Martin Goldstern

Institute of Discrete Mathematics and Geometry Vienna University of Technology

May 2013 — ISMVL Toyama

Outline

Basic definitions

Complexity results

The arithmetical hierarchy

Computable structures

Proof ingredients

Summary

Outline

Basic definitions

Complexity results

The arithmetical hierarch

Computable structures

Proof ingredient

Summar

Łukasiewicz functions on [0, 1]

Definition (Fuzzy operations on [0, 1])

conjunction:
$$x \wedge y := \min(x, y)$$

disjunction:
$$x \lor y := \max(x, y)$$

negation:
$$\neg x := 1 - x$$

weak disjunction: $x+y := \min(x+y,1)$

strong conjunction:
$$x \& y := \max(x + y - 1, 0) = \neg(\neg x + \neg y)$$

implication:
$$x \rightarrow y := (\neg x) + y = \max\{z : (x \& z) \le y\}$$

Note:
$$x \lor y = (x \rightarrow y) \rightarrow y, \neg x = (x \rightarrow 0), ...$$

Definition (Fuzzy operations on [0, 1])

conjunction: $x \wedge y := \min(x, y)$

disjunction: $x \lor y := \max(x, y)$

negation: $\neg x := 1 - x$

weak disjunction: x+y := min(x + y, 1)

strong conjunction: $x \& y := \max(x + y - 1, 0) = \neg(\neg x + \neg y)$

implication: $x \rightarrow y := (\neg x) + y = \max\{z : (x \& z) \le y\}$

Note: $x \lor y = (x \rightarrow y) \rightarrow y, \neg x = (x \rightarrow 0), \dots$

Note: In this talk, fuzzy = Łukasiewicz = Ł

Łukasiewicz functions on [0, 1]

Definition (Fuzzy operations on [0, 1])

 $x \wedge y := \min(x, y)$ conjunction:

disjunction: $x \lor y := \max(x, y)$

 $\neg x := 1 - x$ negation:

weak disjunction: $x+y := \min(x+y,1)$

 $x \& y := \max(x + y - 1, 0) = \neg(\neg x + \neg y)$ strong conjunction:

> $x \to y := (\neg x) + y = \max\{z : (x \& z) < y\}$ implication:

Note: $x \lor y = (x \rightarrow y) \rightarrow y, \neg x = (x \rightarrow 0), \dots$

Łukasiewicz functions on [0, 1]

Definition (Fuzzy operations on [0, 1])

```
conjunction: x \wedge y := \min(x, y)
```

disjunction:
$$x \lor y := \max(x, y)$$

negation:
$$\neg x := 1 - x$$

weak disjunction:
$$x+y := min(x + y, 1)$$

strong conjunction:
$$x \& y := \max(x + y - 1, 0) = \neg(\neg x + \neg y)$$

implication:
$$x \rightarrow y := (\neg x) + y = \max\{z : (x \& z) \le y\}$$

Note:
$$x \lor y = (x \rightarrow y) \rightarrow y$$
, $\neg x = (x \rightarrow 0)$, ...

Note: In this talk, fuzzy = Lukasiewicz = L.

- propositional variables p_1, p_2, \dots
- \triangleright connectives: $+, \&, \lor, \land, \neg, \rightarrow, \top, \bot$
- ▶ formulas: $p_1 \& p_1 \rightarrow p_2, \dots$

Semantics

Ł-Tautologies: $\{\varphi: \forall b \ (\bar{b}(\varphi)=1)\}$

Warning: $p_1 \vee \neg p_1$ is not a tautology.

Syntax:

- propositional variables p_1, p_2, \dots
- ▶ connectives: $+, \&, \lor, \land, \neg, \rightarrow, \top, \bot$
- formulas: $p_1 \& p_1 \rightarrow p_2$, ...

Semantics:

Ł-Tautologies: $\{\varphi: \forall b \ (\bar{b}(\varphi) = 1)\}$

Warning: $p_1 \vee \neg p_1$ is not a tautology.

Syntax:

- propositional variables p₁, p₂, . . .
- ▶ connectives: $+, \&, \lor, \land, \neg, \rightarrow, \top, \bot$
- formulas: $p_1 \& p_1 \rightarrow p_2, \ldots$

Syntax:

- propositional variables p₁, p₂, ...
- connectives: +, &, ∨, ∧, ¬, →, ⊤, ⊥
- formulas: $p_1 \& p_1 \rightarrow p_2, \ldots$

Syntax:

Basic definitions

- propositional variables p₁, p₂, . . .
- \triangleright connectives: $+, \&, \lor, \land, \neg, \rightarrow, \top, \bot$
- ▶ formulas: $p_1 \& p_1 \rightarrow p_2, \dots$

- ▶ Assignments: $b : \{p_1, p_2, ...\} \rightarrow [0, 1]$
- ▶ Truth function \bar{b} : Formulas \rightarrow [0, 1]:

Basic definitions

- propositional variables p₁, p₂, . . .
- \triangleright connectives: $+, \&, \lor, \land, \neg, \rightarrow, \top, \bot$
- ▶ formulas: $p_1 \& p_1 \rightarrow p_2, \dots$

- ▶ Assignments: $b : \{p_1, p_2, ...\} \rightarrow [0, 1]$
- ▶ Truth function \bar{b} : Formulas \rightarrow [0, 1]: $\bar{b}(p) = b(p), \, \bar{b}(\top) = 1, \, \bar{b}(\varphi \& \psi) = \bar{b}(\varphi) \& \bar{b}(\psi), \dots$

Basic definitions

- propositional variables p₁, p₂, . . .
- \triangleright connectives: $+, \&, \lor, \land, \neg, \rightarrow, \top, \bot$
- ▶ formulas: $p_1 \& p_1 \rightarrow p_2, \dots$

- ▶ Assignments: $b : \{p_1, p_2, ...\} \rightarrow [0, 1]$
- ▶ Truth function \bar{b} : Formulas \rightarrow [0, 1]: $\bar{b}(p) = b(p), \, \bar{b}(\top) = 1, \, \bar{b}(\varphi \& \psi) = \bar{b}(\varphi) \& \bar{b}(\psi), \dots$

Basic definitions

- propositional variables p₁, p₂, . . .
- \triangleright connectives: $+, \&, \lor, \land, \neg, \rightarrow, \top, \bot$
- ▶ formulas: $p_1 \& p_1 \rightarrow p_2, \dots$

Semantics:

- ▶ Assignments: $b: \{p_1, p_2, \ldots\} \rightarrow [0, 1]$
- ▶ Truth function \bar{b} : Formulas \rightarrow [0, 1]: $\bar{b}(p) = b(p), \, \bar{b}(\top) = 1, \, \bar{b}(\varphi \& \psi) = \bar{b}(\varphi) \& \bar{b}(\psi), \dots$

Ł-Tautologies: $\{\varphi : \forall b \ (\bar{b}(\varphi) = 1)\}$

Proof ingredients

Syntax:

Basic definitions

- propositional variables p₁, p₂, . . .
- \triangleright connectives: $+, \&, \lor, \land, \neg, \rightarrow, \top, \bot$
- ▶ formulas: $p_1 \& p_1 \rightarrow p_2, \dots$

Semantics:

- ▶ Assignments: $b: \{p_1, p_2, \ldots\} \rightarrow [0, 1]$
- ▶ Truth function \bar{b} : Formulas \rightarrow [0, 1]: $\bar{b}(p) = b(p), \, \bar{b}(\top) = 1, \, \bar{b}(\varphi \& \psi) = \bar{b}(\varphi) \& \bar{b}(\psi), \dots$

Ł-Tautologies: $\{\varphi : \forall b \ (\bar{b}(\varphi) = 1)\}$

Warning: $p_1 \vee \neg p_1$ is not a tautology.

Syntax:

- ► Language A: Relation symbols R S (with arities
- Object variables x, v,
- ▶ connectives, quantifiers: ∧, ∀.

The complexity of Łukasiewicz Logic

- ▶ Language £: Relation symbols R, ..., S (with arities)
- ▶ Object variables x, y, . . .
- ▶ connectives, quantifiers: ∧, ∀, . . .
- ▶ formulas: e.g. $\forall x \exists y (R(x, y) \& R(y, y) \rightarrow S(y, x))$

Syntax:

- ▶ Language \mathcal{L} : Relation symbols R, \ldots, S (with arities)
- Object variables x, y, \dots

- ▶ Language \mathcal{L} : Relation symbols R, \ldots, S (with arities)
- Object variables x, y, ...
- ▶ connectives, quantifiers: ∧, ∀, ...
- ▶ formulas: e.g. $\forall x \exists y (R(x, y) \& R(y, y) \rightarrow S(y, x))$.

Proof ingredients

Łukasiewicz predicate logic

Syntax:

- ▶ Language \mathcal{L} : Relation symbols R, \ldots, S (with arities)
- Object variables x, y, ...
- ▶ connectives, quantifiers: ∧, ∀, ...
- ▶ formulas: e.g. $\forall x \exists y (R(x,y) \& R(y,y) \rightarrow S(y,x))$.

- ▶ \mathcal{L} -structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$: $R^{\mathcal{M}} : M^k \to [0, 1], \dots, S^{\mathcal{M}} : M^m \to [0, 1]$
- ▶ Assignment $v: \{x, v, \ldots\} \rightarrow M$
- > Fuzzv values of formulas: ||∞||³

- $|\varphi||:=\inf\{|\varphi||^M:M$ on C-structure}.
- Ł-validities: $\{ \varphi : \varphi \text{ closed } . \| \varphi \|^{\mathcal{M}} = 1 \text{ for every } \mathcal{M}$

Semantics:

Basic definitions

- ▶ \mathcal{L} -structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$: $R^{\mathcal{M}} : M^k \to [0, 1], \dots, S^{\mathcal{M}} : M^m \to [0, 1].$
- ▶ Assignment $v : \{x, y, \ldots\} \rightarrow M$
- ▶ Fuzzy values of formulas: $\|\varphi\|_{v}^{\mathcal{M}}$.

Ł-validities: $\{\varphi : \varphi \text{ closed }, \|\varphi\|^{\mathcal{M}} = 1 \text{ for every } \mathcal{M}$

Semantics:

- ▶ \mathcal{L} -structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$: $R^{\mathcal{M}} : M^k \to [0, 1], \dots, S^{\mathcal{M}} : M^m \to [0, 1].$
- ▶ Assignment $v : \{x, y, ...\} \rightarrow M$
- ▶ Fuzzy values of formulas: $\|\varphi\|_{\mathcal{V}}^{\mathcal{M}}$.

Semantics:

- \mathcal{L} -structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$: $R^{\mathcal{M}}: M^k \to [0,1], \ldots, S^{\mathcal{M}}: M^m \to [0,1].$
- ▶ Assignment $v: \{x, v, \ldots\} \rightarrow M$
- Fuzzy values of formulas: $\|\varphi\|_{\mathcal{U}}^{\mathcal{M}}$.
 - $||R(x,y)||_{v}^{\mathcal{M}} := R^{\mathcal{M}}(v(x),v(y))$

Semantics:

- \mathcal{L} -structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$: $R^{\mathcal{M}}: M^k \to [0,1], \ldots, S^{\mathcal{M}}: M^m \to [0,1].$
- ▶ Assignment $v: \{x, v, \ldots\} \rightarrow M$
- Fuzzy values of formulas: $\|\varphi\|_{\mathcal{U}}^{\mathcal{M}}$.
 - $||R(x, y)||_{y}^{\mathcal{M}} := R^{\mathcal{M}}(v(x), v(y))$
 - $||\forall x \varphi(x)|| := \inf\{ ||\varphi||_{V_{x_1}, m}^{\mathcal{M}} : m \in M \}$

Semantics:

Basic definitions

- $\mathcal{L}\text{-structure } \mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}}): \\ R^{\mathcal{M}}: M^k \to [0, 1], \dots, S^{\mathcal{M}}: M^m \to [0, 1].$
- ▶ Assignment $v : \{x, y, \ldots\} \rightarrow M$
- Fuzzy values of formulas: $\|\varphi\|_{v}^{\mathcal{M}}$.
 - $||R(x,y)||_{v}^{\mathcal{M}} := R^{\mathcal{M}}(v(x),v(y))$
 - $||\forall x \ \varphi(x)|| := \inf\{ ||\varphi||_{W_{m,n}}^{\mathcal{M}} : m \in M \}$
 - etc

Ł-validities: $\{ \omega : \omega \text{ closed } . \| \omega \|^{\mathcal{M}} = 1 \text{ for every } \mathcal{M}^* = 1 \}$

Semantics:

- \mathcal{L} -structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$: $R^{\mathcal{M}}: M^k \to [0,1], \ldots, S^{\mathcal{M}}: M^m \to [0,1].$
- ▶ Assignment $v: \{x, v, \ldots\} \rightarrow M$
- Fuzzy values of formulas: $\|\varphi\|_{\mathcal{U}}^{\mathcal{M}}$.
 - $||R(x, v)||_{v}^{\mathfrak{M}} := R^{\mathfrak{M}}(v(x), v(v))$
 - $| \forall x \varphi(x) | := \inf \{ \|\varphi\|_{V_0, \infty}^{\mathcal{M}} : m \in M \}$
 - etc.

Semantics:

- \mathcal{L} -structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$: $R^{\mathcal{M}}: M^k \to [0,1], \ldots, S^{\mathcal{M}}: M^m \to [0,1].$
- ▶ Assignment $v: \{x, v, \ldots\} \rightarrow M$
- Fuzzy values of formulas: $\|\varphi\|_{\mathcal{U}}^{\mathcal{M}}$.
 - $||R(x, v)||_{v}^{\mathfrak{M}} := R^{\mathfrak{M}}(v(x), v(v))$
 - $| \forall x \varphi(x) | := \inf \{ \|\varphi\|_{V_0, \infty}^{\mathcal{M}} : m \in M \}$
 - etc.

Semantics:

Basic definitions

- ▶ \mathcal{L} -structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$: $R^{\mathcal{M}} : M^k \to [0, 1], \dots, S^{\mathcal{M}} : M^m \to [0, 1].$
- ▶ Assignment $v : \{x, y, ...\} \rightarrow M$
- ▶ Fuzzy values of formulas: $\|\varphi\|_{V}^{\mathcal{M}}$.
 - $||R(x,y)||_{\nu}^{\mathfrak{M}} := R^{\mathfrak{M}}(\nu(x),\nu(y))$
 - $| | \forall x \ \varphi(x) | | := \inf \{ | | \varphi | |_{W_{m,m}}^{\mathcal{M}} : m \in M \}$
 - etc.

Ł-validities: $\{\varphi : \varphi \text{ closed }, \|\varphi\|^{\mathfrak{M}} = 1 \text{ for every } \mathfrak{M} \}$

Computable structures

Łukasiewicz predicate logic 2

- \mathcal{L} -structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$: $R^{\mathcal{M}}: M^k \to [0,1], \ldots, S^{\mathcal{M}}: M^m \to [0,1].$
- ▶ Assignment $v : \{x, y, ...\} \rightarrow M$
- ▶ Fuzzy values of formulas: $\|\varphi\|_{\mathcal{U}}^{\mathcal{M}}$.
 - $||R(x, v)||_{v}^{\mathfrak{M}} := R^{\mathfrak{M}}(v(x), v(v))$
 - $| \forall x \varphi(x) | := \inf \{ \|\varphi\|_{V_0, \infty}^{\mathcal{M}} : m \in M \}$
 - etc.
- $\|\varphi\| := \inf\{\|\varphi\|^{\mathcal{M}} : \mathcal{M} \text{ an } \mathcal{L}\text{-structure}\}.$

Semantics:

- \mathcal{L} -structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$: $R^{\mathcal{M}}: M^k \to [0,1], \ldots, S^{\mathcal{M}}: M^m \to [0,1].$
- ▶ Assignment $v : \{x, y, ...\} \rightarrow M$
- ▶ Fuzzy values of formulas: $\|\varphi\|_{\mathcal{U}}^{\mathcal{M}}$.
 - $||R(x, v)||_{v}^{\mathfrak{M}} := R^{\mathfrak{M}}(v(x), v(v))$
 - $| \forall x \varphi(x) | := \inf \{ \|\varphi\|_{V_0, \infty}^{\mathcal{M}} : m \in M \}$
 - etc.
- $\|\varphi\| := \inf\{\|\varphi\|^{\mathfrak{M}} : \mathfrak{M} \text{ an } \mathcal{L}\text{-structure}\}.$
- Ł-validities: $\{\varphi : \varphi \text{ closed }, \|\varphi\|^{\mathfrak{M}} = 1 \text{ for every } \mathfrak{M}\}.$

Outline

Pacia definition

Complexity results

The arithmetical hierarch

Computable structures

Proof ingredient

Summar

Classical propositional logic on {0, 1}

The set $\{\varphi : \forall b \ (\bar{b}(\varphi) = 1)\}$ of classical (or "crisp") tautologies is

decidable;

Basic definitions

► co-NP-complete. [folklore?]

```
Propositional Ł-logic on [0,1]:
The set \{\varphi: \forall b \ (\bar{b}(\varphi)=1\} \text{ of Ł-Tautologies is}
```

Classical propositional logic on {0, 1}

The set $\{\varphi : \forall b \ (\bar{b}(\varphi) = 1\}$ of classical (or "crisp") tautologies is

decidable:

Basic definitions

co-NP-complete. [folklore?]

Classical propositional logic on {0, 1}

The set $\{\varphi : \forall b \ (\bar{b}(\varphi) = 1\}$ of classical (or "crisp") tautologies is

decidable:

Basic definitions

co-NP-complete. [folklore?]

Classical propositional logic on {0, 1}

The set $\{\varphi : \forall b \ (\bar{b}(\varphi) = 1) \}$ of classical (or "crisp") tautologies is

decidable:

Basic definitions

co-NP-complete. [folklore?]

Propositional Ł-logic on [0, 1]:

The set $\{\varphi : \forall b \ (\bar{b}(\varphi) = 1\} \text{ of } \& \text{-Tautologies is } \}$

- decidable:
- co-NP-complete. [same proof]

Classical propositional logic on {0, 1}

The set $\{\varphi : \forall b \ (\bar{b}(\varphi) = 1) \}$ of classical (or "crisp") tautologies is

decidable:

Basic definitions

co-NP-complete. [folklore?]

Propositional Ł-logic on [0, 1]:

The set $\{\varphi : \forall b \ (\bar{b}(\varphi) = 1\} \text{ of } \& \text{-Tautologies is } \}$

- decidable:
- co-NP-complete. [same proof]

Complexity - first order

Basic definitions

Classical first order predicate logic on {0, 1}:

The set $\{\varphi : \mathcal{M} \models \varphi \text{ for all crisp } \mathcal{M}\}$ of classical validities is

- not decidable

Classical first order predicate logic on {0, 1}:

The set $\{\varphi : \mathcal{M} \models \varphi \text{ for all crisp } \mathcal{M}\}$ of classical validities is

- not decidable
- computably enumerable (c.e., Σ₁⁰)
- \triangleright in fact: Σ_1^0 -complete.

Classical first order predicate logic on {0, 1}:

The set $\{\varphi : \mathcal{M} \models \varphi \text{ for all crisp } \mathcal{M}\}$ of classical validities is

- not decidable
- computably enumerable (c.e., Σ₁⁰)
- in fact: Σ⁰₁-complete.

Complexity - first order

Basic definitions

Classical first order predicate logic on {0, 1}:

The set $\{\varphi: \mathfrak{M} \models \varphi \text{ for all crisp } \mathfrak{M}\}$ of classical validities is

- not decidable
- ▶ computably enumerable (c.e., Σ_1^0)
- in fact: Σ_1^0 -complete.

```
First order Ł-logic on [0, 1]:
```

The set $\{\varphi : \|\varphi\|^{\mathfrak{M}} = 1 \text{ for all fuzzy } \mathfrak{M}\}$ of Ł-validities is

- ▶ not decidable, not Σ_1^{ν} , not even Σ_2^{ν} (Scarpellini)
- □ Π⁰ (Novak-Pavelka)
- ► Π⁰₂-complete (Ragaz; G*)

Complexity - first order

Basic definitions

Classical first order predicate logic on {0, 1}:

The set $\{\varphi : \mathcal{M} \models \varphi \text{ for all crisp } \mathcal{M}\}$ of classical validities is

- not decidable
- computably enumerable (c.e., Σ₁⁰)
- in fact: Σ⁰₁-complete.

First order Ł-logic on [0, 1]:

The set $\{\varphi : \|\varphi\|^{\mathfrak{M}} = 1 \text{ for all fuzzy } \mathfrak{M}\}$ of \pounds -validities is

- ▶ not decidable, not Σ_1^0 , not even Σ_2^0 (Scarpellini)
- Π⁰ (Novak-Pavelka)

Classical first order predicate logic on {0, 1}:

The set $\{\varphi : \mathfrak{M} \models \varphi \text{ for all crisp } \mathfrak{M}\}$ of classical validities is

- not decidable
- computably enumerable (c.e., Σ_1^0)
- in fact: Σ_1^0 -complete.

First order Ł-logic on [0, 1]:

The set $\{\varphi : \|\varphi\|^{\mathfrak{M}} = 1 \text{ for all fuzzy } \mathfrak{M}\}$ of ξ -validities is

- ▶ not decidable, not Σ_1^0 , not even Σ_2^0 (Scarpellini)
- ► П₂⁰ (Novak-Pavelka)
- ► Π₂⁰-complete (Ragaz; G*)

Complexity - first order

Basic definitions

Classical first order predicate logic on {0, 1}:

The set $\{\varphi : \mathcal{M} \models \varphi \text{ for all crisp } \mathcal{M}\}$ of classical validities is

- not decidable
- computably enumerable (c.e., Σ₁⁰)
- in fact: Σ⁰₁-complete.

First order Ł-logic on [0, 1]:

The set $\{\varphi : \|\varphi\|^{\mathfrak{M}} = 1 \text{ for all fuzzy } \mathfrak{M}\}$ of \pounds -validities is

- ▶ not decidable, not Σ_1^0 , not even Σ_2^0 (Scarpellini)
- Π₀⁰ (Novak-Pavelka)
- Π₂-complete (Ragaz; G*)

Outline

Racio dofinition

Complexity regults

The arithmetical hierarchy

Computable structures

Proof ingredient

Summar

First order language of arithmetic: $+, \cdot, \leq, =, 0, 1$.

Abbreviation: $\vec{x} = (x_1, \dots, x_n), \vec{y} = (y_1, \dots, y_k).$

 $ightharpoonup \Sigma_1^0$ -formulas: $\exists x_1 \psi(x_1, \vec{y})$, where ψ is quantifier-free (or:

only bounded quantitiers: $\forall u < v, \exists u < v.$)

 \blacktriangleright 11;-iormulas: $\forall x_1 \psi$, or $\neg (\Sigma_1)$.

 $ightharpoonup \Sigma_{n}^{+}$ -formulas: $\exists x_1 \ \forall x_2 \cdots \ \exists x_n \ \psi(\bar{x}, \bar{y})$

Remark

Basic definitions

Most arithmetical formulas that appear in practice are Σ_n^0 , for small n. (n = 1, 2, 3.)

Example: "there are infinitely many twin primes"

 $\forall x \exists p \ (p > x, p \ prime, p + 2 \ prime)$.

arithmetical hierarchy: formulas

First order language of arithmetic: $+, \cdot, <, =, 0, 1$.

Abbreviation: $\vec{x} = (x_1, \dots, x_n), \vec{y} = (y_1, \dots, y_k).$

- ▶ Σ_1^0 -formulas: $\exists x_1 \psi(x_1, \vec{y})$, where ψ is quantifier-free (or: only bounded quantifiers: $\forall u < v, \exists u < v.$)
- $ightharpoonup \Pi_1^0$ -formulas: $\forall x_1 \psi$, or $\neg(\Sigma_1^0)$.

Basic definitions

First order language of arithmetic: $+, \cdot, <, =, 0, 1$. Abbreviation: $\vec{x} = (x_1, \dots, x_n), \vec{y} = (y_1, \dots, y_k).$

- ▶ Σ_1^0 -formulas: $\exists x_1 \psi(x_1, \vec{y})$, where ψ is quantifier-free (or: only bounded quantifiers: $\forall u < v, \exists u < v.$)
- $ightharpoonup \Pi_1^0$ -formulas: $\forall x_1 \psi$, or $\neg(\Sigma_1^0)$.
- \triangleright Σ_n^0 -formulas: $\exists x_1 \forall x_2 \cdots \exists x_n \psi(\vec{x}, \vec{y})$

Basic definitions

First order language of arithmetic: $+, \cdot, \leq, =, 0, 1$. Abbreviation: $\vec{x} = (x_1, \dots, x_n), \vec{y} = (y_1, \dots, y_k)$.

- ▶ Σ_1^0 -formulas: $\exists x_1 \psi(x_1, \vec{y})$, where ψ is quantifier-free (or: only bounded quantifiers: $\forall u < v, \exists u < v.$)
- ▶ Π_1^0 -formulas: $\forall x_1 \psi$, or $\neg(\Sigma_1^0)$.
- ► Σ_n^0 -formulas: $\exists x_1 \forall x_2 \cdots \forall x_n \psi(\vec{x}, \vec{y})$

Remark

Basic definitions

Most arithmetical formulas that appear in practice are Σ_n^0 , for small n. (n = 1, 2, 3.)

Example: "there are infinitely many twin primes" $\forall x \exists p \ (p > x, p \text{ prime}, p + 2 \text{ prime}).$

First order language of arithmetic: $+,\cdot,\leq,=,0,1.$

Abbreviation: $\vec{x} = (x_1, \dots, x_n), \vec{y} = (y_1, \dots, y_k).$

- ▶ Σ_1^0 -formulas: $\exists x_1 \psi(x_1, \vec{y})$, where ψ is quantifier-free (or: only bounded quantifiers: $\forall u < v, \exists u < v.$)
- ▶ Π_1^0 -formulas: $\forall x_1 \psi$, or $\neg(\Sigma_1^0)$.
- ► Σ_n^0 -formulas: $\exists x_1 \forall x_2 \cdots \forall x_n \psi(\vec{x}, \vec{y})$

Remark

Basic definitions

Most arithmetical formulas that appear in practice are Σ_n^0 , for small n. (n = 1, 2, 3.)

Example: "there are infinitely many twin primes".

First order language of arithmetic: $+, \cdot, <, =, 0, 1$.

Abbreviation: $\vec{x} = (x_1, \dots, x_n), \ \vec{y} = (y_1, \dots, y_k).$

- ▶ Σ_1^0 -formulas: $\exists x_1 \psi(x_1, \vec{y})$, where ψ is quantifier-free (or: only bounded quantifiers: $\forall u < v, \exists u < v.$)
- $ightharpoonup \Pi_1^0$ -formulas: $\forall x_1 \psi$, or $\neg(\Sigma_1^0)$.
- \triangleright Σ_n^0 -formulas: $\exists x_1 \forall x_2 \cdots \exists x_n \psi(\vec{x}, \vec{y})$

Remark

Basic definitions

Most arithmetical formulas that appear in practice are Σ_n^0 , for small n. (n = 1, 2, 3.)

First order language of arithmetic: $+, \cdot, <, =, 0, 1$.

Abbreviation: $\vec{x} = (x_1, \dots, x_n), \ \vec{y} = (y_1, \dots, y_k).$

- Σ_1^0 -formulas: $\exists x_1 \psi(x_1, \vec{y})$, where ψ is quantifier-free (or: only bounded quantifiers: $\forall u < v, \exists u < v.$)
- $ightharpoonup \Pi_1^0$ -formulas: $\forall x_1 \psi$, or $\neg(\Sigma_1^0)$.
- Σ_{n}^{0} -formulas: $\exists x_{1} \forall x_{2} \cdots \exists x_{n} \psi(\vec{x}, \vec{y})$

Remark

Basic definitions

Most arithmetical formulas that appear in practice are Σ_n^0 , for small n. (n = 1, 2, 3.)

Example: "there are infinitely many twin primes":

$$\forall x \exists p \ (p > x, p \ prime, p + 2 \ prime).$$

arithmetical hierarchy: sets

Basic definitions

- The Σ⁰ sets are exactly the c.e. (r.e.) sets, or semi-decidable sets. (projections of decidable sets in \mathbb{N}^{k+1}
- ▶ The decidable sets are exactly the sets which are both Σ_1^0

- ► The Σ_{\perp}^{0} sets are exactly the c.e. (r.e.) sets, or semi-decidable sets. (projections of decidable sets in \mathbb{N}^{k+1}
- ► The decidable sets are exactly the sets which are both Σ_1^0 and Π_1^0 .
- $\triangleright \Sigma_1^0 \subsetneq \Sigma_2^0 \subsetneq \cdots$, similarly $\Pi_1^0 \subsetneq \Pi_2^0 \subsetneq \cdots$.

- The Σ⁰ sets are exactly the c.e. (r.e.) sets, or semi-decidable sets. (projections of decidable sets in \mathbb{N}^{k+1}
- The decidable sets are exactly the sets which are both Σ₁⁰ and Π_1^0 .
- $ightharpoonup \Sigma_1^0 \subsetneq \Sigma_2^0 \subsetneq \cdots$, similarly $\Pi_1^0 \subsetneq \Pi_2^0 \subsetneq \cdots$.
- ▶ If C is Π_n^0 , and f is computable, then $f^{-1}(C)$ is also Π_n^0 .

- The Σ⁰ sets are exactly the c.e. (r.e.) sets, or semi-decidable sets. (projections of decidable sets in \mathbb{N}^{k+1}
- The decidable sets are exactly the sets which are both Σ₁⁰ and Π_1^0 .
- $ightharpoonup \Sigma_1^0 \subsetneq \Sigma_2^0 \subsetneq \cdots$, similarly $\Pi_1^0 \subsetneq \Pi_2^0 \subsetneq \cdots$.
- ▶ If C is Π_n^0 , and f is computable, then $f^{-1}(C)$ is also Π_n^0 .
- ightharpoonup C is a complete Π_n^0 -set, if C is Π_n^0 , and every Π_n^0 -set B can

arithmetical hierarchy: sets

Basic definitions

- The Σ⁰ sets are exactly the c.e. (r.e.) sets, or semi-decidable sets. (projections of decidable sets in \mathbb{N}^{k+1}
- ► The decidable sets are exactly the sets which are both Σ_1^0 and Π_1^0 .
- $ightharpoonup \Sigma_1^0 \subsetneq \Sigma_2^0 \subsetneq \cdots$, similarly $\Pi_1^0 \subsetneq \Pi_2^0 \subsetneq \cdots$.
- ▶ If C is Π_n^0 , and f is computable, then $f^{-1}(C)$ is also Π_n^0 .
- ightharpoonup C is a complete Π_n^0 -set, if C is Π_n^0 , and every Π_n^0 -set B can be reduced to C, i.e., is of the form $f^{-1}(C)$, for some computable f.

- ► The Σ_{\perp}^{0} sets are exactly the c.e. (r.e.) sets, or semi-decidable sets. (projections of decidable sets in \mathbb{N}^{k+1}
- ► The decidable sets are exactly the sets which are both Σ_1^0 and Π_1^0 .
- $ightharpoonup \Sigma_1^0 \subsetneq \Sigma_2^0 \subsetneq \cdots$, similarly $\Pi_1^0 \subsetneq \Pi_2^0 \subsetneq \cdots$.
- ▶ If C is Π_n^0 , and f is computable, then $f^{-1}(C)$ is also Π_n^0 .
- ightharpoonup C is a complete Π_n^0 -set, if C is Π_n^0 , and every Π_n^0 -set B can be reduced to C, i.e., is of the form $f^{-1}(C)$, for some computable f.

arithmetical hierarchy: sets

Basic definitions

A subset of \mathbb{N}^k is Σ_n^0 iff it can be defined by a Σ_n^0 -formula.

- ► The Σ_{\perp}^{0} sets are exactly the c.e. (r.e.) sets, or semi-decidable sets. (projections of decidable sets in \mathbb{N}^{k+1}
- ► The decidable sets are exactly the sets which are both Σ_1^0 and Π_1^0 .
- $ightharpoonup \Sigma_1^0 \subsetneq \Sigma_2^0 \subsetneq \cdots$, similarly $\Pi_1^0 \subsetneq \Pi_2^0 \subsetneq \cdots$.
- ▶ If C is Π_n^0 , and f is computable, then $f^{-1}(C)$ is also Π_n^0 .
- ightharpoonup C is a complete Π_n^0 -set, if C is Π_n^0 , and every Π_n^0 -set B can be reduced to C, i.e., is of the form $f^{-1}(C)$, for some computable f.

(These are the sets which are maximally complicated among the Π_n^0 sets, similar to co-NP-complete)

Examples of ...-complete sets

- ▶ The set of all (codes for) Turing machines that halt on input 0 is Σ^0 complete.
- The set of all (codes for) programs that describe a function with infinite domain is Π⁰₂-complete.
- ▶ The set $Th_n(\mathbb{N})$ of all (codes for) true Σ_n^0 -formulas is Σ_n^0 -complete.

Definition

The set $Th(\mathbb{N})$ (also called true arithmetic) is defined as the set of all (codes for) true sentences: $Th(\mathbb{N}) = \bigcup_{n=1}^{\infty} Th_n(\mathbb{N})$. $Th(\mathbb{N})$ is "infinitely more" complicated than any $Th_n(\mathbb{N})$.

Examples of ...-complete sets

Complexity results

- The set of all (codes for) Turing machines that halt on input 0 is Σ_1^0 complete.
- ► The set of all (codes for) programs that describe a function

Basic definitions

Examples of ...-complete sets

- The set of all (codes for) Turing machines that halt on input 0 is Σ⁰₁ complete.
- The set of all (codes for) programs that describe a function with infinite domain is Π₂⁰-complete.
- ► The set $Th_n(\mathbb{N})$ of all (codes for) true Σ_n^0 -formulas is Σ_n^0 -complete.

Definition

The set $Th(\mathbb{N})$ (also called true arithmetic) is defined as the set of all (codes for) true sentences: $Th(\mathbb{N}) = \bigcup_{n=1}^{\infty} Th_n(\mathbb{N})$.

- The set of all (codes for) Turing machines that halt on input 0 is Σ_1^0 complete.
- The set of all (codes for) programs that describe a function with infinite domain is Π_2^0 -complete.
- ▶ The set $Th_n(\mathbb{N})$ of all (codes for) true Σ_n^0 -formulas is Σ_n^0 -complete.

- The set of all (codes for) Turing machines that halt on input 0 is Σ₁ complete.
- The set of all (codes for) programs that describe a function with infinite domain is Π₂⁰-complete.
- ► The set $Th_n(\mathbb{N})$ of all (codes for) true Σ_n^0 -formulas is Σ_n^0 -complete.

The set $Th(\mathbb{N})$ (also called true arithmetic) is defined as the set of all (codes for) true sentences: $Th(\mathbb{N}) = \bigcup_{n=1}^{\infty} Th_n(\mathbb{N})$. $Th(\mathbb{N})$ is "infinitely more" complicated than any $Th_n(\mathbb{N})$.

- The set of all (codes for) Turing machines that halt on input 0 is Σ_1^0 complete.
- The set of all (codes for) programs that describe a function with infinite domain is Π_2^0 -complete.
- ▶ The set $Th_n(\mathbb{N})$ of all (codes for) true Σ_n^0 -formulas is Σ_n^0 -complete.

Basic definitions

The set $Th(\mathbb{N})$ (also called true arithmetic) is defined as the set of all (codes for) true sentences: $Th(\mathbb{N}) = \bigcup_{n=1}^{\infty} Th_n(\mathbb{N})$.

- The set of all (codes for) Turing machines that halt on input 0 is Σ_1^0 complete.
- The set of all (codes for) programs that describe a function with infinite domain is Π_2^0 -complete.
- ▶ The set $Th_n(\mathbb{N})$ of all (codes for) true Σ_n^0 -formulas is Σ_n^0 -complete.

Basic definitions

The set $Th(\mathbb{N})$ (also called true arithmetic) is defined as the set of all (codes for) true sentences: $Th(\mathbb{N}) = \bigcup_{n=1}^{\infty} Th_n(\mathbb{N})$. $Th(\mathbb{N})$ is "infinitely more" complicated than any $Th_n(\mathbb{N})$.

Outline

Racio definition

Complexity results

The arithmetical hierarch

Computable structures

Proof ingredient

Summar

Computable structures

Definition

Basic definitions

A crisp structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$ is "computable", if

- M is a decidable subset of N.

Computable structures

Definition

Basic definitions

A crisp structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$ is "computable", if

- M is a decidable subset of N.
- for each relation symbol R, the set \mathbb{R}^{M} is a decidable subset of the respective \mathbb{N}^k .

Basic definitions

A crisp structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$ is "computable", if

- ▶ M is a decidable subset of \mathbb{N} ,
- for each relation symbol R, the set \mathbb{R}^{M} is a decidable subset of the respective \mathbb{N}^k .

Computable structures

Definition

Basic definitions

A crisp structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$ is "computable", if

- \blacktriangleright *M* is a decidable subset of \mathbb{N} .
- for each relation symbol R, the set $R^{\mathbb{M}}$ is a decidable subset of the respective \mathbb{N}^k .

A fuzzy structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$ is "computable", if

- M is a decidable subset of N,
- ▶ for each relation symbol R the sets $\{(\vec{m}, q) : R^{\mathcal{M}}(\vec{m}) < q\}$

Definition

Basic definitions

A crisp structure $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$ is "computable", if

- ▶ M is a decidable subset of \mathbb{N} ,
- for each relation symbol R, the set $R^{\mathbb{N}}$ is a decidable subset of the respective \mathbb{N}^k .

A fuzzy structure $\mathfrak{M}=(M,R^{\mathfrak{M}},\ldots,S^{\mathfrak{M}})$ is "computable", if

- ▶ M is a decidable subset of \mathbb{N} ,
- ▶ for each relation symbol R the sets $\{(\vec{m}, q) : R^{\mathcal{M}}(\vec{m}) < q\}$ and $\{(\vec{m}, q) : R^{\mathcal{M}}(\vec{m}) \leq q\}$ are decidable subsets of the respective $\mathbb{N}^k \times \mathbb{Q}$.

Recall

Basic definitions

- φ is classically valid, if $\mathfrak{M} \models \varphi$ for all crisp structures \mathfrak{M} ;
- ho φ is k-valid, if $\|\varphi\|^{\mathfrak{M}} = 1$, for all fuzzy structures \mathfrak{M} .

Definition

 φ is C-valid,if $\mathcal{M}\models\varphi$ for all computable crisp structures $\mathcal{M}.$ φ is C-Ł-valid,if $\mathcal{M}\models\varphi$ for all computable fuzzy structures \mathcal{M}

Recall

- φ is classically valid, if $\mathfrak{M} \models \varphi$ for all crisp structures \mathfrak{M} ;
- φ is $\mbox{$\xi$-valid}$, if $\|\varphi\|^{\mathcal{M}}=1$, for all fuzzy structures \mathcal{M} .

Definition

 φ is C-valid,if $\mathfrak{M}\models\varphi$ for all computable crisp structures $\mathfrak{M}.$ φ is C-&-valid,if $\mathfrak{M}\models\varphi$ for all computable fuzzy structures \mathfrak{M}

Recall

Basic definitions

- φ is classically valid, if $\mathfrak{M} \models \varphi$ for all crisp structures \mathfrak{M} ;
- φ is $\mbox{$\xi$-valid}$, if $\|\varphi\|^{\mathcal{M}}=1$, for all fuzzy structures \mathcal{M} .

Definition

```
\varphi is C-valid,if \mathfrak{M}\models\varphi for all computable crisp structures \mathfrak{M}. \varphi is C-Ł-valid,if \mathfrak{M}\models\varphi for all computable fuzzy structures \mathfrak{M}
```

Recall

Basic definitions

- φ is classically valid, if $\mathfrak{M} \models \varphi$ for all crisp structures \mathfrak{M} ;
- φ is $\mbox{$\mbox{$\rlap$$L$-valid, if}} \|\varphi\|^{\mbox{$\mbox{$\mbox{$}$}}}=1$, for all fuzzy structures $\mbox{$\mbox{$\mbox{$}$}}$.

Definition

```
\varphi is C-valid,if \mathfrak{M}\models\varphi for all computable crisp structures \mathfrak{M}. \varphi is C-L-valid,if \mathfrak{M}\models\varphi for all computable fuzzy structures \mathfrak{M}
```

Theorem

 The set of C-validities is as complicated as Th(N) (true arithmetic).

Recall

- φ is classically valid, if $\mathfrak{M} \models \varphi$ for all crisp structures \mathfrak{M} ;
- φ is $\mbox{$\mbox{$\rlap$$L$-valid, if}} \|\varphi\|^{\mbox{$\mbox{$\mbox{$}$}}}=1$, for all fuzzy structures $\mbox{$\mbox{$\mbox{$}$}}$.

Definition

```
\varphi is C-valid,if \mathcal{M} \models \varphi for all computable crisp structures \mathcal{M}. \varphi is C-Ł-valid,if \mathcal{M} \models \varphi for all computable fuzzy structures \mathcal{M}.
```

- The set of C-validities is as complicated as Th(N) (true arithmetic).
- 2. The set of C-k-validities is as complicated as Th(N)

Recall

- φ is classically valid, if $\mathfrak{M} \models \varphi$ for all crisp structures \mathfrak{M} ;
- φ is $\mbox{$\mbox{$\rlap$$L$-valid, if}} \|\varphi\|^{\mbox{$\mbox{$\mbox{$}$}}}=1$, for all fuzzy structures $\mbox{$\mbox{$\mbox{$}$}}$.

Definition

```
\varphi is C-valid,if \mathfrak{M} \models \varphi for all computable crisp structures \mathfrak{M}. \varphi is C-Ł-valid,if \mathfrak{M} \models \varphi for all computable fuzzy structures \mathfrak{M}.
```

- 1. The set of C-validities is as complicated as $Th(\mathbb{N})$ (true arithmetic).
- 2. The set of C-k-validities is as complicated as $Th(\mathbb{N})$

Recall

Basic definitions

- $\triangleright \varphi$ is classically valid, if $\mathfrak{M} \models \varphi$ for all crisp structures \mathfrak{M} ;
- φ is Ł-valid, if $\|\varphi\|^{\mathcal{M}} = 1$, for all fuzzy structures \mathcal{M} .

Definition

```
\varphi is C-valid, if \mathcal{M} \models \varphi for all computable crisp structures \mathcal{M}.
\varphi is C-Ł-valid, if \mathfrak{M} \models \varphi for all computable fuzzy structures \mathfrak{M}.
```

- 1. The set of C-validities is as complicated as $Th(\mathbb{N})$ (true arithmetic).
- 2. The set of C-k-validities is as complicated as $Th(\mathbb{N})$.

Outline

Racio definition

Complexity regults

The antilinetical meran

Computable structures

Proof ingredients

Summar

While any set of sentences (true in \mathbb{N}) has uncountably many (pairwise nonisomorphic) countable models, we have:

Theorem (Tennenbaum 1959)

There is a single sentences σ such that $\mathbb N$ is the unique computable model satisfying σ .

Corollary

Basic definitions

Th(N) can be computed from the set of C-validities

Part (1) is well-known and follows easily from Tennenbaum's 1959 theorem. The proof of part (2) is similar.

While any set of sentences (true in \mathbb{N}) has uncountably many (pairwise nonisomorphic) countable models, we have:

Theorem (Tennenbaum 1959)

There is a single sentences σ such that $\mathbb N$ is the unique computable model satisfying σ .

While any set of sentences (true in \mathbb{N}) has uncountably many (pairwise nonisomorphic) countable models, we have:

Theorem (Tennenbaum 1959)

There is a single sentences σ such that \mathbb{N} is the unique computable model satisfying σ .

Corollary

- $Th(\mathbb{N})$ can be computed from the set of C-validities.

While any set of sentences (true in \mathbb{N}) has uncountably many (pairwise nonisomorphic) countable models, we have:

Theorem (Tennenbaum 1959)

There is a single sentences σ such that $\mathbb N$ is the unique computable model satisfying σ .

Corollary

- 1. $Th(\mathbb{N})$ can be computed from the set of C-validities.
- 2. $Th(\mathbb{N})$ can be computed from the set of C-k-validities.

Part (1) is well-known and follows easily from Tennenbaum's 1959 theorem. The proof of part (2) is similar.

While any set of sentences (true in \mathbb{N}) has uncountably many (pairwise nonisomorphic) countable models, we have:

Theorem (Tennenbaum 1959)

There is a single sentences σ such that $\mathbb N$ is the unique computable model satisfying σ .

Corollary

- 1. $Th(\mathbb{N})$ can be computed from the set of C-validities.
- 2. $Th(\mathbb{N})$ can be computed from the set of C-k-validities.

Part (1) is well-known and follows easily from Tennenbaum's 1959 theorem. The proof of part (2) is similar.

While any set of sentences (true in \mathbb{N}) has uncountably many (pairwise nonisomorphic) countable models, we have:

Theorem (Tennenbaum 1959)

There is a single sentences σ such that \mathbb{N} is the unique computable model satisfying σ .

Corollary

Basic definitions

- 1. $Th(\mathbb{N})$ can be computed from the set of C-validities.
- 2. $Th(\mathbb{N})$ can be computed from the set of C-k-validities.

Part (1) is well-known and follows easily from Tennenbaum's 1959 theorem. The proof of part (2) is similar.

Computable structures

From fuzzy to crisp via rounding

Fix a language \mathcal{L} with finitely many relation symbols R, \ldots, S .

- $\triangleright \ \varepsilon_{R} := \exists x_{1} \cdots \exists x_{k} (R(\vec{x}) \land \neg R(\vec{x})) \text{ for } k\text{-ary } R$
- \triangleright $\varepsilon_L := \varepsilon_R \lor \cdots \lor \varepsilon_S$ (disjunction over all relation symbols)

Fix a language \mathcal{L} with finitely many relation symbols R, \ldots, S .

- $\triangleright \ \varepsilon_{R} := \exists x_{1} \cdots \exists x_{k} (R(\vec{x}) \land \neg R(\vec{x})) \text{ for } k\text{-ary } R$
- \triangleright $\varepsilon_L := \varepsilon_R \lor \cdots \lor \varepsilon_S$ (disjunction over all relation symbols)

Let $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$ be a fuzzy \mathcal{L} -structure.

Define a crisp structure $\bar{\mathcal{M}}$ and a number $e^{\mathcal{M}} \in [0, \frac{1}{2}]$ as follows:

From fuzzy to crisp via rounding

Fix a language \mathcal{L} with finitely many relation symbols R, \ldots, S .

- $\triangleright \ \varepsilon_{R} := \exists x_{1} \cdots \exists x_{k} (R(\vec{x}) \land \neg R(\vec{x})) \text{ for } k\text{-ary } R$
- \triangleright $\varepsilon_L := \varepsilon_R \lor \cdots \lor \varepsilon_S$ (disjunction over all relation symbols)

Let $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$ be a fuzzy \mathcal{L} -structure. Define a crisp structure $\bar{\mathcal{M}}$ and a number $e^{\mathcal{M}} \in [0, \frac{1}{2}]$ as follows:

- ▶ The universe \overline{M} is the same as the universe of M: $\overline{M} := M$.
- ► For each *k*-ary relation symbol *R*:

Basic definitions

Fix a language \mathcal{L} with finitely many relation symbols R, \ldots, S .

- $ightharpoonup arepsilon_{\mathcal{L}} := \varepsilon_R \lor \cdots \lor \varepsilon_{\mathcal{S}}$ (disjunction over all relation symbols)

Let $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$ be a fuzzy \mathcal{L} -structure. Define a crisp structure $\overline{\mathcal{M}}$ and a number $e^{\mathcal{M}} \in [0, \frac{1}{2}]$ as follows:

- ▶ The universe \bar{M} is the same as the universe of M: $\bar{M} := M$.
- For each k-ary relation symbol R: For all $\vec{a} \in M^k$: $\bar{\mathcal{M}} \models R(\vec{a})$ iff: $||R(\vec{a})||^{\mathcal{M}} > \frac{1}{2}$
- $ightharpoonup e^{\mathcal{M}} := \|\varepsilon_{\mathcal{L}}\|^{\mathcal{M}}.$

Note: $e^{\mathcal{M}}=0$ iff \mathcal{M} is crisp. Try to avoid the case $e^{M}=rac{1}{2}$.

Basic definitions

From fuzzy to crisp via rounding

Fix a language \mathcal{L} with finitely many relation symbols R, \ldots, S .

- $ightharpoonup arepsilon_{\mathcal{L}} := \varepsilon_R \lor \cdots \lor \varepsilon_{\mathcal{S}}$ (disjunction over all relation symbols)

Let $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$ be a fuzzy \mathcal{L} -structure. Define a crisp structure $\overline{\mathcal{M}}$ and a number $e^{\mathcal{M}} \in [0, \frac{1}{2}]$ as follows:

- ▶ The universe \bar{M} is the same as the universe of M: $\bar{M} := M$.
- For each k-ary relation symbol R: For all $\vec{a} \in M^k$: $\bar{\mathcal{M}} \models R(\vec{a})$ iff: $||R(\vec{a})||^{\mathcal{M}} > \frac{1}{2}$
- $ightharpoonup e^{\mathfrak{M}} := \|\varepsilon_{\mathcal{L}}\|^{\mathfrak{M}}.$

Note: $e^{\mathcal{M}} = 0$ iff \mathcal{M} is crisp. Try to avoid the case $e^{M} = \frac{1}{2}$.

From fuzzy to crisp via rounding

Fix a language \mathcal{L} with finitely many relation symbols R, \ldots, S .

- $ightharpoonup \varepsilon_R := \exists x_1 \cdots \exists x_k (R(\vec{x}) \land \neg R(\vec{x})) \text{ for } k\text{-ary } R$
- $ightharpoonup arepsilon_{\mathcal{L}} := \varepsilon_R \lor \cdots \lor \varepsilon_{\mathcal{S}}$ (disjunction over all relation symbols)

Let $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$ be a fuzzy \mathcal{L} -structure. Define a crisp structure $\overline{\mathcal{M}}$ and a number $e^{\mathcal{M}} \in [0, \frac{1}{2}]$ as follows:

- ▶ The universe \bar{M} is the same as the universe of M: $\bar{M} := M$.
- For each k-ary relation symbol R: For all $\vec{a} \in M^k$: $\bar{\mathcal{M}} \models R(\vec{a})$ iff: $\|R(\vec{a})\|^{\mathcal{M}} > \frac{1}{2}$
- $ightharpoonup e^{\mathfrak{M}} := \|\varepsilon_{\mathcal{L}}\|^{\mathfrak{M}}.$

Note: $e^{\mathcal{M}} = 0$ iff \mathcal{M} is crisp. Try to avoid the case $e^{\mathcal{M}} = \frac{1}{2}$

Fix a language \mathcal{L} with finitely many relation symbols R, \ldots, S .

- $\triangleright \ \varepsilon_{R} := \exists x_{1} \cdots \exists x_{k} (R(\vec{x}) \land \neg R(\vec{x})) \text{ for } k\text{-ary } R$
- \triangleright $\varepsilon_L := \varepsilon_R \lor \cdots \lor \varepsilon_S$ (disjunction over all relation symbols)

Let $\mathcal{M} = (M, R^{\mathcal{M}}, \dots, S^{\mathcal{M}})$ be a fuzzy \mathcal{L} -structure. Define a crisp structure $\bar{\mathcal{M}}$ and a number $e^{\mathcal{M}} \in [0, \frac{1}{2}]$ as follows:

- ▶ The universe \overline{M} is the same as the universe of M: $\overline{M} := M$.
- For each k-ary relation symbol R: For all $\vec{a} \in M^k$: $\bar{\mathcal{M}} \models R(\vec{a})$ iff: $||R(\vec{a})||^{\mathcal{M}} > \frac{1}{2}$
- $e^{\mathfrak{M}} := \|\varepsilon_{\mathfrak{L}}\|^{\mathfrak{M}}.$

Note: $e^{\mathcal{M}} = 0$ iff \mathcal{M} is crisp. Try to avoid the case $e^{M} = \frac{1}{2}$.

Outline

Rasic definition

Complexity results

Computable structures

Proof ingredient

Summary

Summary

Basic definitions

The complexity of the set of valid sentences:

	classical	Łukasiewicz
propositional	co-NP-complete	co-NP-complete
predicate	Σ_1^0 -complete	$Π_2^0$ -complete
computable models	$\mathit{Th}(\mathbb{N})$	$\mathit{Th}(\mathbb{N})$