}$ of composite choices is mutually incompatible if and only if there is no world in which more than one element of $\mathbf{K}$ is true.

We use sets of composite choices as descriptions of sets of worlds. These are typically much more concise descriptions than describing the possible worlds directly (see section 6). This is used to develop an abductive characterisation of the ICL. In order to develop the theory, we define some operations on composite choices. The first is finding the complement of a set of composite choices (a set of composite choices that describe the complementary sets of worlds to the original set), and the notion of a dual, which is a syntactic operation to find a complementary set of composite choices. We then give an abductive characterisation of the ICL, where explanations of a formula correspond to composite choices that entail the formula. This characterisation shows the interaction between the choices and the rules. In particular, negation as failure is handled using the duals of sets of explanations.

The final operation is splitting composite choices which is useful for making mutually incompatible sets of composite choices that describe the same set of possible worlds as the original.

4.2 Duals and Complements

If $\mathbf{K}$ is a set of composite choices describing a set of worlds, we often want a description of all of the other worlds. This will be crucial in defining abduction through negation as failure; if some set of composite choices describes the worlds in which g is true, then g is false in all of the other worlds, so $\sim g$ is true in these other worlds.

Definition 4.5 If $\mathbf{K}$ is a set of composite choices, then a **complement** of $\mathbf{K}$ is a set $\mathbf{K}'$ of composite choices such that for all worlds w_τ, $w_\tau \models \mathbf{K}'$ iff $w_\tau \not\models \mathbf{K}$.

The notion of a dual will be defined to give a way to construct a complement of a set of composite choices. The idea is that a dual of $\mathbf{K}$ contains choices that are incompatible with every element of $\mathbf{K}$:

Definition 4.6 If $\mathbf{K}$ is a set of composite choices, then composite choice κ' is a **dual** of $\mathbf{K}$ with respect to choice space $\mathbf{C}$ if $\forall \kappa \in \mathbf{K}, \exists \alpha \in \kappa, \exists \alpha' \in \kappa', \alpha \neq \alpha'$, $\exists \chi \in \mathbf{C}$ such that $\{\alpha, \alpha'\} \subset \chi$. A dual is minimal if no proper subset is also a dual. Let $duals_{\mathbf{C}}(\mathbf{K})$ be the set of minimal duals of $\mathbf{K}$ with respect to $\mathbf{C}$. (Usually the choice space $\mathbf{C}$ is implicit from the context.)

Example 4.7 Suppose $\mathbf{C} = \{\{a, b, c\}, \{d, e, f\}\}$, then

$$duals_{\mathbf{C}}(\{\{a, d\}, \{b, e\}\}) = \{\{c\}, \{f\}, \{b, d\}, \{a, e\}\}.$$

The following lemma shows the relationship between duals and negation. The duals of a set $\mathbf{K}$ of composite choices is a description of the complementary set of possible worlds that is described by $\mathbf{K}$:

Lemma 4.8 If $\mathbf{K}$ is a set of composite choices, $duals(\mathbf{K})$ is a complement of $\mathbf{K}$.

In other words, for every world w_τ an element of $\mathbf{K}$ is true in w_τ iff no element of $duals(\mathbf{K})$ is true in w_τ. For a proof of this lemma see Appendix A.

4.2.1 Computing Duals

There is a strong relationship between the idea of a dual of **K** and the notion of a hitting set [27]. Instead of hitting one member of every element of the set, a dual hits a complement of one of the members of each element of **K**.

Definition 4.9 [27, Definition 4.3] Suppose C is a collection of sets. A **hitting set** for C is a set $H \subseteq \bigcup_{S \in C} S$ such that $H \cap S \neq \{\}$ for each $S \in C$.

Definition 4.10 If α is an atomic choice, the **contrary** to α with respect to choice space **C**, written $cont_{\mathbf{C}}(\alpha)$ is $\chi - \{\alpha\}$ where χ is the alternative in **C** which contains α. This is well defined as α is in a unique alternative. If κ is a composite choice, the **contrary** of κ with respect to choice space **C**, written $cont_{\mathbf{C}}(\kappa)$, is $\bigcup_{\alpha \in \kappa} cont_{\mathbf{C}}(\alpha)$.

Example 4.11 Consider example 4.7, where $\mathbf{C} = \{\{a, b, c\}, \{d, e, f\}\}$, then

$$\begin{array}{rcl} cont_{\mathbf{C}}(\{a, d\}) & = & \{b, c, e, f\} \\ cont_{\mathbf{C}}(\{b, e\}) & = & \{a, c, d, f\} \end{array}$$

κ' is a dual of **K**, means that for every element $\kappa \in \mathbf{K}$, κ' contains an element that is contrary to one element of κ. But this is that same as κ' contains an element of $cont_{\mathbf{C}}(\kappa)$. Thus we have:

Theorem 4.12 κ' is a dual of **K** with respect to **C** iff κ' is a consistent hitting set of $\{cont_{\mathbf{C}}(\kappa) : \kappa \in \mathbf{K}\}$.

Reiter's hitting set algorithm [27, Section 4.2] is directly applicable for computing duals. We can prune any branches in the hitting set tree where the corresponding set of atomic choices is inconsistent. This is shown in Figure 1.

Example 4.13 Continuing Example 4.11, the set of consistent minimal hitting sets of $\{\{b, c, e, f\}, \{a, c, d, f\}\}$ is $\{\{c\}, \{f\}, \{b, d\}, \{a, e\}\}$. Although the set $\{b, a\}$ is a hitting set it is inconsistent, and so not considered in the set of duals. It is easy to check that each of the 9 worlds is covered by either $\{\{a, d\}, \{b, e\}\}$ or $\{\{c\}, \{f\}, \{b, d\}, \{a, e\}\}$, but not both.

procedure *duals*(**K**)
  Input: **K** — a set of composite choices
  Output: the set of duals to **K**

  Suppose $\mathbf{K} = \{\kappa_1, \ldots, \kappa_n\}$.
  Let $D_0 = \{\{\}\}$; % *D_i is the set of duals of* $\{\kappa_1, \ldots, \kappa_i\}$
  for $i = 1$ **to** n **do**
    Let $D_i = \{d \cup \{c\} : d \in D_{i-1}, c \in cont_{\mathbf{C}}(\kappa_i)\}$;
    remove inconsistent elements from D_i;
    remove any κ from D_i if $\exists \kappa' \in D_i$ such that $\kappa' \subset \kappa$
  endfor;
  return D_n.

Figure 1: Finding the dual of a set of composite choices **K**

4.3 Entailment

Definition 4.14 If α and β are propositions, α **entails** β with respect to independent choice framework theory $\langle \mathbf{C}, \mathbf{F} \rangle$ if β is true in all worlds in which α is true.

Example 4.15 If $\mathbf{C} = \{\{a, b\}\}$ and $\mathbf{F} = \{c \leftarrow a, d \leftarrow b\}$ then $\sim d$ entails c and d entails $\sim c$. There are two worlds here: one with $a, c, \sim b, \sim d$ true and one with $b, d, \sim a, \sim c$ true.

Entailment can be contrasted with the consequence relation of a logic program:

Definition 4.16 If α is a composite choice, we write $\alpha \mid\!\sim \beta$ if β is true in the stable model of $\mathbf{F} \cup \alpha$.

If $\alpha \mid\!\sim \beta$ then α entails β. The following example shows how entailment in the sense of Definition 4.14 is richer than consequence by $\mid\!\sim$, even when the left-side is a composite choice:

Example 4.17 Suppose $\mathbf{C} = \{\{a, b\}, \{c, d\}\}$ and the facts are:

$$g_1 \leftarrow a \wedge c.$$
$$g_1 \leftarrow b \wedge c.$$

c entails g_1 but $c \not\mid\!\sim g_1$. In every world in which c is true, either a is true or b is true, and so g_1 is also true.

We can characterise entailment in terms of a completion of an ICL theory. This will consist of the completion (in the sense of Clark [5]) of the facts, where we complete all predicates except the atomic choices (similar to [6]). Each alternative gets mapped into a formula expressing the exclusivity and covering of the choices in an alternative.

Definition 4.18 The **completion** of independent choice framework theory $\langle \mathbf{C}, \mathbf{F} \rangle$ is the conjunction of

1. Clark's completion of each predicate that is not an atomic choice.
2. $(\alpha_1 \vee \cdots \vee \alpha_k) \wedge \bigwedge_{i \neq j} \sim(\alpha_i \wedge \alpha_j)$ for each $\{\alpha_1, \ldots, \alpha_k\} \in \mathbf{C}$.
3. Clark's identity theory [5].

The following theorem gives the relationship between entailment and completion. It should not be too surprising as we are restricting $\mathbf{F}$ to be acyclic logic programs, and the completion of each non-atomic choice is true in every world (only the atomic choices in each world changes).

Theorem 4.19 α entails β with respect to independent choice framework theory $\langle \mathbf{C}, \mathbf{F} \rangle$ iff $\alpha \rightarrow \beta$ logically follows from the completion of $\langle \mathbf{C}, \mathbf{F} \rangle$.

For a proof see Appendix A.

Example 4.20 The completion of the ICL theory of example 4.15 is

$$(c \leftrightarrow a) \wedge (d \leftrightarrow b) \wedge (a \vee b) \wedge \sim(a \wedge b)$$

which is equivalent to

$$(c \leftrightarrow a) \wedge (d \leftrightarrow b) \wedge (a \leftrightarrow \sim b)$$

Aside: The theory in this paper could have been presented in the form of these sentences. Unfortunately, given this representation, the assumption that each total choice leads to a single possible world is very sensitive to the form of the sentences. If the language is expanded to much beyond what is the completion of an acyclic ICL theory, choices in some alternatives would constrain choices that can be made in other alternatives (some total choices will be inconsistent with the facts) or a total choice would not completely define a possible world (a total choice together with the facts may imply a disjunction $a \vee b$ without entailing a or b). The formulation in terms of stable models for acyclic logic programs seems very natural and allows for simple semantics.

4.4 Explanations

Definition 4.21 If g is a ground propositional formula, an **explanation** of g is a composite choice that entails g. A **minimal explanation** is an explanation such that no subset is an explanation. A **covering set of explanations** of g is a set of explanations of g such that one element of the set is true in all worlds in which g is true.

A covering set of explanations of g will be true in exactly the worlds in which g is true. This will form a concise description of the worlds in which g is true.

Definition 4.22 If $\mathbf{K}_1$ and $\mathbf{K}_2$ are sets of composite choices, define the conjunction of $\mathbf{K}_1$ and $\mathbf{K}_2$ to be the set of composite choices:

$$\mathbf{K}_1 \otimes \mathbf{K}_2 = \{\kappa_1 \cup \kappa_2 : \kappa_1 \in \mathbf{K}_1, \kappa_2 \in \mathbf{K}_2, \mathit{consistent}(\kappa_1 \cup \kappa_2)\}.$$

It is easy to see that $\mathbf{K}_1 \otimes \mathbf{K}_2$ defines those worlds where both $\mathbf{K}_1$ and $\mathbf{K}_2$ are true. We use the symbol "$\otimes$" as the conjunction is like the cross product, but where we are unioning the pairs and removing inconsistent sets.

This operation is used in the following recursive procedure to compute explanations:

Definition 4.23 If G is a ground propositional formula, $\mathit{expl}(G)$ is the set of composite choices defined recursively as follows:

$$\mathit{expl}(G) = \begin{cases} \mathit{mins}(\mathit{expl}(A) \otimes \mathit{expl}(B)) & \text{if } G = A \wedge B \\ \mathit{mins}(\mathit{expl}(A) \cup \mathit{expl}(B)) & \text{if } G = A \vee B \\ \mathit{duals}(\mathit{expl}(A)) & \text{if } G = \sim A \\ \{\{G\}\} & \text{if } G \in \cup\mathbf{C} \\ \{\} & \text{if } G \in \mathit{gr}(\mathbf{F}) \\ \mathit{mins}(\bigcup_i \mathit{expl}(B_i)) & \text{if } G \notin \cup\mathbf{C}, G \leftarrow B_i \in \mathit{gr}(\mathbf{F}) \end{cases}$$

where $\mathit{mins}(S) = \{\kappa \in S : \forall \kappa' \in S, \kappa' \not\subset \kappa\}$. *duals* is defined in Figure 1. *expl* is well defined as the theory is acyclic.

Note that the set the explanations of a formula is compositional on the explanations on the atomic formulae that make up the formula. In particular, the explanations of $\sim A$ are computed from the explanations of A, and the explanations of $A \wedge B$ are derived from the explanations of A and the explanations of B.

expl can be used directly as a recursive procedure to compute explanations, either top-down or bottom-up. The following theorem establishes the correctness of the *expl* procedure:

Theorem 4.24 Ground formula g is true in world w_τ iff there is some $\kappa \in expl(g)$ such that $\kappa \subseteq \mathbf{R}(\tau)$. Moreover $expl(g)$ is a finite set of finite sets.

For a proof see Appendix A.

Corollary 4.25 $expl(G)$ is a covering set of explanations of G.

Note that $expl(G)$ is not necessarily the set of minimal explanations, as the following example shows.

Example 4.26 Suppose $\mathbf{C} = \{\{a, b\}, \{c, d\}, \{e, f\}\}$ and the facts are:

$$
\begin{aligned}
g_1 &\leftarrow a \wedge c. \\
g_1 &\leftarrow b \wedge c. \\
g_2 &\leftarrow a \wedge c. \\
g_2 &\leftarrow b \wedge e.
\end{aligned}
$$

In this case $expl(g_1) = \{\{a, c\}, \{b, c\}\}$. There is one minimal explanation for g_1, namely $\{c\}$.

Also $expl(g_2) = \{\{a, c\}, \{b, e\}\}$, but the set of minimal explanations of g_2 is $\{\{a, c\}, \{b, e\}, \{c, e\}\}$. $\{c, e\}$ is an explanation, because if c and e were true, whichever of a or b were true in a possible worlds would make g_2 true in that possible world.

This shows that the set of minimal explanations of a goal is not necessarily a minimal covering set of explanations. This idea should be compared to the idea of kernel diagnoses and an irredundant set of kernel diagnoses [8].

The set of minimal diagnoses can be computed using a notion of generalised resolution of explanations:

Definition 4.27 If $\chi = \{\alpha_1, \cdots, \alpha_k\} \in \mathbf{C}$, and $\{L_1, \cdots, L_k\}$ is a set of explanations of g such that $\alpha_i \in L_i$ for each $i \in \{1, \cdots, k\}$, the **generalised resolution** of the explanations $\{L_1, \cdots, L_k\}$ with respect to alternative χ is $L_1 \cup \cdots \cup L_k - \{\alpha_1, \cdots, \alpha_k\}$.

Figure 2 gives an algorithm to repeatedly resolve clauses with respect to alternatives in the choice space and remove redundant clauses. It is similar to the use of binary resolution to compute the prime implicates of a set of clauses [13].

Lemma 4.28 The set of all minimal explanations of g is the set $\mathbf{K}$ resulting from termination of the algorithm of Figure 2.

$\mathbf{K} := mins(expl(g));$
while $\exists \chi \in \mathbf{C}$
$\quad\wedge\ \forall \alpha_i \in \chi\ \exists L_i \in \mathbf{K}$ such that $\alpha_i \in L_i$
$\quad\wedge\ consistent(L_1 \cup \cdots \cup L_k - \chi)$
$\quad\wedge\ \not\exists E \in \mathbf{K}$ such that $E \subseteq L_1 \cup \cdots \cup L_k - \chi$
do $\mathbf{K} := mins(\mathbf{K} \cup \{L_1 \cup \cdots \cup L_k - \chi\})$

Figure 2: Finding all minimal explanations of g

For a proof of this lemma see Appendix A.

Example 4.29 Consider the ICL theory of example 4.26.

To find the minimal explanations of g_1 we start off with $\mathbf{K} = expl(g_1) = \{\{a, c\}, \{b, c\}\}$. As $\{a, b\} \in \mathbf{C}$, we can resolve $\{a, c\}$ and $\{b, c\}$ resulting in $\{c\}$. $mins(\mathbf{K} \cup \{\{c\}\}) = mins(\{\{a, c\}, \{b, c\}, \{c\}\}) = \{\{c\}\}$.

To find the minimal explanations of g_2 we start off with $\mathbf{K} = expl(g_2) = \{\{a, c\}, \{b, e\}\}$. As $\{a, b\} \in \mathbf{C}$, we can resolve $\{a, c\}$ and $\{b, e\}$ resulting in $\{c, e\}$. $mins(\mathbf{K} \cup \{\{c, e\}\}) = mins(\{\{a, c\}, \{b, c\}, \{c, e\}\}) = \{\{a, c\}, \{b, c\}, \{c, e\}\}$. No more resolutions can be carried out, and the procedure stops.

4.5 Splitting Composite Choices

The final operation on (sets of) composite choices is splitting a composite choice into a number of composite choices. This will be used to make sets of mutually incompatible composite choices. This will be used for using explanations to compute probabilities (Section 5), but can be used whenever we do not want redundant proofs. Recall that we are assuming that each alternative is finite.

Definition 4.30 If $\chi = \{\alpha_1, \ldots, \alpha_k\}$ is an alternative and κ is a composite choice such that $\kappa \cap \chi = \{\}$, the **split** of κ on χ is the set of composite choices

$$\{\kappa \cup \{\alpha_1\}, \ldots, \kappa \cup \{\alpha_k\}\}$$

It is easy to see that κ and a split of κ describe the same set of possible worlds:

Lemma 4.31 If w_τ is a possible world, $w_\tau \models \kappa$ iff there is some $\alpha_i \in \chi$ such that $w_\tau \models \kappa \cup \{\alpha_i\}$.

procedure *disjoint*(**K**)
Input: **K** — set of composite choices
Output: mutually incompatible set of composite choices equivalent to **K**

repeat
  if $\exists \kappa_1, \kappa_2 \in \mathbf{K}$ and $\kappa_1 \subset \kappa_2$
    then $\mathbf{K} := \mathbf{K} - \{\kappa_2\}$
  else if $\exists \kappa_1, \kappa_2 \in \mathbf{K}$, such that $\kappa_1 \cup \kappa_2$ is consistent
    then
      choose $\alpha \in \kappa_1 - \kappa_2$ and $\chi \in \mathbf{C}$ such that $\alpha \in \chi$
      let $\mathbf{K}_2$ be the split of κ_2 on χ
      $\mathbf{K} := \mathbf{K} - \{\kappa_2\} \cup \mathbf{K}_2$
  else exit and return **K**
forever

Figure 3: Make set of composite choices **K** mutually incompatible

If there is a finite number of alternatives, starting from set **K** of composite choices, repeated splitting of composite choices can produce the set of total choices (and so possible worlds) in which **K** is true. Such an operation is not, however, of much use.

The main use for splitting is, given a set of composite choices to construct a set of mutually incompatible composite choices that describes the same set of possible worlds as the original set. Suppose **K** is a set of composite choices, there are two operations we consider to form a new set $\mathbf{K}'$ of composite choices:

1. removing dominated elements: if $\kappa_1, \kappa_2 \in \mathbf{K}$ and $\kappa_1 \subset \kappa_2$, let $\mathbf{K}' = \mathbf{K} - \{\kappa_2\}$.

2. splitting elements: if $\kappa_1, \kappa_2 \in \mathbf{K}$, such that $\kappa_1 \cup \kappa_2$ is consistent (and neither is a superset of the other), there is a $\alpha \in \kappa_1 - \kappa_2$ and $\chi \in \mathbf{C}$ such that $\alpha \in \chi$. We replace κ_2 by the split of κ_2 on χ. Let $\mathbf{K}_2$ be the split of κ_2 on χ, and $\mathbf{K}' = \mathbf{K} - \{\kappa_2\} \cup \mathbf{K}_2$.

It is easy to see that **K** and $\mathbf{K}'$ describe the same set of possible worlds.

If we repeat the above two operations until neither is applicable, the procedure stops (if **K** is a finite set of finite composite choices) and we end up with a set of mutually incompatible composite choices that is equivalent to the original set (is true in the same set of possible worlds). This procedure is depicted in Figure 3.

Example 4.32 Suppose

$$\mathbf{C} = \{\{a_1, a_2, a_3\}, \{b_1, b_2, b_3\}, \{c_1, c_2, c_3\}\}.$$

$$\mathbf{K} = \{\{a_1, b_1\}, \{a_1, c_1\}\}.$$

The elements of **K** are not mutually incompatible (there is a world in which they are both true — namely the world with total choice $\{a_1, b_1, c_1\}$). We can split the second element of **K** on $\{b_1, b_2, b_3\}$, resulting in

$$\mathbf{K}' = \{\{a_1, b_1\}, \{a_1, b_1, c_1\}, \{a_1, b_2, c_1\}, \{a_1, b_3, c_1\}\}.$$

The second element can be removed, and we end up with

$$\mathbf{K}'' = \{\{a_1, b_1\}, \{a_1, b_2, c_1\}, \{a_1, b_3, c_1\}\}$$

which is a mutually incompatible set of composite choices equivalent to **K**.

It is possible, even with only two composite choices that we will need to spilt multiple times, as the following example shows:

Example 4.33 Suppose

$$\mathbf{C} = \{\{a_1, a_2\}, \{b_1, b_2\}, \{c_1, c_2\}, \{d_1, d_2\}, \{e_1, e_2\}, \{f_1, f_2\}, \{g_1, g_2\}\}.$$

$$\mathbf{K} = \{\{a_1, b_1, c_1\}, \{a_1, d_1, e_1, f_1\}\}.$$

We can split the second element of **K** on $\{b_1, b_2\}$ resulting in

$$\mathbf{K}' = \{\{a_1, b_1, c_1\}, \{a_1, b_1, d_1, e_1, f_1\}, \{a_1, b_2, d_1, e_1, f_1\}\}.$$

We can split the second element of $\mathbf{K}'$ on $\{c_1, c_2\}$ resulting in

$$\mathbf{K}'' = \{\{a_1, b_1, c_1\}, \{a_1, b_1, c_1, d_1, e_1, f_1\}, \{a_1, b_1, c_2, d_1, e_1, f_1\}, \{a_1, b_2, d_1, e_1, f_1\}\}.$$

The second element of $\mathbf{K}''$ is subsumed, so we can remove it, resulting in

$$\mathbf{K}''' = \{\{a_1, b_1, c_1\}, \{a_1, b_1, c_2, d_1, e_1, f_1\}, \{a_1, b_2, d_1, e_1, f_1\}\},$$

which is a mutually incompatible set of composite choices equivalent to **K**.

A few interesting complexity questions about the procedure of Figure 3 can be answered:

1. How many splits may be needed?
2. How many composite choices are in the resulting set of mutually incompatible composite choices?
3. Is there a heuristic that tells us on which element we should split? In the second operation we can split on κ_1 or on κ_2; does one result in fewer composite choices?

The first thing to notice is that to make a set of composite choices mutually incompatible, it may be the case that we have to consider each pair of composite choices. Given a pair of composite choices we can analyze exactly the number of splits and the number of resultant composite choices.

Suppose we are trying to make composite choices κ_1 and κ_2 incompatible. Let $\mathbf{K}_2$ be the split of κ_2 on χ (where κ_1 contains an element of χ). We must have $|\mathbf{K}_2| = |\chi|$. All but one of the elements of $\mathbf{K}_2$ are incompatible with κ_1 (thus there are $|\chi| - 1$ of these composite choices incompatible with κ_1). Let κ_3 be the element of $\mathbf{K}_2$ compatible with κ_1 (κ_3 will be $(\kappa_1 \cap \chi) \cup \kappa_2$). Either κ_1 is a subset of κ_3 (this occurs iff $|\kappa_1 - \kappa_2| = 1$), or we have have to repeat the loop to make κ_1 and κ_3 incompatible.

Suppose $\kappa_1 - \kappa_2 = \{\alpha_1, \ldots, \alpha_k\}$, where $\alpha_i \in \chi_i$. Either we are going to eventually split κ_2 on $\chi_1, \ldots, \chi_k$, or else we are going to have to split κ_1 on the analogous set, in order to make the resultant set of composite choices mutually incompatible. It makes no sense to split both κ_1 and κ_2 in order to make them incompatible. If we repeatedly split κ_2 on the χ_i we will need $k = |\kappa_1 - \kappa_2|$ splits. The resulting set of composite choices will contain $(\sum_{i=1}^{k} |\chi_i|) - k - 1$ elements.

Thus if all of the alternatives have the same length, to minimise the number of composite choices in the mutually incompatible set we should repeatedly split the larger of a pair of composite choices.

4.6 An Example in Detail

Continuing Example 1.1, suppose we also have that the agent often drops the key if it is slippery and if the key isn't slippery, it sometimes fumbles and drops the key:

$$
\begin{aligned}
&\mathit{drops}(\mathit{key}, T) \leftarrow \\
&\quad \mathit{slippery}(\mathit{key}, T) \wedge \\
&\quad \mathit{drop\_slippery\_key}(T).
\end{aligned}
$$

$$
\begin{aligned}
& drops(key, T) \leftarrow \\
& \quad \sim slippery(key, T) \wedge \\
& \quad fumbles\_key(T).
\end{aligned}
$$

Suppose that, independently at each time, the agent either drops or holds a slippery key and either fumbles or retains an unslippery key. This is specified by:

$$
\begin{aligned}
& \forall T \ \{drop\_slippery\_key(T), holds\_slippery\_key(T)\} \in \mathbf{C} \\
& \forall T \ \{fumbles\_key(T), retains\_key(T)\} \in \mathbf{C}
\end{aligned}
$$

Suppose that the key could start slippery and subsequently become unslippery. (We could model the key becoming slippery by adding an extra clause, but this makes the example more complicated.)

$$
\begin{aligned}
& slippery(key, s(T)) \leftarrow \\
& \quad slippery(key, T) \wedge \\
& \quad stays\_slippery(T). \\
& slippery(key, 0) \leftarrow \\
& \quad initially\_slippery(key).
\end{aligned}
$$

Whether the key remains slippery at each step and whether it is initially slippery are both choices:

$$
\begin{aligned}
& \forall T \ \{stays\_slippery(T), stops\_being\_slippery(T)\} \in \mathbf{C} \\
& \forall T \ \{initially\_slippery(key), initially\_unslippery(key)\} \in \mathbf{C}
\end{aligned}
$$

We can axiomatise the location of the robot and the key in a similar manner. The robot, goes to the location of the action if the move was successful, otherwise it stays still. The key stays where it is unless it is being carried in which case it is at the location of the robot.

$$
\begin{aligned}
& at(robot, Pos, s(T)) \leftarrow \\
& \quad do(goto(Pos), T) \wedge \\
& \quad goto\_succeeds(T). \\
& at(robot, Pos1, s(T)) \leftarrow \\
& \quad do(goto(Pos), T) \wedge \\
& \quad at(robot, Pos1, T) \wedge \\
& \quad \sim goto\_succeeds(T).
\end{aligned}
$$

$$at(robot, Pos, s(T)) \leftarrow$$
$$\sim goto\_action(T) \wedge$$
$$at(robot, Pos, T).$$
$$at(key, Pos, T) \leftarrow$$
$$carrying(key, T) \wedge$$
$$at(robot, Pos, T).$$
$$at(key, Pos, s(T)) \leftarrow$$
$$\sim carrying(key, s(T)) \wedge$$
$$at(key, Pos, T).$$

There is only one *goto* action.[6]

$$goto\_action(T) \leftarrow$$
$$do(goto(Pos), T).$$

The *goto* action either succeeds or fails at each time.

$$\forall T\ \{goto\_succeeds(T), goto\_fails(T)\} \in \mathbf{C}$$

Consider the following particular scenario of actions:

$$do(goto(loc1), 0).$$
$$do(pickup(key), s(0)).$$
$$do(goto(loc2), s(0)).$$
$$at(key, loc1, 0).$$
$$at(robot, loc0, 0).$$

Example 4.34 There is one explanation for $slippery(key, s(s(0)))$:

$$[stays\_slippery(s(0)), stays\_slippery(0), initially\_slippery(key)]$$

Example 4.35 The explanation for $slippery(key, s(s(0)))$ has three duals:

$$[stops\_being\_slippery(s(0))]$$
$$[stops\_being\_slippery(0)]$$
$$[initially\_unslippery(key)]$$

[6]This is needed as we don't have explicit quantification, and all variables are universally quantified outside the scope of the clause. Existentially quantified variables in the scope of a negation can be handled by introducing a new clause.

These thus form the explanations for $\sim$*slippery*(*key*, *s*(*s*(0))). These can be made disjoint giving the explanations:

[*initially_slippery*(*key*), *stays_slippery*(0), *stops_being_slippery*(*s*(0))]
[*initially_slippery*(*key*), *stops_being_slippery*(0)]
[*initially_unslippery*(*key*)]

Example 4.36 Consider the explanations for *drops*(*key*, *s*(*s*(0))). The first clause for *drops* results in one explanation:

[*drop_slippery_key*(*s*(*s*(0))), *stays_slippery*(*s*(0)), *stays_slippery*(0),
initially_slippery(*key*)]

The second clause for *drops* results in three more explanations for *drops*(*key*, *s*(*s*(0))), namely:

[*fumbles_key*(*s*(*s*(0))), *initially_slippery*(*key*), *stays_slippery*(0),
stops_being_slippery(*s*(0))]
[*fumbles_key*(*s*(*s*(0))), *initially_slippery*(*key*), *stops_being_slippery*(0)]
[*fumbles_key*(*s*(*s*(0))), *initially_unslippery*(*key*)]

Example 4.37 There are four explanations for $\sim$*drops*(*key*, *s*(*s*(0))):

[*initially_slippery*(*key*), *stays_slippery*(0), *retains_key*(*s*(*s*(0))),
stops_being_slippery(*s*(0))]
[*initially_slippery*(*key*), *retains_key*(*s*(*s*(0))), *stops_being_slippery*(0)]
[*retains_key*(*s*(*s*(0))), *initially_unslippery*(*key*)]
[*initially_slippery*(*key*), *stays_slippery*(0), *stays_slippery*(*s*(0)),
holds_slippery_key(*s*(*s*(0)))]

Example 4.38 Consider explaining *carrying*(*key*, *s*(*s*(*s*(0)))). Using the second clause for *carrying*, we need to combine these four explanations for $\sim$*drops*(*key*, *s*(*s*(0))) with explanations for *carrying*(*key*, *s*(*s*(0))).

There are four explanations of *carrying*(*key*, *s*(*s*(*s*(0)))):

[*initially_slippery*(*key*), *stays_slippery*(0), *retains_key*(*s*(*s*(0))),
stops_being_slippery(*s*(0)), *pickup_succeeds*(*s*(0)), *goto_succeeds*(0)]
[*initially_slippery*(*key*), *retains_key*(*s*(*s*(0))), *stops_being_slippery*(0),

$pickup\_succeeds(s(0)), goto\_succeeds(0)]$

$[retains\_key(s(s(0))), initially\_unslippery(key), pickup\_succeeds(s(0)),$
$goto\_succeeds(0)]$

$[initially\_slippery(key), stays\_slippery(0), stays\_slippery(s(0)),$
$holds\_slippery\_key(s(s(0))), pickup\_succeeds(s(0)), goto\_succeeds(0)]$

Example 4.39 From the above explanations of $carrying(key, s(s(0)))$, we can derive explanations for $\sim carrying(key, s(s(0)))$, three of which are:

$[goto\_fails(0)]$

$[goto\_succeeds(0), pickup\_fails(s(0))]$

$[goto\_succeeds(0), pickup\_succeeds(s(0)), drop\_slippery\_key(s(s(0))),$
$initially\_slippery(key), stays\_slippery(0), stays\_slippery(s(0))]$

There are six other explanations.

What is important to notice about these examples is that we can write declarative clauses defining the dynamics of the world, forgetting about the fact that some of the conditions will be uncertain. The explanations of a ground formula are a description of exactly those worlds in which it is true. The mutual incompatibility means that each world is only described by one explanation.

5 Probabilities

In many applications we would like to assign a probability over the alternatives [19, 23]. This lets us use the logic programming representation for standard Bayesian reasoning. The rule structure mirrors the independence of Bayesian networks, and provides a form of contextual independence that can be exploited in probabilistic inference [24].

Suppose we are given a function P_0 from atomic choices into [0, 1] such that $\sum_{\alpha \in \chi} P_0(\alpha) = 1$ for all alternatives $\chi \in \mathbf{C}$. That is, P_0 is a probability distribution on each alternative. We assume that the alternatives are unconditionally probabilistically independent.

Intuitively, we would like the probability measure of any world to be the product of the probabilities of the atomic choices that make up the total choice defining the world. That is, $\mu(w_\tau) = \prod_{\alpha \in \mathbf{R}(\tau)} P_0(\alpha)$. The probability of any proposition is the sum of the probabilities of the worlds in which proposition is true. That is,

$P(\alpha) = \sum_{w_\tau \models \alpha} \mu(w_\tau)$. This only works when the choice space is finite. However, when the alternatives are parametrized and there are function symbols (as in Section 4.6), there are infinitely many possible worlds, each with measure zero, and we need a more sophisticated construct. The general idea is to define a measure over sets of possible worlds.

Let $W_{\langle \mathbf{C},\mathbf{F}\rangle} = \{w_\tau : \tau \text{ is a selector function on } \mathbf{C}\}$. Thus $W_{\langle \mathbf{C},\mathbf{F}\rangle}$ is the set of all possible worlds. We define the algebra of subsets of $W_{\langle \mathbf{C},\mathbf{F}\rangle}$ that can be described by finite sets of finite composite choices.

$$\begin{aligned}\Omega_{\langle \mathbf{C},\mathbf{F}\rangle} = \{\omega \subset W_{\langle \mathbf{C},\mathbf{F}\rangle} : &\ \exists \text{ finite set of finite composite choices } \mathbf{K}\\ &\text{such that } \forall w,\ \ w \in \omega \text{ iff } w \models \mathbf{K}\}\end{aligned}$$

$\Omega_{\langle \mathbf{C},\mathbf{F}\rangle}$ is closed under finite unions and complementation. That is, if $\omega_1, \omega_2 \in \Omega_{\langle \mathbf{C},\mathbf{F}\rangle}$ then $\omega_1 \cup \omega_2 \in \Omega_{\langle \mathbf{C},\mathbf{F}\rangle}$ and $W_{\langle \mathbf{C},\mathbf{F}\rangle} - \omega_1 \in \Omega_{\langle \mathbf{C},\mathbf{F}\rangle}$.

As shown in Section 4.5, every set of composite choices is equivalent to a mutually incompatible set of composite choices. Thus, for every $\omega \in \Omega_{\langle \mathbf{C},\mathbf{F}\rangle}$, there exists a mutually incompatible set of composite choices $\mathbf{K}$ such that $w \in \omega$ iff $w \models \mathbf{K}$.

Lemma 5.1 If $\mathbf{K}$ and $\mathbf{K}'$ are both mutually incompatible sets of composite choices such that $\mathbf{K} \equiv \mathbf{K}'$, then $\sum_{\kappa \in \mathbf{K}} \prod_{\alpha \in \kappa} P_0(\alpha) = \sum_{\kappa' \in \mathbf{K}'} \prod_{\alpha' \in \kappa'} P_0(\alpha')$.

This lemma is the same as Lemma A.8 in [19].

We can then define a probability measure $\mu : \Omega_{\langle \mathbf{C},\mathbf{F}\rangle} \to [0, 1]$ by:

$$\mu(\omega) = \sum_{\kappa \in \mathbf{K}} \prod_{\alpha \in \kappa} P_0(\alpha)$$

where $\mathbf{K}$ is a mutually incompatible set of composite choices such that $w \in \omega$ iff $w \models \mathbf{K}$. Lemma 5.1 implies that it doesn't matter which $\mathbf{K}$ is chosen.

Lemma 5.2 μ satisfies the axioms of probability,[7] namely:

- $\mu(\overline{\omega}) = 1 - \mu(\omega)$ where $\overline{\omega}$ is the complement of ω and,
- if ω_1 and ω_2 are disjoint sets, $\mu(\omega_1 \cup \omega_2) = \mu(\omega_1) + \mu(\omega_2)$.

[7] Note that we don't require σ-additivity (the sum rule for infinite disjuncts) as, by acyclicity, each ground formula has a finite set of finite explanations.

We can now define the probability of ground formula g by $P(g) = \mu(\{w : w \models g\})$. This is well defined as for any g, by the acyclicity of the facts, there is a finite covering set of finite explanations of g. That is, $\{w : w \models g\} \in \Omega_{\langle \mathbf{C}, \mathbf{F} \rangle}$. In particular, we have the following:

Proposition 5.3 If $\mathbf{K}$ is a covering and mutually incompatible set of explanations of g, then

$$P(g) = \sum_{\kappa \in \mathbf{K}} \prod_{\alpha \in \kappa} P_0(\alpha)$$

We can define a conditional probability in the normal manner: If $P(\beta) \neq 1$,

$$P(\alpha|\beta) =_{\text{def}} \frac{P(\alpha \wedge \beta)}{P(\beta)}.$$

From this we can see that Bayesian conditioning corresponds to abduction. When we observe β and condition on it, this means that we find the explanations for it.

Section 4.4 shows how to construct a set of covering explanations of g. The problem is to generate a covering and mutually incompatible set of explanations of g. There are three approaches than can be used:

1. build the fact base to guarantee that only mutually incompatible explanations are returned by $expl(g)$.

2. construct a covering and mutually incompatible set of explanations from a covering set of explanations.

3. compute the probabilities directly from the set of explanations generated by $expl(g)$.

These are discussed in the next three sections.

5.1 Disjointed Rule Bases

Poole [19] shows the relationship to abduction in the acyclic definite clause case (without negation as failure) under the constraint that the bodies of the ground instances of the rules for any atom are incompatible. That is, if $a \leftarrow b_1$ and $a \leftarrow b_2$ are two ground instances of rules for a, there is no possible world in which b_1 and b_2 is true. Under this restriction, the probability of g can be obtained by adding the probabilities of the explanations of g [19]. In this section we extend this idea to the more general rule formulation given in this paper.

Definition 5.4 A rule base $\mathbf{F}$ is **disjointed** if

- for every pair of different rules $h_1 \leftarrow b_1$ and $h_2 \leftarrow b_2$ in $\mathbf{F}$ and for every pair of grounding substitutions θ_1 and θ_2, if $h_1\theta_1 = h_2\theta_2$, there is no world in which $b_1\theta_1 \wedge b_2\theta_2$ is true,
- for every rule $h \leftarrow b$ in $\mathbf{F}$ and for every pair of grounding substitutions θ_1 and θ_2 such that $h\theta_1 = h\theta_2$ and $b\theta_1 \neq b\theta_2$, there is no world in which $b\theta_1 \wedge b\theta_2$ is true, and
- whenever $b_1 \vee b_2$ appears in the body of any ground rule, there is no world in which any ground instance of $b_1 \wedge b_2$ is true.

Suppose we have a disjointed knowledge base. We can use a variant of *expl* to compute a mutually incompatible and covering set of explanations of g. In particular, none of the *mins* functions are needed (no subset of an explanation will ever be generated as an explanation). The only time that mutually incompatible composite choices can be generated is in computing *duals* (Section 4.2.1). We can make these disjoint using the algorithm of Figure 3. This is summarised in algorithm $expl_d$:

$$expl_d(G) = \begin{cases} \{\} & \text{if } G = true \\ expl_d(A) \otimes expl_d(B) & \text{if } G = A \wedge B \\ expl_d(A) \cup expl_d(B) & \text{if } G = A \vee B \\ disjoint(duals(expl_d(A))) & \text{if } G = {\sim}A \\ \{\{G\}\} & \text{if } G \in \cup\mathbf{C} \\ \bigcup_i expl_d(B_i) & \text{if } G \notin \cup\mathbf{C},\ G \leftarrow B_i \in gr(\mathbf{F}) \end{cases}$$

Proposition 5.5 If $\mathbf{F}$ is a disjointed rule base, $expl_d(g)$ will always return a minimal mutually incompatible and covering set of explanations of g.

The coveringness is a direct consequence of Theorem 4.24. The mutually incompatibility is due to the disjointedness of the rule base; $\otimes$ preserves the mutual incompatibility, and union is only used on pairwise incompatible sets. The minimality is a consequence of the mutually incompatibility — a subset is compatible with its superset if the superset is consistent.

Note that the term *minimal* here means that the set of explanations is minimal; i.e., no subset also has this property. It does not mean that the explanations are minimal.

This idea of exploiting properties of rules to gain efficiency is a powerful and general idea. From experience, it seems that obeying the discipline of writing disjointed rule sets helps to debug knowledge bases and to write clearer and more concise knowledge bases.

5.2 Making Explanations mutually incompatible

Given a covering set of explanations of g, as produced, for example by the use of *expl*, we can use the repeated splitting algorithm of Section 4.5 to create a covering and mutually incompatible set of explanations of g.

5.3 Computing probabilities from arbitrary sets of composite choices

We do not need to create a mutually incompatible set of explanations. Probabilities can be computed from an arbitrary covering set of explanations. The general idea is when adding probabilities of disjunctions we have to subtract the part we have double counted.

The following formula is true whether or not κ_1 and κ_2 are incompatible.

$$P(\kappa_1 \vee \kappa_2) = P(\kappa_1) + P(\kappa_2) - P(\kappa_1 \wedge \kappa_2)$$

The general case is more complicated. If we have $\{\kappa_1, \ldots, \kappa_n\}$ as a covering set of explanations of g, we can use the following formula to compute the probability of g:

$$P(\kappa_1 \vee \cdots \vee \kappa_n) = \sum_{j=1}^{n} \sum_{\substack{i_1 \cdots i_j \\ 1 \le i_1 < \cdots < i_j \le n}} (-1)^{j+1} P(\kappa_{i_1} \wedge \cdots \wedge \kappa_{i_j})$$

The second sum is summing over all subsets of $\{\kappa_1, \ldots, \kappa_n\}$ that contain exactly j elements.

$P(\kappa_{i_1} \wedge \cdots \wedge \kappa_{i_j})$ is easy to compute. It is 0 if $\kappa_{i_1} \cup \cdots \cup \kappa_{i_j}$ is inconsistent, and otherwise it is $\prod_{\alpha \in \kappa_{i_1} \cup \cdots \cup \kappa_{i_j}} P_0(\alpha)$.

The main problem with this sum is that we are summing $2^n - 1$ probabilities, where n is the number of explanations of g. This can often be reduced as we do not need to consider any supersets of an inconsistent composite choice.

6 Combinatorics

In this section we explore how much the abductive view can save over the model theoretic view, and provide some answers to the question: how much smaller will a set of covering explanations be than the set of possible worlds? This is important as it is the set of these covering explanations that we need to sum over to determine probabilities.

In the general case with infinitely many finite alternatives, there are infinitely many possible worlds, but any ground atom always has a finite set of covering explanations, each of which is finite. This is guaranteed by the acyclicity of the rule base.

When there are finitely many finite alternatives, there are examples where there are the same number of covering explanations of some g as there are worlds in which g is true. This occurs when (and only when) the explanations are total choices.

It is interesting to consider the case where the ICL theory is the result of transforming a Bayesian network as in [19]. Although any probabilistic dependence can be modelled with independent choices (hypotheses) in the ICL, this is done at the cost of greatly increasing the number of worlds. However, as we see below, the abductive characterisation can be used to (often more than) counteract this combinatorial explosion.

Suppose there are n (binary) random variables that we want to model, and there are no independencies that we can exploit. The Bayesian network [17] representation for this is a complete graph. There are $2^n - 1$ independent numbers that can be assigned to specify the joint distribution (we can assign a non-negative real number to each of the 2^n worlds, but this is over constrained by one number — we need to divide by the sum in order to get a probability). In order to model this with independent choices, we have $2^n - 1$ binary alternatives. This is exactly the number of alternatives (or disjoint declarations in the terminology of [19]) created in the translation of the Bayesian network into a probabilistic Horn abduction theory [19]. This, however creates 2^{2^n-1} possible worlds in the independent choice logic. This combinatorial argument would seem to indicate that the modelling by independent choices can be ruled out on combinatorial grounds.

However, consider the size of the minimal explanations of any proposition. Each minimal explanation has at most n assumptions; they at most assign one value to each of the original propositions. At the extreme, for the root variable (with $n-1$ parents), there are 2^{n-1} rules, each with its own alternative. Only one of these rules and only one of the atomic choices will be in any minimal explanation, as there will only be one assignment of values to the parents of that node implied by the explanation. There will be at most 2^n explanations. For this case there are thus logarithmically fewer explanations than there are possible worlds — the abductive characterisation makes the combinatorics of the independent choice the same as for the general case with arbitrary dependencies amongst the hypotheses (exactly the same number of numbers have to be assigned as well).

The abductive characterization is interesting because:

1. We can exploit independencies amongst variables in the same way as Bayesian networks [19].

2. We only need to consider the ancestors in the Bayesian network of what we are trying to find the explanations of, and not the set of all valuations of the Bayesian network (although such pruning can also be done in the Bayesian network [2]).

3. Where there are contextual independencies (some variable may only depend on another variable when a third variable has a particular value), the abductive characterisation naturally exploits these independencies. Such contextual independence have been exploited in dynamic programming [3, 21] and probabilistic inference [23] — instead of considering the state space, as dynamic programming traditionally does, we can consider just the explanations of a proposition we were interested in (e.g., the different values of the utility).

7 Beyond Acyclic Logic Programs

We chose acyclic logic programs [1] as the base logic as there is a unique stable model for any acyclic logic program. This is important in properly defining truth in possible worlds (Definition 3.3). The property that we want for our logic programs is that each total choice leads to a single model. This means two things:

1. Each selection of an element from each alternative is consistent. This means that the logic cannot allow a selection of choices from some alternatives to impose any restrictions on choices from other alternatives. This, for example, disallows the logic from being the arbitrary predicate calculus, Horn clauses with integrity constraints [14], or logic programs under the stable model semantics with no stable models (such as $a \leftarrow \sim a$).

2. Each total choice cannot be extended into more than one possible world. This excludes us from having explicit disjunction in our logic. It also means, for example, that we cannot have logic programs under the stable models semantics with more than one stable model (such as $a \leftarrow \sim b$, $b \leftarrow \sim a$ which has two stable models, one with a true and one with b true). We are also excluding three–valued models of logic programs [26] from consideration, as we cannot give a probabilistic interpretation to them.

Acyclic logic programs were chosen because they guarantee the uniqueness of the model for each total choice. The acyclic condition is, however, too strong. One way to weaken the condition is by the use of what we call contingently acyclic logic programs.

One thing that cannot be expressed in an acyclic logic program is to give a relation a default value:

Example 7.1 Suppose $r(O, V)$ is a relation between object O and value V. To state that there is a default value d (i.e., objects should have value d unless they have been given another value), we can use the logic program:

$$r(O, d) \leftarrow \sim has\_another\_r\_value(O)$$
$$has\_another\_r\_value(O) \leftarrow r(O, V) \wedge V \neq d$$

This is not acyclic as there is no way to assign the natural numbers so that

$$\begin{aligned} r(o, d) &> has\_another\_r\_value(o) \\ has\_another\_r\_value(o) &> r(o, d) \end{aligned}$$

But this second constraint seems to be superfluous, as it only applied to a rule whose body is always false (as it has $d \neq d$) in the body.

Example 7.2 Suppose we have the knowledge base:

$$above(X, Y) \leftarrow on(X, Z) \wedge above(Z, Y).$$
$$above(X, Y) \leftarrow on(X, Y).$$
$$on(a, b).$$
$$on(b, c).$$
$$on(c, d).$$

This knowledge base is not acyclic as the first rule has the acyclic constraint that $above(a, b) > above(a, b)$, which cannot be true. This relation however relies on an instance of the rule with $on(X, X)$, but there is no instance of this atom that is true. Given the database for *on*, the only instances of the first rule that do not immediately fail, are those for which $on(X, Z)$ unify with one of the clauses. If we only consider these instances, we can assign a natural number to make these acyclic. If we were to add $on(d, a)$ to the knowledge base, there is a cycle.

We define contingently acyclic programs to exclude from $gr(F)$ those rules whose body is always false.

Definition 7.3 Given a set of clauses, we say a ground atom is **immediately false** if it does not resolve with the head of any clause or, when we have semantic constraints (due to built-in relations like equality), if the atom is false due to the semantic constraints (e.g., $d \neq d$). If G is a set of ground clauses define

$$redundant(G) = \{H \leftarrow B \in G : \text{ one conjunct in } B \text{ is immediately false}\}$$

A logic program F is **contingently acyclic** if $gr(F) - redundant(gr(F))$ is acyclic.

This means that the rule bases of examples 7.1 and 7.2 above are contingently acyclic. It is called contingently acyclic as the instances of the rules happen to be acyclic based on the existence of other clauses and semantic constraints.

Corollary 7.4 The equivalences of theorem 2.3 hold for contingently acyclic programs.

This is true as the semantics of a logic program, and the completion formula are completely determined by the the ground instances. Removing the rules with always false bodies does not change any of the semantics referred to in Theorem 2.3.

The following lemma says that we can use the acyclic logic program results for our semantics if F is acyclic (as $\mathbf{R}(\tau)$ imposes no constraints on the acyclicity).

Lemma 7.5 If F is a (contingently) acyclic logic program, and τ is a selector function, then $F \cup \mathbf{R}(\tau)$ is a (contingently) acyclic logic program.

One could imagine that the idea behind contingently acyclic logic programs could be extended to remove instances that would never be generated at all by forward chaining (contingently acyclic logic programs remove all of those instances that provably cannot be generated by forward chaining one step). In Definition 7.3, once a redundant set of clauses is removed, there may be more immediately false ground atoms, that can in turn make more redundant clauses. However, which clause is redundant depends on the total choice:

Example 7.6 Consider a logic program, that contains the rules

$$a \leftarrow c \wedge b$$
$$b \leftarrow \sim c \wedge a$$

but is otherwise acyclic where the assignment of numbers is such that c is assigned a lower number than both a and b. There is still a unique model for each total choice, as each total choice, together with the facts, entail either c or $\sim c$. Which it entails may depend on the total choice.

Extending the definition to cover such cases would not cover the class of all programs with a unique model for each total choice, as the following example shows:

Example 7.7 Here is an example, which is not contingently acyclic:

$$\begin{aligned} \mathbf{F} \;=\{\;& \mathit{someonewins} \leftarrow \mathit{int}(N) \wedge \mathit{wins}(N), \\ & \mathit{int}(0), \\ & \mathit{int}(s(N)) \leftarrow \mathit{int}(N)\} \\ \mathbf{C} \;=\{\;& \{\mathit{wins}(N), \mathit{loses}(N)\} : N \text{ is a term}\} \end{aligned}$$

This has the property of a unique model for each selection, but is not acyclic because *someonewins* has to be assigned a number bigger than any integer. This program can be interpreted according to our semantics given in Section 3; there are $2^{\aleph_0}$ possible worlds, where *someonewins* is true in all but one of them. When we consider this probabilistically, unless there are only finitely many alternatives with a non-zero probability of a win, the world where *someonewins* is false has measure zero.

One could think of extending the notion of (contingently) acyclic programs to include, for example, limit ordinals, but whether this would either cover all of the naturally occurring cases or be needed for real applications is still an open question.

The motivation for restricting to acyclic programs was to ensure there was a unique model for each total choice. Other uses for acyclic logic programs are the ability to prove termination [16]. These are related, and it is interesting to note that the terminating programs of Marchiori [16] result in unique models, but the converse is not true. One complication is that we don't want to have to, for each total choice, prove there is a unique model, as there can be infinitely many total choices.

8 Comparison with other Formalisms

8.1 Propositional Satisfiability

When the alternatives are binary, the operations on sets of composite choices correspond to operations on DNF formulae. The main point of this paper, is to show how the composite choices interact with the rules that include negation as failure.

Consider a binary alternative $\{a, b\}$. In any world where a is selected, b is false. In any world where b is selected, a is false. Thus $b \equiv \sim a$. Whenever b appears in the facts it can be replaced by $\sim a$, without affecting what is entailed by the theory. We could even thing of this alternative as $\{a, \sim a\}$.

For the binary case, a composite choice corresponds to a consistent conjunction of literals. A set of composite choices corresponds to a DNF formula made up of literals of atomic choices. The dual operation corresponds to negating the DNF and converting the result to DNF. The operation $\otimes$ (Definition 4.22) corresponds to conjoining the DNF formulae and distributing back into DNF. The algorithm of Figure 2 corresponds to the use of binary resolution to compute the prime implicates of a set of clauses [13].

Davydov and Davydova [7] have extended the above Boolean logic notions to allow more that one element in each alternative (as in this paper). Their notion of a dual corresponds to the hitting set in this paper. They have provided an algebra of operations on these dual structures. Their work is orthogonal to the work in this paper. What is important about this paper is how the sets of composite choices interact with the rules. If we consider the constraints on the DNF formulae, then we realise that any composite choices is consistent with the facts; the facts by themselves impose no constraints on the composite choices. The constraints on the composite choices is provided by the interaction between the facts and the observations. Thus this paper is presenting a particular way to provide such constraints (one that corresponds to Bayesian conditioning). Based on this interaction, we have presented an abductive characterization of the logic. We have provided a limited set of operations on these dual structures that are applicable for evidential reasoning. Davydov and Davydova used the dual structures for optimization, where they want to select the best total choice rather based on an evaluation function, rather than the evidential reasoning task of this paper, where we want to sum the measures of the consistent total choices. For decision problems [23], we want to both sum over choices by nature and optimize over choices by the decision making agent. The combination of these techniques is an intriguing possibility, but beyond the scope of this paper.

8.2 Abductive Logic Programming

The combination of abduction and logic programming has a long history (see Kakas et al. [12] for a good survey). The combination proposed in this paper is quite different from other proposals mainly because the abductive characterisation is a consequence of an independently defined semantics. The normal definition of stable models [10] is used to define negation as failure — there is no alternative notion of negation as failure that needs to be defined and motivated. There is a much closer tie between negation as failure used in this paper and so-called "real" negation; $\sim a$ is true in a world if and only if a is not true in the world.

In abductive logic programming, the minimality of explanations has a semantic significance; if E is an explanation for some g, it does not imply that $E \cup \{a\}$, even if internally consistent, is an explanation. However, in the ICL, any consistent superset of an explanation is an explanation: if E is an explanation for g, and a is an atomic choice that is consistent with E, then $E \cup \{a\}$ is an explanation for g.

One of the things unique about the work reported in this paper is that the explanations of $\sim g$ are a function of the explanations of g. In other frameworks for abductive logic programming, if there is an explanation for g, and negation was not used to prove g, there are no explanations for $\sim g$ (all explanations of $\sim g$ are obtained from negation as failure used to prove g).

The main semantic difference is that we interpret failure-to-prove in each world, rather than failure given the whole theory. This means that equivalences that are true for each world, such as Clark's completion for nonassumables, hold for the whole theory. Rather than forcing this meaning, it is a natural consequence of the framework.

Much of the power comes from having a structured hypothesis space. It is this structure that allows us to give such a clean semantics, upon which it is easy to impose a probability measure (Section 5), and upon which it is easy to extend to multiple agents making choices [23].

The use of a rule base that is a complete definition, even if all of the elements of the body of rules are not completely defined, is similar to the completion of non-abducible predicates in the completion semantics for abduction [6, 20], but these don't allow for negation as failure. It is also similar to the motivation for OLP-FOL [9], but in the ICL the aim is to handle all uncertainty in terms of Bayesian decision theory (or game theory when there is more than one agent) [23], as opposed to handling uncertainty in the non-defined predicates using first order logic (as is done in OLP-FOL). Note that OLP-FOL takes a different approach to badly defined predicates (such as p in $p \leftarrow \sim p$). Whereas, Denecker is quite happy to have a three-valued semantics for these predicates (but not for predicates defined in the FOL), we don't allow these because we want to have a normal probabilistic semantics. Rather than allowing a three-valued semantics, we restrict the language to be acyclic to ensure we don't have such badly defined predicates. It also seems that writing acyclic programs is good programming practice; a cyclic program usually indicates a bug.

8.3 Probabilistic Horn abduction

Probabilistic Horn abduction [19, 18] is a pragmatic framework for combining logic and probability with independent hypotheses and definite clauses giving the consequences of the hypotheses. There is a close relationship between Bayesian networks [17] and probabilistic Horn abduction [19].

The independent choice logic extends the logical part of probabilistic Horn abduction in allowing for negation as failure in the body of rules, and in allowing for non-disjoint rules. The modelling language is thus much expanded without losing semantic simplicity or elegance. A complementary paper [23] considers allowing different agents to choose assumptions, and explores the relationship to notions in game theory and stochastic dynamical systems. That paper uses only the model-theoretic semantics and not the abductive characterization explored here.

9 Conclusion

This paper has presented a mix of abduction and logic programming (including negation as failure) that allows for a clean mix of logic programming and probability. This was defined in terms of a semantic framework that allows for the independent choices. This framework allows us to import directly the stable models semantics for our logic programs (or any other semantics that is definitive on total choices). The abductive characterisation is a consequence of the semantics — the set of explanations of a formula is a concise description of the worlds in which the formula is true. The result of this is a clean and useful mix of abduction, logic programming and probabilistic reasoning.

This has been implemented and used for applications in decision theory [23]. The code is available from my web site.

A Proofs

Lemma 4.8 If $\mathbf{K}$ is a set of composite choices, $duals(\mathbf{K})$ is a complement of $\mathbf{K}$.

Proof: Let w_τ be a world. To prove: an element of $\mathbf{K}$ is true in w_τ iff no element of $duals(\mathbf{K})$ is true in w_τ. There are two cases to consider.

Case 1: There is an element $\kappa \in \mathbf{K}$ that is true in w_τ. To show there is no element of $duals(\mathbf{K})$ that is true in w_τ. If $\kappa' \in duals(\mathbf{K})$ then, by definition of *duals*, there is an $\alpha \in \kappa$ and $\alpha' \in \kappa'$ such that $\{\alpha, \alpha'\} \subseteq A \in \mathbf{C}$. As κ is true in w_τ, α must be true in w_τ (i.e., τ selects α from A). So α' is false in w_τ. So κ' is false in w_τ.

Case 2: There is no element of $\mathbf{K}$ that is true in w_τ. This means that for every element $\kappa \in \mathbf{K}$, there is an element $\alpha \in \kappa$ such that τ doesn't select α (and instead selects some α'). Let κ'' be the range of selector function τ. Then κ'' satisfies all of the conditions for membership in $duals(\mathbf{K})$ except perhaps minimality. Then there is some subset κ' of κ'' that is minimal and so in $duals(\mathbf{K})$. κ' is true in w_τ. So there is an element of $duals(\mathbf{K})$ that is true in w_τ.
Q.E.D.

Theorem 4.19 α entails β with respect to independent choice framework theory $\langle \mathbf{C}, \mathbf{F} \rangle$ iff $\alpha \rightarrow \beta$ logically follows from the completion of $\langle \mathbf{C}, \mathbf{F} \rangle$.

Proof: If M is a model of the completion of $\langle \mathbf{C}, \mathbf{F} \rangle$, then one element of each alternative is true in M, as for each alternative $\{\alpha_1, \ldots, \alpha_k\} \in \mathbf{C}$, the completion contains the formula $(\alpha_1 \vee \cdots \vee \alpha_k) \wedge \bigwedge_{i \neq j} \sim(\alpha_i \wedge \alpha_j)$. Each of these selections is consistent with Clark's completion (as we only completed the predicates that were not atomic choices). Thus each M is a model of a total choice, and each total choice has a model M. The total choice together with Clark's completion has a unique stable model, as the theory is acyclic [1].

α entails β with respect to independent choice framework theory $\langle \mathbf{C}, \mathbf{F} \rangle$ means β is true in all possible worlds in which α is true. This is that same as for all total choices $\alpha \rightarrow \beta$ is true in the stable models of the total choice together with $\mathbf{F}$, which is true iff for all total choices $\alpha \rightarrow \beta$ logically follows from the total choice together with Clark's completion of $\mathbf{F}$ [1], which is equivalent to $\alpha \rightarrow \beta$ logically follows from the completion of $\langle \mathbf{C}, \mathbf{F} \rangle$.
Q.E.D.

Theorem 4.24 Ground formula g is true in world w_τ iff there is some $\kappa \in expl(g)$ such that $\kappa \subseteq \mathbf{R}(\tau)$. Moreover $expl(g)$ is a finite set of finite sets.

Proof: The proof is by induction on structure of the formula and on the the level assigned by the acyclicity of F. Acyclicity is needed to make sure the inductive proofs ground out.

Base case: The base case for the induction is unstructured formulae (atoms) that are minimal in the acyclicity ordering. These are atomic choices and atomic facts. The theorem is trivially true for the atomic facts, where $\kappa = \{\}$.

Suppose α is an atomic choice. α is true in w_τ iff $\alpha \in \mathbf{R}(\tau)$. $expl(\alpha) = \{\{\alpha\}\}$ thus $\kappa = \{\alpha\}$, and $\kappa \subseteq \mathbf{R}(\tau)$ is the same as $\alpha \in \mathbf{R}(\tau)$.

Structural Induction: Suppose g is a structured formula, and that the theorem is true for every substructure for g, to show it is true for g. g is either of the form $f \wedge h, f \vee h$ or $\sim f$ where f and h are formulae (for which the theorem holds).

Suppose g is of the form $f \wedge h$. g is true in world w_τ iff both f and h are true in w_τ.

- Suppose g is true in world w_τ, thus f and h are true in w_τ, then by the induction there is a $\kappa_1 \in expl(f)$ such that $\kappa_1 \subseteq \mathbf{R}(\tau)$ and a $\kappa_2 \in expl(h)$ such that $\kappa_2 \subseteq \mathbf{R}(\tau)$. Then $consistent(\kappa_1 \cup \kappa_2)$ (as they are both true in w_τ), as so $\kappa_1 \cup \kappa_2 \in expl(g)$ (or a subset of $\kappa_1 \cup \kappa_2$ is in $expl(g)$), and $\kappa_1 \cup \kappa_2 \subseteq \mathbf{R}(\tau)$.

- Suppose g is false in world w_τ, then one of f or h is false in w_τ. Suppose (without loss of generality) that f is false in w_τ. By the induction argument, there is no $\kappa \in expl(f)$ such that $\kappa \subseteq \mathbf{R}(\tau)$, and as every element of $expl(g)$ is a superset of the elements of $expl(f)$, there is no $\kappa \in expl(g)$ such that $\kappa \subseteq \mathbf{R}(\tau)$.

The proof when g is of the form $f \vee h$ is similar.

Suppose g is of the form $\sim f$. g is true in w_τ iff f is false in w_τ iff there is no element $\kappa \in expl(f)$ such that $\kappa \subseteq \mathbf{R}(\tau)$ (by the inductive assumption) which holds if and only if there is some $\kappa' \in duals(expl(f))$ such that $\kappa' \subseteq \mathbf{R}(\tau)$ (by Theorem 4.8), but then $\kappa' \in expl(g)$.

Acyclicity ordering induction Finally suppose g is a ground atom, and the theorem holds for all atoms lower in the acyclicity ordering, and for all structured formulae build from atoms lower in the acyclicity ordering. Suppose $\{g \leftarrow b_i\}$ is the set of all ground instances of rules in $gr(\mathbf{F})$ with g as the head. g is true in w_τ iff some b_i is true in w_τ which (by the inductive assumption) holds iff there is some $\kappa \in expl(b_i)$ such that $\kappa \subseteq \mathbf{R}(\tau)$. But then $\kappa \in expl(g)$ or a subset of κ is in $expl(g)$; in either case the theorem follows.
Q.E.D.

Lemma 4.28 The set of all minimal explanations of g is the set $\mathbf{K}$ resulting from termination of the algorithm of Figure 2.

Proof: It is easy to see that only explanations are in $\mathbf{K}$. Moreover, if all of the minimal explanations are in $\mathbf{K}$ then because of the use of *mins*, there will be no non-minimal explanations in $\mathbf{K}$. The only thing remaining to show is that if κ is a minimal explanation of g, then it is in $\mathbf{K}$.

The proof of this will mirror proofs of the completeness of binary resolution, with the splitting tree playing the part of the semantic tree (see, for example, [4]).

A splitting tree is a tree with nodes labelled with composite choices. A leaf node is a node such that a subset of the label is in $expl(g)$. If a node is not a leaf node then the children of the node labelled with κ are labelled with the splits of κ on alternate χ (where $\chi \cap \kappa = \{\}$).

If the root of the tree is an explanation of g, then no matter which choice is made for the alternative to split on, there can be no branches that do not lead to leaves, as eventually we will end up with nodes labelled with total choices, and a subset of a total node is in $expl(g)$ as $expl(g)$ is a covering set of explanations of g.

Suppose κ is a minimal explanation of g. Consider a minimal (in the number of nodes on the tree) splitting tree with a root labelled with κ. Claim: this splitting tree can be converted into sequence of resolutions that will derive κ. This will be carried out bottom up. Replace each leaf node by the element of $expl(g)$ that it covers. For each non-root node, when all of its children have been replaced, then we can replace it by the resolution of its replaced children on the alternative on which it was split. The only thing we need to demonstrate is that each of the replaced children contain one element of the splitting alternative (and so can be resolved together). Suppose one child does not contain an element of the splitting alternative, then this split can be replaced by subtree at that node, and we get a smaller splitting tree, which contradicts the minimality of the splitting tree.

Q.E.D.

Acknowledgements

This work was supported by Institute for Robotics and Intelligent Systems, Project B5 and Natural Sciences and Engineering Research Council of Canada Operating Grant OGPOO44121. Thanks to the constructive comments of the anonymous referees; this paper is now much more readable than it was.

References

[1] K. R. Apt and M. Bezem. Acyclic programs. *New Generation Computing*, 9(3-4):335–363, 1991.

[2] M. Baker and T.E. Boult. Pruning Bayesian networks for efficient computation. In *Proc. Sixth Conf. on Uncertainty in Artificial Intelligence (UAI-90)*, pages 257–264, Cambridge, MA, 1990.

[3] C. Boutilier, R. Dearden, and M. Goldszmidt. Exploiting structure in policy construction. In *Proc. 14th International Joint Conf. on Artificial Intelligence (IJCAI-95)*, pages 1104–1111, Montréal, Québec, 1995.

[4] C. L. Chang and R. C. T. Lee. *Symbolic Logical and Mechanical Theorem Proving*. Academic Press, New York, 1973.

[5] K. L. Clark. Negation as failure. In H. Gallaire and J. Minker, editors, *Logic and Databases*, pages 293–322. Plenum Press, New York, 1978.

[6] L. Console, D. Theseider Dupré, and P. Torasso. On the relationship between abduction and deduction. *Journal of Logic and Computation*, 1(5):661–690, 1991.

[7] G.V. Davydov and I.M. Davydova. Dual algorithms in discrete optimization. In V. Kreinovich and G. Mints, editors, *Problems of Reducing the Exhaustive Search*, pages 79–103. American Mathematical Society, Providence, RI, 1997.

[8] J. de Kleer, A. K. Mackworth, and R. Reiter. Characterizing diagnoses and systems. *Artificial Intelligence*, 56:197–222, 1992.

[9] M. Denecker. A terminological interpretation of (abductive) logic programming. In V.W. Markek and A. Nerode and M. Truszczynski, editor, *International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 15–29. Springer, Lecture Notes in Artificial Intelligence 928, 1995.

[10] M. Gelfond and V Lifschitz. The stable model semantics for logic programming. In R. Kowalski and K. Bowen, editors, *Proceedings of the Fifth Logic Programming Symposium*, pages 1070–1080, Cambridge, MA, 1988.

[11] K. Inoue and C. Sakama. A fixpoint characterization of abductive logic programs. *Journal of Logic Programming*, 27(2):107–136, 1996.

[12] A. C. Kakas, R. A. Kowalski, and F. Toni. Abductive logic programming. *Journal of Logic and Computation*, 2(6):719–770, 1993.

[13] A. Kean and G. Tsiknis. An incremental method for generating prime implicants/implicates. *Journal of Symbolic Computation*, 9:185–206, 1990.

[14] R. Kowalski. *Logic for Problem Solving*. Artificial Intelligence Series. North-Holland, New York, 1979.

[15] R. Kowalski and M. Sergot. A logic-based calculus of events. *New Generation Computing*, 4(1):67–95, 1986.

[16] E. Marchiori. Practical methods for proving termination of general logic programs. *Journal of Artificial Intelligence Research*, 4:179–208, 1996.

[17] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA, 1988.

[18] D. Poole. Logic programming, abduction and probability: A top-down anytime algorithm for computing prior and posterior probabilities. *New Generation Computing*, 11(3–4):377–400, 1993.

[19] D. Poole. Probabilistic Horn abduction and Bayesian networks. *Artificial Intelligence*, 64(1):81–129, 1993.

[20] D. Poole. Representing diagnosis knowledge. *Annals of Mathematics and Artificial Intelligence*, 11:33–50, 1994.

[21] D. Poole. Exploiting the rule structure for decision making within the independent choice logic. In P. Besnard and S. Hanks, editors, *Proc. Eleventh Conf. on Uncertainty in Artificial Intelligence (UAI-95)*, pages 454–463, Montréal, Québec, 1995.

[22] D. Poole. A framework for decision-theoretic planning I: Combining the situation calculus, conditional plans, probability and utility. In E. Horvitz and F. Jensen, editors, *Proc. Twelfth Conf. on Uncertainty in Artificial Intelligence (UAI-96)*, pages 436–445, Portland, OR, 1996.

[23] D. Poole. The independent choice logic for modelling multiple agents under uncertainty. *Artificial Intelligence*, 94:7–56, 1997. special issue on economic principles of multi-agent systems.

[24] D. Poole. Probabilistic partial evaluation: Exploiting rule structure in probabilistic inference. In *Proc. 15th International Joint Conf. on Artificial Intelligence (IJCAI-97)*, pages 1284–1291, Nagoya, Japan, 1997.

[25] D. Poole, R. Goebel, and R. Aleliunas. Theorist: A logical reasoning system for defaults and diagnosis. In N. Cercone and G. McCalla, editors, *The Knowledge Frontier: Essays in the Representation of Knowledge*, pages 331–352. Springer-Verlag, New York, NY, 1987.

[26] T.C. Przymusinski. Three-valued nonmonotonic formalisms and semantics of logic programs. *Artificial Intelligence*, 49:309–343, 1991.

[27] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95, April 1987.

[28] M. Shanahan. *Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia*. MIT Press, Cambridge, MA, 1997.

[29] J. Von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, NJ, third edition, 1953.