ISSN 0105-8517

RESULTS ON THE PROPOSITIONAL μ-CALCULUS

Dexter C. Kozen

DAIMI PB-146
Maj 1982

Matematisk Institut . Aarhus Universitet
DATALOGISK AFDELING
Ny Munkegade - 8000 Aarhus C - Danmark
Tlf. 06-12 83 55

PB-146 Kozen: Results on the propositional μ-calculus

TRYK: DAIMI/RECAU

RESULTS ON THE PROPOSITIONAL μ-CALCULUS

Dexter Kozen[1]
Aarhus University
Aarhus, Denmark

Abstract

We define a propositional version of the μ-calculus, and give an exponential-time decision procedure, small model property, and complete deductive system. We also show that it is strictly more expressive than PDL. Finally, we give an algebraic semantics and prove a representation theorem.

1. Introduction

The propositional μ-calculus refers to a class of programming logics consisting of propositional model logic with a μ (least fixpoint) operator. The μ-calculus originated with Scott and deBakker [SdB] and was developed further by Hitchcock and Park [HP], Park [Pa], deBakker and deRoever [dBR], deRoever [dR], and others. The system we consider here is very similar to one appearing in [dB, chp.8]. Our results however are more inspired by the work of Pratt [Pr], who considers a version $P\mu$. He shows that $P\mu$ encodes PDL, and extends his exponential-time decision procedure for PDL to $P\mu$. He leaves open the problem of strict containment of PDL and does not give a deductive system. The usual proof rules do not readily apply to $P\mu$ due to its formulation as a least root calculus rather than a least fixpoint calculus; this was done in order to capture the reverse operator of PDL. Also, Pratt imposes a rather strong version of syntactic continuity on $P\mu$ which we would like to weaken, since it renders illegal by fiat such useful formulas as $\mu Q.[a]Q$ (this is the negation of the infinite-looping operator Δa of Streett [S]). The restriction allows Pratt's filtration-based decision procedure to extend to $P\mu$, whereas no filtration-based decision procedure can work in the presence of $\mu Q.[a]Q$, since the operator $[a]Q$ is not continuous.

Here we propose weakening the syntactic continuity requirement and returning to the original least-fixpoint formulation to get a system $L\mu$. We lose the ability to encode the reverse operator, however we can show

(1) $L\mu$ encodes PDL with tests and looping (Δ) but without reverse;

1) On leave from IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598, USA.

thus by a result of Streett, $L\mu$ is strictly more expressive than PDL.

(2) We give an exponential-time decision procedure. This improves Streett's upper bound for PDL with Δ.

(3) We give a deductive system for $L\mu$, including the fixed point induction rule of Park [Pa], and prove completeness.

(4) We describe briefly an algebraic semantics and prove a representation theorem.

Familiarity with the μ-calculus and PDL is assumed; see [dB,dR,FL].

2. Syntax and Semantics

$L\mu$ has primitive propositions $P,Q,\ldots$ and programs $a,b,\ldots$, and formulas P, $X \vee Y$, $\neg X$, $\langle a \rangle X$, and $\mu Q.X$, the last allowed only if certain syntactic restrictions are met. We at least require syntactic monotonicity: each occurrence of Q in X is under an even number of negations. We will indicate this by writing such formulas $\mu Q.pQ$. Boolean operators $\wedge, \rightarrow, \equiv, 0, 1$ are defined as usual; $[a]X = \neg\langle a \rangle \neg X$ and $\nu Q.pQ = \neg\mu Q.\neg p \neg Q$. ν is the greatest fixpoint operator. σ represents either μ or ν. In practice we will distinguish between variables (those Q bound by some σQ) and other primitive propositions, although formally there is no distinction. We will often think of $\vee, \wedge, [\,], \bar{P} (=\neg P)$ as primitive, eliminating occurrences of $\neg$ by deMorgan's laws. A formula in such form is called positive.

In section 5 we will also impose the following syntactic restriction, which is somewhat weaker than Pratt's restriction:

(2.1) If $\sigma R.qR$ and $\sigma S.rS$ are subformulas of $\mu Q.pQ$ (possibly $\mu Q.pQ$ itself) each containing an occurrence of Q, then no two occurrences of variables R and S are conjunctively related. (Two formulas are conjunctively related in a positive formula X if $\wedge$ is at the root of the smallest subformula of X containing them). This is explained in section 5.

A standard model is a tuple $M = (S^M, \rho^M, \pi^M)$ where S^M is a set of states, ρ^M: $a \mapsto a \subseteq S^M \times S^M$, and π^M: $P \mapsto P^M \subseteq S^M$. Each formula defines both a set in S^M and an operator on subsets of S^M. If $X = X(\bar{Q})$ has free variables all among $\bar{Q} = Q_1, \ldots, Q_k$, then X defines a k-ary set operator X^M: $(S^M)^k \rightarrow S^M$ as follows:

$$P^M(\bar{A}) \quad = P^M \quad , \text{ P a primitive proposition,}$$
$$Q_i^M(\bar{A}) \quad = A_i \quad , Q_i \text{ a variable,}$$
$$X \vee Y^M(\bar{A}) \quad = X^M(\bar{A}) \cup Y^M(\bar{A}),$$
$$\neg X^M(\bar{A}) \quad = S^M - X^M(\bar{A}),$$
$$\langle a \rangle X^M(\bar{A}) \quad = \{s \mid \exists t \in X^M(\bar{A}) \ (s,t) \in a^M\},$$
$$\mu Q.pQ^M(\bar{A}) \quad = \cap\{B \mid p^M(B,\bar{A}) \subseteq B\}.$$

$\mu Q.pQ^M$ can also be defined equivalently as

$$\mu Q.pQ^M(\bar{A}) = \cup_\alpha p^\alpha 0^M(\bar{A}),$$

where

$$p^0X^M(\bar{A}) \quad = X^M(\bar{A})$$
$$p^{\alpha+1}X^M(\bar{A}) = p^M(p^\alpha X^M(\bar{A}),\bar{A})$$
$$p^\lambda X^M(\bar{A}) \quad = \cup_{\beta<\lambda} p^\beta X^M(\bar{A}) \quad , \lambda \text{ a limit ordinal.}$$

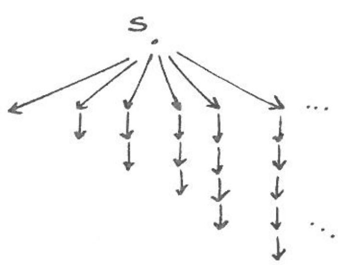
The equivalence of these two definitions is the Knaster-Tarski theorem. $X^M(\bar{Q})$ can also be interpreted as a subset of S^M by taking

$$X^M = X^M(Q_1^M,\ldots,Q_k^M).$$

We write $s \models X$ and say s <u>satisfies</u> X if $s \in X^M$. The infinitary formulas $p^\alpha X$ can be represented physically, modulo the representation of ordinals, by associating the ordinal α with the root of pX, along with a two-way pointer to each occurrence of X. There is no ambiguity, provided X is free for Q in pQ. We will use these formulas and this representation in the algorithm.

3. Expressiveness Results

$L\mu$ subsumes PDL (without reverse), as shown by Pratt [Pr]. For example, the PDL formula $\langle a^* \rangle X$ is given by $\mu Q.X \vee \langle a \rangle Q$. However, unlike PDL, there are monotone operators that are not continuous: $[a]Q$ is one. If pQ is continuous in Q in the model M, then $\mu Q.pQ^M \equiv p^\omega 0^M$, but in any model, $\mu Q.[a]Q = \{s \mid$ there are no infinite a-paths out of s$\}$. (This is $\neg\Delta a$ in the notation of Streett [S]). For example, in the model pictured, $\mu Q.[a]Q \equiv p^{\omega+1}0$ but the top state s satisfies $\mu Q.[a]Q - p^\omega 0$. The question raised by Pratt about the strict expressiveness of $P\mu$ over PDL is still open, but the following result of Street shows that $L\mu$ is strictly more expressive, and in a way which shows why filtration techniques fail for $L\mu$:



Proposition (Streett [S]). $\mu Q.[a]Q$ is not equivalent to any PDL formula.

Proof Suppose $\mu Q.[a]Q \equiv X \in$ PDL. Consider the model pictured above. Then $s \models \mu Q.[a]Q$ and $s \models X$. However, in any finite filtrate over a set containing X, the equivalence class [s] of s still satisfies X, but cannot satisfy $\mu Q.[a]Q$ since there is an infinite a-path out of it. □

The above proof assumes $\mu Q.[a]Q \equiv X$ in all models and derives a contradiction. However we can show that $L\mu$ is strictly more expressive than PDL in the stronger sense that there is a model M and a formula X of $L\mu$ such that no PDL formula Y is equivalent to X on M.

Proposition In the model

the formula $\mu Q.[a]\langle a\rangle Q$ defines the even states, whereas all PDL formulas, even with test and reverse, define only finite and cofinite sets. □

The proof is omitted. Intuitively, PDL cannot simulate an unbounded alternation of $[a]$ and $\langle a\rangle$.

4. A Deductive System

The deductive system is equational, as in [KP]. All formulas in a deduction are of the form $X \equiv Y$ or $X \leq Y$, the latter abbreviating $X \vee Y \equiv Y$. The logical axioms and rules are those for equational logic, including substitution of equals for equals, provided the syntactic restrictions on μ formulas are not violated. The nonlogical axioms and rules are:

(4.1) axioms for Boolean algebra

(4.2) $\langle a\rangle X \vee \langle a\rangle Y \equiv \langle a\rangle(X \vee Y)$

(4.3) $\langle a\rangle X \wedge [a]Y \leq \langle a\rangle(X \wedge Y)$

(4.4) $\langle a\rangle 0 \equiv 0$

(4.5) $p(\mu Q.pQ) \leq \mu Q.pQ$

(4.6) $\dfrac{pX \leq X}{\mu Q.pQ \leq X}$, X free for Q in pQ

(4.1)-4.4) are axioms of propositional modal logic. (4.5) and (4.6) say that $\mu Q.pQ$ is the $\leq$-least object X such that $pX \leq X$. (4.6) is the fixpoint induction rule of Park [Pa].

The following are some basic theorems and derived rules of this system. We refer the reader to [dB,dR] for omitted proofs.

Proposition 4.7. The following are provable:

(i) (change of bound variable) $\mu Q.pQ \equiv \mu P.pP$, provided neither Q nor P occurs in pR.

(ii) (monotonicity) $\dfrac{X \leq Y}{pX \leq pY}$, X,Y free for Q in pQ

(iii) $\mu Q.X \equiv X$, Q not free in X

(iv) $p(\sigma Q.pQ) \equiv \sigma Q.pQ$

(v) $\dfrac{pQ \leq qQ}{\sigma Q.pQ \leq \sigma Q.qQ}$

(vi) $\dfrac{p(\mu Q.X \wedge pQ) \leq X,}{\mu Q.pQ \leq X}$ Q not free in X, X free for Q

(vii) $\dfrac{p(X \wedge \mu Q.pQ) \leq X,}{\mu Q.pQ \leq X}$ X free for Q

Proof(vi).

(a)	$p(\mu Q.X \wedge pQ) \leq X$	(assumption)
(b)	$X \wedge p(\mu Q.X \wedge pQ) \leq X$	(a),(4.1)
(c)	$p(X \wedge \mu Q.X \wedge pQ) \leq X$	(a),(4.1),(ii)
(d)	$p(X \wedge \mu Q.(X \wedge p(X \wedge Q))) \leq X$	(c),(ii),(v)
(e)	$p(X \wedge \mu Q.(X \wedge p(X \wedge Q))) \leq X \wedge p(X \wedge 0 Q.(X \wedge p(X \wedge Q)))$	(d),(4.1)
(f)	$p(X \wedge \mu Q.(X \wedge p(X \wedge Q))) \leq \mu Q.(X \wedge p(X \wedge Q)))$	(e),(4.5)
(g)	$p(X \wedge \mu Q.(X \wedge p(X\ Q))) \leq X \wedge \mu Q.(X \wedge p(X \wedge Q))$	(d),(f)
(h)	$\mu Q.pQ \leq X \wedge \mu Q.(X \wedge p(X \wedge Q))$	(g),(4.6)
(i)	$\mu Q.pQ \leq X$	(h),(4.1). □

4.7(vi) says that if $X \wedge \mu Q.pQ$ is consistent, then $X \wedge p(\mu Q.\neg X \wedge pQ)$ is; intuitively, the iteration $p^{\alpha}0$ has to capture a state of X for the first time.

5. Main Results

In this section we show completeness of the axioms and give an exponential time decision procedure and small model property. These results are based on a common construction which can be described as a tableau or semantic tree method. Similar methods have been used in program logics by Pratt [Pr1] and Emerson and Clarke [EC].

Let W be a positive formula of $L\mu$. Assume by 4.7(i) that no variable is bound twice and no primitive proposition occurs both bound and free. The closure cl(W) is the smallest set containing W and closed under subformula and the rule $\sigma Q.pQ \in cl(W) \rightarrow p(\sigma Q.pQ) \in cl(W)$. cl(W) is no larger than $|W|$, because each element is e(X) for some subformula X of W, where e(X) is obtained from X by repeatedly replacing the variables Q by the unique subformula $\sigma Q.pQ$ of W. The order of replacement does not matter, the process must halt since the only new variables introduced with $\sigma Q.pQ$ are quantified in a subformula of W containing $\sigma Q.pQ$ properly. We distinguish two kinds of σ-subformulas of $e(X) \in cl(W)$: those that have been <u>regenerated</u> and those that have not. The former are those that replaced a variable in the above construction of $e(X)$.

To construct a tableau T for W, start with the root r_T labeled $\Phi_{r_T} = \{W\}$, and apply the following <u>extension</u> <u>rules</u>:

$\wedge$-rule : if $X \wedge Y \in \Phi_s$ then add node t labeled $\Phi_t = \Phi_s \cup \{X,Y\}$ and unlabeled edge $s \to t$.

$\vee$-rule : if $X \vee Y \in \Phi_s$ then add nodes t,u with $\Phi_t = \Phi_s \cup \{X\}$, $\Phi_u = \Phi_s \cup \{Y\}$, and unlabeled edges $s \to u$, $s \to t$.

σ-rule : if $\sigma Q.pQ \in \Phi_s$, add t with $\Phi_t = \Phi_s \cup \{p(\sigma Q.pQ)\}$ and unlabeled edge $s \to t$.

$\diamond$-rule: for each $\langle a \rangle X \in \Phi_s$, add node t with $\Phi_t = \{X\} \cup \{Y \mid [a]Y \in \Phi_s\}$ and edge $s \to t$ labeled a.

The $\wedge$-, $\vee$-, σ-rules are applied until Φ_s does not grow, then the $\diamond$-rule is applied. This must occur eventually since each $\Phi_s \subseteq cl(W)$. Thus there are at most $2^{|W|}$ distinct labels. Call s a <u>strong node</u> if either no rule applies to s or the $\diamond$- rule was applied; a <u>weak node</u> otherwise. The <u>weak interval</u> of s consists of s and all weak ancestors on the path back up to, but not including, the next strong node. A <u>regeneration</u> of a σ-formula is an application of the σ-rule to it. For every occurrence of a subformula X of a formula in Φ_s, its <u>trace</u> is the sequence of occurrences of X stemming from that occurrence of X in Φ_s. For example, if Z is an occurrence of a subformula of X in $X \wedge Y \in \Phi_s$, and the $\wedge$-rule is applied to $X \wedge Y$, the corresponding occurrence of Z in X in the successor of s is on the same trace. A trace can be duplicated at applications of the σ-rule, since there can be several occurrences of the same formula stemming from one occurrence. If the same formula from two different sources appears in Φ_s, then one copy is discarded and the traces of corresponding subformulas merged.

The syntactic condition (2.1) implies that no element of cl(W) contains two conjunctively related occurrences of the same μ-formula. This is proved by induction on the generation of cl(W). Let $\mu Q.pQ$ be an occurrence of a μ-formula in T. Follow its trace until it is exposed and regenerated. There is now one copy of $\mu Q.pQ$ in $p(\mu Q.pQ)$ for each Q in pQ.

Follow the trace of each one of these; by the syntactic requirement (2.1), they must split apart and go down different branches of the tree before one is regenerated. In fact, any two occurrences of $\mu Q.pQ$ in different formulas of Φ_s must have disjoint traces back up to their first regenerations.

We now give an alternating Turing machine algorithm to refute W by constructing the tableau and rejecting if certain conditions are met. The algorithm starts with one process at the root r_T with input $\Phi_{r_T} = \{W\}$. It then constructs the tableau by applying the $\vee$-, $\wedge$-, and σ-rules in a regular fashion; at the application of $\vee$-rules, it makes an existential branch, spawning two processes, each of which takes one of the successors. At the application of $\diamond$-rules it makes a universal branch, spawning a process for each successor. If a process at a strong node s finds both $P, \bar{P} \in \Phi_s$, it rejects. If not, and if no rules apply (for example if Φ_s contains only formulas P and [a]X) then it accepts.

Besides this action, each occurrence of a μ-formula in Φ_s has a <u>priority</u> (position in a priority queue) and a <u>count</u>. The μ-formula goes onto the back of the queue when it is first regenerated, and its count is set to 0. Whenever it is regenerated, its count is incremented and all counts of formulas of lower priority are reset to 0. When two occurrences X,X' of a formula appear in Φ_s, they must be merged. This is done as follows: Let $\mu Q_1.p_1Q_1, \ldots, \mu Q_n.p_nQ_n$ be the regenerated μ-formulas of X in order of decreasing height of $e^{-1}(\mu Q_i.p_iQ_i)$ in W. Note that the $e^{-1}(\mu Q_i.p_iQ_i)$ are linearly ordered by the subformula relation. Also note that all occurrences of $\mu Q_i.p_iQ_i$ have the same count, and that they will have gone down two different branches of the tree by the time they become exposed again; thus there is only need to represent each $\mu Q_i.p_iQ_i$ once in the priority queue. Thus each $\mu Q_i.p_iQ_i$ appears once on the queue and in the order $\mu Q_1.p_1Q_1, \ldots, \mu Q_n.p_nQ_n$, since $\mu Q_1.p_1Q_1$ must have been regenerated first. Let $\mu Q_1.p_1Q_1', \ldots, \mu Q_n.p_nQ_n'$ be the corresponding occurrences in X'. To merge, keep the sequence of lexicographically higher priority (say $\mu Q_1.p_1Q_1, \ldots$) and delete the other one. Think of the sequence $\mu Q_1.p_1Q_1', \ldots$ as merging <u>into</u> $\mu Q_1.p_1Q_1, \ldots$ If any count ever exceeds $2^{|W|}$, the process rejects. We claim

(5.1) there can be no trace of $\mu Q.pQ$ with more than $|W|^2 \cdot 2^{|W|^3}$ regenerations before rejecting.

An element of the queue can only change priority, by merging with something of lexicographically higher priority or something being deleted in front of it, at most $|W|^2$ times, the maximum length of the queue. Every time its count is set back, either it was because something of higher priority was regenerated, or its priority changed, thus either its lexicographic priority increased or something with a higher priority had its count increased. Thus some counter must run out after $|W|^2 \cdot 2^{|W|^3}$ steps. The condition (2.1) was used to insure that new copies of a μ-formula come into the back of the queue; this is exactly what fails in the general case.

The set Φ_s can be represented by a set of pebbles on subterms of W, a pebble on X denoting $e(X) \in \Phi_s$. We also need to maintain the priority queue with $|W|^2$ counters, each holding an integer value $\leq 2^{|W|}$. Thus the algorithm uses alternating $O(|W|^3)$ space, which, despite the possibility of infinite computations, can be simulated in deterministic exponential time [CKS].

<u>Theorem 5.2</u>. The following are equivalent:

(i) W is consistent ;

(ii) the algorithm does not reject ;

(iii) W has a finite tree-like model of depth $d = |W|^2 \cdot 2^{|W|^3}$ and 2^d states.

<u>Proof</u>. (i)$\rightarrow$(ii) Suppose W is consistent. In the tableau T, replace each occurrence $\mu Q.pQ$ in Φ_s by a formula $\mu Q.R \wedge pQ$ inductively down the tree, to get Φ'_s. R is a conjunction of k formulas, where k is the count of $\mu Q.pQ$ at s. If k=0, R=1; otherwise, let t_0 be the most recent time in $\mu Q.pQ$'s history that its counter was 0, and let $t_0, t_1, \ldots, t_{k-1}$ be all the nodes along its trace up to s at which $\mu Q.pQ$ was regenerated; let $R = \bigwedge_{i=0}^{k-1} \neg \Phi''_{t_i}$, where Φ''_t is obtained from Φ'_t by deleting all R's from the lower priority μ-formulas.

We construct a set C of nodes containing r_T such that

(a) if $s \in C$ and s is an $\vee$-node, then some successor of s is in C;

(b) for all other nodes $s \in C$, all successors of s are in C ;

(c) every Φ'_s, $s \in C$, is consistent.

We start by setting $C = \{r_T\}$; W is consistent by assumption. If Φ'_s is consistent and the $\vee$-rule is applied at s to $X \vee Y$, then one of the successors Φ'_t, Φ'_u must be consistent by (4.1) since Φ'_t is $\Phi'_s \cup \{X\}$ with perhaps some formulas deleted due to merging; no counts change. Thus one of t,u can be added to C. Similarly, at applications of the $\wedge$-, $\diamond$-, and ν-rule, C can be extended with all successors (there is no duplication of traces in the ν-rule, due to 2.1), since if Φ_s is consistent then all its successors are, by (4.1)-(4.4) and 4.7(iv). At applications of the μ-rule to $\mu Q.R\wedge pQ$, we need to show that if $\mu Q.R\wedge pQ \in \Phi'_s$ and Φ'_s is consistent, then $p(\mu Q.(R\wedge\neg\Phi''_s\wedge pQ))\wedge\Phi''_s$ is consistent, where Φ''_s is obtained from Φ'_s by deleting the R's in all lower priority subterms, whose counts were reset to 0. But $\Phi'_s\leq\Phi''_s$ by monotonicity, thus if $\mu Q\wedge(R\wedge pQ)\wedge\Phi'_s$ is consistent, then $\mu Q.(R\wedge pQ)\wedge\Phi''_s$ is, and thus $p(\mu Q.(R\wedge\neg\Phi''_s\wedge pQ))\wedge\Phi''_s$ is by 4.7(vi). Thus the set C exists. Moreover, C does not contain any node rejected because of $P,\bar{P}\in\Phi_s$, since all nodes of C are consistent; and any nodes rejected because a counter ran out must have an occurrence of $\mu Q.pQ$ regenerated at two ancestors s an t with $\Phi_s=\Phi_t$, the priority of $\mu Q.pQ$ unchanged and its count nonzero on the path from s down to t. Then $\mu Q.R\wedge pQ\in\Phi'_t$ and R contains $\neg\Phi''_s$, thus $\Phi'_t\leq\neg\Phi''_s$; and $\Phi'_t\leq\Phi''_t\leq\Phi''_s$, since $\Phi_t=\Phi_s$, and all μ-formulas with lower priority have R=1 in both Φ''_t and Φ''_s, and all higher priority μ-formulas were not changed between s and t (otherwise $\mu Q.pQ$'s count would have been reset). Thus Φ'_t is inconsistent and cannot be in C, and neither can any descendant.

Thus the algorithm cannot reject, because the computation tree is isomorphic to the tableau, and the set C forms a barrier to the computation of 0's back up the tree, so 0 cannot be assigned to the root.

(ii)$\rightarrow$(iii) If the algorithm does not reject, prune all nodes in the tableau corresponding to nodes in the computation tree labeled 0; the set so obtained satisfies (a) and (b) above, and contains the root. Moreover no trace in T regenerates $\mu Q.pQ$ more than $|W|^2\cdot 2^{|W|^3}$ times, by (5.1).

Let Φ'_s be Φ_s with all μ-formulas $\mu Q.pQ$ replaced by $p^{d-c}0$, where $d = |W|^2\cdot 2^{|W|^3}$, and c is the maximum number of times $\mu Q.pQ$ is regenerated on any path out of s. Define the model $T = (S^T,\rho^T,\pi^T)$ where S^T = {strong nodes in C}, ρ^T: $a\mapsto a^T = \{(s,t)\,|\,s\rightarrow u$ in T on some edge labeled a, and u is in the weak interval of t$\}$; and π^T: $P\mapsto P^T = \{s\,|\,P\in\Phi_s\}$. Let $X^L = \{s\in S^T\,|\,X\in\Phi'_s\}$. We show by induction on formula structure that

$$(5.3)\qquad p(\bar{X})^L \subseteq p^T(\bar{X}^L),$$

where $\bar{X}^L = X^L_1,\ldots,X^L_n$.

Then $r_T \models W$ follows by specializing $\bar{X}$ to the null sequence and p=W. The basis is given by

$$P(\bar{X})^L = P^L = P^T$$
$$\bar{P}(\bar{X})^L = \bar{P}^L \subset \bar{P}^T$$
$$Q_i(\bar{X})^L = X^L_i = Q^T_i(\bar{X}^L),$$

the first two by definition of π^T. The induction cases $\wedge$ and $\vee$ are straightforward; for $\langle a\rangle$, if $s \in \langle a\rangle p(\bar{X})^{L}$ then $(s,t)\in a^{T}$ for some $t \in p(\bar{X})^{L}$ by definition of the tableau; Then $t \in p^{T}(\bar{X}^{L})$ by induction hypothesis, so $s \in \langle a^{T}\rangle p^{T}(\bar{X}^{L}) = \langle a\rangle p^{T}(\bar{X}^{t})$. If $s \in [a]p(\bar{X})^{L}$ then all $(s,t)\in a^{T}$ have $t \in p^{T}(\bar{X}^{L})$, so $s \in [a]p^{T}(\bar{X}^{L})$. Occurrences of $\mu Q.pQ$ were replaced by $p^{n}0$; for this case,

$$\begin{aligned} p^{n}0(\bar{X})^{L} &\subseteq \cup_{m<n}\ p(p^{m}0(\bar{X}),\bar{X})^{L} \\ &\subseteq \cup_{m<n}\ p^{T}(p^{m}0(\bar{X})^{L},\bar{X}^{L}) \\ &\subseteq \cup_{m<n}\ p^{T}(p^{m}0^{T}(\bar{X}^{L}),\bar{X}^{L}) \\ &\subseteq \qquad p^{n}0^{T}(\bar{X}^{L}) \end{aligned}$$

The first step is by the μ-rule in the generation of the tableau, and the other steps are by induction hypothesis. Finally,

$$\begin{aligned} \nu Q.pQ(\bar{X})^{L} &\subseteq p(\nu Q.pQ(\bar{X}),\bar{X})^{L} \\ &\subseteq p^{T}(\nu Q.pQ(\bar{X})^{L},\bar{X}^{L}) \end{aligned}$$

by induction hypothesis; since $\nu Q.pQ^{T}(\bar{X}^{L})$ is the greatest fixpoint,

$$\nu Q.pQ(\bar{X})^{L} \subseteq \nu Q.pQ^{T}(\bar{X}^{L}).$$

A small model is obtained from T by observing that the tree T is regular, because the tableau rules were applied in a regular fashion, thus there are at most d distinct subtrees up to isomorphism, and hence d distinct theories, since the theory of a node depends only on its subtree (this is false in the presence of the reverse operator). Thus a finite model can be formed by creating loops.

(iii)$\rightarrow$(i) This asserts the soundness of the deductive system and is left to the reader. □

6. Algebraic Semantics and a Representation Theorem

One can give an algebraic semantics whose models are Boolean algebras with operators $\langle a\rangle,\mu$ satisfying the axioms (4.1)-(4.6). This is the approach taken in [Pr]. Over this semantics, completeness is obtained easily by constructing a Lindenbaum algebra from formulas. In this case the completeness theorem of Section 5 can be considered a proof of equivalence between the two semantics. Moreover, every algebraic model is algebraically isomorphic to a nonstandard state model, by the Stone construction (see [K1]); that is,

$$\begin{aligned} \text{states} &= \{\text{ultrafilters}\},\quad P^{M} = \{u \mid P\in u\}, \\ a^{M} &= \{(u,v) \mid \forall X\ X\in v\rightarrow\langle a\rangle X\in u\} \\ &= \{(u,v) \mid \forall X\ [a]X\in u\rightarrow X\in v\}. \end{aligned}$$

The construction insures that $\mu Q.pQ^M$ is the least element of the algebra closed under p^M and that $\cup_\alpha p^\alpha 0^M \subseteq \mu Q.pQ^M$, but equality does not hold in general. Define an algebra to be <u>μ-complete</u> if the $\leq$-supremum $\bigvee_\alpha p^\alpha 0^M$ exists and is equal to $\mu Q.pQ^M$. This property corresponds to *-continuity in dynamic algebras [K3]. Then the set $\mu Q.pQ^M - \cup_\alpha p^\alpha 0^M$ is nowhere dense in the nonstandard representation of the algebra constructed above, thus in countable algebras, the union of all such sets is meager and can be deleted without changing the algebra (see [K2]). In this way we have shown

<u>Theorem</u>. Every countable μ-complete algebra is isomorphic to a standard model.

Moreover, all standard models are μ-complete. Thus μ-completeness characterizes the countable standard models up to isomorphism.

<u>References</u>

[HP] P. Hitchcock and D.M.R. Park, Induction Rules and Termination Proofs, *Proc. 1st ICALP*, 1973, 225-251.

[KP] D. Kozen and R. Parikh, An Elementary Proof of the Completeness of PDL, *TCS 14* (1981), 113-118.

[K1] D. Kozen, A Representation Theorem for Models of *-free PDL, *Proc. 7th ICALP*, Springer LNCS 85 (1980), 352-362.

[K2] D. Kozen, On the Duality of Dynamic Algebras and Kripke Models, *Proc. Workshop on Logic of Programs*, Springer LNCS 125, 1-11.

[K3] D. Kozen, On Induction vs. *-continuity, *Proc. Workshop on Logics of Programs*, Springer LNCS 131 (1982), 167-176.

[Pa] D.M.R. Park, Fixpoint Induction and proof of program semantics, *Mach. Int. 5*, ed. Meltzer and Michie, Edinburgh Univ. Press, 1970, 59-78.

[Pr] V.R. Pratt, A Decidable μ-calculus (Preliminary Report), *Proc. 22nd FOCS*, Oct. 1981, 421-427.

[Pr1] V.R. Pratt, A near optimal method for reasoning about action, *JCSS 20* (1980), 231-254.

[SdB] D. Scott and J. deBakker, *A Theory of Programs*, unpublished, IBM, Vienna, 1969.

[S] R. Streett, Propositional Dynamic Logic of Looping and Converse, *Proc. 13th STOC*, May 1981, 375-383.

[CKS] A. Chandra, D. Kozen, L. Stockmeyer, *Alternation*, JACM, Jan. 1981.

[dB] J. deBakker, *Mathematical Theory of Program Correctness*, Prentice-Hall, 1980.

[dBR] J. deBakker and W. deRoever, A Calculus for Recursive Program Schemes, *Proc. 1st ICALP*, 1973, 167-196.

[dR] W.P. deRoever, *Recursive Program Schemes: Semantics and Proof Theory*, Ph.D. Thesis, Free University, Amsterdam, 1974.

[EC] E.A. Emerson and E.M. Clarke, Design and Synthesis of Synchronization Skeletons using Branching-Time Temporal Logic, *Proc. Workshop on Logics of Programs*, Springer LNCS 131 (1982), 52-71.

[FL] M. Fischer and R. Ladner, Propositional Dynamic Logic of Regular Programs, *JCSS 18:2* (79).

[HR] J. Halpern and J. Reif, The Propositional Dynamic Logic of Deterministic, Well-Structured Programs, (extended abstract), *Proc. 22nd FOCS*, Oct. 1981, 322-334.