Second International Symposium on Logical Foundations of Computer Science (2 nd LFCS '92)
Tver, Russia, July 1992,
A. Nerode, M. Taitslin (eds.), Springer LNCS 620, pp. 441 – 451

ON THE EXPRESSIVE POWER OF MODAL LOGICS ON TREES

BERND–HOLGER SCHLINGLOFF

Institut für Informatik der Technischen Universität München
Orleansstr. 34, D—81667 München, Germany

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 mu-calculus) results in a logic expressivly 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 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 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 $\mathcal{S}$ (**since**) and $\mathcal{U}$ (**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 μ–calculus, i.e. multimodal logic with weakest and greatest 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 ω–tree is a prefix closed subset $B \subset \{1, \ldots, b\}^*$ of nodes.
A labelled ω–tree (B, η) is an ω–tree B together with a labelling function $\eta : B \mapsto 2^{\mathcal{P}}$, 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 ω–tree model (B, η, x) is a labelled ω–tree (B, η) and a current node $x \in B$. An ω–tree language is a set of ω–tree models.

Definition 2 Let $\Im = \{x_1, x_2, \ldots\}$ 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, **SbS** , is the smallest set of formulas such that $\bot \in$ **SbS** , $p(x) \in$ **SbS** for all $x \in \Im$, $p \in \mathcal{P}'$, $xS_iy \in$ **SbS** for all $x, y \in \Im$ and $1 \leq i \leq b$, and if F and $F' \in$ **SbS** , then $(F \rightarrow F')$, $\exists x(F)$ and $\exists q(F) \in$ **SbS** for all $x \in \Im$ and $q \in \mathcal{P}'$.
An **SbS** sentence is a **SbS** 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 $\neg, \wedge, \vee, \top, \forall x, \forall q$ are defined as usual, as well as other relations:

$$\begin{aligned}
xSy &:\leftrightarrow (xS_1y \vee \ldots \vee xS_by)\\
xS^*y &:\leftrightarrow \forall q\big(q(x) \wedge \forall z_1z_2(q(z_1) \wedge z_1Sz_2 \rightarrow q(z_2)) \rightarrow q(y)\big)\\
x = y &:\leftrightarrow (xS^*y \wedge yS^*x)\\
xS^+y &:\leftrightarrow \neg(xS^*y \rightarrow x = y)\\
xS_i^-y &:\leftrightarrow (yS_ix) \qquad \text{etc.}
\end{aligned}$$

Definition 3 The weak second order logic of b successors is defined like **SbS** , but with weak quantification $\exists^f q(F)$ on finite predicates instead of $\exists q(F)$. Path logic contains the additional relation S^*, with second order quantification restricted to path quantifiers $\exists^p q(F)$. A path is a maximal set of nodes which are pairwise comparable w.r.t. S^*, i.e.

$$\exists^p q(F) :\leftrightarrow \exists q\Big(F \wedge \forall y\big(q(y) \leftrightarrow \forall z(q(z) \rightarrow yS^*z \vee zS^*y)\big)\Big)$$

First order logic of $(S_1, \ldots, S_b, S^*)$ is similar, but without any set quantification. Relativized first order logic uses the relations $R \in \{S_i, S^*\}$ only immediately after quantifiers, i.e. in expressions like $\exists x(xRy \wedge F)$ or $\exists x(yRx \wedge F)$.

Note that e.g. $\exists y(y \neq x \wedge p(y))$ is equivalent to the relativized sentence

$$\bigvee_i \exists yS_i^-x\exists zS^{*-}y(p(z)) \vee \exists yS^*x \bigvee_i \exists zS_iy\big(p(z) \vee \bigvee_{j\neq i} \exists z'S_j^-z\exists z''S^{*-}z'(p(z''))\big)$$

Definition 4 The turnstyle relation $M \models F$ gives a truth value to every logical sentence F in a model M. Note that xS_iy iff $y = x \cdot i$ for nodes x, y. The ω–tree language defined by F is the set of models M such that $M \models F$.

Example 5 The set of all trees such that on every path through the current node there are only finitely many nodes labelled p is definable by the path logic sentence

$$\varphi_1 \equiv \forall^p q \Big(q(x) \rightarrow \exists y \big(x S^* y \wedge q(y) \wedge \forall z (z S^+ y \wedge q(z) \rightarrow \neg p(z)) \big) \Big)$$

This set is not definable in weak second order logic.

The set of all trees having only finitely many nodes labelled p below the current node all of which occur in an even distance from it is defined by the weak second order sentence

$$\varphi_2 \equiv \exists^f q \Big(q(x) \wedge \forall y \Big(x S^* y \rightarrow \Big((p(y) \rightarrow q(y)) \wedge \forall z \big(y S z \rightarrow (q(y) \leftrightarrow \neg q(z)) \big) \Big) \Big) \Big)$$

This set is not path logic definable.

The set of all trees such that there is a finite branch from the current node on which exactly one node (but not the last node of this branch) is labelled p can be defined by the first order sentence

$$\varphi_3 \equiv \exists y \big(x S^+ y \wedge \neg p(y) \wedge \exists! z (x S^+ z \wedge z S^+ y \wedge p(z)) \big),$$

where $\exists! z (F(z)) :\leftrightarrow \exists z \big(F(z) \wedge \forall z' (F(z') \rightarrow z = z') \big)$.
This set is not definable in restricted first order logic.

Definition 6 The basic multimodal logic is built from propositional variables $p \in \mathcal{P}$ with connectives $\bot$, $(F_1 \rightarrow F_2)$ and $\langle R \rangle F$, where $R \in \{S_1, \ldots, S_b, S^*, S_1^-, \ldots, S_b^-, S^{*-}\}$.

Its semantics is given via a translation τ from modal formulas to predicate logic formulas:

$$(\bot)^\tau := \bot, \quad (p)^\tau := p(x), \quad \big((F_1 \rightarrow F_2)\big)^\tau := (F_1^\tau \rightarrow F_2^\tau),$$
$$(\langle R \rangle F)^\tau := \exists y (x R y \wedge F^\tau (x := y))$$

where $F^\tau(x := y)$ denotes the result of substituting y for every free occurrence of x in F^τ. (Whenever this substitution causes name conflicts we assume consistent renaming of bound variables.)

The necessity operators $[R]$ can be introduced as abbreviations:

$[R]F \leftrightarrow \neg \langle R \rangle \neg F$ with the semantics

$$([R]F)^\tau := \forall y (x R y \rightarrow F^\tau (x := y))$$

Fact 7 Modal logic is expressively equivalent to relativized first order logic; i.e. the translation of a modal formula yields a relativized first order sentence, and for every relativized first order sentence there exists a modal formula which defines the same language.

So for example the formula corresponding to $\exists y (y \neq x \wedge p(y))$ can be written as $\bigvee_i \langle S_i \rangle \langle S^* \rangle p \vee \langle S^{*-} \rangle \bigvee_i \langle S_i^- \rangle (p \vee \bigvee_{j \neq i} \langle S_j \rangle \langle S^* \rangle p)$.

Definition 8 Temporal logic is obtained from multimodal logic by adding the new two place connectives $\mathcal{U}$ (**until**) and $\mathcal{S}$ (**since**) with the semantics

$$\big(\mathcal{U}(F_1, F_2)\big)^\tau := \exists y \big(x S^+ y \wedge F_1^\tau (x := y) \wedge \forall z (x S^+ z \wedge z S^+ y \rightarrow F_2^\tau (x := z)) \big)$$
$$\big(\mathcal{S}(F_1, F_2)\big)^\tau := \exists y \big(y S^+ x \wedge F_1^\tau (x := y) \wedge \forall z (y S^+ z \wedge z S^+ x \rightarrow F_2^\tau (x := z)) \big)$$

In temporal logic e.g. the third language of example 5 can be characterized by $\mathcal{U}(p \wedge \mathcal{U}(\neg p, \neg p), \neg p)$. The operators $\langle S^* \rangle$ resp. $\langle S^{*-} \rangle$ as well as $\langle S \rangle$ and all $\langle S_i \rangle$ are expressible with $\mathcal{U}$, $\mathcal{S}$ and $\langle S_i^- \rangle$:

$$\langle S^* \rangle p \leftrightarrow p \vee \mathcal{U}(p, \top)$$
$$\langle S^{*-} \rangle p \leftrightarrow p \vee \mathcal{S}(p, \top)$$
$$\langle S \rangle p :\leftrightarrow \mathcal{U}(p, \bot)$$
$$\langle S_i \rangle p \leftrightarrow \langle S \rangle (p \wedge \langle S_i^- \rangle \top)$$

Also the two place connectives $\mathcal{U}$ and $\mathcal{S}$ can be replaced by the one place connectives which are defined by the sentence φ_3 of example 5 and its converse.

It is immediate that the translation of a temporal formula yields a first order sentence. The converse of this inclusion also holds:

Theorem 9 To every first order sentence there exists a temporal formula which defines the same ω–tree language.

The proof of this result is rather intricate and involves difficult transformations on both the predicate logic and the temporal logic side. It relies heavily on the fact that the underlying structures are trees, i.e. for every node there is a unique predecessor node. As a corollary we get that first order logic with only the S^* relation (without specialized successor relations S_i) is expressively equivalent to the temporal logic with operators $\mathcal{U}$, $\mathcal{S}$, and $\langle k \star S\rangle$, where

$$(\langle k \star S\rangle F)^\tau := \exists y_1, ..., y_k \bigwedge_k (xSy_k \wedge \bigwedge_{i \neq k} y_i \neq y_k \wedge F^\tau(x_k)).$$

Definition 10 (Restricted) quantified temporal logic is obtained by adding propositional quantifiers $\exists q$ resp. $\exists^f q$ or $\exists^p q$ to the temporal logic language. Its translation is defined by

$$\big(\exists q(F)\big)^\tau := \exists q(F^\tau),$$

and similarly for $\exists^f$ and $\exists^p$.

Obviously the translation of a temporal formula with quantification $\exists$ ($\exists^f$, $\exists^p$) gives a sentence of monadic second order logic (weak second order logic, path logic). Theorem 9 also gives the induction basis for the other direction:

Fact 11 For every monadic second order (weak second order, path) logic sentence there is an appropriate quantified temporal logic formula.

As examples the languages φ_1 and φ_2 can be defined by

$$\varphi_1 \leftrightarrow \forall^p q(q \rightarrow \langle S^*\rangle(q \wedge [S^+](q \rightarrow \neg p)))$$

$$\varphi_2 \leftrightarrow \exists^f q\Big(q \wedge [S^*]\big((p \rightarrow q) \wedge (q \rightarrow [S]\neg q) \wedge (\neg q \rightarrow [S]q)\big)\Big)$$

We note that the operators $\mathcal{U}$ and $\mathcal{S}$ (and hence all other operators considered so far) can be expressed with (weak) propositional quantifiers and modalities $\langle U\rangle$, $\langle S_i^-\rangle$:

$$\langle S\rangle F \leftrightarrow \forall^f q\big(q \rightarrow \langle U\rangle(\bigvee_i \langle S_i^-\rangle q \wedge F)\big)$$

$$\mathcal{U}(F_1, F_2) \leftrightarrow \exists^f q\Big(q \wedge [U]\big(q \rightarrow \langle S\rangle(F_1 \vee F_2 \wedge q)\big)\Big)$$

$$\mathcal{S}(F_1, F_2) \leftrightarrow \exists^f q\Big(q \wedge [U]\big(q \rightarrow \bigvee_i \langle S_i^-\rangle(F_1 \vee F_2 \wedge q)\big)\Big)$$

Therefore these modalities form a minimal basis for (weak) quantified temporal logic. In path quantified temporal logic however $\mathcal{U}$ and $\mathcal{S}$ are not dispensable! (The situation is similar to predicate logic, where S^* is definable from S by weak quantification but not by path quantification.)

Definition 12 Directed computation tree logic, DCTL*, is the logic which arises by adding the "path modality" E to temporal logic. EF means that there is a path through the current node on which the linear time formula F holds. The semantics can be given as follows:

$$(EF)^\tau := \exists^p q\big(q(x) \wedge \delta(F^\tau)\big)$$

where $\delta(F^\tau)$ is obtained from F^τ by replacing every first order quantification $\exists y(F')$ not in the scope of another path quantifier with $\exists y(q(y) \wedge F')$.

The well–known logic CTL* [CES83, EH83] is the subset of DCTL* using only $\mathcal{U}$ and E, i.e. without $\mathcal{S}$ and $\langle S_i^-\rangle$.
If $A := \neg E \neg$ denotes the dual modality to E, then e.g.

$$\varphi_1 \leftrightarrow A\langle S^*\rangle[S^+]\neg p$$

From the semantics it is clear that DCTL* is at most as expressive as path logic. The following theorem is due to [HaTh88]:

Theorem 13 For every DCTL* formula there is a path logic sentence defining the same language.

By transformation of path quantified temporal logic formulae we obtain a proof of this theorem as a consequence of theorem 9.

The close connection between automata theory and logic is well known. Therefore one could as well think about specification languages which consist of a mixture of logical formulas with transition graphs. Wolper [Wol83] was the first to incorporate grammar operators into (linear time) temporal logic, and Muller, Saoudi and Schupp [MSS88] suggested to add operators based on alternating tree automata into branching time logic. Whereas these automata have an acceptance condition which is any Borel set, Wolpers original grammar operators do not refer to any acceptance condition. We therefore want to investigate the expressive power of tree grammar operators based on b–branching transition systems (without acceptance condition) in propositional logic.

To simplify things we give a labelling for the the nodes of the transition systems rather than for the arcs; this is not a loss of generality, since we allow a finite set of initial states.

Definition 14 A transition system $\Gamma = (\Sigma, \Pi, \Delta)$ is a set of states $\Sigma = \{q_1, \ldots, q_n, t, f\}$, a set of initial states $\Delta \subset \Sigma$, and a transition relation $\Pi \subset \Sigma^{b+1}$.

The state f labels "nonexisting nodes". If we only considered full trees state f would be unneccessary. State t will later on mean that a node exists but no further condition is imposed on this t–labelled node. This state is included, because we only want to deal with total transition systems (from every state there is at least one transition leading from that state). States t and f together are intended to complete the system; we always assume $\{t, f\}^{b+1} \subset \Pi$.

Let (B, η, x_0) be a tree model and let $\xi : \{1, \ldots, b\}^* \mapsto \Sigma$ be a function from nodes to states such that $\xi(x) = f$ iff $x \notin B$ for all $x \in \{1, \ldots, b\}^*$. Then (B, η, x_0, ξ) is generated by the transition system Γ, if

- $\xi(x_0) \in \Delta$
- $\big(\xi(x), \xi(x \cdot 1), \ldots, \xi(x \cdot b)\big) \in \Pi$ for all $x \in \{1, \ldots, b\}^*$.

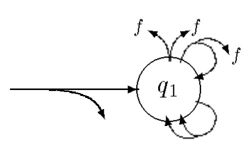
Hence our transition systems generate trees in both directions, upward and downward. This —in spite of being more 'natural'— will be necessary to express the pasttime operators $\mathcal{S}$ and $\langle S_i^- \rangle$ via transition systems.

For example let $b = 2$ and let Γ_1 be the grammar with

$$\Pi = \{(f,f,f),(t,f,f),(f,t,f),(f,f,t),(t,t,f),(t,f,t),(f,t,t),(t,t,t),\\ (t,q_1,t),(t,q_1,f),(t,t,q_1),(t,f,q_1),(q_1,f,f),(q_1,q_1,f),(q_1,f,q_1),(q_1,q_1,q_1)\}.$$

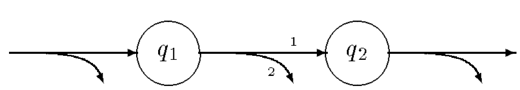
This grammar can be graphically represented as follows:

Γ_1:



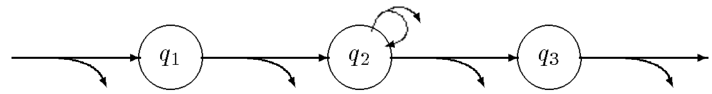
Γ_1 generates all trees in which all nodes below the current node are labelled by q_1.

Γ_2:



With $\Delta = \{q_1\}$ Γ_2 generates all trees in which the current node has a S_1–successor node labelled q_2; with $\Delta = \{q_2\}$ Γ_2 generates all trees on which either the current node is the root or it has a S_1–predecessor labelled q_1.

Γ_3:



If $\Delta = \{q_1\}$ then Γ_3 generates all trees on which there is either an infinite branch labelled q_2 starting in the current node, or a finite branch starting in q_1 and ending in q_3 such that all nodes in between are labelled q_2.

Every transition system Γ represents a grammar operator in the following sense: If $q_1, \ldots, q_n$ are all states except t, f of Γ and if $p_1, \ldots, p_n$ are propositional variables, then $\Gamma(p_1, \ldots, p_n)$ defines the set of all trees (B, η, x) such that there is a ξ for which (B, η, x, ξ) is generated by Γ, and for all $y \in B$ with $\xi(y) = q_i$ is $p_i \in \eta(y)$.

Definition 15 Extended temporal logic is built from propositional variables $p \in \mathcal{P}$ with boolean connectives $\bot$, $(F_1 \to F_2)$ and $\Gamma(F_1, \ldots, F_n)$, where Γ is any transition system with state set $\{q_1, \ldots, q_n, t, f\}$.

The semantics —informally described above— can also be given via a translation into monadic second order logic:

$$\begin{aligned}\big(\Gamma(F_1, \ldots, F_n)\big)^\tau := \exists q_1, \ldots, q_n, t\Big(&\textstyle\bigvee_{q\in\Delta} q(x) \wedge \\ &\forall y \textstyle\bigvee_i \big((q_i(y) \vee t(y)) \wedge \bigwedge_{i\neq j} \neg(q_i(y) \wedge q_j(y))\big) \wedge \\ &\forall y \textstyle\bigwedge_i (q_i(y) \to F_i^\tau(x := y)) \wedge \\ &\forall y, y_1, \ldots, y_b \big(\textstyle\bigwedge_i yS_i y_i \vee \neg\exists z(yS_i z) \to \\ &\textstyle\bigvee_{(s,s_1,\ldots,s_b)\in\Pi} (s'(y) \wedge s'_1(y_1) \wedge \ldots \wedge s'_b(y_b))\big)\Big)\end{aligned}$$

The first line of this sentence states that the current node is labelled by some initial state, the second line says that every node is labelled by exactly one state, the third line tells us that a node labelled q_i satisfies F_i, and the last lines describe the transition relation. The disjunction ranges over all b–tuples $(s, s_1, \ldots, s_b) \in \Pi$, where $s'_i(y_i)$ stands for $s_i(y_i)$, if $s_i \notin \{t, f\}$, and $t'(y_i) := \top$, and $f'(y_i) := \neg y S_i y_i$ (resp. $f'(y) := \bot$).

It is easy to verify that all temporal operators introduced so far are expressible in extended temporal logic, hence extended temporal logic is at least as expressive as temporal logic.

Extended temporal logic formulas are related to the subtree automata defined in [VaWo86]. There it is shown that any subtree automaton can be simulated by a Büchi automaton and hence subtree automata are at most as expressive as existential quantified weak second order sentences. A similar statement holds for extended temporal logic: For every transition system Γ there is a dual transition system Γ' which generates all trees different from trees generated by Γ. Since this difference appears in finite distance from the current node, the complement language of $\Gamma(F_1, \ldots, F_n)$ can be characterized with weak quantifiers. A recent construction by Arnold and Niwinsky [ArNi] can be used to prove also the converse direction; hence

Theorem 16 Extended temporal logic is as expressive as weak second order logic.

The last modal logic for tree languages we look at is the so–called propositional μ–calculus of [Koz82, KP83]. Already classical finite automata can be regarded as generating least fixed points of linear recursion equations. In multimodal logic the following recursion equations are valid:

$$[S^*]F \leftrightarrow F \wedge [S][S^*]F$$
$$\mathcal{U}(F_1, F_2) \leftrightarrow \langle S\rangle(F_1 \vee F_2 \wedge \mathcal{U}(F_1, F_2))$$

(compare also the definition of $\mathcal{U}$ in weak quantified temporal logic!). Therefore $[S^*]F$ is the greatest fixed point of the equation $q \leftrightarrow F \wedge [S]q$, and $\mathcal{U}(F_1, F_2)$ is the least fixed point of the equation $q \leftrightarrow \langle S\rangle(F_1 \vee F_2 \wedge q)$. Here a point is a set of nodes, and the lattice ordering is the subset relation. This can be written as

$$[S^*]F \leftrightarrow \nu q(F \wedge [S]q)$$
$$\mathcal{U}(F_1, F_2) \leftrightarrow \mu q\big(\langle S\rangle(F_1 \vee F_2 \wedge q)\big)$$

The formal description is given by the following

Definition 17 The propositional μ–calculus is obtained from modal logic with modalities $\langle S_i\rangle$, $\langle S_1^-\rangle$ by adding fixed point quantification νq. $\mu q(F)$ is defined as $\neg\nu q'(\neg F(q := \neg q'))$. The semantics of $\nu q(F)$ is

$$\big(\nu q(f)\big)^\tau := \exists q\big(q(x)\big) \wedge \forall y(q(y) \rightarrow F^\tau(x := y))\big)$$

E.g. the language φ_1 from example 5 can be written as

$$\varphi_1 \leftrightarrow \mu q\big([S^*](p \rightarrow q)\big)$$

and φ_2 as

$$\varphi_2 \leftrightarrow \mu q\big((\langle S^+\rangle p \rightarrow [S](\neg p \wedge [S]q)\big)$$

Sometimes the propositional μ–calculus is defined with multiple fixed point equations $\nu q_1 ... q_n(F_1, \ldots, F_n)$. This does not increase the expressive power, since every formula with multiple fixed points can be reduced to a single fixed point formula. (The reduction procedure resembles of the construction of a regular expression from a finite automaton.)

Clearly every extended temporal logic formula corresponds to a μ–calculus formula, since grammar operators define (multiple) greatest fixed points of transition systems. Niwinsky [Niw88] extended a construction of Park [Par81] to tree models and shows that also the liveness condition of Rabin tree automata can be expressed by an appropriate nesting of greatest and least fixed points. We can apply a similar construction to the calculus defined here; hence by the above reduction and Rabins theorem we know

Theorem 18 The propositional μ–calculus is as expressive as monadic second order logic.

2 Summary

In our research we have obtained the following hierarchy of logics on trees:

Restricted first order logic $(S_i, S^*, \exists yRx)$
$= $ Multimodal logic $(\langle S_i\rangle, \langle S^*\rangle, \langle S_i^-\rangle, \langle S^{*-}\rangle)$

First order logic $(S_i, S^*, \exists y)$
$= $ Temporal logic $(\mathcal{U}, \mathcal{S}, \langle S_i^-\rangle)$

Path logic $(S_i, S^*, \exists y, \exists^p q)$
$= $ Path quantified temporal logic
$(\mathcal{U}, \mathcal{S}, \langle S_i^-\rangle, \exists^p q)$
$= $ Directed computation tree logic
$(\mathcal{U}, \mathcal{S}, \langle S_i^-\rangle, E)$

Weak second order logic $(S_i, \exists y, \exists^f q)$
$= $ Weak quantified temporal logic
$(\langle U\rangle, \langle S_i^-\rangle, \exists^f q) = $ Extended temporal logic (Γ)

Monadic second order logic $(S_i, \exists y, \exists q)$
$= $ Quantified temporal logic $(\langle U\rangle, \langle S_i^-\rangle, \exists q)$
$= $ Propositional μ–calculus $(\langle S_i\rangle, \langle S_i^-\rangle, \nu q)$

References

[ArNi] A. Arnold, D. Niwinsky: *Fixed point characterization of weak monadic second order logic of trees*; preprint, (Okt.18,1990)

[CES83] E.M. Clarke, E.A. Emerson, A.P. Sistla: *Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic Specifications*; in: Proc. 10^{th} ACM POPL, pp.117–126; Austin (1983).
reappeared in: ACM Trans. Prog. Lang. & Syst. **8**,2 pp.244–263 (1986).

[Cle90] R. Cleaveland: *Tableau-Based Model Checking in the Propositional Mu-Calculus*; Acta Informatica **27**, pp.725–747; (1990)

[EH83] E.A. Emerson, J.Y. Halpern: *"Sometimes" and "Not Never" Revisited: On Branching versus Linear Time (preliminary report)*; in: Proc. 10^{th} ACM POPL, pp.127–140; Austin (1983).
reappeared in: JACM **33**,1 pp.151–178 (1986).

[HaTh88] T. Hafer, W. Thomas: *Computation Tree Logic CTL* and Path Quantifiers in the Monadic Theory of the Binary Tree*; in: T. Ottmann (ed.): ICALP 1987, Springer LNCS **267**, (1988).
Short version of T. Hafer: *On the Expressive Completeness of CTL**; Bericht Nr. 123, Schriften zur Informatik, RWTH Aachen, (Dez. 1986).

[Heu89] U. Heuter: *Zur Klassifizierung regulärer Baumsprachen*; Dissertation, RWTH Aachen, (1989).

[Koz82] D. Kozen: *Results on the Propositional μ–Calculus*; 9th ICALP, LNCS , pp. 348–359, (1983).

[KP83] D. Kozen, R. Parikh: *A Decision Procedure for the Propositional μ–Calculus*; Logics of Programs, LNCS **164**, pp. 313–325, (1983).

[MSS88] D. Muller, A. Saoudi, P. Schupp: *Alternating Automata Give a Simple Explanation of Why Most Temporal and Dynamic Logics Are Decidable in Exponential Time*; in: *Proc. 3rd IEEE Symp. on Logic in Computer Science*; pp. 422–427 (1988).

[Niw88] D. Niwinsky: *Fixed Points vs. Infinite Generation*; 3^{rd} IEEE LICS, pp. 402–409, (1988).

[Par81] D. Park: *Concurrency an Automata on Infinite Sequences*; 5^{th} GI Conf. on TCS, LNCS **104**, pp. 167–183, (1981).

[Rab69] M.O. Rabin: *Decidability of Second–Order Theories and Automata on Infinite Trees*; Trans. AMS **141**, pp.1–35; (1969).

[Rab70] M.O. RABIN: *Weakly Definable Relations and Special Automata*; in: Y. Bar–Hillel (ed.): Mathematical Logic and Foundations of Set Theory, pp.1–23; North–Holland, Amsterdam (1970).

[Sch90] B.-H. SCHLINGLOFF: *Zur temporalen Logik von Bäumen*; Dissertation, Institut für Informatik der Technischen Universität München, Report TUM–I9012; (1990).

[SE89] R. STREETT, E. A. EMERSON: *An Automata Theoretic Decision Procedure for the Propositional Mu–Calculus*; Information and Computation **81**, pp.249–264; (1989).

[VaWo86] M. VARDI, P. WOLPER: *Automata–Theoretic Techniques for Modal Logics of Programs*; Journ. of Comp. and Syst. Sci. **32**, pp. 183–221 (1986).

[Tak86] M. TAKAHASHI: *The Greatest Fixed–Points and Ratinal Omega–Tree Languages*; TCS **44**, pp.259–274, (1986).

[Wol83] P. WOLPER: *Temporal Logic Can Be More Expressive*; Information and Control *56*, pp. 72–99, (1983).