Success and Failure for Hereditary Harrop Formulae

James Harland
Department of Computer Science
University of Melbourne
Parkville, 3052
Australia
email: jah@cs.mu.oz.au

December, 1992

Abstract

We introduce the foundational issues involved in incorporating the Negation as Failure (NAF) rule into the framework of first-order hereditary Harrop formulae of Miller et al. This is a larger class of formulae than Horn clauses, and so the technicalities are more intricate than in the Horn clause case. As programs may grow during execution in this framework, the role of NAF and the Closed World Assumption (CWA) need some modification, and for this reason we introduce the notion of a completely defined predicate, which may be thought of as a localisation of the CWA. We also show how this notion may be used to define a notion of NAF for a more general class of goals than literals alone. We also show how an extensional notion of universal quantification may be incorporated. This makes our framework somewhat different from that of Miller et al., but not essentially so. We also show how to construct a Kripke-like model for the extended class of programs. This is essentially a denotational semantics for logic programs, in that it provides a mapping from the program to a pair of sets of atoms which denote the success and (finite) failure sets. This is inspired by the work of Miller on the semantics of first-order hereditary Harrop formulae. Note that no restriction on the class of programs is needed in this approach, and that our construction needs no more than ω iterations. This necessitates a slight departure from the standard methods, but the important properties of the construction still hold.

1 Introduction

The standard way to infer negative information from a logic program is to use the Negation as Failure (NAF) rule; that is, $\neg A$ succeeds if A fails [2, 16]. Clearly then the notions of success and failure will be mutually dependent in a logic programming system using the NAF rule. The standard example of such a system is the class of normal programs, in which the body of each clause is a conjunction of literals. The notions of success and failure are determined operationally via the notion of an *SLDNF-derivation* [12], in that an atom succeeds if it has a successful SLDNF-derivation, and (finitely) fails if all SLDNF-derivations are finitely failed. An SLDNF-derivation may be thought of as being induced by the notion of an SLD-derivation, in that a literal $\neg A$ succeeds if every SLDNF-derivation of A is finitely failed, and so the success of $\neg A$ is determined by the failure of A. Thus the notion of an SLD- or SLDNF-derivation may be used to form two judgements: one that a given literal succeeds (as the derivation is successful) and the other that a given literal fails (as all derivations are finitely failed).

Whilst Horn clauses are Turing-complete [18], there is considerable interest in extending the basic formalism to include more expressive constructs than is permitted in the Horn clause framework. However, it is not always clear how the NAF rule may be applied in some of these extensions. For example, for systems in which programs may grow and/or shrink during execution [7, 13, 14, 15], the notion of failure may change with time due to changes in the program, and so the notion of negation may change. As pointed out in [7], this does not coincide with the usual notion of negation in mathematical logic, essentially because the NAF rule is not monotonic; that is, if $P \vdash \neg A$, then it does not necessarily follow that $P' \vdash \neg A$ where $P \subseteq P'$. Hence, the treatment of NAF in such systems requires careful analysis.

Another point of interest is that the notion of failure for an existentially quantified goal should presumably imply something about the success of a universally quantified goal. As some logic programming systems [15] allow universally quantified formulae as goals, it would seem that such systems provide a framework pertinent to the study of NAF, in that the notion of success for a universally quantified goal should be the dual, in some sense, of the notion of success for an existentially quantified goal.

In this paper we explore such issues of success and failure for a class of formulae known as *hereditary Harrop formulae* [15]. Such formulae permit both universal and existential quantifiers as well as implications in goal formulae, making the treatment of the NAF rule in this system rather intricate. Whilst our treatment stresses the role of negation, we also discuss the notion of success for universally quantified goals, and show how there are (at least) two such notions which are of interest, which may be classified as *intensional* and *extensional*. The main difference is that the latter one may use knowledge about the language of the program in order to make inferences, whereas the former may not. Whilst the former approach is the "standard" method used in mathematical logic, the latter one seems most appealing from a logic programming point of view, as it is quite common to assume that all terms come from a known set, known as the Herbrand Universe.

An interesting consequence of this discussion is that for the extensional notion of universal quantification to be feasible, it seems natural to consider not only the ground instances of the program (which is common practice in logic programming semantics) but also instances of the program which may contain variables. As we shall see, this essentially requires that the success of universally quantified goals be "compact", in that the success of the quantified goal depends on the success of only a finite number of instances of the goal. Hence, the failure of an existentially quantified goal is similarly compact, and so no more than ω iterations are needed to decide the truth of a given goal. This contrasts with the standard notion of failure in logic programming semantics. For example, consider the program

$$q \subset \exists x \neg p(x)$$

$$\forall x \; p(f(x)) \subset p(x)$$
$$p(a)$$

If, as is the standard practice, we only consider the ground instances of the program, then in order to come to the conclusion that q fails it is necessary to perform ω iterations (such as the Φ_P operator of Fitting [4]) to determine that $p(t)$ succeeds for every ground term t, and hence $\omega+1$ iterations to determine that $\exists x \neg p(x)$ (and hence q) fails. However, if we consider all instances of the program, and not just the ground instances, we find that after ω iterations there are still instances of $p(x)$ which have not been shown to fail, and $\exists x \neg p(x)$ does not fail in this case. Hence, a more subtle analysis of the notion of success and failure, which is required for the incorporation of universally quantified goals, may shed some interesting light on the notion of failure for normal programs.

Another issue of some technical significance is the use of the NAF rule in the presence of implications in goals. As the execution of such goals may involve adding clauses to the program, the notion of failure may change during execution, making the use of the NAF rule somewhat more intricate. As we shall see, this will involve the notion of a *completely defined* predicate, i.e. one for which it will be safe to apply the NAF rule, as we assume that such predicates should not be extended.

Having introduced these operational notions, it seems natural to turn to the question of a corresponding model-theoretic notion of consequence. Traditionally in logic programming this has been done by means of a continuous operator on a complete lattice, which thus has a least fixed point. This fixed point is then used as a "canonical" model of the program, in that an atom A is a (proof-theoretic) consequence of the program iff A is true in this model. Hence, we may think of this approach as a denotational semantics for logic programs, in that given a program P, this construction provides us with a method of finding a set of atoms which may be considered as the "meaning" of the program, in that the program is considered to be a shorthand for this set of atoms. Given that a construction along these lines has been given by Miller [14] for a large fragment of hereditary Harrop formulae, it would seem that such fixed point methods would be appropriate for our task.

One point to note is that in the case when NAF is involved, we generally wish to think of the denotational semantics of the program as *two* sets of atoms — one for the set of atoms which succeeds, and the other for the set of atoms which (finitely) fails. Hence our approach should presumably be more in the spirit of the three-valued approaches [4, 10]. Thus our construction will be similar in spirit to that of [14], but will be somewhat more intricate, as there is more than one set of atoms involved. We define a T_P operator in the traditional way, and it is not hard to show that the operator involved is in fact continuous, but only on a complete semi-lattice, rather than a complete lattice. This makes it less than trivial to find the least fixed point of this operator, but it is still straightforward. We then show that the model-theoretic notion of consequence corresponds with the operational one, a result which may be interpreted as showing the "compactness" of the operational notion.

The rest of this paper is organised as follows. In Section 2 we introduce the foundational issues involved in adding negation to the framework of first-order hereditary Harrop formulae, and we discuss our motivation for interpreting universal quantifiers extensionally, as well as the role of the Negation as Failure (NAF) rule in our setting. In section 3 we present the formal extensions necessary to incorporate the above features, and define a notion of operational provability. We then consider model-theoretic issues – in particular the problem of finding a model-theoretic notion of consequence which corresponds to the proof-theoretic one introduced above. In Section 4 we discuss the framework of [14], on which our work is based, and the preliminary technicalities of our approach, and in the following section we define the appropriate notion of consequence and present some basic results. In Section 6 we present our fixed point construction, and show that it indeed captures the notion of consequence we desire. Finally in Section 7 we present our conclusions and discuss some possibilities for further work.

2 Preliminaries

The definitions of D and G formulae given in [15] are as follows:

$$D := A \mid \forall x D \mid D_1 \wedge D_2 \mid G \supset A$$
$$G := A \mid \forall x G \mid \exists x G \mid G_1 \wedge G_2 \mid G_1 \vee G_2 \mid D \supset G$$

The addition of negated atoms to goals requires that we extend the definition of a G formula to include the case $\neg A$. The definition of the G formulae which reflects this is given below.

Definition 2.1 A definite formula D *and* a goal formula G *are defined via:*

$$D := A \mid \forall x D \mid D_1 \wedge D_2 \mid G \supset A$$
$$G := A \mid \neg A \mid \forall x G \mid \exists x G \mid G_1 \wedge G_2 \mid G_1 \vee G_2 \mid D \supset G$$

We denote by $\mathcal{D}$ the set of all D formulae, and the set of all G formulae by $\mathcal{G}$. In order to avoid confusion with the class of formulae of [14], we will refer to such definite and goal formulae as D_{HHF-} and G_{HHF-} formulae respectively.

There are two important points to note about the use of this class of formulae as a logic programming language.

Firstly, one feature of logic programming is that we usually consider the set of all closed terms, called the Herbrand Universe and which is here denoted as $\mathcal{U}$, as a set which is fixed prior to the writing of a program, and hence is constant throughout the computation process. For simplicity, we assume that the Herbrand Universe is not empty. We usually think of this set of terms as being generated by a finite number of symbols, i.e. a signature, and so it seems natural to associate a signature with a particular program. Often the signature of a program is taken to contain exactly the function and constant symbols which appear in the program. In our case we will assume that the signature must contain such symbols, but need not be limited to them. For example, consider the program below.

$$\forall x \; \text{non\_zero}(s(x))$$

It is clear that $\text{non\_zero}(s^n(0))$ succeeds for all $n \geq 1$, and that $\text{non\_zero}(0)$ fails. Clearly we wish the latter goal to fail, rather than produce a type error or something similar. Hence we need the external knowledge provided by the signature in order to have the correct "view" of the information in the program. This also makes it clear what to do with a goal such as non_zero($s(a)$), which would otherwise succeed.

Given that we think of the Herbrand Universe in this way, it seems natural to interpret universal quantifiers *extensionally*; i.e. that $\forall x p(x)$ should succeed precisely when $p(t)$ succeeds for every term t in the Herbrand Universe. However, this is at variance with the standard practice in first-order logic. For example, let the Herbrand Universe be $\{a, f(a), f(f(a)), \ldots\}$ and consider the program P below:

$$p(a)$$
$$\forall x\ p(f(x))$$

According to the standard rule, we only have $P \vdash \forall x p(x)$ when $P \vdash p(y)$ for some "new" variable y, and yet it is clear that for every term t in the Herbrand Universe $P \vdash p(t)$. We may think of this as requiring that universally quantified conclusions be independent of the language of the program. In our case we want the success of universally quantified goals to reflect the fact that we are dealing with a known Herbrand Universe, and so we will require something slightly different, i.e. that a universally quantified goal succeeds precisely when all of its instances succeed. However we will still retain the "compactness" of the previous version, in that success of a universally quantified goal will only depend on the success of a finite number of instances, and hence describes a feasible search operation. The precise details of how this is done are given below.

The second important point to note is that the Negation as Failure (NAF) rule relies on the information in the program being complete. For example, the *append* predicate given below is complete in the sense that there is no additional clause we can insert which would correctly extend the *append* relation; all the information we ever want to consider about appending lists together is given, and so it is correct to apply NAF. Thus we may think of the *append* predicate as given below as *completely defined*.

$$\forall x\ \mathrm{append}([], x, x)$$
$$\forall x \forall y \forall z \forall w\ \mathrm{append}(x.y, z, x.w) \subset \mathrm{append}(y, z, w)$$

On the other hand, not every predicate will have such a complete definition. For example, a predicate containing information about carcinogens we would wish to consider *incompletely defined*, as it is possible that our list of carcinogens is not complete. Thus whilst we wish to be able to prove $\neg$append$([], [1, 2], [3])$ from the failure of the goal append$([], [1, 2], [3])$, we may be undecided whether to conclude $\neg$carcinogen(chocolate) from the failure of the goal carcinogen(chocolate). In this way we may classify each definition in the program as completely or incompletely defined, i.e. suitable for NAF or not.

Now for Horn clauses, there are no technical problems with this approach; we may assume, explicitly or otherwise, that all predicates in the program are completely defined, and hence use NAF for all predicates. However, for hereditary Harrop formulae, goals may contain embedded implications, and so we may perform hypothetical reasoning. For example, given the goal $p \supset G$, we add the predicate p to the program, and then evaluate the goal G with respect to the larger program.

This process seems somewhat incompatible with the notion of a completely defined predicate. As pointed out by Gabbay [7], mixing the notions of NAF and embedded implications can cause the rule of modus ponens to fail. Hence it would seem natural to insist that not only must a predicate be completely defined before we can apply the NAF rule to it, but also that a predicate must be incompletely defined before we can add to its definition in the program. Thus we syntactically separate the predicates for which we may use NAF from those for which we may use embedded implication.

Hence when it comes to defining programs, we will require that programs consist of more than just a set of closed definite clauses, as we need to know which predicates are completely defined. In fact, we will include a pair of disjoint sets of predicates names as part of the program, where the first set of names are the predicates which are incompletely defined, and the second are the completely defined predicates. This allows us to make some significant technical simplifications. Whilst we insist that the two sets be disjoint, it is not necessary that the two sets cover all predicate names, as there may be some predicates whose status is somewhat unclear (i.e. it is possible that the definition is complete, but we do not know that it is complete). Thus the first set of names may be thought of as those predicates for which we know our information is incomplete, and hence it is reasonable to extend the definition of such predicates during execution of the program, but not to apply the NAF rule to them. On the other hand, the completely defined predicates may use the NAF rule, but their definitions may not be extended.

This leads us to the definition of a program which appears below.

Definition 2.2 *Given an atom $A = p(t_1, \dots t_n)$, we define $name(A) = p$, and for any formula F, $names(F) = \{name(A) \mid A \text{ appears in } F\}$.*

We say an atom A appears positively (negatively) in a formula F as follows:

- *A appears positively in A*
- *A appears positively (negatively) in $F_1 \vee F_2$ iff A appears positively (negatively) in either F_1 or F_2*
- *A appears positively (negatively) in $F_1 \wedge F_2$ iff A appears positively (negatively) in either F_1 or F_2*
- *A appears positively (negatively) in $\exists x F$ iff A appears positively (negatively) in F*
- *A appears positively (negatively) in $\forall x F$ iff A appears positively (negatively) in F*
- *A appears positively (negatively) in $F_1 \supset F_2$ iff A appears positively (negatively) in F_2 or A appears negatively (positively) in F_1*
- *A appears positively (negatively) in $\neg F$ iff A appears positively (negatively) in F*

The above definition is used to determine the "parity" of an atom A in regard to implications. For example, $p(a)$ occurs positively in $(\neg p(a)) \wedge q(b)$, but negatively in $p(a) \supset q(b)$.

Note that whilst we are ultimately interested in the properties of programs and goals (in which there are no free variables), in general we will need to consider sets of definite formulae and goal formulae which are not closed. This is due to the occurrence of universal quantifiers and implications in goals, which means that during computation we cannot restrict attention purely to closed goal formulae or closed definite formulae. For this reason we introduce the notion of a *derivation state.*

Definition 2.3 *A* derivation state *is a pair $\langle D, N\rangle$ where D is a set of definite formulae and N is a pair $\langle N_1, N_2\rangle$ where $N_i \subseteq$ names($\mathcal{H}$) and $N_1 \cap N_2 = \emptyset$, and which satisfies:*

For all atoms A and B such that name$(A) \in N_2$*, $G \supset A$ is a definite formula appearing in D, and B occurs in G, then*

- *If B occurs positively in G, then* name$(B) \in N_2$.
- *If B occurs negatively in G, then* name$(B) \in N_1$.

If $N = \langle N_1, N_2\rangle$, we say ass$(N) = N_1$, den$(N) = N_2$.

A program *is a derivation state $\langle D, N\rangle$ in which D is a set of closed definite formulae.*

We denote the set of all programs by $\mathcal{P}$. When N is the pair $\langle$names$(D), \emptyset\rangle$, we often write the program as just D. As mentioned above, we may think of D as either a set of closed definite formulae or as a set of clauses.

The restrictions on the occurrences of atoms in the bodies of the clauses of completely defined predicates ensure that completely defined predicates may only depend on the success of other completely defined predicates and the assumption of incompletely defined predicates. For example, given the clause

$$p \subset (r \supset q)$$

then if den(N) contains p, then it must contain q and ass(N) must contain r.

Clearly this property of programs will be maintained throughout execution provided that additions to the program only extend incompletely defined predicates.

We shall find the following definition useful.

Definition 2.4 *Let G be a G_{HHF^-} goal formula. We say G is* negatable *iff all predicates which occur positively in G are completely defined, and all predicates which occur negatively in G are incompletely defined.*

In the next section we will show how we can use these notions to define an operational notion of consequence.

3 Operational Provability

In order to use the class of programs introduced in the previous section as a logic programming language, we need to provide a notion of operational provability. We will do so by introducing two relations $\vdash_s$ and $\vdash_f$, where the former is used to indicate success and hence will be similar to the $\vdash_o$ relation of [14], and the latter is used to indicate failure. There will be some interplay between the two relations, as we wish to identify $P \vdash_s \neg A$ with $P \vdash_f A$ when name(A) is a completely defined predicate.

As mentioned above, we are interested in validity with respect to a given set of ground terms $\mathcal{U}$. In the presence of the Negation as Failure rule, this raises some compactness problems. For example, it seems natural to state that the goal $\exists x\, p(x)$ fails iff $p(t)$ fails for each $t \in \mathcal{U}$. However, this can lead to some technical complications, and is somewhat at variance with what happens in Prolog. Consider the program

$$\begin{gathered} \forall x\ p(s(x)) \subset p(x) \\ q \subset \exists x\ p(x) \end{gathered}$$

where the Herbrand Universe is $\{0, s(0), s(s(0)), \ldots\}$. According to the rule just mentioned above, q fails. However, a Prolog system will not return an answer for the goal q. The problem is that we need more than ω steps in order to show that q fails. Whilst this in itself is not an insurmountable problem, it seems more appropriate (and more elegant) to alter the definition of failure so that q neither fails nor succeeds. Hence we will need an extra condition, in that not only must we have that every ground instance $p(t)$ of $p(x)$ fails, but also that they do so *compactly*, i.e. that there is a finite set of instances which fail, and the failure of this finite set of instances implies the failure of all ground instances. For this reason we will need to consider arbitrary terms, and not just ground terms, in the definition of failure for existentially quantified goals. A formal definition is given below.

We will assume that the number of symbols in the Herbrand universe is finite, and so we may associate a signature with $\mathcal{U}$. This signature will be denoted as Σ.

We also assume the existence of a countably infinite set of variables disjoint from the set of constants and function symbols of all signatures.

Definition 3.1 *Let Σ be a signature containing at least one constant symbol.*
Terms are defined as follows:

- *A variable or a constant in Σ is a term*
- *If f is an n-ary function symbol in Σ and $t_1, \ldots t_n$ are terms, then $f(t_1, \ldots t_n)$ is a term.*
- *Nothing else is a term.*

The Herbrand universe *$\mathcal{U}$ is the set of all ground terms which may be formed from the symbols in Σ.*

We denote by $\mathcal{T}$ the set of all terms which may be formed from the symbols in Σ.

Note that $\mathcal{U}$ is the set of all ground terms, whereas $\mathcal{T}$ is the set of all terms. We think of each of the variables appearing in a term in $\mathcal{T}$ as ranging over elements of $\mathcal{U}$, so that we think of $\mathcal{T}$ as a more sophisticated representation of the same set of terms. Thus the non-ground terms in $\mathcal{T}$ do not have any deep meaning; they merely act as place holders.

As mentioned above, we wish to define the failure of existentially quantified goals (and also the success of universally quantified goals) by way of a finite set of "representative" instances, rather than by way of all ground instances (which is generally infinite). In order so to do, below we introduce the notion of a *representation*.

Definition 3.2 *A* covering set *of* $\mathcal{U}$ *is a set of terms* T *such that* $t \in \mathcal{U}$ *iff* t *is a ground instance of a term* $t' \in T$.

A minimal *covering set is a covering set of which no proper subset is a covering set.*

A representation *of* $\mathcal{U}$ *is a finite minimal covering set of* $\mathcal{U}$.

We refer to the set of all representations of $\mathcal{U}$ *as* $\mathcal{R}(\mathcal{U})$.

We may now state that $\exists x G$ fails if there is a representation R such that $G[t/x]$ fails for all $t \in R$. This essentially guarantees a continuity property, in that $\exists x G$ only fails when there is a finite set of instances of G that fails. Consider again the above example. As any representation must contain a term of the form $s^n(y)$ for some n, for $\exists x p(x)$ to fail we must have that $p(y)$ fails. However it seems that any reasonable definition of failure would not allow $p(y)$ to fail in this instance, as it "matches" a clause in the program which generates the same goal. Hence as $p(y)$ does not fail (even though every ground instance of it does), $\exists x p(x)$ does not fail.

We will define the success of universally quantified goals in a similar way, i.e. that $\forall x G$ succeeds iff there is a representation R such that $G[t/x]$ succeeds for all $t \in R$. Note that as a term in a representation may contain variables, we will have to consider the possibility that variables may occur in atoms, and hence take this into account in the definition of success and failure for atoms. As we desire the failure of $G[t/x]$ for all $t \in R$ to be at least as strong a condition as the failure of $G[t/x]$ for all $t \in \mathcal{U}$, it seems natural to expect that the former property implies the latter. Similarly it seems natural to expect that if $G[t/x]$ succeeds for all $t \in R$ then $G[t/x]$ succeeds for all $t \in \mathcal{U}$. Thus the success or failure of the instances of an atom shall be our guiding intuition in the relevant definitions of success and failure, and so it seems natural to adopt the policy that an atom succeeds if every instance of it succeeds, and an atom fails if every instance of it fails. However this is not quite sufficient for our purposes. Ultimately, we are interested in the validity of sentences, i.e. whether a given set of closed definite formulae implies a given closed goal formula. Free variables and the like are merely tools used in the derivation process. Hence we are not interested in the validity of formulae containing free variables per se, but only in using the success or failure of such formulae to determine the validity of sentences. Thus we know that any free variable in a derivation must be introduced by a quantifier, which allows us to simplify the definition of success for an atom. For example, let Σ be $\{a/0, f/1\}$ (so that the Herbrand Universe consists of $a, f(a), f^2(a), \ldots$), and consider the program

$$p(a)$$
$$\forall x\ p(f(x))$$

As $p(a)$ and $p(f(y))$ succeed, we have that $\forall x\ p(x)$ succeeds. On the other hand, it is less clear what we should expect for $p(z)$. It is clear that every (proper) instance of $p(z)$ succeeds (i.e. that $p(a)$ and $p(f(y))$ succeed for any y), which suggests that $p(z)$ should succeed. However this means that the definition of success for an atom may be somewhat complicated, as we may have to "split" $p(z)$ into a number of instances. This also means that for a goal such as $\forall x\ p(x)$ there are two "layers" of universal quantification – one being the explicit quantifier and the other being the implicit quantification given by the occurrence of free variables in terms such as $p(f(y))$. Hence we shall define the success of an atom as above and in [14], i.e. in terms of the atom itself, rather than its instances. Thus in the above example, $\forall x\ p(x)$ succeeds, but $p(z)$ does not. In this way the definition of success and failure for atoms may appear to be somewhat asymmetric, but as we are ultimately interested only in closed formulae, this will not be of great concern.

Note that the traditional definition of success for $\forall xG$ may be thought of as utilizing only the representation $\{y\}$ of $\mathcal{U}$. Thus our definition of success for $\forall xG$ seems a natural extension when considering validity with respect to a given Herbrand Universe.

The definitions of $\vdash_s$ and $\vdash_f$ follow. For technical reasons, these are not defined (only) over $\mathcal{P} \times \mathcal{G}$, but over the pairs $\langle\langle D, N\rangle, G\rangle$, where $\langle D, N\rangle$ is a derivation state and G is a goal formula. Note that neither D nor G need be closed here. We will refer to such pairs as *derivation pairs*.

Definition 3.3 *Let D be a definite formula. We define $|D|$ by cases as follows:*

$$\begin{array}{ll} |A| & = \{A\} \\ |\forall xD| & = \bigcup\limits_{t\in\mathcal{T}} |D[t/x]| \\ |D_1 \wedge D_2| & = |D_1| \cup |D_2| \\ |G \supset A| & = \{G \supset A\} \end{array}$$

Let $\mathcal{D}$ be a set of definite formulae. Then we define

$$|\mathcal{D}| = \bigcup_{D\in\mathcal{D}} |D|$$

We denote by $A \propto B$ the statement that A is an instance of B.

Definition 3.4 *Let $\langle P, G\rangle$ be a derivation pair where $P = \langle D, N\rangle$.*

Let $match(A) = \{G \supset B \in |D| \mid B \propto A\}$.

An O-derivation *is a tree built using the following rules:*

$$\frac{P \longrightarrow^+ G}{P \longrightarrow^+ A}\ \textit{where } G \supset A \in |D| \qquad \frac{\forall B \in |D|, B \not\propto A \quad \forall G \supset B \in \text{match}(A)\ P \longrightarrow^- G}{P \longrightarrow^- A}$$

$$\frac{P \longrightarrow^- A}{P \longrightarrow^+ \neg A}\ \textit{where } name(A) \in \text{den}(N) \qquad \frac{P \longrightarrow^+ A}{P \longrightarrow^- \neg A}$$

$$\frac{P \longrightarrow^+ G_1 \quad P \longrightarrow^+ G_2}{P \longrightarrow^+ G_1 \wedge G_2} \qquad \frac{P \longrightarrow^- G_i}{P \longrightarrow^- G_1 \wedge G_2}\ i = 1, 2$$

$$\frac{P \longrightarrow^+ G_i}{P \longrightarrow^+ G_1 \vee G_2}\ i = 1, 2 \qquad \frac{P \longrightarrow^- G_1 \quad P \longrightarrow^- G_2}{P \longrightarrow^- G_1 \vee G_2}$$

$$\frac{P \longrightarrow^+ G[t/x]}{P \longrightarrow^+ \exists xG}\ \textit{for some } t \in \mathcal{U} \qquad \frac{P \longrightarrow^- G[t/x]\ \forall t \in R}{P \longrightarrow^- \exists xG}\ \textit{for some } R \in \mathcal{R}(\mathcal{U})$$

$$\frac{P \longrightarrow^{+} G[t/x]\ \forall t \in R}{P \longrightarrow^{+} \forall x G}\ \textit{for some } R \in \mathcal{R}(\mathcal{U}) \qquad \frac{P \longrightarrow^{-} G[t/x]}{P \longrightarrow^{-} \forall x G}\ \textit{for some } t \in \mathcal{U}$$

$$\frac{P, D \longrightarrow^{+} G}{P \longrightarrow^{+} D \supset G} \qquad \frac{P, D \longrightarrow^{-} G}{P \longrightarrow^{-} D \supset G}$$

where the cases $P \longrightarrow^{+} \forall x G$ and $P \longrightarrow^{-} \exists x G$ have the side condition that no variable in R occurs free in P or G and the last two rules have the side condition that names(heads(D)) $\subseteq$ ass(N).

A sequent $P \longrightarrow^{+} G$ is called a positive *sequent, and a sequent $P \longrightarrow^{-} G$ is called a* negative *sequent.*

A positive sequent $P \longrightarrow^{+} G$ is initial *if G is an atom A and $A \in |D|$. A negative sequent $P \longrightarrow^{-} G$ is* initial *if G is an atom A and we have $\forall B \in |D|$, $B \not\approx A$ and $\forall G \supset B \in |D|$, $B \not\approx A$.*

An O-proof *is an O-derivation whose root is positive and whose leaves are initial.*

An O-denial *is an O-derivation whose root is negative and whose leaves are initial.*

We may think of proofs in this system as trees whose nodes are sequents, where each sub-tree is classified as either a "success" sub-tree or a "fail" sub-tree, and so this is a generalisation of the concept of an SLDNF-tree [12].

Note the case for negation for $\longrightarrow^{+}$, which ensures that the NAF rule is only applied to completely defined predicates of P, i.e. those whose names appear in den(N). We do not insist on the same restriction for the failure of $\neg A$ as it seems reasonable for $\neg A$ to fail whenever A succeeds, regardless of whether name(A) is a completely defined predicate or not. As negation is only applied to literals, all this does is allow more things to fail than would be the case otherwise. For example, given the program $P = \langle p, \langle \{p\}, \emptyset \rangle\rangle$, we have that $P \vdash_s p$, and so it seems reasonable that we have that $P \vdash_f \neg p$, even though p is not completely defined. Thus the success of an atom implies the failure of its negation, but the failure of an atom doesn't necessarily imply the success of its negation.

Note also that the implication rule has to be slightly modified so that only predicates in ass(N) may be extended. There may be less restrictive ways of dealing with this problem; this way ensures that only assumptions known to be consistent with the program are allowed to be made, and that a goal of the form $D \supset G$ for which D is an extension of the definition of a predicate not in ass(N) is computationally indeterminate. Without this restriction, computation of the goal append([], [], [1, 2, 3]) $\supset G$ from the standard append program (i.e. the two standard clauses with append being completely defined) involves a program which extends the definition of append, and so it is not obvious what the computational behaviour should be. We do not pursue this problem here; for now, we note that the present way of dealing with the problem is "safe", in that inconsistencies of this kind are avoided.

It should also be noted that this form of the implication rule is not a conservative extension of the implication rule for $\vdash_o$ [14], in that it is not the case that $\langle p \supset p, \langle \emptyset, \{p\}\rangle\rangle \vdash_s p \supset p$, due to the fact that p is completely defined, and hence the antecedent of the goal cannot be added to the program. However, it is not hard to see that for a program $P = \langle D, N\rangle$ and a goal G in which all predicates which occur negatively in G are in ass(N) that $D \vdash_o G$

implies that $P \vdash_s G$. In particular, if G is a goal in which all negatively occurring predicates are in ass(N) and positive occurrences of a universal quantifier are not allowed in goals, it should be clear that $D \vdash_o G$ iff $P \vdash_s G$. Hence ass(N) may be used to identify formulae for which $\vdash_s$ conservatively extends $\vdash_o$. We shall see how this device is useful later. Clearly a conservative extension is desirable and would simplify the definitions of $\vdash_s$ and $\vdash_f$, but raises some difficult problems for the model theory. Since it seems problematic for the model theory to cope with extensions to completely defined predicates, we place this restriction here to avoid considering cases which are semantically meaningless. We may think of this restriction (i.e. that predicates occurring in a negative position in a goal must appear in ass(N) for the goal to succeed or fail) as insisting that additions to the program must be known to be consistent with the program, just as we insist that to use NAF we must know that the predicate involved cannot be consistently extended. In this way this restriction, whilst somewhat undesirable, does seem to be in keeping with our approach.

Note that there are programs and goals for which neither $P \vdash_s G$ nor $P \vdash_f G$. For example, if $P = \langle p \supset p, \langle \emptyset, \emptyset \rangle \rangle$ and $G = p$, then it is clear that $P \not\vdash_s p$ and $P \not\vdash_f p$.

The following proposition shows how $\vdash_s$ and $\vdash_f$ may be thought of in a more inductive style.

Proposition 3.1 *Let $\langle D, N \rangle$ be a derivation state. Then*

- $\langle D, N \rangle \vdash_s A$ *iff* $A \in |D|$ *or* $\exists G \supset A \in |D|$ *such that* $\langle D, N \rangle \vdash_s G$
- $\langle D, N \rangle \vdash_s \neg A$ *iff* $\langle D, N \rangle \vdash_f A$ *and* $name(A) \in \text{den}(N)$
- $\langle D, N \rangle \vdash_s G_1 \vee G_2$ *iff* $\langle D, N \rangle \vdash_s G_1$ *or* $\langle D, N \rangle \vdash_s G_2$
- $\langle D, N \rangle \vdash_s G_1 \wedge G_2$ *iff* $\langle D, N \rangle \vdash_s G_1$ *and* $\langle D, N \rangle \vdash_s G_2$
- $\langle D, N \rangle \vdash_s \exists x G$ *iff* $\langle D, N \rangle \vdash_s G[t/x]$ *for some* $t \in \mathcal{U}$
- $\langle D, N \rangle \vdash_s \forall x G$ *iff* $\exists R \in \mathcal{R}(\mathcal{U})$ *such that* $\langle D, N \rangle \vdash_s G[t/x]$ *for all* $t \in R$ *where no variable in* R *appears free in* D *or* G
- $\langle D, N \rangle \vdash_s D' \supset G$ *iff* $\langle D \cup \{D'\}, N \rangle \vdash_s G$ *and names(heads(D'))* $\subseteq \text{ass}(N)$

- $\langle D, N \rangle \vdash_f A$ *iff* $\forall B \in |D|\ B \not\propto A$ *and* $\forall G \supset B \in |D|$ *such that* $B \propto A$, $\langle D, N \rangle \vdash_f G$
- $\langle D, N \rangle \vdash_f \neg A$ *iff* $\langle D, N \rangle \vdash_s A$
- $\langle D, N \rangle \vdash_f G_1 \vee G_2$ *iff* $\langle D, N \rangle \vdash_f G_1$ *and* $\langle D, N \rangle \vdash_f G_2$
- $\langle D, N \rangle \vdash_f G_1 \wedge G_2$ *iff* $\langle D, N \rangle \vdash_f G_1$ *or* $\langle D, N \rangle \vdash_f G_2$
- $\langle D, N \rangle \vdash_f \exists x G$ *iff* $\exists R \in \mathcal{R}(\mathcal{U})$ *such that* $\langle D, N \rangle \vdash_f G[t/x]$ *for all* $t \in R$ *where no variable in* R *appears free in* D *or* G
- $\langle D, N \rangle \vdash_f \forall x G$ *iff* $\langle D, N \rangle \vdash_f G[t/x]$ *for some* $t \in \mathcal{U}$

- $\langle D, N\rangle \vdash_f D' \supset G$ *iff* $\langle D \cup \{D'\}, N\rangle \vdash_f G$ *and names(heads(D'))* $\subseteq \text{ass}(N)$

It is easy to see that the success (failure) of a goal implies the success (failure) of each of its ground instances. This is formally stated in following proposition.

Proposition 3.2 *Let $\langle P, G\rangle$ be a derivation pair where $P = \langle D, N\rangle$. Then*

1. $P \vdash_s G \Rightarrow P[t/x] \vdash_s G[t/x]$ *for any* $t \in \mathcal{U}$
2. $P \vdash_f G \Rightarrow P[t/x] \vdash_f G[t/x]$ *for any* $t \in \mathcal{U}$

Proof: We proceed by induction on the depth of the relevant **O**-derivation.

In the base case, G is an atom A.

1. As the sequent is initial, $A \in |D|$, and hence $A[t/x] \in |D[t/x]|$ for any t, i.e. $P[t/x] \vdash_s A[t/x]$ for any $t \in \mathcal{U}$.
2. As the sequent is initial, $\forall B \in |D|\ B \not\propto A$ and $\forall G \supset B \in |D|$, $B \not\propto A$, and hence $\forall B \in |D[t/x]|\ B \not\propto A[t/x]$ and $\forall G \supset B \in |D[t/x]|$, $B \not\propto A[t/x]$ for any t, i.e. $P[t/x] \vdash_f A[t/x]$ for any $t \in \mathcal{U}$.

Hence the induction hypothesis is that the proposition holds when the relevant **O**-derivation is of no more than a given depth. There are six cases, many of which are similar, and so we only present the argument for the cases A and $\forall y G$.:

A:
1. If the base case does not hold, then we have $\exists G \supset A \in |D|$ such that $P \vdash_s G$, and hence $G[t/x] \supset A[t/x] \in |D[t/x]|$, and by the hypothesis $P[t/x] \vdash_s G[t/x]$, and so $P[t/x] \vdash_s A[t/x]$.
2. If the base case does not hold, then we have $\forall B \in |D|\ B \not\propto A$ and $\forall G \supset B \in |D|$ such that $B \propto A$, $P \vdash_f G$, and by the hypothesis, $P[t/x] \vdash_f G[t/x]$ for any $t \in \mathcal{U}$. Hence $\forall G' \supset B' \in |D[t/x]|$ such that $B' \propto A[t/x]$ we have $P[t/x] \vdash_f G'$, and as above $\forall B \in |D[t/x]|\ B \not\propto A[t/x]$, and so $P[t/x] \vdash_f A[t/x]$ for any $t \in \mathcal{U}$.

$\forall y G$:
1. $P \vdash_s \forall y G$ iff $\exists R \in \mathcal{R}(\mathcal{U})$ such that $P \vdash_s G[t'/y]$ for all $t' \in R$ and by the hypothesis, $P[t/x] \vdash_s G[t'/y][t/x]$ for any $t \in \mathcal{U}$, and as t' cannot contain x, we have $P[t/x] \vdash_s (\forall y G)[t/x]$ for any $t \in \mathcal{U}$.
2. $P \vdash_f \forall y G$ iff $P \vdash_f G[t'/y]$ for some $t' \in \mathcal{U}$ and by the hypothesis, $P[t/x] \vdash_f G[t'/y][t/x]$ for any $t \in \mathcal{U}$, and so $P[t/x] \vdash_f (\forall y G)[t/x]$ for any $t \in \mathcal{U}$.

□

A possibly surprising result is that there is a weak dual to Proposition 3.2, in that for D_{HHF} programs, if a representative set of instances of a goal fails, then so do all sets of representative instances, including the goal itself. This result is shown below.

Proposition 3.3 *Let $\langle P, G\rangle$ be a D_{HHF} derivation pair where $P = \langle D, N\rangle$. Let $R \in \mathcal{R}(\mathcal{U})$ be such that the terms in R do not contain any variables which occur free or bound in P or G, and $t' \in \mathcal{T}$ be such that no variable in t' occurs bound in P or G. Then*

$$P[t/x] \vdash_f G[t/x] \text{ for all } t \in R \Rightarrow P[t'/x] \vdash_f G[t'/x]$$

Proof: We proceed by induction on the depth of the **O**-derivation of $P[t/x] \vdash_f G[t/x]$.

Let $R = \{t_1, \ldots t_n\}$.

In the base case, G is an atom A.

As $P[t_i/x] \vdash_f A[t_i/x]$ for each i, we have $\forall B \in |D[t_i/x]|$, $B \not\propto A[t_i/x]$ and $\forall G \supset B \in |D[t_i/x]|$, $B \not\propto A[t_i/x]$ for each i. Let $B \in |D[t'/x]|$, and so $B = B'[t'/x]$ for some B', i.e. $B'[t'/x] \in |D[t'/x]|$, and so $B'[t_i/x] \in |D[t_i/x]|$. If $B'[t'/x] \propto A[t'/x]$, then $B'[t_i/x] \propto A[t_i/x]$ for any i, which is a contradiction, and so $B'[t'/x] \not\propto A[t'/x]$. Now let $G \supset B \in |D[t'/x]|$, and so $G \supset B = G'[t'/x] \supset B'[t'/x]$ for some $G' \supset B'$, i.e. $G'[t'/x] \supset B'[t'/x] \in |D[t'/x]|$, and so $G'[t_i/x] \supset B'[t_i/x] \in |D[t_i/x]|$. If $B'[t'/x] \propto A[t'/x]$, then $B'[t_i/x] \propto A[t_i/x]$ for any i, which is a contradiction, and so $B'[t'/x] \not\propto A[t'/x]$. Hence $\forall B \in |D[t'/x]|$, $B \not\propto A[t'/x]$, and $\forall G \supset B \in |D[t'/x]|$, $B \not\propto A[t'/x]$, and so $P[t'/x] \vdash_f G[t'/x]$.

Hence we assume that the proposition is true for all **O**-derivations of no more than a given depth. There are six cases, and again many of them are similar, and so we only present the argument for the cases A and $\forall y G$.

A: If the base case does not hold, then $P[t_i/x] \vdash_f A[t_i/x]$ iff $\forall B \in |D[t_i/x]|$ $B \not\propto A[t_i/x]$ and $\forall G \supset B \in |D[t_i/x]|$ such that $B \propto A[t_i/x]$, $P[t_i/x] \vdash_f G$. As above, if $B \in |D[t'/x]|$, we cannot have $B \propto A[t'/x]$, and so we have that $B \not\propto A[t'/x]$ $\forall B \in |D[t'/x]|$. Let $G \supset B \in |D[t'/x]|$, and as above $G \supset B = G'[t'/x] \supset B'[t'/x]$ for some $G' \supset B'$, i.e. $G'[t'/x] \supset B'[t'/x] \in |D[t'/x]|$. If $B'[t'/x] \propto A[t'/x]$, then $G'[t_i/x] \supset B'[t_i/x] \in |D[t_i/x]|$, and $B'[t_i/x] \propto A[t_i/x]$ for all i. Hence we have that $P[t_i/x] \vdash_f G'[t_i/x]$ for all i, and by the hypothesis we have $P[t'/x] \vdash_f G'[t'/x]$. Hence $\forall B \in |D[t'/x]|$ $B \not\propto A[t'/x]$ and $\forall G \supset B \in |D[t'/x]|$ such that $B \propto A[t'/x]$, $P[t'/x] \vdash_f G$, i.e. $P[t'/x] \vdash_f A[t'/x]$.

$\forall y G$: $P[t/x] \vdash_f (\forall y G)[t/x]$ iff $P[t/x] \vdash_f G[t/x][t''/y]$ for some $t'' \in \mathcal{U}$, and as t does not contain y, by the hypothesis $P[t'/x] \vdash_f G[t''/y][t'/x]$ for some $t'' \in \mathcal{U}$, i.e. $P[t'/x] \vdash_f (\forall y G)[t'/x]$.

$\square$

Note that a corresponding result does *not* hold for $\vdash_s$; consider the program below.

$$\begin{array}{c} p(a) \\ \forall x\, p(f(x)) \end{array}$$

Now $\{a, f(y)\}$ is a representation, and $p(a)$ and $p(f(y))$ both succeed, but $p(y)$ does not succeed. This result cannot be extended to D_{HHF^-} formulae either, as for the above program, $\neg p(y)$ does not fail, although $\neg p(a)$ and $\neg p(f(y))$ both do. Full discussion of this point is beyond the scope of this paper; further discussion may be found in [9].

The intuitive reading of $P \vdash_s G$ and $P \vdash_f G$ may be given as "G succeeds" and "G fails" respectively. The validity of this interpretation is shown by the proposition below.

Proposition 3.4 *Let $\langle P, G\rangle$ be a derivation pair where $P = \langle D, N\rangle$. Then*

1. $P \vdash_s G \Rightarrow P \not\vdash_f G$
2. $P \vdash_f G \Rightarrow P \not\vdash_s G$

Proof: We proceed by simultaneous induction on the depth of the **O**-proof and **O**-denial for $P \vdash_s G$ and $P \vdash_f G$.

In the base case, the sequent is initial, and hence G is just an atom A.

1. As $P \longrightarrow^+ A$ is initial, we have that $A \in |D|$, and so as $A \propto A$, it is not the case that $\forall B \in |D|\ B \not\propto A$, and by Proposition 3.1 it is not the case that $P \vdash_f A$.
2. As $P \longrightarrow^- A$ is initial, we have that $\forall B \in |D|, B \not\propto A$ and $\forall G \supset B \in |D|, B \not\propto A$, and so as $A \propto A$, we must have $A \notin |D|$ and $\not\exists G \supset A \in |D|$, and so by Proposition 3.1 it is not the case that $P \vdash_s A$.

Hence the induction hypothesis is that the proposition holds for all derivation states whose **O**-proof or **O**-denial is no more than a given depth.

There are seven cases, and again we only present the argument for the cases A and $\forall x G$; the others are similar.

A:
1. $P \vdash_s A \Rightarrow \exists G \supset A \in |D|$ such that $P \vdash_s G$, and by the hypothesis it is not the case that $P \vdash_f G$. Hence, it is not the case that $\forall B \in |D|\ B \not\propto A$ and $\forall G \supset B \in |D|$ such that $B \propto A$ we have $P \vdash_f G$, and by Proposition 3.1 it is not the case that $P \vdash_f A$.
2. $P \vdash_f A \Rightarrow \forall B \in |D|, B \not\propto A$ and $\forall G \supset B \in |D|$ such that $B \propto A$ we have $P \vdash_f G$. Now as $A \propto A$, we must have that $A \notin |D|$. Also, by the hypothesis it is impossible that $\exists G \supset A \in |D|$ such that $P \vdash_s G$, and so it is impossible that $P \vdash_s A$.

$\forall x G$:
1. $P \vdash_s \forall x G$ iff $\exists R \in \mathcal{R}(\mathcal{U})$ such that $P \vdash_s G[t/x]$ for all $t \in R$, and by Proposition 3.2, $P \vdash_s G[t/x]$ for all $t \in \mathcal{U}$, as no variables of R appear free in P. By the hypothesis this implies that it is not the case that $P \vdash_f G[t/x]$ for any $t \in \mathcal{U}$, and so it is impossible that $P \vdash_f G[t/x]$ for some $t \in \mathcal{U}$, i.e. it is not true that $P \vdash_f \forall x G$.

2. $P \vdash_f \forall x G$ iff $P \vdash_f G[t/x]$ for some $t \in \mathcal{U}$, and by the hypothesis, this implies that it is not the case that $P \vdash_s G[t/x]$ for some $t \in \mathcal{U}$. Now if $\exists R \in \mathcal{R}(\mathcal{U})$ such that $P \vdash_s G[t'/x]$ for all $t' \in R$, then by Proposition 3.2 $P \vdash_s G[t/x]$ for all $t \in \mathcal{U}$, as no variables of R appear free in P. Hence it is impossible that $\exists R \in \mathcal{R}(\mathcal{U})$ such that $P \vdash_s G[t/x]$ for all $t' \in R$, i.e. it is not true that $P \vdash_s \forall x G$.

$\square$

Thus the above definitions are consistent. Note that a similar result will hold for the case when universal quantification is interpreted intensionally (i.e. using the new constant rule).

This result suggests that it is consistent to define a more general form of negation. If we let G be a goal in which all predicates which occur in positive positions are in den(N) and all those in negative positions are in ass(N), then we may define an extension of the relations $\vdash_s$ and $\vdash_f$ as follows:

- $P \vdash_s \neg G$ iff $P \vdash_f G$
- $P \vdash_f \neg G$ iff $P \vdash_s G$

From the above result we see that $P \vdash_f G$ implies that $P \not\vdash_s G$, and so $P \vdash_s \neg G$ is a consistent conclusion. A similar argument holds for the other case, and so this indicates how we may implement negation for a wider class of formulae than just atoms.

4 A Kripke-like Model

Having established a proof-theoretic notion of consequence, we now turn to finding a corresponding model-theoretic notion. In [14] it was shown how a Kripke-like model may be constructed for a large fragment of hereditary Harrop formulae, being the class of programs in which universal quantifiers are not allowed in goal formulae, using techniques inspired by the possible worlds approach of Kripke [17]. This construction was then shown to precisely model the computational behaviour of these formulae, just as the previous construction of Kowalski and van Emden [3] did for Horn clauses. We now look at how to extend the construction of [14] to cater for the inclusion of negation, and the inclusion of our notion of universally quantified goals.

We wish to extend the programs and goals covered by this model to D_{HHF-} and G_{HHF-} formulae. The key technical features of the Kripke-like model of [14] are a set of worlds $\mathcal{W}$ together with a partial order on this set, and a consequence relation $\models$ which relates intepretations, worlds and goals, i.e. we have $I, w \models G$ for some interpretation I, world w and goal G. In [14], worlds are just programs, and the partial ordering on them is set inclusion. It is then shown how to construct an interpretation J such that $J, P \models G$ iff $P \vdash_o G$.

In our case, we wish to alter the definition of $\models$ to include cases for universal quantification and negation. Using the notion of truth in a Kripke model as a guide, we desire something along the lines of:

$$I, P \models\!\!\models \forall x G \text{ iff } \forall w' \geq P \text{ we have } I, w' \models\!\!\models G[t/x] \text{ for each } t \in \mathcal{U}$$

It should be noted that a feature of the Kripke-like model which is not a general feature of Kripke models is that the "domain" of terms considered does not from world to world, but is constant for all worlds. Hence, we prefer to think of the Kripke-like model as one in which there are no new objects constructed in the process of increasing our knowledge (i.e. as we move upwards in the partial order on worlds). It is this possibility that necessitates the side conditions on the definitions of truth in a Kripke model for the connectives $\forall$, $\supset$ and $\neg$ which state that the formula must not only be true in the current world but also true in every future world, as these are the ones which will be affected if new objects are constructed at a later stage. The worlds structure of [14] may be thought of as circumventing this side condition for $\supset$ due to the fact that the partial order between worlds is just set inclusion Analogously, since we will never construct new objects, it should be possible to do the same for $\forall$, so that we may replace the above definition by the following:

$$I, P \models\!\!\models \forall x G \text{ iff } I, P \models\!\!\models G[t/x] \text{ for all } t \in \mathcal{U}$$

We do not need to consider all worlds $w' \geq P$ here as we know that no new objects can be constructed, and so the Herbrand universe $\mathcal{U}$ is never increased.

A relevant observation at this point is that there are intermediate logics (i.e. strictly between intuitionistic logic and classical logic) which have model-theoretic properties very similar to that of the Kripke-like model. The best known example is called the *logic of constant domains*, whose models are characterised by Kripke models in which the terms "available" are the same at every world. Clearly the Kripke-like model is one such model, as $\mathcal{U}$ is fixed for all worlds. However, the logic of constant domains is not quite right in our case, as we are interested in one particular domain, rather than a class of domains. Another classification of the logic of constant domains is to add the following axiom to intuitionistic logic:

$$\forall x(\phi \vee \psi(x)) \supset \phi \vee (\forall x \psi)$$

where x does not occur free in ϕ. It should be noted that for a program P and a goal of the form $\forall x(G_1 \vee G_2(x))$ where x is not free in G_1 that $P \vdash_s \forall x(G_1 \vee G_2(x))$ iff $P \vdash_s G_1 \vee \forall x G_2(x)$. However, clearly this rule cannot be directly applied to definite formulae, as such formulae do not contain positive occurrences of disjunctions. Nevertheless, this suggests that the natural place to study the semantics seems to be an intermediate logic, rather than intuitionistic logic. This point is taken up in another paper [8].

The way to define the condition for truth of a negated atom in this model is more problematic. The desired definition of $\models\!\!\models$ for negated formulae, again using the analogy of Kripke models, would be

$$I, P \models\!\!\models \neg A \text{ iff } \forall w' \geq P \text{ we have } I, w' \not\models\!\!\models A$$

If the relation $\leq$ between worlds is set inclusion ($\subseteq$), then there is no w and A such that $J, w \models\!\!\models \neg A$ (for any "reasonable" interpretation J, i.e. one in which if $A \in P$, then $A \in J(P)$. The reason is that the world $w \cup \{A\}$ is always reachable from w, and so there is always a world w' above w such that $J, w' \not\models A$. Thus in order to incorporate negated atoms into this model, we need to restrict the reachability relation between worlds, so that there are less worlds "above" a given world w.

It is not hard to see that such " pruning" will be based around completely defined predicates, in that no world above another world can extend the definitions of the completely defined predicates. Hence, we wish the reachability relation between worlds to reflect the following:

$w \cup \{D\}$ is reachable from w iff D does not contain any more information than w about the completely defined predicates of w.

The notion of completely and incompletely defined predicates is used to determine which worlds are reachable and which are not. If P contains an incompletely defined predicate p, then we wish any world which extends the definition of p to be reachable from P. On the other hand, any world which extends the definition of a completely defined predicate of P should not be reachable from P.

As discussed above, the partial order on these worlds will need to be something more restrictive than set inclusion. The natural partial order between worlds is given below.

Definition 4.1 *Let P_1 and P_2 be sets of definite formulae, and $N_i \subseteq$ names(P_i) $\times$ names(P_i), i = 1,2. Then $\langle P_1, N_1\rangle \leq \langle P_2, N_2\rangle$ iff*

1. $P_1 \subseteq P_2$
2. $\text{ass}(N_1) \subseteq \text{ass}(N_2)$
3. $\text{den}(N_1) \subseteq \text{den}(N_2)$
4. *for each $C \in P_2 \backslash P_1$, name(head(C))* $\in \text{ass}(N_1)$

The fourth condition in the definition of $\leq$ is the interesting one, as it ensures that no completely defined predicate of P_1 is extended by P_2. Recall that $\text{ass}(N_1) \cap \text{den}(N_1) = \emptyset$, and so the only permitted extensions are those which extend the definitions of predicates which are known to be incompletely defined.

For example, let P_1 be the two clauses for append given above, and let $P_2 = P_1 \cup \{\text{append}(\text{nil}, \text{nil}, [1,2])\}$. The partial order for $\langle P_i, \langle N, \emptyset\rangle\rangle$ and $\langle P_i, \langle N, \{\text{append}\}\rangle\rangle, i = 1, 2$ is given in Figure 1.

Note that

$$\langle P_1, \langle N, \emptyset\rangle\rangle \leq \langle P_2, \langle N, \{\text{append}\}\rangle\rangle$$

but that

$$\langle P_1, \langle N, \{\text{append}\}\rangle\rangle \not\leq \langle P_2, \langle N, \{\text{append}\}\rangle\rangle$$

Thus our partial order restricts the reachable worlds to those which do not extend the completely defined predicates, and in which incompletely defined predicates remain incompletely defined.

Given this partial order, we know that all worlds above a given world w are those which consistently extend w, and so we know that if $\neg A$ is true at world w, then $\neg A$ will be true for all worlds w' above w.

$$
\begin{tikzcd}[arrows=dash]
 & \langle P_2, \langle N, \{append\} \rangle \rangle & \\
\langle P_2, \langle N, \emptyset \rangle \rangle \arrow[ur] \arrow[dr] & & \langle P_1, \langle N, \{append\} \rangle \rangle \arrow[dl] \\
 & \langle P_1, \langle N, \emptyset \rangle \rangle &
\end{tikzcd}
$$

Figure 1: The append program and its possible extensions

5 Interpretations and Consequence

An important technical difference between our construction and that of [14] is that in our case we require three-valued interpretations [4] in order to specify what is true and what is false. Due to the possible occurrence of free variables in programs and goals, we also need to consider maps which range over sets of atoms which are not necessarily ground. In this way our construction will resemble that of [5]. As mentioned in [5], it seems unreasonable for $p(x)$ to be true and $p(a)$ to be false. We may circumvent this difficulty by thinking of the ground instances of the atoms in the interpretation as the "real" items of interest, and the non-ground atoms as place-holders. This leads us to the following definition of an interpretation. Let $\mathcal{H}'$ be the set of all atomic formulae. Let $\mathcal{W}$ be the set of all derivation states.

Definition 5.1 *Let X be a set of atoms. We refer to the set of all instances of all elements of X as* $\text{inst}(X)$*. Note that $X \subseteq \text{inst}(X)$.*

We define $X_1 \prec X_2$ as $\text{inst}(X_1) \subseteq \text{inst}(X_2)$. When X_1 is a singleton set $\{A\}$ we will often write $X_1 \prec X_2$ as $A \prec X_2$.

Let P be a derivation state. An interpretation *is any function $I : \mathcal{W} \to \mathcal{H}' \times \mathcal{H}'$ satisfying the following conditions, where $I(P) = \langle S, F\rangle$:*

1. $\text{inst}(S) \cap \text{inst}(F) = \emptyset$
2. *for all worlds w_1, w_2 such that $w_1 \leq w_2$ where $I(w_1) = \langle S_1, F_1\rangle$ and $I(w_2) = \langle S_2, F_2\rangle$, we have $S_1 \prec S_2$ and $F_1 \prec F_2$*

Let $I(w) = \langle S, F\rangle$. We define $pos(I)(w) = S$ and $neg(I)(w) = F$.
We define $I_\perp(w) = \langle \emptyset, \emptyset\rangle \forall w \in \mathcal{W}$.

We think of $\text{pos}(I)$ as specifying which atoms are true, and of $\text{neg}(I)$ as specifying which atoms are false. Thus, we may think of the definition of an interpretation given in [14] as the special case of our definition obtained when $\text{neg}(I)(w) = \emptyset$ for any world w. The first condition ensures that no atom is specified as being both true and false, and so this condition ensures that interpretations are internally consistent. The second condition is a generalisation of the previous condition of internal monotonicity. This is justified by the perception that as programs increase, the knowledge contained in the program cannot decrease, and so no extension to a program is allowed to decrease either the set of atoms known to be true or the set of atoms known to be false. Thus we preserve the principle of monotonicity of information.

We use the relation $\prec$ merely as a shorthand; this is a device which allows us to handle the non-ground atoms more easily.

It is not hard to define the usual partial order $\sqsubseteq$ on interpretations, and the operator $\sqcap$. The dual operator $\sqcup$ provides a slight difficulty as there is now no longer one maximal interpretation. This means that the obvious definition of $\sqcup$ may not lead to an interpretation, as if $I_i(w) = \langle S_i, F_i\rangle$ where $i = 1, 2$, then if $\text{inst}(S_2) \cap \text{inst}(F_1) \neq \emptyset$ or $\text{inst}(S_1) \cap \text{inst}(F_2) \neq \emptyset$, the mapping $(I_1 \sqcup I_2)(w) = \langle S_1 \cup S_2, F_1 \cup F_2\rangle$ is not an interpretation as $\text{inst}(\text{pos}(I_1 \sqcup I_2)(w)) \cap$

$\text{inst}(\text{neg}(I_1 \sqcup I_2)(w)) \neq \emptyset$. However, we may consider two interpretations I_1 and I_2 with this property as mutually inconsistent, and so we never wish to consider the mapping $I_1 \sqcup I_2$ as an interpretation. This consideration motivates the definitions below.

Definition 5.2 *Let I_1, I_2 be interpretations.*
I_1 and I_2 are mutually consistent interpretations *if for all worlds w we have*

$$\begin{aligned}\text{inst}(\text{pos}(I_1)(w)) \cap \text{inst}(\text{neg}(I_2)(w)) = \emptyset\\ \text{inst}(\text{neg}(I_1)(w)) \cap \text{inst}(\text{pos}(I_2)(w)) = \emptyset\end{aligned}$$

Otherwise, I_1 and I_2 are mutually inconsistent*.*
We define the relations $\leq$ and $\sqsubset$ and the operator $\sqcap$ as follows:

$$\begin{aligned}&I_1(w) \leq I_2(w) \textit{ iff } \text{pos}(I_1)(w) \prec \text{pos}(I_2)(w) \textit{ and } \text{neg}(I_1)(w) \prec \text{neg}(I_2)(w)\\ &I_1 \sqsubseteq I_2 \textit{ iff } \forall w \in \mathcal{W} \textit{ we have } I_1(w) \leq I_2(w)\\ &(I_1 \sqcap I_2)(w) = \langle \text{inst}(\text{pos}(I_1)(w)) \cap \text{inst}(\text{pos}(I_2)(w)),\\ &\qquad\qquad\qquad \text{inst}(\text{neg}(I_1)(w)) \cap \text{inst}(\text{neg}(I_2)(w))\rangle\end{aligned}$$

If I_1 and I_2 are mutually consistent then we define the operator $\sqcup$ as follows:

$$(I_1 \sqcup I_2)(w) = \langle \text{pos}(I_1)(w) \cup \text{pos}(I_2)(w), \text{neg}(I_1)(w) \cup \text{neg}(I_2)(w)\rangle$$

In this case the interpretations do not form a lattice under the operations $\sqcup$ and $\sqcap$, as there are an infinite number of maximal interpretations. For any two such maximal interpretations I_1 and I_2 with $I_1(P) = \langle S_1, \mathcal{H}'\backslash S_1\rangle$ and $I_2(P) = \langle S_2, \mathcal{H}'\backslash S_2\rangle$ such that $S_1 \neq S_2$, then $I_1 \sqcup I_2$ is not an interpretation. However, it will be seen below that the formal results do not depend upon the interpretations forming a lattice, and so this will not be a problem.

Now we come to the generalisation of the relation $\models\!\!\models$ defined in [14]. As there is both positive and negative information explicitly given in an interpretation, it seems natural to define two relations $\models\!\!\models^+$ and $\models\!\!\models^-$ such that $\models\!\!\models^+$ is used for the positive information and $\models\!\!\models^-$ for the negative information. These are defined below.

Definition 5.3 *Let $\langle P, G\rangle$ be a D_{HHF^-} derivation pair where $P = \langle D, N\rangle$ and I be an interpretation. Then*

- $I, P \models\!\!\models^+ A$ *iff* $A \prec \text{pos}(I)(P)$
- $I, P \models\!\!\models^+ \neg A$ *iff* $A \prec \text{neg}(I)(P)$
- $I, P \models\!\!\models^+ G_1 \vee G_2$ *iff* $I, P \models\!\!\models^+ G_1$ *or* $I, P \models\!\!\models^+ G_2$
- $I, P \models\!\!\models^+ G_1 \wedge G_2$ *iff* $I, P \models\!\!\models^+ G_1$ *and* $I, P \models\!\!\models^+ G_2$
- $I, P \models\!\!\models^+ \exists x G$ *iff* $I, P \models\!\!\models^+ G[t/x]$ *for some* $t \in \mathcal{U}$
- $I, P \models\!\!\models^+ \forall x G$ *iff* $\exists R \in \mathcal{R}(\mathcal{U})$ *such that* $I, P \models\!\!\models^+ G[t/x]$ *for all* $t \in R$ *where the variables in R do not appear free in P or G*
- $I, \langle D, N\rangle \models\!\!\models^+ D' \supset G$ *iff* $I, \langle D \cup \{D'\}, N\rangle \models\!\!\models^+ G$ *and* $\langle D \cup \{D'\}, N\rangle \geq \langle D, N\rangle$

- $I, P \models^- A$ *iff* $A \prec \text{neg}(I)(P)$
- $I, P \models^- \neg A$ *iff* $A \prec \text{pos}(I)(P)$
- $I, P \models^- G_1 \vee G_2$ *iff* $I, P \models^- G_1$ *and* $I, P \models^- G_2$
- $I, P \models^- G_1 \wedge G_2$ *iff* $I, P \models^- G_1$ *or* $I, P \models^- G_2$
- $I, P \models^- \exists x G$ *iff* $\exists R \in \mathcal{R}(\mathcal{U})$ *such that* $I, P \models^- G[t/x]$ *for all* $t \in R$ *where the variables in* R *do not appear free in* P *or* G
- $I, P \models^- \forall x G$ *iff* $I, P \models^- G[t/x]$ *for some* $t \in \mathcal{U}$
- $I, \langle D, N\rangle \models^- D' \supset G$ *iff* $I, \langle D \cup \{D'\}, N\rangle \models^- G$ *and* $\langle D \cup \{D'\}, N\rangle \geq \langle D, N\rangle$

It should be clear that these definitions are similar to those of [14], with the main differences being the cases for universal quantification, implication and negation. The motivation for the universal quantification case is clear from the earlier discussion on the corresponding operational definition. In the case of [14] the side condition on implication is vacuously true, as $D \cup \{D'\} \supseteq D$. Here we explicitly require that $\langle D \cup \{D'\}, N\rangle$ be reachable from $\langle D, N\rangle$, i.e. that the new world is reachable from the first. This may be thought of as ensuring that the assumption makes sense. This restriction is not strictly necessary, in that there may be weaker restrictions that work. However, this is a safe choice, and in our opinion a natural one.

One interesting thing to note is that given two interpretations I_1 and I_2, we have that $I_1 \sqsubseteq I_2$ iff for all P and A we have $I_1, P \models^+ A \Rightarrow I_2, P \models^+ A$ and $I_1, P \models^+ \neg A \Rightarrow I_2, P \models^+ \neg A$. This property will be useful in some subsequent proofs, as it allows us to deduce that if $I_1, P \models^+ G \Rightarrow I_2, P \models^+ G$ for any P and G, then $I_1 \sqsubseteq I_2$.

The following lemma establishes that interpretations respect the reachability relation between worlds.

Lemma 5.1 *Let* P_1 *and* P_2 *be* D_{HHF-} *derivation states,* G *be a* G_{HHF-} *goal formula, and let* I *be an interpretation. If* $P_1 \leq P_2$ *then*

1. $I, P_1 \models^+ G \Rightarrow I, P_2 \models^+ G$
2. $I, P_1 \models^- G \Rightarrow I, P_2 \models^- G$

Proof: We proceed by induction on the size of G. The base case occurs when G is a literal.

A:
1. $I, P_1 \models^+ A$ implies that $A \prec \text{pos}(I)(P_1)$, and as $\text{inst}(\text{pos}(I(P_1))) \subseteq \text{inst}(\text{pos}(I(P_2)))$, we have $I, P_2 \models^+ A$.
2. $I, P_1 \models^- A$ implies that $A \prec \text{neg}(I)(P_1)$, and as $\text{inst}(\text{neg}(I(P_1))) \subseteq \text{inst}(\text{neg}(I(P_2)))$, we have $I, P_2 \models^- A$.

$\neg A$: As $I, P \models^+ \neg A$ iff $I, P \models^- A$ and $I, P \models^+ A$ iff $I, P \models^- \neg A$, this case follows directly from the one above.

Hence we assume that the lemma is true for all goals of no more than a given size. There are five cases; four of them are similar, and so we only present the argument for the cases $G_1 \vee G_2$ and $D \supset G$:

$G_1 \vee G_2$:
1. $I, P_1 \models^+ G_1 \vee G_2$ iff $I, P_1 \models^+ G_1$ or $I, P_1 \models^+ G_2$ and by the hypothesis this implies that $I, P_2 \models^+ G_1$ or $I, P_2 \models^+ G_2$, i.e. $I, P_2 \models^+ G_1 \vee G_2$.
2. $I, P_1 \models^- G_1 \vee G_2$ iff $I, P_1 \models^- G_1$ and $I, P_1 \models^- G_2$ and by the hypothesis this implies that $I, P_2 \models^- G_1$ and $I, P_2 \models^- G_2$, i.e. $I, P_2 \models^- G_1 \vee G_2$.

$D' \supset G$:
1. $I, \langle D_1, N_1 \rangle \models^+ D' \supset G$ iff $I, \langle D_1 \cup \{D'\}, N_1 \rangle \models^+ G$ and $\langle D_1 \cup \{D'\}, N_1 \rangle \geq \langle D_1, N_1 \rangle$, and so names(heads($D'$)) $\subseteq$ ass(N_1). Now as $\langle D_1, N_1 \rangle \leq \langle D_2, N_2 \rangle$, this implies that names(heads(D')) $\subseteq$ ass(N_2), and hence names(heads(D')) $\cap$ den(N_2) $= \emptyset$, i.e. $\langle D_1 \cup \{D'\}, N_1 \rangle \leq \langle D_2 \cup \{D'\}, N_2 \rangle$. Hence by the hypothesis we have $I, \langle D_2 \cup \{D'\}, N_2 \rangle \models^+ G$, and names(heads($D'$)) $\subseteq$ ass(N_2), and so $I, \langle D_2, N_2 \rangle \models^+ D' \supset G$.
2. $I, \langle D_1, N_1 \rangle \models^- D' \supset G$ iff $I, \langle D_1 \cup \{D'\}, N_1 \rangle \models^- G$ and $\langle D_1 \cup \{D'\}, N_1 \rangle \geq \langle D_1, N_1 \rangle$, and so names(heads($D'$)) $\subseteq$ ass(N_1). Now as $\langle D_1, N_1 \rangle \leq \langle D_2, N_2 \rangle$, this implies that names(heads(D')) $\subseteq$ ass(N_2), and hence names(heads(D')) $\cap$ den(N_2) $= \emptyset$, i.e. $\langle D_1 \cup \{D'\}, N_1 \rangle \leq \langle D_2 \cup \{D'\}, N_2 \rangle$. Hence by the hypothesis we have $I, \langle D_2 \cup \{D'\}, N_2 \rangle \models^- G$, and names(heads($D'$)) $\subseteq$ ass(N_2), and so $I, \langle D_2, N_2 \rangle \models^- D' \supset G$.

□

Note that this result depends critically on the fact that if $P_1 \leq P_2$, then ass(N_1) $\subseteq$ ass(N_2). It is difficult to see how such a result could hold in the absence of this property.

It is easy to prove a lemma analogous to lemma 2 of [14].

Lemma 5.2 *Let $\langle P, G \rangle$ be a D_{HHF-} derivation pair where $P = \langle D, N \rangle$ and I_1 and I_2 be two interpretations. Then*

1. *$I_1 \sqsubseteq I_2$ iff $\forall P \, \forall G \; I_1, P \models^+ G \Rightarrow I_2, P \models^+ G$.*
2. *$I_1 \sqsubseteq I_2$ iff $\forall P \, \forall G \; I_1, P \models^- G \Rightarrow I_2, P \models^- G$.*

In the light of Proposition 3.2, it should not be surprising that the following Lemma holds.

Lemma 5.3 *Let $\langle P, G \rangle$ be a derivation state, and let I be an interpretation. Then*

1. *$I, P \models^+ G \Rightarrow I, P \models^+ G[t/x]$ for any $t \in \mathcal{U}$*
2. *$I, P \models^- G \Rightarrow I, P \models^- G[t/x]$ for any $t \in \mathcal{U}$*

Next we show that interpretations conserve the consistency of $\models^+$ and $\models^-$.

Lemma 5.4 *Let I be an interpretation. Then there is no derivation pair $\langle P, G \rangle$ such that*

$$I, P \models^{+} G \text{ and } I, P \models^{-} G$$

The next lemma is important for the construction process, and may be easily established.

Lemma 5.5 *Let I_1 and I_2 be interpretations.*

- *If $I_1 \sqsubseteq I_2$, then I_1 and I_2 are mutually consistent.*
- *If I_1 and I_2 are mutually consistent, then $I_1 \sqcup I_2$ is an interpretation.*
- *If $I_1 \sqsubseteq I_2$, then $I_1 \sqcup I_2$ is an interpretation.*
- *If I_1 and I_2 are mutually consistent, then $I_1 \sqsubseteq I_1 \sqcup I_2$ and $I_2 \sqsubseteq I_1 \sqcup I_2$.*

Another important lemma, again similar to one in [14] is given below.

Lemma 5.6 *Let $I_1 \sqsubseteq I_2 \sqsubseteq I_3 \sqsubseteq \dots$ be an increasing sequence of interpretations, and let $\langle P, G\rangle$ be a D_{HHF-} derivation pair where $P = \langle D, N\rangle$.*

1. *If $\bigsqcup_{i=1}^{\infty} I_i, P \models^{+} G$, then $\exists k \geq 1$ such that $I_k, P \models^{+} G$.*
2. *If $\bigsqcup_{i=1}^{\infty} I_i, P \models^{-} G$, then $\exists k \geq 1$ such that $I_k, P \models^{-} G$.*

Proof: Note that $\sqcup_{i=1}^{\infty} I_i$ is an interpretation by Lemma 5.5. Let $I_i(P) = \langle S_i, F_i\rangle$. We proceed by induction on the structure of G.

1. If G is an atom A and $\bigsqcup_{i=1}^{\infty} I_i, P \models^{+} A$, then $A \prec \bigcup_{i=1}^{\infty} S_i$, and as A is an instance of itself, $A \in \text{inst}(\bigcup_{i=1}^{\infty} S_i) = \bigcup_{i=1}^{\infty} \text{inst}(S_i)$. Hence $\exists k$ such that $A \in \text{inst}(S_k)$, and so $\text{inst}(A) \subseteq \text{inst}(S_k)$, which implies that $A \prec S_k$, i.e. $I_k, P \models^{+} A$.
   If G is $\neg A$ and $\bigsqcup_{i=1}^{\infty} I_i, P \models^{+} \neg A$, then $A \prec \bigcup_{i=1}^{\infty} F_i$, and as A is an instance of itself, $A \in \text{inst}(\bigcup_{i=1}^{\infty} F_i) = \bigcup_{i=1}^{\infty} \text{inst}(F_i)$. Hence $\exists k$ such that $A \in \text{inst}(F_k)$, and so $\text{inst}(A) \subseteq \text{inst}(F_k)$, which implies that $A \prec F_k$, i.e. $I_k, P \models^{+} \neg A$.
2. As $I, P \models^{-} A$ iff $I, P \models^{+} \neg A$ and $I, P \models^{-} \neg A$ iff $I, P \models^{+} A$, this follows from the above argument.

Hence the inductive hypothesis is that the lemma holds for all goals of no more than a given size. There are five cases, most of which are similar, and so we only present the argument for the cases $\exists x G$ and $D \supset G$.

$\exists x G'$:

1. As $\bigsqcup_{i=1}^{\infty} I_i, P \models^{+} \exists x G'$, we have that $\bigsqcup_{i=1}^{\infty} I_i, P \models^{+} G'[x/t]$ for some $t \in \mathcal{U}$. By the hypothesis, $I_k, P \models^{+} G'[x/t]$ for some $k \geq 1$, and so we have $I_k, P \models^{+} \exists x G'$.

2. As $\bigsqcup_{i=1}^{\infty} I_i, P \models^- \exists x G'$, we have that $\exists R \in \mathcal{R}(\mathcal{U})$ such that $\bigsqcup_{i=1}^{\infty} I_i, P \models^- G'[x/t]$ for all $t \in R$, and so by the hypothesis, for each $t \in R$ there is a k_t such that $I_{k_t}, P \models^- G'[t/x]$. Let k be the maximum of all the k_t. Hence $I_k, P \models^- G'[t/x]$ for all $t \in R$, and so we have $I_k, P \models^- \exists x G'$.

$D' \supset G'$:

1. As $\bigsqcup_{i=1}^{\infty} I_i, \langle D, N\rangle \models^+ D' \supset G'$, we have that $\bigsqcup_{i=1}^{\infty} I_i, \langle D \cup \{D'\}, N\rangle \models^+ G'$. By the hypothesis, $I_k, \langle D \cup \{D'\}, N\rangle \models^+ G'$ for some $k \geq 1$, and so we have $I_k, \langle D, N\rangle \models^+ D' \supset G'$.
2. As $\bigsqcup_{i=1}^{\infty} I_i, \langle D, N\rangle \models^- D' \supset G'$, we have that $\bigsqcup_{i=1}^{\infty} I_i, \langle D \cup \{D'\}, N\rangle \models^- G'$. By the hypothesis, $I_k, \langle D \cup \{D'\}, N\rangle \models^- G'$ for some $k \geq 1$, and so we have $I_k, \langle D, N\rangle \models^- D' \supset G'$.

□

Note that this result depends critically on the compactness properties of goals containing quantifiers.

Thus our extended notion of interpretation preserves important semantic properties. We show in the next section how the important properties of the $T^\omega(I_\perp)$ construction are preserved as well.

6 A Fixed Point Construction

We wish to find a single interpretation J such that $P \vdash_s G$ iff $J, P \models^+ G$ and $P \vdash_f G$ iff $J, P \models^- G$. We proceed in a similar manner to that in [14], i.e. we build ordinal powers of a T operator, and use the union of all such powers to produce the desired interpretation.

Definition 6.1 *Let I be an interpretation, and let $w = \langle D, N\rangle$ be a D_{HHF-} derivation state. Then we define*

$$\mathrm{pos}(T(I))(w) = \{A \mid A \in |D| \textit{ or } \exists G \supset A \in |D| \textit{ such that } I, w \models^+ G\}$$
$$\mathrm{neg}(T(I))(w) = \{A \mid \mathrm{name}(A) \in \mathrm{den}(N) \textit{ and } \forall B \in |D|\ B \not\propto A \textit{ and } \forall G \supset B \in |D| \textit{ such that } B \propto A \textit{ we have } I, w \models^- G\ \}$$

$$T^\omega(I) = \bigsqcup_{i=1}^{\infty} T^i(I)$$

For an example of how this process works, consider the program below.

$$\begin{array}{l} \mathrm{even}(0) \\ \forall x\ \mathrm{even}(s(x)) \subset \neg\mathrm{even}(x) \end{array}$$

Let $P = \langle D, \langle \emptyset, \{\mathrm{even}\}\rangle\rangle$ where D is the code in the even program above. Then we have

$T^1(I_\perp)(P) = \langle \{\text{even}(0)\}, \emptyset \rangle$
$T^2(I_\perp)(P) = \langle \{\text{even}(0)\}, \{\text{even}(s(0))\} \rangle$
$T^3(I_\perp)(P) = \langle \{\text{even})(0), \text{even}(s^2(0))\}, \{\text{even}(s(0))\} \rangle$
$T^4(I_\perp)(P) = \langle \{\text{even})(0), \text{even}(s^2(0))\}, \{\text{even}(s(0)), \text{even}(s^3(0))\} \rangle$
$\ldots$

Note that even(x) neither succeeds nor fails, as there are some instances of it which succeed and some which fail.

In this way we may think of the powers of T as using the program to define an increasing sequence of interpretations which is used to model the behaviour of the program. The final interpretation in this sequence (i.e. $T^\omega(I_\perp)$) can indeed be shown to capture this operational behaviour.

It is not hard to show that T is indeed a mapping from interpretations to interpretations, i.e., that if I is an interpretation, then so is $T(I)$.

Thus the construction gives us an increasing sequence of interpretations. It is also not hard to show that T is monotonic and continuous.

Lemma 6.1 *Let I_1 and I_2 be interpretations. Then*

$$I_1 \sqsubseteq I_2 \Rightarrow T(I_1) \sqsubseteq T(I_2)$$

Proof: Let P be a D_{HHF^-} derivation state.

If $A \in \text{pos}(T(I_1))(P)$, then $A \in |D|$ or $\exists G \supset A \in |D|$ such that $I_1, P \models^+ G$, and by lemma 5.2 we have that $I_2, P \models^+ G$, and so $A \in \text{pos}(T(I_2))(P)$.

If $A \in \text{neg}(T(I_1))(P)$, then $\text{name}(A) \in \text{den}(N)$ and $\forall B \in |D|\ B \not\propto A$ and $\forall G \supset B \in |D|$ such that $B \propto A$, $I_1, P \models^- G$, and by lemma 5.2 we have that $I_2, P \models^- G$, and so $A \in \text{neg}(T(I_2))(P)$.

Hence $T(I_1) \sqsubseteq T(I_2)$.

□

Lemma 6.2 *Let $I_1 \sqsubseteq I_2 \sqsubseteq \ldots$ be an increasing sequence of interpretations. Then*

$$\bigsqcup_{i=1}^{\infty} T(I_i) = T(\bigsqcup_{i=1}^{\infty} I_i)$$

Proof: $\sqsubseteq$: $I_j \sqsubseteq \bigsqcup_{i=1}^{\infty} I_i$, and so by lemma 6.1, we have that $T(I_j) \sqsubseteq T(\bigsqcup_{i=1}^{\infty} I_i)$ for any j, and hence $\bigsqcup_{i=1}^{\infty} T(I_i) \sqsubseteq T(\bigsqcup_{i=1}^{\infty} I_i)$.

$\sqsupseteq$: If $A \in \text{pos}(T(\bigsqcup_{i=1}^{\infty} I_i))(P)$, then either $A \in |D|$, in which case $A \in T(I_j)(P)$ for all $j \geq 0$, or $\exists G \supset A \in |D|$ such that $\bigsqcup_{i=1}^{\infty} I_i, P \models^+ G$, which by lemma 5.6 implies that $I_j, P \models^+ G$ for some $j \geq 0$, and so $A \in \text{pos}(S(I_j))(P)$. In either case we have $A \in \text{pos}(\bigsqcup_{i=1}^{\infty} T(I_i))(P)$.

If $A \in \text{neg}(T(\bigsqcup_{i=1}^{\infty} I_i))(P)$, then $\text{name}(A) \in \text{den}(N)$ and $\forall B \in |D|\ B \not\propto A$ and $\forall G \supset B \in |D|$ such that $B \propto A$, $\bigsqcup_{i=1}^{\infty} I_i, P \models^- G$, which by lemma 5.6 implies that $I_j, P \models^- G$ for some $j \geq 0$, and so $A \in \text{neg}(T(I_j))(P)$. Hence $A \in \text{neg}(\bigsqcup_{i=1}^{\infty} T(I_i))(P)$.

Hence, $T(\bigsqcup_{i=1}^{\infty} I_i) \sqsubseteq \bigsqcup_{i=1}^{\infty} T(I_i)$.

Hence $T(\bigsqcup_{i=1}^{\infty} I_i) = \bigsqcup_{i=1}^{\infty} T(I_i)$.

$\square$

Most importantly, the sequence of interpretations respects $\models^+$ and $\models^-$ as shown below.

Proposition 6.3 *Let P be a D_{HHF^-} derivation state and let G be a G_{HHF^-} goal formula. Then*

1. $T^k(I_\perp), P \models^+ G \Rightarrow T^j(I_\perp), P \models^+ G$ *for any* $j \geq k$
2. $T^k(I_\perp), P \models^- G \Rightarrow T^j(I_\perp), P \models^- G$ *for any* $j \geq k$

Thus our construction preserves important properties. In fact, it follows from the monotonicity of T and the fact that $\sqsubseteq$ is a chain-complete partial order that the least fixed point of T is $T^\omega(I_\perp)$ [1, 11]. However it is not hard to establish the same result directly, and as it is somewhat informative, we do so below.

Lemma 6.4 *Let T be defined as above, and so*

$$T^\omega(I_\perp) = \bigsqcup_{i=1}^{\infty} T^i(I_\perp) = T^1(I_\perp) \sqcup T^2(I_\perp) \sqcup T^3(I_\perp) \sqcup \ldots$$

Then $T^\omega(I_\perp)$ is a fixed point of T, i.e. $T(T^\omega(I_\perp)) = T^\omega(I_\perp)$.

Lemma 6.5 *Let J be any fixed point of T. Then for any $i \geq 1$, $T^i(I_\perp) \sqsubseteq J$.*

Proof: Clearly $I_\perp \sqsubseteq J$, and as T is monotonic, we have $T(I_\perp) \sqsubseteq T(J)$, and so $T^i(I_\perp) \sqsubseteq T^i(J)$. Now as J is a fixed point of T, we have that $J = T(J) = T^2(J) = \ldots = T^i(J)$, and so $T^i(I_\perp) \sqsubseteq J$. $\square$

From this lemma we may easily derive the desired result, given below.

Proposition 6.6 *$T^\omega(I_\perp)$ is the least fixed point of T.*

Thus even though our partial order does not form a lattice, our operator is continuous and $T^\omega(I_\perp)$ is its least fixed point. This result may be seen as evidence that our approach does not stray too wildly from the traditional methods.

We now show the relationship between our construction and the relations $\vdash_s$ and $\vdash_f$. First we show that $\vdash_s$ and $\vdash_f$ are sound with respect to the Kripke-like model.

The reader may wish to refer to Definition 2.4.

Proposition 6.7 *Let $\langle P, G \rangle$ be a D_{HHF^-} derivation pair where $P = \langle D, N \rangle$. Then*

1. $P \vdash_s G \Rightarrow T^\omega(I_\perp), P \models^+ G$

2. If G is negatable, then $P \vdash_f G \Rightarrow T^\omega(I_\perp), P \models\!\!\!\models^- G$

Note that the restriction in 2 is necessary. As $T^\omega(I_\perp)$ is an interpretation, enlarging the program must preserve what is known to be true and what is known to be false. On the other hand, enlarging a program may mean that a goal which originally failed does not fail in the larger program.

Proof: We proceed by induction on the depth of the **O**-derivation of G. The base case occurs when G is an atom A and the sequent $P \longrightarrow^+ A$ (resp. $P \longrightarrow^- A$) is initial.

1. As the sequent is initial, we have $A \in |D|$, and hence $A \in \text{pos}(T^1(I_\perp))(P)$, and so $T^\omega(I_\perp), P \models\!\!\!\models^+ A$.
2. As the sequent is initial, we have $\forall B \in |D|$ and $\forall G \supset B \in |D|$ $B \not\propto A$ and as A is negatable, we have $\text{name}(A) \in \text{den}(N)$, and hence $A \in \text{neg}(T^1(I_\perp))(P)$, and so $T^\omega(I_\perp), P \models\!\!\!\models^- A$.

Hence we assume that the proposition is true for all **O**-derivations of no more than a given depth. There are seven cases; as many of these are similar, we only present the argument for the cases A, $\neg A$ and $D \supset G$.

A:
1. If the base case does not hold, we have that $\exists G \supset A \in |D|$ such that $P \vdash_s G$, and by the hypothesis this implies that $T^\omega(I_\perp), P \models\!\!\!\models^+ G$. By lemma 5.6 we have that $T^k(I_\perp), P \models\!\!\!\models^+ G$ for some k, and so $T^{k+1}(I_\perp), P \models\!\!\!\models^+ A$, i.e. $T^\omega(I_\perp), P \models\!\!\!\models^+ A$.
2. If the base case does not hold, we have that $\forall B \in |D|$ $B \not\propto A$, and $\forall G \supset B \in |D|$ such that $B \propto A$, $P \vdash_f G$, and as G is negatable, by the hypothesis, $T^\omega(I_\perp), P \models\!\!\!\models^- G$. By lemma 5.6 we have that $T^k(I_\perp), P \models\!\!\!\models^- G$ for some k, and so $T^{k+1}(I_\perp), P \models\!\!\!\models^- A$, i.e. $T^\omega(I_\perp), P \models\!\!\!\models^- A$.

$\neg A$:
1. $P \vdash_s \neg A$ iff $P \vdash_f A$ and $\text{name}(A) \in \text{den}(N)$, and by the hypothesis, $T^\omega(I_\perp), P \models\!\!\!\models^- A$. By lemma 5.6, we have that $T^k(I_\perp), P \models\!\!\!\models^- A$ for some k, and so $T^k(I_\perp), P \models\!\!\!\models^+ \neg A$, i.e. $T^\omega(I_\perp), P \models\!\!\!\models^+ \neg A$.
2. $P \vdash_f \neg A$ iff $P \vdash_s A$, and by the hypothesis, $T^\omega(I_\perp), P \models\!\!\!\models^+ A$. By lemma 5.6, we have that $T^k(I_\perp), P \models\!\!\!\models^+ A$ for some k, and so $T^k(I_\perp), P \models\!\!\!\models^- \neg A$, i.e. $T^\omega(I_\perp), P \models\!\!\!\models^- \neg A$.

$D' \supset G$:
1. $\langle D, N\rangle \vdash_s D' \supset G$ iff $\langle D \cup \{D'\}, N\rangle \vdash_s G$ and $\text{names}(\text{heads}(D')) \subseteq \text{ass}(N)$, and so by the hypothesis $T^\omega(I_\perp), \langle D \cup \{D'\}, N\rangle \models\!\!\!\models^+ G$, i.e. $T^\omega(I_\perp), \langle D, N\rangle \models\!\!\!\models^+ D' \supset G$.
2. $\langle D, N\rangle \vdash_f D' \supset G$ iff $\langle D \cup \{D'\}, N\rangle \vdash_f G$ and $\text{names}(\text{heads}(D')) \subseteq \text{ass}(N)$, and as $D' \supset G$ is negatable, G is negatable, and so by the hypothesis $T^\omega(I_\perp), \langle D \cup \{D'\}, N\rangle \models\!\!\!\models^- G$, i.e. $T^\omega(I_\perp), \langle D, N\rangle \models\!\!\!\models^- D' \supset G$.

□

Note that this result may be thought of as demonstrating the compactness of $\vdash_s$ and $\vdash_f$, in that if $P \vdash_s G$, then $T^\omega(I_\perp), P \models^+ G$, and by Lemma 5.6 $T^k(I_\omega), P \models^+ G$ for some k, and similarly for $\vdash_f$ when G is negatable.

Next we show that operational provability is complete with respect to the Kripke-like model.

Proposition 6.8 *Let $\langle P, G\rangle$ be a D_{HHF-} derivation pair where $P = \langle D, N\rangle$. Then*

1. $T^\omega(I_\perp), P \models^+ G \Rightarrow P \vdash_s G$
2. $T^\omega(I_\perp), P \models^- G \Rightarrow P \vdash_f G$

Proof: By lemma 5.6 we have that

$$T^\omega(I_\perp), P \models^+ G \Rightarrow \exists k \text{ such that } T^k(I_\perp), P \models^+ G$$
$$T^\omega(I_\perp), P \models^- G \Rightarrow \exists k \text{ such that } T^k(I_\perp), P \models^- G$$

In each case, let k be the smallest such number.

We proceed to show 1 & 2 simultaneously by formal induction on the ordinal measure $\omega.k + n$, where n is the number of connectives in G.

The base case occurs when $n = 1$ and $k = 0$.

- $T^1(I_\perp), P \models^+ A$ implies that $A \prec |D|$, i.e. $A \in |D|$, and hence $P \vdash_s A$.
- $T^1(I_\perp), P \models^- A$ implies that $\forall B \in |D|$ and $\forall G \supset B \in |D|$, $B \not\propto A$, and so $P \vdash_f A$.

Hence we assume that the proposition is true for all programs and goals for which $\omega.k+n$ does not exceed a certain value.

There are seven cases; as above, these are somewhat repetitive, and so we only give the argument below for the cases A, $\neg A$ and $D \supset G$.

A:
1. $T^k(I_\perp), P \models^+ A$ implies that $\exists G \supset A \in |D|$ such that $T^{k-1}(I_\perp), P \models^+ G$, and by the hypothesis, $P \vdash_s G$, and so $P \vdash_s A$.
2. $T^k(I_\perp), P \models^- A$ implies that $\forall B \in |D|$ $B \not\propto A$ and $\forall G \supset B \in |D|$ such that $B \propto A$, we have $T^{k-1}(I_\perp), P \models^- G$, and by the hypothesis, $P \vdash_f G$, and so $P \vdash_f A$.

$\neg A$:
1. $T^k(I_\perp), P \models^+ \neg A$ implies that $T^k(I_\perp), P \models^- A$, and so $\forall B \in |D|$ $B \not\propto A$ and $\forall G \supset B \in |D|$ such that $B \propto A$, $T^{k-1}(I_\perp), P \models^- G$ and name$(A) \in$ den(N). By the hypothesis this implies that $P \vdash_f A$ and name$(A) \in$ den(N), i.e. $P \vdash_s \neg A$.
2. $T^k(I_\perp), P \models^- \neg A$ implies that $T^k(I_\perp), P \models^+ A$, and so $\exists G \supset A \in |D|$ such that $T^{k-1}(I_\perp), P \models^+ G$. By the hypothesis this implies that $P \vdash_s A$, and so $P \vdash_f \neg A$.

$D' \supset G$:
1. $T^k(I_\perp), \langle D, N\rangle \models^+ D' \supset G$ iff $T^k(I_\perp), \langle D \cup \{D'\}, N\rangle \models^+ G$ and $\langle D \cup \{D'\}, N\rangle \geq \langle D, N\rangle$, and by the hypothesis this implies that $\langle D \cup \{D'\}, N\rangle \vdash_s G$ and names(heads$(D')) \subseteq$ ass(N), i.e. $\langle D, N\rangle \vdash_s D' \supset G$.

2. $T^k(I_\perp), \langle D, N\rangle \models\!\!\!\models^- D' \supset G$ iff $T^k(I_\perp), \langle D \cup \{D'\}, N\rangle \models\!\!\!\models^- G$ and $\langle D \cup \{D'\}, N\rangle \geq \langle D, N\rangle$, and by the hypothesis this implies that $\langle D \cup \{D'\}, N\rangle \vdash_f G$ and names(heads(D')) $\subseteq$ ass(N), i.e. $\langle D, N\rangle \vdash_f D' \supset G$.

□

We now come to the main theorem.

Theorem 6.9 *Let $P = \langle D, N\rangle$ be a D_{HHF^-} derivation state and let G be a G_{HHF^-} goal formula. Then*

1. $P \vdash_s G \Leftrightarrow T^\omega(I_\perp), P \models\!\!\!\models^+ G$
2. *If G is negatable, then* $P \vdash_f G \Rightarrow T^\omega(I_\perp), P \models\!\!\!\models^- G$
3. $P \vdash_f G \Leftarrow T^\omega(I_\perp), P \models\!\!\!\models^- G$

Proof: Follows directly from propositions 6.7 and 6.8. □

7 Conclusions and Further Work

We have seen how the Negation as Failure rule may be incorporated into the framework of hereditary Harrop formulae, and how this extension is more intricate than in some other cases due to the possibility that the program may grow during execution. We have also seen how consideration of how to implement universal quantifiers in goals influences the technical choices to be made, and how we may make both of these extensions compact. We have also seen how a Kripke-like model may be constructed in such a way that truth in the model corresponds to derivability from programs. The fact that this construction takes at most ω iterations may be seen as evidence of the compactness of our approach. This property is essentially due to our treatment of non-ground terms, in that we do not consider only the ground instances of a program, but all instances, including both ground and non-ground ones.

Whilst our notion of universal quantification has been extensional, it should not be hard to derive similar results for the intensional case. Indeed, given definitions of $\vdash_s$ and $\vdash_f$ which use the intensional version, and corresponding alterations to $\models\!\!\!\models^+$ and $\models\!\!\!\models^-$, the results stated herein will, it seems, all still hold. Thus we may think of our approach, in this limited sense at least, being parameterised by the choice of consequence relation.

We have seen how inconsistencies, such as extending the definition of completely defined predicates, causes some problems from the model-theoretic point of view, but the operational (or proof-theoretic) one does not have such problems. Whilst the notion of completely defined predicates seems natural enough, the corresponding restrictions on the operational concepts does not seem as natural. This suggests that our approach to model theory may need to become more sophisticated in order to deal with inconsistent programs. A similar remark may be made about extending the class of programs to include negations which occur positively in programs, such as in the "program" $\{p, \neg p\}$.

One feature of the Kripke-like model is that it naturally incorporates the notion of extensions to the program, and so it may be useful for studying the formal development of programs. Another interesting area of investigation is to use this model as a way of investigating the "natural" logic in which to interpret hereditary Harrop formulae. It is known that this logic is slightly stronger than intuitionistic logic [8]; the proof-theory of this logic is as yet somewhat unnatural, and so the Kripke-like model seems a likely way to investigate this logic. The main difference between this logic and intuitionistic logic is that we require the *Independence of Premise* (IP) axioms to hold. These axioms are often studied in conjunction with *Markov's Principle* (MP) (which may be stated as $\neg\neg A \Rightarrow A$), which seems appropriate for NAF. Thus it would seem that the natural logic for hereditary Harrop formulae with NAF is the intermediate logic found by adding IP and MP to intuitionistic logic. This theme will be taken up in a later paper.

8 Acknowledgments

My thanks go to Dale Miller for many interesting discussions, and to Don Sannella for his constant encouragement. Comments from Colin Stirling were also very helpful.

Much of this work was done whilst the author was a student in the Department of Computer Science at the University of Edinburgh. The support of the Australian Research Council through the Machine Intelligence Project is gratefully acknowledged.

References

[1] S. Abian and A.B. Brown, A Theorem on Partially Ordered Sets with Applications to Fixed Point Theorems, *Canadian Journal of Mathematics*:13:78-82, 1961.

[2] K. Clark, Negation as Failure, in *Logic and Databases*, H. Gallaire and J. Minker (eds.), Plenum Press, 1978.

[3] M.H. van Emden and R.A. Kowalski, The Semantics of Predicate Logic as a Programming Language, *Journal of the Association for Computing Machinery* 23:4:733-742, October, 1976.

[4] M. Fitting, A Kripke-Kleene Semantics for Logic Programming, *Journal of Logic Programming* 2:295-312, 1985.

[5] M. Falaschi, G. Levi, M. Martelli and C. Palamidessi, A New Declarative Semantics for Logic Languages, *Proceedings of the International Conference and Symposium on Logic Programming* 993-1005, Seattle, August, 1988.

[6] T. Flannagan, The Consistency of Negation as Failure, *Journal of Logic Programming* 3:93-114, 1986.

[7] D. Gabbay, N-Prolog: An Extension of Prolog with Hypothetical Implication.II. Logical Foundations and Negation as Failure, *Journal of Logic Programming* 2:251-283, 1985.

[8] J. Harland, On Normal Forms and Equivalence for Logic Programs, *Proceedings of the Joint International Conference and Symposium on Logic Programming* 146-160, Washington D.C., November, 1992.

[9] J. Harland, *On Hereditary Harrop Formulae as a Basis for Logic Programming*, Ph.D. Thesis, Department of Computer Science, University of Edinburgh, July, 1991.

[10] K. Kunen, Negation in Logic Programming, *Journal of Logic Programming* 4:289-308, 1987.

[11] J-L. Lassez, V.L. Nyugen and E.A. Sonenberg, Fixed Point Theorems and Semantics: A Folk Tale, *Information Processing Letters*:14:112-116.

[12] J.W. Lloyd, *Foundations of Logic Programming*, Springer-Verlag, Berlin, 1984.

[13] L.T. McCarty, Clausal Intuitionistic Logic I. Fixed Point Semantics, *Journal of Logic Programming* 5:1:1-32, 1988.

[14] D.A. Miller, A Logical Analysis of Modules in Logic Programming, *Journal of Logic Programming* 6:79-108, 1989.

[15] D.A. Miller, G. Nadathur, F. Pfenning and A. Scedrov, Uniform Proofs as a Foundation for Logic Programming, *Annals of Pure and Applied Logic* 51:125-157, 1991.

[16] R. Reiter, On Closed World Databases, in *Logic and Databases*, H. Gallair and J. Minker (eds.), Plenum Press, 1978.

[17] C. Smorynski, Applications of Kripke Models, Chapter V in [19].

[18] S.-A. Tärnlund, Horn Clause Computability, *BIT* 17:215-226, 1977.

[19] A.S. Troelstra, *Metamathematical Investigations of Intuitionistic Arithmetic and Analysis*, Lecture Notes in Mathematics 344, Springer-Verlag, Berlin, 1973.