The complexity of Łukasiewicz Logic

Martin Goldstern

Institute of Discrete Mathematics and Geometry
Vienna University of Technology

May 2013 — ISMVL Toyama

Outline

Outline

Łukasiewicz functions on $[0, 1]$

Definition (Fuzzy operations on $[0, 1]$)

conjunction:	$x \wedge y := \min(x, y)$
disjunction:	$x \vee y := \max(x, y)$
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:	$x \to y := (\neg x) + y = \max\{ z : (x \& z) \leq y \}$

Note: $x \vee y = (x \to y) \to y$, $\neg x = (x \to 0)$, ...

Note: In this talk, fuzzy = Łukasiewicz = Ł.

Łukasiewicz functions on $[0, 1]$

Definition (Fuzzy operations on $[0, 1]$)

$$
\begin{aligned}
\text{conjunction:} \quad & x \wedge y := \min(x, y) \\
\text{disjunction:} \quad & x \vee y := \max(x, y) \\
\text{negation:} \quad & \neg x := 1 - x \\
\text{weak disjunction:} \quad & x + y := \min(x + y, 1) \\
\text{strong conjunction:} \quad & x \& y := \max(x + y - 1, 0) = \neg(\neg x + \neg y) \\
\text{implication:} \quad & x \to y := (\neg x) + y = \max\{ z : (x \& z) \leq y \}
\end{aligned}
$$

Note: $x \vee y = (x \to y) \to y$, $\neg x = (x \to 0)$, ...

Note: In this talk, fuzzy = Łukasiewicz = Ł.

Łukasiewicz functions on $[0, 1]$

Definition (Fuzzy operations on $[0, 1]$)

$$
\begin{aligned}
\text{conjunction:}\quad & x \wedge y := \min(x, y)\\
\text{disjunction:}\quad & x \vee y := \max(x, y)\\
\text{negation:}\quad & \neg x := 1 - x\\
\text{weak disjunction:}\quad & x + y := \min(x + y, 1)\\
\text{strong conjunction:}\quad & x \& y := \max(x + y - 1, 0) = \neg(\neg x + \neg y)\\
\text{implication:}\quad & x \rightarrow y := (\neg x) + y = \max\{\, z : (x \& z) \leq y \,\}
\end{aligned}
$$

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

Note: In this talk, fuzzy = Łukasiewicz = Ł.

Łukasiewicz functions on $[0,1]$

Definition (Fuzzy operations on $[0,1]$)

conjunction:	$x \wedge y := \min(x, y)$
disjunction:	$x \vee y := \max(x, y)$
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:	$x \to y := (\neg x) + y = \max\{ z : (x \& z) \leq y \}$

Note: $x \vee y = (x \to y) \to y$, $\neg x = (x \to 0)$, ...

Note: In this talk, fuzzy = Łukasiewicz = Ł.

Propositional Łukasiewicz logic

Syntax:

- propositional variables $p_1, p_2, \ldots$
- connectives: $+, \&, \vee, \wedge, \neg, \rightarrow, \top, \bot$
- formulas: $p_1 \& p_1 \rightarrow p_2, \ldots$

Semantics:

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

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

Propositional Łukasiewicz logic

Syntax:

- propositional variables $p_1, p_2, \ldots$
- connectives: $+, \&, \vee, \wedge, \neg, \rightarrow, \top, \bot$
- formulas: $p_1 \& p_1 \rightarrow p_2, \ldots$

Semantics:

- Assignments $b : \{p_1, p_2, \ldots\} \rightarrow [0,1]$

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

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

Propositional Łukasiewicz logic

Syntax:

- propositional variables $p_1, p_2, \dots$
- connectives: $+, \&, \vee, \wedge, \neg, \rightarrow, \top, \bot$
- formulas: $p_1 \& p_1 \rightarrow p_2, \dots$

Semantics:

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

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

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

Propositional Łukasiewicz logic

Syntax:

- propositional variables $p_1, p_2, \ldots$
- connectives: $+, \&, \vee, \wedge, \neg, \rightarrow, \top, \bot$
- formulas: $p_1 \& p_1 \rightarrow p_2, \ldots$

Semantics:

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

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

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

Propositional Łukasiewicz logic

Syntax:

- propositional variables $p_1, p_2, \ldots$
- connectives: $+, \&, \vee, \wedge, \neg, \rightarrow, \top, \bot$
- formulas: $p_1 \& p_1 \rightarrow p_2, \ldots$

Semantics:

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

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

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

Propositional Łukasiewicz logic

Syntax:

- propositional variables $p_1, p_2, \dots$
- connectives: $+, \&, \vee, \wedge, \neg, \rightarrow, \top, \bot$
- formulas: $p_1 \& p_1 \rightarrow p_2, \dots$

Semantics:

- Assignments: $b : \{p_1, p_2, \dots\} \rightarrow [0, 1]$
- 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.

Propositional Łukasiewicz logic

Syntax:

- propositional variables $p_1, p_2, \ldots$
- connectives: $+, \&, \vee, \wedge, \neg, \rightarrow, \top, \bot$
- formulas: $p_1 \& p_1 \rightarrow p_2, \ldots$

Semantics:

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

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

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

Propositional Łukasiewicz logic

Syntax:

- propositional variables $p_1, p_2, \ldots$
- connectives: $+, \&, \vee, \wedge, \neg, \rightarrow, \top, \bot$
- formulas: $p_1 \& p_1 \rightarrow p_2, \ldots$

Semantics:

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

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

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

Propositional Łukasiewicz logic

Syntax:

- propositional variables $p_1, p_2, \ldots$
- connectives: $+, \&, \vee, \wedge, \neg, \rightarrow, \top, \bot$
- formulas: $p_1 \& p_1 \rightarrow p_2, \ldots$

Semantics:

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

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

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

Łukasiewicz predicate logic

Syntax:

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

Łukasiewicz predicate logic

Syntax:

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

Łukasiewicz predicate logic

Syntax:

- Language $\mathcal{L}$: Relation symbols $R, \dots, S$ (with arities)
- Object variables x, y, ...
- connectives, quantifiers: $\wedge$, $\forall$, ...
- formulas: e.g. $\forall x \, \exists y \, (R(x, y) \& R(y, y) \to S(y, x))$.

Łukasiewicz predicate logic

Syntax:

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

Łukasiewicz predicate logic

Syntax:

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

Łukasiewicz predicate logic 2

Semantics:

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

- $||\varphi|| := \inf\{||\varphi||^{\mathcal{M}} : \mathcal{M}$ an $\mathcal{L}$-structure$\}$.

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

Łukasiewicz predicate logic 2

Semantics:

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

  $\|\varphi\| = \inf\{\|\varphi\|^{\mathcal{M}} : \mathcal{M}$ an $\mathcal{L}$-structure$\}$.

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

Łukasiewicz predicate logic 2

Semantics:

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

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

Łukasiewicz predicate logic 2

Semantics:

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

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

Łukasiewicz predicate logic 2

Semantics:

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

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

Łukasiewicz predicate logic 2

Semantics:

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

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

Łukasiewicz predicate logic 2

Semantics:

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

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

Łukasiewicz predicate logic 2

Semantics:

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

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

Łukasiewicz predicate logic 2

Semantics:

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

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

Łukasiewicz predicate logic 2

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 $v : \{x, y, \dots\} \to M$
- Fuzzy values of formulas: $\|\varphi\|_v^{\mathcal{M}}$.
  - $\|R(x,y)\|_v^{\mathcal{M}} := R^{\mathcal{M}}(v(x), v(y))$
  - $\|\forall x\ \varphi(x)\| := \inf\{\ \|\varphi\|_{v_{x\mapsto m}}^{\mathcal{M}} : m \in M\ \}$
  - etc.
- $\|\varphi\| := \inf\{\|\varphi\|^{\mathcal{M}} : \mathcal{M}$ an $\mathcal{L}$-structure$\}$.

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

Łukasiewicz predicate logic 2

Semantics:

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

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

Outline

Complexity - propositional

Classical propositional logic on $\{0, 1\}$
The set $\{\varphi : \forall b\ (\bar{b}(\varphi) = 1\}$ of classical (or "crisp") tautologies is

- decidable;
- co-NP-complete. [folklore?]

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

- decidable;

Complexity - propositional

Classical propositional logic on $\{0, 1\}$
The set $\{\varphi : \forall b\ (\bar{b}(\varphi) = 1\}$ of classical (or “crisp”) tautologies is

- decidable;
- co-NP-complete. [folklore?]

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

- decidable;
- co-NP-complete. [illegible]

Complexity - propositional

Classical propositional logic on $\{0, 1\}$
The set $\{\varphi : \forall b\ (\bar{b}(\varphi) = 1\}$ of classical (or "crisp") tautologies is

- decidable;
- co-NP-complete. [folklore?]

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

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

Complexity - propositional

Classical propositional logic on $\{0, 1\}$
The set $\{\varphi : \forall b\ (\bar{b}(\varphi) = 1\}$ of classical (or “crisp”) tautologies is

- decidable;
- co-NP-complete. [folklore?]

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

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

Complexity - propositional

Classical propositional logic on $\{0, 1\}$
The set $\{\varphi : \forall b\,(\bar{b}(\varphi) = 1\}$ of classical (or “crisp”) tautologies is

- decidable;
- co-NP-complete. [folklore?]

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

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

Complexity - first order

Classical first order predicate logic on $\{0, 1\}$:
The set $\{\varphi : \mathcal{M} \models \varphi$ for all crisp $\mathcal{M}\}$ of classical validities is

- not decidable
- computably enumerable (c.e., Σ^0_1)
- in fact: Σ^0_1-complete.

First order Ł-logic on $[0, 1]$:
The set $\{\varphi : \|\varphi\|^{\mathcal{M}} = 1$ for all fuzzy $\mathcal{M}\}$ of Ł-validities is

Complexity - first order

Classical first order predicate logic on $\{0, 1\}$:
The set $\{\varphi : \mathcal{M} \models \varphi$ for all crisp $\mathcal{M}\}$ of classical validities is

- not decidable
- computably enumerable (c.e., Σ^0_1)
- in fact: Σ^0_1-complete.

First order Ł-logic on $[0, 1]$:
The set $\{\varphi : \|\varphi\|^{\mathcal{M}} = 1$ for all fuzzy $\mathcal{M}\}$ of Ł-validities is

- not decidable, not Σ^0_1, not even Σ^0_2 (Scarpellini)

Complexity - first order

Classical first order predicate logic on $\{0, 1\}$:
The set $\{\varphi : \mathcal{M} \models \varphi$ for all crisp $\mathcal{M}\}$ of classical validities is

- not decidable
- computably enumerable (c.e., Σ^0_1)
- in fact: Σ^0_1-complete.

First order Ł-logic on $[0, 1]$:
The set $\{\varphi : \|\varphi\|^{\mathcal{M}} = 1$ for all fuzzy $\mathcal{M}\}$ of Ł-validities is

- not decidable, not Σ^0_1, not even Σ^0_2 (Scarpellini)
- Π^0_2 (Novak-Pavelka)
- Π^0_2-complete (Ragaz; G^*)

Complexity - first order

Classical first order predicate logic on $\{0, 1\}$:
The set $\{\varphi : \mathcal{M} \models \varphi$ for all crisp $\mathcal{M}\}$ of classical validities is

- not decidable
- computably enumerable (c.e., Σ^0_1)
- in fact: Σ^0_1-complete.

First order Ł-logic on $[0, 1]$:
The set $\{\varphi : \|\varphi\|^{\mathcal{M}} = 1$ for all fuzzy $\mathcal{M}\}$ of Ł-validities is

- not decidable, not Σ^0_1, not even Σ^0_2 (Scarpellini)
- Π^0_2 (Novak-Pavelka)
- Π^0_2-complete (Ragaz; G*)

Complexity - first order

Classical first order predicate logic on $\{0, 1\}$:
The set $\{\varphi : \mathcal{M} \models \varphi$ for all crisp $\mathcal{M}\}$ of classical validities is

- not decidable
- computably enumerable (c.e., Σ^0_1)
- in fact: Σ^0_1-complete.

First order Ł-logic on $[0, 1]$:
The set $\{\varphi : \|\varphi\|^{\mathcal{M}} = 1$ for all fuzzy $\mathcal{M}\}$ of Ł-validities is

- not decidable, not Σ^0_1, not even Σ^0_2 (Scarpellini)
- Π^0_2 (Novak-Pavelka)
- Π^0_2-complete (Ragaz; G*)

Complexity - first order

Classical first order predicate logic on $\{0, 1\}$:
The set $\{\varphi : \mathcal{M} \models \varphi$ for all crisp $\mathcal{M}\}$ of classical validities is

- not decidable
- computably enumerable (c.e., Σ^0_1)
- in fact: Σ^0_1-complete.

First order Ł-logic on $[0, 1]$:
The set $\{\varphi : \|\varphi\|^{\mathcal{M}} = 1$ for all fuzzy $\mathcal{M}\}$ of Ł-validities is

- not decidable, not Σ^0_1, not even Σ^0_2 (Scarpellini)
- Π^0_2 (Novak-Pavelka)
- Π^0_2-complete (Ragaz; G*)

Complexity - first order

Classical first order predicate logic on $\{0,1\}$:
The set $\{\varphi : \mathcal{M} \models \varphi$ for all crisp $\mathcal{M}\}$ of classical validities is

- not decidable
- computably enumerable (c.e., Σ^0_1)
- in fact: Σ^0_1-complete.

First order Ł-logic on $[0,1]$:
The set $\{\varphi : \|\varphi\|^{\mathcal{M}} = 1$ for all fuzzy $\mathcal{M}\}$ of Ł-validities is

- not decidable, not Σ^0_1, not even Σ^0_2 (Scarpellini)
- Π^0_2 (Novak-Pavelka)
- Π^0_2-complete (Ragaz; G*)

Outline

arithmetical hierarchy: formulas

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^0_1-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^0_1-formulas: $\forall x_1 \psi$, or $\neg(\Sigma^0_1)$.
- Σ^0_n-formulas: $\exists x_1 \forall x_2 \cdots \overset{\exists}{\forall} x_n\, \psi(\vec{x}, \vec{y})$

Remark

Most arithmetical formulas that appear in practice are Σ^0_n, for small n. ($n = 1, 2, 3$.)
Example: "there are infinitely many twin primes":

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

arithmetical hierarchy: formulas

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

- Σ^0_1-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^0_1-formulas: $\forall x_1 \psi$, or $\neg(\Sigma^0_1)$.
- Σ^0_n-formulas: $\exists x_1 \forall x_2 \cdots \substack{\exists \\ \forall} x_n \, \psi(\vec{x}, \vec{y})$

Remark

Most arithmetical formulas that appear in practice are Σ^0_n, for small n. ($n = 1, 2, 3$.)
Example: "there are infinitely many twin primes":

$$\forall x \exists p \, (\, p > x, \; p \text{ prime}, \; p + 2 \text{ prime} \,).$$

arithmetical hierarchy: formulas

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

- Σ^0_1-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^0_1-formulas: $\forall x_1 \psi$, or $\neg(\Sigma^0_1)$.
- Σ^0_n-formulas: $\exists x_1 \forall x_2 \cdots \overset{\exists}{\forall} x_n \, \psi(\vec{x}, \vec{y})$

Remark

Most arithmetical formulas that appear in practice are Σ^0_n, for small n. ($n = 1, 2, 3$.)
Example: "there are infinitely many twin primes":

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

arithmetical hierarchy: formulas

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

- Σ^0_1-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^0_1-formulas: $\forall x_1 \psi$, or $\neg(\Sigma^0_1)$.
- Σ^0_n-formulas: $\exists x_1 \forall x_2 \cdots \genfrac{}{}{0pt}{}{\exists}{\forall} x_n \psi(\vec{x}, \vec{y})$

Remark

Most arithmetical formulas that appear in practice are Σ^0_n, for small n. ($n = 1, 2, 3$.)
Example: "there are infinitely many twin primes":
$$\forall x \exists p\, (\, p > x,\ p \text{ prime},\ p + 2 \text{ prime}\,).$$

arithmetical hierarchy: formulas

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

- Σ^0_1-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^0_1-formulas: $\forall x_1 \psi$, or $\neg(\Sigma^0_1)$.
- Σ^0_n-formulas: $\exists x_1 \, \forall x_2 \cdots \substack{\exists \\ \forall} x_n \, \psi(\vec{x}, \vec{y})$

Remark

Most arithmetical formulas that appear in practice are Σ^0_n, for small n. ($n = 1, 2, 3$.)
Example: "there are infinitely many twin primes":
$$\forall x \exists p \, (\, p > x, \; p \text{ prime}, \; p + 2 \text{ prime} \,).$$

arithmetical hierarchy: formulas

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

- Σ^0_1-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^0_1-formulas: $\forall x_1 \psi$, or $\neg(\Sigma^0_1)$.
- Σ^0_n-formulas: $\exists x_1 \, \forall x_2 \, \cdots \, \substack{\exists \\ \forall} x_n \, \psi(\vec{x}, \vec{y})$

Remark

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

Example: “there are infinitely many twin primes”:

$\forall x \exists p \, (\, p > x, \; p \text{ prime}, \; p + 2 \text{ prime} \,).$

arithmetical hierarchy: formulas

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

- Σ^0_1-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^0_1-formulas: $\forall x_1 \psi$, or $\neg(\Sigma^0_1)$.
- Σ^0_n-formulas: $\exists x_1 \forall x_2 \cdots \substack{\exists \\ \forall} x_n \, \psi(\vec{x}, \vec{y})$

Remark

Most arithmetical formulas that appear in practice are Σ^0_n, for small n. ($n = 1, 2, 3$.)
Example: "there are infinitely many twin primes":
$\forall x \exists p \, (\, p > x,\; p \text{ prime},\; p + 2 \text{ prime}\,)$.

arithmetical hierarchy: sets

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

- The Σ^0_1 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^0_1 and Π^0_1.
- $\Sigma^0_1 \subsetneq \Sigma^0_2 \subsetneq \cdots$, similarly $\Pi^0_1 \subsetneq \Pi^0_2 \subsetneq \cdots$.
- If C is Π^0_n, and f is computable, then $f^{-1}(C)$ is also Π^0_n.
- C is a complete Π^0_n-set, if C is Π^0_n, and every Π^0_n-set B can be reduced to C, i.e., is of the form $f^{-1}(C)$, for some computable f.

arithmetical hierarchy: sets

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

- The Σ_1^0 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.
- $\Sigma_1^0 \subsetneq \Sigma_2^0 \subsetneq \cdots$, 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.
- 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

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

- The Σ^0_1 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^0_1 and Π^0_1.
- $\Sigma^0_1 \subsetneq \Sigma^0_2 \subsetneq \cdots$, similarly $\Pi^0_1 \subsetneq \Pi^0_2 \subsetneq \cdots$.
- If C is Π^0_n, and f is computable, then $f^{-1}(C)$ is also Π^0_n.
- C is a complete Π^0_n-set, if C is Π^0_n, and every Π^0_n-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^0_n sets, similar to co-NP-complete)

arithmetical hierarchy: sets

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

- The Σ^0_1 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^0_1 and Π^0_1.
- $\Sigma^0_1 \subsetneq \Sigma^0_2 \subsetneq \cdots$, similarly $\Pi^0_1 \subsetneq \Pi^0_2 \subsetneq \cdots$.
- If C is Π^0_n, and f is computable, then $f^{-1}(C)$ is also Π^0_n.
- C is a complete Π^0_n-set, if C is Π^0_n, and every Π^0_n-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^0_n sets, similar to co-NP-complete)

arithmetical hierarchy: sets

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

- The Σ_1^0 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.
- $\Sigma_1^0 \subsetneq \Sigma_2^0 \subsetneq \cdots$, 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.
- 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)

arithmetical hierarchy: sets

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

- The Σ^0_1 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^0_1 and Π^0_1.
- $\Sigma^0_1 \subsetneq \Sigma^0_2 \subsetneq \cdots$, similarly $\Pi^0_1 \subsetneq \Pi^0_2 \subsetneq \cdots$.
- If C is Π^0_n, and f is computable, then $f^{-1}(C)$ is also Π^0_n.
- C is a complete Π^0_n-set, if C is Π^0_n, and every Π^0_n-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^0_n sets, similar to co-NP-complete)

arithmetical hierarchy: sets

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

- The Σ^0_1 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^0_1 and Π^0_1.
- $\Sigma^0_1 \subsetneq \Sigma^0_2 \subsetneq \cdots$, similarly $\Pi^0_1 \subsetneq \Pi^0_2 \subsetneq \cdots$.
- If C is Π^0_n, and f is computable, then $f^{-1}(C)$ is also Π^0_n.
- C is a complete Π^0_n-set, if C is Π^0_n, and every Π^0_n-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^0_n 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_1 complete.
- The set of all (codes for) programs that describe a function with infinite domain is Π^0_2-complete.
- The set $Th_n(\mathbb{N})$ of all (codes for) true Σ^0_n-formulas is Σ^0_n-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

- The set of all (codes for) Turing machines that halt on input 0 is Σ^0_1 complete.
- The set of all (codes for) programs that describe a function with infinite domain is Π^0_2-complete.
- The set $Th_n(\mathbb{N})$ of all (codes for) true Σ^0_n-formulas is Σ^0_n-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

- The set of all (codes for) Turing machines that halt on input 0 is Σ^0_1 complete.
- The set of all (codes for) programs that describe a function with infinite domain is Π^0_2-complete.
- The set $Th_n(\mathbb{N})$ of all (codes for) true Σ^0_n-formulas is Σ^0_n-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

- The set of all (codes for) Turing machines that halt on input 0 is Σ^0_1 complete.
- The set of all (codes for) programs that describe a function with infinite domain is Π^0_2-complete.
- The set $Th_n(\mathbb{N})$ of all (codes for) true Σ^0_n-formulas is Σ^0_n-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

- The set of all (codes for) Turing machines that halt on input 0 is Σ^0_1 complete.
- The set of all (codes for) programs that describe a function with infinite domain is Π^0_2-complete.
- The set $Th_n(\mathbb{N})$ of all (codes for) true Σ^0_n-formulas is Σ^0_n-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

- The set of all (codes for) Turing machines that halt on input 0 is Σ^0_1 complete.
- The set of all (codes for) programs that describe a function with infinite domain is Π^0_2-complete.
- The set $Th_n(\mathbb{N})$ of all (codes for) true Σ^0_n-formulas is Σ^0_n-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

- The set of all (codes for) Turing machines that halt on input 0 is Σ^0_1 complete.
- The set of all (codes for) programs that describe a function with infinite domain is Π^0_2-complete.
- The set $Th_n(\mathbb{N})$ of all (codes for) true Σ^0_n-formulas is Σ^0_n-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 $Th_n(\mathbb{N})$.

Outline

Computable structures

Definition

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

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

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

- M is a decidable subset of $\mathbb{N}$,

Computable structures

Definition

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

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

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

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

Computable structures

Definition

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

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

A fuzzy structure $\mathcal{M} = (M, R^{\mathcal{M}}, \ldots, S^{\mathcal{M}})$ 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 $\mathbb{N}^k \times \mathbb{Q}$.

Computable structures

Definition

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

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

A fuzzy structure $\mathcal{M} = (M, R^{\mathcal{M}}, \ldots, S^{\mathcal{M}})$ 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 $\mathbb{N}^k \times \mathbb{Q}$.

Computable structures

Definition

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

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

A fuzzy structure $\mathcal{M} = (M, R^{\mathcal{M}}, \ldots, S^{\mathcal{M}})$ 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 $\mathbb{N}^k \times \mathbb{Q}$.

Computably valid sentences

Recall

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

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 $\mathcal{M}$.

Theorem

Computably valid sentences

Recall

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

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 $\mathcal{M}$.

Theorem

Computably valid sentences

Recall

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

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 $\mathcal{M}$.

Theorem

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

Computably valid sentences

Recall

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

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 $\mathcal{M}$.

Theorem

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

Computably valid sentences

Recall

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

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 $\mathcal{M}$.

Theorem

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

Computably valid sentences

Recall

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

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 $\mathcal{M}$.

Theorem

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

Computably valid sentences

Recall

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

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 $\mathcal{M}$.

Theorem

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

Outline

Tennenbaum's theorem

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.*

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

Tennenbaum's theorem

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-Ł-validities.*

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

Tennenbaum's theorem

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-Ł-validities.*

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

Tennenbaum's theorem

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-Ł-validities.*

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

Tennenbaum's theorem

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-Ł-validities.*

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

Tennenbaum's theorem

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-Ł-validities.*

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

Tennenbaum's theorem

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-Ł-validities.*

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

From fuzzy to crisp via rounding

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

- $\varepsilon_R := \exists x_1 \cdots \exists x_k \, (R(\vec{x}) \wedge \neg R(\vec{x}))$ for k-ary R
- $\varepsilon_{\mathcal{L}} := \varepsilon_R \vee \cdots \vee \varepsilon_S$ (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 $\bar{\mathcal{M}}$ is the same as the universe of $\mathcal{M}$: $\bar{M} := M$.
- For each k-ary relation symbol R:
  For all $\bar{a} \in M^k$: $\bar{\mathcal{M}} \models R(\bar{a})$ iff: $\|R(\bar{a})\|^{\mathcal{M}} > \frac{1}{2}$

  $e^{\mathcal{M}} := \|\varepsilon_{\mathcal{L}}\|^{\mathcal{M}}$.

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

From fuzzy to crisp via rounding

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

- $\varepsilon_R := \exists x_1 \cdots \exists x_k \, (R(\vec{x}) \wedge \neg R(\vec{x}))$ for k-ary R
- $\varepsilon_{\mathcal{L}} := \varepsilon_R \vee \cdots \vee \varepsilon_S$ (disjunction over all relation symbols)

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

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

From fuzzy to crisp via rounding

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

- $\varepsilon_R := \exists x_1 \cdots \exists x_k \, (R(\vec{x}) \wedge \neg R(\vec{x}))$ for k-ary R
- $\varepsilon_{\mathcal{L}} := \varepsilon_R \vee \cdots \vee \varepsilon_S$ (disjunction over all relation symbols)

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

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

From fuzzy to crisp via rounding

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

- $\varepsilon_R := \exists x_1 \cdots \exists x_k \, (R(\vec{x}) \wedge \neg R(\vec{x}))$ for k-ary R
- $\varepsilon_{\mathcal{L}} := \varepsilon_R \vee \cdots \vee \varepsilon_S$ (disjunction over all relation symbols)

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

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

From fuzzy to crisp via rounding

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

- $\varepsilon_R := \exists x_1 \cdots \exists x_k \, (R(\vec{x}) \wedge \neg R(\vec{x}))$ for k-ary R
- $\varepsilon_{\mathcal{L}} := \varepsilon_R \vee \cdots \vee \varepsilon_S$ (disjunction over all relation symbols)

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

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

From fuzzy to crisp via rounding

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

- $\varepsilon_R := \exists x_1 \cdots \exists x_k \, (R(\vec{x}) \wedge \neg R(\vec{x}))$ for k-ary R
- $\varepsilon_{\mathcal{L}} := \varepsilon_R \vee \cdots \vee \varepsilon_S$ (disjunction over all relation symbols)

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

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

From fuzzy to crisp via rounding

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

- $\varepsilon_R := \exists x_1 \cdots \exists x_k \, (R(\vec{x}) \wedge \neg R(\vec{x}))$ for k-ary R
- $\varepsilon_{\mathcal{L}} := \varepsilon_R \vee \cdots \vee \varepsilon_S$ (disjunction over all relation symbols)

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

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

Outline

Summary

The complexity of the set of valid sentences:

	classical	**Łukasiewicz**
propositional	co-NP-complete	co-NP-complete
predicate	Σ^0_1-complete	Π^0_2-complete
computable models	$Th(\mathbb{N})$	$Th(\mathbb{N})$