Tree Automata Make Ordinal Theory Easy

Thierry Cachat

Tree Automata Make Ordinal Theory Easy

Thierry Cachat

LIAFA/CNRS UMR 7089 & Université Paris 7, France
email: txc@liafa.jussieu.fr

Abstract. We give a new simple proof of the decidability of the First Order Theory of $(\omega^{\omega^i}, +)$ and the Monadic Second Order Theory of $(\omega^i, <)$, improving the complexity in both cases. Our algorithm is based on tree automata and a new representation of (sets of) ordinals by (infinite) trees.

1 Introduction

The connections between automata and logic have been fruitful for many years, see [13] for an introduction. In 1960 Büchi [4] showed that sets of finite words can be equivalently defined by Monadic Second Order (MSO) formulas and by finite automata. This gives in particular a decision procedure for this logic. This result has been extended later to other classes of structures and automata: MSO over infinite words and Büchi automata in [5], MSO over transfinite ordinals and transfinite automata [6], MSO over the full binary tree and Rabin automata in [18], MSO over graphs of the Caucal hierarchy and graph automata [7,15].

The decidability of the first order logic over the integers with addition, also known as Presburger arithmetic, can be easily obtained by using finite automata reading binary representation of numbers. A central idea in all these results is that formulas can be represented by automata: by induction on the formula one can build an automaton accepting exactly the models of the formula. See [22] for a clear exposition of many of the previous results.

More recently many authors have used automata to improve the complexity of certain decisions procedures. In particular in [14] the Presburger arithmetic is considered and in [16] the first order theory of the ordinals with addition.

We address in this article the decision algorithms for the First Order theory (FO) of $(\omega^{\omega^i}, +)$ and the Monadic Second Order theory (MSO) of $(\omega^i, <)$ for any integer i. Our proposal is to use finite labeled trees to represent ordinals and infinite trees to represent sets of ordinals. Then one can use tree automata to represent formulas (namely, all their models). In this way we improve the best known complexity, and we hope that our constructions are easier to understand than previous ones. Note that already MSO$(\omega, +)$ is undecidable, and the decision procedure for MSO$(\omega, <)$ has a non elementary lower bound. In [12] trees are already used to represent ordinals, but only termination of preocesses is considered. Our infinite trees in Section 3 are close to those in [3], where only inclusion of languages is considered.

The paper is organized as follows. The next section is concerned with the first order theory. After recalling definitions we present our tree encoding and our decidability proof. In Section 3 the encoding is adapted to the Monadic Second Order theory, before comparisons to other results and techniques are given.

2 Decidability of the First Order Theory of $(\omega^\omega, +)$

2.1 Definitions: Ordinal Addition, First Order Logic, Tree Automata

We assume basic knowledge about ordinals, see e.g. [20,21]. An *ordinal* is a well and totally ordered set. It is either 0 or a successor ordinal of the form $\beta + 1$ or a limit ordinal. The first limit ordinal is denoted ω. For all ordinal α: $\beta < \alpha \Leftrightarrow \beta \in \alpha$ and $\alpha = \{\beta : \beta < a\}$. The set of natural numbers is identified with ω. Recall e.g. that $1 + \omega = \omega = 2\omega$ and $\omega + \omega^2 = \omega^2$ but $\omega + 1 \neq \omega \neq \omega 2$. By the Cantor Normal Form theorem, for all $0 < \alpha < \omega^\omega$ there exist unique integers $p, n_0, n_1, \ldots, n_p$ such that $n_p > 0$ and

$$\alpha = \omega^p n_p + \omega^{p-1} n_{p-1} + \cdots + \omega^1 n_1 + n_0 \ .$$

Ordinal addition has an *absorption* property: for any $p < p'$, $\omega^p + \omega^{p'} = \omega^{p'}$. Given two ordinals $\alpha = \omega^p n_p + \cdots + \omega^1 n_1 + n_0$ and $\alpha' = \omega^{p'} n'_{p'} + \cdots + \omega^1 n'_1 + n'_0$ both written in Cantor Normal Form, the ordinal $\alpha + \alpha'$ is

$$\omega^p n_p + \cdots + \omega^{p'}(n_{p'} + n'_{p'}) + \cdots + \omega^1 n'_1 + n'_0 \ .$$

Formulas of the *First Order Logic* (FO) over $(\omega^\omega, +)$ are built from

- a countable set of individual variables $x, y, z, \ldots$
- the addition $+$, seen as a ternary relation,
- the Boolean connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$ and $\leftrightarrow$,
- first order quantification $\exists$ over individual variables ($\forall$ is seen as an abbreviation of $\neg\exists\neg$).

Example 1. The order relation $x \leq y$ can be easily defined as $\exists z : x + z = y$.
The relation $x < y$ is defined by $\neg(y \leq x)$.
The ordinal 0 is the only ordinal x such that $\neg\exists y : y < x$ or equivalently such that $x + x = x$.
The equality between x and y can be defined e.g. by $x \leq y \wedge y \leq x$.
The ordinal 1 is definable by $\phi(x) = (x > 0) \wedge \neg\exists y(0 < y \wedge y < x)$.

Example 2. The first limit ordinal, ω, is the only ordinal satisfying the formula

$$\begin{aligned}\varphi_1(x) = (x > 0) \wedge \forall y(y < x \rightarrow y + 1 < x) \wedge \\ \forall x'[(x' > 0) \wedge \forall y(y < x' \rightarrow y + 1 < x') \ \rightarrow \ x \leq x'] \ .\end{aligned}$$

Similarly and by induction ω^{i+1} is defined by

$$\begin{aligned}\varphi_{i+1}(x) = (x > 0) \wedge \forall y(y < x \rightarrow y + \omega^i < x) \wedge \\ \forall x'[(x' > 0) \wedge \forall y(y < x' \rightarrow y + \omega^i < x') \ \rightarrow \ x \leq x'] \ .\end{aligned}$$

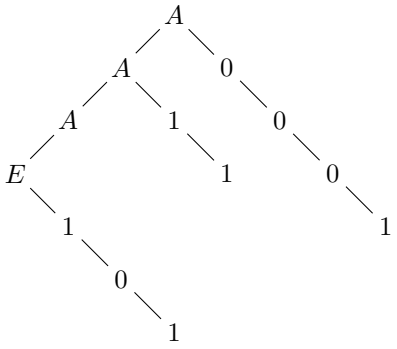
A *finite binary tree* T is a finite prefix closed subset of $\{a,b\}^*$. The root is the empty word ε, and for all $u \in \{a,b\}^*$, ua is the left successor of u and ub the right one. For simplicity we impose that each node has 0 or 2 successors: $\forall u \in \{a,b\}^*$, $ua \in T \Leftrightarrow ub \in T$. A leaf has no successor. Given a finite alphabet Σ, a Σ-labeled tree is a couple $\langle T, \lambda \rangle$ where T is a tree and λ is a function $\lambda : T \mapsto \Sigma$. A *tree automaton* is a tuple $(Q, \Sigma, \Delta, I, F)$ where Q is a finite set of states, Σ is a finite alphabet, $\Delta \subseteq Q \times \Sigma \times Q \times Q$ is the transition relation, $I \subseteq Q$ and $F \subseteq Q$ are the sets of initial and accepting states ("final states"). A Σ-labeled tree is accepted by such a tree automaton iff there exists a run $\rho : T \mapsto Q$ such that

$$\rho(\varepsilon) \in F, \text{ and } \forall u \in T: \text{ either } (\rho(u), \lambda(u), \rho(ua), \rho(ub)) \in \Delta \text{ or } u \text{ is a leaf } (ua \notin T) \text{ and } \rho(u) \in I \ .$$

This presentation is unusual: the labels at the leafs are not important in our constructions. These (bottom up) tree automata can be determinized by a usual subset construction. By exchanging initial and final states they can be seen as top down automata.

2.2 Binary Trees Representing Ordinals

Ordinals less than ω^ω can be easily represented by finite binary trees. The tree representing $\alpha = \omega^p n_p + \cdots + \omega^1 n_1 + n_0$ (where $n_p > 0$) has a leftmost branch of length (at least) p. At depth i on this branch a right branch is attached, holding the binary encoding of the number n_i. For example the ordinal $\omega^3.5 + \omega.3 + 8$ is represented essentially as the following tree.



The letter E marks the last position where there is a non zero right branch. We allow *all* possible ways to add dummy symbols # at the bottom of the tree. There are not represented on the picture, but they are needed for every node to have 0 or 2 successors (not 1). To be more formal the set of tree representations of a given ordinal $\alpha = \omega^p n_p + \cdots + \omega^1 n_1 + n_0$ is exactly the language accepted by the tree automaton to be defined next. The initial state is $q_\#$, the accepting state q_0.

If $\sigma_i^0 \sigma_i^1 \ldots \sigma_i^{m_i}$ is the (little endian) binary encoding of n_i: $n_i = \sum_{j=0}^{m_i} 2^j \sigma_i^j$, then the transitions are:

$$(q_i, A, q_{i+1}, p_i^0) \text{ if } i < p \text{ and } n_i > 0 \qquad (p_i^j, \sigma_i^j, q_\#, p_i^{j+1}) \text{ if } j < m_i$$
$$(q_i, A, q_{i+1}, q_\#) \text{ if } i < p \text{ and } n_i = 0 \qquad (p_i^j, \sigma_i^j, q_\#, q_\#) \text{ if } j = m_i$$
$$(q_i, E, q_\#, p_i^0) \text{ if } i = p \qquad (q_\#, \#, q_\#, q_\#)$$

In the special case where $\alpha = 0$ we have a transition $(q_0, \#, q_\#, q_\#)$. We denote T_α the tree coding an ordinal α.

2.3 Decidability Using Tree-Automata

We adapt a well known method for proving decidability of logic theories. A single tree over the alphabet $\{A, E, \#, 0, 1\}^k$ represents the values of k variables by superposing k corresponding trees (and adding dummy symbols $\#$). For every formula $\psi \in \mathrm{FO}(\omega^\omega, +)$ with free variables $x_1, \ldots, x_k$ we want to build a tree automaton over the alphabet $\{A, E, \#, 0, 1\}^k$ such that a tree is accepted by this automaton iff the corresponding valuation of the variables satisfies ψ. This can be done by induction on the formula. The case of Boolean connectives is easy using standard automata techniques of product and complementation, see [10]. Existential quantification results in projecting out the corresponding variable. The main point is to define an automaton recognizing the relation $x + y = z$, and this is easy with our coding.

In the following transitions $\begin{smallmatrix}\#\\1\\0\end{smallmatrix}$ represents a letter from $\{A, E, \#, 0, 1\}^3$ where the first component is $\#$, the second is 1 and the third is 0. These components are letters from T_x, T_y and T_z respectively. The symbols σ, δ represent digits from $\{0, 1\}$ and $*$ represents any letter. The accepting state is r. Because of the absorption property, above symbol E of T_y, trees T_y and T_z must coincide. State q_y checks that T_y and T_z coincide on the corresponding right branch. Similarly q_x checks that T_x and T_z coincide. State r_y checks that T_y and T_z coincide on the rest of the tree. Similarly r_x checks that T_x and T_z coincide.

$$(r, \begin{smallmatrix}\#\\\#\\\#\end{smallmatrix}, q_\#, q_\#) \qquad (q_\#, \begin{smallmatrix}\#\\\#\\\#\end{smallmatrix}, q_\#, q_\#)$$

$$(r, \begin{smallmatrix}A\\A\\A\end{smallmatrix}, r, q_y) \qquad (q_y, \begin{smallmatrix}*\\\sigma\\\sigma\end{smallmatrix}, q_\#, q_y) \qquad (q_y, \begin{smallmatrix}*\\\#\\\#\end{smallmatrix}, q_\#, q_y) \qquad (q_y, \begin{smallmatrix}\#\\\#\\\#\end{smallmatrix}, q_\#, q_\#)$$

$$(r, \begin{smallmatrix}E\\A\\A\end{smallmatrix}, r_y, q_y) \qquad (r_y, \begin{smallmatrix}\#\\A\\A\end{smallmatrix}, r_y, q_y) \qquad (r_y, \begin{smallmatrix}\#\\E\\E\end{smallmatrix}, q_\#, q_y)$$

$$(r, \begin{smallmatrix}A\\E\\A\end{smallmatrix}, r_x, q_0) \qquad (r_x, \begin{smallmatrix}A\\\#\\A\end{smallmatrix}, r_x, q_x) \qquad (r_x, \begin{smallmatrix}E\\\#\\E\end{smallmatrix}, q_\#, q_x)$$

$$(r, \begin{smallmatrix}E\\E\\E\end{smallmatrix}, q_\#, q_0) \qquad (q_x, \begin{smallmatrix}\sigma\\*\\\sigma\end{smallmatrix}, q_\#, q_x) \qquad (q_x, \begin{smallmatrix}\#\\*\\\#\end{smallmatrix}, q_\#, q_x) \qquad (q_x, \begin{smallmatrix}\#\\\#\\\#\end{smallmatrix}, q_\#, q_\#)$$

The states q_0 and q_1 are in charge of the binary addition with carries.

$$(q_0, \begin{smallmatrix}\sigma\\\delta\\\sigma\ \mathrm{XOR}\ \delta\end{smallmatrix}, q_\#, q_{\sigma\mathrm{AND}\ \delta}) \qquad (q_0, \begin{smallmatrix}\sigma\\\#\\\sigma\end{smallmatrix}, q_\#, q_x) \qquad (q_0, \begin{smallmatrix}\#\\\sigma\\\sigma\end{smallmatrix}, q_\#, q_y)$$

$$(q_1, \begin{smallmatrix}\sigma\\\delta\\\neg(\sigma\ \mathrm{XOR}\ \delta)\end{smallmatrix}, q_\#, q_{\sigma\mathrm{OR}\ \delta}) \qquad (q_1, \begin{smallmatrix}\sigma\\\#\\\neg\sigma\end{smallmatrix}, q_\#, q_\sigma) \qquad (q_1, \begin{smallmatrix}\#\\\sigma\\\neg\sigma\end{smallmatrix}, q_\#, q_\sigma)$$

$$(q_0, \begin{smallmatrix}\#\\\#\\\#\end{smallmatrix}, q_\#, q_\#) \qquad (q_1, \begin{smallmatrix}\#\\\#\\1\end{smallmatrix}, q_\#, q_\#)$$

Some details are omitted here for the sake of simplicity. In state q_y, after reading $\#$ on the first component, one should check that only $\#$ appears. And the most

significant bit of each number should be 1 to have a standard representation. It is left to the reader to add intermediate states to check that the trees T_x, T_y and T_z are well formed. That is needed when the automata defining T_x, T_y or T_z were obtained by complementation (see below). Let Tower stand for the "tower of exponentials" function, *i.e.*, $\text{Tower}(0,n) = n$ and $\text{Tower}(k+1,n) = 2^{\text{Tower}(k,n)}$.

Theorem 1. *The First Order Theory of* $(\omega^\omega, +)$ *is decidable in time* $\mathcal{O}(\mathit{Tower}(n,c))$*, for some constant* c*, where* n *is the length of the formula.*

To our knowledge the best known algorithm for deciding $\text{FO}(\omega^\omega, +)$ goes via a (linear) reduction to the Weak Monadic Second Order logic of $(\omega^\omega, <)$, which in turn is decidable in time $\mathcal{O}(\text{Tower}(6n, c'))$ [16]. See Section 3 for the definition of this logic.

Proof. By induction on the formula $\psi \in \text{FO}(\omega^\omega, +)$ one can construct a tree automaton $\mathcal{A}_\psi$ accepting exactly all valuations satisfying ψ. A valuation is here a tree labeled over $\{A, E, \#, 0, 1\}^k$, where k is the number of free variables in ψ.

- If ψ is an atomic proposition, it is of the form $x + y = z$ and we have seen how to construct $\mathcal{A}_\psi$.
- If ψ is of the form $\neg\psi'$, by induction $\mathcal{A}_{\psi'}$ is constructed. We can determinize and complement it [10], and intersect with the automaton describing the allowed representation of ordinals, to obtain $\mathcal{A}_\psi$.
- If ψ is of the form $\psi_1 \wedge \psi_2$, by induction $\mathcal{A}_{\psi_1}$ and $\mathcal{A}_{\psi_2}$ are constructed. Rearrange the order of the variables, build the product of $\mathcal{A}_{\psi_1}$ and $\mathcal{A}_{\psi_2}$. Declare a state $\langle q_1, q_2\rangle$ final iff both q_1 and q_2 are final. [10]
- Similarly if ψ is of the form $\psi_1 \vee \psi_2$, rearrange the variables, build the product and declare a state $\langle q_1, q_2\rangle$ final iff q_1 or q_2 is final.
- If ψ is of the form $\psi_1 \rightarrow \psi_2$, first determinize $\mathcal{A}_{\psi_1}$ and $\mathcal{A}_{\psi_2}$, then build the product, and declare a state $\langle q_1, q_2\rangle$ final iff $(q_1 \in F_1) \Rightarrow (q_2 \in F_2)$.
- Similarly if ψ is of the form $\psi_1 \leftrightarrow \psi_2$, determinize $\mathcal{A}_{\psi_1}$ and $\mathcal{A}_{\psi_2}$, build the product, and declare a state $\langle q_1, q_2\rangle$ final iff $(q_1 \in F_1) \Leftrightarrow (q_2 \in F_2)$.
- If ψ is of the form $\exists x \psi'$, then the input alphabet of the automaton $\mathcal{A}'_\psi$ is $\{A, E, \#, 0, 1\}^k$, where k is the number of free variables in ψ'. Project out the component corresponding to the variable x to get the automaton $\mathcal{A}_\psi$ that non-deterministically guesses the value of x.

At the end of the procedure it remains to determine whether $\mathcal{A}_\psi$ accepts a tree (labeled over an empty alphabet). This can be done in polynomial time by marking the states reachable from the initial states. Note that the cases of conjunction and disjunction does not need determinization. This is possible with a bottom up tree automaton, where the acceptance condition is checked only once, at the root.

Like for many automata based decision procedures, the most expensive step is the determinization of automata. It costs exponential time and the result is an automaton of exponential space. The number of steps of the construction is the number of Boolean connectives and quantifiers of the formula, whereas the constant c is essentially the number of states of the automaton for $x + y = z$.

To slightly improve the complexity one can easily construct directly automata recognizing the relations $x = y$, $x < y$, $x \leq y$ of Example 1. Of course every ordinal ω^i can also be easily defined directly, without using the formulas of Example 2.

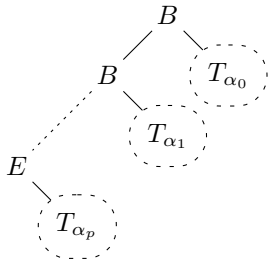
It is also possible to replace $\rightarrow$ and $\leftrightarrow$ by equivalent formulas using only $\neg, \wedge$ and $\vee$ and to push negations symbols inwards (using De Morgan's laws, etc). See [14] for a careful discussion about the cost of these transformations: they can increase the length of the formula and add new quantifiers. Here we do not assume that the formula is in prenex normal form.

2.4 Beyond ω^ω

By using a new letter (B) in the alphabet, it is possible to encode ordinals greater than ω^ω. Any ordinal $\beta < \omega^{\omega^2}$ can be uniquely written in the form

$$\omega^{\omega.p}\alpha_p + \cdots + \omega^{\omega.2}\alpha_2 + \omega^\omega\alpha_1 + \alpha_0 \ , \text{ where } p < \omega, \alpha_i < \omega^\omega, \alpha_p > 0 \ .$$

and we can encode it as a tree where each T_{α_i} appears as a subtree. Namely the leftmost branch will have length p. At depth i on this branch the tree T_{α_i} is attached. The skeleton of the tree is depicted on the right. It is easy to see that a tree automaton can recognize the relation $x + y = z$, and that the proof of Theorem 1 carries over. Note that the letter B is used here only for clarity, one could use A instead.



This can be generalized by induction, and for all $i < \omega$ we can encode ordinals less than ω^{ω^i}.

Theorem 2. *For each $i < \omega$ there exists a constant c_i such that the First Order Theory of $(\omega^{\omega^i}, +)$ is decidable in time $\mathcal{O}(Tower(n, c_i))$, where n is the length of the formula.*

Note that the height of the tower of exponentials do not depend on i, and that c_i is linear in i. When considering $\mathrm{FO}(\omega^{\omega^i}, +)$, even the ordinal 1 is coded by a tree of depth at least i: we need each tree to have the same skeleton to allow the automaton to proceed the addition locally. It was already noticed (without proof) in [11] that any ordinal $\alpha < \omega^{\omega^\omega}$ is tree-automatic, that is to say that the structure $(\alpha, <)$ —without addition— is definable using tree-automata. Moreover [11] proves that any tree-automatic ordinal is less than ω^{ω^ω}.

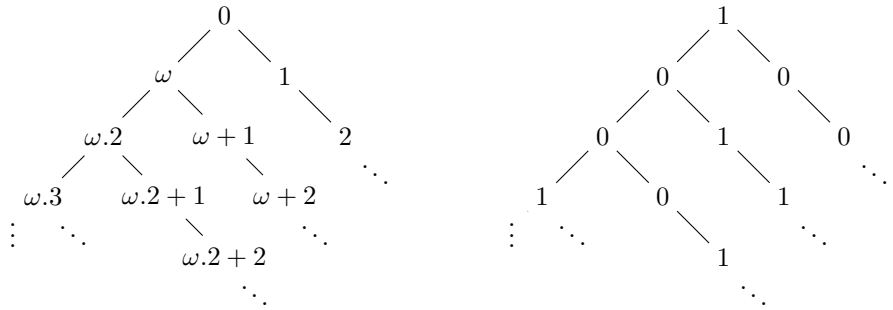
3 Monadic Second Order Theory of $(\omega^k, <)$

In this section we use full *infinite* binary trees. They are given by a mapping $\lambda : \{a, b\}^* \mapsto \Sigma$ for some finite alphabet Σ. Their domain is always $\{a, b\}^*$ so we do not need to mention it. One can adapt the idea of Section 2 to represent *sets* of

ordinals. Given a subset $S \subseteq \omega^2$ it is represented by the tree $\lambda : \{a,b\}^* \mapsto \{0,1\}$ such that

$$\forall i,j \geq 0 : \lambda(a^i b^j) \in \{0,1\} \qquad \forall u \notin a^*b^* : \lambda(u) = \#$$
$$\forall i,j \geq 0 : \lambda(a^i b^j) = 1 \;\Leftrightarrow\; \omega.i + j \in S \ .$$

So positions are associated to ordinals according to the left tree of the next picture. Accordingly the right tree represents the set $\{0, \omega{+}1, \omega{+}2, \omega.2{+}2, \omega.3\}$. In this way one can represent any subset of ω^2.



Languages of infinite trees can be defined by top down *Muller automata* [17]. A Muller automaton $\mathcal{A}$ is a tuple $(Q, \Sigma, \Delta, I, \mathcal{F})$ where Q, Σ, Δ are the same as in Section 2, $I \subseteq Q$ is the set of initial states and $\mathcal{F} \subseteq \mathscr{P}(Q)$ is the acceptance component ($\mathscr{P}(Q)$ is the powerset of Q). A *run* of $\mathcal{A}$ on a Σ-labeled tree λ is a labeling $\rho : \{a,b\}^* \mapsto Q$ such that

$$\rho(\varepsilon) \in I \text{ and } \forall u \in T : (\rho(u), \lambda(u), \rho(ua), \rho(ub)) \in \Delta \ .$$

A run is accepting iff on each (infinite) branch of the run, the set of states appearing infinitely often is equal to one of the $F \in \mathcal{F}$. A tree is accepted iff there exists an accepting run. Muller automata cannot be determinized in general, but the class of languages accepted by Muller automata is closed under union, intersection, projection and complementation. In particular an automaton accepting all trees where only one node is labeled by 1 cannot be deterministic: it has to guess where is the 1.

Formulas of the *(full) Monadic Second Order Logic* (MSO) over $(\omega^\omega, <)$ are built from

- a countable set of first order variables $x, y, z, \ldots$
- a countable set of second order variables (in capitals) $X, Y, Z, \ldots$
- the order relation $(x < y)$ over first order variables,
- the membership relation $(x \in X)$, also written $X(x)$,
- the Boolean connectives $\neg$, $\wedge$ and $\vee$ ($\rightarrow$ and $\leftrightarrow$ are seen here as abbreviations),
- existential quantification ($\exists$) over first order *and* second order variables ($\forall$ is seen as an abbreviation of $\neg\exists\neg$).

The syntax of the *Weak Monadic Second Order Logic* (WMSO) is exactly the same, the difference is that second order variables are interpreted by *finite* subsets of the structure.

Example 3. The formulas of Example 1 above are also expressible in $\mathrm{MSO}(\omega^\omega, <)$ because they do not need the addition. One can also define a relation $x = y + 1$. The next formula shows that the set of even ordinals (less than ω^ω) can be defined in MSO:

$$\exists X : \forall x \ \ (x \in X \leftrightarrow \neg(x + 1 \in X)) \wedge (\neg\exists y(x = y + 1) \rightarrow x \in X) \ .$$

We consider trees labeled over $\{0, 1\}^k$ where k is the number of first-order *and* second-order free variables. It should be clear that one can construct Muller automata recognizing the relations $x \in X$ and $x < y$. Note that for each first-order variable the automaton has to check that only one node in the tree is labeled by 1, *i.e.*, x is treated as a second-order variable $X = \{x\}$. See [2] for a clear exposition of a similar construction in the framework of ordinal automata.

Theorem 3. *The Monadic Second Order Theory of* $(\omega^2, <)$ *is decidable in time* $\mathcal{O}(\mathit{Tower}(n, c))$*, for some constant* c*, where* n *is the length of the formula.*

Recall that the upper bound of [16] is in $\mathcal{O}(\mathrm{Tower}(6n, 1))$ for the *weak* variant $\mathrm{WMSO}(\omega^\omega, <)$. Already $\mathrm{MSO}(\omega, <)$ has a lower bound in $\Omega(\mathrm{Tower}(n, d))$ for some constant $d > 0$ [19], so our bound is really tight.

Proof (sketch). We use again the well known method by induction on the structure of the formula $\psi \in \mathrm{MSO}(\omega^2, +)$.

- If ψ is an atomic proposition, it is clear how to construct $\mathcal{A}_\psi$.
- If ψ is of the form $\neg\psi'$, $\psi_1 \vee \psi_2$ or $\psi_1 \wedge \psi_2$, we use the fact that languages of Muller tree automata are closed under complementation, union and intersection.
- If ψ is of the form $\exists x\psi'$ or $\exists X\psi'$, we use the fact that languages of Muller tree automata are closed under projection.

The most expensive step is the complementation, it can be done in exponential time, and the result has also exponential size, see [17,22]. At the end the test of emptiness is also exponential.

Note that for the case of disjunction the automaton has to guess at the root which subformula can be true. For a formula $\psi = \psi_1 \rightarrow \psi_2$ we cannot do better than transform it into $\neg\psi_1 \vee \psi_2$. It is not correct to simply build the product of $\mathcal{A}_{\psi_1}$ and $\mathcal{A}_{\psi_2}$ and adapt the acceptance component, because the acceptance condition is checked independently on each branch.

Using an idea similar to that of Section 2.4, one can attach ω trees of the form presented above to a left-most branch to encode subsets of ω^3. This can be extended by induction to ω^i for all $i < \omega$.

Theorem 4. *For each* $i < \omega$ *there exists a constant* c_i *such that the Monadic Second Order Theory of* $(\omega^i, <)$ *is decidable in time* $\mathcal{O}(\mathit{Tower}(n, c_i))$*, where* n *is the length of the formula.*

In other works such as [8,1] the emphasis is not placed on the complexity, but it seems that the complementation of ordinal automata is double exponential. It is open how to extend the tree encoding to subsets of ω^ω.

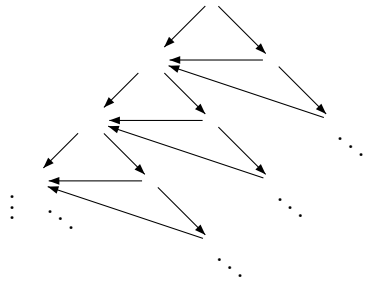
3.1 MSO-interpretation. Comparison with Ordinal Automata

It is possible to put a different light on the previous constructions. The MSO theory of the full binary tree [22], called *S2S*, is build from the atomic propositions $S_a(x,y)$, $S_b(x,y)$ and $P(x)$, where S_a is the relation "left successor", S_b is "right successor" and P is a predicate that indicates that the label of a node is 1. In other words, given a labeled infinite tree $\lambda : \{a,b\}^* \mapsto \{0,1\}$ and $x, y \in \{a,b\}^*$:

$$S_a(x,y) \Leftrightarrow y = x.a \ , \qquad S_b(x,y) \Leftrightarrow y = x.b \ , \qquad P(x) \Leftrightarrow \lambda(x) = 1 \ .$$

Recalling the left figure in page 7, the order among the ordinals/positions in the tree can be interpreted in S2S. That is, one can write a formula $\phi(x,y)$ such that $\phi(x,y)$ is true iff the ordinal of position x is less than that of y. It is easy if one first write formulas $\phi_a(x,y)$ and $\phi_b(x,y)$ that checks that y is a left descendant of x (resp. right descendant).

Alternatively one can see the ordering ω^2 as the transitive closure of the graph pictured on the right. Nevertheless concerning complexity it is better to construct dedicated automata as in the proof of Theorem 3. In other words the graphs of the orderings ω^i, $i < \omega$, are prefix-recognizable graphs [9]. It is open whether graphs of greater ordinals are in the Caucal hierarchy.



The usual proof that $\mathrm{MSO}(\omega^\omega, <)$ is decidable uses ordinal automata reading ordinal words. An ordinal word of length α is a mapping $\alpha \mapsto \Sigma$, where Σ is a finite alphabet. An ordinal automaton has a state space Q, usual one-step transitions of the form $(q, \sigma, q') \in Q \times \Sigma \times Q$ and *limit transitions* of the form $(P, q') \in \mathscr{P}(Q) \times Q$, see e.g. [2]. They are a generalization of Muller (word) automata. A run is a mapping $\rho : \alpha + 1 \mapsto Q$. For a successor ordinal $\beta + 1$, $\rho(\beta + 1)$ is defined in the usual way. For a limit ordinal β, the state $\rho(\beta)$ is obtained by a limit transition according to the states appearing infinitely often "before" β.

We want to point out that a run of a Muller automaton on a tree representing $S \subseteq \omega^2$ is very similar to a run of length ω^2 of an ordinal automaton. Consider a node v at depth i on the left most branch. It corresponds to an ordinal $\omega.i$. The right-most branch from v must satisfy the Muller condition, and the state reached at the left successor of v is like the state reached at the limit transition at $\omega.(i+1)$. In this way we get a new proof that languages accepted by ordinal automata are closed under complementation, restricted to the case of words of length ω^j, for all $j < \omega$.

Comparing both approaches, we see that tree automata can not be determinized in general, they can be complemented, however, using an exponential construction. On the other side ordinal automata can be determinized (and complemented) using a doubly exponential construction, due to the nesting of Muller conditions. We are not aware of a better complementation algorithm for ordinal

automata, see e.g. [8] for a more general result. The transformation from a tree automaton to an equivalent ordinal automaton according to our coding is very simple. The state space remains the same except for one extra final state for the last limit transition. If $(q, \lambda, q_a, q_b) \in \Delta$ in the tree automaton, add transitions (q, λ, q_b), and (P, q_a) for all $P \in \mathcal{F}$, where $\mathcal{F}$ is the Muller acceptance condition. The other way around is more complicated because the tree automaton has to guess what states are going to be visited infinitely often on the right branch, and then allow only these states to be visited infinitely often.

3.2 Weak MSO and FO

We introduce here new material to compare MSO and FO. Any ordinal β can be written in a unique way in the form

$$2^{\gamma_{n-1}} + \cdots + 2^{\gamma_0} \ , \quad \text{where} \quad (\gamma_{n-1}, \ldots, \gamma_0)$$

is a strictly decreasing sequence of ordinals. The set $\{\gamma_{n-1}, \ldots, \gamma_0\}$ is called the *2-development* of β. For example $2^\omega = \omega$, $2^{\omega.i+j} = 2^{\omega.i}.2^j = \omega^i.2^j$, $2^{\omega^2} = (2^\omega)^\omega = \omega^\omega$. Let E be the binary relation on ordinals such that $(x, y) \in E$ iff $x = 2^\gamma$ for some γ that belongs to the 2-development of y. It is known [6] that the theories WMSO$(\alpha, <)$ and FO$(2^\alpha, +, E)$ are equireducible in linear time. Recall that the (weak) theory WMSO is the monadic theory where only finite sets are considered. This mean that any formula of one of the logics can be translated into an equivalent formula of the other logic in linear time.

To extend Theorem 2 to the decidability of FO$(2^\alpha, +, E)$ for $\alpha = \omega^i$, we only need a tree automaton recognizing the relation E. The fact that $x = 2^\gamma$ is equivalent in our coding to the fact that exactly one label is 1 in the tree T_x, and $(x, y) \in E$ if moreover the same node is labeled by 1 in the tree T_y. The automaton recognizing E needs only three states, so the complexity bounds of Theorem 2 are not changed.

On the other side we have proved decidability of the *full* MSO theory of $(\omega^i, <)$ in Theorem 4. It remains to interpret WMSO in MSO. It is known in general how to construct a Muller tree automaton that checks that only finitely many nodes of a tree are labeled by 1. It is possible with only 2 states and can be used to adapt the proof of Theorem 3 to WMSO. Using this reduction, the complexity of the decision procedure of WMSO$(\omega^i, <)$ is in $\mathcal{O}(\text{Tower}(n + 1, c'_i))$ for some (new) constant c'_i. Alternatively, using the property that every subset of an ordinal is also well ordered, it is possible to write an MSO formula that checks that a set of ordinals is finite. This formula should be used together with each second order quantification.

An extension of the previous tree-automata techniques to higher ordinals such as MSO$(\omega^\omega, <)$ would gives also tree-automata techniques for WMSO$(\omega^\omega, <)$ and then FO$(\omega^{\omega^\omega}, +, E)$, which is impossible [11] (see end of Section 2).

Related to the Cantor Normal Form (see Section 2), *any* ordinal β can yet be written in a unique way in the form

$$\alpha = \gamma.\omega^\omega + \omega^p n_p + \omega^{p-1} n_{p-1} + \cdots + \omega^1 n_1 + n_0 \ .$$

where $n_p > 0$. The ω-character of α is the sequence $(\sigma, n_p, \ldots, n_0)$ where $\sigma = 0$ if $\gamma = 0$, and $\sigma = 1$ if $\gamma > 0$. The theories $WMSO(\alpha, <)$ and $WMSO(\beta, <)$ are equal iff α and β have the same ω-character [6]. It follows that $\mathrm{FO}(2^\alpha, +, E)$ and $\mathrm{FO}(2^\beta, +, E)$ are equal iff α and β have the same ω-character.

4 Perspectives

We gave a new decision procedure for $\mathrm{FO}(\omega^{\omega^i}, +)$ and $\mathrm{MSO}(\omega^i, <)$ achieving better complexity bounds. We hope our constructions are easy to understand. As a byproduct we have a new proof of the complementation of ordinal automata restricted to words of length ω^i.

According to [11] (see end of Section 2) and Section 3.2 it is not possible to extend the tree-automata techniques to higher ordinals. But we would like to extend it to other linear orderings. A bi-infinite word is a mapping from the *relative* integers to a finite alphabet. It is easy to represent it as an infinite tree where only the right most and the left most branches are relevant. It seems easy to represent also orderings like $-\omega$ or $\omega \times (-\omega)$. Using a special letter, one could mark branches where the "reverse" ordering $-w$ is used. We conjecture that one can extend the results of Section 3 to more general linear orderings than just ordinals, and give a new proof of the results of [8].

Acknowledgments

Many thanks to Wolfgang Thomas for always saying that we have a proof without theorem (the proof that tree automata are closed under intersection, complementation and projection), to him and Stéphane Demri for pointing out some useful references and to the referees.

References

1. Nicolas Bedon. Finite automata and ordinals. *Theor. Comput. Sci.*, 156(1&2):119–144, 1996.
2. Nicolas Bedon. Logic over words on denumerable ordinals. *J. Comput. System Sci.*, 63(3):394–431, 2001.
3. Véronique Bruyère, Olivier Carton, and Géraud Sénizergues. Tree automata and automata on linear orderings. In *Proceedings of WORDS'03*, volume 27 of *TUCS Gen. Publ.*, pages 222–231. Turku Cent. Comput. Sci., Turku, 2003.
4. J. Richard Büchi. Weak second-order arithmetic and finite automata. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 6:66–92, 1960.
5. J. Richard Büchi. On a decision method in restricted second order arithmetic. In *Proceedings of the 1960 International Congress on Logic, Methodology and Philosophy of Science*, pages 1–11. Stanford University Press, 1962.
6. J. Richard Büchi. Decision methods in the theory of ordinals. *Bull. Amer. Math. Soc.*, 71:767–770, 1965.

7. Thierry Cachat. Higher order pushdown automata, the Caucal hierarchy of graphs and parity games. In *Proceedings of the 30th International Colloquium on Automata, Languages, and Programming, ICALP'03*, volume 2719 of *LNCS*, pages 556–569. Springer, 2003.
8. Olivier Carton and Chloe Rispal. Complementation of rational sets on scattered linear orderings of finite rank. In Martin Farach-Colton, editor, *LATIN*, volume 2976 of *LNCS*, pages 292–301. Springer, 2004.
9. Didier Caucal. On infinite terms having a decidable monadic theory. In *Proceedings of the 27th International Symposium on Mathematical Foundations of Computer Science 2002, MFCS 2002*, volume 2420 of *LNCS*, pages 165–176. Springer, 2002.
10. Hubert Comon, Max Dauchet, Remi Gilleron, Florent Jacquemard, Denis Lugiez, Sophie Tison, and Marc Tommasi. Tree automata techniques and applications. Available on: `http://www.grappa.univ-lille3.fr/tata`, 1997. release October, 1rst 2002.
11. Christian Delhommé. Automaticité des ordinaux et des graphes homogènes. *C. R. Math. Acad. Sci. Paris*, 339(1):5–10, 2004.
12. Nachum Dershowitz. Trees, ordinals and termination. In Marie-Claude Gaudel and Jean-Pierre Jouannaud, editors, *TAPSOFT*, volume 668 of *LNCS*, pages 243–250. Springer, 1993.
13. Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research*, volume 2500 of *LNCS*. Springer, 2002.
14. Felix Klaedtke. On the automata size for Presburger arithmetic. In *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science (LICS 2004)*, pages 110–119. IEEE Computer Society Press, 2004. A full version of the paper is available from the author's web page.
15. Orna Kupferman and Moshe Y. Vardi. An automata-theoretic approach to reasoning about infinite-state systems. In E. A. Emerson and A. P. Sistla, editors, *Proceedings of the 12th International Conference on Computer Aided Verification, CAV'00*, volume 1855 of *LNCS*, pages 36–52. Springer, 2000.
16. Françoise Maurin. Exact complexity bounds for ordinal addition. *Theoretical Computer Science*, 165(2):247–273, 1996.
17. Frank Nießner. Nondeterministic tree automata. In Grädel et al. [13], pages 135–152.
18. Michael O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.
19. Klaus Reinhardt. The complexity of translating logic to finite automata. In Grädel et al. [13], pages 231–238.
20. Joseph G. Rosenstein. *Linear orderings*. Academic Press Inc. [Harcourt Brace Jovanovich Publishers], New York, 1982.
21. Wacław Sierpiński. *Cardinal and ordinal numbers*. Second revised edition. Monografie Matematyczne, Vol. 34. Państowe Wydawnictwo Naukowe, Warsaw, 1965.
22. Mark Weyer. Decidability of S1S and S2S. In Grädel et al. [13], pages 207–230.