Tree Automata, Mu-Calculus and Determinacy
(Extended Abstract)

E.A. Emerson[1] and C.S. Jutla[2]

1. The University of Texas at Austin
2. IBM, T.J. Watson Research Center

Abstract

We show that the propositional Mu-Calculus is equivalent in expressive power to finite automata on infinite trees. Since complementation is trivial in the Mu-Calculus, our equivalence provides a radically simplified, alternative proof of Rabin's complementation lemma for tree automata, which is the heart of one of the deepest decidability results. We also show how Mu-Calculus can be used to establish determinacy of infinite games used in earlier proofs of complementation lemma, and certain games used in the theory of on-line algorithms.

1 Introduction

We propose the propositional Mu-calculus as a uniform framework for understanding and simplifying the important and technically challenging areas of automata on infinite trees and determinacy of infinite games. We show that the Mu-Calculus is precisely equivalent to tree automata in expressive power, permitting a radically simplified proof of the Complementation Lemma for tree automata. We also show how to systematically prove the determinacy of certain infinite games, including games in the theory of on-line algorithms.

Rabin (1969) introduced tree automata (finite state automata on infinite trees) to prove the decidability of the monadic second order theory of n successors (SnS). This is one of the most fundamental decidability results, to which many other decidability results in logic, mathematics, and computer science can be reduced. The proof involves reducing satisfiability of SnS formulae to the nonemptiness problem of tree automata. The reduction entails showing that tree automata are closed under disjunction, projection, and complementation. While the first two are rather easy, the proof of Rabin's Complementation Lemma is extraordinarily complex and difficult. Because of the importance of the Complementation Lemma, a number of authors have endeavored (and continue to endeavor) to simplify the argument ([HR72], [GH82], [MS84], [Mu84].) Perhaps the best known of these is the important work of Gurevich and Harrington [GH82] which attacks the problem from the standpoint of determinacy of infinite games. While the presentation is brief, the argument is still extremely difficult, and is probably best appreciated when accompanied by the 40 page supplement of Monk [Mon].

In this paper, we present a new, enormously simplified proof of the Complementation Lemma. We would argue that the new proof is straightforward and natural. To obtain our proof, we show that the propositional Mu-Calculus ([Ko83]) is expressively equivalent to (nondeterministic) tree automata. Since the Mu-Calculus is trivially closed under complementation, the equivalence immediately implies that tree automata are also closed under complementation.

We remark that the equivalence of the Mu-calculus and tree automata is a new result of some independent interest, since it shows that there indeed exists a natural modal logic of programs precisely equivalent to tree automata in expressive power (cf. [Th88]). Actually, a brief argument establishing this equivalence can be given, by appealing to known lengthy arguments translating through SnS. However, a careful examination of this argument reveals that repeated appeals to the Complementation Lemma are involved.

Thus this argument is not at all suitable for use in a proof of the Complementation Lemma. The relationship of the Mu-Calculus and tree automata has also been investigated previously by Niwinski. In [Ni88] it is shown that tree automata can be translated into the Mu-Calculus, while in [Ni86] it is shown that a restricted Mu-Calculus, in which conjunctions are not allowed, can be translated into tree automata. As we shall see, in our translation of the full Mu-Calculus into tree automata, the main difficulty lies in dealing with the conjuncts.

The Mu-Calculus $L\mu$ is a modal logic (we deal here with the propositional, Temporal version) with fixpoint operators. It provides a *least fixpoint operator* (μ), and a *greatest fixpoint operator* (ν), which makes it possible to give extremal fixpoint characterizations of the branching time modalities. Intuitively, the Mu-Calculus makes it possible to characterize the modalities in terms of recursively defined tree-like patterns. First, we show that the parse tree of a formula of $L\mu$ can be viewed as an alternating Tree Automaton [MS84], giving a translation from $L\mu$ to alternating Tree Automata. The problem of translating the Mu-Calculus into nondeterministic Tree Automata then reduces to translating alternating Tree Automata to nondeterministic Tree Automata.

The alternating Tree Automata obtained from the Mu-Calculus, however, have a nice property in the sense that if there is an accepting run of an automaton $\mathcal{A}$ on an input tree t, then there is a *history-free* accepting run of $\mathcal{A}$ on t. A run of $\mathcal{A}$ on t is *history-free* iff the non-deterministic choices made by $\mathcal{A}$ along any path of t depend only on the current state (i.e. the choices are independent of the history of the different "forall"-runs).

We show how these history-free, alternating Tree Automata can be reduced to nondeterministic Tree Automata using constructions for determinizing[Sa88] and complementing [EJ89] automata on infinite strings. (For general alternating tree automata, we can still do the translation into nondeterministic tree automata, but now we must use Gurevich and Harrington's forgetful strategy theorem [GH82]; the history-freedom of the automata derived from the Mu-Calculus allows us to avoid the appeal to [GH82] in our translation.) That proves the equivalence of nondeterministic Tree Automata and the Mu-Calculus, and hence the Complementation Lemma.

Almost all published proofs of the Complementation Lemma ([Ra69],[Bu77,83], [GH82]) can be seen upon reflection to have taken the following approach, best brought out by [MS87]:

Alternating Tree Automata are easy to complement, given the fact that (1) *certain infinite games are determined*- i.e. at least one of the players has a winning strategy. Nondeterministic Tree Automata, clearly, are alternating Tree Automata. Alternating Tree Automata can be reduced to nondeterministic Automata (as shown in this paper) using determinization of ω-Automata and (2) *a forgetful strategy theorem.*

Theorems (1) and (2) in all the results mentioned above have had difficult proofs. Although [GH82] have somewhat simplified the proofs over [Ra69], the proofs are far from being transparent. Moreover, their proofs of Theorem (1) and (2) are intertwined. The new proof of the Complementation Lemma via Mu-Calculus suggests the following for simplifying the above approach, i.e. (1) and (2), as well: To prove (1), extremal fixpoints (Mu-calculus) can be used to characterize games in which a particular Player has a winning strategy. Indeed, in section 4, we give simple and straightforward Mu-calculus characterizations of games in which Players I and II have winning strategies, which turn out to be trivial complements of each other! Moreover, a totally independent ranking argument provides a simple proof of (2).

Finally, we also show that the above technique of extremal fixpoint (Mu-calculus) characterization gives a general method for proving various determinacy results. Wolfe (1955) (cf [Mo80]) proved the determinacy of infinite games in the second level of the Borel hierarchy (F_σ). We show that it is possible to view such determinacy results as implicitly constructing extremal fixpoints. Mu-Calculus characterization also gives a clear and simple proof of determinacy of specialized F_σ games used in de-randomizing (existentially) On-line algorithms [RS90]. This makes the result in [RS90] only slightly more complicated (being more general) than a determinacy result in [B90].

Detailed proofs of Theorems in section 2 and 3 appear in the second author's dissertation [J90].

2 Alternating Tree Automata to Nondeterministic Tree Automata

We deal here only with finite state Automata.

Definition 2.1: A *nondeterministic binary Tree Automaton* is a tuple $\mathcal{N} = (\Sigma,\ OR,\ AND,\ \psi, \delta,\ L,\ s_0,\ \Omega)$, where
Σ is the input alphabet,
OR is the finite set of states,
AND is a set of nodes, intuitively corresponding to entries in the transition function of the Automaton,
$L : AND \rightarrow \Sigma$, is a labelling of the AND nodes with the input symbols,
ψ is a relation on $OR \times AND$ defining the different nondeterministic choices from OR to AND nodes, essentially the transition function.
$\delta : AND \times \{0,1\} \rightarrow OR$, is a function specifying left and right successor states,
$s_0 \in OR$ is the start state,
Ω is the acceptance condition, to be defined below.

We now give a game theoretic definition of acceptance of a tree t by an Automaton $\mathcal{N}$.

Definition 2.2: Given an input tree $t : \{0,1\}^* \rightarrow \Sigma$, we can define a 2 *player infinite game* $\Gamma(\mathcal{N}, t)$ on the above bipartite graph as follows. Player I picks AND successors of OR nodes picked by Player II, and vice versa.

Formally, a *strategy for Player I* is a map $\rho_I : \{0,1\}^* \rightarrow AND$ with the requirement that $\forall x \in \{0,1\}^* : L(\rho_I(x)) = t(x)$. We say that ρ_I is a *legal strategy for I* iff $\psi(s_0, \rho_I(\lambda))$ and $\forall b_0 b_1 ... b_k \in \{0,1\}^{k+1} : \psi(\delta(\rho_I(b_0 b_1 ... b_{k-1}), b_k), \rho_I(b_0 b_1 ... b_k))$. We need the notion of "legal" to ensure that I indeed picks AND nodes which are successors of previously picked OR-nodes. The class of legal strategies of Player I will be called $Strat_I$. A *strategy for Player II* is a map $\rho_{II} : AND^+ \rightarrow \{0,1\}$, defined similarly.

In a play, given $b = b_0 b_1 ... b_k$ (i.e. the moves of Player II) and a legal strategy ρ_I, there is a unique sequence of OR nodes $s_0 s_1 ... s_{k+1}$ visited along b in the play, which we define by a function $path : Strat_I \times \{0,1\}^* \rightarrow OR^*$.

Similarly, given $b \in \{0,1\}^\omega$, and a legal strategy ρ_I, there is also a unique infinite sequence of OR nodes visited along b, if Player I follows ρ_I. Thus, we define a function $ipath : Strat_I \times \{0,1\}^\omega \rightarrow OR^\omega$, with $ipath(\rho_I, b) = s_0 s_1 ... \in OR^\omega$ (where $b = b_0 b_1 ...$), such that $s_1 = \delta(\rho_I(\lambda), b_0)$ and, $\forall k > 1,\ s_k = \delta(\rho_I(b_0 b_1 ... b_{k-2}), b_{k-1})$. Note that $\{path(\rho_I, b') | b'$ prefix of $b\}$ is exactly the set of non-empty finite prefixes of $ipath(\rho_I, b)$.

The acceptance condition Ω is given by a subset of OR^ω. We say that $x \in OR^\omega$ *satisfies* Ω iff x is in the subset representing Ω. Usually, Ω is defined by a Temporal formula interpretted over infinite sequences of OR nodes. For example, the Streett acceptance condition has m pairs of subsets of OR, $\{(GREEN_1, RED_i), ...(GREEN_m, RED_m)\}$. Then the subset of OR^ω defining Ω is given by $\bigwedge_{i \in [1..m]}(GF\ GREEN_i \Rightarrow GF\ RED_i)$. In *Temporal Logic*, G stands for everywhere, and F stands for eventually. The Pairs acceptance condition is the complement of the Streett condition.

Definition 2.3: A legal strategy ρ_I for I is a *winning strategy for I* iff

$\forall b \in \{0,1\}^\omega$, $ipath(\rho_I, b)$ satisfies Ω.

We say that $\mathcal{N}$ *accepts* t iff I has a winning strategy in $\Gamma(\mathcal{N}, t)$. $\mathcal{L}(\mathcal{N}) = \{t | \mathcal{N}$ accepts tree $t\}$, is the language accepted by $\mathcal{N}$.

Non-deterministic Automata and TMs were generalized to Alternating Automata and TMs in [CKS81]. Alternating tree Automata were first defined in [MS84]. The following definition of Alternating Tree Automaton is the generalization of the definition of nondeterministic Tree Automaton defined above, to include "universal states". Intuitively, we now have two kinds of AND nodes: $DAND$ and $AAND$. The $DAND$ nodes are as before, the directional AND nodes with two successors specifying the transitions in the left and right directions. The $AAND$ nodes are the dual of the OR nodes representing the $\forall$-nondeterminism.

Definition 2.4: An *Alternating binary Tree Automa-*

ton is a tuple $\mathcal{A} = (\Sigma, OR, DAND, AAND, \psi, \delta, \chi, L, s_0, \Phi)$, where
Σ is the input alphabet,
OR is the finite set of states,
$DAND$ is a set of nodes, with successors specifying the transitions in the different directions,
$AAND$ is a set of nodes, with successors specifying the different $\forall$-nondeterministic transitions,
$\psi \subseteq OR \times DAND$, defines the different nondeterministic choices from the OR nodes,
$\delta : DAND \times \{0,1\} \rightarrow AAND$ is a function defining the transitions along the different directions,
$\chi \subseteq AAND \times OR$, defines the different $\forall$- nondeterministic transitions,
$L : DAND \rightarrow \Sigma$, is a labelling of $DAND$ nodes with input symbols,
$s_0 \in OR$ is the start state,
Φ is the Acceptance Condition, to be defined later.

We now give a game theoretic definition of acceptance of a tree t by an Automaton $\mathcal{A}$.

Definition 2.5: Given $t : \{0,1\}^* \rightarrow \Sigma$, a 2 *player infinite game* $\Gamma(\mathcal{A}, t)$ can be defined as for Nondeterministic Automata. Player I picks a node from $DAND$, while Player II picks a pair: a binary digit defining an $AAND$ node and then an OR node successor of the resulting $AAND$ node. Thus Player II has additional choice. It not only picks a path $b \in \{0,1\}^\omega$, it also picks a $\forall$-nondeterministic sequence of transitions.

Thus a *strategy for Player I* is a map $\rho_I : \{s_0\} \cdot (\{0,1\} \cdot OR)^* \rightarrow DAND$ with the requirement that $\forall b\hat{\ }s \in (\{0,1\} \cdot OR)^*$, $L(\rho_I(s_0 \cdot b\hat{\ }s)) = t(b)$. We say that ρ_I is *legal* iff $\forall b\hat{\ }s \in (\{0,1\} \cdot OR)^*$, $\psi(s_k, \rho_I(s_0 \cdot b\hat{\ }s))$ where $s = s_1..s_k$. The class of strategies of Player I will be called $AStrat_I$. A *strategy for player II* is a map $\rho_{II} : DAND^+ \rightarrow \{0,1\} \cdot OR$.

In contrast to the games for Nondeterministic Automata, given $b \in \{0,1\}^\omega$, and a legal strategy ρ_I, there is a set of sequences of OR nodes (rather than a unique sequence of OR nodes). The different sequences in the set correspond to different $\forall$-nondeterministic choices of Player II. Player I's strategy ρ_I now, is a winning strategy iff for all $b \in \{0,1\}^\omega$, all the resulting infinite sequences of OR nodes along b satisfy the acceptance condition.

Given $b \in \{0,1\}^*$, and a legal strategy ρ_I, we now define a function *bundle*, which gives the set of sequences of OR nodes. Each sequence in the bundle will be referred to as a *thread*. Thus $bundle : AStrat_I \times \{0,1\}^* \rightarrow 2^{OR^*}$, such that

$bundle(\rho_I, \lambda) = \{s_0\}$,

$bundle(\rho_I, b_0) = \{s_0 s_1 | \chi(\delta(\rho_I(s_0), b_0), s_1)\}$,

$bundle(\rho_I, b_0 b_1 ... b_k) = \{s_0 s_1 ... s_k s_{k+1} |\ s_0 s_1 ... s_k \in bundle(b_0..b_{k-1})$ and $\chi(\delta(\rho_I(s_0 b_0..b_{k-1} s_k), b_k), s_{k+1})\}$.

Similarly, we define a function $ibundle : AStrat_I \times \{0,1\}^\omega \rightarrow 2^{OR^\omega}$, such that $ibundle(\rho_I, b_0 b_1 ...) = \{s_0 s_1 ... | \forall k > 0\ s_0 s_1 ... s_k \in bundle(\rho_I, b_0 b_1 ... b_{k-1})\}$. Each element of $ibundle(\rho_I, b)$ will be referred to as *ithread*. Note that *ibundle* has Automaton $\mathcal{A}$ and tree t as implicit arguments.

As for Nondeterministic Tree Automata, the acceptance condition Φ is given by a subset of OR^ω. We say that $x \in OR^\omega$ *satisfies* Φ iff x is in the subset representing Φ. Once again, we may specify Φ using Temporal Logic.

Definition 2.6: A legal strategy ρ_I for Player I is a *winning strategy* for Player I iff

$\forall b \in \{0,1\}^\omega, \forall s \in ibundle(\rho_I, b) : s$ satisfies Φ.

We say that $\mathcal{A}$ *accepts* t iff I has a winning strategy in $\Gamma(\mathcal{A}, t)$. $\mathcal{L}(\mathcal{A}) = \{t |\ \mathcal{A}$ accepts tree $t\}$ is the language accepted by $\mathcal{A}$.

Definition 2.7: A strategy ρ_I for Player I in a game $\Gamma(\mathcal{A}, t)$ is a *History-free Strategy* iff at each node of the input tree, the strategy depends only on the current state of the Automaton $\mathcal{A}$ and the node in the tree t. Formally, iff

$\forall k \geq 0, \forall b \in \{0,1\}^k, \forall s = s_0 s_1 ... s_k \in OR^{k+1}, \forall s' = s'_0 s'_1 ... s'_k \in OR^{k+1} \quad s_k = s'_k \Rightarrow \rho_I(s\hat{\ }b) = \rho_I(s'\hat{\ }b)$

An Alternating Automaton $\mathcal{A}$ is a *History-free Alternating Automaton* iff for every tree t, if Player I has a winning strategy in $\Gamma(\mathcal{A}, t)$, then Player I has a history-free winning strategy in $\Gamma(\mathcal{A}, t)$.

Theorem 2.1 [GH82]: If Player I has a winning strategy in $\Gamma(\mathcal{A}, t)$ then it has a forgetful winning strategy in $\Gamma(\mathcal{A}, t)$, i.e. a strategy which only depends on a small finite history of the play, where $\mathcal{A}$ is defined with Muller Acceptance condition. □

2.1 Construction

Given an Alternating Automaton $\mathcal{A}$ the obligation is to construct a nondeterministic Tree Automaton $\mathcal{N}$, such that for every tree t, Player I has a winning strategy in $\Gamma(\mathcal{A}, t)$ iff Player I has a winning strategy in $\Gamma(\mathcal{N}, t)$. A legal strategy ρ_I in $\Gamma(\mathcal{A}, t)$ generates for each $b \in \{0,1\}^\omega$, an $ibundle(\rho_I, b)$ of infinite *ithreads*, whereas a legal strategy ρ_I in $\Gamma(\mathcal{N}, t)$ generates for each $b \in \{0,1\}^\omega$, a single thread, $ipath(\rho_I, b)$.

Thus, $\mathcal{N}$ must collect all the *ithreads* in *ibundle* into a single *ipath* (and if *ibundle* is generated by a finite state mechanism then the *ipath* must also be generated by a finite state mechanism). Safra's construction [Sa88], e.g. is a mechanism which generates a single *ipath* satisfying Streett acceptance condition iff all the *ithreads* satisfy the complement of the Buchi acceptance condition. A modification of the co-Safra Construction [EJ89] generates a single *ipath* satisfying pairs acceptance condition iff all *ithreads* satisfy the Streett acceptance condition, which is what we require here.

There are two main points to be noted. Firstly, $\mathcal{N}$ must at each OR state have a transition function, which is the cross product of the transition functions of all the last nodes of the *threads* it is collecting into a single *path*. Secondly, since the number of *threads* increase arbitrarily with increasing length, if $\mathcal{N}$ were to collect all the *ithreads*, $\mathcal{N}$ would require infinitely many states. However, by Theorem 2.1, every Streett Alternating tree Automaton is forgetful, and hence $\mathcal{N}$ only needs to keep the small finite history of the *threads*, thus requiring only finitely many states. Of course, for history-free Alternating Automaton, we do not need theorem 2.1.

Theorem 2.2: Given $\mathcal{A} = (OR,\ DA,\ AA,\ \psi,\ \delta,\ \chi,\ L,\ s_0,\ \Sigma,\ \Phi)$, a Streett alternating Tree Automaton, we construct an equivalent nondeterministic Pairs Tree Automaton $\mathcal{N} = (OR',\ AND',\ \psi',\ \delta',\ L',\ s_0',\ \Sigma,\ \Omega)$, i.e. $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{N})$. If $|OR| = n$, $|DA| = m$, and Φ has k pairs then $|OR'| = 2^{k^2 2^{O(nlogn)}}$, $|AND'| = 2^{k^2 2^{O(nlogn)}} \cdot m^{2^{O(nlogn)}}$, with Ω having $k2^{O(nlogn)}$ pairs.

3 Propositional Mu-Calculus to Tree Automata

Using the Construction in the previous section we translate Propositional Mu-Calculus to Rabin Tree Automata. We begin by defining the Temporal version of Propositional Mu-Calculus. Following that we convert a Mu-Calculus formula to an equivalent Streett history-free Alternating Tree Automaton.

3.1 Propositional Mu-Calculus

Definition 3.0: The *formulae* of the Propositional Mu-Calculus are:
(1) Propositional letters $P, Q, R...$
(2) Propositional variables $..., X, Y, Z$
(3) $\neg p$, $p \vee q$, and $p \wedge q$, where p and q are any formulae
(4) EXp and AXp, where p is any formula
(5) $\mu X.f(X)$ and $\nu X.f(X)$, where $f(X)$ is any formula syntactically monotone in the propositional variable X.

A *sentence* is a formula containing no free propositional variables In the sequel, we will use λ as a generic symbol for μ or ν. Sentences are interpreted in Kripke Trees.

Definition 3.1: A *Kripke Tree* is an infinite binary tree $t : \{0,1\}^* \rightarrow Prop$. A satisfaction relation $\models$ is defined between $\{0,1\}^*$ and $Prop$. We say, $x \models P$ iff $P = t(x)$. Note that, usually a Kripke tree is defined as a map $t : \{0,1\}^* \rightarrow 2^{Prop}$, but since we want to show equivalence of the Mu-Calculus to Tree Automata, we use the labelling as in Tree Automata.

Definition 3.2: A *model* is a Kripke Tree with the Satisfaction relation extended to all sentences by means of the usual boolean rules and
(1) $x \models EXp$ iff $\exists i \in \{0,1\}\ : x \cdot i \models p$
(2) $x \models AXp$ iff $\forall i \in \{0,1\}\ : x \cdot i \models p$
(3) $x \models \nu X.f(X)$ iff $x \in \bigcap\{S \subseteq U | S = \{y | y \models f(X)$ with X interpreted as $s\}\}$,
(4) $x \models \nu X.f(X)$ iff $x \in \bigcup\{S \subseteq U | S = \{y | y \models f(X)$ with X interpreted as $s\}\}$.

Every formula has a *positive normal form* in which all negations apply directly to proposition letters. Define $not(p)$ to be the positive normal form of $\neg p$.

Definition 3.3: The *Fischer Ladner Closure* of a sentence p in positive normal form, is the smallest set $FL(p)$ of sentences satisfying the following constraints:

(1) $p \in FL(p)$
(2) if $q \in FL(p)$ then $not(q) \in FL(p)$
(3) if $q \vee r \in FL(p)$ then $q, r \in FL(p)$
(4) if $q \wedge r \in FL(p)$ then $q, r \in FL(p)$
(5) if $EXq \in FL(p)$ then $q \in FL(p)$
(6) if $AXp \in FL(p)$ then $q \in FL(p)$
(7) if $\mu X.f(X) \in FL(p)$ then $f(\mu X.f(X)) \in FL(p)$
(8) if $\nu X.f(X) \in FL(p)$ then $f(\nu X.f(X)) \in FL(p)$.

Definition 3.4: A *pre-model* is a Kripke Tree with a satisfaction relation $\models$ extended to $FL(p)$ under the following constraints:
(1) $x \models p$ iff $x \not\models not(p)$
(2) $x \models p \vee q$ iff either $x \models p$ or $x \models q$
(3) $x \models EXp$ iff $\exists i \in \{0,1\} : x \cdot i \models p$
(4) $x \models \mu X.f(X)$ iff $x \models f(\mu X.f(X))$.

A premodel is almost a model, except rule (4) permits $\mu x.f(X)$ to be interpreted as an arbitrary fixpoint.

3.2 Construction

Theorem 3.1: Given a Mu-Calculus formula f_0, we build a Streett History-free Alternating Tree Automaton equivalent to f_0, in the sense that the Automaton accepts exactly those binary trees labelled with $Prop$, which are models of f_0. Here $Prop$ is the set of Proposition symbols in f_0.

Proof Sketch: Consider the parse tree of f_0. It can be viewed as an Alternating Automaton's transition diagram. The $\wedge$- nodes in the parse tree correspond to $AAND$ nodes, the $\vee$-nodes to OR nodes, while AXp and EXp correspond to $DAND$ nodes. Moreover, an occurence of variable x bound in $\lambda x.f(x)$, is identified with the node $\lambda x.f(x)$ (i.e. makes a loop). The above transition diagram with acceptance condition $\Phi = true$ defines an Alternating Automaton $\mathcal{A}$. Note that the transition diagram defined above is not necessarily a tripartite graph.

Given a map $\rho_I : \{s_0\} \cdot \{\{0,1\} \cdot OR\}^* \rightarrow DAND$, a pre-model $T(\rho_I)$ can be generated from the Alternating Automaton $\mathcal{A}$. It can be shown that $T(\rho_I)$ is a pre-model of f_0 extending the labelling of t iff ρ_I is a winning strategy for Player I in $\Gamma(\mathcal{A}, t)$.

Moreover, the winning strategy (also called the choice function in [StE84]) determines a derivation relation between occurences of sentences in the pre-model so obtained [StE84]. We would like to say that a pre-model is in fact a model when there is no infinite derivation sequence which rederives a mu-sentence infinitely often. However, [StE84] show that this claim is true only when restricted to derivations in which the given μ-sentences appear as a subsentence of every derivation step. We say that a μ-sentence $\mu X.f(X)$ is *regenerated* from x to y if $\mu X.f(X)$ at x derives $\mu X.f(X)$ at y in such a way that $\mu X.f(X)$ is a subsentence of every derivation step. A winning strategy ρ_I in the above game is a *well-founded winning strategy* when the regeneration relations for μ-sentences are well-founded. The main theorem in [StE84] states that a pre-model is a *Model* iff there is a well-founded winning strategy generating it from the transition diagram. Moreover, a well-founded winning strategy defines a straightforward history-free well-founded winning stategy, because Player I can pick the choice with the least rank with respect to the the regeneration relations (for a formal proof, see [StE84]).

For a fixpoint formula $\lambda X.f(X)$, we say that X is bound to this formula. W.l.o.g. assume that in f_0, every fixpoint subexpression has a unique X bound to it. We say that a fixpoint subexpression is of higher precedence than another, if the latter is contained as a strict subexpression in the first. For each fixpoint subformula there is a fixpoint sentence in $FL(f_0)$, given inductively as follows: For fixpoint sub-formulae of maximal precedence the sub-formulae, which are sentences in this case, itself are in $FL(f_0)$. We define a 1-1 and onto map H between variables and the fixpoint sentences occuring in $FL(f_0)$ corresponding to the fixpoint subexpression to which the variable is bound. Thus, as already stated, $H(X) = f$, where f is a maximal precedence subexpression such that X is bound to it. $H(X) = \lambda X.f(X, H(Y_1), ..H(Y_k))$, where X is bound to $\lambda X.f(X, Y_1, ..Y_k)$ in f_0. Clearly, Y_i are of higher precedence than X, and hence the above definition is well-founded.

It is a simple exercise to note that any rederivation of $H(X)$ at s to $H(X)$ at t is a regeneration iff there is no $H(Y)$ derived inbetween, such that Y is of higher precedence than X. This motivates the following modification in the acceptance condition of the Alternating Automaton $\mathcal{A}$ obtained above, such that in the new Streett Alternating Automaton $\mathcal{B}$ a winning strategy is a well-founded winning strategy in $\mathcal{A}$ and vice versa. Thus, the input tree t is a model (i.e. has an extendable satisfaction relation as in Definition 3.2) iff $\mathcal{B}$ has a winning strategy in $\Gamma(\mathcal{B}, t)$. Since, the only change is in the acceptance condition, a history-free strategy in $\Gamma(\mathcal{A}, t)$ remains a history-free strategy in $\Gamma(\mathcal{B}, t)$. Thus $\mathcal{B}$ is a history-free Alternating Automaton.

Consider the partial order "$<$" on the set of variables in f_0 given by the precedence relation. We now assign an integer $height(X)$ to each X. All leaf variables (i.e. $\neg\exists Y : (Y < X)$) X in the partial order have $height(X) = 1$. Otherwise, $height(X) = 1+\max(height(X_i) : X_i < X)$. The set $GREEN_i$ is the set of nodes corresponding to X, such that $height(X) = i$ and X is bound to a μ-subexpression. The set RED_i is the set of nodes corresponding to X such that $height(X) \geq i$ and X is bound to a ν-subexpression. The Streett acceptance condition is given by $\forall i : (GREEN_i$ i.o. $\Rightarrow RED_i$ i.o.).

It can be shown that $\mathcal{A}$ has a well-founded winning strategy iff $\mathcal{B}$ has a winning strategy. Thus $\mathcal{B}$ is the required Streett history-free Alternating Automaton. □

4 Tree Complementation Lemma

¿From Theorem 3.1 and Theorem 2.2 we conclude that Propositional Mu-calculus can be translated to Non-deterministic Rabin (Pairs) Tree Automata. Moreover, Niwinski [Ni88] had shown that Non-deterministic Pairs Automata can be translated to Propositional Mu-Calculus. Thus, Propositional Mu-Calculus and Non-deterministic Tree Automata are expressively equivalent. Rabin's Tree Complementation Lemma, i.e. languages accepted by Tree Automata are closed under complementation, follows by trivial complementation of Mu-Calculus.

Note that our proof of complementation via Mu-Calculus did indeed involve Alternating Tree Automata which was also, implicitly or explicitly, used in all earlier proofs, as outlined in the Introduction. However, our proof did not seem to involve difficult proofs of (1) and (2) (whereas, all known proofs of (1) and (2) have been intimidating). This suggests that their might be simple proofs of (1) and (2). Indeed, we now give simple proofs of (1) determinacy of infinite games for Tree Automata and (2) history-free strategies for players in such games.

First, note that in the Proof of theorem 3.1, instead of translating Mu-Calculus to Streett Automata, the natural Automata to which Mu-Calculus translates easily is the one with the following "parity" acceptance condition.

Definition 4.0: Let the states of a Tree Automaton, or the OR nodes, be labelled with colors [0..m]. For an infinite path or sequence of nodes,
$even_k$ = largest i.o. occurring color index among $[0..k]$ is even,
odd_k = largest i.o. occurring color index among $[0..k]$ is odd.
We say that $x \in OR^\omega$ satisfies the parity condition iff x satisfies $even_m$.

Stated in terms of the usual Green, Red pairs, the parity condition is: $\exists i$ (i.o. $Green_i$ and $\forall j \geq i$ f.o. Red_j). Note that as opposed to Streett and Pairs acceptance condition, parity acceptance condition is trivially closed under complementation. It is this property which makes proving determinacy of games with parity acceptance condition much easier. In fact, a Hossley-Rackoff like finite model theorem for Parity Tree Automata also turns out to be much simpler. Moreover, Parity Tree Automata are trivially convertable to Pairs Tree Automata. Also, a simple conversion from Pairs Tree Automata to Parity Tree Automata can be obtained by a slight modification of [Sa89], which converts a deterministic Pairs Automata to deterministic Streett Automata. This is also the essence of the LAR argument in [GH82]. Similarly, infinite games with Muller or Pairs condition easily reduce to Infinite games with Parity conditions.

4.1 Determinacy of infinite games with parity winning conditions

We studied a number of different games in the previous sections. However in general, a two player infinite game can be given by a *game tree* which is an AND/OR infinite tree with its nodes labelled with colors [0..m]. W.l.o.g. we assume that each OR node has the same color as all its AND successors. We generalize the above game trees to syntactic game trees, in which nodes could be labelled with Mu-Calculus expressions with or without free variables (modality F stands for eventually). A strategy for player I picks an AND successor at each OR node, while a strategy for Player II picks an OR successor at each AND node. A strategy ρ in a game tree defines another tree (which we will call ρ), from the given game tree.

Definition 4.1:
$I_k(t)$ = Set of game trees, s.t. Player I has a strategy ρ with all paths satisfying: ($even_k$ or Ft).
$I_{-1}(t) = t$.
$II_k(t)$ = Set of game trees, s.t. Player II has a strategy ρ with all paths satisfying: (odd_k or Ft).

$\lambda_n = \nu$ if n is even, and μ if n is odd.

Theorem 4.1: $I_n(t) = \lambda_n x_n..\mu x_1 \nu x_0 (\bigvee_{i\in[0..n]}(color_i \wedge EXAXx_i) \vee t)$.

Proof: We prove by induction over n. Consider the equation,
$y = \lambda_{n-1}x_{n-1}...\nu x_0(\bigvee_{i\in[0..n-1]}(color_i \wedge EXAXx_i) \vee (color_n \wedge EXAXy) \vee t)$, when $n > 0$,
and $y = t$ when $n = 0$.

Using induction hypothesis on the r.h.s, the equation above becomes
$y = I_{n-1}(t \vee color_n \wedge EXAXy)$.

$I_n(t)$ is easily seen to be the fixpoint of the above equation $y = I_{n-1}(t \vee color_n \wedge EXAXy)$ (just follow the definition of $I_k(t)$).

When n is even, to prove that $I_n(t)$ is the greatest fixpoint, let $\psi \in y$ be a game tree. We show that $\psi \in I_n(t)$. We prove by induction on k that ψ is a game tree, with a strategy ρ for player I, s.t.
$A(Ft \vee even_{n-1} \vee s_0...s_1...s_i...s_{k-1} : \forall i > 0 : s_i$ is $color_n$, and at s_{k-1}: $EXAXy)$.

For $k = 1$, it holds because, $y = I_{n-1}(t \vee color_n \wedge EXAXy)$. Suppose it holds for k, then for paths in ρ where: $s_0..s_1...s_i..s_{k-1} : \forall i > 0 : s_i$ is $color_n$, and at s_{k-1}: $EXAXy$, there is a choice (extension of ρ) such that all successors y, and hence $I_{n-1}(t \vee color_n \wedge EXAXy)$. A trivial inspection shows that along all extensions of the path one of the three disjuncts holds, with k incremented, and we are done.

If n is odd, to prove $I_n(t)$ is the least fixpoint, let ρ be a strategy for player I in $\psi \in I_n(t)$, s.t. along all paths of ρ: $(even_n \vee Ft)$. Then, the relation R given as follows on the nodes of ρ is well-founded: uRv iff there is a path from u to v, and u or v is labelled n; for otherwise we have an infinite path in ρ with infinitely many n. Let u be a R-minimal node in ρ, such that the game tree starting at u is not in y. By virtue of ρ, u is in $I_{n-1}(t \vee v)$, where v is a node labelled $color_n$ and has successors $v_1, ..v_i, ..v_r$. Note that, for every i, uR^+v_i, since v is labelled $color_n$. Now, either for all such nodes v (if any), for all i, v_i is in y, in which case u is in y (by (1)). Else, we found a v_i not in y, and uR^+v_i, which contradicts the R-minimality of u. Thus, every u in ρ is in y. □

Theorem 4.2: $II_n(t) = \lambda_{n-1}x_n...\nu x_1 \mu x_0(\bigvee_{i\in[0..n]} (color_i \wedge AXEXx_i) \vee t)$.

Proof: An argument symmetric to Theorem 4.1. □

Corrollary 4.3: Infinite games defined over game trees with nodes labelled with colors $[0..m]$ and with parity winning (or acceptance) conditions are determined. □

4.2 History free winning strategies

We show that if a player has a winning strategy in a parity infinite game, then the player has a winning strategy which does not depend on the history. More precisely, a Player I's winning strategy is history-free if it has the following property: if the partial game trees beginning at two different OR nodes are identical, then the winning strategy picks the same AND successor at these two OR nodes. A history-free winning strategy for Player II is defined similarly.

Theorem 4.4: If a player A (I/II) has a winning strategy in a parity infinite game then A has a history-free winning strategy.

A partial game tree beginning at node x of a game tree ψ, will be denoted ψ_x. A partial winning strategy (and the tree defined by it) beginning at node x in the winning strategy ρ will be denoted by ρ_x.

Proof: We prove the result for A being Player I. The case of Player II is handled similarly, by switching the OR and the AND nodes.

Suppose, player I has a winning strategy ρ. As remarked earlier, the winning strategy defines a tree ρ in the game tree. Consider the following maps μ_{2k+1} from the nodes in ρ to the class of all ordinals:

$\mu_{2k+1}(x) = 0$, if along all paths in ρ_x if there is a node labelled $color_{2k+1}$, then there is a node before it labelled $color_n$, where $n > 2k+1$,

$\mu_{2k+1}(x) = \delta_{2k+1}(x)+ \sup\{\mu_{2k+1}(y)$: there is a path from x to y such that y is labelled $color_{2k+1}$, and there is no node in this path with $color_n$, $n > 2k+1\}$, where $\delta_{2k+1}(x)$ is 1 if x is labelled $color_{2k+1}$, else 0.

The above definition is well-defined, because along all paths, for all k, every occurence of $color_{2k+1}$ is followed by a $color_n$, $n > 2k+1$. Thus, the above inductive definition is well-founded.

Let $\mu = \mu_{2m+1}\mu_{2m-1}..\mu_1$ with a left-lexicographic ordering, which is again a well-ordering. We let $\mu_{\geq 2k+1} = \mu_{2m+1}\mu_{2m-1}..\mu_{2k+1}$. The history-free strategy we define, essentially picks at each OR-node x, the AND node with the least μ, in case of contention (i.e. if in ψ there are other OR nodes with partial game trees identical to ψ_x, with ρ picking different AND-successors at these OR nodes).

We now show that the strategy ρ' so obtained is indeed a winning strategy for Player I. Suppose, in ρ' there is a path such that eventually $color_{2k+1}$ appears infinitely often, and no $color_n$, $n > 2k+1$ appears. We show that with every two such occurences x_1 and x_i of $color_{2k+1}$, μ decreases, thus contradicting that μ is a well-order. Here, x_j is the successor of x_{j-1} in ρ'. Let the successor of x_{j-1} in ρ be x'_j.

Fact 1: $\forall j : i-1 \geq j \geq 1 : \mu_{\geq 2k+1}(x_j) \geq \mu_{\geq 2k+1}(x'_{j+1}) \geq \mu_{\geq 2k+1}(x_{j+1})$.

The second inequality follows because in ρ' the least mu is picked, while the first inequality follows because x_j has $color_n$, $n \leq 2k+1$.

Fact 2: $\mu_{2k+1}(x_1) > \mu_{2k+1}(x'_2)$.

This follows by definition of μ_{2k+1}, noting that $\delta_{2k+1}(x_1) = 1$.

Thus, $\mu_{\geq 2k+1}(x_1) > \mu_{\geq 2k+1}(x_i)$. □

5 Determinacy of On-line games

Consider an infinite game tree in which each OR node is colored with $color_n$, where n is a natural number. [RS90] construct a game to study competitiveness of on-line algorithms in which Player I wins a particular play, say $x_0, x_1, ... \in OR^\omega$, iff $\exists i \forall k\ x_k$ has $color_n$, $n < i$.

As before, let I be the set of game trees in which player I has a winning strategy, and II be the set of game trees in which player II has a winning strategy. Then, a proof much simpler than Theorem 4.1 gives the following, proving determinacy of this game.

Theorem 5.1:
$I = \mu y \exists i \nu x((EXAXx \wedge color_n, n < i)\ \vee y)$,
$II = \nu y \forall i \mu x((AXEXx \vee color_n, n \geq i)\ \wedge y)$.

This simplifies the existential proof (in [RS90]) of de-randomization of competitive on-line algorithms for task systems, where [RS90] was invoking Wolfe's proof ([W55]) of determinacy of F_σ games. Infact, a similar Mu-Calculus characterization can be given for F_σ (countable union of closed sets) games, elucidating Wolfe's proof. We use terminology from [Mo80].

Let $A = \bigcup_{i \in \omega} F_i$ be the winning set for Player I, where each F_i is closed in the product topology of ${}^\omega X$. It is well known [Mo80] that a closed set F_i is exactly the infinite paths in a Tree T_i on X. We say that T_i holds on the last node of a path p in the game tree iff p occurs in T_i.

Theorem 5.2: $I = \mu y \exists i \nu x((EXAXx \wedge T_i) \vee y)$, and
$II = \nu y \forall i \mu x((AXEXx \vee \neg T_i)\ \wedge y)$. □

6 Acknowledgement

We thank Joe Halpern and Prasad Sistla for pointing out the alternative argument establishing equivalence of the Mu-Calculus and tree automata by translation through SnS.

References

[Bu77] J.R. Buchi, " Using Determinacy to eliminate quantifiers", *Fundamentals of Computation Theory*, LNCS 56, 367-378.

[Bu83] J.R. Buchi, " State-Strategies for Games in $F_{\sigma\delta} \cap G_{\delta\sigma}$", J. Symbolic Logic 48, 1983, 1171-1198.

[B90] S. Ben-David, A. Borodin, R. Karp, G. Tardos, A. Wigderson, "On the Power of Randomization in On-line algorithms", Proc. ACM STOC 90.

[CKS81] A.K. Chandra, D.C. Kozen, L.J. Stockmeyer, "Alternation", J. ACM 28 (1981), pp. 117-141.

[EJ88] E. A. Emerson and C.S. Jutla, "Complexity of Tree Automata and Modal logics of Programs", IEEE FOCS 1988.

[EJ89] E. A. Emerson and C.S. Jutla, "On Simultaneously Determinizing and Complementing ω-Automata", IEEE LICS 1989.

[GH82] Y. Gurevich, L. Harrington, "Trees, Automata, and Games", *14th ACM STOC*, 1982.

[HR72] R. Hossley and C. Rackoff, "The emptiness problem for automata on infinite trees", Proc. 13th IEEE Symp. Switching and Automata Theory, 1972, pp. 121-124.

[J90] C.S. Jutla, "Automata on Infinite Objects and Modal Logics of Programs", PhD Thesis, The University of Texas at Austin, May 1990.

[Ko83] D. Kozen, "Results on the Propositional Mu-Calculus", *Theoretical Computer Science* , 27, 1983, pp. 333-354.

[Ma75] D.A. Martin, "Borel Determinacy", *Annals of Mathematics*, 1975, 102, 363-371.

[Mc66] R. McNaughton, "Testing and Generating Infinite Sequences by a finite Automaton", *Information and Control*, 9, 521-530, 1966.

[Mo80] Y. Moschovakis, "Descriptive Set Theory", North Holland, 1980.

[Mon] J. D. Monk, unpublished notes.

[Mu84] A.A. Muchnik, Games on Infinite trees and automata with dead-ends: a new proof of the decidability of the monadic theory of two successors, *Semiotics and Information*, 24, 1984, 17-40 (in Russian).

[MS84] D.E. Muller, P.E. Schupp, "Alternating Automata on Infinite Objects, Determinacy and Rabin's Theorem", *Automata on Infinite Words*, 1984, Lecture Notes in Computer Science, 192.

[MS87] D.E. Muller, P.E. Schupp, " Alternating Automata on Infinite Trees", Theoretical Computer Science 54, 267-276.

[Ni84] D. Niwinski, "The Propositional μ-Calculus is more expressive than the propositional dynamic logic of looping", manuscript, June 84, Institute of Mathematics, University of Warsaw.

[Ni86] D. Niwinski, "On fixed-point Clones", Proc. 13th ICALP, LNCS 226, 464-473, 1986.

[Ni88] D. Niwinski, "Fixed Points vs. Infinite Generation", Proc. 3rd. IEEE LICS 1988, 402-409.

[Ra69] M.O. Rabin, "Decidability of Second Order Theories and Automata on Infinite Trees", *Trans. AMS*, 141(1969), pp. 1-35.

[RS90] P. Raghavan, and M. Snir, " Memory versus Randomization in On-Line Algorithms", Research Report IBM T.J. Watson Research Center, March 1990.

[Sa88] S. Safra, "On Complexity of ω-automata", Proc. 29th IEEE FOCS, 1988.

[Sa89] S. Safra, "Complexity of Automata on Infinite Objects", PhD Thesis, Weizmann Institute of Science, Rehovo, Israel, March 1989.

[St81] R.S. Streett, "A Propositional dynamic Logic of Looping and Converse", *MIT LCS* Technical Report TR-263.

[StE84] R.S.Streett, E.A. Emerson, "An Elementary Decision Procedure for the Mu-calculus", *Proc. 11th Int. Colloq. on Automata, languages and Programming*, 1984, Lecture Notes in Computer Science, Springer-Verlag.

[Th88] W. Thomas, " Automata on Infinite Objects", to appear in *The Handbook of Theoretical Computer Science*, North-Holland.

[W55] P. Wolfe, "The strict determinateness of certain infinite games", Pacific J. Math., 5, Supplement I:841-847, 1955