A Complete Axiomatization of MSO on Infinite Trees

Das, Anupam; Riba, Colin

DOI:
10.1109/LICS.2015.44

A complete axiomatization of MSO on infinite trees

Anupam Das Colin Riba
Laboratoire LIP, ENS de Lyon, CNRS, Inria, UCBL, Université de Lyon

***Abstract*—We show that an adaptation of Peano's axioms for second-order arithmetic to the language of MSO completely axiomatizes the theory over infinite trees. This continues a line of work begun by Büchi and Siefkes with axiomatizations of MSO over various classes of linear orders.**

Our proof formalizes, in the axiomatic theory, a translation of MSO formulas to alternating parity tree automata. The main ingredient is the formalized proof of positional determinacy for the corresponding parity games which, as usual, allows us to complement automata in order to deal with negation of MSO formulas. The Comprehension Scheme of monadic second-order logic is used to obtain *uniform* winning strategies, whereas most usual proofs of positional determinacy rely on forms of the Axiom of Choice or transfinite induction.

I. Introduction

Rabin's tree theorem [9], that the monadic theory of two successors (S2S) is decidable, is one of the most powerful results known concerning the decidability of logics. In this paper, we present a complete axiomatization of S2S, i.e. a set of sound MSO formulas from which one can infer all of S2S. Our axiomatization can be seen as a subsystem of second-order Peano arithmetic (PA2).

We continue a line of work begun by Büchi and Siefkes, who gave axiomatizations of MSO on various classes of linear orders (see e.g. [11], [2]). These works essentially rely on formalizations of automata in the logic. A major result in the axiomatic treatment of logics over infinite structures is Walukiewicz's proof of completeness of Kozen's axiomatization of the modal μ-calculus [16]. More recent works include complete axiomatizations of MSO and the modal μ-calculus over finite trees [13], [5]. These works (as well as the recent reworking [10] of Siefkes's completeness proof of MSO on ω-words [11]), rely on model-theoretic techniques, which allow elegant reformulations of algebraic approaches. For the case of infinite trees the algebraic approach (in [1]) appears to be much more complicated. We therefore directly formalize a translation of formulas to automata in the axiomatic theory.

There are several ways to translate MSO formulas to automata (see e.g. [6], [14], [17]). We choose to translate formulas to alternating parity automata. The two non-trivial steps in the translation of formulas to alternating automata are negation and existential quantification. Negation requires the *complementation* of automata while, for existential quantifiers, we simulate an alternating automaton by an equivalent nondeterministic one (*nondeterminization*) whence we can obtain an automaton computing the appropriate projection.

Work partially funded by the ANR Blanc project RÉCRÉ.

Recall that alternating (tree) automata generalize nondeterministic automata by admitting positive boolean formulas as transitions. We adopt (a special case of) the presentation of [17], in which transitions are represented using a double powerset (corresponding to irredundant disjunctive normal forms), which allows for a smooth treatment of complementation and nondeterminization.

Working in the language of MSO imposes some constraints on the formalization. This is mainly due to the fact that it is not possible to compare the length of tree positions, since this would allow us to define a Choice formula on tree positions, contradicting [7] (see also [3]). Consequently, when formalizing acceptance of tree automata, we can only deal with *positional* strategies, i.e. those dependent on only the current position of the game, not the entire history of the play.

It is well-known (see e.g. [4], [8], [17]) that nondeterminization of alternating automata is strongly linked to McNaughton's Theorem (the determinization of ω-word automata). At this point, we rely on already known axiomatizations of MSO on ω-words [11], [10]. These provide, for each formula of MSO on trees defining an ω-regular language, a (provably equivalent) formula describing a deterministic ω-word automaton recognizing this language.

Our main difficulty concerns complementation. As usual we rely on positional determinacy of parity games, and we show that our axiomatization of MSO proves that acceptance games of alternating (parity) automata are positionally determined. Our proof formalizes a variant of the positional determinacy argument of [14]. The main difficulty is the obtainment of *uniform* winning strategies: from a set of positions from which, say, player Eloise has a winning strategy, the problem of computing a strategy for Eloise which is winning from all positions of that set. This property is naturally proved using the Axiom of Choice or some form of transfinite induction, which are unavailable in our axiomatization. Nonetheless, for the restricted type of game graphs induced by tree automata, we show that the Comprehension axiom scheme, together with induction on tree positions, allows us to build such a strategy.

It is well-known that SkS, the monadic theory of k-ary trees, can be embedded in S2S [9]. However, it does not seem as direct to use such an embedding to obtain an axiomatization of SkS from an axiomatization of S2S. Therefore, in this work, we axiomatize the MSO theory of the full infinite D-ary tree, for an arbitrary finite ordered set D. We denote by MSO_D the corresponding formal system.

As expected, formalization in (subsystems of) arithmetic involves a lot of coding. But since MSO on infinite trees is a

decidable logic, it does not admit the usual primitive recursive codings (see e.g. [12]), and so we use Boolean codings of finite structures instead. We work in an extension of MSO_D which we call *bounded* FSO_D. FSO_D ('functional second-order logic') is just an extension of MSO_D by function symbols and a Choice axiom scheme. In *bounded* FSO_D functions are required to range over finite sets only, and we show that this system is directly interpretable in MSO_D. This setting allows us to smoothly handle much of the bureaucracy caused by the finite Boolean codings required by the formalization.

We also often use *Henkin models*. Thanks to the corresponding completeness result, they allow us to reason in genuine mathematical structures rather than at the syntactic level of formal proofs. This eases the presentation of some arguments.

A potentially interesting aspect of this work is the possibility of new decision procedures for MSO on infinite trees based on *proof search*. This seems quite remarkable since, besides the algebraic approach of [1], the only known approaches to Rabin's theorem proceed first via translation of formula to automata. We elaborate on this in Sect. VII.

Organization of the paper

We present our axiomatization in Sect. II. In Sect. III we gather the required logical tools, in particular basic facts about MSO_D and its relation to MSO on ω-words, and the system FSO_D and its relation to MSO_D. The formalization of alternating parity tree automata is then presented in Sect. IV, which also contains (assuming positional determinacy of parity games) proofs of the correctness of basic operations on automata. The proof of the completeness of our axiomatization, based on the formalized translation of formulas to automata, is then assembled in Sect. V. Section VI contains our main technical contribution, the proof of positional determinacy. We elaborate on possible further directions in Sect. VII and give concluding remarks in Sect. VIII.

II. An axiomatization of MSO on infinite trees

Throughout this work we fix given countable sets $\mathcal{V}^\iota = \{x, y, z, \dots\}$ and $\mathcal{V}^o = \{X, Y, Z, \dots\}$ of *individual* and *monadic* variables, respectively. We also fix a nonempty set D of the form $[n] := \{0, \dots, n-1\}$ for some $n \in \mathbb{N}$.

A. The system MSO_D

Definition II.1 (Language of MSO_D)**.** *The individual terms of MSO_D, denoted a, b, etc. are built from individual variables and the symbols $\dot\varepsilon$ (nullary, intended to denote the root of the tree) and S_d for each $d \in D$ (all unary, intended to denote immediate successors).*

The formulas, denoted $\phi, \psi \in \Lambda_D$, are built from atomic formulas of the form Xa by means of negation $\neg\phi$, disjunction $\phi \vee \psi$ and existential quantifications $\exists x.\phi$ and $\exists X.\phi$.

We may also write $a \mathrel{\dot\in} X$ rather than Xa for convenience.

The symbols $\wedge, \to, \longleftrightarrow, \forall x$ and $\forall X$ are defined as usual. We often omit brackets to reduce syntax, in particular interpreting sequences of implications by associating brackets to the right, e.g. $\phi \to \psi \to \chi$ is interpreted as $\phi \to (\psi \to \chi)$.

TABLE I
Axioms for MSO_D

Tree axioms: $\forall x \neg(\mathsf{S}_d(x) \doteq \mathsf{S}_{d'}(x))$ (for each $d \neq d' \in D$)
For each $d \in D$:

$$\forall xy(\mathsf{S}_d(x) \doteq \mathsf{S}_d(y) \to x \doteq y) \qquad \forall x \neg(\mathsf{S}_d(x) \doteq \dot\varepsilon)$$

Induction:

$$\forall X \left[X\dot\varepsilon \to \forall y \bigwedge_{d \in D} (Xy \to X\mathsf{S}_d(y)) \to \forall y Xy \right]$$

Comprehension: for all formula ϕ,

$$\exists X \forall y \, [Xy \longleftrightarrow \phi] \qquad (X \notin \mathrm{FV}(\phi))$$

1) Axiomatization: Provability in MSO_D is obtained by deduction in the predicate calculus (rules of Table II, where Γ denotes a finite multiset of formulas), using the axioms of Table I, where the $(x \doteq y)$ is defined as $\forall X(Xx \to Xy)$. We write $\Gamma \vdash_{\mathsf{MSO}_D} \phi$ if $\Gamma \vdash \phi$ is derivable in MSO_D and $\mathsf{MSO}_D \vdash \phi$ for $\vdash_{\mathsf{MSO}_D} \phi$.

2) Interpretation: MSO_D is interpreted in the standard model $\mathfrak{T}_D$ of the D-ary tree D^* as expected: individual variables $x \in \mathcal{V}^\iota$ range over tree positions D^*, $\dot\varepsilon$ is interpreted by the root $\varepsilon \in D^*$, successors S_d map $a \in D^*$ to $a \cdot d \in D^+$ and monadic variables $X \in \mathcal{V}^o$ range over $\mathcal{P}(D^*)$.

Our axiomatization is sound, in the following sense:

Fact II.2. *If $\mathsf{MSO}_D \vdash \phi$ then $\mathfrak{T}_D \models \phi$.*

B. Main result

The main result of this paper is that the axiomatization MSO_D is complete with respect to MSO on the infinite D-ary tree. This is an immediate consequence, via Fact II.2 above, of the following result, which is our main contribution:

Theorem II.3 (Completeness of MSO_D)**.** *For every closed formula ϕ of MSO_D, either $\mathsf{MSO}_D \vdash \phi$ or $\mathsf{MSO}_D \vdash \neg\phi$.*

Corollary II.4. *For every closed formula ϕ of MSO_D, $\mathsf{MSO}_D \vdash \phi$ if and only if $\mathfrak{T}_D \models \phi$.*

The argument for Thm. II.3 can be outlined as follows: we prove in MSO_D that every formula is equivalent to a formula representing a tree automaton. For closed formulas the corresponding automata work on the singleton alphabet, and in this case we show that, for any automaton $\mathcal{A}$, MSO_D proves that the unique tree on this alphabet is accepted either by $\mathcal{A}$ or by its 'complement' automaton.

III. A logical toolbox

We introduce the various tools and concepts used to formalize the completeness argument and proofs in later sections.

A. Henkin completeness

We recall the notion of Henkin models and their corresponding completeness results. These allow us to reason in genuine mathematical structures rather than at the syntactic level of formal proofs, thus easing the presentation of some arguments.

TABLE II
DEDUCTION RULES

Rules for Propositional Logic

$$\frac{}{\Gamma \vdash \phi \vee \neg\phi} \qquad \frac{}{\Gamma, \phi \vdash \phi} \qquad \frac{\Gamma \vdash \phi \qquad \Gamma \vdash \neg\phi}{\Gamma \vdash \psi}$$

$$\frac{\Gamma \vdash \phi}{\Gamma \vdash \phi \vee \psi} \qquad \frac{\Gamma \vdash \psi}{\Gamma \vdash \phi \vee \psi} \qquad \frac{\Gamma \vdash \phi \vee \psi \qquad \Gamma, \phi \vdash \varphi \qquad \Gamma, \psi \vdash \varphi}{\Gamma \vdash \varphi}$$

Rules for First-Order Logic
(where either $\mathcal{X} \in \mathcal{V}^\iota$ and $\mathcal{Y}$ is a term, or $\mathcal{X}, \mathcal{Y} \in \mathcal{V}^o$)

$$\frac{\Gamma \vdash \phi[\mathcal{Y}/\mathcal{X}]}{\Gamma \vdash \exists \mathcal{X} \phi}$$

$$\frac{\Gamma \vdash \exists \mathcal{X} \phi \qquad \Gamma, \phi \vdash \psi}{\Gamma \vdash \psi} \ (\mathcal{X} \notin \mathrm{FV}(\Gamma, \psi))$$

A *Henkin structure* $\mathfrak{M}$ for monadic second-order order logic is given by a set $\mathfrak{M}^\iota$ of individuals and a set $\mathfrak{M}^o \subseteq \mathcal{P}(\mathfrak{M}^\iota)$ of monadic predicates (over which individual variables x, y, z etc. and monadic variables X, Y, Z etc. range, respectively).

A *Henkin structure* for MSO_D is a Henkin structure $\mathfrak{M}$ equipped with a constant $\varepsilon^{\mathfrak{M}} \in \mathfrak{M}^\iota$ and functions $\mathsf{S}_d^{\mathfrak{M}} : \mathfrak{M}^\iota \to \mathfrak{M}^\iota$ for each $d \in D$. Such a structure is a *Henkin model* of MSO_D if it satisfies all the axioms of MSO_D, and if moreover $\mathfrak{M} \models \mathfrak{a} \doteq \mathfrak{b}$ iff $\mathfrak{a} = \mathfrak{b}$ for all $\mathfrak{a}, \mathfrak{b} \in \mathfrak{M}^\iota$.

As usual, we have the following property (see e.g. [15]):

Theorem III.1 (Henkin Completeness)**.** *Given a closed formula ϕ of MSO_D, we have that $\mathsf{MSO}_D \vdash \phi$ iff $\mathfrak{M} \models \phi$ for all Henkin models $\mathfrak{M}$ of MSO_D.*

B. Some basic facts on MSO_D

First, MSO_D proves the usual axioms $\forall x(x \doteq x)$ and $\forall X \forall xy\,[x \doteq y \to Xx \to Xy]$ on the defined equality $\doteq$.

The prefix order is definable as,

$$(x \dot{\leq} y) \quad := \quad \forall X\,[\mathrm{Up}(X) \to Xx \to Xy]$$

where $\mathrm{Up}(X) := \bigwedge_{d \in D} \forall x\,[Xx \to X(\mathsf{S}_d(x))]$. The strict prefix order is $(x \dot{<} y) := (x \dot{\leq} y) \wedge \neg(x \doteq y)$.

We give some common important facts below.

Proposition III.2. *The following are provable in MSO_D:*

(i) $\dot{\leq}$ is reflexive, transitive, antisymmetric, and form a tree domain with root $\dot{\varepsilon}$:

$$\forall x(\dot{\varepsilon} \dot{\leq} x) \quad \forall xyz[y \dot{\leq} x \to z \dot{\leq} x \to (y \dot{\leq} z \vee z \dot{\leq} y)]$$

(ii) $\dot{<}$ is antireflexive, symmetric and unbounded.

(iii) $\dot{<}$-induction:

$$\forall X\,[\forall x\,(\forall y(y \dot{<} x \to Xy) \to Xx) \to \forall x Xx]$$

(iv) Every non $\dot{<}$-minimal element has a predecessor:

$$\forall x\,[\exists y(y \dot{<} x) \to \exists y\,(y \dot{<} x \wedge \neg\exists z(y \dot{<} z \dot{<} x))]$$

We now give a particularly useful consequence of $\dot{<}$-induction:

Proposition III.3 (Recursion Theorem)**.** *Let $\phi(X, x)$ be a formula such that MSO_D proves:*

$$\forall x \forall XY[\forall y \dot{<} x(Xy \longleftrightarrow Yy) \to (\phi(X, x) \longleftrightarrow \phi(Y, x))]$$

Then MSO_D also proves $\exists! X \forall x[Xx \longleftrightarrow \phi(X, x)]$.

We define $\mathrm{Path}^\infty(X)$, that X is an infinite path, as follows:

$$\begin{array}{l} \mathrm{Ub}(X) \wedge \mathrm{Lin}(X) \wedge \\ \forall xyz\,\big[Xx \to Xy \to x \dot{\leq} z \dot{\leq} y \to Xz\big] \end{array} \tag{1}$$

where $\mathrm{Ub}(X)$ is $\forall x\,[Xx \to \exists y(Xy \wedge x \dot{<} y)]$ and $\mathrm{Lin}(X)$ is $\forall xy\,\big[Xx \to Xy \to (x \dot{\leq} y \vee y \dot{\leq} x)\big]$.

One of the most important properties of infinite paths in MSO_D is Prop. III.5 below.

C. MSO on ω-words

For $D = [1] = \{0\}$, the standard model $\mathfrak{T}_D$ is the standard model $(\mathbb{N}, \mathcal{P}(\mathbb{N}), (n \mapsto n+1))$ of ω-words, and MSO_D completely axiomatizes its MSO theory [11]. We shall use a stronger fact, namely that for any finite D, relativization to any (provably) infinite path of MSO_D gives the MSO theory of ω-words.

Definition III.4 (Relativization)**.** *Given two formulas ϕ and $\psi(y)$ with disjoint free variables, the relativization of ϕ to $\psi(y)$, denoted $\phi^{\psi(y)}$ (or simply ϕ^ψ when y is clear from the context), is obtained by recursively replacing in ϕ every first-order quantification $\exists x \varphi$ by $\exists x(\psi(x) \wedge \varphi^{\psi(y)})$.*

Proposition III.5. *If ϕ is a closed MSO formula ϕ built on the purely relational language consisting of only $\dot{<}$, then $\mathrm{Path}^\infty(X) \vdash_{\mathsf{MSO}_D} \phi^{X(-)}$ if and only if $(\mathbb{N}, \mathcal{P}(\mathbb{N}), <) \models \phi$.*

MSO formulas built as in Prop. III.5 are the same as those considered in the axiomatization of MSO on ω-words of [10].

D. Functional second-order logic

In this section we present an extension of MSO_D with unary function symbols and a Choice axiom scheme, denoted FSO_D. It is in this system that we will conduct the proofs in later sections. In particular we show that, when the codomains of these functions are provably finite we have only a conservative extension of MSO_D.

Definition III.6 (The system FSO_D)**.** *The language of FSO_D is that of MSO_D together with countably many* unary *function symbols f, g etc. We allow quantification over these symbols, i.e. if ϕ is a formula then $\exists f \phi$ is a formula, and extend the notion of 'free variable' appropriately.* Terms *of FSO_D are permitted to include function symbols. A term is* closed *if it contains no free variables.*

The rules of FSO_D *extend* MSO_D *as follows:*

- *In the '$\exists$' rules in Table II, the symbols $\mathcal{X}, \mathcal{Y}$ are further allowed to be function symbols.*
- *We add the* Choice *axiom scheme,*

$$\forall x \exists y \phi \ \rightarrow \ \exists f \forall x \, \phi[f(x)/y] \qquad \textit{(for } f \notin \mathrm{FV}(\phi)\textit{)} \quad (2)$$

Henkin completeness (Thm. III.1) extends to FSO_D with respect to the following notion of model: A *Henkin structure* for FSO_D is a Henkin structure $\mathfrak{M}$ for MSO_D equipped with an additional set of functions $\mathfrak{M}^{\iota \to \iota} \subseteq \mathfrak{M}^{\iota \mathfrak{M}^{\iota}}$, over which the function variables f, g etc. range. Such a structure is a *model* of FSO_D if it is a model of MSO_D and satisfies the additional axioms of FSO_D.

Our main use for Choice is to define new functions in FSO_D. In these cases, it is often implicit that we might be using the following consequence of it, a version of comprehension for functions.

Definition III.7 (Functional Comprehension)**.** *We define the* Functional Comprehension *schema, where $f \notin \mathrm{FV}(\phi)$, as*

$$\forall x \exists! y \phi \ \rightarrow \ \exists f \forall x y \, (f(x) \doteq y \longleftrightarrow \phi) \quad (3)$$

We show that Functional Comprehension is valid in FSO_D:

Proposition III.8. $\mathsf{FSO}_D \vdash (3)$.

Proof. We reason inside an arbitrary Henkin model of FSO_D. By the antecedent of (3), we have that $\forall x. \exists y. \phi$, whence we can apply Choice to obtain,

$$\exists f. \forall x. \phi[f(x)/y] \quad (4)$$

which implies the left-right direction of the succedent of (3).

Now let $\mathfrak{f}$ be the function witnessing (4) above. We have that $\phi[\mathfrak{f}(x)/y]$, but again by the antecedent of (3) we have that any witness for $\forall x. \exists y. \phi$ is unique, and so equal to $\mathfrak{f}(x)$, giving the right-left direction of the equivalence required. □

In the definition of boundedness below, it may seem that we use an odd bounding relation (the newly defined '$\trianglelefteq$'). In fact it does not matter which relation we use, as long as it is well-founded, but this convention allows us to present the interpretation of FSO_D in MSO_D, Def. III.11, without the need for cumbersome codings. [1]

Definition III.9 (Boundedness)**.** *For words $\sigma, \tau \in \mathbf{2}^*$, let us write $\sigma \trianglelefteq \tau$ if $|\sigma| = |\tau|$ and σ is lexicographically less than or equal to τ.*

In FSO_D *we define the following:*

$$x \dot{\trianglelefteq} \overline{\sigma} \quad := \quad \bigvee_{\tau \trianglelefteq \sigma} x = \overline{\tau}$$

where $\sigma, \tau \in \mathbf{2}^*$ and $\overline{\sigma}$, $\overline{\tau}$ are the corresponding terms built using $\dot{\varepsilon}$, S_0 and S_1.

We define 'bounded' functional quantifiers as follows, where b is a closed term built using $\dot{\varepsilon}$, S_0 and S_1:

$$\begin{array}{lll} \exists f \dot{\trianglelefteq} b. \phi & : & \exists f. (\forall x. f(x) \dot{\trianglelefteq} b) \wedge \phi \\ \forall f \dot{\trianglelefteq} b. \phi & : & \forall f. (\forall x. f(x) \dot{\trianglelefteq} b) \rightarrow \phi \end{array}$$

[1] These might be required if the relation compares strings of different length.

A formula is bounded *if it is logically equivalent to one where each function quantifier is bounded by a* closed *term.*

An FSO_D*-proof is* bounded *if every formula occurring in it is bounded.*

Notice, in particular, that an instance of Functional Comprehension is bounded if its Comprehension formula (ϕ in (3) above) is bounded.

Our aim in this section is to obtain the following result:

Theorem III.10. *Bounded* FSO_D *is interpretable in* MSO_D.

It might serve as helpful intuition to consider 0 and 1 as defined truth constants standing for falsum and truth,[2] respectively. We can then associate vectors of formulas with bit strings specifying their respective truth values, as below.

Definition III.11 (Interpretation)**.** *For a tuple $\vec{X} = (X_1, \ldots, X_k)$ and a bit string $\sigma \in \mathbf{2}^k$ we define:*

$$\vec{X} x \longleftrightarrow \sigma \quad := \quad \bigwedge_{i=1}^{k} X_i x \longleftrightarrow \sigma_i$$

The interpretation $\langle \cdot \rangle$ of FSO_D*-formulas commutes with the connectives $\neg, \vee$ and $\exists$ over individual and set variables. We define $\langle \exists f \dot{\trianglelefteq} \sigma. \phi \rangle$ as,*

$$\exists X_1, \ldots, X_{|\sigma|}. \langle \phi \rangle_{f,\sigma}$$

where $\langle \phi \rangle_{f,\sigma}$ is obtained from $\langle \phi \rangle$ by replacing all atomic subformulas of the form $X f(a)$ by:

$$\bigvee_{\tau \trianglelefteq \sigma} X \overline{\tau} \wedge (\vec{X} a \longleftrightarrow \tau)$$

In order to show that this interpretation is correct, i.e. preserves validity, it will be helpful for us to establish a partial converse to Prop. III.8. Let us denote by FSO'_D the system obtained from FSO_D by omitting the Choice schema (2).

Proposition III.12. *If ϕ is bounded, then* $\mathsf{FSO}'_D \vdash (3) \rightarrow (2)$.

Proof. Working in an arbitrary model of FSO'_D. Notice that any bounded instance of Choice has antecedent of the form $\forall x. \exists y \dot{\trianglelefteq} \overline{\sigma}. \phi(x, y)$. In particular we can find the least $\tau \trianglelefteq \sigma$ such that $\phi(x, \overline{\tau})$ by propositional logic, i.e. we have:

$$\forall x. \exists! y. [\phi(x, y) \wedge \forall z. (\phi(x, z) \rightarrow y \dot{\trianglelefteq} z)]$$

Now we can apply (bounded) (3) to obtain,

$$\exists f \dot{\trianglelefteq} \overline{\sigma}. \forall x, y. f(x) = y \ \longleftrightarrow (\phi(x, y) \wedge (\forall z \phi(x, z). y \leq z))$$

whence we can deduce by first-order logic,

$$\exists f \dot{\trianglelefteq} \overline{\sigma}. \forall x. \phi(x, f(x))$$

which is equivalent to the succedent required. □

Theorem III.13. *The interpretation $\langle \cdot \rangle$ is correct, i.e. if bounded-*$\mathsf{FSO}_D \vdash \phi$ *then* $\mathsf{MSO}_D \vdash \langle \phi \rangle$.

[2] For example by defining 0 as $(\dot{\varepsilon} \doteq \mathsf{S}_0(\dot{\varepsilon}))$ and 1 as $\neg 0$.

Proof. We extend $\langle\cdot\rangle$ locally to FSO_D-proofs, replacing each formula with its interpretation under $\langle\cdot\rangle$, showing that each proof rule of FSO_D can be simulated in MSO_D under $\langle\cdot\rangle$.

It suffices to consider only the function rules. Instances of the $\exists$ rule are simply translated to many instances of $\exists$ on set variables, and so it remains to deal with Choice.

By Prop. III.12 and boundedness we will deal with bounded Functional Comprehension steps instead, which we can assume have the following form:

$$\begin{array}{rl} & \forall x.\exists! y \mathrel{\dot\trianglelefteq} \overline{\sigma}.\phi(x,y) \\ \to & \exists f \mathrel{\dot\trianglelefteq} \overline{\sigma}.\forall xy \;.\; f(x) \doteq y \longleftrightarrow \phi(x,y) \end{array} \tag{5}$$

Now notice that, by the translation above and previously introduced notation, up to equivalence under first-order logic the succedent of (5) is interpreted as:

$$\exists \vec{X}.\forall x. \bigvee_{\tau \trianglelefteq \sigma} \langle \phi(x,\overline{\tau})\rangle \wedge (\vec{X}x \longleftrightarrow \tau) \tag{6}$$

Working in an arbitrary Henkin model of MSO_D, we deduce (6) assuming the interpretation of the antecedent of (5):

$$\forall x.\exists! y. \bigvee_{\tau \trianglelefteq \sigma} y \doteq \overline{\tau} \wedge \langle \phi(x,y)\rangle \tag{7}$$

First, by Comprehension, define sets $\mathfrak{X}_i$ such that $x \in \mathfrak{X}_i$ iff

$$\bigvee_{\substack{\tau \trianglelefteq \sigma \\ \tau_i = 1}} \left(\bigwedge_{j \neq i} \mathfrak{X}_j x \longleftrightarrow \tau_j \right) \wedge \langle \phi(x,\overline{\tau})\rangle$$

for $i = 1, \ldots, |\sigma|$. Now, if any of the $\mathfrak{X}_i x$ are true then, by construction we have $\langle \phi(x,\overline{\tau})\rangle$ for some unique $\tau \mathrel{\dot\trianglelefteq} \sigma$ with $\tau_i = 1$ and $\vec{\mathfrak{X}}x \longleftrightarrow \tau$.

If no $\mathfrak{X}_i x$ is true then we have that $\vec{\mathfrak{X}}x \longleftrightarrow \overline{0^{|\sigma|}}$. However we also have that $\langle \phi(x,\overline{\tau}\rangle$ is not true for any τ containing a 1, and so we must have $\langle \phi(x,\overline{0^{|\sigma|}})\rangle$ by (7). □

Corollary III.14. *Bounded* FSO_D *is a conservative extension of* MSO_D.

From now on we say 'Comprehension' to refer to either Comprehension or Functional Comprehension.

E. Some remarks on codings of finite objects

Throughout this work we refer to a binary coding $\ulcorner\cdot\urcorner$ of finite objects. For technical reasons, this coding will be context dependent, and we elaborate further on this now.

We assume that all finite objects we use are enumerated and that $\mathbf{2}^k$ contains codes of the first 2^k objects. Notice that this means there are infinitely many codes for the same object: one of each sufficiently large length. In a given formula, $\ulcorner\cdot\urcorner$ should map each object to its code of smallest length that also admits a code for every other object occurring in the formula.[3]

For an n-ary relation $\mathcal{R}$ and objects $r_1, \ldots, r_n$ let $\ddot{\mathcal{R}}\ (\overline{\ulcorner r_1 \urcorner}, \ldots, \overline{\ulcorner r_n \urcorner})$ be the truth value of $\mathcal{R}(r_1, \ldots, r_n)$. Similarly, given closed terms $a_1, \ldots, a_n$, let $\ddot{\mathcal{R}}\ (a_1, \ldots, a_n)$ denote the following Boolean formula computing $\mathcal{R}$:

$$\bigvee_{\overline{r} \in \mathcal{R}} \bigwedge_{i=1}^{n} a_i \doteq \overline{\ulcorner r_i \urcorner}$$

We similarly use this 'double dot' notation for infix and other relation symbols when speaking about the codes of objects.

Since the syntax $\overline{\ulcorner\cdot\urcorner}$ is quite heavy, we will systematically admit a certain abuse of this notation: we associate every finite object we use with the closed term computing its code, e.g. we associate $[n] = \{0, \ldots, n-1\}$ with $\overline{\ulcorner [n] \urcorner}$.

In light also of the aforementioned comments on bounding, we give some examples of the expressions we later use.

For a finite set K, with k ranging over its elements, we may write the following:

$$\begin{array}{rcl} x \ddot{=} k & := & x \doteq \overline{\ulcorner k \urcorner} \\ \forall x \ddot{\in} K \;.\; \phi & := & \forall x. \bigwedge_{k \in K} x \doteq \overline{\ulcorner k \urcorner} \to \phi[\overline{\ulcorner k \urcorner}/x] \\ \exists X \ddot{\subseteq} K.\phi & := & \exists X. \bigvee_{L \subseteq K} \forall x.(x \mathrel{\dot\in} X \longleftrightarrow x \ddot{\in} L) \wedge \phi \end{array}$$

We may also write,

$$f \text{ to } K \quad := \quad \forall x \;.\; \bigvee_{k \in K} f(x) \ddot{=} \overline{\ulcorner k \urcorner}$$

and so, for example, the following expression,

$$\exists f \text{ to } L^K \;.\; \forall y, z \ddot{\in} K \;.\; f(x)(y) \ddot{=} f(x)(z) \to y \ddot{=} z$$

is read, using our abuse of notation, as,

$$\exists f. \bigvee_{F \in L^K} f(x) \ddot{=} F \wedge \bigwedge_{k,k' \in K} F(k) \ddot{=} F(k') \to k \ddot{=} k'$$

stating that there is a function f that, on input x, outputs the code of an injection from K to L.[4]

All of these expressions are considered bounded, due to equivalence under first-order logic.

Notice also that the use of the double dot notation should prevent ambiguity in such circumstances, although we retain the code notation in the case that it eases parsing of a formula.

IV. Games, automata and closure under logical operations

The focus of this section is to define automata accepting infinite tree languages, define their acceptance using abstract games, and use this setting to show that the ω-regular tree languages are closed under basic logical operations.

Throughout this and later sections, we freely switch between meta-mathematical reasoning and working inside a model, taking advantage of the results in the previous section.

[3] This is due to the bounding relation $\trianglelefteq$ that we use in the definition of boundedness, which only compares strings of the same length.

[4] It also states that, on any input, the output is a function from K to L.

A. Abstract games

In this section we introduce concepts and notation to deal with abstract games over a product of the infinite tree D^*.

Recall that we may abuse notation by using meta-notation for a finite set in place of the term computing its code.

Let us fix disjoint finite sets P and O of *proponent* (or "Eloise") and *opponent* (or "Abelard") *labels*, respectively. In fact these will be parameters at the meta-level, and this becomes relevant in Sect. IV-D4.

Definition IV.1 (Game positions)**.** *A* game position *is a pair* (x, y) *such that,* $y \ddot{\in} \mathsf{P} \sqcup \mathsf{O}$*, and a set of game positions is a pair* (X, f) *such that* $\forall x \dot{\in} X.f(x) \ddot{\subseteq} \mathsf{P} \sqcup \mathsf{O}$*.*

We reserve variables u, v, w *etc. to vary over game positions and* U, V, W *etc. to vary over sets of game positions, and we use the following notation,*

$$\begin{array}{rcl}(x,y) \doteq (x',y') & := & x \doteq x' \wedge y \ddot{=} y' \\ (x,y) \dot{\in} (X,f) & := & x \dot{\in} X \wedge y \ddot{\in} f(x) \\ (X,f) \dot{\subseteq} (Y,g) & := & X \dot{\subseteq} Y \wedge \forall x \dot{\in} X.f(x) \ddot{\subseteq} g(x) \\ \exists v.\phi & := & \exists x.\exists y \dot{\in} \mathsf{P} \sqcup \mathsf{O} \;.\; \phi[(x,y)/v] \\ \exists V.\phi & := & \exists X.\exists f \text{ to } \mathcal{P}(\mathsf{P} \sqcup \mathsf{O}) \;.\; \phi[(X,f)/V]\end{array}$$

where, in the $\exists$ *cases, we choose* x, X, f *not free in* ϕ*.*

We define f_{P} *and* f_{O} *by Comprehension, denoting the* P *and* O *parts of the range of* f *respectively, i.e.:*

$$\begin{array}{rcl} f_{\mathsf{P}}(x) & \doteq & f(x) \ddot{\cap} \mathsf{P} \\ f_{\mathsf{O}}(x) & \doteq & f(x) \ddot{\cap} \mathsf{O}\end{array}$$

We use other standard set-theoretic and quantifier notations for game positions, defined accordingly from the above.

Recall that D is the finite set of tree directions. Below we define games as a labeling of the nodes in D^*. Each node is thus associated with a finite set of proponent moves, a relation from P to O, and a finite set of opponent moves, a relation from O to $D \times \mathsf{P}$. A *play* is thence obtained by alternating compatible proponent and opponent moves, and traces a path in the tree by the opponent's choices of directions $d \in D$.

Definition IV.2 (Games and plays)**.** *We define* games*, or* edge functions*, as follows:*

$$\mathrm{Game}(e, \mathsf{P}, \mathsf{O}) \;:=\; \forall x \;.\; e(x) \ddot{\subseteq} (\mathsf{P} \times \mathsf{O}) \cup (D \times \mathsf{O} \times \mathsf{P})$$

We define e_{P} *and* e_{O} *by Comprehension, denoting the parts of* e *consisting of just* P*-edges and just* O*-edges respectively. I.e.*

$$\begin{array}{rcl} e_{\mathsf{P}}(x) & \doteq & e(x) \ddot{\cap} \mathcal{P}(\mathsf{P} \times \mathsf{O}) \\ e_{\mathsf{O}}(x) & \doteq & e(x) \ddot{\cap} \mathcal{P}(D \times \mathsf{O} \times \mathsf{P})\end{array}$$

We use the following notation,

$$\begin{array}{rcl}(x,a) \overset{}{\dashrightarrow}_{e_{\mathsf{P}}} (y,b) & := & x \doteq y \wedge (a,b) \ddot{\in} e_{\mathsf{P}}(x) \\ (x,a) \overset{}{\dashrightarrow}_{e_{\mathsf{O}}} (y,b) & := & \bigvee_{d \in D} \mathsf{S}_d(x) \doteq y \wedge (d,a,b) \ddot{\in} e_{\mathsf{O}}(x)\end{array}$$

and: $u \overset{}{\dashrightarrow}_{e} v \;:=\; u \overset{}{\dashrightarrow}_{e_0} v \;\vee\; u \overset{}{\dashrightarrow}_{e_1} v$*.*

We may further write the following:[5]

$$V : u \overset{\omega}{\dashrightarrow}_{e} \infty \;:=\; u \dot{\in} V \wedge \forall v \in V. \exists! w \dot{\in} V \;.\; v \overset{}{\dashrightarrow}_{e} w$$

$$u \overset{*}{\dashrightarrow}_{e} v \;:=\; \exists V : u \overset{\omega}{\dashrightarrow}_{e} \infty \;.\; v \dot{\in} V$$

We define a play V *from* u *of a game* e*, as follows:*

$$\begin{array}{rcll}\mathrm{Play}(V,u,e) & := & & \mathrm{Game}(e, \mathsf{P}, \mathsf{O}) \\ & & \wedge & V : u \overset{\omega}{\dashrightarrow}_{e} \infty \\ & & \wedge & \forall v \dot{\in} V \;.\; u \overset{*}{\dashrightarrow}_{e} v\end{array}$$

Notice that our definition of play is somewhat specific to the tree-like setting that we are in. E.g. For general game graphs the notion of play we have defined does not determine a unique starting position in a play that is cyclic. Notice also that we insist our plays are deadlock-free, i.e. all plays are infinite.

B. Automata, acceptance and winning

We use a notion of an *alternating* automaton that is essentially a special case of that used in [17]. We typically consider *parity* acceptance conditions, or 'Rabin chain' acceptance, but present automata with arbitrary conditions below.

Informally, an alternating automaton is to an automaton what an alternating Turing machine is to a Turing machine: the transition function may code arbitrary boolean conditions on the state and letter read.

In what follows, we fix a finite alphabet Σ.

Definition IV.3 (Alternating tree automata)**.** *An* alternating tree automaton *is a tuple:*

$$\mathcal{A} \;=\; (Q, q^{\imath}, \delta, \Omega) \tag{8}$$

where Q *is a finite set of 'states',* $q^{\imath} \in Q$ *is the 'initial' state,* $\Omega \subseteq Q^{\omega}$ *is the acceptance condition and* δ *is the 'transition function', having the following format:*

$$\delta : Q \times \Sigma \to \mathcal{P}(\mathcal{P}(Q \times D))$$

To define what it means for an automaton to accept a tree, we refer to the notions of game introduced earlier.

Let us fix an automaton $\mathcal{A}$, as specified in (8). In what follows we will specialize the concepts of Sect. IV-A by setting $\mathsf{P} = Q$ and $\mathsf{O} = \mathcal{P}(Q \times D)$.

We use the function variable t as a parameter coding some Σ-labelled tree, i.e. we intend that $\forall x.t(x) \ddot{\in} \Sigma$.

Definition IV.4 (Acceptance games)**.** *We define the* acceptance game $e^{\mathcal{A},t}$ *of* $\mathcal{A}$ *on* t *by Comprehension as follows:*[6]

$$\begin{array}{rcl} e^{\mathcal{A},t}_{\mathsf{P}}(x) & \ddot{=} & \{(q,\gamma) \mid \gamma \ddot{\in} \delta(q, t(x))\} \\ e^{\mathcal{A},t}_{\mathsf{O}}(x) & \ddot{=} & \{(d,\gamma,q) \mid (q,d) \ddot{\in} \gamma\} \\ e^{\mathcal{A},t}(x) & \doteq & e^{\mathcal{A},t}_{\mathsf{P}}(x) \ddot{\sqcup} e^{\mathcal{A},t}_{\mathsf{O}}(x)\end{array}$$

[5]We point out that, while there is an existential set quantifier $\exists V$ in $\mathrm{Play}(V, u, e)$ from the use of $u \overset{*}{\dashrightarrow}_{e} v$, it is not difficult to see that this is not necessary. This is an important consideration if one is concerned with the quantifier complexity of such concepts.

[6]Technically, in this definition, the occurrence of δ (as well as that of c in the definition of Par^c below) should be double-dotted, since it is a relation over the code represented by the term $t(x)$. However we choose to abuse notation once more in favor of keeping the syntax light.

A parity condition is given by a mapping from the finite set P of proponent labels to $\mathbb{N}$. An infinite P-sequence, e.g. induced by a labeled path in D^*, is *winning* under this condition if the least number associated with some label in P occuring infinitely often in the sequence is even.

Recall the definition of an infinite path, $\mathrm{Path}^\infty(\cdot)$, from (1).

Definition IV.5 (Parity)**.** *Let $c : \mathsf{P} \to [0, n]$ and (X, f) be a set of game positions.*[7] *We define $\mathrm{Par}^c(X, f)$ as follows:*

$$\mathrm{Path}^\infty(X) \wedge \forall x.\exists! y.y \mathrel{\ddot{\in}} f(x)$$
$$\wedge \bigvee_{i=0}^{\lfloor \frac{n}{2} \rfloor} \left[\begin{array}{rl} & \forall x \in X.\exists y \in X \;.\; y \geq x \wedge c(f_{\mathsf{P}}(y)) \mathrel{\doteq} 2i \\ \wedge & \exists x \in X.\forall y \in X \;.\; y \geq x \to c(f_{\mathsf{P}}(y)) \mathrel{\ddot{\geq}} 2i \end{array} \right]$$

C. Strategies

Acceptance for an automaton will be defined by the existence of a winning P-strategy under the parity condition above. Games with such conditions are known to be *positionally determined*: there is a winning strategy for P or O that is history-free, i.e. one whose moves are dependent only on the current position and not on the history of the play. This choice of condition is ultimately why we are able to express acceptance in bounded FSO_D (and so MSO_D).

Recall that, in bounded FSO_D, we cannot define a relation comparing the length of tree positions, since this would allow us to define a Choice formula on tree positions, contradicting [7] (see also [3]).

Consequently, when formalizing acceptance of tree automata, we can only deal with *positional* strategies. Indeed, general strategies have to be represented by trees of arbitrary finite branching (possibly different from D). Such trees can be represented in bounded FSO_D, but it seems that (without length comparison) we cannot define a formula relating a play in such a strategy to its underlying position in the input tree.

However, positional strategies in acceptance games for D-ary trees can be described by D-ary trees labeled by functions on finite sets, specify the local moves admitted by the strategy. This allows us to adequately define a bounded FSO_D formula relating a play in a positional strategy to its underlying input tree position.[8][9]

What we call a 'winning strategy' below corresponds, in fact, to this notion of positional winning strategy. We parametrize this notion by some winning condition $\mathcal{W}(V)$.

Definition IV.6 (Winning strategies)**.** *We define the formula,*

$$\mathrm{WinStrat}(s, v, e, \mathcal{W})$$

as follows:

$$s \text{ to } \mathsf{O}^{\mathsf{P}} \;\wedge\; \forall x \;.\; s(x) \mathrel{\ddot{\subseteq}} e_{\mathsf{P}}(x)$$
$$\wedge \quad \forall V \mathrel{\dot{\ni}} v.(\mathrm{Play}(V, v, s \mathrel{\ddot{\cup}} e_{\mathsf{O}}) \to \mathcal{W}(V))$$

[7]Below we actually abuse notation by associating a singleton $\{q\}$ with its only element q. This could be rectified by defining c instead as a function $\{\{q\} : q \in \mathsf{P}\} \to [0, n]$.

[8]The complementation construction used in [14] relies on this.

[9]Another possibility would have been to instead use games with the more general Muller conditions, which are known to be determined with strategies of uniformly bounded finite memory. But formalizing this in bounded FSO_D (and proving determinacy) would have been significantly more complicated.

Notice that s is seen here as inducing a *subgame* of e, i.e. if $\mathrm{Game}(e, \mathsf{P}, \mathsf{O})$ then also $\mathrm{Game}(s \mathrel{\ddot{\cup}} e_{\mathsf{O}}, \mathsf{P}, \mathsf{O})$, one in which there is always a unique choice of move by P.

Definition IV.7. *For an automaton $\mathcal{A}$ with acceptance condition specified by an FSO_D formula $\mathcal{W}(V)$, we define,*

$$t \in \mathcal{L}(\mathcal{A}) \quad := \quad \exists s \text{ to } \mathsf{O}^{\mathsf{P}} \;.\; \mathrm{WinStrat}(s, \iota, e^{\mathcal{A},t}, \mathcal{W})$$

where $\iota \doteq (\varepsilon, q^\imath)$.

D. Logical operations on automata in FSO

In this section, we present formalized constructions for the operations of complementation, finite union and projection. Finite union is, as usual, easy. Complementation relies on positional determinacy of (parity) acceptance games, whose formalized proof is deferred to Sect. VI.

For projection, we use the nondeterminization construction of [17]. It cleanly distinguishes the powerset construction on alternating tree automata from the definition of the resulting acceptance condition, for which we rely on the embedding of MSO on ω-words in MSO_D provided by Prop. III.5.

We work in bounded FSO_D, relying on Th. III.13. We freely use the Recursion Theorem in bounded FSO_D (obtained from the analogous result for MSO_D, Prop. III.3, by Thm. III.13).

1) Singleton alphabet: We sketch a proof of the following:

Lemma IV.8. *If $\mathcal{A}$ is an automaton over the singleton alphabet $\{\varnothing\}$, then in FSO_D we have that t to $\{\varnothing\} \vdash t \in \mathcal{L}(\mathcal{A})$ or t to $\{\varnothing\} \vdash t \notin \mathcal{L}(\mathcal{A})$.*

The main point here is that it is easy to check whether an automaton without an input accepts or not. Any automaton over the singleton alphabet $\mathbf{2}^0 = \{\varnothing\}$ can be rewritten to an automaton without an input, by replacing the input parameter for $\{\varnothing\}^*$ and reducing as appropriate.

From the point of view of the acceptance game, the opponent O does not any more care which direction is chosen, since every successive node will be labeled by $\varnothing$. In this way we can view the game from any tree position as follows:

$$\begin{array}{rcl} \mathsf{P}\text{-moves} & : & \{(q, \gamma) \mid \gamma \in \delta(q, \varnothing)\} \\ \mathsf{O}\text{-moves} & : & \{(\gamma, q) \mid (q, d) \in \gamma \text{ for some } d \in D.\} \end{array}$$

Now, this game has only finitely many states and is necessarily positionally determined. Moreover, there are only finitely many strategies and they can be checked for winningness by loop-detection in an exhaustion of their plays.

This can be expressed by a finite quantifier-free formula over FSO_D and so the existence or otherwise of a winning strategy has a purely propositional proof in FSO_D.

2) Complement: Consider an automaton $\mathcal{A} = (Q, q^\imath, \delta, \mathcal{W})$ and define $\widetilde{\mathcal{A}} = (Q, q^\imath, \widetilde{\delta}, \widetilde{\mathcal{W}})$, where,

$$\begin{array}{rcl} \widetilde{\mathcal{W}}(V) & := & \neg \mathcal{W}(V) \\ \widetilde{\delta}(q, \mathtt{a}) & := & \{\gamma \in \mathcal{P}(Q \times D) \mid \gamma \cap \gamma' \neq \varnothing \text{ for all } \gamma' \in \delta\} \end{array}$$

Lemma IV.9. $\mathsf{FSO}_D \vdash t \in \mathcal{L}(\widetilde{\mathcal{A}}) \longleftrightarrow t \notin \mathcal{L}(\mathcal{A})$.

Proof. This follows almost immediately from positional determinacy, Thm. VI.12. It only remains to show that FSO_D proves that $e^{\widetilde{\mathcal{A}},t} \doteq \widetilde{e}^{\mathcal{A},t}$, which is a simple exercise. □

3) Union: Consider automata $\mathcal{A}_i = (Q_i, q_i^\imath, \delta_i, \mathcal{W}_i)$ for $i \in \{1,2\}$. Define $\mathcal{A}_\cup$ as $(Q_1 \sqcup Q_2 \sqcup \{q^\imath\}, q^\imath, \delta_\vee, \mathcal{W}_\vee)$, where

- $\delta_\vee(q^\imath, \mathtt{a}) := \delta_1(q_1^\imath, \mathtt{a}) \cup \delta_1(q_2^\imath, \mathtt{a})$ for $\mathtt{a} \in \Sigma$.
- $\delta_\vee(q, \mathtt{a}) := \delta_i(q, \mathtt{a})$ for $q \in Q_i$
- $\mathcal{W}_\vee(V) := \mathcal{W}_1(V) \vee \mathcal{W}_2(V)$.

Lemma IV.10. *The following is provable in* FSO_D.

$$t \in \mathcal{L}(\mathcal{A}_\cup) \longleftrightarrow (t \in \mathcal{L}(\mathcal{A}_1) \vee t \in \mathcal{L}(\mathcal{A}_2))$$

4) Projection: We now sketch the proof of the following:

Lemma IV.11. *For a parity automaton $\mathcal{A}$, there is a parity automaton $\mathcal{A}^\exists$ such that,*

$$t \text{ to } \Sigma \times \{0,1\}, u \text{ to } \Sigma, (\pi_\Sigma t = u) \vdash u \in \mathcal{L}(\mathcal{A}^\exists) \longleftrightarrow t \in \mathcal{L}(\mathcal{A})$$

where $(\pi_\Sigma t = u)$ stands for:

$$\forall x\, [(t(x) \doteqdot (u(x), 0)) \vee (t(x) \doteqdot (u(x), 1))]$$

We first translate a parity automaton to a nondeterministic automaton with an alternative acceptance condition, whence projection is simple to define, before converting back to a parity automaton.

a) Nondeterminization of an automaton: Consider an automaton $\mathcal{A} = (Q, q^\imath, \delta, \mathcal{W})$ on alphabet Σ. We will first build an equivalent *nondeterministic* automaton

$$\mathcal{A}^{\mathrm{ND}} \quad = \quad (\mathcal{P}(Q \times Q), \{(q^\imath, q^\imath)\}, \delta^{\mathrm{ND}}, \mathcal{W}^{\mathrm{ND}})$$

as follows. Consider $\boldsymbol{q} \in \mathcal{P}(Q \times Q)$ and $a \in \Sigma$ and let $\boldsymbol{q}\restriction 2 := \{q_1, \ldots, q_n\}$. Following [17], we let $\gamma^{\mathrm{ND}} \in \delta^{\mathrm{ND}}(\boldsymbol{q}, a)$ whenever there are $\gamma_1 \in \delta(q_1, a), \ldots, \gamma_n \in \delta(q_n, a)$ such that,

$$\gamma^{\mathrm{ND}} \quad = \quad \{(\boldsymbol{q}_d, d) \mid d \in D\}$$

where, for each $d \in D$:

$$\boldsymbol{q}_d \quad = \quad \{(q_k, q) \mid 1 \leq k \leq n \text{ and } (q, d) \in \gamma_k\}$$

For the acceptance condition of $\mathcal{A}^{\mathrm{ND}}$, the natural choice (from [17]) is to take the set of sequences $(\boldsymbol{q}_n)_{n\in\mathbb{N}}$ such that for every $(q_n)_{n\in\mathbb{N}} \in Q^\omega$ with $\boldsymbol{q}_{n+1} = (q_n, q_{n+1})$, the sequence $(q_n)_{n\in\mathbb{N}}$ satisfies the parity condition of $\mathcal{A}$. This is definable by an FSO_D formula. In order to fit in our framework, we actually take for $\mathcal{W}^{\mathrm{ND}}$ the collection of plays (X, f) such that (X, f_{P}) satisfies the above requirement.

Note that $\mathcal{A}^{\mathrm{ND}}$ is nondeterministic, but in general not a parity automaton. By formalizing the argument of [17] (using the Recursion Theorem), we get:

Lemma IV.12. *For a parity automaton $\mathcal{A}$:*

$$\mathsf{FSO}_{\mathbf{2}} \vdash t \in \mathcal{L}(\mathcal{A}) \longleftrightarrow t \in \mathcal{L}(\mathcal{A}^{\mathrm{ND}})$$

Note that $\mathcal{A}^{\mathrm{ND}}$ does not a priori generate *positionally* determined games. However, since $\mathcal{A}$ is a parity automaton, a winning P strategy witnessing $t \in \mathcal{L}(\mathcal{A})$ can be assumed to be positional, and so the corresponding winning P strategy for $t \in \mathcal{L}(\mathcal{A}^{\mathrm{ND}})$ will be positional. In the context of Lem. IV.12, the converse direction follows from the fact that by definition of acceptance in our setting, $t \in \mathcal{L}(\mathcal{A}^{\mathrm{ND}})$ implies that there is a winning *positional* P strategy, which is then easily mapped to a winning positional P strategy witnessing $t \in \mathcal{L}(\mathcal{A})$.

b) Conversion to parity automaton: In order to maintain compatibility with the rest of our structural induction (e.g. closure under complements), we need to convert our nondeterministic automaton to one with a parity acceptance condition.

As noted in [17], the acceptance condition of $\mathcal{A}^{\mathrm{ND}}$ is an ω-regular language, which can thus be recognized by a deterministic parity ω-word automaton, say $\mathcal{D} = (Q_\mathcal{D}, q_\mathcal{D}^\imath, \delta_\mathcal{D}, \mathcal{W}_\mathcal{D})$ on the alphabet $\mathcal{P}(Q \times Q)$.

Following [17], we let

$$\mathcal{A}_{\mathrm{nd}} := (\mathcal{P}(Q \times Q) \times Q_\mathcal{D}, (q_{\mathcal{A}^{\mathrm{ND}}}^\imath, q_\mathcal{D}^\imath), \delta_{\mathrm{nd}}, \mathcal{W}_{\mathrm{nd}})$$

where $\delta_{\mathrm{nd}}((\boldsymbol{q}, q), a) := (\delta_{\mathcal{A}^{\mathrm{ND}}}(\boldsymbol{q}, a), \delta_\mathcal{D}(q, \delta_{\mathcal{A}^{\mathrm{ND}}}(\boldsymbol{q}, a)))$, and $\mathcal{W}_{\mathrm{nd}}$ is generated by the parity condition c_{nd} which assigns to state $(\boldsymbol{q}, q)$ the priority of q under the parity condition $\mathcal{W}_\mathcal{D}$.

By showing that $\mathcal{A}^{\mathrm{ND}}$ and $\mathcal{A}^{\mathrm{nd}}$ accept the same languages, along with Lemma IV.12, we obtain Lemma IV.11 by the usual projection operation on nondeterministic automata, whose formalization is straightforward.

Proposition IV.13. *For a parity automaton $\mathcal{A}$:*

$$\mathsf{FSO}_{\mathbf{2}} \vdash t \in \mathcal{L}(\mathcal{A}^{\mathrm{ND}}) \longleftrightarrow t \in \mathcal{L}(\mathcal{A}_{\mathrm{nd}})$$

Proof. By definition of δ_{nd}, one can easily map a strategy,

$$s_{\mathrm{nd}} \text{ to } \mathcal{P}(Q_{\mathcal{A}^{\mathrm{ND}}} \times Q_\mathcal{D} \times D)^{Q_{\mathcal{A}^{\mathrm{ND}}} \times Q_\mathcal{D}}$$

on $e^{\mathcal{A}_{\mathrm{nd}},t}$ to a strategy,

$$s^{\mathrm{ND}} \text{ to } \mathcal{P}(Q_{\mathcal{A}^{\mathrm{ND}}} \times D)^{Q_{\mathcal{A}^{\mathrm{ND}}}}$$

on $e^{\mathcal{A}^{\mathrm{ND}},t}$ and vice-versa. The point is to show that both strategies are equally winning.

Consider a (necessarily infinite) play V_{nd} of s_{nd} and a (necessarily infinite) play V^{ND} of s^{ND}, both from the initial positions of these games.

We need to show that:

$$\mathsf{FSO}_D \vdash \mathcal{W}_{\mathrm{ND}}(X^{\mathrm{ND}}, f^{\mathrm{ND}}) \longleftrightarrow \mathrm{Par}^{c_{\mathrm{nd}}}(X_{\mathrm{nd}}, f_{\mathrm{nd}})$$

Consider the closed formula without parameters obtained by prefixing

$$\mathcal{W}_{\mathrm{ND}}(X^{\mathrm{ND}}, f^{\mathrm{ND}}) \longleftrightarrow \mathrm{Par}^{c_{\mathrm{nd}}}(X_{\mathrm{nd}}, f_{\mathrm{nd}})$$

with suitably relativized universal quantifiers, and rewrite this formula to an FSO_D-equivalent formula ψ on the purely relational vocabulary consisting only of the symbol $\dot{<}$.

By construction of the formula Play, in MSO_D we have,

$$\psi \longleftrightarrow \forall X[\mathrm{Path}^\infty(X) \to \psi^X]$$

since by Def. IV.2 and Def. IV.1, we have that $V_{\mathrm{nd}} = (X_{\mathrm{nd}}, f_{\mathrm{nd}})$ and $V^{\mathrm{ND}} = (X^{\mathrm{ND}}, f^{\mathrm{ND}})$ where, by Def. IV.4, X_{nd} and X^{ND} are linearly ordered sets with least element $\dot{\varepsilon}$.

Then, by construction of $\mathcal{D}$, ψ holds in the standard model of infinite words and we conclude by Prop. III.5. □

V. Completeness argument

This short section gathers the preceding results to prove our main result, Thm. II.3. Recall that we proceed by formalizing a translation of formulas to automata. We begin by putting MSO_D formulas into a convenient form for this translation, namely with a purely relational vocabulary and only monadic variables. We then give a proof of Thm. II.3.

A. Reduced syntaxes of MSO_D

For the translation of formulas to automata, it is useful and customary to work with formulas on a slightly different syntax.

1) Relational syntax: We first restrict to a purely relational vocabulary. It is based on the defined formulas $\mathrm{S}_d(x,y) := (\mathsf{S}_d(x) \doteq y)$ for each $d \in D$. The *relational formulas* $\phi, \psi \in \Lambda_D^R$ are built from atomic formulas Xy and $\mathrm{S}_d(x,y)$ by means of $\neg$, $\vee$, $\exists x$ and $\exists X$.

To each formula $\phi \in \Lambda_D$ we associate a formula ϕ^R as follows. For a a term of MSO_D, define the formula $(z \triangleq a)$ by structural induction on a:

$$(z \triangleq y) \;:=\; (z \doteq y) \qquad (z \triangleq \dot{\varepsilon}) \;:=\; \neg\exists z' \bigvee\nolimits_{d\in D} \mathrm{S}_d(z', z)$$
$$(z \triangleq \mathsf{S}_d(a)) \;:=\; \exists z'[z' \triangleq a \wedge \mathrm{S}_d(z', z)]$$

Then, ϕ^R is obtained from ϕ by replacing each atomic Xa, where a is not a variable, by $\exists z[(z \triangleq a) \wedge Xz]$, where z is a fresh variable. Note that $\mathsf{MSO}_D \vdash (z \triangleq a) \longleftrightarrow (z \doteq a)$.

Lemma V.1. *For every formula $\phi \in \Lambda_D$, we have $\mathsf{MSO}_D \vdash \big(\phi \longleftrightarrow \phi^R\big)$.*

2) Individual-free syntax: The next step is to get rid of individual quantifiers. Consider the defined formulas:

$$\begin{array}{lcl}(X \dot{\subseteq} Y) & := & \forall x(Xx \to Yx)\\ \mathrm{S}_d(X,Y) & := & \exists xy\,[Xx \wedge Yy \wedge \mathrm{S}_d(x,y)]\end{array}$$

The *individual-free formulas* $\phi, \psi \in \Lambda_D^0$ are built from atomic formulas $(X \dot{\subseteq} Y)$ and $\mathrm{S}_d(X,Y)$ by means of negation, disjunction and monadic second-order quantification $\exists X$ only.

Let $\phi \in \Lambda_D^R$ with $\mathrm{FV}(\phi) = \{x_1, \ldots, x_p, Y_1, \ldots, Y_q\}$. We inductively associate to ϕ a formula $\phi^0 \in \Lambda_D^0$ with free variables $\{X_1, \ldots, X_p, Y_1, \ldots, Y_q\}$ as follows: we define,

$$(\exists x_{p+1}\phi)^0 \;:=\; \exists X_{p+1}\left[\mathrm{Sing}(X_{p+1}) \wedge \phi^0\right]$$

where $\mathrm{Sing}(X)$ is defined as,

$$\neg(X \doteq \varnothing) \wedge \forall Y\left[Y \dot{\subseteq} X \to \big(Y \doteq \varnothing \vee X \dot{\subseteq} Y\big)\right]$$

with $(X \doteq \varnothing) := \forall Y(X \dot{\subseteq} Y)$. The other inductive cases are defined as follows:

$$\begin{array}{rclcrcl}(Y_j x_i)^0 & := & X_i \dot{\subseteq} Y_j & \quad & (\mathrm{S}_d(x_i,x_j))^0 & := & \mathrm{S}_d(X_i,X_j)\\ (\neg\phi)^0 & := & \neg\phi^0 & & (\phi\vee\psi)^0 & := & \phi^0 \vee \psi^0\\ (\exists Y_{q+1}\phi)^0 & := & \exists Y_{q+1}\,\phi^0 & & & & \end{array}$$

Lemma V.2. *For every formula $\phi \in \Lambda_D^R$ with $\mathrm{FV}(\phi) = \{\vec{x}, \vec{Y}\}$, we have $\overrightarrow{Xx}, \mathrm{Sing}(\vec{X}) \vdash_{\mathsf{MSO}_D} \big(\phi \longleftrightarrow \phi^0\big)$.*

Putting everything together we have:

Corollary V.3. *For every closed formula $\phi \in \Lambda_D$, there is a closed formula $\psi \in \Lambda_D^0$ such that $\mathsf{MSO}_D \vdash (\phi \longleftrightarrow \psi)$.*

B. From formulas to automata

We now give the interpretation of formulas to automata. To each formula $\phi \in \Lambda_D^0$ and sequence of distinct monadic variables $X_1, \ldots, X_n$ such that $\mathrm{FV}(\phi) \subseteq \{X_1, \ldots, X_n\}$ we associate an automaton by induction on ϕ as usual.

Let $\mathcal{L}(\mathcal{A})[\vec{X}]$ denote the interpretation of $t \in \mathcal{L}(\mathcal{A})$ in Def. IV.7 under Def. III.11 for a function t to $\mathbf{2}^{|\vec{X}|}$.

For the atomic formulas of Λ_D^0, we use the following facts:

Lemma V.4. *Given $X_1, \ldots, X_n$ and $i, j \in \{1, \ldots, n\}$, there is an automaton $\mathcal{A}$ on the alphabet $\mathbf{2}^n$ such that,*

$$\mathsf{MSO}_D \vdash \forall X_1, \ldots, X_n \left((X_i \dot{\subseteq} X_j) \longleftrightarrow \mathcal{L}(\mathcal{A})[X_1, \ldots, X_n]\right)$$

Lemma V.5. *Given $X_1, \ldots, X_n$, $i, j \in \{1, \ldots, n\}$ and $d_0 \in D$, there is an automaton $\mathcal{A}$ on the alphabet $\mathbf{2}^n$ such that*

$$\mathsf{MSO}_D \vdash \forall X_1, \ldots, X_n \left(\mathrm{S}_{d_0}(X_i, X_j) \longleftrightarrow \mathcal{L}(\mathcal{A})[X_1, \ldots, X_n]\right)$$

Proposition V.6. *Consider a formula $\phi \in \Lambda_D^0$ and a sequence of distinct monadic variables $X_1, \ldots, X_n$ containing the free variables of ϕ.*

There is an automaton $\mathcal{A}$ on the alphabet $\mathbf{2}^n$ such that

$$\mathsf{MSO}_D \vdash \forall X_1, \ldots, X_n \left(\phi \longleftrightarrow \mathcal{L}(\mathcal{A})[X_1, \ldots, X_n]\right)$$

Proof. For the atomic formulas we use Lemmas V.4 and V.5 respectively. For the logical connectives $\neg$, $\vee$ and $\exists X$, we use Lemmas IV.9, IV.10 and IV.11 respectively. At each step we rely on Thm. III.13 for the interpretation of bounded FSO_D formulas in MSO_D. □

C. Completeness of MSO_D

We can now prove Theorem. II.3.

Proof of Thm. II.3. Consider a closed formula $\phi \in \Lambda_D$. By Cor. V.3, there is a closed formula $\psi \in \Lambda_D^0$ such that:

$$\mathsf{MSO}_D \vdash \phi \longleftrightarrow \psi$$

By Prop. V.6, there is an automaton $\mathcal{A}$ on the singleton alphabet such that:

$$\mathsf{MSO}_D \vdash \phi \longleftrightarrow \mathcal{L}(\mathcal{A})[\,] \tag{9}$$

By Lem. IV.8 we have that in FSO_D either t to $\{\varnothing\} \vdash t \in \mathcal{L}(\mathcal{A})$ or t to $\{\varnothing\} \vdash t \notin \mathcal{L}(\mathcal{A})$. Hence, by (9) above along with the interpretation of Def. III.11, we have that either $\mathsf{MSO}_D \vdash \phi$ or $\mathsf{MSO}_D \vdash \neg\phi$, as required. □

VI. Positional determinacy

The goal of this section is to prove that abstract games, as defined in our setting, with parity winning conditions are positionally determined, provably in FSO_D and so also MSO_D. This result is critical to show that the languages recognized by alternating tree automata are closed under complement, cf. Lemma IV.9.

Our approach essentially follows that occurring in [14], differing only in the construction of uniform winning strategies. For brevity, here we only present some intermediate results required for the formalization of the argument.

A. Topology of abstract games

Before we proceed we need to introduce some terminology on the *topology* of abstract games. We express all concepts from the point of view of player P, for simplicity. This is sufficient since we will be able to switch to a *dual* game where the roles of P and O are morally switched.

In line with the notational conventions of Def. IV.1 and IV.2, we use subscripts P and O on variables for sets of game positions to denote the subsets of P-positions and O-positions respectively. I.e., for a set of game positions V, we apply Comprehension to define:[10]

$$\begin{array}{lll}(x,y) \mathrel{\dot\in} V_{\mathsf{P}} & \longleftrightarrow & y \mathrel{\ddot\in} \mathsf{P} \wedge (x,y) \mathrel{\dot\in} V \\ (x,y) \mathrel{\dot\in} V_{\mathsf{O}} & \longleftrightarrow & y \mathrel{\ddot\in} \mathsf{O} \wedge (x,y) \mathrel{\dot\in} V\end{array}$$

Of course, as one would expect, we have that $V \doteq V_{\mathsf{P}} \mathbin{\dot\sqcup} V_{\mathsf{O}}$.

Definition VI.1 (Subsets of game positions)**.** *We define the* complement $\widetilde{V}$ *of a set of game positions V by Comprehension as follows:*

$$(x,y) \mathrel{\dot\in} \widetilde{V} \longleftrightarrow (y \mathrel{\ddot\in} \mathsf{P} \sqcup \mathsf{O} \wedge \neg (x,y) \mathrel{\dot\in} V)$$

We define the reachability *condition with respect to U:*

$$\mathrm{Reach}^U(V) := \quad \exists v.(v \mathrel{\dot\in} U \wedge v \mathrel{\dot\in} V)$$

For a winning condition $\mathcal{W}(V)$ and game e we define the winning set $V^{e,\mathcal{W}}$ *by Comprehension as follows:*

$$v \mathrel{\dot\in} V^{e,\mathcal{W}} \longleftrightarrow \exists s \text{ to } \mathsf{O}^{\mathsf{P}}.\mathrm{WinStrat}(s,v,e,\mathcal{W})$$

Finally, a trap V *under a game e is defined as follows:*

$$\begin{array}{rll}\mathrm{Trap}(V,e) := & & \forall u \mathrel{\dot\in} V_{\mathsf{P}} \,.\, \forall v \,.\, (u \xrightarrow[e_{\mathsf{P}}]{} v \;\rightarrow\; v \mathrel{\dot\in} V_{\mathsf{O}}) \\ & \wedge & \forall u \mathrel{\dot\in} V_{\mathsf{O}} \,.\, \exists v \,.\, (u \xrightarrow[e_{\mathsf{O}}]{} v \;\wedge\; v \mathrel{\dot\in} V_{\mathsf{P}})\end{array}$$

An important observation is that, in deadlock-free games, traps induce deadlock-free subgames: each player always has a valid move that remains in the trap.

Proposition VI.2. FSO_D *proves the following:*

$$\begin{array}{ll} & \exists U \mathrm{Play}(U,v,e) \wedge \mathrm{Trap}(V,e) \wedge v \mathrel{\dot\in} V \\ \rightarrow & \exists U' \mathrel{\dot\subseteq} V \,.\, \mathrm{Play}(U',v,e)\end{array}$$

Let us denote the corresponding subgame by $e{\restriction}V$, obtained by Comprehension as follows:

$$u \xrightarrow[e{\restriction}V]{} v \;\longleftrightarrow\; (u \mathrel{\dot\in} V \wedge v \mathrel{\dot\in} V \wedge u \xrightarrow[e]{} v) \tag{10}$$

We also have that complements of winning sets under reachability are traps.

Proposition VI.3. $\mathsf{FSO}_D \vdash \mathrm{Trap}(\widetilde{V}^{e,\mathrm{Reach}^U}, e)$

We now wish to show that plays differing on only finitely many game positions are equi-winning. As a consequence we have that parity winning sets are closed under reachability.

[10]Here we are technically applying Comprehension for function and set variables, due to the translation of variables for game positions.

Definition VI.4. *We write $X \mathrel{\dot\sim} Y$ if X and Y are paths differing on only a finite set, i.e.:*

$$\begin{array}{ll} & \mathrm{Path}^\infty(X) \\ \wedge & \mathrm{Path}^\infty(Y) \\ \wedge & \exists x \mathrel{\dot\in} X \mathbin{\dot\cap} Y \,.\, \forall y \mathrel{\dot\geq} x \,.\, (y \mathrel{\dot\in} X \longleftrightarrow y \mathrel{\dot\in} Y)\end{array} \tag{11}$$

We extend this definition to sets of game positions, $(X,f) \mathrel{\dot\sim} (Y,g)$, as follows:

$$\begin{array}{ll} & X \mathrel{\dot\sim} Y \\ \wedge & \forall x \mathrel{\dot\in} X \,.\, \exists! y \,.\, y \mathrel{\ddot\in} f(x) \\ \wedge & \forall x \mathrel{\dot\in} Y \,.\, \exists! y \,.\, y \mathrel{\ddot\in} g(x) \\ \wedge & \exists x \mathrel{\dot\in} X \mathbin{\dot\cap} Y \,.\, \forall y \geq x \,.\, f(y) \mathrel{\ddot=} g(y)\end{array} \tag{12}$$

Lemma VI.5. $\mathsf{FSO}_D \vdash (U \mathrel{\dot\sim} V \wedge \mathrm{Par}^c(U)) \rightarrow \mathrm{Par}^c(V)$.

We can use this lemma to show that winning sets under parity conditions are already winning sets under reachibility conditions, helping us to induce subgames later.

Theorem VI.6. $\mathsf{FSO}_D \vdash V^{e,\mathrm{Par}^c} \doteq V^{e,\mathrm{Reach}^{V^{e,\mathrm{Par}^c}}}$.

B. Main result

We give the main argument of positional determinacy, assuming the existence of *uniform* winning strategies:

Theorem VI.7 (Uniformity)**.** FSO_D *proves the following,*

$$\exists s \text{ to } \mathsf{O}^{\mathsf{P}}.\forall v \mathrel{\dot\in} V^{e,\mathcal{W}}.\mathrm{WinStrat}(s,v,e,\mathcal{W})$$

for $\mathcal{W} \in \{\mathrm{Par}, \mathrm{Reach}\}$.

We give a construction of such strategies in the next section.

Definition VI.8 (Dual and union games)**.** *Let e, e' be games on label sets P and O, we define the* dual *game $\widetilde{e}$ by Comprehension as follows:*

$$\begin{array}{lll}\widetilde{e}_{\mathsf{P}}(x) & \mathrel{\ddot=} & \{(q,\gamma) \in \mathsf{P} \times \mathsf{O} : (q,\gamma') \mathrel{\ddot\in} e_{\mathsf{P}}(x) \Rightarrow \gamma \cap \gamma' \neq \varnothing\} \\ \widetilde{e}_{\mathsf{O}}(x) & \mathrel{\ddot=} & \{(d,\gamma,q) \in D \times \mathsf{O} \times \mathsf{P} : (q,d) \in \gamma\} \\ \widetilde{e}(x) & \mathrel{\ddot=} & \widetilde{e}_{\mathsf{P}}(x) \mathbin{\ddot\cup} \widetilde{e}_{\mathsf{O}}(x)\end{array}$$

We also define the union *game $e \mathbin{\ddot\cup} e'$ by Comprehension:*

$$(e \mathbin{\ddot\cup} e')(x) = e(x) \mathbin{\ddot\cup} e'(x)$$

Our notion of duality commutes with arbitrary restrictions:

Proposition VI.9. $\mathsf{FSO}_D \vdash \widetilde{e{\restriction}V} \doteq \widetilde{e}{\restriction}V$.

In the general case, for arbitrary V, the restricted games might not be deadlock-free.

Definition VI.10 (Dual and equivalent parity conditions)**.** *Let $c, c' : \mathsf{P} \to [0,n]$ be parity functions. We say that c and c' are* equivalent *if:*

1) *$c(q) < c(q')$ iff $c'(q) < c'(q')$.*
2) *$c(q)$ is even iff $c'(q)$ is even.*

We also define $\widetilde{c}$, the dual *of c, by $\widetilde{c}(q) = c(q) + 1$.*

As expected, we can prove in FSO_D that equivalent parity conditions are equi-winning and that dual conditions are winning on complementary sets.

Proposition VI.11. *Let c be a parity function $\mathsf{P} \to [0,n]$. We have that $\mathsf{FSO}_D \vdash \mathrm{Par}^c(V) \longleftrightarrow \neg\mathrm{Par}^{\widetilde{c}}(V)$.*

If c' is a parity function equivalent to c then $\mathsf{FSO}_D \vdash \mathrm{Par}^c(V) \longleftrightarrow \mathrm{Par}^{c'}(V)$.

In particular we can assume, without loss of generality, that each parity function has image $[0,n)$ or $(0,n]$ for some n.

We are now ready to present our main result, for which we give only a brief outline of a proof, working in an arbitrary Henkin model of FSO_D.

Theorem VI.12. FSO_D *proves the following:*

$$\forall v.\left(\mathrm{Game}(e,\mathsf{P},\mathsf{O}) \to \left[\begin{array}{ll} & \exists s.\mathrm{WinStrat}(v,s,e,\mathrm{Par}^c) \\ \vee & \exists s.\mathrm{WinStrat}(v,s,\widetilde{e},\mathrm{Par}^{\widetilde{c}}) \end{array}\right]\right)$$

Proof sketch. The argument is by an external induction on the size of the image of c. Assuming the image of c is $[0,n)$ and $\widetilde{c}$ is $(0,n]$, by Prop. VI.11, we show that the complement of the winning set for $\widetilde{e}$, say V, is the winning set for e.

Notice that V is a trap in $\widetilde{e}$ by Thm. VI.6 and Prop. VI.3, so let us restrict to the subgame $e{\restriction}V$, appealing to Prop. VI.9.

If $v \in V$ is in the winning set under reachability, say R_0, for the set of positions colored 0 by c, say C_0, then let us associate v with the uniform strategy winning this reachability game. Otherwise, v is in a trap, say U, in e with no 0-colored positions, and so we can associate v with the uniform strategy winning in $e{\restriction}U$, by the inductive hypothesis.[11] Since these two sets of positions are disjoint, we can take the union of the two uniform strategies to give a uniform strategy s^V for V.

Now, any play P in V under s^V either visits R_0 infinitely often or it eventually remains in U. In the former case P must also visit C_0 infinitely often by the reachability strategy, and is winning by definition of Par_c, and in the latter case P is winning by the strategy for $e{\restriction}U$, concluding our proof. □

C. Construction of uniform strategies

In this section we give the construction of the uniform winning strategies that we required for the positional determinacy proof, Thm. VI.12. For arbitrary parity games, this seems to require the Axiom of Choice or transfinite well-orderings, which are unavailable in the setting of FSO_D and MSO_D. In our argument, we instead rely on the fact that the game graph induced by a tree automaton is a product of trees, from which we can make use of the underlying tree order.

In what follows let us fix a well-order, $\leq$, on O^{P} and parametrise our results by some winning condition $\mathcal{W} \in \{\mathrm{Par}, \mathrm{Reach}\}$ on a game e.

Definition VI.13 (Ordering strategies)**.** *For an individual symbol x, we define $s \dot{\leq}_x s'$ as:*

$$\begin{array}{ll} & \exists y \dot{\leq} x.\exists z \ddot{\in} \mathsf{P} \ddot{\cup} \mathsf{O}. \\ & \quad \mathrm{WinStrat}(s,(y,z),e,\mathrm{Par}^c) \\ \wedge & \left[\begin{array}{l} \forall y' \dot{\leq} x.\forall z \ddot{\in} \mathsf{P} \ddot{\cup} \mathsf{O}. \\ \left(\begin{array}{ll} & \mathrm{WinStrat}(s',(y',z'),e,\mathrm{Par}^c) \\ \to & y \dot{<} y' \vee (y \doteq y' \wedge s(y) \ddot{\leq} s(y')) \end{array}\right) \end{array}\right] \end{array}$$

[11] Recall that we already have that v is not in the winning set for $\widetilde{e}$.

We also define a 'least' strategy with respect to a formula ϕ:[12]

$$\mathrm{Least}_x(s,\phi,f) \quad := \quad \phi[f/s] \wedge \forall g.(\phi[g/s] \to f \dot{\leq}_x g)$$

Finally we give a definition of the uniform strategy from a winning position.

Definition VI.14 (Uniform strategies)**.** *We define $s^{\mathrm{un}}(x) \doteq y$ by Comprehension as follows:*

$$\begin{array}{ll} & y \ddot{\in} \mathsf{O}^{\mathsf{P}} \\ \wedge & \forall z \ddot{\in} \mathsf{P}\ .\ \forall f\ . \\ & \left(\begin{array}{ll} & \mathrm{Least}_x(s,\mathrm{WinStrat}(s,(x,z),e,\mathcal{W}),f) \\ \to & y(z) \ddot{=} f(z) \end{array}\right) \end{array}$$

The idea here is that a "minimal" winning strategy from some game position (x,l) is a strategy, winning from (x,l), such that the first tree node $\leq x$ from which this strategy is winning (for some label in $\mathsf{P} \sqcup \mathsf{O}$) is minimal among the set of such tree nodes induced by each strategy winning from (x,l). If there is not a unique such minimal strategy, i.e. there are distinct strategies associated with the same minimal first tree node, we make a choice by invoking the well-order $\leq$ on O^{P}.

In this way, the game position witnessing the minimality of the current strategy can only decrease during a play. Hence each play of the uniform strategy eventually converges to a play of some winning strategy, whence we invoke Lemma VI.5 to deduce that it is winning.

Theorem VI.15. FSO_D *proves the following:*

$$v \dot{\in} V^{e,\mathcal{W}} \to \mathrm{WinStrat}(s^{\mathrm{un}},v,e,\mathcal{W}) \tag{13}$$

VII. Further work

A. On the proof-theoretic strength of MSO

An important future line of research is to determine the *proof-theoretic strength* of MSO (over words or trees) as a subsystem of PA2. For this, it is customary to compare subsystems of PA2 obtained by restricting the Comprehension scheme to formulas of given logical complexity [12].

We conjecture that the axiomatization of MSO on ω-words can be carried out in a system called *Weak Koening Lemma*. For the case of trees, the topological complexity of MSO suggests that at least Δ^1_2-Comprehension is required.

We point out that perhaps our most interesting use of Comprehension was in the definition of s^{un} in Def. VI.14, on a Π^1_2 MSO formula (under the interpretation of Def. III.11).

B. Some remarks on proof search

One outcome of this work is the possibility to implement decision procedures for SnS based on proof search. Our main result, the complete axiomatization, can be seen as a 'proof of concept' for this approach:

Algorithm VII.1. *Under an input closed MSO_D-formula ϕ, enumerate all MSO_D-proofs until one with conclusion ϕ or $\neg\phi$ is reached.*

[12] This formula should be read, "the least strategy s, with respect to x, satisfying a formula $\phi(s)$ is f."

Of course, this is not a very sophisticated algorithm, and it is worth restating that its correctness is itself due to the usual automata-logics argument. However the algorithm, nonetheless, makes no mention of automata and so can be adapted and improved purely in the setting of proof theory. In this sense, the algorithm is the first of its kind: a decision procedure for SnS that remains internal to the language.

We now state some ideas for future work on how to adapt this proof search procedure. They are fundamentally linked to notions of proof-theoretic strength explained above.

a) Complexity of nonlogical rules: At the heart of the difficulty of proof search is the complexity of Induction and Comprehension formulas used, i.e. proof-theoretic strength as we previously discussed.[13] This is due to the 'free-cut elimination' theorem, allowing a proof to be transformed into one where all formulas occurring are subformulas[14] of the conclusion, an axiom, or a nonlogical step.

b) Towards a terminating bottom-up proof search procedure: An interesting course of future work would be to conduct a bona fide proof search procedure, starting from the conclusion and building the proof bottom-up.

Progress towards this might be made by the following related goals:

1) Eliminate as many cuts as possible.
2) Eliminate as many nonlogical steps as possible.

One avenue worth considering is that there are implementations of Comprehension as inference rules which admit cut-reduction steps. Usually the problem with this approach is that these steps do not terminate, but due to the inherent decidability of SnS, and termination of proof-search in MSO_D, perhaps there is some hope to adapt the cut-reductions to achieve termination.

One could also choose to work in the FSO setting. Here, since Choice is valid in the standard model of MSO_D (and indeed is an axiom of FSO_D), one can apply Choice *eagerly* via Skolemization, eliminating the consideration of where it might occur in a proof. The trade-off here is the added difficulty of working with function symbols, but this is nonetheless a trade-off worth exploring.

c) Proof interaction and verification: Even without any improvement on the proof search algorithm, the very presence of a proof theory admits the possibility of *interactive* proofs, ones where a user may state as much or as little of the proof information to a proof assistant. For example, if the user gives all required Comprehension formulas then the rest of the proof can be reconstructed bottom-up quite efficiently.

In the same vein, a complete axiomatization provides a means to *communicate* proofs that can be checked efficiently, rather than having to redecide formulas. Such a tool might be useful for *cyclic* theorem provers, where correctness criteria boil down to checking the inclusion between Büchi automata, a priori a very complex procedure.

[13]Due to our formulation of Induction, however, this complexity is devolved to just the instances of Comprehension in a proof.

[14]In the usual 'wide' sense when working in first-order logic.

VIII. Concluding remarks

We presented a complete axiomatization of MSO on D-ary trees, for D an arbitrary non-empty finite set. Our axiomatization is a natural subsystem of PA2. This provides an interesting basis to study proof-theoretical aspects of MSO, and also paves the way for new decision procedures for MSO formulas; this will be the object of future investigations.

Acknowledgments

Early stages of this work greatly benefited from the participation of Alexander Kreuzer, who suggested a formalization of the algebraic approach of [1] instead of the translation of formulas to automata. Such algebraic tools might be useful for further work on proof-theoretic complexity and for the axiomatization of MSO on the countably branching tree $\mathbb{N}^*$.

This work also benefited from regular discussions with Thomas Colcombet and Arnaud Carayol.

References

[1] A. Blumensath, "An algebraic proof of Rabin's Tree Theorem," *Theor. Comput. Sci.*, vol. 478, pp. 1–21, 2013.

[2] J. R. Büchi and D. Siefkes, "Axiomatization of the Monadic Second Order Theory of ω_1," in *Decidable Theories II : The Monadic Second Order Theory of All Countable Ordinals*, ser. LNM, J. R. Büchi and D. Siefkes, Eds. Springer, 1973, vol. 328, pp. 129–217.

[3] A. Carayol and C. Löding, "MSO on the Infinite Binary Tree: Choice and Order," in *CSL*, ser. Lecture Notes in Computer Science, vol. 4646. Springer, 2007, pp. 161–176.

[4] E. A. Emerson and C. S. Jutla, "Tree Automata, Mu-Calculus and Determinacy (Extended Abstract)," in *FOCS*. IEEE Computer Society, 1991, pp. 368–377.

[5] A. Gheerbrant and B. ten Cate, "Complete Axiomatizations of Fragments of Monadic Second-Order Logic on Finite Trees," *Logical Methods in Computer Science*, vol. 8, no. 4, 2012.

[6] E. Grädel, W. Thomas, and T. Wilke, Eds., *Automata, Logics, and Infinite Games: A Guide to Current Research*, ser. Lecture Notes in Computer Science, vol. 2500. Springer, 2002.

[7] S. Gurevich and S. Shelah, "Rabin's Uniformization Problem," *J. Symb. Log.*, vol. 48, no. 4, pp. 1105–1119, 1983.

[8] D. E. Muller and P. E. Schupp, "Simulating Alternating Tree Automata by Nondeterministic Automata: New Results and New Proofs of the Theorems of Rabin, McNaughton and Safra," *Theor. Comput. Sci.*, vol. 141, no. 1&2, pp. 69–107, 1995.

[9] M. O. Rabin, "Decidability of Second-Order Theories and Automata on Infinite Trees," *Transactions of the American Mathematical Society*, vol. 141, pp. 1–35, 1969.

[10] C. Riba, "A model theoretic proof of completeness of an axiomatization of monadic second-order logic on infinite words," in *Proceedings of IFIP-TCS'12*, 2012.

[11] D. Siefkes, *Decidable Theories I : Büchi's Monadic Second Order Successor Arithmetic*, ser. LNM. Springer, 1970, vol. 120.

[12] S. Simpson, *Subsystems of Second Order Arithmetic*, 2nd ed., ser. Perspectives in Logic. Cambridge University Press, 2010.

[13] B. ten Cate and G. Fontaine, "An Easy Completeness Proof for the Modal μ-Calculus on Finite Trees," in *FOSSACS*, ser. Lecture Notes in Computer Science, vol. 6014. Springer, 2010, pp. 161–175.

[14] W. Thomas, "Languages, Automata, and Logic," in *Handbook of Formal Languages*, G. Rozenberg and A. Salomaa, Eds. Springer, 1997, vol. III, pp. 389–455.

[15] D. van Dalen, *Logic and Structure*, 4th ed., ser. Universitext. Springer, 2004.

[16] I. Walukiewicz, "Completeness of Kozen's Axiomatisation of the Propositional μ-Calculus," *Information and Computation*, vol. 157, no. 1-2, pp. 142–182, 2000.

[17] ——, "Monadic second-order logic on tree-like structures," *Theor. Comput. Sci.*, vol. 275, no. 1-2, pp. 311–346, 2002.