Thin MSO with a Probabilistic Path Quantifier

Mikołaj Bojańczyk

University of Warsaw, Warsaw, Poland

Abstract

This paper is about a variant of MSO on infinite trees where:

- there is a quantifier "zero probability of choosing a path $\pi \in 2^\omega$ which makes $\varphi(\pi)$ true";
- the monadic quantifiers range over sets with countable topological closure.

We introduce an automaton model, and show that it captures the logic.

1998 ACM Subject Classification F.4.1 Mathematical Logic

Keywords and phrases Automata, MSO, infinite trees, probabilistic temporal logics

Digital Object Identifier 10.4230/LIPIcs.ICALP.2016.96

1 Introduction

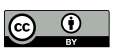
The ambient topic of this paper is MSO on infinite binary trees, extended by a quantifier $\mathsf{zero}\pi\ \varphi(\pi)$ which says that there is zero probability of choosing a path π in the tree so that $\varphi(\pi)$ is true. Here we assume that each bit (i.e. turn) in the path is chosen independently at random. This logic was introduced by Michalewski and Mio in [10], where the decidability of satisfiability was left open.

That satisfiability question is not solved here, but we make a small step in its direction. We consider a fragment of the logic, called TMSO+zero, standing for *thin* MSO+zero. In this fragment, the monadic set quantifiers are restricted to sets which are thin in the following sense: a set of nodes is *thin* if there are countably many paths which visit it infinitely often. For example, every path (when seen as a set of nodes) is thin, and every finite set is thin. In the logic TMSO+zero, one has existential and universal quantification over nodes and thin sets of nodes, as well as the probabilistic path quantifier zero. Being thin is definable in MSO, and therefore without the zero quantifier, the logic would be a special case of MSO, and with the zero quantifier it is a special case of the logic from [10].

The contribution of this paper is the definition of an automaton model, called zero automata, and a proof that every formula of TMSO+zero can be effectively translated to an equivalent zero automaton.

Motivation

The first source of motivation for this paper is the study of probabilistic temporal logics [1, 8, 13, 4]. An important example is the logic PCTL. It is an open problem whether this logic has decidable satisfiability. Much of the difficulty stems from the ability of talking about probabilities like 1/2 or 1/3. If one can only compare probabilities to 0 or 1, which is in the spirit of our logic TMSO+zero, then we get *qualitative* PCTL, whose satisfiability was shown decidable by Brázdil, Forejt, Kretínský and Kucera in [4]. Actually, the qualitative fragment of PCTL, as well as stronger qualitative logics like PCTL*, can be straightforwardly formalised in TMSO+zero, and therefore, by the main result of this paper, translated into zero automata. Another example that we discuss later in the paper is the probabilistic version of tree automata by Carayol, Haddad and Serre [6]; these are also a special case of zero automata.



43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016).
Editors: Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi;
Article No. 96; pp. 96:1–96:13

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

The second source of motivation is trying to find a robust classes of languages of infinite words or trees which remains decidable (e.g. with respect to satisfiability). The point of departure is MSO, with its famous decidability results by Büchi [5] and Rabin [11]. One way of departing from that point is to add unary predicates, e.g. extending MSO over ω-words by a predicate "x is a position of the form $n!$", see [12] for a survey. Another way is to add new quantifiers. Due to the strength of MSO, it is not so easy to come up with a quantifier extending MSO that is not obviously undecidable, and yet not already definable in MSO. For example, a nice quantifier is "there exist uncountably many sets with a given property" – but as shown in [2], this quantifier does not add to the expressive power of MSO. A logic that *does* properly extend MSO is MSO+U, which is an extension of MSO by a quantifier which can say that a given property is true for finite sets of unbounded size. The logic is itself undecidable, but has many decidable fragments, typically variants of weak MSO. See [3] for a survey of MSO+U and related logics, including the cost logics of Colcombet [7]. The logic studied in this paper, TMSO+zero, is another example of a logic that is not contained in MSO (and even contains MSO, if the logic is extended by allowing an outermost layer of non-weak existential set quantifiers, which does not affect decidability of satisfiability).

2 The logic and the automaton

This section describes the two main models used in the paper: the logic TMSO+zero and zero automata. The following sections discuss how the logic is translated into the automaton.

Tree notation

The logics and automata of this paper describe properties of possibly infinite binary labelled trees. We treat a node in a tree as a sequence in 2^*, with 2 denoting the set of directions $\{0, 1\}$. Define a *tree* over an alphabet Σ to be a partial function $t : 2^* \to \Sigma$ whose domain is closed under prefixes. The special case when function is total is called a *complete tree*, but we do allow incomplete trees, e.g. trees with finite domains. We use standard terminology for trees: node, root, left child, right child, leaf, ancestor and descendant. In our definition, a node might have a right child but not a left child. We write $\mathsf{trees}\Sigma$ for the set of trees over Σ.

Probability measure over paths

A *path* is defined to be a sequence in 2^ω, which is viewed as an infinite sequence of left or right turns. An equivalent definition is that a path is an ancestor-closed set of nodes that is totally ordered by the ancestor relation. When saying that a path is contained in a set of nodes, or contains a node, the second definition is used. When talking about the probability of a subset of 2^ω we use the *coin-flipping* measure, i.e. we assume that each bit is chosen independently at random, with 0 and 1 having equal probability. The probability is defined at least for all Borel subsets of 2^ω.

▶ **Definition 1.** We say a set $\Pi \subseteq 2^\omega$ has *zero probability* if it is contained in a Borel set with coin-flipping measure zero.

The sets of paths that will appear in the logic TMSO+zero will always be Borel, so the closure under subsets in the above definition will not play much of a role.

2.1 The logic

Before defining the logic TMSO+zero, we discuss the probability-free fragment TMSO.

Thin MSO without the zero quantifier

A set of nodes $X \subseteq 2^*$ is called *thin* if its *closure* defined by

$$\bar{X} \stackrel{\text{def}}{=} \{\pi \in 2^\omega : \pi \text{ passes through infinitely many nodes from } X\}$$

is countable. For example, every finite set is thin, because it has empty closure, and every path is thin, when viewed as a set of nodes, because its closure has one path. Thin sets are closed under arbitrary intersections and finite unions, but not under countable unions, because the countable set of all nodes has all paths in its closure, and is therefore not thin.

The logic *thin* MSO, denoted by TMSO, is the variant of MSO as in Rabin's theorem, except that set quantifiers range only over thin sets. The syntax of the logic is the same as for MSO from Rabin's theorem: there are two types of variable in the logic: *node variables*, which range over nodes in the domain of the input tree, and *(thin) set variables*, which range over thin subsets of the domain of the input tree. There are binary predicates for the left and right child relations, and there is a unary predicate for every label in the input alphabet. By the Cantor-Bendixson theorem, a set of nodes X is thin if and only if one cannot find a subset $Y \subseteq X$ such that Y, when ordered by the descendant relation, is a complete binary tree. Since this alternative characterisation can be formalised in MSO, it follows that TMSO is a fragment of MSO in terms of expressive power. On the other hand, TMSO is at least as expressive as WMSO with path quantifiers.

As far as the author knows, the logic TMSO was not considered explicitly in the literature so far, and it might be interesting to examine its expressive power, e.g. prove that it is strictly weaker than MSO and maybe, in the long run, find an algorithm which inputs a formula of MSO and decides if the formula is equivalent to some formula in TMSO. This investigation, however, is not the topic of the present paper. The present paper is about extending TMSO with a quantifier for zero probability.

Thin MSO with the zero quantifier

We now define the main topic of this paper, i.e. the logic TMSO+zero. First we explain why our point of departure for adding the zero quantifier is TMSO and not some other fragment of MSO. The reason is that TMSO is the strongest logic we could find such that the set quantifiers commute with the probabilistic quantifier in a way which will be made more precise in Section 6. The key observation reason is this: if the domain of the input tree is thin, then it has countably many paths, and therefore the zero quantifier can be eliminated because it always says "yes".

A parameter in the definition of TMSO+zero is a family zero of subsets of 2^ω. The example we have in mind is that zero is the sets with zero probability according to Definition 1, but the results will also work for other choices of zero. The logic TMSO+zero is the extension of the logic TMSO defined above, by adding a quantifier, called zero, which binds a thin set variable π, and such that

$$\mathsf{zero}\pi\ \varphi(\pi)$$

is true if zero contains the set of paths π which are contained in the domain of the input tree and make $\varphi(\pi)$ true, assuming that a path is treated as a set of nodes. (Formally speaking, the path π is seen as a set of nodes when evaluating $\varphi(\pi)$, and as an element of 2^ω when measuring how many paths π make $\varphi(\pi)$ true.)

▶ **Example 2.** Consider an alphabet $\{a, b\}$. The following formula says that zero contains the set of paths that visit at least one a:

$$\mathsf{zero}\pi\ \exists x\ (x \in \pi \wedge a(x)).$$

If the parameter zero is prefix independent (see Definition 7 for a more precise treatment) and does not contain the set 2^ω of all paths, then the above formula is equivalent to $\forall x\ \neg a(x)$, and therefore the zero quantifier can be avoided.

▶ **Example 3.** Consider an alphabet $\{a, b\}$. The following formula says that zero contains the set of paths which visit b finitely often:

$$\mathsf{zero}\pi\ \exists x\big(x \in \pi \wedge \forall y\ (y \geq x \wedge y \in \pi \Rightarrow b(x))\big)$$

If zero is our guiding example of zero probability, the negation of the above formula says that the Büchi condition is satisfied with probability one. As shown in [6], Theorem 21, the property above is not definable in MSO.

▶ **Example 4.** The reduction from qualitative PCTL* in Theorem 5 from [10] produces formulas where set quantification is only used for paths. Therefore, qualitative PCTL* is a special case of TMSO+zero.

Beyond Thin MSO with the zero quantifier

In the logic TMSO+zero, the set variables are restricted to thin sets. The obvious question is about the more general case, where set variables range over arbitrary sets of nodes, not necessarily thin ones. As mentioned in the introduction, the more general logic was introduced in [10], under the name MSO$+\forall_\pi^{=1}$, and the authors asked about decidability of its satisfiability problem. A long term project for this research is to find out if the satisfiability problem for the more general logic is decidable – or not. In this paper we only begin the project, by studying the thin variant. One scenario is that the thin variant is decidable, but the non-thin variant is undecidable, which would be similar to the situation for MSO+U, where weak variants are decidable, but the full logic is undecidable. However, one should not take the analogy with MSO+U too far: e.g. the thin variant of MSO+U would already be undecidable, because MSO+U is undecidable already for ω-words.

Another natural version of MSO with probability would be to choose a subset of 2^* at random, with each node chosen independently, and then have a quantifier that says there is zero probability of finding a subset with a given property. This logic was proved undecidable in [10], already for ω-words (which can be seen as a special case of TMSO), and the undecidability proof works also for formulas of the form

$$\text{there is zero probability of choosing a set } X \subseteq \mathbb{N} \text{ which makes } \varphi(X) \text{ true,}$$

where $\varphi(X)$ is a formula of first-order logic that defines a set of ω-words over alphabet 2. Therefore, it seems that this kind of probabilistic quantifier is doomed to undecidability.

2.2 The automaton

Having defined the logic TMSO+zero, we define our main automaton model, which is called a zero *automaton*. Like in the logic TMSO+zero, a parameter of the semantics for the automaton is a family zero of subsets of 2^ω. The idea is that the automaton extends a nondeterministic parity automaton with the ability to say that the set of paths satisfying the parity condition belongs, or does not belong, to zero.

▶ **Definition 5.** The syntax of a zero automaton is a tuple

$$\underbrace{Q}_{\text{states}} \qquad \underbrace{\Sigma}_{\text{input alphabet}} \qquad \underbrace{I \subseteq Q}_{\text{initial states}} \qquad \underbrace{\bigcup_{C \subseteq 2} \delta_C \subseteq Q \times \Sigma \times Q^{|C|}}_{\text{transitions}},$$

with all components finite, together with a total order on Q and four subsets

$$Q_{\text{all}}, Q_{\text{zero}}, Q_{\text{nonzero}}, Q_{\text{seed}} \subseteq Q.$$

The idea behind the transitions is that $\delta_{\{0,1\}}$ is used for those nodes which have both children defined, but e.g. $\delta_{\{1\}}$ is used for nodes where only the right child is defined, and $\delta_\emptyset$ is used for leaves.

The semantics are defined as follows. The automaton is run on a tree over the input alphabet, which might not necessarily be complete. A *run* of the automaton is a tree labelled by states with the same domain as the input tree, which is consistent with the transition relation in the following sense: if a node x is in the domain, and we define

$$C \stackrel{\text{def}}{=} \{i \in 2 : xi \text{ is in the domain}\}$$

then there must be a transition in δ_C which relates the state in x, the label of x in the input tree, and the states in the children of x that are in the domain. A tree is accepted if it admits a run which has the initial state in the root and is accepting in the following sense. Define the *maxinf state* on a path in a run to be the maximal state that appears infinitely often on the path. When talking about a maximal state, we refer to the total order on states that is given in the syntax of the automaton. A run ρ is accepting if all of the following conditions hold, assuming that $\mathsf{paths}\,\rho \subseteq 2^\omega$ denotes the set of paths contained in ρ:

1. **all paths acceptance condition:** every path from $\mathsf{paths}\,\rho$ has maxinf in Q_{all}; and
2. **zero acceptance condition:** zero contains the set of paths from $\mathsf{paths}\,\rho$ which have maxinf state in Q_{zero}; and
3. **nonzero acceptance condition:** for every node x in the run with state $q \in Q_{\text{seed}}$:

$$\mathsf{zero} \not\ni \{\pi \in \mathsf{paths}\,\rho : \begin{cases} \pi \text{ passes through } x, \text{ and} \\ \pi \text{ sees only states } < q \text{ after } x, \text{ and} \\ \pi \text{ has maxinf state in } Q_{\text{nonzero}} \end{cases} \}$$

An automaton is called *zeroless* if Q_{zero} is empty (which makes the zero condition vacuously true) and *seedless* if there are no seed states, i.e. Q_{seed} is empty (which makes the nonzero condition vacuously true). In particular, a zeroless and seedless automaton is the same thing as a parity automaton, which proves the zero automata are at least as powerful as MSO.

▶ **Example 6.** Assume that zero is probability zero as in Definition 1. Consider the special case of a zero automaton where Q_{all} is all states and Q_{seed} is empty. A run is accepting if and only if there is zero probability of having maxinf state in Q_{zero}. Equivalently, the probability of having maxinf state outside Q_{zero} is one. Languages recognised by such automata are the *qualitative tree languages* from [6]. The class of *positive tree languages* from [6] is obtained when Q_{all} and Q_{zero} are empty, and the initial state is used only once in the root, is maximal in the total order, and is the unique seed state.

3 Fat Cantor

In this section, we illustrate the logic and automaton with an extended example. Let us fix zero to be probability zero according to Definition 1. Define the *fat Cantor language* to be the set of complete trees over the alphabet $\{a, b\}$ which satisfy the following property:

$$\underbrace{\neg \mathsf{zero}\pi\big(\forall x\ x \in \pi \wedge b(x)\big)}_{\text{nonzero probability of avoiding } a} \quad \wedge \quad \underbrace{\forall x \exists y\ y \geq x \wedge a(y)}_{a\text{'s are dense}}$$

Note that "avoiding a" is a Borel property of paths, and therefore "nonzero probability of avoiding a" means that the sets of paths avoiding a have defined positive probability. This argument will be true in general for our logic – for every fixed input tree, any property of paths definable in the logic will be Borel, and therefore not belonging to zero will mean that it there is defined and positive probability.

The fat Cantor language is nonempty. To construct a tree in the fat Cantor language, choose a fast growing sequence of natural numbers

$$n_1 < n_2 < n_3 < \cdots$$

and then choose a tree (which is unique up to reordering siblings) where a labels are found only at depths from the sequence above, and every node at depth n_i has a unique descendant at depth n_{i+1} with label a. If the sequence (n_i) grows fast enough, then there is nonzero probability of avoiding a. Let us now argue that the fat Cantor language contains no regular tree, i.e. no tree with finitely many nonisomorphic subtrees. Suppose then that t is a regular tree, with n distinct subtrees. If a's are dense in this tree, it follows from regularity that every node has a descendant at distance at most n that has label a. This means there is some constant $\epsilon > 0$ such that for every interval $I \subseteq \mathbb{N}$ of n consecutive positions, the probability of a path visiting a at depth from I is at least ϵ. These events are independent for disjoint intervals, and therefore the probability of seeing a at least once, and even infinitely often, is 1. Summing up: the fat Cantor language is nonempty but contains no regular trees.

Fat Cantor automaton

We now show a zero automaton which recognises the fat Cantor language described above. To simplify notation, we define an automaton which works only on complete trees, i.e. it recognises the intersection of the fat Cantor language with the set of complete trees. In particular, when talking about transitions, we only consider transitions δ_C for $C = \{0, 1\}$.

The input alphabet is $\{a, b\}$. The automaton has four states, totally ordered as follows:

$$\underbrace{q_a}_{\text{already saw } a} < \underbrace{q_1}_{\text{searching for } a} < \underbrace{q_2}_{\text{not searching for } a} < \underbrace{q_0}_{\text{initial state}}$$

The automaton begins in state q_0 in the root, this state will not be visited again during the run. When the automaton is in state q_i with $i \in \{0, 1, 2\}$ and it reads a node with label b, then it sends q_1 to some child and q_2 to the other child, as witnessed by the following transitions:

$$(q_i, b, q_j, q_k) \quad \text{for } i \in \{0, 1, 2\} \text{ and } \{j, k\} = \{1, 2\}.$$

Choosing which child gets q_1 and which child gets q_2 is the only source of nondeterminism in this automaton. When the automaton sees letter a, it sends q_a to both children regardless of

its current state, and q_a is a sink state that cannot be left, as witnessed by the following transitions:

$$(q, a, q_a, q_a) \quad \text{for all } q \in Q \qquad (q_a, a, q_a, q_a) \qquad (q_a, b, q_a, q_a)$$

Since q_0 is used only once in the root, and q_a is a sink state, it follows that on every path either q_a is seen from some point on, or q_a is never seen and the maxinf state is one of q_1, q_2. The acceptance condition is defined by the following sets:

$$Q_{\text{all}} = \{q_a, q_2\} \qquad Q_{\text{zero}} = \emptyset \qquad Q_{\text{nonzero}} = \{q_1, q_2\} \qquad Q_{\text{seed}} = \{q_0\}$$

Because Q_{zero} is empty, every run satisfies the zero acceptance condition. The state q_0 appears only once in the root, and therefore it is never used as a maxinf state. By choice of Q_{all}, the state q_1 is forbidden as a maxinf state, which means that in an accepting run, every path eventually stabilises on either q_a or q_2. Since the only way of leaving q_1 is by seeing an a letter, it follows that a's must be dense. The only seed state is the initial state, which is used only once in the root, and is also the most important state. Therefore, a run satisfies the nonzero acceptance condition if and only if its there is nonzero probability of having maxinf state in $\{q_1, q_2\}$, which means there is nonzero probability of avoiding a.

4 From logic to automata

The main technical result of this paper is that every formula of TMSO+zero can be effectively translated to an equivalent zero automaton. The result works not just for zero probability, but for other choices of zero, as described in the following definition.

▶ **Definition 7.** For a family zero of subsets of 2^ω, consider the following properties:

1. **σ-ideal:** zero is closed under subsets and countable union;
2. **atomless:** zero contains all singletons;
3. **prefix independence:** every set $\Pi \subseteq 2^\omega$ satisfies

$$\Pi \in \mathsf{zero} \Leftrightarrow i\Pi \in \mathsf{zero} \qquad \text{for every } i \in 2$$

4. **recurrent nonzero:** there is a zero automaton which recognises the language

$$\{t \in \mathsf{trees}\{1,2,3\} : \text{for every subtree, the set of paths with maxinf 2 is } \notin \mathsf{zero}\}$$

In the recurrent nonzero condition, it is important that the trees are not necessarily complete. For such a tree, a subtree is obtained by shifting the root to some node in the domain. In particular, if a tree belongs to the language from the recurrent nonzero condition, then it cannot have any leaves.

Here is the main result of this paper.

▶ **Theorem 8.** *Let* zero *be a family of subsets of* 2^ω *satisfying conditions 1-4 in Definition 7. Then for every formula of* TMSO+zero *one can compute an equivalent* zero *automaton.*

The proof has three steps. In Section 5, we show closure properties of languages recognised by zero automata, of which the most interesting is closure under intersection. In Section 6, we show that the logic TMSO+zero has the same expressive power as a certain transducer model. In Section 7, we complete the proof of the theorem, by translating transducers into zero automata. The results in Section 5 and 6 only use properties 1-3 in Definition 7, while Section 7 uses also property 4.

The following corollary shows the main application of Theorem 8.

▶ **Corollary 9.** *Let* zero *be the subsets of* 2^ω *that have zero probability in the sense of Definition 1. Then for every formula of* TMSO+zero *one can compute an equivalent* zero *automaton.*

Other examples of zero which can be shown to satisfy the assumptions of Theorem 8 include "countable sets of paths [2]" and "meagre sets of paths [9]". These other examples are less interesting because they can already be formalised in MSO alone, i.e. parity automata are sufficient. Theorem 8 can be seen as an alternative way of recovering the results from [2, 9]: one only needs to check that the assumptions of Theorem 8 are satisfied for a particular choice of zero, and that zero automata can be captured by MSO. In view of the results from [2, 9], we have only one example of zero that satisfies the assumptions of Theorem 8, and which strictly extends MSO, namely probability zero.

5 Closure properties of zero automata

This section is about closure properties of the class of languages recognised by zero automata. We show that this class is closed under positive Boolean operations – with intersection being by far the more interesting case. We do not know if languages recognised by zero automata are closed under complementation. If they would be, then zero automata would have exactly the same expressive power as full MSO+zero.

Define a *Mealy machine* to be a deterministic finite automaton on words over some input alphabet Σ, where every transition is labelled by a letter from some output alphabet Γ. Such a machine can be run on a finite word, yielding a length preserving function $\Sigma^* \to \Gamma^*$, it can also be run on an ω-word, yielding a function $\Sigma^\omega \to \Gamma^\omega$, or finally it can be run on all paths in a tree, yielding a function $\mathsf{trees}\Sigma \to \mathsf{trees}\Gamma$ which does not change the domain of the tree. The last case will be called a *tree transducer recognised by a Mealy machine.*

▶ **Lemma 10.** *Languages recognised by* zero *automata are closed under union, as well as images and inverse images under tree transducers recognised by Mealy machines.*

Proof sketch. The lemma does not require any closure properties from the set zero. For union, we use disjoint union of automata (and gluing the initial state). For images use nondeterminism, and for both images and inverse images use a Cartesian product construction to simulate the Mealy machine in the state space of the zero automaton. Note that state spaces in zero automata are ordered. Therefore we impose some random total order on a Mealy machine, and in the Cartesian product we use a lexicographic ordering, with the order on the original zero automaton being more important. ◀

We now show another closure property, which is closure under factorisations, as described below. Define a *factor* to be a set of nodes that is connected with respect to the child relation. In particular, a factor has a unique *root*, i.e. a unique node which is least with respect to the descendant ordering. If X is a factor, then define the *restriction to* X of a tree t to be the tree obtained from t by keeping only the nodes from X. We now show that if L is a language recognised by a zero automaton, then there is a zero automaton which inputs a tree together with a decomposition into disjoint factors, and checks that L contains every tree obtained by restricting the input tree to one of the factors in the partition.

We begin by describing how a decomposition into factors is given on the input. If X is a set of nodes, then define an X-factor to be a set of nodes obtained by taking some $x \in X$ and adding all descendants y such that $(x..y]$ is disjoint with X, where $(x..y]$ denotes proper descendants of x that are (not necessarily proper) ancestors of y. By abuse of notation, we

define an X-factor of a tree t to be any tree obtained from t by restricting it to some X-factor. Finally, if X is a set of nodes in a tree $t \in \mathsf{trees}\Sigma$, then define $t \otimes X \in \mathsf{trees}(\Sigma \times 2)$ to be the tree obtained from t by extending the label of each node by a bit indicating membership in X.

▶ **Lemma 11** (Factorisation Lemma). *Assume that* zero *satisfies conditions 1–3 in Definition 7. If $L \subseteq \mathsf{trees}\Sigma$ is recognised by a* zero *automaton, then so is*

$$\{t \otimes X : t \in \mathsf{trees}\Sigma \text{ and } X \text{ is a set of nodes in } t \text{ such that } L \text{ contains every } X\text{-factor of } t\}.$$

The main idea in the proof is that to use the "nested" character of the nonzero acceptance condition; here by nesting we mean that the paths contributing to the nonzero condition are cut off whenever a more important state is seen.

We finish this section by stating the most challenging result, which is closure under intersection, as stated in the following lemma.

▶ **Lemma 12** (Intersection Lemma). *Assume that* zero *satisfies conditions 1-3 in Definition 7. Then languages recognised by* zero *automata are closed under intersection.*

The proof has several steps. One of these steps, namely the first step, is showing that languages recognised by zero automata are closed under intersection with languages recognised by zero automata which do not use the nonzero acceptance condition. The first step uses McNaughton's Latest Appearance Record construction.

6 Transducers

To prove Theorem 8, we use a transducer characterisation of the logic TMSO+zero. The transducer characterisation is an "if and only if" characterisation, unlike the translation in the main Theorem 8.

Transducers

Define a *tree transducer* to be any function $\mathsf{trees}\Sigma \to \mathsf{trees}\Gamma$ which does not change the domain of the input tree. Our goal is to show each language definable MSO+zero can be described by composing transducers of certain basic types. To model a language as a transducer, we use the following definition.

▶ **Definition 13.** For a tree language $L \subseteq \mathsf{trees}\Sigma$, define

$$\mathsf{trans}L : \mathsf{trees}\Sigma \to \mathsf{trees}2,$$

called the *characteristic transducer of* L, to be the transducer which labels each node of the input tree by a bit saying whether or not the subtree rooted in that node belongs to L.

We define the *combination* $t_0 \otimes t_1$ of two trees t_0, t_1 over possibly different alphabets Σ_0, Σ_1 but with equal domains, to be the unique tree over $\Sigma_0 \times \Sigma_1$ which projects to each t_i on the i-th coordinate. In the following theorem, composition of transducers is defined as for functions, while the combination of two transducers f_1, f_2 with the same input alphabet but possibly different output alphabets is the transducer $t \mapsto f_1(t) \otimes f_2(t)$.

▶ **Theorem 14.** *Assume that* zero *has the closure properties 1-3 from Theorem 8. Then a tree language is definable in* TMSO+zero *if and only if its characteristic transducer belongs to the smallest class of transducers which is closed under composition and combination, and which contains the following transducers:*

1. **Zero base.** *The characteristic transducers of all languages of the form:*

$$\mathsf{Z}_n \stackrel{def}{=} \{t \in \mathsf{trees}\{1,\dots,n,\bot\} : \mathsf{zero} \ni \{\pi \in \mathsf{paths}\, t : \begin{cases} \pi \text{ does not visit } \bot, \text{ and} \\ \pi \text{ has even maxinf} \end{cases} \}\}$$

2. **Zeroless base.** *The characteristic transducers of all languages definable in* TMSO.
3. **Child number transducer.** *Transducers of the form* $\mathsf{trees}\Sigma \to \mathsf{trees}2$ *which map each node to its child number, with the convention that the root gets label* 0.
4. **Mealy machine on trees.** *Transducers recognised by Mealy machines.*

The difficult implication is from logic to transducers; here we use the composition method. Intuitively speaking, the above theorem shows that formula of TMSO+zero can be decomposed into parts that do not talk about zero at all, and into the very basic property Z_n.

7 From transducers to zero automata

In this section we complete the proof of Theorem 8, by showing that the transducers from the previous section can be compiled into zero automata. We say that a tree transducer f is *recognised* by a zero automaton if there is a zero automaton recognising the set of trees $t \otimes f(t)$ where t ranges over all input trees for the tree transducer.

▶ **Lemma 15.** *Transducers recognised by* zero *automata are closed under composition, combination and include the child number transducers, transducers induced by Mealy machines, and the characteristic transducers of all languages definable in* TMSO.

Proof sketch. For composition, the automaton guesses the intermediate result, and checks both underlying transducers in parallel, using the Intersection Lemma. Combination also uses intersection. For the child-number transducers, Mealy machines and characteristic transducers of languages definable in TMSO, one observes that their corresponding languages are definable in MSO, and zero automata generalise nondeterministic parity tree automata. ◀

By Theorem 14 and the above lemma, in order to prove Theorem 8 it suffices to show that zero automata recognise the characteristic transducers of the languages of the form Z_n as used in Theorem 14. By unraveling the definitions, we need to show the following lemma.

▶ **Lemma 16.** *For every* $n \in \mathbb{N}$ *there is a* zero *automaton recognising the set of trees*

$$t \otimes s \qquad \text{with } t \in \mathsf{trees}\{1,\dots,n,\bot\},\ s \in \mathsf{trees}2$$

such that for every node x*, its label in* s *is* 1 *iff* Z_n *contains the subtree of* t *rooted in* x*.*

Proof. Let L be the language in the statement of the lemma. For a tree $t \otimes s$, define a $\bot$-factor to be a maximal factor contained in the domain of the tree that does not use label $\bot$ in t. It is not difficult to see that $t \otimes s$ belongs to L if and only if: (a) every node with label $\bot$ in t has label 1 in s; and (b) every $\bot$-factor belongs to L. Condition (a) can be easily checked by a parity automaton, so thanks to the Intersection Lemma it suffices to produce a zero automaton which checks (b). By the Factorisation Lemma, it suffices to find a zero automaton which tests memberhip in L for individual $\bot$-factors.

Summing up, we can assume without loss of generality that t does not use label $\bot$ at all. Therefore, in the rest of the proof, we show a zero automaton which recognises the language L restricted to the case where $t \in \{1,\dots,n\}$.

For $i \in \{1, \ldots, n\}$, consider the function

$$f_i : \mathsf{trees}\{1, \ldots, n\} \to \mathsf{trees}\{1, 2, 3\} \qquad \text{label of } x \text{ in } f_i(t) = \begin{cases} 1 & \text{if label of } x \text{ in } t \text{ is } < i \\ 2 & \text{if label of } x \text{ in } t \text{ is } = i \\ 3 & \text{if label of } x \text{ in } t \text{ is } > i \end{cases}.$$

We will only use this function for even i. For $t \in \mathsf{trees}\{1, \ldots, n\}$, define nonzero$(t)$ to be the set of nodes in t whose subtree does not belong to Z_n. In terms of this definition, a tree $t \otimes s$ belongs to L if and only if nonzero(t) is exactly the nodes that have label 0 in s. Also, condition 4 from Definition 7 says that there is a zero automaton recognising the language

$$\mathsf{N} \stackrel{\text{def}}{=} \{t \in \mathsf{trees}\{1, 2, 3\} : \text{nonzero}(t) \text{ is all nodes of } t\}$$

▶ **Claim 17.** *Let $t \in \mathsf{trees}\{1, \ldots, n\}$ and $s \in \mathsf{trees2}$ be trees with the same domain. Then $t \otimes s \in L$ if and only if one can find an ancestor closed set of nodes $\{X_i\}_i$, with i ranging over even numbers in $\{1, \ldots, n\}$, such that the following conditions hold:*

1. *a node has label 0 in s if and only if it belongs to some X_i;*
2. *for every even $i \in \{1, \ldots, n\}$, restricting $f_i(t)$ to the nodes from X_i yields a tree in N;*
3. $\mathsf{zero} \ni \{\pi \in \mathsf{paths}\, t : \pi$ *has even t-maxinf and sees 0 finitely often in s*$\}$

Before proving the claim, let us observe how it implies the lemma. Since a zero automaton can nondeterministically guess the sets X_i, it suffices to show that there is a zero automaton which checks conditions 1, 2, 3 in the claim. By the Intersection Lemma, it suffices to check each condition individually. Condition 1 is definable in MSO. Condition 2, for any fixed i, follows from the assumption that N is recognised by a zero automaton and the Factorisation Lemma. For condition 3, it is straightforward to construct a zero automaton – it essentially copies the labels from t into its states, except that nodes with label 0 in s trigger a state which is maximal in the total order. It remains to prove the claim.

Proof. We begin with the following observation, which follows from the assumption that zero satisfies conditions 1-3 in Definition 7. For every $t \in \mathsf{trees}\{1, \ldots, n\}$, the set nonzero$(t)$ is closed under ancestors and a node x belongs to nonzero(t) if and only if

$$\mathsf{zero} \not\ni \{\pi \in \mathsf{paths}\, t : \begin{cases} \pi \text{ is contained in nonzero}(t), \text{ and} \\ \pi \text{ passes through } x, \text{ and} \\ \pi \text{ has even } t\text{-maxinf} \end{cases} \} \tag{1}$$

By definition of the tree transducers f_i, a path has even t-maxinf if and only if it has even $f_i(t)$-maxinf for some even i. Therefore, by closure of zero under countable – and therefore also finite – unions, we see that

$$\text{nonzero}(t) = \bigcup_i \text{nonzero}(f_i(t)), \tag{2}$$

where i ranges over even numbers in $\{1, \ldots, n\}$.

Let us now prove the claim.

Let us begin with the bottom-up implication. From condition 2 it follows that every node in X_i belongs to nonzero(t). From condition 1 it follows that all nodes with label 0 are in nonzero(t). From condition 1, it follows that the set of nodes with label 0 in s is closed under ancestors. Therefore, condition 3 implies that for every node with label 1 in s is outside

nonzero(t). Thus nonzero(t) is exactly the nodes which have label 0 in s, which means that $t \otimes s \in L$.

Consider the top-down implication. Our assumption is that nonzero(t) is exactly the nodes which have label 0 in s. Define X_i to be nonzero($f_i(t)$). By (2), we see that condition 1 in the statement of the claim holds. From (1) applied to the trees $f_i(t)$, we get condition 2. To prove condition 3, by definition of nonzero(t) and prefix independence of zero, we know that every node $x \notin$ nonzero(t) satisfies

$$\mathsf{zero} \ni \{\pi \in \mathsf{paths}\, t : \pi \text{ passes through } x \text{ and has even } t\text{-maxinf}\}$$

Since nonzero(t) is ancestor closed, it follows that a path passes through some $x \notin$ nonzero(t) if and only if it sees 0 in s finitely often. Therefore, by closure of zero under countable unions, we get condition 3 in the statement of the claim. ◀
◀

8 Conclusion

We have proved that, under certain conditions on zero, every formula of the logic TMSO+zero is recognised by a zero automaton. Therefore, in order to decide satisfiability of TMSO+zero, it suffices to decide emptiness for zero automata. Unlike the logic, zero automata involve no nesting, which makes the emptiness check easier. A planned followup paper will show that emptiness is indeed decidable for zero automata, assuming that zero is the sets of probability zero.

Apart from the emptiness question for zero automata, the main open problem is decidability for the full logic MSO+zero, and not just the thin variant considered in this paper. It is not at all clear if zero automata are closed under complement, and therefore it is quite possible that zero automata are not the right model for MSO+zero. There is another logic, which sits between TMSO+zero and MSO+zero, and which might still admit a translation to zero automata. In this intermediate logic, the condition on sets $X \subseteq 2^*$ is relaxed: instead of thin sets, we consider sets which satisfy $\bar{X} \in$ zero. We leave open the question whether this intermediate logic admits a translation to zero automata.

Acknowledgment. I would like to thank Henryk Michalewski and Matteo Mio for introducing me into this area, and for their many valuable comments and suggestions.

References

1 Christel Baier, Marcus Größer, and Nathalie Bertrand. Probabilistic ω-automata. *J. ACM*, 59(1):1, 2012. doi:10.1145/2108242.2108243.

2 Vince Bárány, Łukasz Kaiser, and Alexander Rabinovich. Cardinality quantifiers in MLO over trees. In *Proc. of CSL*, 2009.

3 Mikolaj Bojanczyk. U. *ACM SIGLOG News*, 2(4):2–15, 2015.

4 Tomás; Brázdil, Vojtech Forejt, Jan Kretínský, and Antonín Kucera. The satisfiability problem for probabilistic CTL. In *Proc. of LICS*, pages 391–402, 2008.

5 Julius R. Büchi. On a decision method in restricted second-order arithmetic. In *Proc. 1960 Int. Congr. for Logic, Methodology and Philosophy of Science*, pages 1–11, 1962.

6 Arnaud Carayol, Axel Haddad, and Olivier Serre. Randomization in automata on infinite trees. *ACM Trans. Comput. Log.*, 15(3):24:1–24:33, 2014. doi:10.1145/2629336.

7 Thomas Colcombet. Regular cost functions, part I: logic and algebra over words. *Logical Methods in Computer Science*, 9(3), 2013. doi:10.2168/LMCS-9(3:3)2013.

8 Daniel Lehmann and Saharon Shelah. Reasoning with time and chance. *Information and Control*, 53(3):165–1983, 1982.

9 Henryk Michalewski and Matteo Mio. Baire category quantifier in monadic second order logic. In *Automata, Languages, and Programming – 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part II*, pages 362–374, 2015. doi:10.1007/978-3-662-47666-6_29.

10 Henryk Michalewski and Matteo Mio. Measure quantifier in monadic second order logic. In *Logical Foundations of Computer Science – International Symposium, LFCS 2016, Deerfield Beach, FL, USA, January 4-7, 2016. Proceedings*, pages 267–282, 2016. doi:10.1007/978-3-319-27683-0_19.

11 Michael O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of American Mathematical Society*, 141:1–35, 1969.

12 Alexander Rabinovich. On decidability of monadic logic of order over the naturals extended by monadic predicates. *Inf. Comput.*, 205(6):870–889, 2007. doi:10.1016/j.ic.2006.12.004.

13 Sergiu Hart Micha Sharir. Probabilistic propositional temporal logics. *Information and Control*, 70(2–3):97–155, 1986.