Theoretical Computer Science 97 (1992) 233–244
Elsevier

Alternating automata, the weak monadic theory of trees and its complexity*

David E. Muller**
University of Illinois

Ahmed Saoudi
Université Paris 13

Paul E. Schupp**
University of Illinois, and Université Paris 7

Communicated by D. Perrin
Received June 1989
Revised June 1990

1. Introduction

Beginning with the fundamental article of Chandra et al. [2], the notation of alternation has clarified several results concerning the complexity of logical theories. Muller and Schupp [9] extended the idea of alternation to automata working on trees. Although such automata are a generalization of Rabin's model [12] of non-deterministic automata working on infinite trees, complementation and, thus, all Boolean operations, are easy for such automata. In particular, complementation costs no extra states. There is, of course, no free lunch and, in the alternating model, one must pay for projection. Thus, alternating automata do not give a simple proof of Rabin's fundamental theorem [12] on the decidability of the full monadic theory of the tree. Indeed, one must appeal to the powerful "forgetful determinacy" theorem of Gurevitch and Harrington [3] to show that alternating automata can be simulated by nondeterministic Rabin automata.

*This article is the full version of the paper [10] which we presented at ICALP'86.
**The research of the first and third authors was supported by National Science Foundation grant CCR87-03807.

Nonetheless, we feel that alternating automata give the "natural" theory of automata working on trees. It is often the case that one works with a logic less powerful than full monadic logic, such as the weak monadic theory of the tree, which permits quantifiers only over finite sets, or temporal or dynamic logic, where quantification is extremely restricted. Alternating automata become progressively more advantageous as quantification is restricted since Boolean operations are always easy. The study of automata on infinite trees rests on the fundamental articles of Rabin [12, 13]. Rabin gave an ingenious characterization of weakly definable languages of k-ary trees. A language L is definable by a formula of weak monadic logic if and only if both L and its complement $\bar{L}$ are accepted by automata using Büchi acceptance. We shall define a "weak acceptance" condition and prove that a language L is weakly definable if and only if L is accepted by an alternating automaton using weak acceptance. As explained below, this result both uses and simplifies Rabin's characterization. Secondly, since alternating automata are closely related to complexity, we give a simple proof of an $(n+1)$-exponential time bound on the complexity of deciding weak monadic sentences having at most n blocks of quantifiers in their prenex normal form.

We begin with a discussion of acceptance conditions and explain Rabin's result. In his pioneering work on finite automata accepting infinite words Büchi [1] worked with nondeterministic automata and supposed the acceptance condition to be defined by a subset F of the state set Q. An infinite calculation h of the automaton accepts if h contains some state from F infinitely often. The problem with ~~non~~deterministic automata is, of course, complementation. In order to be able to determinise determine a Büchi automaton, one must use the acceptance condition of Muller [7], which is defined by a family $\mathscr{F}$ of subsets of the state set. An infinite calculation h of the automaton accepts if $\mathrm{Inf}(h) \in \mathscr{F}$, where $\mathrm{Inf}(h)$ is the set of states occurring infinitely often in h. McNaughton [6] proved that any regular set of infinite words can be accepted by a deterministic Muller automaton. The relationship between Muller acceptance and complementation is not surprising when one notes that the negation denial of a Büchi acceptance condition is not a condition of the same type, while the denial of the Muller condition defined by a family $\mathscr{F}$ is simply the Muller condition defined by the complementary family $\bar{\mathscr{F}}$.

Rabin [12] showed that, in general, it is necessary to use Muller acceptance when considering automata on trees. Nonetheless, automata using the Büchi acceptance condition are used by Rabin [13] to characterize the weakly definable languages. Rabin calls such automata *special* but we shall call them *Büchi*, and we say that a language is *Büchi* if it is accepted by a Büchi automaton. Rabin proves that a language L is weakly definable if and only if both L and $\bar{L}$ are Büchi. There are several characterizations of this general character in logic; for example, the basic fact that a set S of natural numbers is recursive if and only if both S and $\bar{S}$ are recursively enumerable.

We shall consider a "weak acceptance condition" which would indeed be extremely weak for nondeterministic automata. We consider alternating automata whose state set is partitioned as a disjoint union $Q = \bigcup Q_i$, and we suppose that there is a partial ordering on the collection of the Q_i. Furthermore, we suppose that the transition function is such that given a $q \in Q_i$; if q' is any state occurring in $\delta_a(a, q)$ then $q' \in Q_j$, where $Q_i \geqslant Q_j$. Thus, if h is an infinite individual history, from some point onwards all the states in h belong to the same set Q_i. We say that $Q_i = f(h)$ is the *finality* of h. We suppose that each Q_i is designated as accepting or rejecting. The history h is accepting if $f(h)$ is an accepting set. We say that an alternating automaton using the weak acceptance condition is a *weak alternating automaton*.

Our main result is stated in the following theorem.

Theorem 1.1. *Let L be a language of k-ary trees labelled from an alphabet Σ. The following conditions are equivalent*:

(1) *L is weakly definable.*
(2) *L is accepted by a weak alternating automaton.* ↗ priority blow-up?
(3) *Both L and $\bar{L}$ are Büchi.*

Rabin's characterization is that (1) is equivalent to (3). Our proof has the form (1)⇒(2)⇒(3)⇒(1) and proceeds as follows. We first recall the basic definitions about alternating automata from [9] and the complementation theorem, which remains valid for weak acceptance. If M accepts a language L then the dual automaton $\tilde{M}$ accepts $\bar{L}$. The class of languages accepted by weak alternating automata is, thus, closed under complementation. We first show that the class of languages accepted by weak alternating automata includes all weakly definable languages. We next show that a weak alternating automaton can be simulated by a Büchi automaton. Since we already have closure under complementation, if L is accepted by a weak alternating automaton then both L and $\bar{L}$ are Büchi and L is, thus, weakly definable by Rabin's theorem. Note that we have used only one direction of Rabin's theorem and this direction is, in fact, the one with the shorter, more conceptual proof. We view this fact as strengthening our contention that it is simply much easier to calculate with alternating automata. Finally, we consider the complexity of deciding the truth of formulas with n blocks of quantifiers. The close connection between alternating automata and complexity yields a simple proof of the following result.

Theorem 1.2. *Let P_n be the class of sentences of the weak monadic logic of the k-ary tree which are in prenex normal form and which have at most n alternations of type of quantifier. There is an $(n+1)$-exponential-time algorithm which decides the truth of sentences in P_n.*

2. Weak alternating automata on the tree

We review the conception of alternating automata as given in [9]. In Rabin's theory of nondeterministic automata on the binary tree, a single copy of the automaton begins in its initial state at the root of the tree. The automaton then splits into two copies, one moving to the left successor and the other moving to the right successor. The states of the two copies are given by a nondeterministic choice from the possibilities allowed by the transition function. In Rabin's notation, if the automaton is in state q_0 reading the letter a, the value of the transition function for (q_0, a) might be $\{(q_1, q_2)(q_0, q_3)\}$, where the left (right) member of a pair denotes the next state of the automaton moving to the left (right) successor vertex. We represent this situation in our lattice formulation by using the free distributive lattice $\mathscr{L}(\{0,1\} \times Q)$ generated by all the possible pairs (direction, state). We write

$$\delta(a, q_0) = (0, q_1) \wedge (1, q_2) \vee (0, q_0) \wedge (1, q_3)$$

(where, as usual, $\wedge$ has precedence over $\vee$).

We interpret this expression as saying that the automaton has the choice of splitting into one copy in state q_1 going to the left successor *and* one copy in state q_2 going to the right successor, *or* of splitting into one copy in state q_0 going to the left *and* one copy in state q_3 going to the right. We note that both "and" and "or" are present in the conception of an automaton on the binary tree.

In the general case of an alternating automaton we allow $\delta(a, q)$ to be an arbitrary element of the free distributive lattice $\mathscr{L}(\{0,1\} \times Q)$. For example, the dual of the expression above is

$$\tilde{\delta}(a, q_0) = (0, q_1) \wedge (0, q_0) \vee (0, q_1) \wedge (1, q_3) \vee (0, q_0) \wedge (1, q_2) \vee (1, q_2) \wedge (1, q_3).$$

This expression illustrates that we do not require the automaton to send copies in all directions (although, at least one copy must go in some direction) and that several copies may go in the same direction. One may think of an alternating automaton as a sort of completion of a nondeterministic automaton. It is only by going to $\mathscr{L}(\{0,1\} \times Q)$ that one can always calculate the dual of a given expression.

We review our conventions on the k-ary tree T_K viewed as a structure. The vertex set of T_K is the set K^* of all words on the *direction alphabet* $K = \{0, \ldots, k-1\}$, with the empty word being the origin of the tree. Given a vertex v and a letter $d \in K$, there is an *edge* e with label d from v to vd and vd is the *d-successor* of v. The *level* $|v|$ of a vertex v is the length of v as a word. Thus, we think of the edges in T_K as being labelled by letters from K, while the vertices are unlabelled.

Definition 2.1. A *weak alternating automaton* on K-ary Σ-trees is a tuple

$$M = \langle \mathscr{L}(K \times Q), \Sigma, \delta, q_0, F \rangle,$$

where K is the set of *directions*, the *state set* Q is written as a disjoint union $Q = \bigcup Q_i$ and there is a partial order $\geqslant$ on the collection of the Q_i. The set Σ is the *input alphabet* and the *transition function* $\delta : \Sigma \times Q \to \mathscr{L}(K \times Q)$ has the property that if $q \in Q_i$ and q' occurs in the expression $\delta(a, q)$ then $q' \in Q_j$, where $Q_j \leqslant Q_i$. The *final family* F is a list of these Q_i considered to be accepting. We denote the cardinality of the state set of M by $|M|$. The *dual* of M is the automaton

$$\tilde{M} = \langle \mathscr{L}(K \times Q), \Sigma, \tilde{\delta}, q_0, \bar{F} \rangle$$

obtained by dualizing the transition function (by interchanging $\wedge$ and $\vee$ as usual) and taking the complement $\bar{F}$ of F.

The reader may consult [9] for complete details, but the only result which we use in this article is the fact that acceptance of the complementary language by the dual automaton remains valid for weak alternating automata.

Complementation Theorem. *Let M be a weak alternating automaton and let $L(M)$ be the language accepted by M. Then the dual automaton $\tilde{M}$ accepts the complement of $L(M)$.*

Note that for alternating automata complementation does not cost additional states. The operation of union is, as always, trivial, but note that union and intersection are symmetric for alternating automata and both operations thus cost only one additional state.

Lemma 2.2. *Let L_i be accepted by weak alternating automata M_i, $i = 1, 2$. Then there are weak alternating automata with $|M_1| + |M_2| + 1$ states accepting $L_1 \cup L_2$ and $L_1 \cap L_2$.*

Proof. Let $M_i = \langle (K \times Q_i), \Sigma, \delta_i, q_i, F_i \rangle$, where Q_1 and Q_2 are disjoint. To define the desired automaton M, let z_0 be a new initial state; let $Q = Q_1 \cup Q_2 \cup \{z_0\}$ be the state set of M, and let $F = F_1 \cup F_2$.The transition function δ of M is the same as δ_i on Q_i. To accept $L_1 \cup L_2$, define $\delta(z_0) = \delta_1(q_1) \vee \delta_2(q_2)$. To accept $L_1 \cap L_2$, define $\delta(z_0) = \delta_1(q_1) \wedge \delta_2(q_2)$. $\square$

Definition 2.3. Let Δ and Σ be alphabets with $\Delta \supseteq \Sigma$ and let $\eta : \Delta \to \Sigma$ be a function which is the identity function on Σ. Let L be a language of k-ary trees labelled from Δ. We define the language $\eta_f(L)$ on the alphabet Σ which is obtained from L and η by *finite projection* as follows. A tree t' belongs to $\eta_f(L)$ if there exists a tree $t \in L$ containing only *finitely* many vertices labelled from $\Delta - \Sigma$ and such that $t' = \eta(t)$, where, as usual, $\eta(t)$ denotes the result of replacing the label of each vertex of t by its image under η.

Lemma 2.4. *The class of languages accepted by weak alternating automata is closed under finite projection. If L is accepted by M then there is a weak alternating automaton M', with $2^{|M|} + |M| + 2$ states, which accepts $\eta_f(L)$.*

Proof. Given a weak alternating automaton $M = \langle \mathscr{L}(K \times Q), \delta, q_0, F \rangle$, we can construct M' accepting $\eta_f(L(M))$ as follows. In order to simulate the behaviour of M up to a finite distance, it is necessary to keep track of all the possible copies running in M only up to the information of current state. We think that two machines having the same information "merge". (We discuss infinite "uniformization" in the next lemma.)

The automaton M' begins in a "nondeterministic mode" where, at a given vertex, it keeps track of a possibility for an existing collection of machines in M up to the information of current state. Thus, M' requires a state set of the form $\mathscr{P}(Q)$ for its nondeterministic mode. When in state $S \in \mathscr{P}(Q)$ and reading the letter a, M' chooses a pre-image in $\eta^{-1}(a)$, a set of choices of the copies of M represented in S on this pre-image, and sends in each direction d the collection S_d resulting from S and the choices made.

At any vertex, M' also has the choice of guessing that it will not see any more letters from $\Delta - \Sigma$ in the subtree beginning at the vertex. If M' makes this choice, it enters its alternating mode when it simply simulates M in an alternating fashion but will go into a special rejecting state q_r if it sees a letter of $\Delta - \Sigma$. In order to do this, M' has a copy of Q which is disjoint from $\mathscr{P}(Q)$. The ordering is $\mathscr{P}(Q) > Q_i$ for each Q_i in the decomposition of Q in M. The transition from a "nondeterministic" state $S \neq \phi$ to the alternating mode is from S to $\wedge_i q_i$, where $q_i \in S$. M also has a special state q_ϕ indicating the absence of any copy of M and the transition from ϕ is to q_ϕ. The transition function of M' on a state $q \in Q$ is exactly the same as that of M on letters from Σ but M' goes into a special rejecting state q_r on any letter from $\Delta - \Sigma$. A copy in q_ϕ or in q_r always stays in that state in every direction, except that q_ϕ changes to q_r on reading a letter from $\Delta - \Sigma$.

The ordering on the special states is $Q_i > \{q_r\}$ and $Q_i > \{q_\phi\}$ for each Q_i. The final family F' of M' consists of the final family F of M together with $\{q_\phi\}$. Note that $\mathscr{P}(Q)$ is rejecting. Thus, that an individual history h in M' has $f(h) \in F'$ requires that M' has made the transition to the alternating mode and that the simulation of M is accepting, and that no letters from $\Delta - \Sigma$ are encountered in the alternating mode. The König infinity lemma assures that, since M' has guessed on every branch, the total subtree covered in the nondeterministic mode is indeed finite. We note that the construction of M' is really simply the subset construction. $\square$

We now wish to prove that a weak alternating automaton can be simulated by a Büchi automaton. In order to do this, we need to "uniformize" the behaviour of M. In the case of weak automata this is a simple lemma, which is in marked contrast to the deep theorem of Gurevitch and Harrington [3] concerning automata using Muller acceptance. Gurevitch and Harrington show that if an automaton with Muller acceptance accepts an input then there exists a "finite memory" strategy for acceptance. We show that for weak automata there is a "zero memory" or "completely uniform" accepting strategy. For the definition of the computation tree of an alternating automaton on a given input see [9]. memoryless

Definition 2.5. Suppose that M accepts an input tree t. A branch β' of the computation tree $T(M, t)$ is *uniform* if, in β', any two copies of M which are in the same state and are at the same vertex of t make the same transition.

Lemma 2.6. *Let M be a weak alternating automaton. If M accepts an input t then there is a uniform accepting branch of $T(M, t)$.*

Proof. Fix any accepting branch β of $T(M, t)$. Let h be any finite history lying along β whose last state lies in a rejecting set Q_j. We shall prove that there is a bound J on the lengths of all extensions of h which lie along β and which *remain* in Q_j (i.e. all further states after the last state of h are in Q_j). To see this, let $T(\beta, h, Q_j)$ be the tree of all extensions of h which lie along β and which remain in Q_j. If $T(\beta, h, Q_j)$ contained arbitrarily long finite branches then $T(\beta, h, Q_j)$ would contain an infinite path (history) h^* by König's lemma. Thus, h^* would be rejecting; but h^* lies along β by definition, contradicting the fact that β is accepting. Thus, there is a bound J on the lengths of paths in $T(\beta, h, Q_j)$. If h is a finite history lying along β whose last state lies in a rejecting set Q_j, we call this bound the *Q_j-bound* of h. Note that if h' is a proper extension of h and h' remains in Q_j, then the Q_j-bound of h' is less than the Q_j-bound of h.

We shall uniformize β by choosing inductively, for each vertex v of the input tree and each state q, which history of an automaton present at v in state q is to be followed. A chosen history $h_{v,q}$ is *unaltered* and we say that it *controls* the other copies of M at v which are in state q. Any history other than the chosen one is marked "changed to follow $h_{v,q}$". This relation of transferring control is transitive. Thus, if a history h is "changed to follow h^*" and later h^* is "changed to follow h'", then, after the second instruction, the successors of h now follow h', and so on. The resulting uniform branch β' is the branch where each copy follows its control instructions.

It remains to show that β' can be chosen to be accepting. In order to make choices, fix any total ordering of the state set Q. (This ordering has nothing to do with the partial ordering on the subsets of Q.) The initial history q_0 of length zero is chosen. We successively consider vertices in the canonical well-ordering of the tree. Suppose that there is an unaltered history in β present at v in state q. If $q \in Q_j$, where Q_j is rejecting, consider only such histories whose Q_j-bound is minimal. Among these histories, choose the history $h_{v,q}$ which is least in the lexicographical ordering on histories induced by the ordering on states. If $q \in Q_i$, where Q_i is accepting, simply choose the least such history in the lexicographical ordering. The chosen history remains unaltered and all other histories present at v in state q are marked "changed to follow $h_{v,q}$".

Since β' is uniform by construction, it remains only to verify that it is accepting. Let h' be any history lying along β' and suppose that h' contains states from some rejecting set Q_j. Let h'_1 be the initial segment of h' up to the first state of Q_j. Let h^* be the unaltered history (possibly h'_1 itself) which h' follows. Note that h' cannot remain in Q_j longer than the Q_j-bound of h^*. Thus, h' leaves every rejecting set and β' is accepting. □

$\approx$ Miyano - Hayashi de-alternation construction

Lemma 2.7. *A weak alternating automaton can be simulated by a Büchi automaton B with $|B| \leqslant |M| 4^{|M|}$.*

Proof. Let $M = \langle \mathscr{L}(K \times Q), \Sigma, \delta, q_0, F \rangle$ be a weak alternating automaton. We can construct a Büchi automaton B simulating M as follows. As in the proof of Lemma 2.2, B uses a copy of $\mathscr{P}(Q)$ to keep track of the possibilities of machines running in M up to the information of current state. But we must now test that all the individual histories of machines in M are accepting. Let $\bar{F} = \{Q_0, \dots, Q_{r-1}\}$ be a consecutive list of those sets in the decomposition of Q which are rejecting. (This indexing has nothing to do with the partial order on the Q_i.) Let Z_r be a copy of the integers modulo r. An individual history h is rejecting if and only if it eventually stays in some set $Q_j \in \bar{F}$. The state set of B is $\mathscr{P}(Q) \times Z_r \times \mathscr{P}(Q)$. The first component is the *simulation track*, the Z_r-component is the *testing index* and the last component is the *testing track*. The initial state of B is $(\{q_0\}, 0, \phi)$. As in Lemma 2.4, if a copy of B reads a letter a at a vertex and S is the first component of the current state, then B selects a possible choice for each copy of M represented in S and in each direction d puts the collection S_d, resulting from S and the choices made, in the simulation track.

Suppose that the testing track contains ϕ. In this case, B advances the testing index (modulo r) by one, say to i, and, for each direction d, puts $S_d \cap Q_i$ in the testing track.

Suppose now that the testing index is i and the testing track contains a nonempty set $C \subseteq S$. In each direction d, B puts in the testing track the set C_d recording those copies in C which *remain* in a state from Q_i according to the selection of choices made for the simulation track. (Note that states not in C may give rise to states in Q_i but these are not recorded in C_d.) If $C_d = \phi$, the testing track is *discharged*. The acceptance condition for B is that one encounters ϕ infinitely often in the testing track. This condition exactly prevents an infinite history of a copy of M from forever remaining in a rejecting set Q_i. Thus, B accepts an input if and only if M does. □

3. The complexity of the weak monadic theory of the tree

Let k be a positive integer and let T_K denote the k-ary tree. In this section we show that weak alternating automata give a very simple proof of an $(n+1)$-exponential bound on the time complexity of deciding the truth of formulas in the class P_n consisting of these sentences of the weak monadic theory of T_K which are written in prenex normal form and which have no more than n blocks of quantifiers. Given a constant $c_1 > 1$ and a polynomial p, one can easily calculate c_2 such that $p(c_1^x) \leqslant c_2^x$. Thus, at each stage except the first, it will suffice to show that we can make the desired calculations in a polynomial function of the basic amount of time allowed.

We review our conventions on monadic logic. First of all, we shall assume that our logic contains *only* set variables. (This is in marked contrast to Robertson [14].) For each direction $d \in K$ there is a unary function symbol σ_d representing the set-valued successor function in the direction d. If X is a set of vertices of T_K then $X\sigma_d = \{xd, x \in X\}$ is the set of *d-successors* of elements of X. If w is a word in the σ_d and X is variable then Xw is a *term*. We suppose that our logic contains the binary relation $\subseteq$ and the unary relation $|X|=1$, saying that a set has cardinality one (cf. [8]).

We assume that the reader is familiar with the basic coding between formulas and regular languages used by Rabin [12]. Fix m variables, $X_1, \ldots, X_m$, and let $\Sigma = 2^m$. If t is a k-ary tree labelled from Σ, the ith component of the letter at a vertex v specifies whether or not v belongs to the set X_i. If $\varphi(X_1, \ldots, X_m)$ is a formula whose free variables are among $X_1, \ldots, X_m$, then the language L_φ associated with φ consists of all those k-ary trees which satisfy φ under the interpretation given above.

At the first stage, when given a quantifier-free formula $\varphi(X_1, \ldots, X_m)$, we must write down an alternating automaton M_φ accepting L_φ in such a way that the description is linear in $|\varphi|$, the length of φ. The slight technicality that the size of the alphabet Σ may be exponential in $|\varphi|$ requires us to "abbreviate" input letters. We first consider the case of an atomic formula.

For definiteness, suppose that $k=2$ and that σ_0 and σ_1 are denoted by 0 and 1. Consider, for example, an atomic formula such as $X0 \subseteq Z10$. Since a vertex cannot be both a 0-successor and a 1-successor, one may successively "cancel" the same successor symbol occurring at the right of terms. We replace the example by $X \subseteq Z1$. (If one arrives at an inclusion where the terms end in different successor symbols, replace the formula by "$Y = \phi$", where Y is the variable occurring on the left-hand side.) Our automaton must remember which points are 1-successors of elements of Z and verify that all members of X are such points. We need three states: q_0 indicating that the present vertex is not in $Z1$, q_1 indicating that the present vertex is in $Z1$, and a terminal state r indicating rejection. We can write the transition function as

$$\delta(q_0, x) = (0, r) \wedge (1, r), \qquad \delta(q_0, \neg x \wedge z) = (0, q_0) \wedge (1, q_1),$$

$$\delta(q_0, \neg x \wedge \neg z) = (0, q_0) \wedge (1, q_0), \qquad \delta(q_1, z) = (0, q_0) \wedge (1, q_1),$$

$$\delta(q_1, \neg z) = (0, q_0) \wedge (1, q_0), \qquad \delta(r, \_) = (0, r) \wedge (1, r),$$

where the symbol x indicates any letter with 1 in the X-component, $\neg x$ indicates a letter with 0 in the X-component, etc. Also, _ indicates an arbitrary letter. Modulo the abbreviation of input letters; this is a complete description of the automaton for the atomic formula $X0 \subseteq Z10$. We note that we have calculated this description in linear time.

An automaton for a formula $|X| = 1$ verifies that there is exactly one point with 1 in the X-component of its letter. The expression "$X = \phi$" is not formally part of our logic but is the condition which we wish to verify. For the negation $\neg\psi$ of a formula ψ, calculate the automaton for ψ and then dualize. Automata for $\varphi \vee \psi$ and $\varphi \wedge \psi$ are described by Lemma 2.2. This completes the description of the automaton for a quantifier-free formula.

Now assume inductively that a complete description of the automaton M_ψ for a formula $\psi(X_1, \dots, X_n, Y_1, \dots, Y_m)$ with b blocks of quantifiers has been constructed in the amount of time permitted. Note that we do *not* assume that the transition function of M_ψ is written in disjunctive normal form.

We now describe the automaton for $\exists X_1 \dots \exists X_n\, \psi(X_1, \dots, X_n, Y_1, \dots, Y_m)$. If $b=0$, first write out in full the transition function of M_ψ by eliminating the abbreviations of input letters. This takes exponential time. Now one application of Lemma 2.4 calculates the automaton M' corresponding to $\exists X_1 \dots \exists X_n\, \psi$. The number of states of M' is exponential in $|M_\psi|$ and a complete description of the transition function of M' can be written down in the time allowed. For the case $\forall X_1 \dots \forall X_n\, \psi$, first dualize the automaton M_ψ to obtain $\tilde{M}_\psi$ which corresponds to $\neg\psi$. Calculate the automaton M for $\exists X_1 \dots \exists X_n\, \neg\psi$ as above. Now dualize to obtain $\tilde{M}$ corresponding to $\forall X_1 \dots \forall X_n\, \psi$. This concludes the description of the procedure for passing from b to $b+1$ blocks of quantifiers.

Thus, given a formula ψ with n blocks of quantifiers, one can construct a weak alternating automaton M_ψ corresponding to ψ, where $|M_\psi|$ is n-exponential in $|\psi|$. One application of Lemma 2.7 constructs an equivalent Büchi automaton B_ψ, whose size is one-exponential in $|M_\psi|$. Finally, applying Rabin's polynomial algorithm for the emptiness problem for B_ψ establishes Theorem 1.2.

4. Weak alternating automata on $\mathbb{N}$

It seems to us that the following example is a good illustration of the way in which weak alternating automata work. Let $\Sigma = \{a, b\}$ and let $L \subseteq \Sigma^\omega$ be the set of words which contain an infinite number of b's. We want to construct a weak alternating automaton M working on the 1-ary tree $\mathbb{N}$ which accepts L. Since we are working on $\mathbb{N}$, we suppress the set of directions. Incidentally, the reader may easily convince himself that there does not exist a nondeterministic automaton using weak acceptance which accepts L. In contrast, the complementary language $\bar{L}$ consisting of those words containing only a finite number of b's is so simple that it can be accepted by a nondeterministic automaton A using weak acceptance. The automaton A simply guesses that it has seen the last b and will now read only a's, by passing from the initial state q_0 to a state q_1. If this condition is violated, A goes into a reject state q_2 and remains there. The transition diagram of A is given in Fig. 1. The accepting family

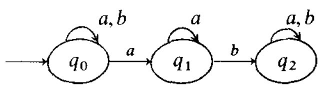

Fig. 1. The transition diagram of A.

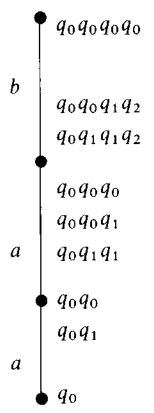

Fig. 2.

$\mathcal{F}$ consists of the single set $\{q_0, q_1\}$ which requires that A has indeed made its guess and, thereafter, sees only a's.

To accept L we now need simply take the machine $M = \tilde{A}$, which is the dual of A. Note that M is indeed an alternating automaton since there is an $\wedge$ in its transition function δ which is as follows:

$$\delta_a(q_0) = q_0 \wedge q_1, \qquad \delta_a(q_1) = q_1, \qquad \delta_a(q_2) = q_2,$$
$$\delta_b(q_0) = q_0, \qquad \delta_b(q_1) = q_2, \qquad \delta_b(q_2) = q_2.$$

Figure 2 shows the computation tree of M on an initial segment aab of an input word.

The accepting family $\bar{\mathcal{F}}$ of M consists of all subsets of $Q = \{q_0, q_1, q_2\}$ *except* $\{q_0, q_1\}$. The figure should reveal how M accepts L. At every moment, there will be a unique history containing only the state q_0. Whenever this machine reads the letter a, it gives rise to history $q_0 \dots q_0 q_1$. This history will contain exactly the states q_0 and q_1 as long as the letter a continues to be read. Reading a letter b "discharges" this history by completing the occurring set of states to Q. Thus, M accepts those words such that whenever the letter a occurs, it is eventually followed by the letter b. This is exactly the desired language L.

In view of McNaughton's theorem that every ω-regular language is accepted by a deterministic Muller automaton, our example is essentially the general case. This has also been observed by Lindsay [4].

Lemma 4.1. *Every ω-regular language L is accepted by a weak alternating automaton on $\mathbb{N}$.*

Proof. Let D be a deterministic Muller automaton accepting L. Since we have closure under union, we can reduce to the case where the accepting family of D consists of a single subset S of the set Q. The desired weak alternating automaton M works in the following way. In its first mode of operation, M simply simulates D. At any moment, M can guess that D will now stay in the set S forever. This guess causes M to enter its second mode of operation. In this mode M continues to simulate D and also, at every point, gives rise to a new "verifier". A verifier, which continues to simulate D, has the task of cycling *once* through all the states of S. When this task is accomplished, a verifier is "discharged" by entering an accepting dead-end state in which it always remains. If verifier sees a state not in S, it enters a "rejecting" state. The acceptance condition simply requires that M has made its guess and that a verifier enters the accepting state and does not reject. Since a verifier is started at every moment after the guess, D must pass through all the states of S infinitely often. □

References

[1] J.R. Büchi, On a decision method in restricted second-order arithmetic, in: *Proc. 1960 Internat. Congr. on Logic, Methodology and Philosophy of Science* (Stanford University Press, Stanford, CA, 1960) 1–11.

[2] A. Chandra, D. Kozen and L. Stockmeyer, Alternation, *J. ACM* **28** (1981) 114–133.

[3] Y. Gurevitch and L. Harrington, Trees, automata and games, in: *Proc. ACM Symp. on the Theory of Computing* (San Francisco, 1982) 60–65.

[4] P. Lindsay, On alternating ω-automata, *Theoret. Comput. Sci.* **43** (1986) 107–116.

[5] P. Lindsay, Alternation and ω-type Turing acceptors, *J. Comput. System Sci.* **36** (1988) 16–24.

[6] R. McNaughton, Testing and generating infinite sequences by a finite automaton, *Inform. and Control* **9** (1966) 521–530.

[7] D.E. Muller, Infinite sequences and finite machines, in: *Proc. 4th Ann. IEEE Symp. on Switching Circuit Theory and Logical Design* (1963) 3–16.

[8] D. E. Muller and P.E. Schupp, The theory of ends, pushdown automata and second-order logic, *Theoret. Comput. Sci.* **37** (1985) 51–75.

[9] D.E. Muller and P. Schupp, Alternating automata on infinite trees, *Theoret. Comput. Sci.* **54** (1987) 267–276.

[10] D.E. Muller, A. Saoudi and P.E. Schupp, Alternating automata, the weak monadic theory of the tree and its complexity, in: *Proc. ICALP'86*, Lecture Notes in Computer Science, Vol. 226 (Springer, Berlin, 1986) 275–283.

[11] D.E. Muller, A. Saoudi and P.E. Schupp, Weak alternating automata give a simple uniform explanation of why most temporal and dynamic logics are decidable in exponential time, in: *Proc. 3rd IEEE Ann. Symp. on Logic in Computer Science* (1988) 422–427.

[12] M.O. Rabin, Decidability of second-order theories and automata on infinite trees, *Trans. Amer. Math. Soc.* **141** (1969) 1–35.

[13] M.O. Rabin, Weakly definable relations and special automata, in: Bar-Hillel, ed., *Mathematical Logic and Foundations of Set Theory* (North-Holland, Amsterdam, 1970) 1–23.

[14] E.L. Robertson, Structure of complexity in the weak monadic second-order theory of the natural numbers, in: *Proc. 6th ACM STOC Conference* (1974) 161–171.