Non-Monotonic Predicate Logics
Syntax, Semantics, Completeness

W. Marek,[1] A. Nerode[2] and J. Remmel[3]

1 Summary

To simplify our account, rewrite your favorite usual predicate logic, classical, modal, intuitionistic, as a logic of the set L of all statements. This is an inessential change. As Tarski once did, arrange it so that only statements occur in all axioms a and rules of inference I, all of which have the (monotonic) form "from $a, a', ...,$ infer a''. Then a deductively closed set D is merely a subset D of L such that if a is an axiom, then a is in D; and for all such I, if $a, a', ...$ are in D, then c is in D. If A is a subset of L, call D grounded in A if every member of D has a finite derivation from A using the rules of inference I. Call E an extension of A if E is deductively closed and grounded in A. In the usual (monotonic) logics if E is an extension of A, then E is uniquely determined, the deductive closure of A via an ordinary inductive definition. We associate with each usual logic "nonmonotonic prolongations" based on additional rules of inference only, keeping the statements exactly the same

[1]Department of Computer Science, University Kentucky, Lexington, KY 40506–0027. Currently in Mathematical Sciences Institute at Cornell University. Work partially supported by N.S.F. grant RII 8610671 and Kentucky EPSCoR program and the ARO contract DAAL03-89-K-0124

[2]Mathematical Sciences Institute, Cornell University, Ithaca, NY 14853. Work partially supported by NSF grant MCS-83-01850 and ARO contract DAAG629-85-C-0018.

[3]Department of Mathematics, University of California at San Diego, La Jolla, CA 92903. Work partially supported by NSF grant DMS-87-02473.

and keeping all old axioms and rules of inference. But now allow a set N of additional (nonmonotone) rules of inference, all of the form "if a, a', a'', ... are in E, and b, b', b''... are not in E, then c is in E. Define extension of A as before, but allowing rules of inference in N too, getting the notion of extension relative to N. This notion is not given by a finite monotone inductive definition, and A may have no, one (consistent or inconsistent), or many extensions relative to N. Non-monotonicity resides solely in the additional rules of inference of N, not in the statements, and is adequate for intended computer science applications, and gives well-behaved non-monotonic predicate logics for default logics, diagnosis logics, truth maintanence logics, PROLOG with negation as failure, in classical, intuitionistic or modal predicate logics. Consistent extensions of A relative to N represent possible "complete sets of beliefs" that can be held, after having accepted the "facts" in A, and the "rules of thumb" in N. Given the underlying logic and N, under what conditions does a given A have at least one consistent extension? We outline the proof rules that answer this by a completeness theorem proved via systematic tableaux appropriate to the calculus, for classical, intuitionistic, and modal predicate logics. We also outline one inteneded application area. With our results, non-monotonic predicate logics attain approximately the same degree of maturity as programming logics such as predicate dynamic logic. A lot of previously undiscovered connections with dynamic logic then surface, but cannot be explored here. Proof rules can be given in any of the usual styles, tableaux, natural deduction, sequents, etc. Here are some comparisons with the usual logics mentioned above. In the usual predicate logics, tableau proofs are recursive, finitely branching, well founded, trees labelled with statements; for our nonmonotonic predicate logics, the change needed here is to allow the trees to countably branch. Just as for Beth-Smullyan-Fitting tableaux for the usual logics, with every statement is recursively associated a recursively generated systematic tableaux such that the statement is true in all extensions of A (relative to N) if and only if the systematic tableaux is well-founded (and then is a proof). This method, not withstanding the potentially countably branching trees, does have a lot of ordinary computational content: an incidental byproduct is decision methods for propositional finite nonmonotone theories for the usual propositional logics, classical, intuitionistic, modal, dynamic, most of which were not known before this. Just as in ordinary logic the tableaux proof, which uses elimination rules only, suggests introduction rules, leading to a full natural deduction system, and these lead to a full sequent calculus, and a corresponding consistency property. In the usual logics, if a recursive theory has a model at all, then it has a model recursive in a complete r.e. set. Here the analogue of a model is a consistent extension, and here a recursive axiom set which has at least one consistent extension has a consistent extension recursive in a complete Π^1_1 set. This is as well as one can do; there are recursive (or even finite) N which have consistent extensions, but no consistent hyperarithmetic extensions (of the null set). We can also develop a purely finitary version, based on induction principles, analogous to arithmetical, or interpreted, dynamic logic, which may prove useful for a belief revision calculus. The outline below is enough so that, with [Marek-Nerode Remmel (a)], one could in principle work out the rest. The full abstract for publication will outline all these results. In this abstract we give definitions, motivation, and one proof outline.

2 Background

Non-monotonic logics are now ubiquitous in AI and computer science. These can be thought of conveniently as dealing with possible "points of view" (technically, consistent extensions) based on knowledge (that is, currently presented facts) and "rules of thumb" (technically, additional nonmonotonic inference rules). Among non-monotonic logics with large literatures are truth maintainance systems [Doyle], default logics [Reiter], and PROLOG with "negation as failure" [Clark]. In 1990 most papers on nonmonotonic logics still only offer theorems in classical based propositional logics. We gave a logic-free common generalization of the most important existing theories in our theory of nonmonotonic rule systems [Marek-Nerode-Remmel (a) (b) (c) (d)], announced in LICS 90. This gave a common simple abstract framework for definitions, theorems, algorithms, semantics, syntax, for all these systems. Among other things, we showed by uniform methods that any of the subjects above is coextensive with any of the others. But, more important, these subjects were seen to have the same set theoretic and computational content as various mathematical theories, such as theories of marriage problems or theories of coverings of partially ordered sets, a mathematical connection not previously known, allowing application of known bodies of theories to nonmonotonic reasoning. But our previous papers did not grapple directly with the question of what are appropriate nonmonotonic predicate logics. Nonmonotonic predicate logics are of great potential importance. Most papers on nonmonotonic propositional logics use as their principal computer science motivation examples which are definitely in nonmonotonic predicate logic, which therefore are not covered by the mostly propositional logic theorems of the papers. There have been earlier attempts at developing nonmononotonic predicate logics. They have a tenative flavor because either their proof rules or their semantics or their motivations have been sketchy. This tenative flavor looks in hindsight as if it was due to two sources. One was that there is a need to fix firmly in mind the intended computer science application in order to distill from many possibilities a single semantics, for which it makes sense to develop proof rules and a completeness theorem. (In fact, we can now see that one gets different proof rules and semantics for different applications.) Second, there is a need to use somewhat more than ordinary first order logic methods, nonmonotonic predicate proof rules and semantics are a bit more complicated than the literature might lead one to expect. We find here that they are tractable for about the same reason that predicate dynamic logic was tractable. In fact, there are deep relations between non-monotone predicate logics and predicate dynamic logics, which will be explored in later papers.

3 Non-monotone rule systems

We repeat here some of the terminology of [Marek-Nerode-Remmel (a)], so as to set our work in perspective, although the definitions in the summary above are enough for this paper. Define a nonmonotonic rule system as a pair (U, R) consisting of a nonempty set U and a set R of rules of the following form. (These rules are (implicit) conditions on a subset E of U.) If $a, a', a'', ...,$ are in E, and $b, b', b'', ...$ are not in E, then c is in E. Here $a, a', ..., b, b', ..., c$ is a finite sequence of elements of U. The $a, a', ...$ are called premises, the c is called the conclusion, the $b, b', ...$ are called restraints. If there are no restraints, the rule is monotone; if there are no premises or restraints, the rule is called an axiom. Each nonmonotone rule gives rise to a corresponding monotone rule obtained by omitting all its restraints. Suppose A and E are subsets of U. The fundamental concept is "E is an extension of A", defined as a subest of U satisfying two conditions.

1) E is closed under application of the nonmonotone rules. (This says that E is deductively closed under the rules.)

2) Every element of E has a finite deduction from A using only monotone rules arising from those nonmonotone rules of the system whose restraint statements are correct for E . (This says that E is grounded in A.)

Without further restrictions on the nonmonotone rule system (U, R), an A may have no extensions, one extension (which may be U), or many extensions. In logics, extensions are regarded as possible "points of view" based on facts A and "rules of thumb" the rules of the system.

Here is a convention extending that of the summary, leading to a short exposition of an intended application and its connection with Dynamic Logic. Choose a usual logic, such as propositional or predicate classical, intuitionistic, or modal logic. Let L consist of all formulas. Use a formulation where free variables are disjoint from the set of all bound variables. Further, make the restiction that in deductions we never substitute for free variables and never quantify over them; that is, treat them as if they are constants in all deductions. All the standard proof systems, such as sequents, natural deduction, Hilbert type systems, or tableaux, can be trivially readjusted so that all the axioms and rules of inference respect this. Write that predicate logic as a nonmonotone rule system (L, M) with set M of monotone rules, all obtained by translating an instance of each predicate logic rule of inference or axiom into a rule. This is simply the nonmonotone rule system uniform notation for the usual logic. If N is a set of (possibly non-monotone) rules containing M, then we call (L, N) a non-monotone predicate logic theory for L. To repeat, we may now investigate consistent extensions for (L, N), for N a set of rules in L, for any of the usual logics.

4 Motivating example

Think of a particular execution sequence of states of a computer, and fix in mind a particular current machine state S from that sequence. Generalizing the Propositional Dynamic Logic philosophy, choose a logic L which whose statements denote propositions about machine states. Associate with S the theory in L of all statements true of that state. Usually, when using the computer, we already know some set A of truths about the state S and also have a set of "rules of thunmb" for diagnosing some features of the state from these known features, which we express as a set N of nonmonotone rules. There may be no, one, or several consistent extensions of A in L to use as a diagnosis system for diagnosing some propositions about current machine state S from known propositions about machine state. If L is a propositional logic, we have said enough to have a precise interpretation comparable with the usual one for propositional dynamic logic. Now let's look at the predicate case.

Assume, as in models of Dynamic Logic based on shared memory machines, that machine state is described completely by the values stored in program variables $x, x', x'', ...$ Assume these values are from a fixed relational system. With abuse of notation, let these be the only allowed free variables in formulas of L. (Remember that we don't bind these variables or substitute for them, we treat them as constants). A machine state S can be defined either as a map of these program variables into the relational system or as an n-tuple $(S(x), S(x'), ...,)$. This is interpreted as the n-tuple of contents of the respective memory locations named by $x, x', x'', ...$ We omit discussion of the apparatus of dynamic logic, namely modal operators corresponding to programs, programs denoting accessibility relations between states, test and assignment as atomic programs, etc. (But we would be perfectly happy if L were predicate dynamic logic We will use extensions in the nonmonotone versions of dynamic logic in later papers introducing "belief revision programming".)

At any state S, let $T(S)$ be the set of all formulas $F(x, x', ...)$ such that $(S(x), S(x'), ...)$ satisfies $F(x, x', ...)$. That is, $T(S)$ is the type of S in the sense of model theory. Dynamic logic is concerned with statements in $T(S)$ as preconditions and postconditions of programs. What is the diagnosis interpretation? Suppose we know some facts A from L satisfied at S and have a set N of nonmonotonic rules of L, which we always assume contains at least the actual deduction rules of the underlying calculus). This gives us nonmonotone theory (L, N). A consistent extension E of A relative to N is then a coherent system for diagnosis. We emphasize that the facts A and the nonmonotone rules are applied to get "diagnoses" assuming we know the particular S. Note that A was merely assumed satisfied at state S, not at other states the machine might take on. This is a "local diagnosis" system at a single current state, to used no matter what the current state is.

Remark. Here is another view of free variables in nonmonotonic rules, the universal interpretation. Interpret them instead as universally quantified and allow substitution of terms in x, x', x'', for the free variables x, x', x''.... What would be the meaning of a universal interpretation nonmonotonic rule in the states model? In dynamic logic, the machine changes state $(x, x', ...)$ according to a program always by an atomic assignment program substituting a term for (say) x, to get new state $(t(x), x', x'', ...)$. If we use a universally interpreted nonmonotonic rule, allowing arbitrary substitutions of terms for the program variables, to be used this rule has to be one valid in all accessible machine states. Thus such a rule is a "frame rule", or universal program invariant, correct at any state caused by execution of any program. Only a few of the rules of a diagnostic system are likely to have this property, rules distinguishing different states are the norm for diagnosis. Of course, using these rules is no real generalization, each is merely a schema over our system; that is, we can replace such a rule by all its substitutioin instances, and freeze the free variables afterwards. Noteworthy is that this model makes sense in the case of classical or constructive dynamic logic, sequential or concurrent; see [Nerode-Wijesekera]. We can give similar motivating examples for the other standard default applications, default logic, truth maintenanxce, and PROLOG with "negation as failure".

5 Completeness

In the usual logics, semantic inconsistency of A means there is no model of A. In clasical (intuitionistic) propositional logic, this means that there is no proper maximal (prime) filter containing A in the Lindenbaum algebra of statements. This represents the essential element of completeness for such a calculus. What corresponds for nonmonotonic theories in logics like these? In our view, based on the literature and the motivating examples, the semantic inconsistency of a set A of statements in the usual logics relative to a nonmonotone rule set N should be defined as the non-existence of any consistent extension for A. This leaves the question of defining syntactic consistency, and giving a completeness proof.

We give a short proof-outline for completeness by tableau which applies to all the usual calculi. But one gets a better computational proof and a clearer semantic proof and better proof systems by doing the subjects individually. The systems in [Nerode (a),(b)], [Nerode and Wijesekera of tableaux for intuitionistic, modal, and dynamic logic respectively can be extended to nonmonotonic theories in these logics neatly, and an outline of this will go in the final abstract.

We look at the question of rules for developing tableaux for determining whether a given N has at least one consistent extension (of the empty set A); the case for general A presdents no additional difficulties.

Given N, what are the rules of development for a systematic tableaux searching for a consistent extension? What we want to ensure is that any infinite branch of the completed systematic tableaux describes a consistent extension, and simultaneaously that in case every branch of the completed systematic tableaux is finite, there cannot exist any consistent extension at all. Every systematic tableau node is labelled "x in E" or "x not in E" for x a formula of L. We outline the systematic tableaux development rules, and leave the actual proof to the imagination. See Nerode, loc. cit, or Smullyan [1968] or Fitting [1981] for unexplained noation.

EXHAUSTION DEVELOPMENT: At any stage n, we append to the base of every open branch B two branching nodes labelled respectively "x in E" and "x not in E", where the n-th formula of L is x. This rule assures that we cover all possibilities for extensions.

For each nonmonotone rule I in N of the form:

If $a, a', a'', ...$ are in E, and $b, b', b'', ..$ are not in E, then c is in E.

we have two corresponding tableau development rules

NONMONOTONE RULE DEVELOPMENT FOR I.
1) Suppose "a in E", "a' in E", ..., "b in E, "b' in E" all occur as enties on open branch B. Then we must eventually append at the base of B a node with "c in E" as label, and the branch remains open.

2) Suppose that "c not in E" occurs as an entry on open branch B. Then we must eventually append at the base of B separate branching nodes labelled respectively "a not in E", "a' not in E", ..."b in E", .."b' in E", ...

This development rule is much more powerful than it may appear. Although it only enforces a Horn sentence, these Horn sentences I encompass the whole deductive power of the underlying logic, since they encompass both the axioms and rules of inference of the whole underlying logic.

VACUOUS EXTENSION RULE: At any stage n, for every open branch B we append an unlabelled node below.

This rule simply will enforce that no finite branch defines a consistent edxtension, or that all finite branches are closed.

CLOSURE RULES:
1) An open branch B is closed as soon as it contains as two entries a direct contradiction "a in E", "a not in E".

2) An open branch is closed as soon as it contains two enties "b in E" and "(not b) in E".

The first enforcs that we eliminate impossible extensions, the second that they be consistent.

Create a complete systematic tableaux development procedure using these rules. This gives a finitely branching tree in which every infinite branch G describes the consistent deductively closed set consisting of all x in L such that "x in E" is an entry on G. Further, if a consistent deductively closed set exists at all relative to N, then there exists such an infinite branch. But we are only interested in infinite branches G which descibe extensions, that is, which are grounded. This means that every entry "x in E" on B has a valid (monotone) deduction arising from a valid non-monotone rule system deduction from the axioms "y in E" on that branch. (the axioms are the conclusions of premiseless restraintless rules I.)

GROUNDING. Introduce a systematic procedure for listing all monmonmotone deductions whatsoever, a certain finite number having been finished by stage n of the construction of the systematic tableaux above. At a given stage n, call a finite branch B', properly extending a finite branch B, a justifying extension at stage n, of B if the nonmonmotone deductions constructed by stage n are adequate to ground every entry on B of the form "x in E" from axioms which are entries on B', and were not similarly adaquate at any earlier stage. Say that the triple (B, B', n) is acceptable. Say that accepted triple (C, C', m) is an immediate succsssor of accepted triple (B, B', m) if m is at least n, and C is B'. This gives the desired countably branching tree such that if there is an extension at all, there is an infinite branch describing an extension.

The listing of subsidiary grounding deductions can harmioniously be integrated with the tableaux proof procedures for the underlying calculus, which we do not do here. This can be done using classical tableaux for classical predicate logic [Smullyan 1968], intuitionistic tableaux for intuitionistic logic [Fitting 1981], [Nerode (a)], using modal tableaux for predicate modal logics [Nerode (b)], useing dynamic logic tableaux for dynamic logic [Nerode-Wijesekera]. Once one has a completely described tableaux procedure, including the subsidiary tableaux, one write out natural deduction and sequent calculi, consistency properties, and Hilbert type systems. This will be done in the full abstract, along with a more detailed statement of other results alluded to in the summary.

6 Conclusions

We have, for the first time, given a complete account of nonmonotonic extensions of the usual predicate logics of all kinds, and indicated the form of completeness theorem. This is a mathematical foundation for much predicate nonmonotonic predicate reasoning, including default logics, truth maintainance systems, diagnosis systems. The method of proof incidentally gives the first known decision procedures for nonmonotonic theories in many nonclassical logics, including intuitionistic and modal.

Reesearch areas for the future include:

- Nonmonotonic dynamic logics as program logics for belief revison

- a finitary interpreted version of nonmonotonic logics akin to the interpreted versions of dynamic logics

- A fundamental requirement on nonmonotone rule systems used as hard-wired ”rules of thumb” to control motion is that as new facts about obstacles are perceived, that a new extension to control motion can always be computed. We have found a general class of nonmonotone systems (generalizing Reiter’s normal systems) which satisfy this requirement and are much easier to compute with since they have a much easier completeness theorem which much smaller proofs. These systems are under investigation, have an understandable structure, and are suitable for a belief revision calculus.

7 References

Clark, K. L., ”Negation as Failure”, in Logic and Databases (H. Gallier and J. Minker, eds.) Plenum Press, NY, 293-322.

Doyle, J., ”Truth Maintenance Systems”, Art. Int. 12 (1979), 231-272.

Fitting, M., ”Proof Methods for Modal and Intuitionistic Logics”, D. Reidel, 1981.

Harel, D., First Order Dynamic Logic, Lecture Notes in Computer Science, 68, 1976.

Marek, W. Nerode, A., Remmel, J, A, (a), ”Theory of Nonmonotonic Rule Systems I”, Ann. Math. and Art. Intell (1990), 241-273

Marek, W. Nerode, A., Remmel, J, (b), ”A Theory of Nonmonotonic Rule Systems II”, Math. Sci. Institute Cornell Tech. Rpt. 1990, to appear in Ann. Math. and Art. Intell (1991).

Marek, W. Nerode, A., Remmel, J, (c), ”How complicated is the set of stable models of a recursive logic program?”, Math. Sci. Instrute Cornell Tech. Rpt. 1991.

Marek, W. Nerode, A., Remmel, J, (d), ”Classifying rule systems according to the number of derivations of elements”, in prep.

Nerode, A., (a), ”Some lectures in modal logic”, Technical Report, M.S.I. Cornell University, 1989, to appear in Marktoberdorf volume, 1989 NATO Summer Instutute, NATO Science series, Springer 1991.

Nerode, A., (b), ”Some lectures in intuitionistic logic”, in Logic and Computer Science (G. Odifreddi, ed.), LCNS 1429, Springer, 1990.

Nerode., A. and Wijesekera, D., ”Constructive concurrent dynamic logic”, Ann. Math. and Art. Intell, to appear 1991.

Reiter, R., ”A Logic for Default Reasoning”, Art. Intell 13 (1980), 81-132.

Smullyan, First Order Logic, Springer, 1968.