The Isomorphism Problem for Automatic Trees and Linear Orders

Jiamou Liu
Auckland University of Technology, New Zealand

(Joint work with D.Kuske and M.Lohrey)
Barcelona, Spain, 2011

From Computable Structures to Automatic Structures

- A structure $\mathcal{A}$ consists of a set D(universes) and relations and functions on D. We assume all structures are countably infinite and relational.
- Computable structures:
  - Universe is a computable set
  - Relations are computable.

From Computable Structures to Automatic Structures

- A structure $\mathcal{A}$ consists of a set D(universes) and relations and functions on D. We assume all structures are countably infinite and relational.
- Computable structures:
  - Universe is a computable set
  - Relations are computable.
- Automatic Structures [Khoussainov&Nerode 1995]: Replace Turing machines by finite automata.

Automata recognizing relations

Automata recognizing relations

Definition.

For words $w_1, \ldots, w_n \in \Sigma^*$, the convolution is the word $\otimes(w_1, \ldots, w_n)$ in alphabet $(\Sigma \cup \{\diamond\})^n$

$$(w'_1[1], \ldots, w'_n[1])(w'_1[2], \ldots, w'_n[2])(w'_1[3], \ldots, w'_n[3]) \cdots (w'_1[\ell], \ldots, w'_n[\ell]).$$

where $\ell = \max\{|w_i| \mid 1 \leq i \leq n\}$ and $w'_i[j] = w_i[j]$ if $j < |w_i|$ and $\diamond$ otherwise.

Automata recognizing relations

Definition.

For words $w_1, \ldots, w_n \in \Sigma^*$, the convolution is the word $\otimes(w_1, \ldots, w_n)$ in alphabet $(\Sigma \cup \{\diamond\})^n$

$$(w_1'[1], \ldots, w_n'[1])(w_1'[2], \ldots, w_n'[2])(w_1'[3], \ldots, w_n'[3]) \cdots (w_1'[\ell], \ldots, w_n'[\ell]).$$

where $\ell = \max\{|w_i| \mid 1 \leq i \leq n\}$ and $w_i'[j] = w_i[j]$ if $j < |w_i|$ and $\diamond$ otherwise.

e.g. $\otimes(010, 11100) = \binom{0}{1}\binom{1}{1}\binom{0}{1}\binom{\diamond}{1}\binom{\diamond}{0}\binom{\diamond}{0}$

Automata recognizing relations

Definition.

For words $w_1, \ldots, w_n \in \Sigma^*$, the convolution is the word $\otimes(w_1, \ldots, w_n)$ in alphabet $(\Sigma \cup \{\diamond\})^n$

$$(w'_1[1], \ldots, w'_n[1])(w'_1[2], \ldots, w'_n[2])(w'_1[3], \ldots, w'_n[3]) \cdots (w'_1[\ell], \ldots, w'_n[\ell]).$$

where $\ell = \max\{|w_i| \mid 1 \leq i \leq n\}$ and $w'_i[j] = w_i[j]$ if $j < |w_i|$ and $\diamond$ otherwise.

e.g. $\otimes(010, 11100) = \binom{0}{1}\binom{1}{1}\binom{0}{1}\binom{\diamond}{1}\binom{\diamond}{0}\binom{\diamond}{0}$

Definition.

An n-ary relation $R \subseteq (\Sigma^*)^n$ is automatic if the language $\otimes R = \{\otimes(\overline{w}) \mid \overline{w} \in R\}$ is accepted by some automaton M.

Automatic Structure

Definition.[KN]

- A structure $\mathcal{S}$ is automatic if its domain D is a regular language and each of its relation R is automatic.

Definition.[KN]

- A structure $\mathcal{S}$ is automatic if its domain D is a regular language and each of its relation R is automatic.
- If an automatic structure $\mathcal{S}'$ is isomorphic to a structure $\mathcal{S}$, then $\mathcal{S}'$ is an automatic copy of $\mathcal{S}$. In this case, $\mathcal{S}$ is automatically presentable, we simply say automatic.

Automatic Structure

Definition.[KN]

- A structure $\mathcal{S}$ is automatic if its domain D is a regular language and each of its relation R is automatic.
- If an automatic structure $\mathcal{S}'$ is isomorphic to a structure $\mathcal{S}$, then $\mathcal{S}'$ is an automatic copy of $\mathcal{S}$. In this case, $\mathcal{S}$ is automatically presentable, we simply say automatic.
- Any tuple $\mathcal{P}$ of automata that accept the domain and the relations of $\mathcal{S}$ is called an automatic presentation of $\mathcal{S}$.

Examples of automatic structures

- $(\mathbb{N}; <) \cong (0^*; \{\otimes(0^i, 0^j) \mid i < j\})$.
- $(\mathbb{N}; +)$
- $(\mathbb{Q}; \leq) \cong ((0+1)^*1; \leq_{\text{lex}})$.
- The full tree $(\mathbb{N}^*; \leq_{\text{pref}}) \cong (\{1\} \cup 1\{0,1\}^*1; \leq_{\text{pref}})$.
- Configuration graph of a Turing machine.

FO-Decidability

Theorem. [KN]

There is an algorithm that, given an automatic structure $\mathcal{S}$ and a FO-formula $\varphi(\overline{n})$, produces an automaton recognizing precisely those tuples $\overline{a} \in \mathcal{S}$ that make φ true. In particular the FO-theory of $\mathcal{S}$ is decidable.

FO-Decidability

Theorem. [KN]

There is an algorithm that, given an automatic structure $\mathcal{S}$ and a FO-formula $\varphi(\overline{n})$, produces an automaton recognizing precisely those tuples $\overline{a} \in \mathcal{S}$ that make φ true. In particular the FO-theory of $\mathcal{S}$ is decidable.

[Blumensath, Rubin, Kuske, Lohrey, etc.] FO-decidability also holds if we extend FO by $\exists^{\infty}$, $\exists^{(m,n)}$, and some restricted form of SO existential quantifier.

The Isomorphism Problem

The Isomorphism Problem

Fix a class $\mathcal{K}$ of structures. Decide if two automatic presentations recognize the same structure up to isomorphism, i.e.,

$$\{< P_1, P_2 >| \mathcal{S}(P_1), \mathcal{S}(P_2) \in \mathcal{K} \wedge \exists f : \mathcal{S}(P_1) \cong \mathcal{S}(P_2)\}$$

The Isomorphism Problem

Fix a class $\mathcal{K}$ of structures. Decide if two automatic presentations recognize the same structure up to isomorphism, i.e.,

$$\{< P_1, P_2 >| \mathcal{S}(P_1), \mathcal{S}(P_2) \in \mathcal{K} \wedge \exists f : \mathcal{S}(P_1) \cong \mathcal{S}(P_2)\}$$

- Automatic Structures Σ_1^1-complete [KNRS]
- Automatic well-orders/Boolean algebras Decidable [KNRS]
- Automatic (a) successor trees (b) undirected graphs (c) commutative monoids (d) partial orders (e) lattices of height 4 (f) unary functions. Σ_1^1-complete [Nies]
- Automatic locally finite graphs Π_3^0-complete[Rubin]

Structures with a transitive relation

Questions[KN08]

- What about for other classes of structures? e.g. equivalence structures, order trees, linear orders
- For any level of the arithmetic hierarchy, give a class of automatic structures for which isomorphism problem is complete for that level.

Structures with a transitive relation

Questions[KN08]

- What about for other classes of structures? e.g. equivalence structures, order trees, linear orders
- For any level of the arithmetic hierarchy, give a class of automatic structures for which isomorphism problem is complete for that level.

Theorem.[Kuske,Liu,Lohrey10]

The isomorphism problem is

- Π^0_1-complete for automatic equivalence structures.
- Π^0_{2n-3}-complete for automatic trees of height n ($n \geq 2$).
- computably equivalent to true arithmetic for automatic trees of finite height.
- not arithmetical for automatic linear orders.

- Hilbert's 10th problem:
  $\{p \in \mathbb{Z}[x_1, \ldots, x_k] \mid \exists x_1, \ldots, x_k \in \mathbb{N}^+ : p(x_1, \ldots, x_k) = 0\}$.

Π^0_1-Hardness for Trees of Height 2

- Hilbert's 10th problem:
$\{p \in \mathbb{Z}[x_1, \ldots, x_k] \mid \exists x_1, \ldots, x_k \in \mathbb{N}^+ : p(x_1, \ldots, x_k) = 0\}$.
- [Matiyasevich] The following problem is Π^0_1-complete:
$Prob = \{< p_1, p_2 >\mid p_i \in \mathbb{N}[x_1, \ldots, x_k], \forall x_1, \ldots, x_k \in \mathbb{N}^+ :$
$p_1(x_1, \ldots, x_k) \neq p_2(x_1, \ldots, x_k)\}$.

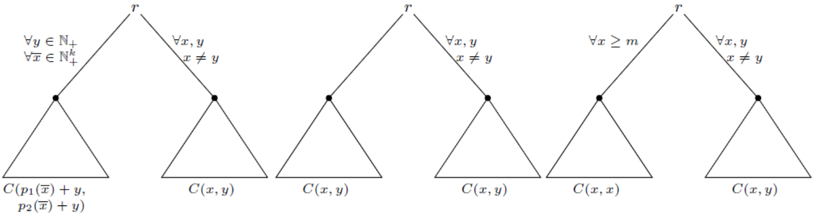
Π^0_1-Hardness for Trees of Height 2

- Hilbert's 10th problem:
  $\{p \in \mathbb{Z}[x_1, \ldots, x_k] \mid \exists x_1, \ldots, x_k \in \mathbb{N}^+ : p(x_1, \ldots, x_k) = 0\}$.
- [Matiyasevich] The following problem is Π^0_1-complete:
  $Prob = \{< p_1, p_2 >\mid p_i \in \mathbb{N}[x_1, \ldots, x_k], \forall x_1, \ldots, x_k \in \mathbb{N}^+ :$
  $p_1(x_1, \ldots, x_k) \neq p_2(x_1, \ldots, x_k)\}$.
- [Honkala 06] For any polynomial $p \in \mathbb{N}[x_1, \ldots, x_n]$, we can construct an automaton $\mathcal{A}_p$ such that on input word $\otimes(0^{x_1}, \ldots, 0^{x_n})$, $\mathcal{A}_p$ has exactly $p(x_1, \ldots, x_n)$ accepting runs.

Π_1^0-Hardness for Trees of Height 2

- Hilbert's 10th problem:
  $\{p \in \mathbb{Z}[x_1, \ldots, x_k] \mid \exists x_1, \ldots, x_k \in \mathbb{N}^+ : p(x_1, \ldots, x_k) = 0\}$.
- [Matiyasevich] The following problem is Π_1^0-complete:
  $Prob = \{< p_1, p_2 >\mid p_i \in \mathbb{N}[x_1, \ldots, x_k], \forall x_1, \ldots, x_k \in \mathbb{N}^+ :$
  $p_1(x_1, \ldots, x_k) \neq p_2(x_1, \ldots, x_k)\}$.
- [Honkala 06] For any polynomial $p \in \mathbb{N}[x_1, \ldots, x_n]$, we can construct an automaton $\mathcal{A}_p$ such that on input word $\otimes(0^{x_1}, \ldots, 0^{x_n})$, $\mathcal{A}_p$ has exactly $p(x_1, \ldots, x_n)$ accepting runs.
- We can construct automatic height-2 trees $T^2_{p_1,p_2}, T^2_{Good}, T^2_{Bad,m}$ $(m \in \mathbb{N})$ such that
  - $< p_1, p_2 > \in Prob$ if and only if $T^2_{p_1,p_2} \cong T^2_{Good}$
  - $< p_1, p_2 > \notin Prob$ if and only if $T^2_{p_1,p_2} \cong T^2_{Bad,m}$ for some m.

Fix an injective polynomial $C : \mathbb{N}^2 \to \mathbb{N}$.



The tree $T^2_{p_1,p_2}$ The tree T^2_{Good} The tree $T^2_{Bad,m}$

Trees of Height > 2

- This construction can be generalized to trees of arbitrary finite height > 2 to show the Π^0_{2n-3}-completeness mentioned above.

Trees of Height > 2

- This construction can be generalized to trees of arbitrary finite height > 2 to show the Π^0_{2n-3}-completeness mentioned above.
- Here we want to show that the isomorphism problem for all automatic order trees is Σ^1_1-complete.

- This construction can be generalized to trees of arbitrary finite height > 2 to show the Π^0_{2n-3}-completeness mentioned above.
- Here we want to show that the isomorphism problem for all automatic order trees is Σ^1_1-complete.
- For this we only need Σ^0_2-hardness for automatic trees of height 3.

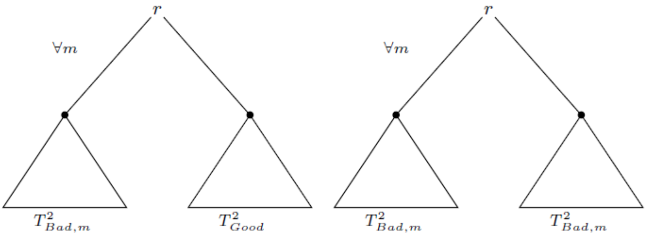
Trees of Height 3

Lemma.

There exists two height-3 trees T^3_{Good} and T^3_{Bad} ($T^3_{Good} \ncong T^3_{Bad}$) such that the following holds: For a given Σ^0_2-set $A \subseteq \{0,1\}^*1$ one can effectively construct an automatic forest F_A of height 3 such that

- The set of roots of F_A is $\{0,1\}^*1$.
- For every $w \in \{0,1\}^*1$, $F_A(w) \cong T^3_{Good}$ if $w \in A$ and $F_A(w) \cong T^3_{Bad}$ if $w \notin A$.

T^3_{Good} and T^3_{Bad}



The tree T^3_{Good} The tree T^3_{Bad}

Σ_1^1-Hardness for Automatic Order Trees

Theorem.[Kuske,Liu,Lohrey, in preparation]

The isomorphism problem for automatic order trees is Σ_1^1-complete.

Proof

Proof

- A computable tree is a prefix-closed and decidable subset $T \subseteq \mathbb{N}^*$.

Proof

- A computable tree is a prefix-closed and decidable subset $T \subseteq \mathbb{N}^*$.
- The isomorphism problem for computable trees is Σ^1_1-complete.

Proof

- A computable tree is a prefix-closed and decidable subset $T \subseteq \mathbb{N}^*$.
- The isomorphism problem for computable trees is Σ_1^1-complete.
- Start with the automatic presentation $(\{1\} \cup \{0, 1\}^* 1; \leq_{pref})$ of the full tree $(\mathbb{N}^*; \leq_{pref})$.

Σ_1^1-Hardness for Automatic Order Trees

Proof

- A computable tree is a prefix-closed and decidable subset $T \subseteq \mathbb{N}^*$.
- The isomorphism problem for computable trees is Σ_1^1-complete.
- Start with the automatic presentation $(\{1\} \cup \{0,1\}^* 1; \leq_{\text{pref}})$ of the full tree $(\mathbb{N}^*; \leq_{\text{pref}})$.
- For a computable tree T, construct an automatic order tree *$aut(T)$* as follows:
  - Append to each node $x \in \mathbb{N}^*$ of the full tree a copy of the tree T^3_{Good} if $x \in T$; and append a copy of T^3_{Bad} if $x \notin T$.

Proof

- A computable tree is a prefix-closed and decidable subset $T \subseteq \mathbb{N}^*$.
- The isomorphism problem for computable trees is Σ^1_1-complete.
- Start with the automatic presentation $(\{1\} \cup \{0,1\}^*1; \leq_{\mathrm{pref}})$ of the full tree $(\mathbb{N}^*; \leq_{\mathrm{pref}})$.
- For a computable tree T, construct an automatic order tree $aut(T)$ as follows:
  - Append to each node $x \in \mathbb{N}^*$ of the full tree a copy of the tree T^3_{Good} if $x \in T$; and append a copy of T^3_{Bad} if $x \notin T$.
- $T \cong T'$ if and only if $aut(T) \cong aut(T')$

Automatic Linear Orders

Theorem [KLL, in preparation]

The isomorphism problem for automatic linear order is Σ_1^1-complete.

Automatic Linear Orders

Theorem [KLL, in preparation]

The isomorphism problem for automatic linear order is Σ_1^1-complete.

Proof.

Automatic Linear Orders

Theorem [KLL, in preparation]

The isomorphism problem for automatic linear order is Σ^1_1-complete.

Proof.

- Recall $(\{0,1\}^*1; \leq_{lex})$ is a copy of $(\mathbb{Q}; \leq)$.

Automatic Linear Orders

Theorem [KLL, in preparation]

The isomorphism problem for automatic linear order is Σ^1_1-complete.

Proof.

- Recall $(\{0,1\}^*1; \leq_{lex})$ is a copy of $(\mathbb{Q}; \leq)$.
- From a computable linear order L, one can compute an index of a computable set $P(L) \subseteq \{0,1\}^*1$ whose complement is dense in $(\{0,1\}^*1; \leq_{lex})$ such that $L \cong L'$ iff $(\{0,1\}^*1; \leq_{lex}, P(L)) \cong (\{0,1\}^*1; \leq_{lex}, P(L'))$.

Automatic Linear Orders

Theorem [KLL, in preparation]

The isomorphism problem for automatic linear order is Σ^1_1-complete.

Proof.

- Recall $(\{0,1\}^*1; \leq_{lex})$ is a copy of $(\mathbb{Q}; \leq)$.
- From a computable linear order L, one can compute an index of a computable set $P(L) \subseteq \{0,1\}^*1$ whose complement is dense in $(\{0,1\}^*1; \leq_{lex})$ such that $L \cong L'$ iff $(\{0,1\}^*1; \leq_{lex}, P(L)) \cong (\{0,1\}^*1; \leq_{lex}, P(L'))$.
- There exist two automatic linear orders M_0 M_1 such that
  - We can construct an automatic linear order $aut(L)$ by replacing each $w \in \{0,1\}^*1$ by a copy of M_0 if $w \in P(L)$, and replacing w by a a copy of M_1 if $w \notin P(L)$.

Automatic Linear Orders

Theorem [KLL, in preparation]

The isomorphism problem for automatic linear order is Σ^1_1-complete.

Proof.

- Recall $(\{0,1\}^*1; \leq_{lex})$ is a copy of $(\mathbb{Q}; \leq)$.
- From a computable linear order L, one can compute an index of a computable set $P(L) \subseteq \{0,1\}^*1$ whose complement is dense in $(\{0,1\}^*1; \leq_{lex})$ such that $L \cong L'$ iff $(\{0,1\}^*1; \leq_{lex}, P(L)) \cong (\{0,1\}^*1; \leq_{lex}, P(L'))$.
- There exist two automatic linear orders M_0 M_1 such that
  - We can construct an automatic linear order $aut(L)$ by replacing each $w \in \{0,1\}^*1$ by a copy of M_0 if $w \in P(L)$, and replacing w by a a copy of M_1 if $w \notin P(L)$.
  - $(\{0,1\}^*; \leq_{lex}, P(L)) \cong (\{0,1\}^*; \leq_{lex}, P(L'))$ iff $aut(L) \cong aut(L')$.

Non-existence of Hyperarithmetic Isomorphisms

Corollary

There exists two isomorphic automatic order trees (linear orders) without a hyperarithmetic isomorphism.