The Propositional Mu-Calculus is Elementary

Robert S. Streett
Computer Science Department
Boston University
Boston, MA 02215
USA

E. Allen Emerson
Computer Sciences Department
University of Texas
Austin, TX 78712
USA

ACKNOWLEDGEMENT: The work of the second author was supported in part by NSF grant MCS-8302878.

ABSTRACT: The propositional mu-calculus is a propositional logic of programs which incorporates a least fixpoint operator and subsumes the Propositional Dynamic Logic of Fischer and Ladner, the infinite looping construct of Streett, and the Game Logic of Parikh. We give an elementary time decision procedure, using a reduction to the emptiness problem for automata on infinite trees. A small model theorem is obtained as a corollary.

1. Introduction

First-order logic is inadequate for formalizing reasoning about programs; concepts such as termination and totality require logics strictly more powerful than first-order (Kfoury and Park, 1975). The use of a least fixpoint operator as a remedy for these deficiencies has been investigated by Park (1970, 1976), Hitchcock and Park (1973), deBakker and deRoever (1973), deRoever (1974), Emerson and Clarke (1980), and others. The resulting formal systems are often called mu-calculi and can express such important properties of sequential and parallel programs as termination, liveness, and freedom from deadlock and starvation.

Propositional versions of the mu-calculus have been proposed by Pratt (1981) and Kozen (1982). These logics use the least fixpoint operator to increase the expressive power of Propositional Dynamic Logic (*PDL*) of Fischer and Ladner (1979). Kozen's formulation captures the infinite looping construct of Streett (1982) and subsumes Parikh's Game Logic (1983a, 1983b), whereas Pratt's logic is designed to express the converse operator of *PDL*. The filtration-based decision procedure and small model theorem obtained for *PDL* extend to Pratt's mu-calculus, but the ability to express infinite looping renders the filtration technique inapplicable to Kozen's version.

Kozen (1982) and Vardi and Wolper (1984) have obtained exponential time decision procedures for fragments of Kozen's mu-calculus. Both fragments can expresses all of *PDL*, but are not strong enough to capture the infinite looping construct of Streett (1982). Kozen and Parikh (1983) have shown that the satisfiability problem for the full

propositional mu-calculus can be reduced to the second-order theory of several successor functions (SnS). By results of Rabin (1969) this supplies a decision procedure for the propositional mu-calculus, but one which runs in non-elementary time, i.e., time not bounded by any fixed number of compositions of exponential functions. Meyer (1974) has shown that this is the best that can be achieved using a reduction to SnS.

2. Syntax and Semantics

The formulas of the propositional mu-calculus are:

(1) Propositional letters P, Q, R, . . .
(2) Propositional variables . . . , X, Y, Z.
(3) Ap, where A is a member of a set of program letters A, B, C, . . . and p is any formula,
(4) $\neg p$,
(5) $p \vee q$,
(6) $\mu X.f(X)$, where $f(X)$ is any formula syntactically monotone in the propositional variable X, i.e., all occurrences of X in $f(X)$ fall under an even number of negations.

A sentence is a formula containing no free propositional variables, i.e., no variables unbound by a operator. Mu-calculus sentences are satisfied in Kripke structures, which interpret propositional letters as subsets of states and program letters as binary relations on states. The formula Ap is true in a state when there is an A edge to a state satisfying p. In the formula $\mu X.f(X)$, f denotes a monotone operator on sets of states, and $\mu X.f(X)$ is interpreted as the least fixpoint of this operator, i.e., the least set of states X such that $f(X) = X$.

Examples: The sentence $\mu X.P \vee AX$ is true at a state x if there is a chain (possibly empty) of A edges leading from x to a state satisfying P. It is equivalent to the sentence $\langle A^* \rangle P$ of Propositional Dynamic Logic (PDL). The sentence $\mu X.P \vee A(Y.X \vee BY)$ is equivalent to the PDL sentence $\langle (AB^*)^* \rangle P$.

It is convenient to reduce the problem of satisfiability over the general models described above to satisfiability over a special class of models, the tree models.

Definition: A deterministic model is a Kripke structure in which the relations corresponding to the programs are partial functions; for each state x and program A there is at most one A edge from x. A tree model is a deterministic model whose universe of states is the set of words over an alphabet of program letters. Each program is interpreted as a binary relation in the obvious way: there is an A edge from x to xA.

Proposition 1. There is a translation of mu-calculus

sentences such that a sentence is satisfiable if and only if its translation is satisfied in a tree model.

Outline of Proof: Kozen and Parikh (1983) establish a Lowenheim-Skolem theorem for the propositional mu-calculus; if a sentence is satisfiable, then it has a countable model. These countable models can be further restricted to be deterministic; this is accomplished by translating Ap as $A(\mu X.p \vee BX)$, where B is a new program, a technique due to Parikh (1978). It is not difficult to expand and unwind the resulting models into tree models.

In a tree model, any sentence can be put into a special positive form, by using the following DeMorgan-like laws to move negations until they are only applied to propositional letters.

$$(1)\ \neg\neg p \rightarrow p,$$
$$(2)\ \neg(p \vee q) \rightarrow (\neg p) \,\&\, (\neg q),$$
$$(3)\ \neg Ap \rightarrow A(\neg p),$$
$$(4)\ \neg(\mu X.f(X)) \rightarrow \nu X.\neg f(\neg X).$$

The formula $\nu X.f(X)$ represents the greatest fixpoint of the monotone operator f.

Examples: The sentence $\mu X.P \vee (AX \,\&\, BX)$ is true when there is a finite binary tree of A and B edges with a frontier of states satisfying P. The sentence $\nu X.P \,\&\, (\mu Y.BX \vee AY)$ is true when there is an infinite $AB*$ chain of states satisfying P.

In what follows we shall assume that all sentences are in positive form and that all models are tree models.

3. Ordinal Ranks and Signatures

By the Tarski-Knaster theorem, $\mu X.f(X)$ can be defined by transfinite induction, i.e., $\mu X.f(X) = \bigcup_\alpha f^\alpha(false)$, where

$$f^0(false) = false$$
$$f^{\alpha+1}(false) = f(f^\alpha(false))$$
$$f^\lambda(false) = \bigcup_{\alpha<\lambda} f^\alpha(false), \lambda \text{ a limit ordinal.}$$

A mu-sentence $\mu X.f(X)$ has rank α at a state x if $f^\alpha(false)$ is true at x. Since a mu-sentence can contain other mu-sentences as subsentences, it is useful to associate a sequence of ordinal ranks to a sentence. Bounded length sequences of ordinals can be well-ordered lexicographically.

Definition. The mu-height of a sentence is the depth of nesting of mu-subsentences of the sentence.

Example: The sentence $\mu X.P \vee A(\mu Y.X \vee BY)$ has mu-height 1, since the subformula $\mu Y.X \vee BY$ is not a sentence.

Given a sentence p of mu-height n and a sequence of ordinals $s = \alpha_1 \cdots \alpha_n$, we let $p{:}s$ denote the sentence obtained by replacing each mu-subsentence $\mu X.f(X)$ of p by $f^{\alpha_i}(false)$, where i is the mu-height of $\mu X.f(X)$. A sentence p has signature s at a state x if $p{:}s$ is true at x.

Examples: Consider $\mu Y.(\mu X.P \vee A(\mu Z.X \vee BZ)) \vee BY$, equivalent to the *PDL* sentence $\langle B^*\rangle\langle(AB^*)^*\rangle P$. This sentence has mu-height 2, and if P is true at a state $xBABABBBBB$, then this sentence has signature 3-2 at x, 3-1 at xB, 2-2 at xBA, 2-1 at $xBAB$, 1-6 at $xBABA$, and so on down to 1-1 at $xBABABBBBB$. Infinite ordinals can arise in signatures through the interaction of mu-sentences and nu-sentences. Consider $\nu X.(\mu Y.(P \vee BY) \,\&\, AX)$, equivalent to the *PDL* sentence $[A^*]\langle B^*\rangle P$. In a tree model in which the states satisfying P are precisely A^nB^n, for $n \geq 0$, the signature of this sentence at the root will be ω.

Lemma: The following rules hold of signatures:

(1) if $p \vee q$ has signature s at x, then either p or q has signature s at x.
(2) if $p \,\&\, q$ has signature s at x, then both p and q have signature s at x.
(3) if Ap has signature s at x, then p has signature s at xA.
(4) if $\mu X.f(X)$ has signature s at x, then $f(\mu X.f(X))$ has signature t at x, where t lexicographically precedes s.
(5) if $\nu X.f(X)$ has signature s at x, then $f(\nu X.f(X))$ has signature s at x.

Proof (for case 4 only): Suppose $\mu X.f(X)$ has mu-height n. The mu-subsentences of $f(\mu X.f(X))$ can be divided into three classes:

(1) The proper mu-subsentences of $\mu X.f(X)$, with mu-height $< n$.
(2) $\mu X.f(X)$ itself, with mu-height n.
(3) Mu-sentences properly containing $\mu X.f(X)$, with mu-height $> n$.

If $\mu Y.g(Y)$ is in the first class and can be replaced by $g^{\alpha}(false)$ within $\mu X.f(X)$ at x, then it can be similarly replaced within $f(\mu X.f(X))$ at x. If $\mu X.f(X)$ has rank α at x, then $\mu X.f(X)$ can be replaced by $f^{\beta}(false)$, for $\beta < \alpha$, within $f(\mu X.f(X))$ at x. Hence if $\mu X.f(X)$ has signature $s = \alpha_1 \cdots \alpha_n$ at x, then $f(\mu X.f(X))$ will have signature $t = \alpha_1 \cdots \alpha_{n-1} \beta_n \beta_{n+1} \cdots \beta_m$ at x, where $\beta_n < \alpha_n$, so that t lexicographically precedes s.

Example: Consider $\mu X.(\mu Y.P \vee CY) \vee A(\mu Z.X \vee BZ)$, equivalent to the *PDL* sentence $\langle(AB^*)^*\rangle\langle C^*\rangle P$. In a model in which P is true at $xABBBACCCC$, this sentence has signature

5-3 at x, indicating that P can be reached via two $(AB*)$'s and four C's. The derived sentence, equivalent to the *PDL* sentence $\langle C*\rangle P \vee \langle A\rangle\langle B*\rangle\langle (AB*)*\rangle\langle C*\rangle P$, has signature 5-2-4 at x, indicating that, from xA, P can be reached via three B's, one $AB*$, and four C's .

4. The Decision Procedure

Given a sentence p, we will construct a finite automaton on infinite trees (Rabin, 1969; Hossley and Rackoff, 1972) which recognizes the tree models of p. This automaton will evaluate a given sentence in a candidate tree structure by recursive descent, i.e., by recursively evaluating its consequences. At a disjunction $q \vee r$ contained within a mu-sentence it is necessary to make a careful choice of which disjunct to evaluate. Consider the formula $\mu X.P \vee AX$, equivalent to the PDL sentence $\langle A*\rangle P$, which is satisfied in a tree structure when the formula P is satisfied somewhere along the path of A's. Consistently choosing the disjunct AX of $P \vee AX$ will cause tree structures in which p is false along the path of A's to be mistakenly regarded as models of $\mu X.P \vee AX$.

A choice function for a model is a function which chooses, for every disjunction, one of the disjuncts. Ordinal signatures can be used to define a choice function which selects the true disjunct with lexicographically least signature.

Any choice function over a tree structure determines a derivation relation between occurrences of sentences.

(1) A disjunction, $q \vee r$, derives the disjunct chosen by the choice function,
(2) A conjunction, $q \& r$, derives both conjuncts,
(3) A program sentence, Aq, occurring at a state x, derives q at Ax.
(4) A mu-sentence, $\mu X.f(X)$, derives $f(\mu X.f(X))$.
(5) A nu-sentence, $\nu X.f(X)$, derives $f(\nu X.f(X))$.

Definition. A sentence p at x generates q at y if p at x derives q at y in such a way that q is a subsentence of every derivation step. In particular, note that q must be a subsentence of p, so that a sentence can only generate its subsentences.

Example: $\mu X.P \vee A(\mu Y.X \vee BY)$ at x can derive, but not generate, $\mu Y.((\mu X.(P \vee A(\mu Y.X \vee BY)) \vee BY)$ at xA.

Definition. A mu-sentence $\mu X.f(X)$ is regenerated from state x to state y if an occurrence at x generates an occurrence at y.

Example: $\mu Y.((\mu X.(P \vee A(\mu Y.X \vee BY)) \vee BY)$ can be regenerated from x to xB, but from x to xA. A derivation

from x to xA is possible, but requires $\mu X.P \vee A(\mu Y.X \vee BY)$ as a derivation step.

If we start with a tree model and construct a choice function based on ordinal signatures, then by the Lemma of Section 3 the regeneration relations for mu-sentences will always decrease signature and hence be well-founded. Conversely, if a candidate tree structure can be supplied with a choice function which make the regeneration relations well-founded, it will in fact be a model.

In particular, if the regeneration relations are well-founded, then each occurrence of a mu-sentence is associated with an ordinal, the well-ordering ordinal of the regeneration relation from that occurrence. It is then possible to calculate a signature $s = \alpha_1 \cdots \alpha_n$ for every sentence q at state x, via the definition:

$$\alpha_i = \text{l.u.b.}(\alpha : q \text{ at } x \text{ generates mu-sentence } r \text{ at } y, \; r \text{ has mu-depth } i, \text{ and } r \text{ at } y \text{ has regeneration ordinal } \alpha).$$

We have therefore shown:

Proposition 2. A sentence p is satisfiable if and only if there is a tree model with an attached choice function over which the regeneration relations for mu-sentences are well-founded.

It is then a exercise in automaton programming to show:

Proposition 3. Given a sentence p, we can effectively construct an automaton which expects as input a tree structure with attached choice functions and accepts precisely when the choice function guarantees that the structure is a model of p. The size of this automaton (and the time required to construct it) can be kept elementary in the length of the formula.

Proof: Given a candidate tree structure, the desired automaton checks that the structure is both locally and globally consistent. A structure is locally consistent when no state contains both a propositional letter and its negation, or contains a disjunction without one of its disjuncts, etc. Global consistency consists of the well-foundedness of the regeneration relation for mu-sentences derived from p. It is straightfoward to construct an automaton on infinite strings which, when run down a single path of a tree structure, nondeterministically searches for an infinite descending chain in the regeneration relations. A complement construction then supplies an automaton which, when run down every path of a tree structure, checks global consistency.

Combining Propositions 1, 2, and 3, we have reduced the

satisfiability problem for the propositional mu-calculus to the emptiness problem for finite automata on infinite trees. Since this last problem is elementarily decidable, the mu-calculus is also. The methods of Streett (1981) can be used to show that the decision procedure runs in triple exponential time. As a corollary, we obtain a small model theorem. For if the set of tree models is automaton recognizable, then there must be a finitely generable tree model, i.e., one obtained by unwinding a finite graph. This graph will be a finite model.

5. References

deBakker, J., and deRoever, W. P. (1973), A Calculus for Recursive Program Schemes, in "First International Colloquium on Automata, Languages, and Programming", 167-196.

deRoever, W. P. (1974), "Recursive Program Schemes: Semantics and Proof Theory", Ph. D. thesis, Free University, Amsterdam.

Emerson, E. A., and Clarke, E. C. (1980), Characterizing Correctness Properties of Parallel Programs Using Fixpoints, in "Seventh International Colloquium on Automata, Languages, and Programming", 169-181.

Fischer, M. J., and Ladner, R. E. (1979), Propositional Dynamic Logic of Regular Programs, *Journal of Computer System Science 18*, 194-211.

Hitchcock, P., and Park, D. M. R. (1973), Induction Rules and Termination Proofs, in "First International Colloquium on Automata, Languages, and Programming", 225-251.

Hossley, R., and Rackoff, C. W. (1972), The Emptiness Problem for Automata on Infinite Trees, in "Thirteenth IEEE Symposium on Switching and Automata Theory", 121-124.

Kfoury, A. J., and Park, D. M. R. (1975), On Termination of Program Schemes, *Information and Control 29*, 243-251.

Kozen, D. (1982), Results on the Propositional Mu-Calculus, in "Ninth International Colloquium on Automata, Languages, and Programming", 348-359.

Kozen, D., and Parikh, R. J. (1983), A Decision Procedure for the Propositional Mu-Calculus, to appear in "Second Workshop on Logics of Programs".

Meyer, A. R. (1974), Weak Monadic Second Order Theory of Successor is not Elementary Recursive, *Boston Logic Colloquium, Springer-Verlag Lecture Notes in Mathematics 453*.

Parikh, R. J. (1979), A Decidability Result for a Second Order Process Logic, in "Nineteenth IEEE Symposium on the Foundations of Computing", 177-183.

Parikh, R. J. (1983a), Cake Cutting, Dynamic Logic, Games, and Fairness, to appear in "Second Workshop on Logics of Programs".

Parikh, R. J. (1983b), Propositional Game Logic, to appear in "Twenty-third IEEE Symposium on the Foundations of Computer Science".

Park, D. M. R. (1970), Fixpoint Induction and Proof of Program Semantics, *Machine Intelligence 5*, Edinburgh University Press.

Park, D. M. R. (1976), Finiteness is Mu-Ineffable, *Theoretical Computer Science 3*, 173-181.

Pratt, V. R. (1982), A Decidable Mu-Calculus: Preliminary Report, in "Twenty-second IEEE Symposium on the Foundations of Computer Science", 421-427.

Rabin, M. O. (1969), Decidability of Second Order Theories and Automata on Infinite Trees", *Transactions of the American Mathematical Society 141*, 1-35.

Streett, R. S. (1981), "Propositional Dynamic Logic of Looping and Converse", MIT LCS Technical Report TR-263.

Streett, R. S. (1982), Propositional Dynamic Logic of Looping and Converse is Elementarily Decidable, *Information and Control 54*, 121-141.

Vardi, M., and Wolper, P. (1984), Automata Theoretic Techniques for Modal Logics of Programs, to appear in "Sixteenth ACM Symposium on the Theory of Computing".