Various logical languages are compared regarding their expressive power with respect to models consisting of finitely bounded branching infinite trees. The basic multimodal logic with backward- and forward necessity operators is equivalent to restricted first order logic; by adding the binary temporal operators "since" and "until" we get the expressive power of first order logic on trees. Hence (restricted) propositional quantifiers in temporal logic correspond to (restricted) set quantifiers in predicate logic. Adding the CTL* path modality "E" to temporal logic gives the expressive power of path logic. Tree grammar operators give a logic as expressive as weak second order logic, whereas adding fixed point quantifiers (in the so-called propositional μ-calculus) results in a logic expressively equivalent to monadic second order logic on trees.

Keywords: Modal logic, temporal logic, branching time logic, computation tree logic, CTL*, propositional μ-calculus, Lμ, definability, expressiveness, expressive completeness, ω-tree automata, ω-trees, ω-tree languages, specification languages

0 Introduction

ω-trees arise in various areas of logic and computer science. Therefore there have been many different approaches to specify sets of ω-trees: via first and second order logic, tree automata, term rewriting systems, and modal and temporal logics. In this paper we consider logics related to branching time temporal logic. However, unlike the usual branching time logic, we include several “nexttime” operators into the logic, one for each successor relation in the tree. In branching time logics the tree structure is intended to model the nondeterministic behaviour of a program; therefore in these logics one can not distinguish different subtrees which look alike. In many contexts however it is important whether a node has only one child or twin children. Also the relative order of the children may be of some interest. We regard as “reference logic” the predicate logic containing interpreted binary successor predicates \( S_1, \ldots, S_b \), where \( 0 < b < \omega \) is the branching factor of the underlying tree structures. The classical papers [Rab69],[Rab70] showed that monadic second order logic with \( b \) successors is as expressive as Rabin tree automata, and that the weak second order definable tree languages are exactly those which are Büchi tree

*In this extended abstract all proofs are omitted; they appear in the full version of the paper

†this work was partly done when the author was on leave at Carnegie-Mellon-University, Pittsburgh, PA; it was supported by DFG grant Schl 310/1-1
automaton and complement Büchi tree automaton definable. Hafer and Thomas [HaTh88] gave expressive completeness results for branching time logics with respect to path logic and chain logic with signature \((S, S^*)\). Here we relate the following logics to predicate logic with signature \((S_1, \ldots, S_b, S^*)\):

- basic multimodal logic with necessity operators \([S_i], [S^*], [S_i^*], [S^*]\]
- temporal logic with additional binary operators \(S\) (since) and \(U\) (until)
- temporal logic with (restricted) propositional quantifiers
- directed computation tree logic \(DCTL^*\) with path modalities \(E\) and \(A\)
- extended temporal logic with tree grammar operators
- propositional \(\mu\)-calculus, i.e. multimodal logic with fixed point quantifiers

We give a uniform semantics of these logics in terms of second order logic, and compare their expressive power to certain fragments of monadic second order logic. It turns out that the first of the above logics corresponds to relativized first order logic, the second to first order logic, the third to (restricted) set quantification, the fourth to path logic, the fifth to weak second order, and the last to full monadic second order logic.

1 Definitions and Results

Definition 1 Let \(0 < b < \omega\) be a natural number. An \(\omega\)-tree is a prefix closed subset \(B \subseteq \{1, \ldots, b\}^*\) of nodes.

A labelled \(\omega\)-tree \((B, \eta)\) is an \(\omega\)-tree \(B\) together with a labelling function \(\eta : B \rightarrow 2^\mathcal{P}\), where \(2^\mathcal{P}\) is a finite powerset alphabet, i.e. \(\mathcal{P} = \{p_1, \ldots, p_n\}\), and \(2^\mathcal{P}\) is the set of all subsets of \(\mathcal{P}\).

An \(\omega\)-tree model \((B, \eta, x)\) is a labelled \(\omega\)-tree \((B, \eta)\) and a current node \(x \in B\). An \(\omega\)-tree language is a set of \(\omega\)-tree models.

Definition 2 Let \(\mathcal{S} = \{x_1, x_2, \ldots\}\) be a countable set of individual variables and \(\mathcal{P}' = \{p_1, \ldots, p_n, q_1, q_2, \ldots\}\) be a countable set of monadic predicate signs. The monadic second order logic of \(b\) successors, \(\mathcal{S}b\mathcal{S}\), is the smallest set of formulas such that \(\bot \in \mathcal{S}b\mathcal{S}\), \(p(x) \in \mathcal{S}b\mathcal{S}\) for all \(x \in \mathcal{S}\), \(p \in \mathcal{P}'\), \(xS_1y \in \mathcal{S}b\mathcal{S}\) for all \(x, y \in \mathcal{S}\) and \(1 \leq i \leq b\), and if \(F\) and \(F'\) \(\in \mathcal{S}b\mathcal{S}\) for all \(x \in \mathcal{S}\) and \(q \in \mathcal{P}'\), then \((F \rightarrow F')\), \(\exists x(F')\) and \(\exists q(F) \in \mathcal{S}b\mathcal{S}\) for all \(x \in \mathcal{S}\) and \(q \in \mathcal{P}'\).

An \(\mathcal{S}b\mathcal{S}\) sentence is a \(\mathcal{S}b\mathcal{S}\) formula in which there occurs at most one free individual variable \(x\) and all free predicate signs are among \(p_1, \ldots, p_n\).

Other connectives \(\neg, \land, \lor, \forall, \exists\) are defined as usual, as well as other relations:

\[
\begin{align*}
xS_1y & : \leftrightarrow (xS_1y \lor \ldots \lor xS_by) \\
xS^*_y & : \leftrightarrow \forall q(q(x) \land \forall z_1z_2(q(z_1) \land z_1Sz_2 \rightarrow q(z_2)) \rightarrow q(y)) \\
x = y & : \leftrightarrow (xS^*_y \land yS^*_x) \\
xS^+_y & : \leftrightarrow \neg(xS^*_y \rightarrow x = y) \\
xS^{-}_i y & : \leftrightarrow (yS_i x) \quad \text{etc.}
\end{align*}
\]