16 The Monadic Theory of Tree-like Structures

Dietmar Berwanger and Achim Blumensath

Mathematische Grundlagen der Informatik
RWTH Aachen

16.1 Introduction

Initiated by the work of Büchi, Läuchli, Rabin, and Shelah in the late 60s, the investigation of monadic second-order logic (MSO) has received continuous attention. The attractiveness of MSO is due to the fact that, on the one hand, it is quite expressive subsuming – besides first-order logic – most modal logics, in particular the μ-calculus. On the other hand, MSO is simple enough such that model-checking is still decidable for many structures. Hence, one can obtain decidability results for several logics by just considering MSO.

For these reasons it is an important task to classify those structures for which MSO model-checking is decidable. So far, only partial results are known and it seems doubtful whether a complete characterisation can be obtained.

On the one hand, a useful tool to prove undecidability is the result that MSO model-checking for the grid $\omega \times \omega$ is undecidable. On the other hand, Rabin's famous tree theorem states that, for the complete binary tree, model-checking is decidable. Since many structures can be interpreted in the binary tree this provides a wide range of decidability results. Furthermore, we often only need to consider trees, as many modal logics have the tree-model property.

In this chapter we present a generalisation of Rabin's Tree Theorem. Given a structure $\mathfrak{A}$ we construct its *iteration* $\mathfrak{A}^*$ which is a tree whose vertices are finite sequences of elements of $\mathfrak{A}$. For each relation R of $\mathfrak{A}$ its iteration has the relation

$$R^* := \{\, (wa_0, \dots, wa_r) \mid \bar{a} \in R,\ w \in A^* \,\}.$$

Additionally, we include the successor relation son containing all pairs (w, wa) for $w \in A^*$, $a \in A$, and the clone relation cl consisting of all elements of the form waa. Muchnik's Theorem states that model-checking is decidable for $\mathfrak{A}$ if and only if it is so for $\mathfrak{A}^*$. The first published proof appears in Semenov [3]. It generalises an unpublished result of Stupp [5] described in Shelah [4] where the clone relation was left out. Our presentation follows Walukiewicz [6].

For the proof we employ the usual technique of translating formulae into automata and vice versa. Since, in general, we are operating on trees of infinite degree, a new type of automaton is needed where the transition function is defined by MSO-formulae. Furthermore, in order to handle the clone relation, the transition function has to depend on the current position in the input tree.

In the next section we introduce the kind of automaton we will use to prove Muchnik's Theorem but in a more general version than needed, and we prove that these automata are closed under boolean operations and projection.

E. Grädel, W. Thomas, and T. Wilke (Eds.): Automata, Logics, and Infinite Games, LNCS 2500, pp. 285-301, 2002.

In Section 16.3 we will restrict the class of automata to those with MSO-definable transition function and the translation between automata and MSO-formulae is presented.

Finally, Section 16.4 contains the proof of Muchnik's Theorem.

16.2 Automata

To fix our notation, let $[n] := \{0, \dots, n-1\}$. By $\mathcal{B}^+(X)$ we denote the set of (infinitary) positive boolean formulae over X, i.e., all formulae constructed from X with disjunction and conjunction. An interpretation of a formula $\varphi \in \mathcal{B}^+(X)$ is a set $I \subseteq X$ of atoms we consider true. A **Σ-labelled A-tree** is a function $T : A^* \to \Sigma$ which assigns a label $T(w)$ to each vertex $w \in A^*$.

The main tool used for the investigation of MSO are automata on A-trees. Since A is not required to be finite we need a model of automaton which can work with trees of arbitrary degree. In addition the clone relation cl makes it necessary that the transition function depends on the current position in the input tree. Thus, we define a very general type of automaton which we will restrict suitably in the next section.

Definition 16.1. A **tree automaton** is a tuple

$$\mathcal{A} = (Q, \Sigma, A, \delta, q_{\mathrm{I}}, W)$$

where the input is a Σ-labelled A-tree, Q is the set of states, q_{I} is the initial state, $W \subseteq Q^\omega$ is the acceptance condition, and

$$\delta : Q \times \Sigma \to \mathcal{B}^+(Q \times A)^{A^*}$$

is the transition function which assigns to each state q and input symbol c a function $\delta(q, c) : A^* \to \mathcal{B}^+(Q \times A)$. Frequently we will write $\delta(q, c, w)$ instead of $\delta(q, c)(w)$.

Note that the transition function and acceptance condition of these automata are not finite. To obtain finite automata we will represent the transition function by an MSO-formula and consider only parity acceptance conditions in the next section. For simplicity all results in this section are stated and proved for the general model.

The language accepted by an automaton is defined by way of games. Recall that a game $\mathcal{G} = (V_0, V_1, E, W)$ consists of sets V_0 and V_1 of vertices associated to the respective players, an edge relation E, and the set of winning plays W.

Definition 16.2. Let $\mathcal{A} = (Q, \Sigma, A, \delta, q_{\mathrm{I}}, W)$ be an automaton and $T : A^* \to \Sigma$ a tree. The game $\mathcal{G}(\mathcal{A}, T)$ is defined as follows:

(a) The set of vertices is

$$\big(Q \cup \mathcal{B}^+(Q \times A)\big) \times A^*.$$

V_0 consists of all pairs $(q, w) \in Q \times A^*$ and all pairs of the form (φ, w) where φ is either atomic or a disjunction, and V_1 consists of all pairs where φ is a conjunction.

(b) The initial position is $(q_{\mathrm{I}}, \varepsilon)$.

(c) Each node (q, w) has the successor $\bigl(\delta(q, T(w), w), w\bigr)$. The successors of nodes $(\varphi \circ \psi, w)$ are (φ, w) and (ψ, w) where $\circ$ is either $\wedge$ or $\vee$. Finally, the successor of nodes $\bigl((q, a), w\bigr)$ with atomic formulae is (q, wa).

(d) Let $(\xi_i, w_i)_{i<\omega}$ be a play. Consider the subsequence $(\xi_{i_k}, w_{i_k})_{k<\omega}$ of positions where $\xi_{i_k} = q_k$ is a state. The play is winning if the sequence $q_0 q_1 \dots$ is in W.

The language $L(\mathcal{A})$ recognised by $\mathcal{A}$ is the set of all trees T such that player 0 has a winning strategy for the game $\mathcal{G}(\mathcal{A}, T)$.

Sometimes it is more convenient to use a simpler game where several moves of the same kind are replaced by a single one. Assume that δ is in disjunctive normal form. The abridged game $\check{\mathcal{G}}(\mathcal{A}, T)$ is defined by replacing (a) and (c) in the above definition by:

(a′) The set of vertices consists of $V_0 := Q \times A^*$ and $V_1 := \mathscr{P}(Q \times A) \times A^*$.

(c′) Each node $(q, w) \in V_0$ with $\delta(q, T(w), w) = \bigvee_i \bigwedge \Phi_i$ has the successors (Φ_i, w) for each i. The successors of some node $(\Phi, w) \in V_1$ are the nodes (q, wa) for $(q, a) \in \Phi$.

Both versions of the game are obviously equivalent. In the following sections we will consider only parity acceptance conditions.

Definition 16.3. A **parity condition** is given by a function $\Omega : Q \to [n]$ and defines the set of all sequences $(q_i)_{i<\omega} \in Q^\omega$ such that the least number appearing infinitely often in the sequence $(\Omega(q_i))_{i<\omega}$ is even.

In the remainder of this section we will prove that automata as defined above are closed under union, complement, and projection. This property is needed in the next section in order to translate formulae into automata. We start with the union.

Definition 16.4. Let $\mathcal{A}_i = (Q_i, \Sigma, A, \delta_i, q_i^{\mathrm{I}}, W_i)$, $i = 1,\ 2$, be tree automata. Their **sum** is the automaton

$$\mathcal{A}_0 + \mathcal{A}_1 := \bigl(Q_1 \cup Q_2 \cup \{q_{\mathrm{I}}\}, \Sigma, A, \delta, q_{\mathrm{I}}, W\bigr)$$

where

$$\begin{aligned} \delta(q, c, w) &:= \delta_i(q, c, w) && \text{for } q \in Q_i, \\ \delta(q_{\mathrm{I}}, c, w) &:= \delta_0(q_0^{\mathrm{I}}, c, w) \vee \delta_1(q_1^{\mathrm{I}}, c, w), \end{aligned}$$

and W consists of all sequences $q_0 q_1 q_2 \dots$ such that $q_0 = q_{\mathrm{I}}$ and $q_i^{\mathrm{I}} q_1 q_2 \dots \in W_i$ for some i.

Lemma 16.5. $L(\mathcal{A}_0 + \mathcal{A}_1) = L(\mathcal{A}_0) \cup L(\mathcal{A}_1)$.

Proof. Note that $\mathcal{G}(\mathcal{A}_0 + \mathcal{A}_1, T)$ consists of disjoint copies of $\mathcal{G}(\mathcal{A}_0, T)$ and $\mathcal{G}(\mathcal{A}_1, T)$, and a new initial position from which player 0 has to choose one of the two subgames. Obviously, each winning strategy for player 0 in $\mathcal{G}(\mathcal{A}_0, T)$ or $\mathcal{G}(\mathcal{A}_1, T)$ is also a winning strategy in $\mathcal{G}(\mathcal{A}_0 + \mathcal{A}_1, T)$. On the other hand, if σ is a winning strategy for player 0 in the compound game it is also winning in either $\mathcal{G}(\mathcal{A}_0, T)$ or $\mathcal{G}(\mathcal{A}_1, T)$ depending on which subgame player 0 chooses in his first move. □

Complementation is easy as well.

Definition 16.6. Let $\mathcal{A} = (Q, \Sigma, A, \delta, q_{\mathrm{I}}, W)$. $\bar{\mathcal{A}} := (Q, \Sigma, A, \bar{\delta}, q_{\mathrm{I}}, \bar{W})$ is the automaton with

$$\bar{\delta}(q, c, w) := \overline{\delta(q, c, w)} \qquad \text{and} \qquad \bar{W} := Q^\omega \setminus W.$$

Here $\overline{\varphi}$ denotes the dual of φ, i.e., the formula where each $\wedge$ is replaced by $\vee$ and vice versa.

Lemma 16.7. $T \in L(\bar{\mathcal{A}})$ iff $T \notin L(\mathcal{A})$.

Proof. Let $\mathcal{G}(\bar{\mathcal{A}}, T) = (\bar{V}_0, \bar{V}_1, \bar{E}, \bar{W})$. Note that in $\mathcal{G}(\bar{\mathcal{A}}, T)$ the roles of player 0 and 1 are exchanged. $\bar{V}_0$ consists of all former V_1-nodes, and $\bar{V}_1$ contains all V_0-nodes except for the atomic ones. Since the latter have exactly one successor it is irrelevant which player they are assigned to. Thus, each choice of player 0 in the old game is made by player 1 in the new one and vice versa. Hence, each winning strategy σ for player 0 in $\mathcal{G}(\mathcal{A}, T)$ is a strategy for player 1 in $\mathcal{G}(\bar{\mathcal{A}}, T)$ which ensures that the resulting play induces a sequence in $W = Q^\omega \setminus \bar{W}$. Thus, σ is winning for 1. The other direction follows by symmetry. □

The closure under projections is the hardest part to prove. The projection $\Pi(L)$ of a tree-language L is the set of all trees $T : A^* \to \Sigma$ such that there is a tree $T' : A^* \to \Sigma \times \{0, 1\}$ in L with $T'(w) = (T(w), i_w)$ for some $i_w \in \{0, 1\}$ and all $w \in A^*$.

The proof is split into several parts. We prove closure under projection for non-deterministic automata, and show that each alternating automaton can be transformed into an equivalent non-deterministic one.

Definition 16.8. An automaton $\mathcal{A} := (Q, \Sigma, A, \delta, q_{\mathrm{I}}, W)$ is **non-deterministic** if each formula $\delta(q, c, w)$ is in disjunctive normal-form $\bigvee_i \bigwedge_k (q_{ik}, a_{ik})$ where, for each fixed i, all the a_{ik} are different.

Definition 16.9. Let $\mathcal{A} = (Q, \Sigma \times \{0, 1\}, A, \delta, q_{\mathrm{I}}, W)$ be a non-deterministic automaton. Define $\mathcal{A}_\Pi := (Q, \Sigma, A, \delta_\Pi, q_{\mathrm{I}}, W)$ where

$$\delta_\Pi(q, c, w) := \delta(q, (c, 0), w) \vee \delta(q, (c, 1), w).$$

Lemma 16.10. $L(\mathcal{A}_\Pi) = \Pi(L(\mathcal{A}))$

Proof. ($\supseteq$) Let σ be a winning strategy for player 0 in $\mathcal{G}(\mathcal{A}, T)$. $\mathcal{G}(\mathcal{A}_\Pi, \Pi(T))$ contains additional vertices of the form $(\varphi_0 \vee \varphi_1, w)$ where $\varphi_i = \delta(q, (c, i), w)$. By defining

$$\sigma(\varphi_0 \vee \varphi_1, w) := \varphi_i \quad \text{for the } i \text{ with } T(w) = (c, i)$$

we obtain a strategy for player 0 in the new game. This strategy is winning since, if one removes the additional vertices from a play according to the extended strategy, a play according to σ in the original game is obtained which is winning by assumption.

($\subseteq$) Let σ be a winning strategy for player 0 in $\mathcal{G}(\mathcal{A}_\Pi, T)$. We have to define a tree $T' \in L(\mathcal{A})$ with $T = \Pi(T')$. Since $\mathcal{A}_\Pi$ is non-deterministic the game has the following structure: At each position $((q, a), w)$ with

$$\delta(q, T(w), w) = \textstyle\bigvee_i \bigwedge_k (q_{ik}, a_{ik})$$

player 0 chooses some conjunction $\bigwedge_k (q_{ik}, a_{ik})$ out of which player 1 picks a successor (q_{ik}, a_{ik}). Thus, for each word $w \in A^*$ there is at most one state q such that a play according to σ reaches the position (q, w). Let $\sigma(\varphi_0 \vee \varphi_1, w) = (\varphi_i, w)$ where $\varphi_0 \vee \varphi_1 = \delta(q, T(w), w)$. We define T' by $T'(w) := (T(w), i)$. $\square$

It remains to show how to translate alternating automata to non-deterministic ones. To do so we need some notation to modify transition relations.

Definition 16.11. Let $\varphi \in \mathcal{B}^+(Q \times A)$.

(a) The **collection** of φ is defined as follows. Let $\bigvee_i \bigwedge_k (q_{ik}, a_{ik})$ be the disjunctive normal-form of φ.

$$\operatorname{collect}(\varphi) := \bigvee_i \bigwedge_{a \in A} \big(Q_i(a), a\big) \in \mathcal{B}^+(\mathscr{P}(Q) \times A)$$

where $Q_i(a) := \{\, q_{ik} \mid a_{ik} = a \,\}$.

(b) Let $q' \in Q'$. The **shift** of φ by q' is the formula $\operatorname{sh}_{q'} \varphi \in \mathcal{B}^+(Q' \times Q \times A)$ obtained from φ by replacing all atoms (q, a) by (q', q, a).

(c) For $S \subseteq Q \times Q$ let

$$(S)_2 := \{\, q \mid (q', q) \in S \text{ for some } q' \,\}.$$

The translation is performed in two steps. First, the alternating automaton is transformed into a non-deterministic one with an obscure non-parity acceptance condition. Then, the result is turned into a normal non-deterministic parity automaton. The construction used for the first step is the usual one. For each node of the input tree the automaton stores the set of states of the original automaton from which the corresponding subtree must be accepted. That is, for universal choices of the alternating automaton, all successors are remembered, whereas for existential choices, only one successor is picked non-deterministically. What makes matters slightly more complicated is the fact that, in order to define the acceptance condition, the new automaton has to remember not only the set of current states but their predecessors as well, i.e., its states are of the form (q', q) where q is the current state of the original automaton and q' is the previous one.

Definition 16.12. Let $\mathcal{A} = (Q, \Sigma, A, \delta, q_{\mathrm{I}}, W)$ be an alternating automaton.

$$\mathcal{A}_{\mathrm{n}} := \big(\mathscr{P}(Q \times Q), \Sigma, A, \delta_{\mathrm{n}}, \{(q_{\mathrm{I}}, q_{\mathrm{I}})\}, W_{\mathrm{n}}\big)$$

is the automaton where

$$\delta_{\mathrm{n}}(S, c, w) := \text{collect} \bigwedge_{q \in (S)_2} \mathrm{sh}_q\, \delta(q, c, w).$$

A sequence $(q_i)_{i<\omega} \in Q^\omega$ is called a **trace** of $(S_i)_{i<\omega} \in \mathscr{P}(Q \times Q)^\omega$ if $(q_i, q_{i+1}) \in S_i$ for all $i < \omega$. W_n consists of all sequences $(S_i)_{i<\omega} \in \mathscr{P}(Q \times Q)^\omega$ such that every trace of $(S_i)_{i<\omega}$ is in W.

Lemma 16.13. *$\mathcal{A}_{\mathrm{n}}$ is a non-deterministic automaton with $L(\mathcal{A}_{\mathrm{n}}) = L(\mathcal{A})$.*

Proof. The definition of collect ensures that $\mathcal{A}_{\mathrm{n}}$ is non-deterministic.

($\supseteq$) Let $T \in L(\mathcal{A})$ and let σ be the corresponding winning strategy for player 0 in $\check{\mathcal{G}}(\mathcal{A}, T)$. To define a strategy σ_{n} in $\check{\mathcal{G}}(\mathcal{A}_{\mathrm{n}}, T)$ consider a position $(S, w) \in \check{\mathcal{G}}(\mathcal{A}, T)$. Let $\sigma(q, w) = (\Phi_q, w)$ for $q \in (S)_2$. We define $\sigma_{\mathrm{n}}(S, w) := (\text{collect} \bigwedge \Phi, w)$ where

$$\Phi = \bigcup_{q \in (S)_2} \mathrm{sh}_q\, \Phi_q.$$

This is valid since $(\text{collect} \bigwedge \Phi, w)$ is a successor of (q, w).

To show that σ_{n} is a winning strategy consider the result $(S_i)_{i<\omega}$ of a play according to σ_{n}. If $(\Phi, w) \in \sigma_{\mathrm{n}}(S_i, w)$ and $(S_{i+1}, a) \in \Phi$, then for each $(q, q') \in S_{i+1}$ it holds that $(q', a) \in \Phi_q$. Thus, all traces of $(S_i)_{i<\omega}$ are plays according to σ and therefore winning.

($\subseteq$) Let σ_{n} be a – not necessarily memoryless – winning strategy for player 0 in $\check{\mathcal{G}}(\mathcal{A}_{\mathrm{n}}, T)$. We construct a winning strategy for player 0 in $\check{\mathcal{G}}(\mathcal{A}, T)$ as follows. Let p_{n} be a play according to σ_{n} in $\check{\mathcal{G}}(\mathcal{A}_{\mathrm{n}}, T)$ with last position (S, w), and let p be the play according to σ. By induction we ensure that the last position in p is of the form (q, w) for some $q \in (S)_2$. Let $(\Phi_{\mathrm{n}}, w) = \sigma_{\mathrm{n}}(p_{\mathrm{n}})$ and define

$$\Phi := \{\, (q', a) \mid (S', a) \in \Phi_{\mathrm{n}} \text{ and } ((q, q'), a) \in S' \text{ for some } S' \,\}.$$

Then $\bigwedge \Phi$ is a conjunction in $\delta(q, T(w), w)$, by definition of δ_{n}, and we can set $\sigma(p) := (\Phi, w)$. The answer of player 0 to this move consists of some position (q', wa) for $(q', a) \in \Phi$. Suppose that in $\check{\mathcal{G}}(\mathcal{A}_{\mathrm{n}}, T)$ player 1 chooses the position (S_a, wa) where S_a is the unique state such that $(S_a, a) \in \Phi_{\mathrm{n}}$. Since $(q, q') \in S_a$ the induction hypothesis is satisfied for the extended plays $p(\Phi, w)(q', wa)$ and $p_{\mathrm{n}}(\Phi_{\mathrm{n}}, w)(S_a, a)$.

It follows that each play p according to σ in $\check{\mathcal{G}}(\mathcal{A}, T)$ is a trace of some play p_{n} according to σ_{n} and therefore winning by construction of $\mathcal{A}_{\mathrm{n}}$. □

The automaton $\mathcal{A}_{\mathrm{n}}$ constructed above does not have a parity acceptance condition. Since we intend to consider only parity automata in the next section, we have to construct a non-deterministic automaton with such an acceptance condition. It is easy to see that, provided that the original automaton does have a parity acceptance condition, there is some parity automaton on infinite words $\mathcal{B} = (P, \mathscr{P}(Q \times Q), \delta, p^0, \Omega)$ which recognises $W_{\mathrm{n}} \subseteq \mathscr{P}(Q \times Q)^\omega$. Let $\mathcal{A}_{\mathrm{p}}$ be the product automaton of $\mathcal{A}_{\mathrm{n}}$ and $\mathcal{B}$, that is,

$$\mathcal{A}_{\mathrm{p}} = \big(P \times \mathscr{P}(Q \times Q),\ \Sigma,\ A,\ \delta_{\mathrm{p}},\ (p^0, q_{\mathrm{n}}^{\mathrm{I}}),\ \Sigma_{\mathrm{p}}\big)$$

where

$$\delta_{\mathrm{p}}((p,S),c,w) = \mathrm{sh}_{p'}\, \delta_{\mathrm{n}}(S,c,w) \qquad \text{for } p' := \delta(p,S)$$

and $\Omega_{\mathrm{p}}(p,S) = \Omega(p)$.

Lemma 16.14. *$\mathcal{A}_{\mathrm{p}}$ is a parity automaton with $L(\mathcal{A}_{\mathrm{p}}) = L(\mathcal{A}_{\mathrm{n}})$.*

Proof. Let σ be a winning strategy for player 0 in $\check{\mathcal{G}}(\mathcal{A}_{\mathrm{n}}, T)$. We define a corresponding strategy σ' in $\check{\mathcal{G}}(\mathcal{A}_{\mathrm{p}}, T)$ by

$$\sigma'((p,S),w) := \big(\mathrm{sh}_{p'}\, \Phi, w\big)$$

where $(\Phi, w) = \sigma(S, w)$ and $p' = \delta(p, S)$. That way every play

$$\big((p_0, S_0), w_0\big)\big(\Phi'_0, w_0\big)\big((p_1, S_1), w_1\big)\big(\Phi'_1, w_1\big)\dots$$

in $\check{\mathcal{G}}(\mathcal{A}_{\mathrm{p}}, T)$ according to σ' is induced by a play

$$(S_0, w_0)(\Phi_0, w_0)(S_1, w_1)(\Phi_1, w_1)\dots$$

in $\check{\mathcal{G}}(\mathcal{A}_{\mathrm{n}}, T)$ according to σ. Further, $(p_i)_{i<\omega}$ is the run of $\mathcal{B}$ on $(S_i)_{i<\omega}$. Since the second play is winning, the first one is so as well, by definition of $\mathcal{B}$. Hence, σ' is a winning condition. The other direction is proved analogously. $\square$

In the next section we will define a restricted class of automata where we only allow transition-functions which are MSO-definable. In order to transfer the results of this section we need to extract the required closure properties of the set of allowed transition-functions from the above proofs.

Theorem 16.15. *Let $\mathcal{T}$ be a class of functions $f : A^* \to \mathcal{B}^+(Q \times A)$ where A and Q may be different for each $f \in \mathcal{T}$. If $\mathcal{T}$ is closed under disjunction, conjunction, dual, shift, and collection then the class of automata with transition functions $\delta : Q \times \Sigma \to \mathcal{T}$ is closed under union, complement, and projection.*

16.3 Tree-like Structures

The type of automata defined in the previous section is much too powerful. In order to prove Muchnik's Theorem we have to find a subclass which corresponds exactly to MSO on the class of trees obtained from relational structures by the operation of iteration.

Definition 16.16. Let $\mathfrak{A} = (A, R_0, \dots)$ be a τ-structure. The **iteration** of $\mathfrak{A}$ is the structure $\mathfrak{A}^* := (A^*, \mathrm{son}, \mathrm{cl}, R_0^*, \dots)$ of signature $\tau^* := \tau \cup \{\mathrm{son}, \mathrm{cl}\}$ where

$$\begin{aligned}
\mathrm{son} &:= \{\, (w, wa) \mid w \in A^*,\ a \in A \,\}, \\
\mathrm{cl} &:= \{\, waa \mid w \in A^*,\ a \in A \,\}, \\
R_i^* &:= \{\, (wa_0, \dots, wa_r) \mid w \in A^*,\ \bar{a} \in R_i \,\}.
\end{aligned}$$

For simplicity we will use a variant of monadic second-order logic where all first-order variables are eliminated. That is, formulae are constructed from atoms of the form $X \subseteq Y$ and $RX_0 \dots X_r$ by boolean operations and set quantification. Using slightly non-standard semantics we say that $R\bar{X}$ holds if $\bar{a} \in R$ for some elements $a_i \in X_i$. Note that we do not require the X_i to be singletons. Obviously, each MSO-formula can be brought into this form.

Example 16.17. The iteration $\mathfrak{G}^* := (V^*, \mathrm{son}, \mathrm{cl}, E^*)$ of a graph $\mathfrak{G} = (V, E)$ consists of all finite sequences $w \in V^*$ of vertices. We will construct an MSO-definition of those sequences which are paths in the original graph $\mathfrak{G}$. A word $w \in V^*$ is a path in $\mathfrak{G}$ if for all prefixes of the form uab with $u \in V^*$ and $a, b \in V$ there is an edge $(a, b) \in E$. The prefix relation $\preceq$ is MSO-definable being the transitive closure of the son relation. Given a prefix $y := uab$ the word $z := uaa$ can be obtained using the clone relation as follows:

$$\psi(y, z) := \exists u \big(\mathrm{son}(u, y) \wedge \mathrm{son}(u, z) \wedge \mathrm{cl}(z)\big).$$

Thus, the set of paths in $\mathfrak{G}$ can be defined by

$$\varphi(x) := \forall y \forall z (y \preceq x \wedge \psi(y, z) \rightarrow E^* yz).$$

In order to evaluate MSO-formulae over the iteration of some structure we translate them into automata where the transition function is defined by MSO-formulae. This is done in such a way that the resulting class of automata is expressively equivalent to monadic second-order logic.

Definition 16.18. Let $\mathfrak{A}$ be a structure and fix some $n \in \omega$. The function

$$\langle\!\langle \varphi \rangle\!\rangle_{\mathfrak{A}} : A^* \rightarrow \mathcal{B}^+([n] \times A)$$

induced by $\varphi(C, \bar{Q}) \in \mathrm{MSO}$ on $\mathfrak{A}$ is defined by

$$\begin{aligned}
\langle\!\langle \varphi \rangle\!\rangle_{\mathfrak{A}}(\varepsilon) &:= \bigvee \Big\{ \bigwedge \{\, (q, b) \mid b \in S_q \,\} \;\Big|\; S_0, \dots, S_{n-1} \subseteq A \text{ such that } \mathfrak{A} \models \varphi(\emptyset, \bar{S}) \Big\}, \\
\langle\!\langle \varphi \rangle\!\rangle_{\mathfrak{A}}(wa) &:= \bigvee \Big\{ \bigwedge \{\, (q, b) \mid b \in S_q \,\} \;\Big|\; S_0, \dots, S_{n-1} \subseteq A \text{ such that } \mathfrak{A} \models \varphi(\{a\}, \bar{S}) \Big\}.
\end{aligned}$$

Let $\mathcal{T}_{\mathfrak{A}}^n$ be the set of all such functions.

Definition 16.19. An **MSO-automaton** is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_{\mathrm{I}}, \Omega)$ where $Q = [n]$ for some $n \in \omega$ and $\delta : Q \times \Sigma \to \mathrm{MSO}$. $\mathcal{A}$ accepts a Σ-labelled structure $\mathfrak{A}^*$ if the automaton $\mathcal{A}_{\mathfrak{A}} := (Q, \Sigma, A, \delta_{\mathfrak{A}}, q_{\mathrm{I}}, \Omega)$ does so, where $\delta : Q \times \Sigma \to \mathcal{T}^n_{\mathfrak{A}}$ is defined by $\delta_{\mathfrak{A}}(q, c) := \langle\!\langle \delta(q, c) \rangle\!\rangle_{\mathfrak{A}}$.

In order to translate formulae into automata, the latter must be closed under all operations available in the respective logic.

Proposition 16.20. MSO*-automata are closed under boolean operations and projection.*

Proof. By Theorem 16.15 it is sufficient to show closure under or, and, dual, shift, and collection. To do so we will frequently need to convert between interpretations $I \subseteq Q \times A$ of boolean formulae $\langle\!\langle \varphi \rangle\!\rangle_{\mathfrak{A}}(w) \in \mathcal{B}^+(Q \times A)$ and sets $\bar{Q}$ such that $\mathfrak{A} \models \varphi(C, \bar{Q})$. Given $I \subseteq Q \times A$ define

$$Q_i(I) := \{\, a \in A \mid (q_i, a) \in I \,\}$$

for $i < n$, and given $Q_0, \dots, Q_{n-1} \subseteq A$ define

$$I(\bar{Q}) := \{\, (q_i, a) \mid a \in Q_i,\ i < n \,\}.$$

Note that $I(\bar{Q}(I)) = I$ and $Q_i(I(\bar{Q})) = Q_i$. Then

$$I \models \langle\!\langle \varphi \rangle\!\rangle_{\mathfrak{A}}(w) \quad \text{iff} \quad \mathfrak{A} \models \varphi(C, \bar{Q}(I))$$

and vice versa. (Here and below C denotes the set consisting of the last element of w.)

(or) For the disjunction of two MSO-definable functions we can simply take the disjunction of their definitions since

$$\begin{aligned}
&I \models \langle\!\langle \varphi_0 \rangle\!\rangle_{\mathfrak{A}}(w) \lor \langle\!\langle \varphi_1 \rangle\!\rangle_{\mathfrak{A}}(w) \\
\text{iff}\ \ &I \models \langle\!\langle \varphi_i \rangle\!\rangle_{\mathfrak{A}}(w) \text{ for some } i \\
\text{iff}\ \ &\mathfrak{A} \models \varphi_i(C, \bar{Q}(I)) \text{ for some } i \\
\text{iff}\ \ &\mathfrak{A} \models \varphi_0(C, \bar{Q}(I)) \lor \varphi_1(C, \bar{Q}(I)) \\
\text{iff}\ \ &I \models \langle\!\langle \varphi_0 \lor \varphi_1 \rangle\!\rangle_{\mathfrak{A}}(w).
\end{aligned}$$

(dual) The definition of the dual operation is slightly more involved.

$$\begin{aligned}
&I \models \overline{\langle\!\langle \varphi \rangle\!\rangle_{\mathfrak{A}}(w)} \\
\text{iff}\ \ &Q \times A \setminus I \not\models \langle\!\langle \varphi \rangle\!\rangle_{\mathfrak{A}}(w) \\
\text{iff}\ \ &J \models \langle\!\langle \varphi \rangle\!\rangle_{\mathfrak{A}}(w) \text{ implies } J \cap I \neq \emptyset \\
\text{iff}\ \ &\mathfrak{A} \models \varphi(C, \bar{P}) \text{ implies } P_i \cap Q_i(I) \neq \emptyset \text{ for some } i \\
\text{iff}\ \ &\mathfrak{A} \models \forall \bar{P}\big(\varphi(C, \bar{P}) \to \textstyle\bigvee_{i<n} P_i \cap Q_i \neq \emptyset\big)
\end{aligned}$$

(and) follows from (or) and (dual).

(shift) For a shift we simply need to renumber the states. If the pair (q_i, q_k) is encoded as number $ni + k$ we obtain

$$\varphi(C, Q_{ni+0}, \dots, Q_{ni+n-1}).$$

(collection) The collection of a formula can be defined the following way:

$$\begin{aligned}
& I \models \text{collect}\,\langle\!\langle\varphi\rangle\!\rangle_{\mathfrak{A}}(w) \\
\text{iff } & \text{there are } Q'_S \subseteq Q_S(I) \text{ such that } \bar{Q}' \text{ partitions } A \text{ and } \mathfrak{A} \models \varphi(C, \bar{P}) \\
& \text{where } a \in P_i \;:\text{iff } \; i \in S \text{ for the unique } S \subseteq [n] \text{ with } a \in Q'_S \\
\text{iff } & \text{there are } \bar{Q}' \text{ partitioning } A \text{ such that } \mathfrak{A} \models \varphi(C, \bar{P}) \text{ where} \\
& P_i := \textstyle\bigcup_{S:i \in S} Q'_S \\
\text{iff } & \mathfrak{A} \models \varphi(C, \bar{P}) \text{ for some } P_i \subseteq \textstyle\bigcup_{S:i\in S} Q_S \text{ with} \\
& P_i \cap Q_S = \emptyset \text{ for all } S \text{ with } i \notin S \\
\text{iff } & \mathfrak{A} \models \exists \bar{P}\big(\varphi(C, \bar{P}) \wedge \textstyle\bigwedge_{i<n} P_i \subseteq \bigcup_{S:i\in S} Q_S \wedge \bigwedge_{S\subseteq[n]} \bigwedge_{i\notin S} P_i \cap Q_S = \emptyset\big). \square
\end{aligned}$$

Using the preceding proposition we can state the equivalence result. We say that an automaton $\mathcal{A}$ is **equivalent** to an MSO-formula $\varphi(X_0, \dots, X_{m-1})$ if $L(\mathcal{A})$ consists of those structures whose labelling encode sets $\bar{U}$ such that $\varphi(\bar{U})$ holds. The encoding of $\bar{U}$ is the $\mathscr{P}([m])$-labelled tree T such that

$$T(w) = \{\, i \in [m] \mid w \in X_i \,\}$$

for all $w \in \{0, 1\}^*$.

Theorem 16.21. *For every formula* $\varphi \in$ MSO *there is an equivalent* MSO*-automaton and vice versa.*

Proof. ($\Rightarrow$) By induction on $\varphi(\bar{X})$ we construct an equivalent MSO-automaton $\mathcal{A} := (Q, \mathscr{P}([m]), \delta, q_0, \Omega)$. Since or corresponds to union, negation to complement, and existential quantifiers to projection, and MSO-automata are closed under all of those operations we only need to construct automata for atomic formulae.

($X_i \subseteq X_j$) We have to check for every element w of the input tree T that $i \notin T(w)$ or $j \in T(w)$. Thus, we set $Q := \{q_0\}$ with $\Omega(q_0) := 0$ and define the transition function such that

$$\delta_{\mathfrak{A}}(q_0, c, w) = \begin{cases} \bigwedge_{a\in A}(q_0, a) & \text{if } i \notin c \text{ or } j \in c, \\ \text{false} & \text{otherwise.} \end{cases}$$

for each input structure $\mathfrak{A}^*$. This can be done by setting

$$\delta(q_0, c) := \begin{cases} \forall x Q_0 x & \text{if } i \notin c \text{ or } j \in c, \\ \text{false} & \text{otherwise.} \end{cases}$$

($R^*(X_{i_1}, \dots, X_{i_k})$) Set $Q := \{q_0, \dots, q_k\}$ and $\Omega(q_i) := 1$. The automaton guesses a node in the input tree while in state q_0 and checks whether its children are in the relation R. That is,

$$\delta_{\mathfrak{A}}(q_0, c, w) = \bigvee_{a \in A} (q_0, a) \vee \bigvee \{ (q_1, a_1) \wedge \dots \wedge (q_k, a_k) \mid \bar{a} \in R^{\mathfrak{A}} \},$$

$$\delta_{\mathfrak{A}}(q_j, c, w) = \begin{cases} \text{true} & \text{if } j \in c, \\ \text{false} & \text{otherwise,} \end{cases} \qquad \text{for } 1 \leq j \leq k.$$

The corresponding MSO-definition is

$$\delta(q_0, c) := \exists x Q_0 x \vee \exists \bar{x} (R\bar{x} \wedge Q_1 x_1 \wedge \dots \wedge Q_k x_k),$$

$$\delta(q_j, c) = \begin{cases} \text{true} & \text{if } i_j \in c, \\ \text{false} & \text{otherwise,} \end{cases} \qquad \text{for } 1 \leq j \leq k.$$

($\mathrm{son}(X_i, X_j)$) Let $Q := \{q_0, q_1\}$ and $\Omega(q_i) := 1$. We guess some element $w \in X_i$ having a successor in X_j.

$$\delta_{\mathfrak{A}}(q_0, c, w) = \begin{cases} \bigvee_{a \in A} (q_0, a) & \text{if } i \notin c, \\ \bigvee_{a \in A} \big((q_0, a) \vee (q_1, a)\big) & \text{otherwise,} \end{cases}$$

$$\delta_{\mathfrak{A}}(q_1, c, w) = \begin{cases} \text{true} & \text{if } j \in c, \\ \text{false} & \text{otherwise.} \end{cases}$$

The corresponding MSO-definition is

$$\delta(q_0, c) := \begin{cases} \exists x Q_0 x & \text{if } i \notin c, \\ \exists x (Q_0 x \vee Q_1 x) & \text{otherwise,} \end{cases}$$

$$\delta(q_1, c) := \begin{cases} \text{true} & \text{if } j \in c, \\ \text{false} & \text{otherwise.} \end{cases}$$

($\mathrm{cl}(X_i)$) Let $Q := \{q_0, q_1\}$ and $\Omega(q_i) := 1$. We guess some element wa such that its successor waa is in X_i.

$$\delta_{\mathfrak{A}}(q_0, c, w) = \begin{cases} \bigvee_{a \in A} (q_0, a) & \text{if } w = \varepsilon, \\ \bigvee_{a \in A} (q_0, a) \vee (q_1, b) & \text{if } w = w'b, \end{cases}$$

$$\delta_{\mathfrak{A}}(q_1, c, w) = \begin{cases} \text{true} & \text{if } i \in c, \\ \text{false} & \text{otherwise.} \end{cases}$$

The corresponding MSO-definition is

$$\delta(q_0, c) := \exists x Q_0 x \vee \exists x (Cx \wedge Q_1 x),$$

$$\delta(q_1, c) := \begin{cases} \text{true} & \text{if } i \in c, \\ \text{false} & \text{otherwise.} \end{cases}$$

Note that this is the only place where the transition function actually depends on the current vertex.

($\Leftarrow$) Let $\mathcal{A} = (Q, \Sigma, \delta, 0, \Omega)$ be an MSO-automaton and fix an input structure $\mathfrak{A}^*$. W.l.o.g. assume that $\mathcal{A}$ is non-deterministic. $\mathfrak{A}^*$ is accepted by $\mathcal{A}$ if there is an accepting run $\varrho : A^* \to Q$ of $\mathcal{A}$ on $\mathfrak{A}^*$. This can be expressed by an MSO-formula $\varphi(\bar{X})$ in the following way: we quantify existentially over tuples $\bar{Q}$ encoding ϱ (i.e., $Q_i = \varrho^{-1}(i)$), and then check that at each position $w \in A^*$ a valid transition is used and that each path in ϱ is accepting. ☐

Before proceeding to the proof of Muchnik's Theorem let us take a look at the case of empty signature. A structure with empty signature is simply a set A. Its iteration is the tree $(A^*, \mathrm{son}, \mathrm{cl})$. The clone relation is not very useful in this case, so we drop it. Hence, the transition formulae of MSO-automata do not depend on C and the following lemma implies that we can restrict our attention to MSO-automata with monotone formulae.

Lemma 16.22. *For every* MSO*-automaton there is an equivalent one where the formulae* $\varphi(C, \bar{Q}) := \delta(q, c)$ *are monotone in* $Q_0, \dots, Q_{n-1}$.

Proof. Suppose that $\varphi(C, \bar{Q})$ is not monotone. We can replace it by

$$\varphi'(C, \bar{Q}) := \exists \bar{P} \Big(\bigwedge_{i<n} P_i \subseteq Q_i \wedge \varphi(C, \bar{P}) \Big).$$

φ' is obviously monotone. Further it is easy to see that the automaton obtained in this way is equivalent to the original one by constructing an accepting run of the former from one of the latter and vice versa. ☐

Let z be a first-order variable and $X_0, \dots, X_{n-1}$ set variables. A **type** of z over $\bar{X}$ is a formula of the form

$$\tau(z; \bar{X}) := \bigwedge_{i \in S} X_i z \wedge \bigwedge_{i \notin S} \neg X_i z$$

for some $S \subseteq [n]$. Further, define

$$\mathrm{diff}(\bar{x}) := \bigwedge_{i<k} x_i \neq x_k.$$

The next lemma provides a normalform for MSO-automata over the empty signature.

Lemma 16.23. *Every monotone* MSO*-formula* $\varphi(\bar{X})$ *over the empty signature is equivalent to a disjunction of* FO*-formulae of the form*

$$\exists \bar{y} \Big(\mathrm{diff}(\bar{y}) \wedge \bigwedge_{i<n} \vartheta_i(y_i) \wedge \forall z \Big(\mathrm{diff}(\bar{y}, z) \to \bigvee_{i<m} \vartheta'(z) \Big) \Big)$$

where the ϑ_i *and* ϑ'_i *are the positive part of some type.*

Proof. Using Ehrenfeucht-Fraïssé games it is easy to show that two structures are n-equivalent, i.e., indistinguishable by formulae of quantifier rank at most n, if, for every type $\tau(z; \bar{X})$, the number of elements satisfying τ are equal or both are greater than n. Thus, every first-order formula $\varphi(\bar{X})$ with n-quantifiers is equivalent to a disjunction of formulae of the form

$$\exists \bar{y}\Big(\mathrm{diff}(\bar{y}) \wedge \bigwedge_{i<n} \tau_i(y_i) \wedge \forall z\Big(\mathrm{diff}(\bar{y}, z) \rightarrow \bigvee_{i<m} \tau'(z)\Big)\Big)$$

each of which defines one of those n-equivalence classes where φ holds. If $\varphi(\bar{X})$ is monotone we can drop all negative atoms of the τ_i, τ'_i.

Analogously, one can show the claim also for MSO-formulae

$$Q_0 Y_0 \cdots Q_{n-1} Y_{n-1} \varphi(\bar{X}, \bar{Y})$$

with $\varphi \in$ FO, since the effect of set quantifiers amounts to splitting each type into two. □

16.4 Muchnik's Theorem

We are now ready to prove the main result of this chapter.

Theorem 16.24 (Muchnik)**.** *For every sentence $\varphi \in$ MSO one can effectively construct a sentence $\hat{\varphi} \in$ MSO such that*

$$\mathfrak{A} \models \hat{\varphi} \quad \text{iff} \quad \mathfrak{A}^* \models \varphi$$

for all structures $\mathfrak{A}$.

Corollary 16.25. *Let $\mathfrak{A}$ be a structure.* MSO *model-checking is decidable for $\mathfrak{A}$ if and only if it is so for $\mathfrak{A}^*$.*

Before giving the proof let us demonstrate how Rabin's Tree Theorem follows from Muchnik's Theorem.

Example 16.26. Consider the structure $\mathfrak{A}$ with universe $\{0, 1\}$ and two unary predicates $L = \{0\}$ and $R = \{1\}$. MSO model-checking for $\mathfrak{A}$ is decidable since $\mathfrak{A}$ is finite. According to Muchnik's Theorem, model-checking is also decidable for $\mathfrak{A}^*$. $\mathfrak{A}^*$ is similar to the binary tree. The universe is $\{0, 1\}^*$, and the relations are

$$\begin{aligned} L^* &= \{\, w0 \mid w \in \{0,1\}^* \,\}, \\ R^* &= \{\, w1 \mid w \in \{0,1\}^* \,\}, \\ \mathrm{son} &= \{\, (w, wa) \mid a \in \{0,1\},\ w \in \{0,1\}^* \,\}, \\ \mathrm{cl} &= \{\, waa \mid a \in \{0,1\},\ w \in \{0,1\}^* \,\}. \end{aligned}$$

In order to prove that model-checking for the binary tree is decidable it is sufficient to define its relations in $\mathfrak{A}^*$:

$$S_0 xy := \mathrm{son}(x, y) \wedge L^* y, \qquad S_1 xy := \mathrm{son}(x, y) \wedge R^* y.$$

Similarly the decidability of SωS can be obtained directly without the need to interpret the infinitely branching tree into the binary one.

Example 16.27. Let $\mathfrak{A} := (\omega, \leq)$. The iteration $\mathfrak{A}^*$ has universe ω^* and relations

$$\begin{aligned} \leq^* &= \{ (wa, wb) \mid a \leq b,\ w \in \omega^* \}, \\ \mathrm{son} &= \{ (w, wa) \mid a \in \omega,\ w \in \omega^* \}, \\ \mathrm{cl} &= \{ waa \mid a \in \omega,\ w \in \omega^* \}. \end{aligned}$$

The proof of Muchnik's Theorem is split into several steps. First, let $\mathcal{A} = (Q, \Sigma, \delta, q_{\mathrm{I}}, \Omega)$ be the MSO-automaton equivalent to φ. W.l.o.g. assume that $\Omega(i) = i$ for all $i \in Q = [n]$. Note that the input alphabet $\Sigma = \{\emptyset\}$ of $\mathcal{A}$ is unary since φ is a sentence. We construct a formula $\hat{\varphi}$ stating that player 0 has a winning strategy in the game $\check{\mathcal{G}}(\mathcal{A}, \mathfrak{A})$. Hence,

$$\mathfrak{A} \models \hat{\varphi} \quad \text{iff} \quad \mathfrak{A}^* \in L(\mathcal{A}) \quad \text{iff} \quad \mathfrak{A}^* \models \varphi.$$

A μ-calculus formula defining the winning set is given in Example 10.8 of Chapter 10. Translated into monadic fixed point logic it looks like

$$\mathrm{LFP}_{Z_n, x} \cdots \mathrm{GFP}_{Z_1, x} \bigvee_{i \leq n} \eta_i(x, \bar{Z})$$

with

$$\eta_i := S_i x \wedge [V_0 x \rightarrow \exists y(Exy \wedge Z_i y)] \wedge [V_1 x \rightarrow \forall y(Exy \rightarrow Z_i y)].$$

The game structure. In order to evaluate the above formula we need to embed $\check{\mathcal{G}}(\mathcal{A}, \mathfrak{A})$ in the structure $\mathfrak{A}$. First, we reduce the second component of a position (X, w) from $w \in A^*$ to a single symbol $a \in A$. Let $\mathcal{G}'(\mathcal{A}, \mathfrak{A})$ be the game obtained from $\check{\mathcal{G}}(\mathcal{A}, \mathfrak{A}^*)$ by identifying all nodes of the form (q, wa) and $(q, w'a)$, i.e.:

(a) Let $V_0 := Q \times A$. The vertices of player 0 are $V_0 \cup \{(q_0, \varepsilon)\}$, those of player 1 are $V_1 := \mathscr{P}(Q \times A)$.

(b) The initial position is (q_0, ε).

(c) Let $\langle\!\langle \delta(q, \emptyset) \rangle\!\rangle_{\mathfrak{A}}(a) = \bigvee_i \bigwedge \Phi_i$ for $a \in A \cup \{\varepsilon\}$. The node $(q, a) \in V_0$ has the successors Φ_i for all i. Nodes $\Phi \in V_1$ have their members $(q, a) \in \Phi$ as successors.

(d) A play $(q_0, a_0), \Phi_0, (q_1, a_1), \Phi_1, \ldots$ is winning if the sequence $(q_i)_{i<\omega}$ satisfies the parity condition Ω.

Lemma 16.28. *Player* 0 *has a winning strategy from the vertex* (q, wa) *in the game* $\check{\mathcal{G}}(\mathcal{A}, \mathfrak{A}^*)$ *if and only if he has one from the vertex* (q, a) *in the game* $\mathcal{G}'(\mathcal{A}, \mathfrak{A})$.

Proof. The unravelings of $\check{\mathcal{G}}(\mathcal{A}, \mathfrak{A}^*)$ and $\mathcal{G}'(\mathcal{A}, \mathfrak{A})$ from the respective vertices are isomorphic. $\square$

In the second step we encode the game $\mathcal{G}'(\mathcal{A}, \mathfrak{A})$ as the structure

$$\mathfrak{G}(\mathcal{A}, \mathfrak{A}) := \big(V_0 \cup V_1,\ E,\ \mathrm{eq}_2,\ V_0,\ V_1,\ (S_q)_{q \in Q}, R_0, \ldots\big)$$

where (V_0, V_1, E) is the graph of the game,

$$\begin{array}{ll} \mathrm{eq}_2(q,a)(q',a') & :\text{iff } a = a', \\ S_q(q',a) & :\text{iff } q = q', \\ R_i(q_0,a_0)\dots(q_r,a_r) & :\text{iff } (a_0,\dots,a_r) \in R_i^{\mathfrak{A}}. \end{array}$$

Note that these relations only contain elements of V_0. Let $\mathfrak{G}(\mathcal{A},\mathfrak{A})|_{V_0}$ denote the restriction of $\mathfrak{G}(\mathcal{A},\mathfrak{A})$ to V_0.

Finally, we can embed $\mathfrak{G}(\mathcal{A},\mathfrak{A})|_{V_0}$ in $\mathfrak{A}$ via an interpretation.

Definition 16.29. Let $\mathfrak{A} = (A, R_0, \dots, R_r)$ and $\mathfrak{B}$ be structures. An **interpretation** of $\mathfrak{A}$ in $\mathfrak{B}$ is a sequence

$$\mathcal{I} := \langle k, (\vartheta_{\bar{\imath}}^R)_{R,\bar{\imath}} \rangle$$

where, given R of arity r, the indices $\bar{\imath}$ range over $[k]^r$, such that

(i) $A \cong B \times [k]$,

(ii) $R_j \cong \{ ((a_1,i_1),\dots,(a_r,i_r)) \mid \mathfrak{B} \models \vartheta_{\bar{\imath}}^{R_j}(\bar{a}) \}$.

The use of interpretations is made possible by the following property.

Lemma 16.30. *Let $\mathcal{I}$ be an interpretation and $\varphi \in$ MSO. There is a formula $\varphi^{\mathcal{I}}$ such that*

$$\mathcal{I}(\mathfrak{A}) \models \varphi \quad \text{iff} \quad \mathfrak{A} \models \varphi^{\mathcal{I}}$$

for every structure $\mathfrak{A}$.

To construct $\varphi^{\mathcal{I}}$ one simply replaces each relation in φ by its definition.

Lemma 16.31. *There is an interpretation $\mathcal{I}$ with $\mathfrak{G}(\mathcal{A},\mathfrak{A})|_{V_0} = \mathcal{I}(\mathfrak{A})$ for all structures $\mathfrak{A}$.*

Proof. Let $\mathcal{I}$ be defined by

$$\begin{aligned} \vartheta_{ik}^{\mathrm{eq}_2}(X,Y) &:= X = Y, \\ \vartheta_k^{S_i}(X) &:= \begin{cases} \text{true} & \text{if } i = k, \\ \text{false} & \text{otherwise,} \end{cases} \\ \vartheta_{\bar{k}}^{R_i}(\bar{X}) &:= \begin{cases} R\bar{X} & \text{if } k_0 = \dots = k_r, \\ \text{false} & \text{otherwise.} \end{cases} \end{aligned}$$

□

In order to speak about all of $\mathfrak{G}(\mathcal{A},\mathfrak{A})$ in its restriction to V_0 we treat elements $\Phi \in V_1 = \mathscr{P}(V_0)$ as sets $\Phi \subseteq V_0$. All we have to do is to define the edge relation. We split E into three parts

$$E_0 \subseteq V_0 \times V_1, \quad E_1 \subseteq V_1 \times V_0, \quad \text{and} \quad E_2 \subseteq \{(q_0,\varepsilon)\} \times V_1$$

which we have to define separately by formulae $\varepsilon_0(x,Y)$, $\varepsilon_1(X,y)$, and $\varepsilon_2(Y)$.

Lemma 16.32. *There are formulae $\varepsilon_0(x, Y)$, $\varepsilon_1(X, y)$, and $\varepsilon_2(Y)$ defining the edge relations E_0, E_1 and E_2 respectively.*

Proof. Since $\big(\Phi, (q, a)\big) \in E_1$ iff $(q, a) \in \Phi$ we set

$$\varepsilon_1(Y, x) := Yx.$$

The definition of ε_0 is more involved. Let $\delta_q(C, \bar{Q}) := \langle\!\langle \delta(q, \emptyset) \rangle\!\rangle_{\mathfrak{A}}$. We have

$$((q, a), \Phi) \in E_0 \quad \text{iff} \quad \mathfrak{A} \models \delta_q(\{a\}, \bar{Q})$$

where $Q_i := \{\, b \mid (i, b) \in \Phi \,\}$. In order to evaluate δ_q we need to define $\mathfrak{A}$ inside $\mathfrak{G}(\mathcal{A}, \mathfrak{A})$. Since the latter consists of $|Q|$ copies of $\mathfrak{A}$ with universes $(S_q)_{q \in Q}$, we pick one such copy and relativise δ_q to it. For simplicity we choose S_q corresponding to the first component of (q, a).

$$((q, a), \Phi) \in E_0 \quad \text{iff} \quad \mathfrak{G}(\mathcal{A}, \mathfrak{A})|_{V_0} \models \delta_q^{S_q}\big(\{(q, a)\}, \bar{Q}'\big)$$

where $Q'_i := \{\, (q, b) \mid (i, b) \in \Phi \,\}$. This condition can be written as

$$\mathfrak{G}(\mathcal{A}, \mathfrak{A})|_{V_0} \models \exists C \exists \bar{Q} \Big(\delta_q^{S_q}(C, \bar{Q}) \wedge C = \{(q, a)\} \\ \wedge \bigwedge_{i \in Q} Q_i = \{\, (q, b) \mid (i, b) \in \Phi \,\} \Big).$$

Thus, we define

$$\varepsilon_0(x, Y) := \bigvee_{q \in Q} \big(S_q x \wedge \varepsilon_0^q(x, Y) \big)$$

where

$$\varepsilon_0^q(x, Y) := \exists C \exists \bar{Q} \Big(\delta_q^{S_q}(C, \bar{Q}) \wedge C = \{x\} \wedge \bigwedge_{i \in Q} Q_i = \{\, (q, b) \mid (i, b) \in Y \,\} \Big).$$

Obviously, $Q_i = \{\, (q, b) \mid (i, b) \in Y \,\}$ can be expressed by an MSO-formula using eq_2.

In the same way we define

$$\varepsilon_2(Y) := \exists \bar{Q} \Big(\delta_{q_0}^{S_{q_0}}(\emptyset, \bar{Q}) \wedge \bigwedge_{i \in Q} Q_i = \{\, (q_0, b) \mid (i, b) \in Y \,\} \Big).$$

□

The winning set. It remains to evaluate the formula

$$\mathrm{LFP}_{Z_1, x} \cdots \mathrm{GFP}_{Z_n, x} \bigvee_{i \le n} \eta_i(x, \bar{Z})$$

with

$$\eta_i := S_i x \;\wedge\; [V_0 x \to \exists y (Exy \wedge Z_i y)] \;\wedge\; [V_1 x \to \forall y (Exy \to Z_i y)]$$

which defines the winning set in the original game graph $\mathcal{G}'(\mathcal{A}, \mathfrak{A})$. Since in the given game the nodes of V_0 and V_1 are strictly alternating, we remain in V_0 if we take two steps each time.

$$\eta'_i := S_i x \wedge V_0 x \wedge \exists y \big(V_1 x \wedge Exy \wedge \forall z (Eyz \to Z_i z) \big)$$

It is easy to prove the following result:

Lemma 16.33. *The formulae*

$$\mathrm{GFP}_{Z_1,x} \bigvee_{i\leq n} \eta_i \quad \textit{and} \quad \mathrm{GFP}_{Z_1,x} \bigvee_{i\leq n} \eta'_i$$

define the same subset of V_0 in $\mathfrak{G}(\mathcal{A},\mathfrak{A})$ for each assignment of the free variables.

Finally, interpreting elements of V_1 by subsets of V_0, as explained above, we obtain

$$\eta''_i := S_i x \wedge V_0 x \wedge \exists Y\big(Y \subseteq V_0 \wedge \varepsilon_0(x,Y) \wedge \forall z(\varepsilon_1(Y,z) \rightarrow Z_i z)\big)$$

Again, the equivalence of η'_i and η''_i is checked easily. Thus, we can state that player 0 has a winning strategy in $\mathcal{G}'(\mathcal{A},\mathfrak{A})$ from position (q_0,ε) by

$$\hat{\varphi} := \exists Y\big[\varepsilon_2(Y) \wedge \forall x\big(\varepsilon_0(Y,x) \rightarrow \mathrm{LFP}_{Z_1,x} \cdots \mathrm{GFP}_{Z_n,x} \bigvee_{i\leq n} \eta''_i\big)\big].$$

This concludes the proof of Theorem 16.24.

We end this chapter with an application of Muchnik's Theorem to algebraic trees. Trees are represented as structures $\mathfrak{T} = (T, (E_a)_{a\in\Sigma}, (P_c)_{c\in\Gamma})$ where Σ is a finite alphabet, $T \subseteq \Sigma^*$, $P_c \subseteq T$ are unary predicates, and the edge relations are

$$E_a := \{\, (w, wa) \mid w \in T \,\}.$$

Such a tree is called **algebraic** if the set

$$\{\, wc \in \Sigma^*\Gamma \mid w \in T, w \in P_c \,\} \subseteq \Sigma^*\Gamma$$

is a deterministic context-free language.

Algebraic trees can be obtained using a variant of iterations. The **unraveling** of a graph $\mathfrak{G} = (V, (E_a)_{a\in\Sigma}, (P_c)_{c\in\Gamma})$ is the tree $\widehat{\mathfrak{G}} := (T, (\widehat{E}_a)_{a\in\Sigma}, (\widehat{P}_c)_{c\in\Gamma})$ where T consists of all paths of $\mathfrak{G}$ and the relations are defined by

$$\widehat{E}_a := \{\, wuv \in T \mid (u,v) \in E_a, w \in V^* \,\},$$
$$\widehat{P}_c := \{\, wv \in T \mid v \in P_c, w \in V^* \,\}.$$

We have already seen that the set T of paths is definable in the iteration of a graph. Obviously, the predicates $\widehat{E}_a$ and $\widehat{P}_c$ are also definable. Thus, the unraveling of a graph can be interpreted in its iteration.

The following characterisation of algebraic trees was given by Courcelle [1,**?**].

Proposition 16.34. *Every algebraic tree is the unraveling of an HR-equational graph.*

We omit the definition of HR-equational graphs. Their only property that is important in this context is that MSO-model-checking is decidable for them. Thus, we obtain the following result:

Theorem 16.35. MSO*-model-checking is decidable for algebraic trees.*

Literature

1. B. Courcelle, *The monadic second order logic of graphs, II: Infinite graphs of bounded width*, Mathematical System Theory, 21 (1989), pp. 187–222.
2. ———, *The monadic second order logic of graphs, IX: Machines and their behaviours*, Theoretical Computer Science, 151 (1995), pp. 125–162.
3. A. L. Semenov, *Decidability of monadic theories*, in Proceedings of the 11th International Symposium on Mathematical Foundations of Computer Science, MFCS '84, vol. 176 of Lecture Notes in Computer Science, Springer-Verlag, 1984, pp. 162–175.
4. S. Shelah, *The monadic second order theory of order*, Annals of Mathematics, 102 (1975), pp. 379–419.
5. J. Stupp, *The lattice-model is recursive in the original model.*, tech. rep., Institute of Mathematics, The Hebrew University, Jerusalem, Israel, 1975.
6. I. Walukiewicz, *Monadic second-order logic on tree-like structures*, Theoretical Computer Science, 275 (2002), pp. 311–346.