On Synthesizing Computable Skolem functions for FO logic

Supratik Chakraborty and S. Akshay

Indian Institute of Technology Bombay

MFCS 2022, Vienna

Skolem functions

Given a FOL formula $\varphi(X, Y)$ over (inputs) X and (outputs) Y, $F(\cdot)$ is a Skolem function iff

$$\forall X \left(\exists Y \varphi(X, Y) \Leftrightarrow \varphi(X, F(X))\right)$$

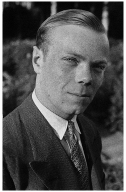
Skolem functions

Given a FOL formula $\varphi(X, Y)$ over (inputs) X and (outputs) Y, $F(\cdot)$ is a Skolem function iff

$$\forall X \left(\exists Y \varphi(X, Y) \Leftrightarrow \varphi(X, F(X))\right)$$

- Classical concept arising from quantifier elimination in FOL.
- Known to always exist! But,
  1. Is the function computable?
  2. Can we effectively compute/synthesize such a function?

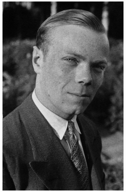
A storied history



Skolem functions play an important role in first order logic

- Getting rid of existential quantifiers
- Seminal work by Thoralf Skolem 1920s and Jacques Herbrand 1930s.
- Skolemization and “Skolem-Normal form”
- Focus on existence of form, NOT computability.

A storied history

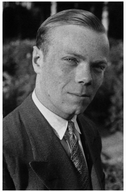


Skolem functions play an important role in first order logic

- Getting rid of existential quantifiers
- Seminal work by Thoralf Skolem 1920s and Jacques Herbrand 1930s.
- Skolemization and “Skolem-Normal form”
- Focus on existence of form, NOT computability.

We can trace this history even further back

A storied history

Skolem functions play an important role in first order logic

- Getting rid of existential quantifiers
- Seminal work by Thoralf Skolem 1920s and Jacques Herbrand 1930s.
- Skolemization and “Skolem-Normal form”
- Focus on existence of form, NOT computability.

We can trace this history even further back

- Existence and construction of Boolean unifiers
- Boole’1847, Lowenheim’1908.


Applications

Why should we be interested in synthesizability of Skolem functions?

- Heart of Automated Program Synthesis and repair.

$g(x_1, x_2) \geq x_1$ and
$g(x_1, x_2) \geq x_2$ and
$(g(x_1, x_2) == x_1$ or
$g(x_1, x_2) == x_2)$

Synthesize program for g

Why should we be interested in synthesizability of Skolem functions?

- Heart of Automated Program Synthesis and repair.

$g(x_1, x_2) \geq x_1$ and
$g(x_1, x_2) \geq x_2$ and
$(g(x_1, x_2) == x_1$ or
$g(x_1, x_2) == x_2)$

Synthesize program for g

$y_1 \geq x_1$ and
$y_1 \geq x_2$ and
$(y_1 == x_1$ or
$y_1 == x_1)$

$\forall x_1 x_2 \exists y_1 \varphi$

Golia et al, IJCAI'21

Why should we be interested in synthesizability of Skolem functions?

- Heart of Automated Program Synthesis and repair.

$g(x_1, x_2) \geq x_1$ and
$g(x_1, x_2) \geq x_2$ and
$(g(x_1, x_2) == x_1$ or
$g(x_1, x_2) == x_2)$

Synthesize program for g

$y_1 \geq x_1$ and
$y_1 \geq x_2$ and
$(y_1 == x_1$ or
$y_1 == x_1)$

$\forall x_1 x_2 \exists y_1 \varphi$

Golia et al, IJCAI'21

Prior work

- Propositional setting: Akshay et al.'17,'18,'19,'20,'21, Rabe et al. '17,'18, Golia et al.'20,'21,etc., Fried et al'16, John et al.'15, Heule et al.'14, etc.
- Beyond Propositional setting:
  - Results on specific theories: Linear rational arithmetic Kuncak et al.'10, Bit vectors Spielman et al.,Priener et al.
  - Partial approach for Quantifier Elimination Jiang'09.

Skolem functions beyond terms

- Skolem functions are often conflated with terms in the logic.

Skolem functions beyond terms

- Skolem functions are often conflated with terms in the logic.

Consider Presburger arithmetic, integers over vocabulary $\mathcal{V} = \{<, +, =, 0, 1\}$

Skolem functions beyond terms

- Skolem functions are often conflated with terms in the logic.

Consider Presburger arithmetic, integers over vocabulary $\mathcal{V} = \{<, +, =, 0, 1\}$

- Consider the formula
$$\forall y \forall z \exists x ((y > 0) \to (x > z))$$

Skolem functions beyond terms

- Skolem functions are often conflated with terms in the logic.

Consider Presburger arithmetic, integers over vocabulary $\mathcal{V} = \{<, +, =, 0, 1\}$

- Consider the formula
$$\forall y \forall z \exists x ((y > 0) \rightarrow (x > z))$$
- What is a Skolem function for x?

Skolem functions beyond terms

- Skolem functions are often conflated with terms in the logic.

Consider Presburger arithmetic, integers over vocabulary $\mathcal{V} = \{<, +, =, 0, 1\}$

- Consider the formula
$$\forall y \forall z \exists x ((y > 0) \to (x > z))$$
- What is a Skolem function for x? $F(x) = y + z$

Skolem functions beyond terms

- Skolem functions are often conflated with terms in the logic.

Consider Presburger arithmetic, integers over vocabulary $\mathcal{V} = \{<, +, =, 0, 1\}$

- Consider the formula
$$\forall y \forall z \exists x ((y > 0) \rightarrow (x > z))$$
- What is a Skolem function for x? $F(x) = y + z$, which is a term in the logic.

Skolem functions beyond terms

- Skolem functions are often conflated with terms in the logic.

Consider Presburger arithmetic, integers over vocabulary $\mathcal{V} = \{<, +, =, 0, 1\}$

- However, suppose we have

$$\forall y \forall z \exists x (((x = y) \vee (x = z)) \wedge ((x \geq y) \wedge (x \geq z)))$$

Skolem functions beyond terms

- Skolem functions are often conflated with terms in the logic.

Consider Presburger arithmetic, integers over vocabulary $\mathcal{V} = \{<, +, =, 0, 1\}$

- However, suppose we have

$$\forall y \forall z \exists x (((x = y) \vee (x = z)) \wedge ((x \geq y) \wedge (x \geq z)))$$

- No term can serve as a Skolem function for x (all terms are linear functions).

Skolem functions beyond terms

- Skolem functions are often conflated with terms in the logic.

Consider Presburger arithmetic, integers over vocabulary $\mathcal{V} = \{<, +, =, 0, 1\}$

- However, suppose we have

$$\forall y \forall z \exists x (((x = y) \vee (x = z)) \wedge ((x \geq y) \wedge (x \geq z)))$$

- No term can serve as a Skolem function for x (all terms are linear functions).
- But $F(x) = max(y, z)$ is clearly a Skolem function

Skolem functions beyond terms

- Skolem functions are often conflated with terms in the logic.

Consider Presburger arithmetic, integers over vocabulary $\mathcal{V} = \{<, +, =, 0, 1\}$

- However, suppose we have

$$\forall y \forall z \exists x (((x = y) \vee (x = z)) \wedge ((x \geq y) \wedge (x \geq z)))$$

- No term can serve as a Skolem function for x (all terms are linear functions).
- But $F(x) = max(y, z)$ is clearly a Skolem function , which can be written as a program:
  "**input(y,z); if** $\mathrm{y} \geq \mathrm{z}$ **then return** y **else return** z"

Skolem functions beyond terms

- Skolem functions are often conflated with terms in the logic.

Consider Presburger arithmetic, integers over vocabulary $\mathcal{V} = \{<, +, =, 0, 1\}$

- However, suppose we have

$$\forall y \forall z \exists x (((x = y) \vee (x = z)) \wedge ((x \geq y) \wedge (x \geq z)))$$

- No term can serve as a Skolem function for x (all terms are linear functions).
- But $F(x) = max(y, z)$ is clearly a Skolem function , which can be written as a program:

  "**input(y,z); if** $\mathrm{y} \geq \mathrm{z}$ **then return** ***y*** **else return** ***z***"
- In fact, for ANY formula in this theory, Skolem functions can be written this way!

Skolem functions beyond terms

- Skolem functions are often conflated with terms in the logic.

Consider Presburger arithmetic, integers over vocabulary $\mathcal{V} = \{<, +, =, 0, 1\}$

- However, suppose we have

$$\forall y \forall z \exists x (((x = y) \vee (x = z)) \wedge ((x \geq y) \wedge (x \geq z)))$$

- No term can serve as a Skolem function for x (all terms are linear functions).
- But $F(x) = max(y, z)$ is clearly a Skolem function , which can be written as a program:
  "**input(y,z); if** $\mathrm{y} \geq \mathrm{z}$ **then return** y **else return** z"
- In fact, for ANY formula in this theory, Skolem functions can be written this way!

The thesis of this paper

For computability/synthesis, Skolem functions should be seen as programs aka Turing machines!

Skolem functions beyond terms

- Skolem functions are often conflated with terms in the logic.

Consider Presburger arithmetic, integers over vocabulary $\mathcal{V} = \{<, +, =, 0, 1\}$

- However, suppose we have

$$\forall y \forall z \exists x (((x = y) \vee (x = z)) \wedge ((x \geq y) \wedge (x \geq z)))$$

- No term can serve as a Skolem function for x (all terms are linear functions).
- But $F(x) = max(y, z)$ is clearly a Skolem function , which can be written as a program:
  "**input(y,z); if** $\mathrm{y} \geq \mathrm{z}$ **then return** y **else return** z"
- In fact, for ANY formula in this theory, Skolem functions can be written this way!

The thesis of this paper

For computability/synthesis, Skolem functions should be seen as programs aka Turing machines!

Idea of going beyond terms not new: Skolem functions as set of conditional statements [Jiang'09]

Testing the limits

Can we always synthesize Skolem functions as Turing machines?

Can we always synthesize Skolem functions as Turing machines?

- Is there a theory where even programs fail? A theory where there is a formula for which there is no Skolem function as a program?

Testing the limits

Can we always synthesize Skolem functions as Turing machines?

- Is there a theory where even programs fail? A theory where there is a formula for which there is no Skolem function as a program?
- Unfortunately yes.

Testing the limits

Can we always synthesize Skolem functions as Turing machines?

- Is there a theory where even programs fail? A theory where there is a formula for which there is no Skolem function as a program?
- Unfortunately yes. Natural numbers over $\mathcal{V} = \{=, +, *, 0, 1\}$

Testing the limits

Can we always synthesize Skolem functions as Turing machines?

- Is there a theory where even programs fail? A theory where there is a formula for which there is no Skolem function as a program?
- Unfortunately yes. Natural numbers over $\mathcal{V} = \{=, +, *, 0, 1\}$
- Follows from the classical Matiyasevich-Robinson-Davis-Putnam (MRDP) theorem!

The problem statements

Given a vocabulary $\mathcal{V}$ and a $\mathcal{V}$-structure $\mathfrak{M}$.

The problem statements

Given a vocabulary $\mathcal{V}$ and a $\mathcal{V}$-structure $\mathfrak{M}$.

Questions of concern

1. For every $\mathcal{V}$-formula $\xi = \forall X \exists Y \varphi(X, Y)$, does there exist a Turing Machine $TM_{\xi,\mathfrak{M}}$ that serves as a Skolem function for Y in ξ, when evaluated over $\mathfrak{M}$? (SkExist)

The problem statements

Given a vocabulary $\mathcal{V}$ and a $\mathcal{V}$-structure $\mathfrak{M}$.

Questions of concern

1. For every $\mathcal{V}$-formula $\xi = \forall X \exists Y \varphi(X, Y)$, does there exist a Turing Machine $TM_{\xi,\mathfrak{M}}$ that serves as a Skolem function for Y in ξ, when evaluated over $\mathfrak{M}$? (SkExist)
2. Is there an algorithm $A_{\mathfrak{M}}$ that takes ξ as input and returns $TM_{\xi,\mathfrak{M}}$? (SkSyn)

The problem statements

Given a vocabulary $\mathcal{V}$ and a $\mathcal{V}$-structure $\mathfrak{M}$.

Questions of concern

1. For every $\mathcal{V}$-formula $\xi = \forall X \exists Y \varphi(X, Y)$, does there exist a Turing Machine $TM_{\xi,\mathfrak{M}}$ that serves as a Skolem function for Y in ξ, when evaluated over $\mathfrak{M}$? (SkExist)
2. Is there an algorithm $A_{\mathfrak{M}}$ that takes ξ as input and returns $TM_{\xi,\mathfrak{M}}$? (SkSyn)

Question 1

- Can SkExist ever return No?
- Is SkExist decidable?

The problem statements

Given a vocabulary $\mathcal{V}$ and a $\mathcal{V}$-structure $\mathfrak{M}$.

Questions of concern

1. For every $\mathcal{V}$-formula $\xi = \forall X \exists Y \varphi(X, Y)$, does there exist a Turing Machine $TM_{\xi,\mathfrak{M}}$ that serves as a Skolem function for Y in ξ, when evaluated over $\mathfrak{M}$? (SkExist)
2. Is there an algorithm $A_{\mathfrak{M}}$ that takes ξ as input and returns $TM_{\xi,\mathfrak{M}}$? (SkSyn)

Question 1

- Can SkExist ever return No?
- Is SkExist decidable?

Question 2

When SkExist returns Yes, then

- can SkSyn return No?

The problem statements

Given a vocabulary $\mathcal{V}$ and a $\mathcal{V}$-structure $\mathfrak{M}$.

Questions of concern

1. For every $\mathcal{V}$-formula $\xi = \forall X \exists Y \varphi(X, Y)$, does there exist a Turing Machine $TM_{\xi,\mathfrak{M}}$ that serves as a Skolem function for Y in ξ, when evaluated over $\mathfrak{M}$? (SkExist)
2. Is there an algorithm $A_{\mathfrak{M}}$ that takes ξ as input and returns $TM_{\xi,\mathfrak{M}}$? (SkSyn)

Question 1

- Can SkExist ever return No?
- Is SkExist decidable?

Question 2

When SkExist returns Yes, then

- can SkSyn return No?
- can we characterize precisely when SkSyn returns Yes ?

The problem statements

Given a vocabulary $\mathcal{V}$ and a $\mathcal{V}$-structure $\mathfrak{M}$.

Questions of concern

1. For every $\mathcal{V}$-formula $\xi = \forall X \exists Y \varphi(X, Y)$, does there exist a Turing Machine $TM_{\xi,\mathfrak{M}}$ that serves as a Skolem function for Y in ξ, when evaluated over $\mathfrak{M}$? (SkExist)
2. Is there an algorithm $A_{\mathfrak{M}}$ that takes ξ as input and returns $TM_{\xi,\mathfrak{M}}$? (SkSyn)

Question 1

- Can SkExist ever return No?
- Is SkExist decidable?

Question 2

When SkExist returns Yes, then

- can SkSyn return No?
- can we characterize precisely when SkSyn returns Yes ?
- Moreover, can we explicitly construct $A_{\mathfrak{M}}$?

The problem statements

Given a vocabulary $\mathcal{V}$ and a $\mathcal{V}$-structure $\mathfrak{M}$.

Questions of concern

1. For every $\mathcal{V}$-formula $\xi = \forall X \exists Y \varphi(X, Y)$, does there exist a Turing Machine $TM_{\xi,\mathfrak{M}}$ that serves as a Skolem function for Y in ξ, when evaluated over $\mathfrak{M}$? (SkExist)
2. Is there an algorithm $A_{\mathfrak{M}}$ that takes ξ as input and returns $TM_{\xi,\mathfrak{M}}$? (SkSyn)

Question 1

- Can SkExist ever return No?
- Is SkExist decidable?

Note: We assume structures to be "computable": predicates/functions are effectively computable.

Question 2

When SkExist returns Yes, then

- can SkSyn return No?
- can we characterize precisely when SkSyn returns Yes ?
- Moreover, can we explicitly construct $A_{\mathfrak{M}}$?

Negative results

1. Depending on $\mathfrak{M}$, SkExist can return Yes as well as No.

Main results

Negative results

1. Depending on $\mathfrak{M}$, SkExist can return Yes as well as No.
2. SkExist is undecidable,

Main results

Negative results

1. Depending on $\mathfrak{M}$, SkExist can return Yes as well as No.
2. SkExist is undecidable,
   1. even when $\mathcal{V}$ has a single binary predicate and a single constant.

Main results

Negative results

1. Depending on $\mathfrak{M}$, SkExist can return Yes as well as No.
2. SkExist is undecidable,
   1. even when $\mathcal{V}$ has a single binary predicate and a single constant.
   2. even for ξ in quantifier prefix classes $\exists\forall\exists$ and $\forall\exists\exists$ (but not $\exists^+\forall^*$).

Main results

Negative results

1. Depending on $\mathfrak{M}$, SkExist can return Yes as well as No.
2. SkExist is undecidable,
   1. even when $\mathcal{V}$ has a single binary predicate and a single constant.
   2. even for ξ in quantifier prefix classes $\exists\forall\exists$ and $\forall\exists\exists$ (but not $\exists^{+}\forall^{*}$).
3. There are instances where SkExist has Yes answer but not SkSyn.

Main results

Negative results

1. Depending on $\mathfrak{M}$, SkExist can return Yes as well as No.
2. SkExist is undecidable,
   1. even when $\mathcal{V}$ has a single binary predicate and a single constant.
   2. even for ξ in quantifier prefix classes $\exists\forall\exists$ and $\forall\exists\exists$ (but not $\exists^+\forall^*$).
3. There are instances where SkExist has Yes answer but not SkSyn.

But we know many theories where Skolem functions can be synthesized for all formulas. So what makes them decidable?

Main results

Negative results

1. Depending on $\mathfrak{M}$, SkExist can return Yes as well as No.
2. SkExist is undecidable,
   1. even when $\mathcal{V}$ has a single binary predicate and a single constant.
   2. even for ξ in quantifier prefix classes $\exists\forall\exists$ and $\forall\exists\exists$ (but not $\exists^+\forall^*$).
3. There are instances where SkExist has Yes answer but not SkSyn.

But we know many theories where Skolem functions can be synthesized for all formulas. So what makes them decidable?

A characterization for Synthesis

Let $\mathfrak{M}$ be a computable $\mathcal{V}$-structure for vocabulary $\mathcal{V}$.

- SkSyn has a positive answer for $\mathfrak{M}$ iff

Main results

Negative results

1. Depending on $\mathfrak{M}$, SkExist can return Yes as well as No.
2. SkExist is undecidable,
   1. even when $\mathcal{V}$ has a single binary predicate and a single constant.
   2. even for ξ in quantifier prefix classes $\exists\forall\exists$ and $\forall\exists\exists$ (but not $\exists^+\forall^*$).
3. There are instances where SkExist has Yes answer but not SkSyn.

But we know many theories where Skolem functions can be synthesized for all formulas. So what makes them decidable?

A characterization for Synthesis

Let $\mathfrak{M}$ be a computable $\mathcal{V}$-structure for vocabulary $\mathcal{V}$.

- SkSyn has a positive answer for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

So what is the elementary diagram of $\mathfrak{M}$?

A brief detour into Model theory

- Vocabulary $\mathcal{V}$
  e.g., $\{<, =, +, 0, 1\}$

A brief detour into Model theory

- Vocabulary $\mathcal{V}$
  e.g., $\{<, =, +, 0, 1\}$
- Structure $\mathfrak{M}$
  Universe $\mathbb{Z}$
  $<$: $(0,1), (-1,0), (5,7), \ldots,$
  $=$: $(0,0), \ldots$
  $+ : (0,1) \to 1, (-3,2) \to -1, \ldots$
  $0 : 0$, $1 : 1$

- Vocabulary $\mathcal{V}$
  e.g., $\{<, =, +, 0, 1\}$
- Structure $\mathfrak{M}$
  Universe $\mathbb{Z}$
  $<: (0,1), (-1,0), (5,7), \ldots,$
  $=: (0,0), \ldots$
  $+ : (0,1) \rightarrow 1, (-3,2) \rightarrow -1, \ldots$
  0 : 0, 1 : 1
- *Th*($\mathfrak{M}$) is the set of all true sentences in $\mathfrak{M}$.

A brief detour into Model theory

- Vocabulary $\mathcal{V}$
  e.g., $\{<,=,+,0,1\}$
- Structure $\mathfrak{M}$
  Universe $\mathbb{Z}$
  $<$: $(0,1),(-1,0),(5,7),\ldots,$
  $=$: $(0,0),\ldots$
  $+: (0,1) \to 1, (-3,2) \to -1, \ldots$
  $0:0,\ 1:1$
- $Th(\mathfrak{M})$ is the set of all true sentences in $\mathfrak{M}$.

- Expansion of Vocabulary $\mathcal{V}(\mathfrak{M})$
  $\{<,=,+,0,1,c_0,c_1,c_{-1},\ldots\}$

A brief detour into Model theory

- Vocabulary $\mathcal{V}$

  e.g., $\{<, =, +, 0, 1\}$
- Structure $\mathfrak{M}$

  Universe $\mathbb{Z}$

  $<$: $(0,1), (-1,0), (5,7), \ldots,$

  $=$: $(0,0), \ldots$

  $+ : (0,1) \rightarrow 1, (-3,2) \rightarrow -1, \ldots$

  $0 : 0,\ 1 : 1$
- $Th(\mathfrak{M})$ is the set of all true sentences in $\mathfrak{M}$.

- Expansion of Vocabulary $\mathcal{V}(\mathfrak{M})$

  $\{<, =, +, 0, 1, c_0, c_1, c_{-1}, \ldots\}$
- Expansion of Structure $\mathfrak{M}_{exp}$

  Universe $\mathbb{Z}$

  $<$: $(0,1), (-1,0), (5,7), \ldots,$

  $=$: $(0,0), \ldots$

  $+ : (0,1) \rightarrow 1, (-3,2) \rightarrow -1, \ldots$

  $0 : 0,\ 1 : 1$

  $c_0 : 0,\ c_1 : 1, \ldots, c_{-1} : -1, \ldots$

A brief detour into Model theory

- Vocabulary $\mathcal{V}$
  e.g., $\{<, =, +, 0, 1\}$
- Structure $\mathfrak{M}$
  Universe $\mathbb{Z}$
  $<: (0,1), (-1,0), (5,7), \ldots,$
  $=: (0,0), \ldots$
  $+ : (0,1) \to 1, (-3,2) \to -1, \ldots$
  $0 : 0,\ 1 : 1$
- $Th(\mathfrak{M})$ is the set of all true sentences in $\mathfrak{M}$.

- Expansion of Vocabulary $\mathcal{V}(\mathfrak{M})$
  $\{<, =, +, 0, 1, c_0, c_1, c_{-1}, \ldots\}$
- Expansion of Structure $\mathfrak{M}_{exp}$
  Universe $\mathbb{Z}$
  $<: (0,1), (-1,0), (5,7), \ldots,$
  $=: (0,0), \ldots$
  $+ : (0,1) \to 1, (-3,2) \to -1, \ldots$
  $0 : 0,\ 1 : 1$
  $c_0 : 0,\ c_1 : 1, \ldots, c_{-1} : -1, \ldots$
- $Th(\mathfrak{M}_{\text{exp}})$ is the set of all true sentences in $\mathfrak{M}_{exp}$.

A brief detour into Model theory

- Vocabulary $\mathcal{V}$
  e.g., $\{<, =, +, 0, 1\}$
- Structure $\mathfrak{M}$

  Universe $\mathbb{Z}$
  $<$: $(0,1), (-1,0), (5,7), \ldots,$
  $=$: $(0,0), \ldots$
  $+ : (0,1) \to 1, (-3,2) \to -1, \ldots$
  $0 : 0,\ 1 : 1$

- *Th*$(\mathfrak{M})$ is the set of all true sentences in $\mathfrak{M}$.

- Expansion of Vocabulary $\mathcal{V}(\mathfrak{M})$
  $\{<, =, +, 0, 1, c_0, c_1, c_{-1}, \ldots\}$
- Expansion of Structure $\mathfrak{M}_{exp}$

  Universe $\mathbb{Z}$
  $<$: $(0,1), (-1,0), (5,7), \ldots,$
  $=$: $(0,0), \ldots$
  $+ : (0,1) \to 1, (-3,2) \to -1, \ldots$
  $0 : 0,\ 1 : 1$
  $c_0 : 0,\ c_1 : 1, \ldots, c_{-1} : -1, \ldots$

- *Th*$(\mathfrak{M}_{\text{exp}})$ is the set of all true sentences in $\mathfrak{M}_{exp}$.
  Also called *elementary diagram of* $\mathfrak{M}$, *ED*$(\mathfrak{M})$.

- Vocabulary $\mathcal{V}$
  e.g., $\{<,=,+,0,1\}$
- Structure $\mathfrak{M}$
  Universe $\mathbb{Z}$
  $<$: $(0,1),(-1,0),(5,7),\ldots,$
  $=$: $(0,0),\ldots$
  $+ : (0,1) \to 1, (-3,2) \to -1, \ldots$
  $0 : 0,\ 1 : 1$
- $Th(\mathfrak{M})$ is the set of all true sentences in $\mathfrak{M}$.

- Expansion of Vocabulary $\mathcal{V}(\mathfrak{M})$
  $\{<,=,+,0,1,c_0,c_1,c_{-1},\ldots\}$
- Expansion of Structure $\mathfrak{M}_{exp}$
  Universe $\mathbb{Z}$
  $<$: $(0,1),(-1,0),(5,7),\ldots,$
  $=$: $(0,0),\ldots$
  $+ : (0,1) \to 1, (-3,2) \to -1, \ldots$
  $0 : 0,\ 1 : 1$
  $c_0 : 0,\ c_1 : 1, \ldots, c_{-1} : -1, \ldots$
- $Th(\mathfrak{M}_{exp})$ is the set of all true sentences in $\mathfrak{M}_{exp}$.
  Also called *elementary diagram of $\mathfrak{M}$, $ED(\mathfrak{M})$*.

Elementary diagram is said to be decidable if given any sentence φ in $\mathcal{V}(\mathfrak{M})$, we can algorithmically decide if $\varphi \in ED(\mathfrak{M})$.

A brief detour into Model theory

- Vocabulary $\mathcal{V}$
  e.g., $\{<, =, +, 0, 1\}$
- Structure $\mathfrak{M}$
  Universe $\mathbb{Z}$
  $<$: $(0,1), (-1,0), (5,7), \ldots,$
  $=$: $(0,0), \ldots$
  $+ : (0,1) \to 1, (-3,2) \to -1, \ldots$
  $0 : 0,\ 1 : 1$
- *Th*$(\mathfrak{M})$ is the set of all true sentences in $\mathfrak{M}$.

- Expansion of Vocabulary $\mathcal{V}(\mathfrak{M})$
  $\{<, =, +, 0, 1, c_0, c_1, c_{-1}, \ldots\}$
- Expansion of Structure $\mathfrak{M}_{exp}$
  Universe $\mathbb{Z}$
  $<$: $(0,1), (-1,0), (5,7), \ldots,$
  $=$: $(0,0), \ldots$
  $+ : (0,1) \to 1, (-3,2) \to -1, \ldots$
  $0 : 0,\ 1 : 1$
  $c_0 : 0,\ c_1 : 1, \ldots, c_{-1} : -1, \ldots$
- *Th*$(\mathfrak{M}_{\text{exp}})$ is the set of all true sentences in $\mathfrak{M}_{exp}$.
  Also called *elementary diagram of* $\mathfrak{M}$, *ED*$(\mathfrak{M})$.

Elementary diagram is said to be decidable if given any sentence φ in $\mathcal{V}(\mathfrak{M})$, we can algorithmically decide if $\varphi \in ED(\mathfrak{M})$.

This is the necessary and sufficient condition for synthesis!

Consequences and more!

Theorem

SkSyn has a positive answer for $\mathfrak{M}$

iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

Consequences and more!

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ and we can effectively synthesize Skolem functions as halting Turing machines for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

Consequences and more!

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ and we can effectively synthesize Skolem functions as halting Turing machines for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

Consequences

1. SkSyn has a negative answer for $(\mathbb{N}, <, =, +, *, 0, 1)$.
2. SkSyn has a positive answer and we can effectively synthesize Skolem functions for
   1. Presburger arithmetic
   2. Linear rational arithmetic
   3. Real algebraic numbers
   4. Dense linear orders without endpoints

Consequences and more!

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ and we can effectively synthesize Skolem functions as halting Turing machines for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

Consequences

1. SkSyn has a negative answer for $(\mathbb{N}, <, =, +, *, 0, 1)$.
2. SkSyn has a positive answer and we can effectively synthesize Skolem functions for
   1. Presburger arithmetic
   2. Linear rational arithmetic
   3. Real algebraic numbers
   4. Dense linear orders without endpoints

- In each case, we reduce to decidability of underlying theory *Th*$(\mathfrak{M})$.

Consequences and more!

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ and we can effectively synthesize Skolem functions as halting Turing machines for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

Consequences

1. SkSyn has a negative answer for $(\mathbb{N}, <, =, +, *, 0, 1)$.
2. SkSyn has a positive answer and we can effectively synthesize Skolem functions for
    1. Presburger arithmetic
    2. Linear rational arithmetic
    3. Real algebraic numbers
    4. Dense linear orders without endpoints

- In each case, we reduce to decidability of underlying theory $Th(\mathfrak{M})$.
- Not true in general! There exist $\mathfrak{M}$ s.t. $Th(\mathfrak{M})$ is decidable but $ED(\mathfrak{M})$ is not (see paper).

Consequences and more!

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ and we can effectively synthesize Skolem functions as halting Turing machines for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

Consequences

1. SkSyn has a negative answer for $(\mathbb{N}, <, =, +, *, 0, 1)$.
2. SkSyn has a positive answer and we can effectively synthesize Skolem functions for
   1. Presburger arithmetic
   2. Linear rational arithmetic
   3. Real algebraic numbers
   4. Dense linear orders without endpoints

- In each case, we reduce to decidability of underlying theory $Th(\mathfrak{M})$.
- Not true in general! There exist $\mathfrak{M}$ s.t. $Th(\mathfrak{M})$ is decidable but $ED(\mathfrak{M})$ is not (see paper).

Complexity

- Lower bound follows from complexity of deciding theory.

Consequences and more!

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ and we can effectively synthesize Skolem functions as halting Turing machines for $\mathfrak{M}$ iff the “elementary diagram” of $\mathfrak{M}$ is decidable.

Consequences

1. SkSyn has a negative answer for $(\mathbb{N}, <, =, +, *, 0, 1)$.
2. SkSyn has a positive answer and we can effectively synthesize Skolem functions for
    1. Presburger arithmetic
    2. Linear rational arithmetic
    3. Real algebraic numbers
    4. Dense linear orders without endpoints

- In each case, we reduce to decidability of underlying theory $Th(\mathfrak{M})$.
- Not true in general! There exist $\mathfrak{M}$ s.t. $Th(\mathfrak{M})$ is decidable but $ED(\mathfrak{M})$ is not (see paper).

Complexity

- Lower bound follows from complexity of deciding theory.
- If theory admits effective constraint solving, then can give upper bounds! (see paper)

Conclusion - A beginning

A framework to the study algorithmic computation of Skolem functions.

- Skolem functions as Turing machines/programs.
- A characterization resulting in strong positive and negative results.

Conclusion - A beginning

A framework to the study algorithmic computation of Skolem functions.

- Skolem functions as Turing machines/programs.
- A characterization resulting in strong positive and negative results.

Other results in paper

- e.g., what happens if you fix the formula and vary the structure?

Conclusion - A beginning

A framework to the study algorithmic computation of Skolem functions.

- Skolem functions as Turing machines/programs.
- A characterization resulting in strong positive and negative results.

Other results in paper

- e.g., what happens if you fix the formula and vary the structure?

The future

- Synthesizing succinct Skolem functions and algorithms with better complexity.
- Characterization of when terms are sufficient.
- Implementation for certain theories?

Conclusion - A beginning

A framework to the study algorithmic computation of Skolem functions.

- Skolem functions as Turing machines/programs.
- A characterization resulting in strong positive and negative results.

Other results in paper

- e.g., what happens if you fix the formula and vary the structure?

The future

- Synthesizing succinct Skolem functions and algorithms with better complexity.
- Characterization of when terms are sufficient.
- Implementation for certain theories? Work in progress!

Thank you!

A Short Sketch of proof

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

A Short Sketch of proof

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

($\Longrightarrow$) Program/TM for Skolem function for Y in $\forall X \exists Y \varphi(X, Y)$ is as follows:

A Short Sketch of proof

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

($\Longrightarrow$) Program/TM for Skolem function for Y in $\forall X \exists Y \varphi(X, Y)$ is as follows:

1. Given value of X, say σ, construct $\Psi_\sigma = \exists Y \varphi(\sigma, Y)$

A Short Sketch of proof

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

($\Longrightarrow$) Program/TM for Skolem function for Y in $\forall X \exists Y \varphi(X, Y)$ is as follows:

1. Given value of X, say σ, construct $\Psi_\sigma = \exists Y \varphi(\sigma, Y)$
2. Use dec proc for $ED(\mathfrak{M})$ on this.

A Short Sketch of proof

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

($\Longrightarrow$) Program/TM for Skolem function for Y in $\forall X \exists Y \varphi(X, Y)$ is as follows:

1. Given value of X, say σ, construct $\Psi_\sigma = \exists Y \varphi(\sigma, Y)$
2. Use dec proc for $ED(\mathfrak{M})$ on this.
   - if false, output arbitrary value.

A Short Sketch of proof

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

($\Longrightarrow$) Program/TM for Skolem function for Y in $\forall X \exists Y \varphi(X, Y)$ is as follows:

1. Given value of X, say σ, construct $\Psi_\sigma = \exists Y \varphi(\sigma, Y)$
2. Use dec proc for $ED(\mathfrak{M})$ on this.
   - if false, output arbitrary value.
   - if true, for each elt ρ in $dom(Y)$, do

A Short Sketch of proof

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

($\Longrightarrow$) Program/TM for Skolem function for Y in $\forall X \exists Y \varphi(X, Y)$ is as follows:

1. Given value of X, say σ, construct $\Psi_\sigma = \exists Y \varphi(\sigma, Y)$
2. Use dec proc for $ED(\mathfrak{M})$ on this.
   - if false, output arbitrary value.
   - if true, for each elt ρ in $dom(Y)$, do
     1. construct $\varphi(\sigma, \rho)$

A Short Sketch of proof

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the “elementary diagram” of $\mathfrak{M}$ is decidable.

($\Longrightarrow$) Program/TM for Skolem function for Y in $\forall X \exists Y \varphi(X, Y)$ is as follows:

1. Given value of X, say σ, construct $\Psi_\sigma = \exists Y \varphi(\sigma, Y)$
2. Use dec proc for $ED(\mathfrak{M})$ on this.
   - if false, output arbitrary value.
   - if true, for each elt ρ in $dom(Y)$, do
     1. construct $\varphi(\sigma, \rho)$
     2. apply dec proc for $ED(\mathfrak{M})$ on this.

A Short Sketch of proof

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

($\Longrightarrow$) Program/TM for Skolem function for Y in $\forall X \exists Y \varphi(X, Y)$ is as follows:

1. Given value of X, say σ, construct $\Psi_\sigma = \exists Y \varphi(\sigma, Y)$
2. Use dec proc for $ED(\mathfrak{M})$ on this.
   - if false, output arbitrary value.
   - if true, for each elt ρ in $dom(Y)$, do
     1. construct $\varphi(\sigma, \rho)$
     2. apply dec proc for $ED(\mathfrak{M})$ on this.
     3. if true, output ρ, quit loop, else goto next elt.

$\square$

A Short Sketch of proof

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

($\Longleftarrow$) Dec proc for $ED(\mathfrak{M})$ using Sk fn generator for formulas over $\mathfrak{M}$.

A Short Sketch of proof

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the “elementary diagram” of $\mathfrak{M}$ is decidable.

($\Longleftarrow$) Dec proc for $ED(\mathfrak{M})$ using Sk fn generator for formulas over $\mathfrak{M}$.

1. Given $\mathcal{V}(\mathfrak{M})$ sentence φ with constant $c \in \mathcal{V}(\mathfrak{M})$, construct $\varphi'(y)$ where c replaced by fresh var y.

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the “elementary diagram” of $\mathfrak{M}$ is decidable.

($\Longleftarrow$) Dec proc for $ED(\mathfrak{M})$ using Sk fn generator for formulas over $\mathfrak{M}$.

1. Given $\mathcal{V}(\mathfrak{M})$ sentence φ with constant $c \in \mathcal{V}(\mathfrak{M})$, construct $\varphi'(y)$ where c replaced by fresh var y.
2. For fresh var, z_1, z_2 define $\psi = \forall y \forall z_1 \forall z_2 \exists x(((x = z_1) \wedge \varphi') \vee (x = z_2) \wedge \neg\varphi')$ (note: this is a valid formula!)

A Short Sketch of proof

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

($\Longleftarrow$) Dec proc for $ED(\mathfrak{M})$ using Sk fn generator for formulas over $\mathfrak{M}$.

1. Given $\mathcal{V}(\mathfrak{M})$ sentence φ with constant $c \in \mathcal{V}(\mathfrak{M})$, construct $\varphi'(y)$ where c replaced by fresh var y.
2. For fresh var, z_1, z_2 define $\psi = \forall y \forall z_1 \forall z_2 \exists x(((x = z_1) \wedge \varphi') \vee (x = z_2) \wedge \neg\varphi')$ (note: this is a valid formula!)
3. Use Sk fn gen on ψ to synthesize Sk fn for x, $F(y, z_1, z_2)$.

A Short Sketch of proof

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the "elementary diagram" of $\mathfrak{M}$ is decidable.

($\Longleftarrow$) Dec proc for $ED(\mathfrak{M})$ using Sk fn generator for formulas over $\mathfrak{M}$.

1. Given $\mathcal{V}(\mathfrak{M})$ sentence φ with constant $c \in \mathcal{V}(\mathfrak{M})$, construct $\varphi'(y)$ where c replaced by fresh var y.
2. For fresh var, z_1, z_2 define $\psi = \forall y \forall z_1 \forall z_2 \exists x(((x = z_1) \wedge \varphi') \vee (x = z_2) \wedge \neg\varphi')$ (note: this is a valid formula!)
3. Use Sk fn gen on ψ to synthesize Sk fn for x, $F(y, z_1, z_2)$.
4. For two distinct elements $d, e \in \mathfrak{M}$, evaluate $F(c, d, e)$.

Theorem

SkSyn has a positive answer for $\mathfrak{M}$ iff the “elementary diagram” of $\mathfrak{M}$ is decidable.

($\Longleftarrow$) Dec proc for $ED(\mathfrak{M})$ using Sk fn generator for formulas over $\mathfrak{M}$.

1. Given $\mathcal{V}(\mathfrak{M})$ sentence φ with constant $c \in \mathcal{V}(\mathfrak{M})$, construct $\varphi'(y)$ where c replaced by fresh var y.
2. For fresh var, z_1, z_2 define $\psi = \forall y \forall z_1 \forall z_2 \exists x(((x = z_1) \wedge \varphi') \vee (x = z_2) \wedge \neg\varphi')$ (note: this is a valid formula!)
3. Use Sk fn gen on ψ to synthesize Sk fn for x, $F(y, z_1, z_2)$.
4. For two distinct elements $d, e \in \mathfrak{M}$, evaluate $F(c, d, e)$.
   - if $F(c, d, e) = d$, then φ is valid.
   - else $F(c, d, e) = e$ and φ is not valid.

$\square$