Monadic Second-order Logic of Order

Alexander Rabinovich

April 29, 2015

Objectives

(i) Explain Second-order Monadic logic - fundamental formalism.

(ii) Explain its connection with Automata theory.

Outline

(i) First-order Monadic Logic of Order

(ii) Definability and some formalizations over finite orders

(iii) Monadic Second-order Logic

(iv) Monadic second order logic vs automata over finite and ω-strings

(v) Extensions

First Order Monadic Logic of Order

Syntax of FOMLO

Signature: $\{<, =\}$ and $\{P, Q, \ldots\}$ unary predicate symbols

Variables: $V = \{x, y, z, \ldots\}$

Atomic Formulas: $\varphi ::= \quad x < y \;\mid\; x = y \;\mid\; P(x)$

Formulas: $\varphi ::= \; atomic \mid \neg\varphi_1 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \exists x \varphi_1 \mid \forall x \varphi_1$

First Order Monadic Logic of Order

Syntax of FOMLO

Signature: $\{<, =\}$ and $\{P, Q, \ldots\}$ unary predicate symbols

Variables: $V = \{x, y, z, \ldots\}$

Atomic Formulas: $\varphi ::= \quad x < y \mid x = y \mid P(x)$

Formulas: $\varphi ::= \; atomic \mid \neg\varphi_1 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \exists x \varphi_1 \mid \forall x \varphi_1$

Semantics

- Structure: $\mathcal{M} = (\mathcal{T}, <, \mathcal{I})$ - (linear order + interp. $\mathcal{I} : \{P, Q, \ldots\} \to \mathcal{P}(\mathcal{T})$)
- Truth value of $\varphi(x_1, x_2, \ldots, x_n)$ in $\mathcal{M}$ under assignment of $t_i \in \mathcal{T}$ to x_i is defined as usual (x_i free in φ).

Strings as structures for FOMLO

Strings as structures for FOMLO

Example: A string $abcaab$ can be considered as a structure for FOMLO with unary predicates P_a, P_b, P_c.
Domain: $\{1, 2, \ldots 6\}$
Interpretation of $<$ and $=$ as usual.
Interpretation of unary predicates: $P_a := \{1, 4, 5\}$, $P_b := \{2, 6\}$, $P_c := \{3\}$.

Strings as structures for FOMLO

Example: A string $abcaab$ can be considered as a structure for FOMLO with unary predicates P_a, P_b, P_c.
Domain: $\{1, 2, \dots 6\}$
Interpretation of $<$ and $=$ as usual.
Interpretation of unary predicates: $P_a := \{1, 4, 5\}$, $P_b := \{2, 6\}$, $P_c := \{3\}$.

To a string of length n a structure over $\{1, \dots n\}$ is assigned in a natural way.

Strings as structures for FOMLO

Example: A string $abcaab$ can be considered as a structure for FOMLO with unary predicates P_a, P_b, P_c.
Domain: $\{1, 2, \ldots 6\}$
Interpretation of $<$ and $=$ as usual.
Interpretation of unary predicates: $P_a := \{1, 4, 5\}$, $P_b := \{2, 6\}$, $P_c := \{3\}$.

To a string of length n a structure over $\{1, \ldots n\}$ is assigned in a natural way.

Write a sentence that holds on a string s iff s contains a letter b.

Strings as structures for FOMLO

Example: A string $abcaab$ can be considered as a structure for FOMLO with unary predicates P_a, P_b, P_c.
Domain: $\{1, 2, \ldots 6\}$
Interpretation of $<$ and $=$ as usual.
Interpretation of unary predicates: $P_a := \{1, 4, 5\}$, $P_b := \{2, 6\}$, $P_c := \{3\}$.

To a string of length n a structure over $\{1, \ldots n\}$ is assigned in a natural way.

Write a sentence that holds on a string s iff s contains a letter b.

Write a sentence that holds on a string s iff s contains a letter b and a letter a.

Strings as structures for FOMLO

Example: A string $abcaab$ can be considered as a structure for FOMLO with unary predicates P_a, P_b, P_c.
Domain: $\{1, 2, \ldots 6\}$
Interpretation of $<$ and $=$ as usual.
Interpretation of unary predicates: $P_a := \{1, 4, 5\}$, $P_b := \{2, 6\}$, $P_c := \{3\}$.

To a string of length n a structure over $\{1, \ldots n\}$ is assigned in a natural way.

Write a sentence that holds on a string s iff s contains a letter b.

Write a sentence that holds on a string s iff s contains a letter b and a letter a.

Write a sentence that holds on a string s iff s contains an occurrence of letter b before an occurrence of letter a.

Definability

Definability

Write a sentence that holds on a string s iff all occurrences of letter b in s precede all occurrences of letter a.

Definability

Write a sentence that holds on a string s iff all occurrences of letter b in s precede all occurrences of letter a.

Def. The language definable by a sentence φ is the set $\{s \mid \varphi \text{ holds in } s\}$.

Definability

Write a sentence that holds on a string s iff all occurrences of letter b in s precede all occurrences of letter a.

Def. The language definable by a sentence φ is the set $\{s \mid \varphi \text{ holds in } s\}$.

Def. A language is FOMLO definable iff it is definable by FOMLO sentence.

Definability

Write a sentence that holds on a string s iff all occurrences of letter b in s precede all occurrences of letter a.

Def. The language definable by a sentence φ is the set $\{s \mid \varphi$ holds in $s\}$.

Def. A language is FOMLO definable iff it is definable by FOMLO sentence.

Is the set of strings that starts by "b" FOMLO definable?

Definability

Write a sentence that holds on a string s iff all occurrences of letter b in s precede all occurrences of letter a.

Def. The language definable by a sentence φ is the set $\{s \mid \varphi \text{ holds in } s\}$.

Def. A language is FOMLO definable iff it is definable by FOMLO sentence.

Is the set of strings that starts by "b" FOMLO definable?

Is the set of strings that contains "bc" FOMLO definable?

Definability

Write a sentence that holds on a string s iff all occurrences of letter b in s precede all occurrences of letter a.

Def. The language definable by a sentence φ is the set $\{s \mid \varphi \text{ holds in } s\}$.

Def. A language is FOMLO definable iff it is definable by FOMLO sentence.

Is the set of strings that starts by "b" FOMLO definable?

Is the set of strings that contains "bc" FOMLO definable?

Is the set of strings that contains "bc" and does not contain "ab" FOMLO definable?

Sugaring

Add to the signature of FOMLO constant $\min$, $\max$ and the binary relation suc. In the structure for a string s the constants $\min$, $\max$ are interpreted as the minimal, maximal elements of its domain and suc is interpreted as the successor relation on its domain.

Sugaring

Add to the signature of FOMLO constant $\min$, $\max$ and the binary relation suc. In the structure for a string s the constants $\min$, $\max$ are interpreted as the minimal, maximal elements of its domain and suc is interpreted as the successor relation on its domain.

Prop. Every formula of extended language is equivalent (over strings) to an FOMLO formula. Moreover, there is a linear time equivalence preserving translation.

Sugaring

Add to the signature of FOMLO constant $\min$, $\max$ and the binary relation suc. In the structure for a string s the constants $\min$, $\max$ are interpreted as the minimal, maximal elements of its domain and suc is interpreted as the successor relation on its domain.

Prop. Every formula of extended language is equivalent (over strings) to an FOMLO formula. Moreover, there is a linear time equivalence preserving translation.

Strings as structures for FOMLO

Letters $\{a, b, c, d\}$ can be coded by two bits.
$code(a) = (0, 0)$, $code(b) = (0, 1)$, $code(c) = (1, 0)$, $code(d) = (1, 1)$.

Strings as structures for FOMLO

Letters $\{a, b, c, d\}$ can be coded by two bits.
$code(a) = (0, 0)$, $code(b) = (0, 1)$, $code(c) = (1, 0)$, $code(d) = (1, 1)$.

A string s over $\{a, b, c, d\}$ can be considered as a FOMLO structure for two unary predicates P_1 and P_2.
P_1 is interpreted as the set of position where in s letters c or d appear;
P_2 is interpreted as the set of position where in s letters b or d appear;

Strings as structures for FOMLO

Letters $\{a, b, c, d\}$ can be coded by two bits.
$code(a) = (0, 0)$, $code(b) = (0, 1)$, $code(c) = (1, 0)$, $code(d) = (1, 1)$.

A string s over $\{a, b, c, d\}$ can be considered as a FOMLO structure for two unary predicates P_1 and P_2.
P_1 is interpreted as the set of position where in s letters c or d appear;
P_2 is interpreted as the set of position where in s letters b or d appear;

Prop.(equivalence of representation.) A language over $\{a, b.c.d\}$ is FOMLO definable under the representation with predicates P_a, P_b, P_c, P_d iff it is definable under coding with two predicates P_1 and P_2.

Strings as structures for FOMLO

Letters $\{a, b, c, d\}$ can be coded by two bits.
$code(a) = (0, 0)$, $code(b) = (0, 1)$, $code(c) = (1, 0)$, $code(d) = (1, 1)$.

A string s over $\{a, b, c, d\}$ can be considered as a FOMLO structure for two unary predicates P_1 and P_2.
P_1 is interpreted as the set of position where in s letters c or d appear;
P_2 is interpreted as the set of position where in s letters b or d appear;

Prop.(equivalence of representation.) A language over $\{a, b.c.d\}$ is FOMLO definable under the representation with predicates P_a, P_b, P_c, P_d iff it is definable under coding with two predicates P_1 and P_2.

Similar coding and prop hold for more general alphabets. A string over an alphabet of size n can be considered as a structure for $1 + \log_2 n$ predicates.
.

FOMLO definability and Regular Languages

Is the language $(ab)^+ := \{(ab)^n \mid n > 0\}$ FOMLO definable?

FOMLO definability and Regular Languages

Is the language $(ab)^+ := \{(ab)^n \mid n > 0\}$ FOMLO definable?

Is the set of strings over $\{a, b\}$ of even length FOMLO definable?

FOMLO definability and Regular Languages

Is the language $(ab)^+ := \{(ab)^n \mid n > 0\}$ FOMLO definable?

Is the set of strings over $\{a, b\}$ of even length FOMLO definable?

Prop. The set of strings of even length is not FOMLO definable.

Second-Order Monadic Logic of Order

Second-Order Monadic Logic of Order

Syntax of MLO

Signature: $\{<, =\}$ and $\{P, Q, \ldots\}$ unary predicate symbols

Variables: individual variables $\{x, y, z, \ldots\}$ and monadic set variables $\{X, Y, Z, \ldots\}$.

Atomic Formulas: $\varphi ::= \quad x < y \mid x = y \mid P(x)$ and $X(y)$

Formulas: $\varphi ::= \; atomic \mid \neg\varphi_1 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \exists x \varphi_1 \mid \forall x \varphi_1$ and $\exists X \varphi_1 \mid \forall X \varphi_1$

Second-Order Monadic Logic of Order

Syntax of MLO

Signature: $\{<,=\}$ and $\{P, Q, \dots\}$ unary predicate symbols

Variables: individual variables $\{x, y, z, \dots\}$ and monadic set variables $\{X, Y, Z, \dots\}$.

Atomic Formulas: $\varphi ::= \quad x < y \mid x = y \mid P(x)$ and $X(y)$

Formulas: $\varphi ::= \ atomic \mid \neg\varphi_1 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \exists x \varphi_1 \mid \forall x \varphi_1$ and $\exists X \varphi_1 \mid \forall X \varphi_1$

Semantics

- Structure: $\mathcal{M} = (\mathcal{T}, <, \mathcal{I})$ - (linear order + interp. $\mathcal{I} : \{P, Q, \dots\} \to \mathcal{P}(\mathcal{T})$)
- Truth value of $\varphi(x_1, x_2, \dots, x_n, Y_1, \dots, Y_m)$ in $\mathcal{M}$ under assignment of $t_i \in \mathcal{T}$ to x_i and $\mathcal{T}_j \subseteq \mathcal{T}$ to Y_j is defined as usual.

MLO definability

Def. The language definable by an MLO sentence φ ...

MLO definability

Def. The language definable by an MLO sentence φ ...

Def. MLO definable language . . . .

MLO definability

Def. The language definable by an MLO sentence φ ...

Def. MLO definable language $\ldots$.

Is the set of strings over $\{a, b\}$ of even length MLO definable?

Buchi-Elgot Trakhtenbrot Theorem

Th. A language is MLO definable iff it is accepted by a finite state automaton.

Buchi-Elgot Trakhtenbrot Theorem

Th. A language is MLO definable iff it is accepted by a finite state automaton.

Prop. There is an algorithm which for every finite state automaton constructs an equivalent MLO sentence.

Buchi-Elgot Trakhtenbrot Theorem

Th. A language is MLO definable iff it is accepted by a finite state automaton.

Prop. There is an algorithm which for every finite state automaton constructs an equivalent MLO sentence.

Proof is an Easy formalization. Write a sentence which holds on a string s iff there is an accepting run of an automaton A on s.

Buchi-Elgot Trakhtenbrot Theorem

Th. A language is MLO definable iff it is accepted by a finite state automaton.

Prop. There is an algorithm which for every finite state automaton constructs an equivalent MLO sentence.

Proof is an Easy formalization. Write a sentence which holds on a string s iff there is an accepting run of an automaton A on s.

Prop. There is an algorithm which for every MLO sentence constructs an equivalent automaton.

Buchi-Elgot Trakhtenbrot Theorem

Th. A language is MLO definable iff it is accepted by a finite state automaton.

Prop. There is an algorithm which for every finite state automaton constructs an equivalent MLO sentence.

Proof is an Easy formalization. Write a sentence which holds on a string s iff there is an accepting run of an automaton A on s.

Prop. There is an algorithm which for every MLO sentence constructs an equivalent automaton.

Proof proceeds by structural induction on formulas; it uses the closure property of regular languages.

Decidability of MLO over strings

Input: an MLO sentence φ.
Question : Is φ satisfiable on a string?

Decidability of MLO over strings

Input: an MLO sentence φ.
Question : Is φ satisfiable on a string?

Thm. The satisfiability problem is decidable.

Decidability of MLO over strings

Input: an MLO sentence φ.
Question : Is φ satisfiable on a string?

Thm. The satisfiability problem is decidable.

Algorithm.

1. Construct an automaton $\mathcal{A}$ equivalent to φ.
2. Check that its language is non-empty.

ω-strings as structures

ω-strings as structures

Example: An ω-string $abcaab\cdots$ can be considered as a structure for FOMLO or MLO with unary predicates P_a, P_b, P_c.
Domain: $\{1, 2, \dots 6, \dots\}$ the set of natural numbers
Interpretation of $<$ and $=$ as usual.
Interpretation of unary predicates: $P_a := \{1, 4, 5, \dots\}$ - the set of positions with letter a, $P_b := \{2, 6, \dots\}$ - he set of positions with letter b , $P_c := \{3, \dots\}$ -the set of positions with letter a.

ω-strings as structures

Example: An ω-string $abcaab\cdots$ can be considered as a structure for FOMLO or MLO with unary predicates P_a, P_b, P_c.
Domain: $\{1, 2, \ldots 6, \ldots\}$ the set of natural numbers
Interpretation of $<$ and $=$ as usual.
Interpretation of unary predicates: $P_a := \{1, 4, 5, \ldots\}$ - the set of positions with letter a, $P_b := \{2, 6, \ldots\}$ - he set of positions with letter b , $P_c := \{3, \ldots\}$ -the set of positions with letter a.

Write a sentence that holds on an ω-string s iff s contains a letter b and a letter a.

ω-strings as structures

Example: An ω-string $abcaab\cdots$ can be considered as a structure for FOMLO or MLO with unary predicates P_a, P_b, P_c.
Domain: $\{1, 2, \ldots 6, \ldots\}$ the set of natural numbers
Interpretation of $<$ and $=$ as usual.
Interpretation of unary predicates: $P_a := \{1, 4, 5, \ldots\}$ - the set of positions with letter a, $P_b := \{2, 6, \ldots\}$ - he set of positions with letter b , $P_c := \{3, \ldots\}$ -the set of positions with letter a.

Write a sentence that holds on an ω-string s iff s contains a letter b and a letter a.

Write a sentence that holds on an ω-string s iff s contains a substring ab.

ω-strings as structures

Example: An ω-string $abcaab\cdots$ can be considered as a structure for FOMLO or MLO with unary predicates P_a, P_b, P_c.
Domain: $\{1, 2, \ldots 6, \ldots\}$ the set of natural numbers
Interpretation of $<$ and $=$ as usual.
Interpretation of unary predicates: $P_a := \{1, 4, 5, \ldots\}$ - the set of positions with letter a, $P_b := \{2, 6, \ldots\}$ - he set of positions with letter b , $P_c := \{3, \ldots\}$ -the set of positions with letter a.

Write a sentence that holds on an ω-string s iff s contains a letter b and a letter a.

Write a sentence that holds on an ω-string s iff s contains a substring ab.

Write a sentence that holds on an ω-string s iff s contains infinitely many b and infinitely many c.

Definability of ω-languages

Def. The ω-language definable by a sentence $\varphi \cdots$

Definability of ω-languages

Def. The ω-language definable by a sentence $\varphi \cdots$

Def. MLO definable language $\cdots$. FOMLO definable ω-language $\cdots$

Definability of ω-languages

Def. The ω-language definable by a sentence $\varphi \cdots$

Def. MLO definable language $\cdots$. FOMLO definable ω-language $\cdots$

Buchi Theorem

Th. An ω-language is MLO definable iff it is accepted by a finite state Buchi automaton.

Buchi Theorem

Th. An ω-language is MLO definable iff it is accepted by a finite state Buchi automaton.

Prop. There is an algorithm which for every finite state Buchi automaton constructs an equivalent MLO sentence.

Buchi Theorem

Th. An ω-language is MLO definable iff it is accepted by a finite state Buchi automaton.

Prop. There is an algorithm which for every finite state Buchi automaton constructs an equivalent MLO sentence.

Proof is an Easy formalization.

Buchi Theorem

Th. An ω-language is MLO definable iff it is accepted by a finite state Buchi automaton.

Prop. There is an algorithm which for every finite state Buchi automaton constructs an equivalent MLO sentence.

Proof is an Easy formalization.

Prop. There is an algorithm which for every MLO sentence constructs an equivalent automaton.

Buchi Theorem

Th. An ω-language is MLO definable iff it is accepted by a finite state Buchi automaton.

Prop. There is an algorithm which for every finite state Buchi automaton constructs an equivalent MLO sentence.

Proof is an Easy formalization.

Prop. There is an algorithm which for every MLO sentence constructs an equivalent automaton.

Proof proceeds by structural induction on formulas; it uses the closure property of ω-languages definable by finite state automata.

From Automata to Logic

Prop. There is an algorithm which for every finite state Buchi automaton $\mathcal{A}$ constructs an equivalent MLO sentence.

From Automata to Logic

Prop. There is an algorithm which for every finite state Buchi automaton $\mathcal{A}$ constructs an equivalent MLO sentence.

A run $\rho := q_1 a_1 q_2 a_2 \ldots$ can be considered as an ω-string over alphabet $Q_{\mathcal{A}} \times \Sigma_{\mathcal{A}}$

$$\langle q_1, a_1 \rangle \langle q_2, a_2 \rangle \ldots$$

From Automata to Logic

Prop. There is an algorithm which for every finite state Buchi automaton $\mathcal{A}$ constructs an equivalent MLO sentence.

A run $\rho := q_1 a_1 q_2 a_2 \dots$ can be considered as an ω-string over alphabet $Q_{\mathcal{A}} \times \Sigma_{\mathcal{A}}$

$$\langle q_1, a_1 \rangle \langle q_2, a_2 \rangle \dots$$

Write a sentence that expresses that this string is an accepting run of $\mathcal{A}$. If $\mathcal{A}$ has m states and is over an alphabet $\{a, b, \dots\}$ - our sentence will use predicates $Q_1, \dots Q_m$ and $P_a, P_b, \dots$

From Automata to Logic

Prop. There is an algorithm which for every finite state Buchi automaton $\mathcal{A}$ constructs an equivalent MLO sentence.

A run $\rho := q_1 a_1 q_2 a_2 \dots$ can be considered as an ω-string over alphabet $Q_\mathcal{A} \times \Sigma_\mathcal{A}$

$$\langle q_1, a_1 \rangle \langle q_2, a_2 \rangle \dots$$

Write a sentence that expresses that this string is an accepting run of $\mathcal{A}$. If $\mathcal{A}$ has m states and is over an alphabet $\{a, b, \dots\}$ - our sentence will use predicates $Q_1, \dots Q_m$ and $P_a, P_b, \dots$
It says
1. The first position is labeled by the initial state.

From Automata to Logic

Prop. There is an algorithm which for every finite state Buchi automaton $\mathcal{A}$ constructs an equivalent MLO sentence.

A run $\rho := q_1 a_1 q_2 a_2 \ldots$ can be considered as an ω-string over alphabet $Q_{\mathcal{A}} \times \Sigma_{\mathcal{A}}$

$$\langle q_1, a_1\rangle\langle q_2, a_2\rangle \ldots$$

Write a sentence that expresses that this string is an accepting run of $\mathcal{A}$. If $\mathcal{A}$ has m states and is over an alphabet $\{a, b, \ldots\}$ - our sentence will use predicates $Q_1, \ldots Q_m$ and $P_a, P_b, \ldots$
It says
1. The first position is labeled by the initial state.
2. Next state relation is observed:

$$\forall x(Q_i(x) \wedge P_a(x) \rightarrow \vee_{q_j \in \delta(q_i,a)} Q_j(x+1))$$

From Automata to Logic

Prop. There is an algorithm which for every finite state Buchi automaton $\mathcal{A}$ constructs an equivalent MLO sentence.

A run $\rho := q_1 a_1 q_2 a_2 \dots$ can be considered as an ω-string over alphabet $Q_{\mathcal{A}} \times \Sigma_{\mathcal{A}}$

$$\langle q_1, a_1 \rangle \langle q_2, a_2 \rangle \dots$$

Write a sentence that expresses that this string is an accepting run of $\mathcal{A}$. If $\mathcal{A}$ has m states and is over an alphabet $\{a, b, \dots\}$ - our sentence will use predicates $Q_1, \dots Q_m$ and $P_a, P_b, \dots$
It says
1. The first position is labeled by the initial state.
2. Next state relation is observed:

$$\forall x (Q_i(x) \land P_a(x) \rightarrow \lor_{q_j \in \delta(q_i, a)} Q_j(x+1))$$

3. Buchi-Acceptance condition: One of the states in $\mathcal{F}$ appears infinitely often.

From Automata to Logic - continued

Observe that the above sentence $ACCRUN$ is in FOMLO.

From Automata to Logic - continued

Observe that the above sentence $ACCRUN$ is in FOMLO.

Now write an MLO sentence that expresses that there is an accepting run on an ω strings.

Decidability of MLO over ω-strings

Input: an MLO sentence φ.
Question : Is φ satisfiable on an ω-string?

Decidability of MLO over ω-strings

Input: an MLO sentence φ.
Question : Is φ satisfiable on an ω-string?

Thm (Buchi)l The satisfiability problem is decidable.

Decidability of MLO over ω-strings

Input: an MLO sentence φ.
Question : Is φ satisfiable on an ω-string?

Thm (Buchi)l The satisfiability problem is decidable.

Algorithm.

1. Construct an automaton $\mathcal{A}$ equivalent to φ.
2. Check that its ω-language is non-empty.

MLO over linear orders

A structure for MLO over the rational/reals or general linear orders is defined as expected

MLO over linear orders

A structure for MLO over the rational/reals or general linear orders is defined as expected

Thm(Rabin 69) It is decidable whether an MLO sentence is satisfiable over the rationals.

MLO over linear orders

A structure for MLO over the rational/reals or general linear orders is defined as expected

Thm(Rabin 69) It is decidable whether an MLO sentence is satisfiable over the rationals.

Thm It is decidable whether an MLO sentence is satisfiable over a countable linear order.

MLO over linear orders

A structure for MLO over the rational/reals or general linear orders is defined as expected

Thm(Rabin 69) It is decidable whether an MLO sentence is satisfiable over the rationals.

Thm It is decidable whether an MLO sentence is satisfiable over a countable linear order.

Thm(Shelah 75) It is undecidable whether an MLO sentence is satisfiable over the reals.

MLO over linear orders

A structure for MLO over the rational/reals or general linear orders is defined as expected

Thm(Rabin 69) It is decidable whether an MLO sentence is satisfiable over the rationals.

Thm It is decidable whether an MLO sentence is satisfiable over a countable linear order.

Thm(Shelah 75) It is undecidable whether an MLO sentence is satisfiable over the reals.

The proof of these theorems are not based on automata.

MLO over trees

A labelled trees can be considered as a structure for monadic logic.
Domain: the set of its nodes.
Interpretation of $<$ - ancestor relation (the root is the minimal; there are incomparable nodes)
Interpretation of monadic predicates as usual.

MLO over trees

A labelled trees can be considered as a structure for monadic logic.
Domain: the set of its nodes.
Interpretation of $<$ - ancestor relation (the root is the minimal; there are incomparable nodes)
Interpretation of monadic predicates as usual.

Thm(Rabin 69) It is decidable whether an MLO sentence is satisfiable over a tree.