Monadic second order logic on tree-like structures

Igor Walukiewicz[1]
BRICS[2,3]
Department of Computer Science
University of Aarhus
Ny Munkegade
DK-8000 Aarhus C, Denmark
e-mail: igw@mimuw.edu.pl

(Extended abstract)

Abstract

An operation M^+ constructing from a given structure M a tree-like structure which domain consists of the sequences of elements of M is considered. A notion of automata running on such tree-like structures is defined. This notion is parametrised by a set of *basic formulas*. It is shown that if basic formulas satisfy some conditions then the class of languages recognised by automata is closed under disjunction, complementation and projection. For one choice of basic formulas we obtain a characterisation of MSOL over tree-like structures. This characterisation allows us to show that MSOL theory of tree-like structures is effectively reducible to that of the original structures. For a different choice of basic formulas we obtain a characterisation of MSOL on trees of arbitrary degree and the proof that it is equivalent to the first order logic extended with the unary least fixpoint operator.

1 Introduction

In [12] Shelah mentions the following construction. For a given structure $M = \langle D_M, r_1, \ldots\rangle$ one constructs the structure $M^\sharp = \langle D_M^+, son, r_1^+, \ldots\rangle$, where D_M^+ is the set of nonempty finite sequences of elements of D_M; $son(w, wd)$ holds for every $w \in D_M^+$ and $d \in D_M$; and $r^+(wd_1, \ldots, wd_k)$ iff $r(d_1, \ldots, d_k)$. Stupp is credited there for showing that if monadic second order (MSO for short) theory of

[1] This work was partially supported by Polish KBN grant No. 2 P301 009 06
[2] **B**asic **R**esearch **i**n **C**omputer **S**cience,
Centre of the Danish National Research Foundation.
[3] On leave from: Institute of Informatics, Warsaw University,
Banacha 2, 02-097 Warsaw, POLAND

M is decidable then MSO theory of $M^\sharp$ is decidable. This statement is repeated in [13].

Semenov in [11] mentions an extension of this result which he attributes to Muchnik. The stronger operation M^+ is considered. It creates a structure $M^+ = \langle D_M^+, son, cl, r_1^+, \ldots \rangle$, where the new *clone* relation is an unary relation holding for elements of the form wdd for $w \in D_M^*$ and $d \in D_M$. The statement of the theorem is also strengthened. It says that for every sentence φ of MSOL over the extended signature one can effectively find a sentence $\widehat{\varphi}$ over the original signature such that for every structure M of the original signature:

$$M \vDash \widehat{\varphi} \quad \text{iff} \quad M^+ \vDash \varphi$$

Semenov hints an idea of the proof but as far as we know the proof was never published.

One of the applications of Muchnik's theorem is proving decidability of MSO theories for big variety of structures. For example, as MSO theory of any finite structure is decidable, it immediately follows that MSO theory of the sequence or tree of such structures is decidable. In particular one obtains Rabin's Theorem [10] for M being two element structure. As a different kind of application the theorem answers a question about unravelling operation stated in [14]. There it is asked for what kind of structures M over the signature containing only unary and binary relations it is the case that the MSOL theory of an unravelling $Un(M)$ is decidable provided the MSOL theory of M is decidable. It turns out that this holds for all the structures over this kind of signatures. Weaker versions of this result were proved in [2, 3]. In [3] it was also shown how the result follows from Muchnik's theorem. For this proof the use of clone relation seams to be essential.

In the present paper we propose a notion of automata which work on the structures of the form M^+. The notion is parametrised by the set of *basic formulas*. We show that if the set of basic formulas satisfies some conditions then the class of structures accepted by the corresponding automata are closed under disjunction, complementation and projection. We apply this result to show two facts. Taking all MSOL formulas as basic formulas we can deduce Muchnik's theorem. Restricting to first order formulas we can show that MSOL over trees of arbitrary degree is equivalent to the first order logic extended with unary least fix point operator (FPL). This equivalence can be seen as a natural generalisation of the equivalence between $S2S$ and the μ-calculus over binary trees [9].

The paper is organised as follows. After the preliminary section introducing MSOL, we show a forgetful determinacy result for parity games played on (not necessary finite) graphs. In the next section we introduce a notion of automata parametrised by a set of basic formulas and state the closure properties. In the last section we prove Muchnik's theorem and show equivalence of MSOL and FPL over trees. Most of the proofs are omitted in this abstract.

Acknowledgements: I would like to thank Bruno Courcelle, David Janin, Damian Niwiński and Wieslaw Zielonka for stimulating discussions and comments. I am also grateful to the anonymous referee for his helpful remarks.

2 Preliminaries

Let $Sig = \{r_1, \dots\}$ be a signature containing only relational symbols; each relational symbol has some fixed arity. Let $Var_1 = \{x, y, z, x_1, \dots\}$ and $Var_2 = \{X, Y, Z, X_1, \dots\}$ be sets of first and second order variables respectively. The set of predicates and formulas of MSOL is defined by a mutual recursion as follows:

- every variable from Var_2 is a predicate; if $\varphi(x)$ is a formula and $F(X)$ is a predicate then: $\{x : \varphi(x)\}$, $\mathsf{LFP}\, X.F(X)$, $\mathsf{GFP}\, X.F(X)$ are predicates,
- if $P_1, \dots, P_n$ are predicates, $x \in Var_1$, $X \in Var_2$, $r \in Sig$ and α, β are formulas then: $P_1 \subseteq P_2$, $x \in P_1$, $r(P_1, \dots, P_k)$, $\neg\alpha$, $\alpha \vee \beta$, $\exists x.\alpha$, $\exists X.\alpha$ are formulas.

In a model $M = \langle D_M, r_1^M, \dots \rangle$ each relation symbol of an arity k is interpreted as a set of k-tuples of subsets of D_M. The variables from Var_1 range over elements of D^M and the variables from Var_2 over all subsets of D^M. The meaning of a predicate P in a given model M and valuation Val is the set of elements of D_M denoted $\| P \|_{Val}^M$. The meaning of the formula is either true or false. The clauses for the meanings of disjunction, negation and existential quantification are standard. In particular we have existential quantification over sets of elements of D_M.

The meaning of a predicate $\{x : \varphi(x)\}$ in a given model $\mathcal{M}$ and valuation Val is the set of elements of D_M which when taken for the meaning of x make the formula $\varphi(x)$ true. Formally we have:

$$\| \{x : \varphi(x)\} \|_{Val}^M = \{d \in D^M : M, Val[d/x] \models \varphi(x)\}$$

Observe that there can be free second order variables in $\varphi(x)$. Varying the meaning of such a free variable Z the meaning of the whole predicate $F(Z) = \{x : \varphi(x, Z)\}$ varies. Hence $F(Z)$ defines an operator form $\mathcal{P}(D^M)$ to $\mathcal{P}(D^M)$. This operator may have the least and the greatest fixpoint and these will be the meanings of $\mathsf{LFP}\, Z.F(Z)$ and $\mathsf{GFP}\, Z.F(Z)$ respectively. If this operator does not have the fixpoints then the meaning of the predicates is the empty set.

For convenience we have decided not to take a minimal syntax for MSO logic but it should be clear that for every formula in our language one can write an equivalent formula using only second order variables, relational symbols, inclusion relation, negation, disjunction and existential quantification. In other words all the other constructs are definable in this core language.

3 Game determinacy for parity conditions

Let $G = \langle V = V_I \cup V_{II}, E, \Omega : V \to Ind \rangle$ be a bipartite graph with vertices labeled by elements of Ind which is a finite subset of natural numbers.

A game from some vertex $v_1 \in V_I$ is played as follows: first player I chooses a vertex $v_2 \in V_{II}$, s.t. $E(v_1, v_2)$ then player II chooses a vertex $v_3 \in V_I$, s.t. $E(v_2, v_3)$ and so on ad infinitum unless one of the players cannot make a move.

If a player cannot make a move he looses. The result of an infinite play is an infinite sequence of vertices $v_1, v_2, v_3, \ldots$ Using function Ω we obtain a sequence of natural numbers $\Omega(v_1), \Omega(v_2), \Omega(v_3)\ldots$ Player I wins the game iff the smallest number appearing infinitely often in the sequence is even. The play from vertices of V_{II} is defined similarly but this time player II starts.

Strategy σ for player I is a function assigning to every sequence of vertices $\vec{v}$ ending in a vertex $v \in V_I$ a vertex $\sigma(\vec{v}) \in V_{II}$ such that $E(v, \sigma(\vec{v}))$. A strategy is called *memoryless* iff $\sigma(\vec{v}) = \sigma(\vec{v}')$ whenever $\vec{v}$ and $\vec{v}'$ end in the same vertex. A strategy is *winning* iff it guarantees a win for player I whenever he follows the strategy. Similarly we define a strategy for player II.

The idea to consider bounded-memory strategies originates from Büchi [1] and Gurevich and Harrington [5]. The theorem below can be deduced with some work from [5] but we prefer to give a much simpler direct proof. The discovery of parity conditions and the proof that parity games do not need memory at all was done independently by Mostowski [8] and Emerson and Jutla [4]. It seams that they were also known to Büchi in early 80-ties. The proof given here is based on [4].

Theorem 1
For every graph G as above and every vertex $v \in V_I \cup V_{II}$ one of the players has a memoryless winning strategy from v.

For the proof we will need several definitions and lemmas. Without a loss of generality we may assume that $Ind = \{1, \ldots, n\}$ for n even. The graph of the game can be presented as a relational structure:

$$\mathcal{G} = \langle V_I \cup V_{II}, E(x,y), I(x), II(x), 1(x), \ldots, n(x)\rangle$$

Relations $I(v)$ and $II(v)$ say whether x belongs to V_I or to V_{II} respectively and each relation $i(v)$ holds iff $\Omega(v) = i$. (As we assumed that relations hold between sets of elements and not between elements we may formally consider that the relations hold only for appropriate singleton sets.)

Consider the operator defined in MSO logic:

$$\begin{aligned} F_I(Z_1, \ldots, Z_n) = \big\{x : \big(I(x) \Rightarrow \exists y.(xEy \;\wedge \bigwedge_{i \in Ind} (i(y) \Rightarrow y \in Z_i))\big) \wedge \\ \big(II(x) \Rightarrow \forall y.(xEy \;\Rightarrow \bigwedge_{i \in Ind} (i(y) \Rightarrow y \in Z_i))\big)\big\} \end{aligned} \quad (1)$$

and let

$$\mathcal{S}_I = \|\, \mathsf{LFP}\, Z_1 .\, \mathsf{GFP}\, Z_2 \ldots \mathsf{LFP}\, Z_{n-1} .\, \mathsf{GFP}\, Z_n . F_I(Z_1, \ldots, Z_n)\, \|^{\mathcal{G}}$$

(LFP is used to close variables with odd indices and GFP is used for even indices).

Lemma 2 Player I has a memoryless strategy from every vertex $v \in \mathcal{S}_I$.

Proof
For every ordinal τ define $\mathsf{LFP}^\tau\, Z.F(Z)$ by the transfinite induction:

$$\begin{aligned}
\|\ \mathsf{LFP}^0\, Z.F(Z)\ \|^{\mathcal{G}}_{Val} &= \emptyset\\
\|\ \mathsf{LFP}^{\tau+1}\, Z.F(Z)\ \|^{\mathcal{G}}_{Val} &= \|\ F(Z)\ \|^{\mathcal{G}}_{Val[\|\mathsf{LFP}^\tau\, Z.F(Z)\|^{\mathcal{G}}_{Val}/Z]}\\
\|\ \mathsf{LFP}^\tau\, Z.F(Z)\ \|^{\mathcal{G}}_{Val} &= \bigcup_{\rho<\tau} \|\ \mathsf{LFP}^\rho\, Z.F(Z)\ \|^{\mathcal{G}}_{Val} \quad (\tau \text{ a limit ordinal})
\end{aligned}$$

By Knaster-Tarski theorem $\|\ \mathsf{LFP}\, Z.F(Z)\ \|^{\mathcal{G}}_{Val} = \bigcup_\tau \|\ \mathsf{LFP}^\tau\, Z.F(Z)\ \|^{\mathcal{G}}_{Val}$.

We define a notion of a *signature*, $Sig(v)$, of a vertex $v \in \mathcal{S}_I$. This is the smallest in the lexicographical ordering sequence of ordinals $(\tau_1, \ldots, \tau_n)$ such that:

$$v \in \|\ \mathsf{LFP}^{\tau_1}\, Z_1.\, \mathsf{GFP}\, Z_2 \ldots \mathsf{LFP}^{\tau_{n-1}}\, Z_{n-1}.\, \mathsf{GFP}\, Z_n.F_I(Z_1, \ldots, Z_n)\ \|^{\mathcal{G}}_{Val}$$

In this expression $\mathsf{LFP}^{\tau_i}\, Z_i$ is used instead of $\mathsf{LFP}\, Z_i$ (for $i = 1, \ldots, n$). As for even i the ordinals τ_i are not used, the definition implies that $\tau_i = 0$ for every even i. We prefer to have this redundancy rather than to calculate right indices each time.

One can show that for every $v \in \mathcal{S}_I \cap V_I$ there is a vertex v' such that $E(v, v')$, $v' \in \mathcal{S}_{II}$ and

$$\begin{array}{l}Sig(v') \text{ is not bigger than } Sig(v) \text{ on the positions up to } \Omega(v')\\ \text{and it is strictly smaller up to this position if } \Omega(v') \text{ is odd.}\end{array} \qquad (2)$$

Similarly if $v \in \mathcal{S}_I \cap V_{II}$ then for all v' such that $E(v, v')$ we have $v' \in \mathcal{S}_I$ and the condition (2) holds.

This allows us to define a strategy σ for player I: for every $v \in V_I \cap \mathcal{S}_I$ let $\sigma(v)$ be a vertex v' which has the smallest signature among all v'' such that $E(v, v'')$.

To see that this strategy is winning assume conversely that there is a play in which player I uses the strategy but player II wins. From the above follows that the play must be infinite. The result of the play is then an infinite sequence of vertices $v_1, v_2, \ldots$ such that the smallest number k which appears i.o. in the sequence $\Omega(v_1), \Omega(v_2), \ldots$ is odd. Let us take a position in the sequence after which the numbers smaller than k do not appear. By property (2) from this point signatures up to position k never increase and decrease every time we meet a vertex with index k. This is a contradiction with the fact that the ordering of signatures is well-founded. $\square$

Similarly we can define:

$$\begin{aligned}
F_{II}(Z_1, \ldots, Z_n) = \big\{x : \big(II(x) \Rightarrow \exists y.(xEy\ \wedge \bigwedge_{i \in Ind} (i(y) \Rightarrow y \in Z_i))\big) \wedge\\
\big(I(x) \Rightarrow \forall y.(xEy\ \Rightarrow \bigwedge_{i \in Ind} (i(y) \Rightarrow y \in Z_i))\big)\big\} \quad (3)
\end{aligned}$$

Observe that $F_{II}(Z_1, \ldots, Z_n)$ is equivalent to $\neg F_I(\neg Z_1, \ldots, \neg Z_n)$ thus

$$\mathcal{S}_{II} = \| \, \mathsf{GFP}\, Z_1 \ldots \mathsf{LFP}\, Z_n . F_{II}(Z_1, \ldots, Z_n) \, \|^G$$

is the complement of $\mathcal{S}_I$.

This completes the proof of Theorem 1 as one can use Lemma 2 to show that II has a winning memoryless strategy from every vertex of $\mathcal{S}_{II}$.

4 Automata on tree-like structures

Let $Sig = \{r_1, \ldots\}$ be a signature containing relational symbols only. In this section all considered formulas and structures are over this signature.

We define an operation which for a given structure $M = \langle D_M, r_1, \ldots \rangle$ constructs a tree-like structure $M^+ = \langle D_M^+, son\ cl, r_1^+, \ldots \rangle$ over the *extended signature* $Sig^+ = Sig \cup \{son, cl\}$, where D_M^+ is the set of nonempty finite sequences over D_M and the relations are defined by:

$$\begin{aligned}
son =& \{(w, wd) : w \in D_M^+, d \in D_M\} \\
cl =& \{wdd : w \in D_M^+ \cup \{\varepsilon\}, d \in D_M\} \\
r^+ =& \{(wd_1, \ldots, wd_k) : w \in D_M^+ \cup \{\varepsilon\}, (d_1, \ldots, d_k) \in r_M\} \\
& (r \in Sig,\ k\text{-array})
\end{aligned}$$

we use ε to denote an empty word.

We are going to define a notion of automaton running on structures of the form M^+. The problem we face here is how to describe a transition relation when the number of sons of the node depends on an input structure M and is not given in advance. Actually the sons of a node form a copy of M itself so there can be also some relations between them. Let us try to look at ordinary automata on binary trees and generalise from this case. Consider two sons of a given node in a binary tree as a very simple structure $\mathcal{B} = \langle \{l, r\} \rangle$. Transition function of a nondeterministic tree automaton gives for a state and a letter a set of pairs of states. Suppose for some state and letter its value is $\{(q_1, q_2), (q_3, q_4)\}$. This means that the automaton has two possibilities: it can assign q_1 to the left son and q_2 to the right, or it can assign q_3 to the left and q_4 to the right. An assignment of states to nodes can be described by a valuation of second order variables $Q_1, \ldots, Q_4$ in $\mathcal{B}$; the meaning of each variable Q_i being a set of states which have q_i assigned to them. Hence we can try to describe the set of allowed state assignments by a formula with free second order variables. For our particular example it would be $(Q_1(l) \wedge Q_2(r)) \vee (Q_3(l) \wedge Q_4(r))$. (Actually this formula allows, among others, to assign both, q_1 and q_3, to the left son and q_4 to the right son but "in the interest of automaton" is to choose a minimal assignment. We also slightly cheat here because we use elements as names of the constants.) In this approach transitions of nondeterministic automata will be always translated into disjunctions of formulas of the form $Q_i(l) \wedge Q_j(r)$ while transitions of alternating automata will require arbitrary positive boolean combinations of formulas of the form $Q_i(l)$ or $Q_i(r)$. The next generalisation step is to allow quantification which leads one to the conclusion that MSOL may be a right language for describing state assignments. It turns out that for some applications this language is to strong and it is better to parametrise the definition of automaton by the set of allowed formulas. Hence below we will postulate some properties of formulas used to describe transition function. These properties will be true for the set of all the formulas of MSOL but there are other sets of formulas satisfying these conditions. In particular we will use first order formulas of a special kind when we will be considering the case of empty signatures in Section 5.2.

We now proceed with the formal definition of the automata. Let $\{\mathrm{BF}(n)\}_{n\in\mathbb{N}}$ be a family of sets of formulas over Sig, where $\mathbb{N}$ denotes the set of natural numbers. Formulas from this family are called *basic formulas*. The only free variables in formulas from $\mathrm{BF}(n)$ can be second order variables $Q_1, \ldots, Q_n$ and an individual variable x. Hence in a model M each formula in $\mathrm{BF}(n)$ defines a relation $R \subseteq D_M \times \mathcal{P}(D_M)^n$.

We require that the family of formulas $\{\mathrm{BF}(n)\}_{n\in\mathbb{N}}$ has the following properties:

C1 $\mathrm{BF}(n)$ is closed under taking conjunctions and disjunctions.

C2 $\mathrm{BF}(n)$ is closed under choice operation: if $\varphi(x, Q_1, \ldots, Q_n) \in \mathrm{BF}(n)$ then there is a formula $\overline{\varphi}(x, Q_1, \ldots, Q_n) \in \mathrm{BF}(n)$ equivalent to:

$$\forall Z_1, \ldots, Z_n.(\varphi(x, Z_1, \ldots, Z_n) \Rightarrow Q_1 \cap Z_1 \neq \emptyset \vee \ldots \vee Q_n \cap Z_n \neq \emptyset)$$

C3 $\{\mathrm{BF}(n)\}_{n\in\mathbb{N}}$ admits power set construction: for every formula $\varphi \in BF(n)$ there is a formula $\widehat{\varphi}(x, Q_\emptyset, \ldots, Q_{\{1,\ldots,n\}}) \in \mathrm{BF}(2^n)$ such that for every model M:

- for every $d \in D_M$ and $S_1, \ldots, S_n \subseteq D_M$:

  if $M \vDash \varphi(d, S_1, \ldots, S_n)$ then $M \vDash \widehat{\varphi}(d, T_\emptyset, \ldots, T_{\{1,\ldots,n\}})$, where $T_A = \bigcap\{S_i : i \in A\} \setminus \bigcup\{S_i : i \notin A\}$, for $A \subseteq \{1, \ldots, n\}$
- for every $d \in D_M$ and $T'_\emptyset, \ldots, T'_{\{1,\ldots,n\}} \subseteq D_M$:

  if $M \vDash \widehat{\varphi}(d, T'_\emptyset, \ldots, T'_{\{1,\ldots,n\}})$ then $M \vDash \varphi(d, S_1, \ldots, S_n)$ for some $S_1, \ldots, S_n$ such that $T_A \subseteq T'_A$ for every $A \subseteq \{1, \ldots, n\}$ and T_A as defined above.

C4 $\{\mathrm{BF}\}_{n\in\mathbb{N}}$ is closed under lifting: if $\varphi \in \mathrm{BF}(n)$ then for every $i \in \mathbb{N}$ and every permutation of p of $\{1, \ldots, n+i\}$ the formula $\varphi(x, Q_{p(1)}, \ldots, Q_{p(n)})$ belongs to $\mathrm{BF}(n+i)$ (variables $Q_{p(n+1)}, \ldots, Q_{p(n+i)}$ do not appear in the formula).

We will also need a family of *initial formulas* $\{I(n)\}_{n\in\mathbb{N}}$. This family is required to satisfy the same conditions as basic formulas do and additionally the variable x is required not appear in initial formulas. This means that initial formulas from $I(n)$ describe subsets of $\mathcal{P}(D_M)^n$ and not subsets of $D_M \times \mathcal{P}(D_M)^n$ as basic formulas do.

An automaton is a tuple

$$\mathcal{A} = \langle Q, \Sigma, \iota \in I(|Q|), \delta : Q \times \Sigma \to \mathrm{BF}(|Q|), \Omega : Q \to \mathbb{N} \rangle \tag{4}$$

where $Q = \{q_1, \ldots, q_{|Q|}\}$ is a finite set of states, Σ is a finite alphabet, usually $\{0,1\}^k$.

Such an automaton is intended to run on the structures of the form M^+. As we will be interested in characterising the expressive power of formulas with free variables we account for this by introducing an alphabet and making the automaton run not on M^+ itself but on M^+ together with a *labelling function* $L : D_M^+ \to \Sigma$. This function describes valuation of free variables. We define an accepting process of an automaton in terms of games.

Definition 3 (Acceptance) Given a structure $M = \langle D_M, r_1, \ldots \rangle$ with a labelling function $L : D_M^+ \to \Sigma$, an automaton $\mathcal{A}$ as above defines a graph of the game $G(\mathcal{A}, M, L)$ in the following way:

- A *position* is an element of $D_M \times Q$. We let $V_I = (D_M \times Q) \times D_M^*$.
- A *marking* is a function $m : Q \to \mathcal{P}(D_M)$. Let $V_{II} = (Q \to \mathcal{P}(D_M)) \times D_M^*$.
- There is an edge from $(m, w) \in V_{II}$ to $((d, q), w) \in V_I$ iff $d \in m(q)$.
- There is an edge from $((d, q), w) \in V_I$ to (m, wd) iff m is a marking such that $M \vDash \varphi(d, m(q_1), \ldots, m(q_n))$, where φ is the basic formula $\delta(q, L(wd))$.
- If a vertex v belongs to V_I then it is labeled by some $((d, q), w)$ and we let $\Omega(v) = \Omega(q)$. If $v \in V_{II}$ then $\Omega(v) = \max\{\Omega(q) : q \in Q\}$.

Observe that the defined graph is not a forest but it is an acyclic graph. A vertex of such a graph will be called *initial* iff it is a vertex (m, ε) of V_{II} with ε an empty sequence and m a marking such that $M \vDash \iota(m(q_1), \ldots, m(q_n))$. We say that the structure M^+ with a labelling L is *accepted* by $\mathcal{A}$ if there is a winning strategy for player I from some initial vertex. The *language* accepted by $\mathcal{A}$ is the class of pairs (M^+, L) such that $\mathcal{A}$ accepts M^+ with the labelling L.

Example: Parity automaton on binary, Σ-valued trees [7] is a tuple:

$$\mathcal{A}_b = \langle Q_b, \Sigma_b, \iota_b \in Q_b, \delta_b : Q \times \Sigma \to \mathcal{P}(Q \times Q), \Omega_b : Q \to \mathbb{N} \rangle \tag{5}$$

where Q_b is a finite set of states, Σ_b is a finite input alphabet, i.e., the set of node labels, ι_b is the starting state, δ_b is a transition function and Ω_b is a function assigning priority to every state. A run of such an automaton is defined as usual, i.e., as for Rabin or Muller tree automata [13]. A run is accepting iff for every path, the smallest index appearing infinitely often on the path is even. Mostowski [7] showed that every Rabin automaton is equivalent to a parity automaton.

Consider the structure $M = \langle \{0,1\}, l, r \rangle$, where l holds only for $\{0\}$ and r only for $\{1\}$. We define a canonical binary tree in M^+ to be the set of all descendants of the element 0. We would like to show how to translate a parity automaton $\mathcal{A}_b$ on binary trees into an automaton $\mathcal{A}$ accepting M^+ with some labelling L iff the canonical tree, with L restricted to it, is accepted by $\mathcal{A}_b$. Automaton $\mathcal{A}$ will have the same set of states, the same alphabet and the same indexing function Ω_b. Condition ι will say that the element 0 must be labeled by the state ι_b. Hence ι is $\forall x.(l(x) \Rightarrow x \in Q_{\iota_b})$. Finally, for every $q \in Q$ and $a \in \Sigma$, transition $\delta_b(q,a) = \{(q_{l_1}, q_{r_1}), \dots, (q_{l_i}, q_{r_i})\}$ of the original automaton is translated into a formula $\delta(q,a)$:

$$\bigvee_{j=1,\dots,i} \Big(x \in l \wedge x \in Q_{l_j} \wedge \exists y. y \in r \wedge y \in Q_{r_j} \Big) \wedge \\ \bigvee_{j=1,\dots,i} \Big(x \in r \wedge x \in Q_{r_j} \wedge \exists y. y \in l \wedge y \in Q_{l_j} \Big)$$

$\square$

The main result of this section is

Theorem 4
The class of languages accepted by automata as in (4) is closed under sum, complementation and projection.

To show the closure under sum one uses the closure of basic formulas under lifting (C4) and the closure of the initial formulas under disjunction (C1). For the closure under complementation one uses additionally the closure under the choice operation (C2). The the power-set construction (C3) is used to show the closure under projection.

Finally let us comment on the form of the closure condition (C3) which is by far the least intuitive condition. One simpler choice would be to use "iff" in the first item of the condition. Unfortunately it would make formula $\widehat{\varphi}$ not monotone even for monotone φ. This in turn would make it impossible to use Theorem 4 to show the second of the applications presented below. The other simple possibility is to weaken the condition to something like:

$$M \models \widehat{\varphi}(d, T'_{\emptyset}, \dots, T'_{\{1,\dots,n\}}) \quad \text{iff} \quad M \models \varphi(d, \bigcup_{1 \in A} T'_A, \dots, \bigcup_{n \in A} T'_A)$$

Unfortunately this condition is too weak to prove closure properties of automata.

5 Applications

5.1 Muchnik's theorem

Let, as in the previous section, Sig be a signature consisting of relational symbols only and $Sig^+ = Sig \cup \{son, cl\}$ be the extended signature.

Here we will consider the case when both $\mathrm{BF}(n)$ and $I(n)$ are the set of all monadic second order formulas over Sig with free variables $x, Q_1, \ldots, Q_n$ (or just $Q_1, \ldots, Q_n$ in case of $I(n)$). It should be obvious that conditions (C1)–(C4) are satisfied.

It is quite easy to observe that the inclusion relation and all the relations from the extended signature are definable by automata with these families of basic and initial formulas. Hence from Theorem 4 we have:

Theorem 5
Automata as in (4), with $BF(n)$ and $I(n)$ being the sets of all MSOL formulas as described above, characterise the expressive power of MSOL on tree-like structures of the signature Sig^+.

Our goal is Muchnik's theorem:

Theorem 6
For every MSOL sentence φ over the signature Sig^+ one can effectively find a sentence $\widehat{\varphi}$ over the signature Sig, s.t., for every structure M over Sig:

$$M \vDash \widehat{\varphi} \quad \textit{iff} \quad M^+ \vDash \varphi$$

Given Theorem 5 for the proof of this theorem it remains to show:

Lemma 7 For every automaton $\mathcal{A}$ as in (4) over one letter alphabet there is a sentence $\widehat{\varphi}$ over Sig such that for every structure M:

$$M \vDash \widehat{\varphi} \quad \text{iff} \quad M^+ \text{ is acepted by } \mathcal{A}$$

Observe that in the case of unary alphabet the labelling function does not matter and the graph of the game described in Definition 3 is "regular" meaning that the trees issued from two vertices labeled with positions having the same first components are isomorphic. In this case we can present the graph of the game in a simpler form by letting V_I to be the set of positions and V_{II} to be the set of markings with the edges between the two sets defined appropriately. To show the lemma we use the characterisation of the winning set for player I by a fix-point closure of a MSO predicate (1).

5.2 MSOL on trees of arbitrary degree

Here we present an automata characterisation of MSOL over trees of arbitrary degree. From this characterisation we deduce that MSOL over trees is equivalent to the first order logic with the least fix point operator. This logic is the extension of the first order logic with the construction $[\mathsf{FIX}\, Z(x).\varphi(Z, x)]$ which defines a new unary predicate. We will call this logic FPL for short.

More formally FPL is obtained by adding to the first order signature a countable set of unary predicate variables (i.e. second order variables) and by extending the definition of the syntax of the first order formulas with the clause:

if $\varphi(Z,x)$ is a formula with the predicate variable Z appearing only positively in $\varphi(Z,x)$ then $[\mathsf{FIX}\, Z(x).\varphi(Z,x)]$ is a predicate.

Please note that $\varphi(Z,x)$ may itself contain fixpoint predicates. If semantics is concerned, given a model M, a valuation V_1 of first order variables and a valuation V_2 of predicate variables we have:

$$\| [\mathsf{FIX}\, Z(x).\varphi(Z,x)] \|^M_{V_1,V_2} = \bigcap\{S : \{d : M, V_1[d/x], V_2[S/Z] \vDash \varphi(Z,x)\} \subseteq S\}$$

Hence FIX construction is analogous to LFP construction defined in Section 2, the only difference being that here it is used in the context of the first order logic. Observe that FPL defined here is a fragment of the least fixpoint logic often considered in finite model theory. The difference is that we allow only unary fixpoints.

Trees of arbitrary (non zero) degrees can be presented as structures over the signature $\{son(x,y)\}$. Trees are representable in the structures of the form M^+, even for structures M over the empty signature. We say that a set $S \subseteq D^+_M$ represents a tree if the following conditions hold: (i) there is a unique element in S which is not a son of any node, (ii) if some element belongs to S then the father of this element also belongs to S, (iii) every element in S has a son which belongs to S.

Let T be a tree and κ an upper bound on the cardinalities of the degrees. Tree T is definable in M^+, where M is the structure over the empty signature with the carrier κ.

It turns out that when dealing with empty signatures it is enough to take for basic formulas only disjunctions of formulas of the form:

$$\begin{aligned}\exists y_1,\ldots,y_k.\big(&\mathrm{diff}(y_1,\ldots,y_k) \wedge \alpha(y_1,\ldots,y_k) \wedge \\ &\forall z.\mathrm{diff}(z,y_1,\ldots,y_k) \Rightarrow \beta(z)\big) \qquad (6)\end{aligned}$$

where α,β are disjunctions of conjunctions of atomic formulas of the form $y_i \in Q_j$ (negations are not allowed) and $\mathrm{diff}(y_1,\ldots,y_k)$ is a formula stating that the meanings of all the variables are different. More precisely let $\mathrm{BF}(n)$ be the set of disjunctions of formulas of the form (6) with no free first order variables and free second order variables among $Q_1,\ldots,Q_n$. In the general case we have used free first order variable in basic formulas to interpret the clone relation, as we will not need this relation here we can also omit this variable.

It can be shown that this family of basic formulas satisfies all the closure conditions (C1)–(C4). Using Theorem 4 we can obtain:

Theorem 8
Considering only models which are trees. For every MSOL formula over the signature $\{son(x,y)\}$ there is an equivalent automaton of the form

$$\langle Q, \Sigma, q_1 \in Q, \delta : Q \times \Sigma \to BF(|Q|), \Omega : Q \to \mathbb{N}\rangle$$

where $BF(|Q|)$ is the set of disjunctions of formulas of the form (6) with free variables among $Q_1 \ldots, Q_{|Q|}$.

Using similar ideas to those employed to prove Muchnik's theorem we can show that for every automaton as above there is an equivalent FPL formula. This shows:

Theorem 9
Let φ be a formula of MSOL over the signature $\{son(x,y)\}$ with free variables $Z_1,\ldots,Z_k$. There is a sentence $\widehat{\varphi}$ of FPL over the bigger signature $\{son(x,y), Z_1(x),\ldots,Z_k(x)\}$ such that for every tree T and valuation Val:

$$T, Val \vDash \varphi \quad \textit{iff} \quad \langle T, Val(Z_1),\ldots, Val(Z_{|Q|})\rangle \vDash \widehat{\varphi}$$

6 Concluding remarks

We have considered an operation M^+ of constructing tree-like structures and defined automata working on the structures of this kind. These automata are parametrised by the notion of basic formulas. We have given some conditions on basic formulas which guarantee that automata are closed under disjunction, negation and existential quantification. This parametrisation is useful if only because we were able to get two different results using different sets of basic formulas. It must be noted that other sets of basic formulas are possible. One can consider, for example, basic formulas corresponding to alternating automata on binary trees or to the μ-calculus on arbitrary trees [6]. In all these cases basic formulas will be some special first order formulas. One can also consider families of basic formulas bigger than MSOL formulas, as for example MSOL with counting modulo predicates. In this case our closure theorem gives a characterisation of MSOL with counting. We think that it is an interesting question whether one can formulate and prove some counterpart of Theorem 9 in this case.

We have not mentioned in this abstract regular model properties or complexities of the translations. These are obtained by standard arguments and will be discussed in the full version of the paper. We think that conditions (C1)–(C4) also deserve some discussion. These conditions were chosen because they have made the proofs go through but maybe there is some good reason for weakening or changing them?

References

[1] J. Büchi. State strategies for games in $F_{\sigma\delta} \cap G_{\delta\sigma}$. *Journal of Symbolic Logic*, 48:1171–1198, 1983.

[2] B. Courcelle. The monadic second-order logic on graphs IX: machines and behaviours. *Theoretical Computer Science*, 149, 1995.

[3] B. Courcelle and I. Walukiewicz. Monadic second-order logic, graphs and unfoldings of transition systems. University of Aarhus BRICS report RS-95-44. Presented at CSL'95.

[4] E. Emerson and C. Jutla. Tree automata, mu-calculus and determinacy. In *Proc. FOCS 91*, 1991.

[5] Y. Gurevich and L. Harrington. Trees, automata and games. In *14th Symp. on Theory of Computations, ACM*, pages 60–65, 1982.

[6] D. Janin and I. Walukiewicz. Automata for the μ-calculus and related results. In *MFCS'95*, volume 969 of *LNCS*, pages 552–562, 1995.

[7] A. W. Mostowski. Regular expressions for infinite trees and a standard form of automata. In A. Skowron, editor, *Fifth Symposium on Computation Theory*, volume 208 of *LNCS*, pages 157–168, 1984.

[8] A. W. Mostowski. Games with forbidden positions. Technical Report 78, University of Gdansk, 1991.

[9] D. Niwiński. Fixed points vs. infinite generation. In *Proc. 3rd. IEEE LICS*, pages 402–409, 1988.

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

[11] A. Semenov. Decidability of monadic theories. In *MFCS'84*, volume 176 of *LNCS*, pages 162–175. Springer-Verlag, 1984.

[12] S. Shelah. The monadic second order theory of order. *Annals of Mathematics*, 102:379–419, 1975.

[13] W. Thomas. Automata on infinite objects. In J. Leeuven, editor, *Handbook of Theoretical Computer Science Vol.B*, pages 995–1072. Elesvier, 1990.

[14] W. Thomas. On logics, tilings, and automata. In *ICALP '91*, volume 510 of *LNCS*, pages 441–454, 1991.