Logique & Analyse **147-148** (1994), 379-406

A LOGIC-BASED MODELLING OF PROLOG RESOLUTION SEQUENCES INCLUDING THE NEGATION AS FAILURE RULE

John S. JEAVONS and John N. CROSSLEY

Introduction

In Jeavons and Crossley [5] we introduced a logic based model for the *procedural* aspect of resolution based logic programming. A formal system L_I was introduced and we specified an injective mapping from the set of all finite *SLD*-derivations into the set of L_I theorems. The system L_I lacked the usual structural rules of weakening, contraction and exchange. The system was chosen to represent the essentially sequential nature of an *SLD*-derivation. A single resolution step from a goal $\mathbf{G_0}$ to its immediate successor goal $\mathbf{G_1}$ is considered to comprise three sub-steps:

(i) application of the computation rule to select an atom from $\mathbf{G_0}$
(ii) resolution with the head of a selected program clause, $\mathbf{H_1}$
(iii) ordering of the atoms in the resultant goal $\mathbf{G_1}$.

In the *declarative* semantics we have that the sentence G_1 (representing the goal $\mathbf{G_1}$) is a logical consequence of the sentences G_0, H_1 (representing the goal $\mathbf{G_0}$ and the clause $\mathbf{H_1}$, respectively). The computation rule and ordering of the atoms play no rôle here as the PROLOG comma is interpreted as the (commutative, idempotent) connective &. By contrast, in our approach which models the sequential aspect of *SLD*-derivations, the computation rule and ordering of the atoms now assume a semantic significance and the resolution step above is represented by a provable sequent

$$G_0, COMP_1, H_1, R_1 =\vdash G_1$$

where $=\vdash$ is the separator of the antecedent and succedent (see below for the distinction between $=\vdash$ and $\vdash$) $COMP_1$ is a sentence corresponding to the application of the computation rule and R_1 is a sentence corresponding to the arrangement of atoms in the resultant goal. The comma here cannot be interpreted as & but instead is interpreted as a new connective $\bullet$, so that $A \bullet B$ is read as A followed by B.

The non-standard separator $= \vdash$ was introduced for the following reason. In conventional logic an expression $A_1, \ldots, A_n \vdash B$ may either denote a formal sequent, or may stand for the assertion that there is a derivation of B from the non-logical axioms $A_1, \ldots, A_n$ The two notions are equivalent in logic equipped with the usual structural rules, but not in our system. Thus we choose to write $A_1, \ldots, A_n = \vdash B$ for a formal sequent with the sequence $A_1, \ldots, A_n$ as antecedent, and we write $\{A_1, \ldots, A_n\} \vdash B$ for the assertion that we can derive the sequent $= \vdash B$ from the set of axioms $\{= \vdash A_1, \ldots, = \vdash A_n\}$ where each axiom may occur any number of times as a leaf node in the derivation tree.

In this paper we extend the analysis in [5] to *SLDNF*-derivations (see Lloyd [18]). If the computation rule selects a negative ground literal $\sim P$ from a goal then an attempt is made to construct a finitely failed *SLDNF*-tree with root node $\leftarrow P$. If the construction succeeds, the sub-goal $\sim P$ succeeds, and if the construction fails finitely by obtaining a success branch for P, then the goal $\leftarrow \sim P$ fails. This attempted construction of a finitely failed tree is thus regarded as a test carried out in the course of an *SLD*-derivation. In order to extend our system to model this construction we first enrich our logic to a modal logic which allows us to re-instate the abandoned structural rules for the case of formulae of the form $!A$ where ! is a new modal connective. We are then able to associate a modal formula to a given (finite) *SLDNF*-tree, and thus we can represent negation as failure in our enriched logic.

In section 1.1 we define our new logical system and in section 1.2 we give a semantics for the logic. In section 2 we consider the representation of negation as failure in our system and specify an injective mapping from the set of all finite *SLDNF*-derivations into the set of theorems in our system, thereby extending the analysis in [5] to *SLDNF*-derivations.

1.1 *The formal system* L_{IM}

We take as our basic alphabet the following symbols:

(i) a countable set of variables $x, y, \cdots$;
(ii) function symbols $f, g, \cdots$;
(iii) predicate symbols $P, Q, \cdots$;
(iv) propositional constants $\bot, \top, \mathbf{0}, \mathbf{1}$;
(v) binary connectives $\bullet, \supset, \supset', \&, \oplus$;
(vi) unary modal connectives !, ? ;
(vii) quantifiers $\forall, \exists$;
(viii) punctuation symbols (,) .

Terms and formulae are defined in the standard manner. The axioms and rules are presented below, where upper case Greek letters stand for sequences of formulae. ! Δ stands for a finite sequence of formulae of the form $!A_1, !A_2, \cdots, !A_n$.

Axioms

(i) $A = \vdash A$ (for any formula A)
(ii) $\Delta, \bot, \Sigma = \vdash A$ (for any finite sequences Δ, Σ and any formula A)
(iii) $= \vdash \mathbf{1}$
(iv) $\mathbf{0} = \vdash$
(v) $\Gamma = \vdash \top$ (for any finite sequence Γ).

Rules

$$(\text{cut})\ \frac{\Gamma = \vdash A \quad \Delta, A, \Sigma = \vdash B}{\Delta, \Gamma, \Sigma = \vdash B}$$

$$(\mathbf{1}\text{-}w)\ \frac{\Gamma, \Delta = \vdash B}{\Gamma, \mathbf{1}, \Delta = \vdash B} \qquad (\mathbf{0}\text{-}w)\ \frac{\Gamma = \vdash}{\Gamma = \vdash \mathbf{0}}$$

$$(\supset\text{-}L)\ \frac{\Gamma = \vdash A \quad \Delta, B, \Sigma = \vdash C}{\Delta, A \supset B, \Gamma, \Sigma = \vdash C} \qquad (\supset\text{-}R)\ \frac{\Gamma, A, = \vdash B}{\Gamma = \vdash A \supset B}$$

$$(\supset'\text{-}L)\ \frac{\Gamma = \vdash A \quad \Delta, B, \Sigma = \vdash C}{\Delta, \Gamma, A \supset' B, \Sigma = \vdash C} \qquad (\supset'\text{-}R)\ \frac{A, \Gamma = \vdash B}{\Gamma = \vdash A \supset' B}$$

$$(\bullet\text{-}L)\ \frac{\Gamma, A, B, \Sigma = \vdash C}{\Gamma, A \bullet B, \Sigma = \vdash C} \qquad (\bullet\text{-}R)\ \frac{\Gamma = \vdash A \quad \Delta = \vdash B}{\Gamma, \Delta = \vdash A \bullet B}$$

$$(\&_1\text{-}L)\ \frac{\Gamma, A, \Delta = \vdash C}{\Gamma, A\&B, \Delta = \vdash C}$$

$$(\&_1\text{-}R)\ \frac{\Gamma = \vdash A \quad \Gamma = \vdash B}{\Gamma = \vdash A\&B}$$

$$(\&_2\text{-}L)\ \frac{\Gamma, B, \Delta = \vdash C}{\Gamma, A\&B, \Delta = \vdash C}$$

$$(\oplus_1\text{-}R)\ \frac{\Gamma =\vdash A}{\Gamma =\vdash A\oplus B}$$

$$(\oplus\text{-}L)\ \frac{\Gamma, A, \Delta =\vdash C \quad \Gamma, B, \Delta =\vdash C}{\Gamma, A\oplus B, \Delta =\vdash C}$$

$$(\oplus_2\text{-}R)\ \frac{\Gamma =\vdash B}{\Gamma =\vdash A\oplus B}$$

$$(!\text{-}L)\ \frac{\Gamma, A, \Delta =\vdash C}{\Gamma, !A, \Delta =\vdash C} \qquad (!\text{-}R)\ \frac{!\Gamma =\vdash A}{!\Gamma =\vdash !A}$$

$$(?\text{-}L)\ \frac{!\Gamma, A, !\Delta =\vdash ?B}{!\Gamma, ?A, !\Delta =\vdash ?B} \qquad (?\text{-}R)\ \frac{\Gamma =\vdash B}{\Gamma =\vdash ?B}$$

$$(!\text{-}w)\ \frac{\Gamma, \Delta =\vdash B}{\Gamma, !A, \Delta =\vdash B} \qquad (?\text{-}w)\ \frac{\Gamma =\vdash}{\Gamma =\vdash ?A}$$

$$(!\text{-}ex)\ \frac{\Gamma, !A, \Delta =\vdash C}{\Gamma', !A, \Delta' =\vdash C}$$

where the sequence $\Gamma', !A, \Delta'$ is obtained from $\Gamma, !A, \Delta$ by exchanging $!A$ with any formula in Γ or Δ.

$$(!\text{-}c)\ \frac{\Gamma, !A, !A, \Delta =\vdash C}{\Gamma', !A, \Delta =\vdash C}$$

$$(\forall\text{-}L)\ \frac{\Gamma, A(t), \Delta =\vdash B}{\Gamma, \forall xA(x), \Delta =\vdash B} \qquad (\forall\text{-}R)\ \frac{\Gamma =\vdash A(y)}{\Gamma =\vdash \forall xA(x)}$$

$$(\exists\text{-}L)\ \frac{\Gamma, A(y), \Delta =\vdash B}{\Gamma, \exists xA(x), \Delta =\vdash B} \qquad (\exists\text{-}R)\ \frac{\Gamma =\vdash A(t)}{\Gamma =\vdash \exists xA(x)}$$

As usual, for the rules $(\forall - R)$ and $(\exists - L)$ we require that the eigenvariable y does not occur free in the conclusion.

Remark 1.

The absence of an exchange rule allows two implications $\supset$ and $\supset'$. The comma in the antecedent represents the connective $\bullet$ and the constant 1 is neutral for this connective in the sense that we have $\Gamma =\vdash A$ is provable if, and only if, $\Gamma =\vdash A \bullet \mathbf{1}$ is provable, and also $\Gamma =\vdash A$ is provable if, and only if, $\Gamma =\vdash \mathbf{1} \bullet A$ is provable. Similarly the constant $\top$ is neutral for the connective &, and the constant $\bot$ is neutral for the connective $\oplus$. We also have that $\Gamma =\vdash \mathbf{0}$ is provable if, and only if, $\Gamma =\vdash$ is provable.

Logics lacking some (or all) structural rules have been considered in Lambek [7], Komori [6], Ono [10], and Girard's [3] linear logic is now a well-known logic lacking both the contraction and weakening rules. Recently non-commutative versions of linear logic have been considered; see for example Abrusci [1], Yetter [12]. Our system is taken from Ono [11], where a number of substructural logics are considered. The modal connective ! enables us to relate the notions of $\Delta \vdash B$ and $\Delta =\vdash B$ as shown by the following lemma.

Lemma. If $A_1, \cdots, A_m$ are L_{IM} sentences then $\{A_1, \cdots, A_m\} \vdash B$ holds if, and only if, the sequent $!A_1, \cdots, !A_n =\vdash B$ is provable.

Proof. Suppose that $\{A_1, \cdots, A_m\} \vdash B$ holds. Then we have a derivation tree $\mathcal{T}$ with root node $=\vdash B$ such that all leaf nodes of $\mathcal{T}$ are either members of $\{=\vdash A_i : i = 1, \cdots, m\}$ or are logical axiom sequents. We establish that if $\Delta =\vdash C$ is any sequent which occurs in $\mathcal{T}$ then the sequent $!A_1, \cdots, !A_m, \Delta =\vdash C$ is a provable sequent of L_{IM}. Consider first any leaf node of $\mathcal{T}$, if the leaf node is a logical axiom $\Delta =\vdash C$ then by application of the (! - w) rule we can construct a proof of the sequent $!A_1, \cdots, !A_m, \Delta =\vdash C$; on the other hand if the leaf node of $\mathcal{T}$ is $=\vdash A_i$ then we can construct a proof of $!A_1, \cdots, !A_m =\vdash A_i$ by taking the logical axiom $A_i =\vdash A_i$ and applying the rules (! - L) and (! - w). Consider now any two-premiss inference in $\mathcal{T}$,

$$\frac{\Sigma =\vdash A \quad \Pi =\vdash B}{\Delta =\vdash C}$$

and assume that both $!A_1, \cdots, !A_m, \Sigma =\vdash A$ and $!A_1, \cdots, !A_m, \Pi =\vdash B$ are provable. Then we can construct a proof of $!A_1, \cdots, !A_m, \Delta =\vdash C$ by applying the inference rule to yield two copies of the sequence $!A_1, \cdots, !A_m$ in the antecedent of the conclusion, and then application of (! - ex) and (! - c) may be employed to give the proof of the sequent $!A_1, \cdots, !A_m, \Delta =\vdash C$. For a one-premiss rule we have no difficulty in duplication; if the inference in $\mathcal{T}$ is

$$\frac{\Sigma = \vdash A}{\Delta = \vdash C}$$

then the corresponding inference

$$\frac{!A_1, \cdots, !A_m, \Sigma = \vdash A}{!A_1, \cdots, !A_m, \Delta = \vdash C}$$

is correct since the added formulae are all modal so that (! - R) and (? - L) can still be applied, and since the A_i are closed we can still apply the quantifier rules.

Thus we conclude that if $\Delta = \vdash C$ occurs in $\mathcal{T}$ then $!A_1, \cdots, !A_m, \Delta = \vdash C$ is provable and hence $!A_1, \cdots, !A_m = \vdash B$ is provable if $\{A_1, \cdots, A_m\} \vdash B$ holds.

Now for the converse, suppose that $!A_1, \cdots, !A_m = \vdash B$ is provable. If we introduce the non-logical axiom $= \vdash A_i$ we can apply the rule (! - R) and then apply cuts to eliminate the formulae in the antecedent of $!A_1, \cdots, !A_m = \vdash B$ to yield a derivation of $= \vdash B$ from the non-logical axioms $\{= \vdash A_i : i = 1, \cdots, m\}$.

Corollary. $\{A_1, \cdots, A_m\} \vdash B$ holds if, and only if, the sequent $!(A_1 \& \cdots \& A_n) = \vdash B$ is provable.

Proof. This follows from the lemma by observing that the sequents $!(A_1 \& \cdots \& A_n) = \vdash !A_1 \bullet \cdots \bullet !A_m$ and $!A_1, \cdots, !A_m = \vdash !(A_1 \& \cdots \& A_m)$ are both provable, so that $!A_1 \bullet \cdots \bullet !A_m$ and $!(A_1 \& \cdots \& A_m)$ are logically equivalent formulae.

Remark 2. Negation in L_{IM}

We define $\neg A$ as standing for $A \supset \bot$. A weaker negation ~ A may be defined by $A \supset 0$, and $\neg A = \vdash$ ~ A is provable, but not the other way around. Similarly there is a retrograde negation, $A \supset' \bot$. Neither of these forms of negation seems to be suitable for modelling negation as failure in our logic because of the following. In logic programming the failure of a sub-goal $\leftarrow B$ from goal $\leftarrow A, B, C$ results in the failure of the goal. This is sound with respect to classical logic since $\neg B \supset \neg(A \& B \& C)$ is a theorem. However, in our system with the PROLOG comma interpreted as the connective $\bullet$, the formula $\neg B \supset \neg(A \bullet B \bullet C)$ is not a theorem. On the other hand, the formula $! \neg B \supset !\neg(A \bullet B \bullet C)$ is a theorem and we therefore choose a translation from PROLOG clauses to formulae in L_{IM} in which a clause $P \leftarrow Q, \sim R, S$ translates to the universal closure of $(Q \bullet !\neg R \bullet S) \supset P$.

1.2 *Semantics*

We first give an algebraic semantics, then a Kripke-style semantics. We follow the development in Ono [11].

1.2.1 *Algebraic semantics*

We define a *unital quantale* as a structure $\mathbf{B} = \langle B, \vee, \otimes, 1 \rangle$ where $\langle B, \vee \rangle$ is a complete lattice and $\langle B, \otimes, 1 \rangle$ is a monoid in which $\otimes$ distributes on both sides over arbitrary $\vee$, that is,

$$b \otimes \bigvee_i c_i = \bigvee_i (b \otimes c_i), \bigvee_i c_i \otimes b = \bigvee_i (c_i \otimes b)$$

hold for arbitrary elements b, c_i of B.

Abrusci [1] and Yetter [12] have investigated the semantics of non-commutative linear logic using quantales. In [1] a special type of quantale called a non-commutative classical phase space is introduced to provide a semantics for non-commutative linear logic (i.e. linear logic with the exchange rule deleted). In [12] another special type of quantale is introduced to give a semantics for cyclic non-commutative linear logic (i.e. linear logic in which the exchange rule is restricted to a cyclic exchange).

Given a unital quantale $\mathbf{B}$ we define the binary operations $\rightarrow$ and $\rightarrow'$ on B by

$$x \rightarrow y = \vee \{z : z \otimes x \leq y\}$$
$$x \rightarrow' y = \vee \{z : x \otimes z \leq y\}$$

We write $a \vee b$ for $\vee \{a, b\}$, and similarly $a \wedge b$ for $\wedge \{a, b\}$. Also the symbol 0 will stand for a fixed designated element of B. A modal unital quantale is a structure $\mathbf{B_M} = \langle B, \vee, \otimes, 1, 0, ?, ! \rangle$ in which the unital quantale is enriched by the unary operations ?, ! which satisfy the conditions (i) -- (x) below.

(i) $!a \leq a$
(ii) $!1 = 1$
(iii) $!(a \wedge b) = !a \otimes !b$
(iv) $!a \leq !!a$
(v) $!a \otimes b = b \otimes !a$
(vi) $!(a \rightarrow b) \leq ?a \rightarrow ?b$

(vii) $a \leq ?a$
(viii) $??a \leq ?a$
(ix) $0 \leq ?a$
(x) $?0 = 0$

where a, b are arbitrary elements of B.

Lemma 1.In every modal unital quantale, if elements a, b, c satisfy $!a \otimes b \leq ?c$, then $!a \times ?b \leq ?c$ also holds.

Proof. By definition of $\rightarrow$, if $!a \otimes b \leq ?c$, then $!a \leq (b \rightarrow ?c)$. Conditions (i)-(iii) enable us to show that ! is order preserving so we have $!!a \leq !(b \rightarrow ?c)$, and then condition (vi) enables us to conclude that $!!a \leq (?b \rightarrow ??c)$. Now $!!a = !a$ and $??c = ?c$ follow from (i), (iv), (vii) and (viii), so we have $!a \leq (?b \rightarrow ?c)$ and this in turn gives the desired result, $!a \otimes ?b \leq ?c$.

The following lemma in Ono 11 gives a condition for extending a unital quantale to a modal unital quantale.

Lemma 2. Let $B = \langle B, \vee, \otimes, 1 \rangle$ be a unital quantale and let R, S be subsets of B satisfying the conditions:

(1) R is closed under $\otimes$
(2) $r \otimes r = r$, for every $r \in R$
(3) $r \otimes b = b \otimes r$, for every $r \in R$ and every $b \in B$
(4) 1 is the greatest element of R
(5) 0 is the least element of S
(6) If $r \in R$, s $\in S$, then $r \rightarrow s \in S$.

Then if we define the unary operations on B by

$$!b = \vee \{r \in R : r \leq b\}$$
$$?b = \wedge \{s \in S : b \leq s\}$$

for every $b \in B$, then $B_M = \langle B, \vee, \otimes, 1, 0, ?, ! \rangle$ is a modal unital quantale.

A *structure* for our logic consists of a pair $\langle B_M, U \rangle$ where B_M is a modal unital quantale and U is a pre-interpretation for our language, that is, U consists of a non-empty set equipped with functions f^U for each function symbol of our language. We denote by $L_{IM}(U)$ the language formed from L_{IM} by adjoining a new set of constants $\{C_u : u \in U\}$ to name each element

of U. An *assignment* F of $\langle B_M, U\rangle$,is a mapping from the set of all closed atomic formulae of $L_{IM}(U)$ into B which satisfies

$$\begin{aligned} F(\bot) &= \bigwedge B = \text{the least element of } B \\ F(\top) &= \bigvee \mathrm{B} = \text{the greatest element of } B \\ F(\mathbf{1}) &= 1 \\ F(\mathbf{0}) &= 0 \end{aligned}$$

The assignment F is extended to all closed formulae by defining

$$\begin{aligned} F(A \bullet B) &= F(A) \otimes F(B) \\ F(A \supset B) &= F(A) \rightarrow F(B) \\ F(A \supset' B) &= F(A) \rightarrow' F(B) \\ F(A \oplus B) &= F(A) \vee F(B) \\ F(A \,\&\, B) &= F(A) \wedge F(B) \\ F(\forall x A(x)) &= \bigwedge_{u \in U} F\left(A\left[x / c_u\right]\right) \\ F(\exists x A(x)) &= \bigvee_{u \in U} F\left(A\left[x / c_u\right]\right) \\ F(!A) &= !F(A) \\ F(?A) &= ?F(A) \end{aligned}$$

We say that a formula A is *true* under the assignment F if $1 \leq F\left(\hat{A}\right)$, where $\hat{A}$ is the universal closure of A. A formula A is *valid* if A is true under all assignments in all structures $\langle B_M, U\rangle$. We say that a sequent $A_1, \cdots, A_n = \vdash B$ is valid if the formula $\left(A_1 \bullet \cdots \bullet A_n\right) \supset B$ is valid. For the case $n = 0$ we take $1 \supset B$, and if the succedent is empty we take $A_1 \bullet \cdots \bullet A_n \supset 0$.

Soundness and Completeness

In order to establish soundness of L_{IM} with respect to validity in modal unital structures, we verify that all axioms are valid and that the rules of inference preserve validity. We treat the case of the rule (? - L) and leave the remaining cases as exercises. Suppose then that the sequent $!\Gamma, A, !\Delta = \vdash ?B$ is valid. We have to verify that the sequent $!\Gamma, ?A, !\Delta = \vdash ?B$ is also valid. Let $!\hat{\gamma}_1, \cdots !\hat{\gamma}_n, \hat{A}, !\hat{\delta}_1, \cdots !\hat{\delta}_m = \vdash ?\hat{B}$ be any closed instance of the sequent $!\Gamma, A, !\Delta = \vdash ?B$ obtained by a uniform replacement of free variables by the new constants used to name elements of U. Suppose further that under an assignment F we have $F\left(\hat{\gamma}_1\right) = c_i$ for $i = 1, \cdots, n$, and $F\left(\hat{\delta}_j\right) = d_j$, for $j = 1, \cdots, m$ and $F\left(\hat{A}\right) = a$, $F\left(\hat{B}\right) = b$. Since the formula $!\hat{\gamma}_1 \bullet \cdots \bullet !\hat{\gamma}_n \bullet A \bullet !\hat{\delta}_1 \bullet \cdots \bullet !\hat{\delta}_m \supset ?\hat{B}$ is true, we have $\left(!c_1 \otimes \cdots \otimes !c_n \otimes a \otimes !d_1 \otimes \cdots \otimes !d_m \rightarrow ?b\right) \geq 1$ and this is equivalent to

$$!c_1 \otimes \cdots \otimes !c_n \otimes a \otimes !d_1 \otimes \cdots \otimes !d_m \leq ?b$$

Now (using condition (iii) in the definition of the modal operations) this last condition is equivalent to

$$!(c_1 \wedge c_2 \wedge \cdots \wedge c_n) \otimes a \otimes !(d_1 \wedge d_2 \wedge \cdots \wedge d_m) \leq ?b,$$

and then (using condition (v)) we can write this as

$$!(c_1 \wedge c_2 \wedge \cdots c_n \wedge d_1 \wedge d_2 \wedge \cdots d_m) \otimes a \leq ?b$$

Finally, from Lemma 1, the relation $! e \otimes a \leq ?b$ implies $! e \otimes ?a \leq ?b$ and this suffices to establish that the sequent

$$!\hat{\gamma}_1, \cdots !\hat{\gamma}_n, ?\hat{A}, !\hat{\delta}_1, \cdots !\hat{\delta}_m = \vdash ?\hat{B}$$

is also true. Hence the rule (? - L) preserves validity.

To establish completeness with respect to modal unital quantales we must exhibit a structure $\langle B_M, U \rangle$ and an assignment F with the property that if $A_1, \cdots, A_n = \vdash B$ is not provable then the formula $(A_1 \bullet \cdots \bullet A_n) \supset B$ is not true under the assignment. We first need to define the notion of a closure operation φ on a unital quantale. If $\langle B, \vee, \otimes, 1 \rangle$ is a unital quantale and $\mathfrak{C}$ is a unary operation on B we say $\mathfrak{C}$ is a closure operation if it satisfies the conditions

(1) $b_1 \leq \mathfrak{C}(b_1)$

(2) $b_1 \leq b_2$ implies $\mathfrak{C}(b_1) \leq \mathfrak{C}(b_2)$

(3) $\mathfrak{C}\left(\mathfrak{C}(b_1)\right) \leq \mathfrak{C}(b_1)$

(4) $\mathfrak{C}(b_1) \otimes \mathfrak{C}(b_2) \leq \mathfrak{C}(b_1 \otimes b_2)$.

for all $b_1, b_2 \in B$. An element b is $\mathfrak{C}$-closed if $\mathfrak{C}(b) = b$.

Any closure operation $\mathfrak{C}$ on a unital quantale satisfies the properties

$$\mathfrak{C}\left(\bigvee_i \mathfrak{C}(b_1)\right) = \mathfrak{C}\left(\bigvee_i b_i\right)$$

$$\mathfrak{C}\left(\bigwedge_i \mathfrak{C}(b_1)\right) = \bigwedge_i \mathfrak{C}(b_1)$$

$$\mathfrak{C}\big(\mathfrak{C}(b_1)\otimes\mathfrak{C}(b_2)\big)=\mathfrak{C}(b_1\otimes b_2)$$

If $\mathfrak{C}(B)$ denotes the set of $\mathfrak{C}$-closed elements of B, and if we define the operation $\otimes_{\mathfrak{C}}$ on $\mathfrak{C}(B)$ by $b_1\otimes_{\mathfrak{C}} b_2=\mathfrak{C}(b_1\otimes b_2)$, and also define $\bigvee_{\mathfrak{C}} b_i=\mathfrak{C}\left(\bigvee b_i\right)$, then the structure $\langle\mathfrak{C}(B), \bigvee_{\mathfrak{C}}, \otimes_{\mathfrak{C}}, \mathfrak{C}(1)\rangle$ is a unital quantale (see Niefield and Rosenthal [9]).

Another construction we employ is the following.

Let $\langle M, *, 1\rangle$ be a monoid and denote by $P(M)$ the power set of M. For $X, Y\subseteq M$ define the binary operation $\cdot$ on $P(M)$ by $X\cdot Y=\{x*y : x\in X, y\in Y\}$. Then $\langle P(M), \bigcup, \cdot, \{1\}\rangle$ forms a unital quantale.

We now begin our construction of the structure $\langle B_M, U\rangle$ required to establish the completeness property. Without loss of generality we assume that our language contains at least one constant symbol and one function symbol. We take as our universe U the Herbrand universe generated by the constant symbols and function symbols of L_{IM}. Thus the expanded language $L_{IM}(U)$ contains an infinite number of new constants to name the elements of the Herbrand universe. We now define the Lindenbaum algebra LA of the logic $L_{IM}(U)$. Define an equivalence relation on the set of closed formulae of $L_{IM}(U)$ by A ≡ B if, and only if, both $A=\vdash B$ and $B=\vdash A$ are provable. We denote by [A] the equivalence class containing A, and we let $\mathcal{V}=\{[A] : A$ is a closed formula of $L_{IM}(U)\}$. We can impose a lattice structure on $\mathcal{V}$ by defining $[A]\vee[B]=[A\otimes B]$ and $[A]\wedge[B]=[A\&B]$, the greatest element of the lattice is $[\top]$and the least element is $[\bot]$, and $[C]\leq[D]$ if, and only if, $C=\vdash D$ is provable. Furthermore, $\mathcal{V}$ is equipped with a monoid structure by defining $[A]*[B]=[A\bullet B]$, for which [1] is the neutral element. The operation $*$ distributes on both sides over finite joins. We further equip $\mathcal{V}$ with two operations $\rightarrow, \rightarrow'$ by defining $[A]\rightarrow[B]=[A\supset B]$, and $[A]\rightarrow'[B]=[A\supset' B]$. It is readily verified that we have $[C]\leq[A]\rightarrow[B]$ if, and only if, $[C]*[A]\leq[B]$, and similarly $[C]\leq[A]\rightarrow'[B]$ if, and only if, $[A]*[C]\leq[B]$. The structure $\mathbf{LA}=\langle\mathcal{V}, \rightarrow, \rightarrow', \vee, \wedge, *, [1], [0],, \top, \bot\rangle$ is an example of an algebra called a full Lambek algebra in Ono [11]. The lattice structure of **LA** is not a complete lattice.

Now if we consider just the monoidal structure $\mathbf{LA}^{-}=\langle\mathcal{V}, *, [\mathbf{1}]\rangle$ then by the preceding remarks the structure $\langle P(\mathcal{V}), \bigcup, \cdot, \{[1]\}\rangle$ is a unital quantale. We now specify a closure operation $\mathfrak{C}$ on this quantale *via* the Dedekind-MacNeille completion of the lattice structure of **LA** (see [4]). For any $X\subseteq\mathcal{V}$ define $\mathfrak{C}(X)$ to be the set of lower bounds of the set of upper bounds of the set X, so that $x\in\mathfrak{C}(X)$ if, and only if, for every $y\in\mathcal{V}$ if $z\leq y$ holds for all $z\in X$, then $x\leq y$. We must verify that $\mathfrak{C}$ is a closure operation on the quantale $\langle P(\mathcal{V}), \bigcup, \cdot, \{[1]\}\rangle$, the partial order in this quantale being set inclusion. Conditions (1) - (3) for a closure operation are easily verified and we shall only establish condition (4). We have to establish that

if $X, Y \in P(\mathcal{V})$ then $\mathfrak{C}(X)\cdot\mathfrak{C}(Y)\subseteq\mathfrak{C}(X\cdot Y)$ Let $z\in\mathfrak{C}(X)\cdot\mathfrak{C}(Y)$ so that $z = x*y$ for some $x\in\mathfrak{C}(X)$, $y\in\mathfrak{C}(Y)$. If w is any upper bound of $(X\cdot Y)$, we have to establish that $x * y \leq w$. Suppose then $x' * y' \leq w$ holds for all $x' \in X$, $y' \in Y$, then by the definition of the operation $\rightarrow$ in LA we have $x' \leq y' \rightarrow w$ holds for all $x' \in X$, so that $x \leq y' \rightarrow w$ holds for all $y' \in Y$. But $x \leq y' \rightarrow w$ holds if, and only if, $x * y' \leq w$. and this condition is equivalent to $y' \leq x \rightarrow' w$. But if $y' \leq x \rightarrow' w$ holds for all $y' \in Y$ then $y \leq x \rightarrow' w$ holds and this is equivalent to $x * y \leq w$, as required.

Having established that $\mathfrak{C}$ is indeed a closure operation on $\langle P(\mathcal{V}), \cup, \cdot, \{[1]\}\rangle$ we now consider the unital quantale $\langle \mathfrak{C}(P(\mathcal{V})), \cup_{\mathfrak{C}}, \cdot_{\mathfrak{C}}, \mathfrak{C}\{[1]\}\rangle$. To enrich this to a modal unital quantale we use Lemma 2 above. Define the subsets R, S of $\mathfrak{C}(P(\mathcal{V}))$ by

$$R = \{\, \mathfrak{C}(\{[!A]\}) : A \text{ is a closed formula of } L_{IM}(U)\,\}$$

$$S = \{\, \mathfrak{C}(\{[?A]\}) : A \text{ is a closed formula of } L_{IM}(U)\,\}$$

Conditions (1) and (2) of Lemma 2 are satisfied since

$$\mathfrak{C}(\{[!A]\})\cdot_{\mathfrak{C}}\mathfrak{C}(\{!B\}) = \mathfrak{C}(\{[!(A\&B)]\})$$

and

$$\mathfrak{C}(\{[!A]\})\cdot_{\mathfrak{C}}\mathfrak{C}(\{[!A]\}) = \mathfrak{C}(\{[!A]\}).$$

Conditions (4), (5) are satisfied since $[!A] \leq \mathbf{[1]}$, $\mathbf{[0]} \leq [?A]$ holds for all closed formulae A.

Condition (6) is satisfied because $[!A \supset ?B] = [?(!A \supset ?B)]$. Finally, to verify condition (2), we have to show that for any $X \in \mathfrak{C}(P(\mathcal{V}))$,

$$X\cdot_{\mathfrak{C}}\mathfrak{C}(\{[!A]\}) = \mathfrak{C}(\{[!A]\})\cdot_{\mathfrak{C}} X$$

holds for any closed formula A. Now if $X = \mathfrak{C}(Y)$ then

$$\begin{aligned}\mathfrak{C}(Y)\cdot_{\mathfrak{C}}\mathfrak{C}(\{[!A]\}) &= \mathfrak{C}(\mathfrak{C}(Y)\cdot\mathfrak{C}(\{[!A]\}))\\ &= \mathfrak{C}(Y\cdot\{[!A]\})\end{aligned}$$

But since $[!A \bullet B] = [B \bullet !A]$ holds for any closed formula B, we have $Y\cdot\{[!A]\} = \{[!A]\}\cdot Y$, and this suffices to establish condition (2).

Our modal operations on $\mathfrak{C}(P(\mathcal{V}))$ are then defined by

$$!X = \bigcup\{Y \in R : Y \subseteq X\}$$
$$?X = \bigcap\{Y \in S : X \subseteq Y\}$$

for any $X \in \mathfrak{C}(P(\mathcal{V}))$.

This completes the construction of our modal unital quantale. The mapping $h : \mathbf{LA} \to \mathfrak{C}(P(\mathcal{V}))$ defined by $h([A]) = \mathfrak{C}\{[A]\}$ preserves all existing meets and joins in LA. Now take the assignment F given by $F(A) = h([A])$, for all closed atomic formulae A. It is readily verified that we have for the extension F, $F(A \supset B) = F(A) \to F(B)$, $F(A \supset' B)\ F(A) \to' F(B)$, $F(A \oplus B) = F(A) \cup_{\mathfrak{C}} F(B)$, $F(A \& B) = F(A) \cap F(B)$, $F(!A) = !F(A)$, $F(?A) = ?F(A)$, $F(A \bullet B) = F(A) \cdot_{\mathfrak{C}} F(B)$, for each closed formula A, B.

Now suppose that the sequent $A_1, \cdots, A_n =\vdash B$ is *not* provable, and write $\langle A_1, \cdots, A_n =\vdash B\rangle$ for the universal closure of the formula $A_1 \bullet \cdots \bullet A_n \supset B$. We show that $\langle A_1, \cdots, A_n =\vdash B\rangle$ is not true in our assignment. Suppose to the contrary that

$$1_B \leq F(\langle A_1, \cdots, A_n =\vdash B\rangle)$$

where $1_B = \mathfrak{C}(\{[1]\}) = h[\mathbf{1}]$.

Since $F(\langle A_1, \cdots, A_n =\vdash B\rangle) = h([\langle A_1, \cdots, A_n =\vdash B\rangle])$ and h is injective, we have that $[\mathbf{1}] \leq (\langle A_1, \cdots, A_n =\vdash B\rangle)$ holds in LA, but this in turn implies that $\mathbf{1} =\vdash \langle A_1, \cdots, A_n =\vdash B\rangle$ is provable, and this yields our contradiction, because $1 =\vdash \langle A_1, \cdots, A_n =\vdash B\rangle$ is provable if, and only if, $A_1, \cdots, A_n =\vdash B$ is provable, and this sequent is not provable by assumption.

1.2.2 *Kripke semantics*

In [5] we gave a Kripke style semantics for the logic L_I using partially ordered monoid structures. The system L_I did not include the connective $\oplus$ or the quantifier $\exists$, nor the modal symbols !, ? $\cdot$. To incorporate $\oplus$, $\exists$ we extend our partially ordered monoids to complete meet semi-lattice monoids (*SO*-monoids) as defined below. Let L_I^+ denote the language $L_I \cup \{\oplus, \exists\}$.

Definition. A structure $M = \langle M, \wedge, \cdot, 1, \infty\rangle$ is a complete *SO*-monoid if

(i) $\langle M, \wedge\rangle$ is a complete meet semi-lattice with greatest element ∞.

(ii) $\langle M, \cdot, 1\rangle$ is a monoid with identity 1, and for each $m \in M$ we have $m \cdot \infty = \infty \cdot m = \infty$.

(iii) $m \cdot (\wedge_i n_i) \cdot m' = \wedge_i (m \cdot n_i \cdot m')$ holds for every $m, m', n_i \in M$.

If U is a universe then a *valuation* is a mapping F from the set of closed atomic formulae of $L_I^+(U)$ to the set of principal filters in the lattice structure. The relation $\models$ is then defined by

(1) $m \models \bot$, if, and only if, $m = \infty$
(2) $m \models \mathbf{1}$, if, and only if, $m \geq 1$
(3) $m = \top$ holds for every $m \in M$
(4) $m = P$ if, and only if, $m \in F(P)$, where P is an atomic sentence of $L_I^+(U)$ excluding $\bot$, $\top$, **1** but includin g **0**.
(5) $m \models A \supset B$ if, and only if, for all $\langle m_1, m_2\rangle \in M^2$ such that $m_1 \models A$ and $m \cdot m_1 \leq m_2$ we have $m_2 \models B$
(6) $m \models A \supset' B$ if, and only if, for all $\langle m_1, m_2\rangle \in M^2$ such that $m \models A$ and $m \cdot m_1 \leq m_2$ we have $m_2 \models B$
(7) $m \models A \bullet B$ if, and only if, there exists $\langle m_1, m_2\rangle \in M^2$ with $m_1 \models A$ and $m_2 \models B$ and $m_1 \cdot m_2 \leq m$
(8) $m \models A \oplus B$ if, and only if, there exist $\langle m_1, m_2\rangle \in M^2$ such that $m_1 \wedge m_2 \leq m$ and both ($m_1 \models A$; or $m_1 \models B$) and ($m_2 \models A$ or $m_2 \models B$) hold.
(9) $m \models A \,\&\, B$ if, and only if, $m \models A$ and $m \models B$
(10) $m \models \forall x A(x)$ if, and only if, for each $u \in U$, $m \models A(\underline{u})$.
(11) $m \models \exists x A(x)$ if, and only if, there exists a subset $\{m_i\}_{i \in I}$ of M and a subset $\{u_i\}_{i \in I}$ of U such that $\wedge_{i \in I} m_i \leq m$ and $m_i \models A(\underline{u}_i)$ holds for each $i \in I$.

Ono [11] makes the observation that a complete SO-monoid is just a unital quantale with the reverse order. Thus given a unital quantale structure **B** for L_I^+ we obtain a Kripke structure **M** by reversing the order in the quantale so that $b_1 \leq_B b_2$ holds if, and only if, $b_2 \leq_M b_1$. Then if F_B is an assignment for the unital quantale **B** in which $F_B(P) = b$, where P is a closed atomic formula of $L_I^+(U)$, we have a corresponding valuation F_M in the associated Kripke structure M by defining $F_M(P) = \{m : b \leq_M m\}$, a principal filter of **M** (equivalently, a principal ideal of B). Conversely, given a Kripke model we can obtain a corresponding quantale structure and assignment. Ono [11] points out that this correspondence no longer holds if we restrict our attention to propositional logic in which case the Kripke models are SO-monoids (not necessarily complete), and an assignment is only required to associate a filter (not necessarily principal) in the SO-monoid with a closed atomic formula.

In order to extend our Kripke semantics to incorporate the modal symbols we can slightly modify the approach given in Ono [11] (for propositional logic) so it can be applied to predicate logic. Ono [11] defines conditions to be satisfied by two binary relations F, G on SO-monoids, called accessibility relations, and then extends valuations to include the modal connectives using these accessibility relations. For example, $m \models !A$ if, and only if, $n \models A$ for some n such that nFm holds. Then a soundness and completeness result is obtained for the modal propositional logic with respect to modal Kripke structures. The conditions given in [11] for the accessibility relations can easily be modified to apply to *complete SO*-monoids; basically we replace conditions involving $\wedge$ by analogous conditions using $\bigwedge$. For example, one of Ono's [11] conditions is

(6) If yFx and $y'Fx'$ hold then $(y \wedge y')F(x \wedge x')$ holds.

This is replaced by

(6') If $y_i Fx_i$ holds for each $i \in I$, then $\left(\bigwedge_{Iy_i}\right)F\left(\bigwedge_{Ix_i}\right)$ holds.

These modifications ensure that $\{m : m \models A\}$ is still a *principal* filter of M even if A contains modal connectives. The soundness of L_{IM} with respect to complete SO-monoids further equipped with our accessibility relations follows from Ono [11] since our modified conditions on F, G imply Ono's conditions. The completeness argument in Ono is also easily adapted to apply to complete SO-monoids equipped with accessibility relations.

2. *Representing negation as failure in* L_{IM}

In the declarative semantics for logic programming the negation as failure rule is sound with respect to the completed program (see e.g. Lloyd [18]). In general the completed program is an infinite collection of formulae, the axiom schema for the equality predicate is an infinite schema. However, if we have a finitely failed *SLDNF*-tree with root node $\leftarrow P$ then the negation of P is a logical consequence of a finite subset of the completed program. Similarly, if an attempt is made to construct a finitely failed *SLDNF*-tree for $\leftarrow P$ but the attempt fails due to the tree containing a success branch, then we can show that P is a logical consequence of a finite subset of the completed program.

In section 2.1 we define the completion of a normal program. In section 2.2 we address the construction of an *SLDNF*-tree in our logic. If $\mathcal{P}$ is a normal program and $\mathcal{T}$ is an *SLDNF*-tree for $\mathcal{P} \cup \{\leftarrow L_1, \cdots, L_n\}$, then we specify how a finite set Γ of formulae may be associated with $\mathcal{T}$ such that

if $\mathcal{T}$ is a finitely failed tree then $\Gamma \vdash \neg\exists(L_1 \bullet \cdots \bullet L_n)$ holds, and on the other hand, if $\mathcal{T}$ contains a success branch then $\Gamma \vdash \neg\forall(L_1 \bullet \cdots \bullet L_n)\Theta$ where Θ is the computed answer substitution associated with the success branch. The formulae of Γ are instances of the completed program scheme.

Once Γ has been defined for a given tree $\mathcal{T}$, it follows from the result in section 1.1 that given $\mathcal{T}$ we can associate a formula $!T_{\mathcal{T}}$ (where T is the conjunction of the formulae in Γ) such that either $!T_{\mathcal{T}} = \vdash \neg\exists(L_1 \bullet \cdots \bullet L_n)$ or $!T_{\mathcal{T}} = \vdash \forall(L_1 \bullet \cdots \bullet L_n\Theta)$ is a provable sequent in our logic. The formula $!T_{\mathcal{T}}$ is the representation of the construction of the tree $\mathcal{T}$ in L_{IM}.

2.1 *The completion of a normal program*

Let $\mathcal{P}$ be a normal program (see Lloyd [18]) and suppose that the clauses in $\mathcal{P}$ with head predicate letter Q form the non-empty set

$$\{Q(s_i) \leftarrow \mathbf{L_i} : i = 1, \cdots, m\}$$

where s_i stands for $s_{i,1}, \cdots, s_{i,n}$ and $\mathbf{L_i}$ stands for $L_{i,1}, \cdots, L_{i,m_i}$, where each $L_{i,j}$ is a literal. In the translation into formulae of L_{IM} a positive literal remains unchanged, but a negative literal, say $\sim P$, will translate to $!\neg P$. The translation of a program literal L_i will be written as $\hat{L}_i$. The process of obtaining the completed definition of the predicate letter Q with respect to the program $\mathcal{P}$ is given below. A program clause $Q(s_i) \leftarrow \mathbf{L_i}$ is first transformed to the L_{IM} formula

$$\left(\left((x_1 = s_{i,1}) \& \cdots \& (x_n = s_{i,n})\right) \bullet \hat{L}_{i,1} \bullet \cdots \bullet \hat{L}_{i,m_i}\right) \supset Q(x_1, \cdots, x_n)$$

where $x_1, \cdots, x_n$ are new variables not appearing in $\mathcal{P}$, and = is a predicate which does not occur in $\mathcal{P}$. We abbreviate this formula to

$$(\mathbf{x} = \mathbf{s_i}) \bullet \hat{\mathbf{L}}_\mathbf{i} \supset Q(x)$$

If $y_{i,1}, \cdots, y_{i,r_i}$ are the variables which occur in the program clause, then we now form the formula

$$\exists y_{i,1}, \cdots, \exists y_{i,r_i}\left((\mathbf{x} = \mathbf{s_i}) \bullet \hat{\mathbf{L}}_\mathbf{i}\right) \supset Q(x)$$

which we abbreviate to $E_i \supset Q(x)$.

We thus have formulae $E_i \supset Q(x)$ for $i = 1, \cdots, m$, and we now define the completed definition of the predicate letter Q (with respect to $\mathcal{P}$) as the sentence

$$\forall x\Big(\big((E_1 \oplus \cdots \oplus E_m) \supset Q(x)\big) \& \big(Q(x) \supset (E_1 \oplus \cdots \oplus E_m)\big)\Big).$$

For the case $m = 0$ where $\mathcal{P}$ does not contain any clause with head predicate letter Q, we define the completed definition of Q by the sentence $\forall x\big(Q(x) \supset \bot\big)$. We write $\mathcal{P}$-comp(Q) for the completed definition of Q with respect to $\mathcal{P}$. The predicate = is required to satisfy the following translation of Clark's [12] equality theory.

2.1.1 *Equality axiom schema*

The axioms for equality are given by (1) - (9) below.

(1) $\forall x(x = x)$

(2) $\forall x \forall y\big((x = y) \supset (y = x)\big)$

(3) $\forall x \forall y \forall z\big((x = y) \& (y = z) \supset (x = z)\big)$

(4) $\forall x \forall y\Big(\hat{L}(x) \bullet (x = y) \bullet \hat{M}(x) \supset \hat{L}(y) \bullet \hat{M}(y)\Big)$, where $\hat{L}$ stands for a co-joining $\hat{L}_1 \bullet \cdots \bullet \hat{L}_m$ of literals, and likewise $\hat{M}$.

(5) $\forall x \forall y\big((x = y) \& t_1(x) = t_2(x) \supset t_1(y) = t_2(y)\big)$ where t_1 and t_2 are terms

(6) $\forall x \forall y\big((x = y) \supset f(x) = f(y)\big)$ for each function symbol f

(7) $\forall x \forall y\big(f(x) = g(y) \supset \bot\big)$ for each pair of distinct function symbols f, g (including constants)

(8) $\forall x \forall y\big(f(x) = f(y) \supset (x = y)\big)$

(9) $\forall x\big(t(x_1) = x_1 \supset \bot\big)$ where $t(x_1)$ is any term containing x_1, which is distinct from x_1.

The axioms of (4) - (9) are schemata so our list of axioms is an infinite list (assuming that we have at least one function symbol, or an infinite set of constant symbols). Now let $P(s)$, $P(t)$ denote atoms with the same predicate letter. If these atoms are unifiable (see Lloyd [18]) then the unification algorithm generates a most general unifier (mgu); if the atoms are not unifiable then the algorithm reports this fact. We assume a fixed deterministic instance of the unification algorithm. If we let $S_0 = \{P(s), P(t)\}$ then

we denote by $(5)_{s0}$ the finite set of instances of schema (5) obtained by restricting t_1, t_2 to terms generated by the algorithm applied to S_0. Similarly for $(9)_{s0}$. Also, $(6)_{s0}$ denotes the finite number of instances of (6) obtained by restricting the function symbols to the function symbols occurring in S_0. Similarly for $(7)_{s0}$, $(8)_{s0}$. Finally denote by $\{\text{EQUALS}_{s0}\}$ the union of $(5)_{s0}$, $(6)_{s0}$, $(7)_{s0}$, $(8)_{s0}$, $(9)_{s0}$ and $\{(1), (2), (3)\}$. The schema (4) plays no part in the following lemma. The formula $A \leftrightarrow B$ stands for $(A \supset B)\&(B \supset A)$.

Lemma 1.
(i) If $P(s)$ and $P(t)$ are unifiable with mgu $\Theta = \{v_1/r_1, \cdots, v_k/r_k\}$ then

$$\{\text{EQUALS}_{s0}\} \vdash \forall\big((s_1 = t_1)\& \cdots \&(s_n = t_n) \leftrightarrow (v_1 = r_1)\& \cdots \&(v_k = r_k)\big)$$

(ii) If $P(s)$ and $P(t)$ are not unifiable then}

$$\{\text{EQUALS}_{s0}\} \vdash \forall\big((s_1 = t_1)\& \cdots \&(s_n = t_n) \supset \bot\big).$$

Proof. Both parts are proved by induction on the number of steps required by the algorithm.

2.2 *Negation as failure*

In the following two lemmas we establish syntactic analogues of results given in Lloyd [8] pertaining to the soundness of negation as failure with respect to models of the completed program. We begin with some terminology.

Let $\mathcal{P}$ be a normal program and let G_0 be a normal goal $\leftarrow L_1, \cdots, M(u), \cdots, L_q$ in which the computation rule selects the positive literal $M(u)$. We assume that $\mathcal{P}$ contains n-clauses with head predicate letter M, say $\{M(t_i) \leftarrow A_{i,1}, \cdots, A_{i,r_i} : i = 1, \cdots n\}$, and without loss of generality we may assume that $M(u)$ unifies with the first m of these clauses. The m resultant goals are $\{G_1, \cdots, G_m\}$ where G_i is the goal $\leftarrow (L_i, \cdots, A_{i,1}, \cdots, A_{i,r_i}, \cdots, L_q)\Theta_i$, and Θ_i is at most a general unifier of $S_i = \{M(u), M(t_i)\}$. We will write G_j for the sentence $\neg\exists\big(\hat{L}_1 \bullet \cdots \bullet \hat{A}_{j,1} \bullet \cdots \bullet \hat{A}_{j,r_i} \bullet \cdots \bullet \hat{L}_q\big)$ corresponding to the goal G_j.

Lemma 2. If $\mathcal{P}$ is a normal program and the computation rule selects the positive literal $M(u)$ from the normal goal G_0 then

(i) if there are no derived goals

$$\{\mathcal{P} - \text{comp}(M)\} \cup \{\text{EQUALS}_{s_i}\} \cup \cdots \cup \{\text{EQUALS}_{s_n}\} \vdash G_o;$$

(ii) If the derived goals form the non-empty set $\{G_1, \cdots, G_m\}$ then

$$\{\mathcal{P} - \text{comp}(M_j)\} \cup \{\text{EQUALS}_{s_1}\} \cup \cdots \cup \{\text{EQUALS}_{s_n}\} \cup$$
$$\{EQ4_{G_1}, \cdots, EQ4_{G_m}\} \vdash (G_1 \& \cdots \& G_m) \supset G_0,$$

where $EQ4_{G_j}$ stand for instances of the equality axiom (4), defined in the proof below.

Proof.
(i) Suppose that the completed definition of M is $\forall(\hat{M}(x) \supset \bot)$. The sequent $\forall(\hat{M}(x) \supset \bot) = \vdash \neg\exists(L_1 \bullet \cdots \bullet \hat{M}(u) \bullet \cdots \bullet \hat{L}_q)$ is provable and hence we have

$$\{\mathcal{P} - \text{comp}(M)\} \vdash \mathbf{G_o}.$$

The other case for no derived goals is when the completed definition of M is

$$\forall(\hat{M}(x) \leftrightarrow E_1 \oplus \cdots \oplus E_n)$$

and $\{M(u), M(t_i)\}$ is not unifiable for each $i = 1, \cdots, n$. The formula E_i can be written as

$$\exists y_i((x = t_i) \bullet \hat{A}_i)$$

where $\hat{A}_i$ stands for $\hat{A}_{i,1} \bullet \cdots \bullet \hat{A}_{i,r_i}$. We have

$$\{\mathcal{P} - \text{comp}(M)\} \vdash \hat{M}(u) \leftrightarrow \bigoplus_{i=1}^{n} E_i' \tag{1}$$

where E_i' is obtained from E_i by replacing x by u. Now from Lemma 1(ii) we have

$$\{\text{EQUALS}_{s_i}\} \vdash \forall((u = t_i) \supset \bot) \tag{2}$$

for each $i = 1, \cdots, n$. The sequent

$$\forall\big((u=t_i)\supset\perp\big)\vdash\neg E_i' \tag{3}$$

is provable, and (1), (2), (3) give us

$$\{\mathcal{P}-\text{comp}(M)\}\cup\{\text{EQUALS}_{s_1}\}\cup\cdots\cup\{\text{EQUALS}_{s_n}\}\vdash\neg\hat{M}(u) \tag{4}$$

and from (4) we can infer

$$\{\mathcal{P}-\text{comp}(M)\}\cup\{\text{EQUALS}_{s_1}\}\cup\cdots\cup\{\text{EQUALS}_{s_n}\}\vdash\mathbf{G_0}.$$

(ii) The distributivity of $\bullet$ over $\oplus$ and the completed definition of M gives

$$\{\mathcal{P}-\text{comp}(M)\}\vdash\big(\hat{L}_1\bullet\cdots\bullet\hat{M}(u)\bullet\cdots\bullet\hat{L}_q\big)\leftrightarrow\bigoplus_{i=1}^{n}\big(\hat{L}_1\bullet\cdots\bullet E_i'\bullet\cdots\hat{L}_q\big) \tag{5}$$

Since $\{M(u), M(t_i)\}$ do not unify for $i=m+1,\cdots,n$ we have from Lemma 1(ii)

$$\{\text{EQUALS}_{s_j}\}\vdash\forall\big((u=t_j)\supset\perp\big),\ \text{for}\ j=m+1,\cdots,n,$$

and from this we can establish

$$\{\text{EQUALS}_{s_j}\}\vdash\big(\hat{L}_1\bullet\cdots\bullet\hat{M}(u)\bullet\cdots\bullet\hat{L}_q\big)\supset\perp \tag{6}$$

for $j=m+1,\cdots,n$. The constant $\perp$ is neutral for $\oplus$ so we now have from (5), (6)

$$\begin{aligned}&\{\mathcal{P}-\text{comp}(M)\}\cup\{\text{EQUALS}_{s_{m+1}}\}\cup\cdots\cup\{\text{EQUALS}_{s_n}\}\vdash\\&\big(\hat{L}_1\bullet\cdots\bullet\hat{M}(u)\bullet\cdots\bullet\hat{L}_q\big)\leftrightarrow\bigoplus_{i=1}^{n}\big(\hat{L}_1\bullet\cdots\bullet E_i'\bullet\cdots\hat{L}_q\big)\end{aligned} \tag{7}$$

The formula E_i' is $\exists y_i\big((u=t_i)\bullet\hat{A}_i\big)$, and by standardizing apart the variables in y_i do not appear in $\hat{L}_1\bullet\cdots\bullet\hat{M}(u)\bullet\cdots\bullet\hat{L}_q$, so that we can move the quantifiers outwards in (7) to give

$$\{\mathcal{P} - \text{comp}(M)\} \cup \{\text{EQUALS}_{s_{m+1}}\} \cup \cdots \cup \{\text{EQUALS}_{s_n}\} \vdash$$
$$\left(\hat{L}_1 \bullet \cdots \bullet \hat{M}(u) \bullet \cdots \bullet \hat{L}_q\right) \leftrightarrow \overset{n}{\underset{i=1}{\oplus}} \exists y_i \left(\hat{L}_1 \bullet \cdots \bullet (u = t_i) \bullet \cdots \bullet \hat{L}_q\right) \tag{8}$$

This last step is justified because if x is not free in the formulae B, C then $\exists x(B \bullet D(x) \bullet C) \leftrightarrow B \bullet \exists x D(x) \bullet C$ is a theorem.

Lemma 1(i) gives for $i = 1, \cdots, m$

$$\{\text{EQUALS}_{s_j}\} \vdash (u = t_i) \leftrightarrow (v_i = r_i)$$

where $\Theta_i = \{v_{i,1}/r_{i,1}, \cdots v_{i,x_i}/r_{i,k_i}\}$ is a *mgu* of S_i This result together with (8) give

$$\{\mathcal{P} - \text{comp}(M)\} \cup \{\text{EQUALS}_{s_1}\} \cup \cdots \cup \{\text{EQUALS}_{s_n}\} \vdash$$
$$\left(\hat{L}_1 \bullet \cdots \bullet \hat{M}(u) \bullet \cdots \bullet \hat{L}_q\right) \leftrightarrow \overset{n}{\underset{i=1}{\oplus}} \exists y_i \left(\hat{L}_1 \bullet \cdots \bullet (v_i = r_i) \bullet \hat{A}_i \bullet \cdots \bullet \hat{L}_q\right) \tag{9}$$

To complete the proof we now use the equality axiom (4) to give

$$\{\text{EQ4G}_i\} \vdash \hat{L}_1 \bullet \cdots \bullet (v_i = r_i) \bullet \hat{A}_i \bullet \cdots \bullet \hat{L}_q \supset \left(\hat{L}_1 \bullet \cdots \bullet \hat{A}_i \bullet \cdots \bullet \hat{L}_q\right)\Theta_i \tag{10}$$

where EQ4G_i stands for the following particular instance of the equality schema (4)

$$\forall\left(\hat{L}_1 \bullet \cdots \bullet (v_i = r_i) \bullet \hat{A}_i \bullet \cdots \bullet \hat{L}_q\right) \supset \left(\hat{L}_1 \bullet \cdots \bullet \hat{A}_i \bullet \cdots \bullet \hat{L}_q\right)\Theta_i$$

The results in (10) and (9) allow us to infer

$$\{\mathcal{P} - \text{comp}(M)\} \cup \{\text{EQUALS}_{s_1}\} \cup \cdots \cup \{\text{EQUALS}_{s_n}\} \cup \{\text{EQ4G}_i : i = 1, \cdots, m\} \vdash \left(\hat{L}_1 \bullet \cdots \bullet \hat{M}(u) \bullet \cdots \bullet \hat{L}_q\right) \supset \overset{n}{\underset{i=1}{\oplus}} \exists y_i \left(\hat{L}_1 \bullet \cdots \bullet \hat{A}_i \bullet \cdots \bullet \hat{L}_q\right)\Theta_i \tag{11}$$

and our result follows from (11) by observing that the sequent

$$B \supset \overset{m}{\underset{i=1}{\oplus}} C_i = \vdash \&_{i=1}^{m} (\neg \exists C_i) \supset \neg(\exists B)$$

is a theorem of our logic.

The following proposition will be used in Lemma 3.

Proposition.1 Let G_i be a goal clause $\leftarrow L_1, \cdots L_{i-1}, P(s), L_{i+1}, \cdots, L_r$ and suppose that the positive literal $P(s)$ is selected. If $\mathcal{P}$ is a program containing a clause $P(t) \leftarrow A_1, \cdots, A_k$ whose head unifies with $P(s)$ with *mgu* Θ, yielding the derived goal $G_{i+1}, \leftarrow L_1\Theta, \cdots, L_{i-1}\Theta, A_1\Theta, \cdots, A_k\Theta, L_{i+1}\Theta, \cdots, L_r\Theta$, then we have

$$\mathcal{P} - \text{comp}(P) \vdash \left(!\hat{L}_1\Theta \bullet \cdots \bullet !\hat{L}_{i-1}\Theta \bullet !\hat{A}_1\Theta \bullet \cdots \bullet !\hat{A}_k\Theta \bullet !\hat{L}_{i+1}\Theta \bullet \cdots !\hat{L}_r\Theta\right)$$
$$\supset \left(!\hat{L}_1\Theta \bullet \cdots \bullet !P(s)\Theta \bullet \cdots \bullet !\hat{L}_r\Theta\right)$$

Proof. The result follows easily from the fact that $\mathcal{P} - \text{comp}(P) \vdash !\left(\hat{A}_1\Theta \bullet \cdots \bullet \hat{A}_k\Theta \supset \hat{P}(t)\Theta\right)$. □

Lemma 3 below establishes a soundness result for negation as failure with respect to models of the completed program. Our proof shows that if $\mathcal{P} \cup \{\leftarrow L_1, \cdots, L_n\}$ has a finitely failed *SLDNF*-tree, $\mathcal{T}$ say, then the sentence $\neg\exists\left(\hat{L}_1 \bullet \cdots \bullet L_n\right)$ is a logical consequence of a (finite) subset of the completed program, this subset being determined by $\mathcal{T}$. Let $\mathcal{P}$-comp (PRED($\mathcal{T}$)) be the set of completed predicate definitions for each predicate which occurs in a goal clause in $\mathcal{T}$, including any subsidiary trees to the main tree. (Actually, we can restrict ourselves to the predicate letters P such that the computation rule selects an atom $P(t)$ or a negative literal $\sim P(t)$ during the construction of $\mathcal{T}$.) Similarly we define the instances of the equality axioms associated with $\mathcal{T}$, $\mathcal{P}$-comp(EQUALS($\mathcal{T}$)), as follows.

(1) If $\mathcal{T}$ contains a goal in which a positive literal is selected and the immediate derived goals form the non-empty set $\{G_1, \cdots, G_m\}$, then (using the terminology of Lemma 2), each member of the union of the sets $\{\text{EQUALS}_{s_1}\}, \cdots, \{\text{EQUALS}_{s_n}\}, \{\text{EQ4}_{G_1}, \cdots, \text{EQ4}_{G_m}\}$ belongs to $\mathcal{P}$-comp(EQUALS($\mathcal{T}$)).
(2) If $\mathcal{T}$ contains a goal in which a positive literal is selected and there are no derived goals then each member of the union of the sets $\{\text{EQUALS}_{s_1}\}, \cdots, \{\text{EQUALS}_{s_n}\}$ belongs to $\mathcal{P}$-comp(EQUALS($\mathcal{T}$)). (Again, see Lemma 2 for the terminology.)
(3) The only members of $\mathcal{P}$-comp(EQUALS($\mathcal{T}$)) are given by (1) and (2) above.

The following notation will be used in Lemma 3. If G is the clause $\leftarrow L_1, \cdots, L_n$ then we write $\underline{G}$ for the formula $\hat{L}_1 \bullet \cdots \bullet \hat{L}_n$, and $!\underline{G}$ for the

formula $!\hat{L}_1 \bullet \cdots \bullet !\hat{L}_n$. Note that $\underline{G}$ is not the same as the sentence G defined at the beginning of this section.

Lemma 3. Let $\mathcal{P}$ be a normal program and suppose that $\mathcal{P} \cup \{\leftarrow L_1, \cdots, L_n\}$ has a finitely failed *SLDNF*-tree, $\mathcal{T}$. Then we have

$$\mathcal{P} - \text{comp}(\text{PRED}(\mathcal{T})) \cup \mathcal{P} - \text{comp}(\text{EQUALS}(\mathcal{T})) \vdash \neg\exists(\hat{L}_1 \bullet \cdots \bullet !\hat{L}_n)$$

Proof. We proceed by induction on the number N of negative sub-goals selected in the construction of $\mathcal{T}$, including any negative sub-goals selected in subsidiary trees to the main tree.

(i) The base case $N = 0$ is handled by an inductive argument on the depth of the tree $\mathcal{T}$, using Lemma 2.

(ii) For the inductive step suppose that $N \neq 0$ so that $\mathcal{T}$ contains at least one goal clause (in the main tree) from which a negative literal is selected. Let the goal $\leftarrow Q_1, \cdots, Q_{i-1}, \sim P, Q_{i+1}, \cdots, Q_r$ be such a goal of least depth d in $\mathcal{T}$. It suffices to show that for such a goal we have

$$\mathcal{P} - \text{comp}(\text{PRED}(\mathcal{T})) \cup \mathcal{P} - \text{comp}(\text{EQUALS}(\mathcal{T})) \vdash \neg\exists (\hat{Q}_1 \bullet \cdots \bullet \hat{Q}_{i-1} \bullet \neg P \bullet \hat{Q}_{i+1} \bullet \cdots \bullet \hat{Q}_r) \tag{1}$$

since the induction hypothesis applies to other goals of depth d in $\mathcal{T}$ in which positive literals are selected, and then Lemma 2 can be applied to complete the argument.

To establish (1) we first consider the case where the negative ground literal $\sim P$ is selected and the sub-goal $\leftarrow\sim$ P succeeds yielding the derived goal $\leftarrow Q_1, \cdots, Q_{i-1}, Q_{i+1}, \cdots, Q_r$ The induction hypothesis applied to this derived goal gives

$$\mathcal{P} - \text{comp}(\text{PRED}(\mathcal{T}_1)) \cup \mathcal{P} - \text{comp}(\text{EQUALS}(\mathcal{T}_1)) \vdash \neg\exists (\hat{Q}_1 \bullet \cdots \bullet \hat{Q}_{i-1} \bullet \hat{Q}_{i+1} \bullet \cdots \bullet \hat{Q}_r) \tag{2}$$

where $\mathcal{T}_1$ is the subtree of $\mathcal{T}$ with root node $\leftarrow Q_1, \cdots, Q_{i-1}, Q_{i+1}, \cdots, Q_r$. Since $\mathcal{T}_1$ is a sub-tree of $\mathcal{T}$ we have $\mathcal{P} - \text{comp}(\text{PRED}(\mathcal{T}_1)) \subseteq \mathcal{P} - \text{comp}(\text{PRED}(\mathcal{T}))$ and similarly for $\mathcal{P} - \text{comp}(\text{EQUALS}(\mathcal{T}_1))$, so we

can replace $\mathcal{T}_1$ by $\mathcal{T}$ in (2). Now the sequent $\exists\left(\hat{Q}_1 \bullet \cdots \bullet !\sim P \bullet \cdots \bullet \hat{Q}_r\right) =$ $\vdash \exists\left(\hat{Q}_1 \bullet \cdots \bullet \hat{Q}_{i-1} \bullet \hat{Q}_{i+1} \bullet \cdots \bullet \hat{Q}_r\right)$ is provable, and from (2) we have

$$\mathcal{P} - \text{comp}(\text{PRED}(\mathcal{T})) \cup \mathcal{P} - \text{comp}(\text{EQUALS}(\mathcal{T})) \vdash \neg\exists \left(\hat{Q}_1 \bullet \cdots \bullet !\neg P \bullet \cdots \bullet \hat{Q}_r\right)$$

as required.

The other case is when the sub-goal $\leftarrow\sim P$ fails so that $\mathcal{T}$ contains a subsidiary tree with root note $\leftarrow P$, which has a success branch. Let $\mathbf{G}_i$ and $\mathbf{G}_{i+1}$ be successive goals along this success branch such that a positive literal in G_i is selected by the computation rule. By the proposition following Lemma 2
we have

$$\mathcal{P} - \text{comp}(\text{PRED}(\mathcal{T})) \cup \mathcal{P} - \text{comp}(\text{EQUALS}(\mathcal{T})) \vdash !\underline{G}_{i+1} \supset !\underline{G}_i\Theta \quad (3)$$

where Θ is the *mgu* associated with the resolution step. If $\mathbf{G}_{i+1}$ is derived from $\mathbf{G}_i$ by the selection of a negative ground literal which succeeds, say $\sim R$, then from the induction hypothesis on N we infer

$$\mathcal{P} - \text{comp}(\text{PRED}(\mathcal{T})) \cup \mathcal{P} - \text{comp}(\text{EQUALS}(\mathcal{T})) \vdash !\neg L \quad (4)$$

since if $\Gamma \vdash A$, then clearly $\Gamma \vdash !A$ by an application of (!-R). Corresponding to (3) above we have

$$\mathcal{P} - \text{comp}(\text{PRED}(\mathcal{T})) \cup \mathcal{P} - \text{comp}(\text{EQUALS}(\mathcal{T})) \vdash !\underline{G}_{i+1} \supset !\underline{G}_i \quad (5)$$

Now an application of (3) and (5) along the success branch for P yields

$$\mathcal{P} - \text{comp}(\text{PRED}(\mathcal{T})) \cup \mathcal{P} - \text{comp}(\text{EQUALS}(\mathcal{T})) \vdash !P \quad (6)$$

(Since P is ground, the application of the unifier sequence to P has no effect.) Our result (1) now follows readily from (6), and this completes the argument for the inductive step. □

For *SLDNF*-refutations suppose that $\mathcal{P} \cup \{\leftarrow L_1, \cdots, L_n\}$ has an *SLDNF*-tree $\mathcal{T}$ which contains a success branch β. (The tree may be infinite.) Denote by $\mathcal{T}_\beta$ the partial *SLDNF*-tree obtained from $\mathcal{T}$ by taking the suc-

cess branch together with any subsidiary trees associated with negative subgoals along the branch. Then from the argument in Lemma 3 we have the following corollary.

Corollary. If the *SLDNF*-tree for $\mathcal{P} \cup \{\leftarrow L_1, \cdots, L_n\}$ contains a success branch β with associated unifier sequence $\Theta_1\Theta_2\cdots\Theta_m$, then

$$\mathcal{P} - \text{comp}\big(\text{PRED}(\mathcal{T}_\beta)\big) \cup \mathcal{P} - \text{comp}\big(\text{EQUALS}(\mathcal{T}_\beta)\big) \vdash \forall\Big(\big(\hat{L}_1 \bullet \cdots \bullet \hat{L}_n\big)\Theta_1\cdots\Theta_m\Big).$$

We can now specify a formula $!T_{\mathcal{T}}$ corresponding to a finite possibly partial) *SLDNF*-tree $\mathcal{T}$. Let $\mathcal{P}$ be a normal program and let $\leftarrow L_1, \cdots, L_n$ be a normal goal such that either

(i) $\mathcal{P} \cup \{\leftarrow L_1, \cdots, L_n\}$ has a finitely failed *SLDNF*-tree $\mathcal{T}$,

or

(ii) $\mathcal{P} \cup \{\leftarrow L_1, \cdots, L_n\}$ has an *SLDNF*-tree $\mathcal{T}$ with a success branch β and associated unifier sequence $\Theta_1\cdots\Theta_m$.

Then if case (i) holds, the formula $T_{\mathcal{T}}$ is the conjunction (&) of all formulae in $\mathcal{P} - \text{comp}\big(\text{PRED}(\mathcal{T})\big) \cup \mathcal{P} - \text{comp}\big(\text{EQUALS}(\mathcal{T})\big)$. In case (ii) $T_{\mathcal{T}}$ is the corresponding conjunction using $\mathcal{T}_\beta$ instead of $\mathcal{T}$. Then Lemma 3 and its corollary give us that in case (i) the sequent $!T_{\mathcal{T}} = \vdash \neg\exists\big(\hat{L}_1 \bullet \cdots \bullet \hat{L}_n\big)$ is provable, and in case (ii) the sequent $!T_{\mathcal{T}} = \vdash \forall\Big(\big(\hat{L}_1 \bullet \cdots \bullet \hat{L}_n\big)\Theta_1\cdots\Theta_m\Big)$ is provable.

2.3 *Representing SLDNF-derivations in* L_{IM}

In [5] we represented an *SLD*-resolution step from goal $\mathbf{G_0}$ to derived goal $\mathbf{G_1}$ (via application of the computation rule to select an atom in $\mathbf{G_0}$) by a provable sequent

$$G_0, \text{COMP}_1, H_1, R_1 = \vdash G_1 \qquad \text{(i)}$$

where if G_0 is $\leftarrow A_1, \cdots, A_m, \cdots, A_k$ and A_m is selected by the computation rule, the sentence COMP_1 was defined as $\forall(A_m \bullet A_1 \bullet \cdots \bullet A_{m-1} \bullet A_{m+1} \bullet \cdots \bullet A_k \supset A_1 \bullet \cdots \bullet A_m \bullet \cdots \bullet A_k)$ The sentence H_1 represents the selected program input clause $C \leftarrow B_1, \cdots, B_r$ by $\forall(B_1 \bullet \cdots \bullet B_r \supset C)$, and if A_m and C unify with *mgu* Θ yielding the derived goal $\leftarrow (A_1, \cdots, A_{m-1}, B_1, \cdots B_r, A_{m+1}, \cdots, A_k)\Theta$ then R_1 (the sentence representing the ordering of atoms in the derived goal) is defined as $\forall(A_1 \bullet \cdots \bullet A_{m-1} \bullet B_1 \bullet \cdots \bullet B_r \bullet A_{m+1} \bullet \cdots \bullet A_k)\Theta \supset (B_1 \bullet \cdots \bullet B_r \bullet A_1 \bullet \cdots \bullet A_{m-1} \bullet A_{m+1} \bullet \cdots \bullet A_k)\Theta)$. Now consider

an *SLDNF*-derivation step in which the negative ground literal $\sim P$ is selected from the goal $\leftarrow Q_1, \cdots, Q_{i-1}, \sim P, Q_{i+1}, \cdots, Q_k$, and $\leftarrow\sim P$ succeeds yielding the derived goal $\leftarrow Q_1, \cdots, Q_{i-1}, Q_{i+1}, \cdots, Q+k$. Let $\mathcal{T}$ be the finitely failed *SLDNF*-tree for $\leftarrow P$. This step is represented by the provable sequent

$$G_0, \mathrm{COMP}_1, !T_{\mathcal{T}} = \vdash G_1, \tag{ii}$$

where COMP_1 is the sentence (which in this case is a theorem of L_{IM}) $\forall(!\neg P \bullet Q_1 \bullet \cdots \bullet Q_{i-1} \bullet Q_{i+1} \bullet \cdots \bullet Q_k \supset Q_1 \bullet \cdots \bullet Q_{i-1} \bullet !\neg P \bullet Q_{i+1} \bullet \cdots \bullet Q_k)$. The sentence $!T_{\mathcal{T}}$ is as defined at the end of the previous section and represents the tree $\mathcal{T}$.

We can now specify an injective map from the set of all finite *SLDNF*-derivations into provable sequents of L_{IM}, in a similar manner to the map from *SLD*-derivations into provable sequents of L_I, given in [5].

Consider first an *SLDNF*-refutation of $\mathcal{P} \cup \{\leftarrow L_1, \cdots, L_n\}$ with associated answer substitution Θ. If the penultimate goal is an atom P then we have the associated provable sequent of the form

$$\mathrm{COMP}_1, \cdots, \mathrm{COMP}_r, H_r, R_r, \cdots, \mathrm{COMP}_m, !T_{\mathcal{T}}, \cdots, \mathrm{COMP}_{i-1}, H_i = \vdash \forall(L_1 \bullet \cdots \bullet L_n)\Theta \tag{i}$$

where the sub-sequence $\mathrm{COMP}_r, H_r, R_r$ corresponds to a resolution step in which an atom is selected by the computation rule, and the sub-sequence $\mathrm{COMP}_m, !T_{\mathcal{T}}$ corresponds to the selection of a negative ground literal and the construction of the associated *SLDNF*-tree, $\mathcal{T}$. H_i represents the final (unit) program clause selected in the derivation. On the other hand, if the penultimate goal is a negative literal the associated sequent is

$$\mathrm{COMP}_1, \cdots, \mathrm{COMP}_r, H_r, R_r, \cdots, \mathrm{COMP}_m, !T_{\mathcal{T}}, \cdots, \mathrm{COMP}_{i-1_j} !T_{\mathcal{T}'} = \vdash \forall(L_1 \bullet \cdots \bullet L_n)\Theta \tag{ii}$$

where $\mathcal{T}'$ is the tree associated with the negative literal. For the case of finitely failed *SLDNF*-derivations suppose the derivation terminates with final goal $\leftarrow Q_1, \cdots, Q_r$ and let Θ denote the composition of the associated unifier sequence. Now if the final goal fails because the computation rule selects a positive literal Q_j which fails then (similarly to the corresponding case in [5]), we may associate with this derivation the provable sequent

$$\mathrm{COMP}_1, \cdots, \mathrm{COMP}_r, H_r, R_r, \cdots, \mathrm{COMP}_m, !T_{\mathcal{T}}, \cdots, \mathrm{COMP}_i = \vdash \forall(Q_j \bullet Q_1 \bullet \cdots \bullet Q_{j-1} \bullet Q_{j+1} \bullet \cdots \bullet Q_r) \supset \forall(L_1 \bullet \cdots \bullet L_n)\Theta \tag{iii}$$

But this does not capture the final step in which after the computation rule selects Q_j, a (failed) attempt is made to unify Q_j with the head of a program clause. This means that $\mathcal{P} \cup \{\leftarrow Q_j\}$ has a finitely failed *SLDNF*-tree of depth zero, and if we denote this tree by $\mathcal{T}$, we choose to replace (iii) above by the following provable sequent

$$\begin{aligned} &\mathrm{COMP}_1, \cdots, \mathrm{COMP}_r, H_r, R_r, \cdots, \mathrm{COMP}_m, !T_{\mathcal{T}}, \cdots, \mathrm{COMP}_i, !T_{\mathcal{T}'} = \\ &\vdash \forall\big((Q_j \bullet Q_1 \bullet \cdots \bullet Q_{j-1} \bullet Q_{j+1} \bullet \cdots \bullet Q_r) \supset \\ &\forall(L_1 \bullet \cdots \bullet L_n\Theta)\big) \bullet !\neg\exists\big(Q_j \bullet Q_1 \bullet \cdots \bullet Q_{j-1} \bullet Q_{j+1} \bullet \cdots \bullet Q_r\big) \end{aligned} \tag{iv}$$

The remaining case is that of a failed derivation in which the final goal is $\leftarrow Q_1, \cdots, Q_{j-1}, \sim P, Q_{j+1}, \cdots, Q_r$ and the negative ground literal $\sim P$ is selected and fails. The following provable sequent corresponds to this case, where $\mathcal{T}$ is the final tree constructed (containing the success branch for P).

$$\begin{aligned} &\mathrm{COMP}_1, \cdots, \mathrm{COMP}_r, H_r, R_r, \cdots, \mathrm{COMP}_m, !T_{\mathcal{T}}, \cdots, \mathrm{COMP}_i, !T_{\mathcal{T}'} = \\ &\vdash \big(\forall\big(!\neg P \bullet Q_1 \bullet \cdots \bullet Q_{j-1} \bullet Q_{j+1} \bullet \cdots \bullet Q_r\big) \supset \\ &\forall(L_1 \bullet \cdots \bullet L_n)\big)\Theta \bullet !\neg\exists\big(!\neg P \bullet Q_1 \bullet \cdots \bullet Q_{j-1} \bullet Q_{j+1} \bullet \cdots \bullet Q_r\big). \end{aligned}$$

Monash University

REFERENCES

[1] Abrusci, V.M., Phase Semantics and Sequent Calculus for Pure Noncommutative Classical Linear Propositional Logic, *Journal of Symbolic Logic* 56 : 1403-1451, 1991.

[2] Clark, K.L., Negation as Failure, in *Logic and Databases*: 293-322, Gallaire, H. and Minker, J. editors, Plenum Press, New York, 1978.

[3] Girard, J-Y., Linear Logic, *Theoretical Computer Science* 50 : 1-102, 1987.

[4] Grätzer, G., *General Lattice Theory*, Academic Press, 1978.

[5] Jeavons, J.S. and Crossley, J.N., A Logic-based Modelling of Prolog Resolution Sequences, *Logique et Analyse*, 137-138: pp.189-205, 1992 (appeared 1996)

[6] Komori, Y., Predicate Logics Without the Structure Rules, *Studia Logica* 45 : 393-404, 1985.

[7] Lambek, J., The mathematics of sentence structure, *Am. Math.Monthly* 65, 1958.

[8] Lloyd, J.W., Foundations of Logic Programming, Second Edition, Springer-Verlag, Heidelberg, 1987.

[9] Niefield, S.B. and Rosenthal, K.I., Constructing locales from quantales, *Mathematical Proceedings of the Cambridge Philosophical Society* 104: 215-234, 1988.

[10] Ono, H., Semantical Analysis of Predicate Logics Without the Contraction Rule, *Studia Logica* 44: 187-196, 1985.

[11] Ono, H., Semantics for Substructural Logics, in *Substructural Logics*: 259-291, Schroeder-Heister, P. and Došen, K., Editors, Clarendon Press, 1993.

[12] Yetter, D.N., Quantales and (Noncommutative) Linear Logic, *Journal of Symbolic Logic* 55 : 41-64, 1990.