Logic Programming was created under the auspices of efficient computation from deduction. As David Warren said, "Programming should be an intellectually rewarding activity. Prolog helps to make it so."

In the most general terms, this project aims to examine and extend the ways that Prolog handles negation. The meta-logical manner that nearly all existing systems employ is not particularly graceful.

Preliminary Definitions

**Quantifiers** ᴲ- for all, the universal quantifier

there exists, the existential quantifier

**Literals** Atom A is a positive literal whilst ~A, the negation of A, is a negative literal.

**Clausal Form** A clausal form sentence is the normal form of a first order logic sentence where all of its conditions are conjoined together and all of its conclusions disjoined. Variables are implicitly universally quantified. In clausal form µ § represents 'For All variables X1,...,Xn, µ § if µ §.

**Horn Clause** A Horn clause is a definite program clause of the form µ § or a definite clause goal of the form µ §

**Negation As Failure** Negation as Failure uses the not relation and not G fails if G fails to establish itself as a consequence of the program under evaluation.

**First Order Logical Expression** A first order logical expression is a well formed formulae taking one of the following forms, where A and B are well formed formulae - A, ~A, µ §, µ §, A if B and A iff B.

**Non-monotonic Logic** Non-monotonic Logic is where the rules of inference may be extended to reason with incomplete information. In non-monotonic logic the discovery/augmentation of additional information may invalidate previous conclusions in contrast with any monotonic logic.

**Induction** Mathematical induction is the basis of inductive proofs of program properties. A property is proven if it can be shown inductively for all values which possess the property.

In Prolog all quantification is implicit thus µ § is logically equivalent to µ §.

**Preliminary Definitions**

**Classical Negation** Here ~P is defined as P implying all propositions. ~P is true iff P is false.

**Closed World Assumption** For a ground atom A, ~A is a consequence of program P if A cannot be proved from P.

**Clark Completion** Given a set of predicates of the form µ § the completed definition is µ § where each µ § corresponding to a predicate definition for p is of the form µ §. Axioms for defining the '=' predicate used in the above are included to define the action of unification. The Clark Completion together with the equality theory will be denoted by CompEQ.

**Herbrand Interpretation** A Herbrand Interpretation is a free assignment of truth values to all of the atom in the Herbrand Base, the set of all ground atoms from a given language.

**Minimal Herbrand Model** The minimal Herbrand model of program P is the least number of atoms which are true so that P is satisfied.

**Safe Computation Rule** Given a set of normal goals the computation rule ensures that the selected literal for evaluation is a positive literal or a ground negative literal in the goal.

**Unsafe Computation Rule** A computation rule that does not possess safeness, where only ground negative literals can be selected. Prolog has an unsafe computation rule.

**Floundering** A goal G and program P flounders in the computation of P U {G} reaches a goal containing non-ground negative literals.

**Allowed Clause** µ § is allowed if all variables in A occur positively in µ §.

**Admissible Clause** µ § is admissible if all variables occur in A or in positive positive literals in µ §. Admissibility is more general than Allowedness.

**Stratified Program** A program is stratified if if has a strata such that all clauses µ § occur such that if Li is positive in a strata less or equal to p or if Li is negative in a strata less than the clause. Each separate strata is mapped to an integer.

**Declarative Semantics** These are the model theoretics from first order logic where goals are answered by evaluating there truth values. Informally it is the interpretation of meaning based on logic inference.

**Procedural Semantics** This is the proof procedure of the logic programming system consisting of SLD-resolution and a control strategy.

**Occur Check** Checks when unifying a variable with a term that the variable itself is not embedded in the term

Preliminary Definitions

**Call-Consistency** A program is call consistent iff its dependency graph contains no odd number of negatives in any cycles so that no relation wil depend oddly on itself. In the dependency graph for any clause A:-...,Bi,... there is a postive edge in the graph from Bi to A if Bi is a positive literal and a negative edge if Bi is a negative literal. Call-Consistency is due to Sato.

**Independence of the Computation Rule** All distinct answers will be returned regardless of any computation rule.

**Covering** ¥ is a covering of term t iff for each ground Herbrand instance g\* of t there exists t" in ¥ such that g is a ground instance of t". {0,s(X)} and {X} are coverings of the language L = {0,s(X)}. ¥ is an exact covering of t iff it is a covering and for all terms t" in ¥, t" is a ground instance of t. {s(0),s(s(X))} is an exact covering of {s(X)}.

**Abduction** The most basic definition of abduction, due to Poole, is: from B and B:- A we can derive A. A is said to be the hypothesis that explains observation B.

**Modal Negation as Failure** Provability under negation as failure is treated as a modal operator and the semantics are formed accordingly.

**Ground Instantiation** The ground instantiation of a program P is the set of all ground instances for all of its constituent clauses, denoted by G(P).

2.3 The Domain Closure Axiom

The completion augmented with the Domain Closure Axiom (DCA, in the sequel) is another contender for NAF reference theory and is introduced thoroughly in [Man88]. The DCA, in addition to the equality theory, forces any model to have an interpretation domain where each object is constructed using interpretations of the given constants and functions. The Domain Closure Axiom is:-

µ §

Maher stated, "EQDCA is a complete theory," where EQ is the completion equality axioms that themselves characterise the unification process. Therefore the EQDCA interpretations are much closer to Herbrand interpretations. The domain closure axiom is innate within the CWA.

2.4 The Strict Completion

[Dra91] introduce a slight variation of the Clark Completion which is shown to always be consistent. The strict completion is developed in such a way that examines how the undesirable aspects of the completion arise and alters the program so that they are removed. The original program is transformed which possesses the same answer and finite failure sets yet forms a consistent completion.

The strict completion has the following definition,

compSTRICT(P) = comp(split(P))

**2.6 Non-monotonic Techniques**

**Perfect/ Stable models**

Other semantic theories for NAF are now presented briefly. Perfect models [Ser90] give special attention to circumscribed predicates in models where their extensions are as small as possible. Informally, perfect model creation drops as many of the circumscribed predicates as possible so that we still have a model. Priorities for circumscription may be asserted to create an ordering for the minimisation of predicates. The highest degree of circumscription is achieved via the prioritisation of specific atoms. Priorities may capture different readings of the predicates from non-Horn clauses. For example, the clause µ § can have the different readings of q µ § p,~r or r µ § p,~q depending on prioritisation.

Perfect models then use these prioritised predicates to build the canonical model repeatedly applying the prioritied predicates. Priorities are found from strata of the stratified program. A perfect model is one which has the most atoms of a low priority and is minimal. Perfect model semantics can also be applied to constructive negation (§3).

Stable models do not require stratification, they use the notion of a belief modal operator; ~q expresses the fact that q is not believed. A set of beliefs is stable if it represents beliefs of a 'rational' agent where the program is a premise. A stable set of beliefs is equivalent to logical consequences of its beliefs. If this is the case then it has been proven [Ser90] that we have a minimal model.

Negation as failure can be read abductively based on the work of Eshgi and Kowalski in 1989. The semantics of any program P is a set \_ of variable free negative literals whose union satisfies the following integrity constrants:

µ §and

µ §

The contents of the set \_ are used to decide the truth (or falsity) of all ground clauses. The first constraint decides consistency and the second totality. Problems on deciding whether the integrity constraints are too stringent or weak have still to be resolved as we see for µ § which has two sets \_1 = {~q} and \_2 = {~p}.

Preliminary Definitions

**Negation By Constraints** NBC is an extension to NAF that allows any universal and existentially quantified formulae and inequalites to be in the set of goals and subgoals. In this mode answers may be returned from non-ground negative literals. Negation By Constraints is only called into operation when NAF would flounder.

**Deductive Databases** A deductive database, according to John Lloyd, is a finite collection of statements of the form µ § where W is any typed first order formula. A typed formula implies that all variables and terms belong to a finite set of elements called types.

**Constraint Sets** A constraint set is a list of constraints, one for each variable in the goal. Ordering of variables in a constraint set is dictated by their locality.

**Constraint Dropping** A constraint is dropped after successful evaluation when it is not required for the final answer. If the goal is ?r(X) for the program r(X):- q(X,Y) q(X,Y):- Y \_ 2 then {X/a} is an answer and the constraint Y\_2 is dropped from the constraint set.

Preliminary Definitions

**Constructive Negation** This explicitly negates the disjunction given by the completion to a goal. Formally it is:

µ §

Preliminary Definitions

**Intensional Negation** This is a transformation technique that given a set of predicates p synthesises the definition µ § whose success set is equivalent to p's finite failure set.

**Herbrand Generator** A predicate for generating all Herbrand terms from the given language. For every constructor in the language the generator requires a recursive instance for all terms that will behave fairly to all terms if possible.

**Tamaki-Sato Transforms** These refer in the most basic context to unfolding, where the body of a clause replaces its head in the body of another and folding where a conjunct of literals in a clause body may be replaced by a corresponding clause head in the program.

**Left-Linear Program** No clause head contains more than one occurrenceof the same variable. Explicit unification must occur in the body for variables to be unified in the head. Non-left linear programs will lose the unification in clause heads during set-theoretict complement calculation.

**SLDN Resolution** As SLD resolution but incorporates a special procedure to deal with universally quantified variables that occur during transformation.

Qualified Answers are examined not with regard to partial evaluation, their more general use, but for obtaining transformed goals so that a query will return more answers than negation as failure.

Preliminary Definitions

**Qualified Answer** Given a program P, goal G and an answer substitution ß such that G·ß follows from the program. A Qualified Answer imposes a hypothesis H such that the theorem [G·ß if H·ß] follows from the program. Answers are qualified by H·ß and so are not the more standard answers substutions.

Manipulation of goal formulas through substitutions are also introduced. We detail generation of an induction schema using the following program taken from [Ster86]:

sumlist([],0).

sumlist([I|Is],Sum):-

sumlist(Is,Partsum),

add(I,Partsum,Sum).

This relation includes the smallest set of pairs of terms that includes a pair ([],0) and that for any term I, includes ([I|Is],S) whenever it includes (Is,Ps) and (I,Ps,S) is in the add relation. For any Herbrand Interpretation we let Q(A,B) denote a binary relation over terms. Hence the following computational induction scheme is arrived at:-

µ §

The application of this induction scheme is now achieved by resorting to inference rules such as definite clause inference, and, or and implication deletion, negation as failure inference and simplification where a goal is assumed to be true or false.