Computational Logic

Automated Reasoning

Georg Moser

Institute of Computer Science @ UIBK

Winter 2013

Summary Last Lecture

Selection of Applications

- Program Analysis; logical product of abstract interpreters
- Databases; disjunctive datalog
- Programming Languages; types as formulas
- Computational Complexity; implicit complexity

Lessons Learnt

- (mathematical) logic is the science of (mathematical) reasoning
- logic has been and is very successfully used as workbench for various areas in computer science
- applications are not trivial (in both senses)

Outline of the Lecture

Propositional Logic

short reminder of propositional logic, soundness and completeness theorem, natural deduction, propositional resolution

First Order Logic

introduction, syntax, semantics, Löwenheim-Skolem, compactness, model existence theorem, natural deduction, completeness, normalisation

Properties of First Order Logic

Craig's Interpolation Theorem, Robinson's Joint Consistency Theorem, Herbrand's Theorem

Limits and Extensions of First Order Logic

Intuitionistic Logic, Curry-Howard Isomorphism, Limits, Second-Order Logic

First-Order Logic

The Language of First-Order Logic

a first-order language is determined by specifying its constants, variables, logical symbols, auxiliary (brackets, comma)

Definition
- individual constants: $k_0, k_1, \dots, k_j, \dots$ denoted c, d, etc.
- function constants with i arguments: $f_0^i, f_1^i, \dots$ denoted f, g, h etc.
- predicate constants with i arguments: $R_0^i, R_1^i, \dots$ denoted as P, Q, R, etc.

Definition
- variables: $x_0, x_1, \dots, x_j, \dots$ denoted x, y, z, etc.

Definition
- propositional connectives: $\neg$, $\wedge$, $\vee$, $\rightarrow$
- quantifiers $\forall$, $\exists$
- equality sign $=$

the equality sign $=$ is a predicate but treated like a logical symbol

Definition
if the cardinality of the set of constants in $\mathcal{L}$ is countable, we say $\mathcal{L}$ is countable

Example
the language of arithmetic $\mathcal{L}_{\text{arith}}$ contains $=$ and consists of
- individual constant 0
- function constants s, $+$, $\cdot$
- predicate constant $<$

Terms of a Language

Definition
terms (of $\mathcal{L}$) are defined as follows
- any individual constant c in $\mathcal{L}$ is a term
- any variable x is a term
- if $t_1, \dots, t_n$ are terms, f an n-ary function constant in $\mathcal{L}$, then $f(t_1, \dots, t_n)$ is a term

Example
- $\mathsf{s}(\mathsf{s}(\mathsf{s}(0)))$ is a term (of $\mathcal{L}_{\text{arith}}$)
- $\mathsf{s}(x)$ is a term (of $\mathcal{L}_{\text{arith}}$)

Convention
if the language $\mathcal{L}$ is clear from context the phrase "of $\mathcal{L}$" will be dropped

Formulas (of a Language)

Definition
- $P(t_1, \dots, t_n)$ is an atomic formula; P a constant of arity n, t_i terms
- $t_1 = t_2$ is an atomic formula, if $=$ is present

Definition
formulas are defined as follows
- atomic formulas are formulas
- A and B are frms: $(\neg A)$, $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$ are formulas
- if A a formula, x a variable, then
$$\forall x A \qquad \exists x A$$
are formulas

Convention

if brackets are not necessary they are omitted:

$$\exists, \forall \; > \; \neg \; > \; \vee, \wedge \; > \; \to \qquad \text{right-associativity of } \to$$

Example

consider $\mathcal{L}_{\text{arith}}$, which of the following are formulas over $\mathcal{L}_{\text{arith}}$?

- $x < y \wedge \neg \exists z (x < z \wedge z < y)$ ✓
- $\forall x (x = 0) \to \exists x (x = 0)$ ✓
- $\forall x (x < y \wedge \exists x (y = x))$ ✓

The Semantics of First-Order Logic

Definition

a structure is a pair $\mathcal{A} = (A, a)$ such that:

- A is a non-empty set, A is called domain
- mapping a associates constants with the domain:
  - any individual constant c is associated with an element $a(c) \in A$.
  - any n-ary function constant f is associated with $a(f) \colon A^n \to A$.
  - any n-ary predicate constants P is associated with a subset $a(P) \subseteq A^n$.
- equality sign $=$ is associated with the identity relation $a(=)$.

we write $c^{\mathcal{A}}$. $f^{\mathcal{A}}$, and $P^{\mathcal{A}}$, instead of $a(c)$, $a(f)$, and $a(P)$; for brevity we write $=$ for the equality sign and the identity relation

Remark

a structure is equivalent to the definition of model in LICS

Definition

- an environment for $\mathcal{A}$ is a mapping $\ell \colon \{x_n \mid n \in \mathbb{N}\} \to A$
- $\ell\{x \mapsto t\}$ denotes the environment mapping x to t and all other variables $y \neq x$ to $\ell(y)$

Definition

an interpretation $\mathcal{I}$ is a pair $(\mathcal{A}, \ell)$ such that

- $\mathcal{A}$ is a structure
- ℓ is an environment

Definition

the value of a term t (wrt interpretation $\mathcal{I}$)

$$t^{\mathcal{I}} = \begin{cases} \ell(t) & \text{if } t \text{ a variable} \\ c^{\mathcal{A}} & \text{if } t = c \\ f^{\mathcal{A}}(t_1^{\mathcal{I}}, \dots, t_n^{\mathcal{I}}) & \text{if } t = f(t_1, \dots, t_n),\ n \geqslant 1 \end{cases}$$

Definition (The Satisfaction Relation)

$\mathcal{I} = (\mathcal{A}, \ell)$ an interpretation; F a formula, we define $\mathcal{I} \models F$

$$\begin{aligned}
\mathcal{I} &\models t_1 = t_2 & &:\Longleftrightarrow \text{ if } t_1^{\mathcal{I}} = t_2^{\mathcal{I}} \\
\mathcal{I} &\models P(t_1, \dots, t_n) & &:\Longleftrightarrow \text{ if } (t_1^{\mathcal{I}}, \dots, t_n^{\mathcal{I}}) \in P^{\mathcal{A}} \\
\mathcal{I} &\models \neg F & &:\Longleftrightarrow \text{ if } \mathcal{I} \not\models F \\
\mathcal{I} &\models F \wedge G & &:\Longleftrightarrow \text{ if } \mathcal{I} \models F \text{ and } \mathcal{I} \models G \\
\mathcal{I} &\models F \vee G & &:\Longleftrightarrow \text{ if } \mathcal{I} \models F \text{ or } \mathcal{I} \models G \\
\mathcal{I} &\models F \to G & &:\Longleftrightarrow \text{ if } \mathcal{I} \models F, \text{ then } \mathcal{I} \models G \\
\mathcal{I} &\models \forall x F & &:\Longleftrightarrow \text{ if } \mathcal{I}\{x \mapsto a\} \models F \text{ holds for all } a \in A \\
\mathcal{I} &\models \exists x F & &:\Longleftrightarrow \text{ if } \mathcal{I}\{x \mapsto a\} \models F \text{ holds for some } a \in A
\end{aligned}$$

let $\mathcal{G}$ be a set of formulas

- $\mathcal{I} \models \mathcal{G}$, if $\mathcal{I} \models F$ for all $F \in \mathcal{G}$
- $\mathcal{I}$ models $\mathcal{G}$, if $\mathcal{I} \models \mathcal{G}$

Definition

F, $G_1, \dots, G_n$ be formulas

- $G_1, \dots, G_n \models F$ iff
  $\forall$ interpretations $\mathcal{I}$ of all $G_1, \dots, G_n$ such that
  $\mathcal{I}$ models $G_1, \dots, G_n$, we have $\mathcal{I}$ models F
- F is called satisfiable
  if $\exists$ an interpretation that is a model of F — $\mathsf{Sat}(F)$
- F is valid if
  F is satisfiable in any interpretation — $\models F$

Example

consider the formula $A := x < y \land \neg\exists z(x < z \land z < y)$

- $\mathcal{N} = (\mathbb{N}, 0, \mathsf{succ}, +, \cdot, <)$ denote the standard structure of arithmetic
- $\ell(x) = 1, \ell(y) = 2$

then $(\mathcal{N}, \ell) \models A$

Example

consider concatenation of lists

$$\mathsf{app}(\mathsf{nil}, Ys) = Ys \qquad \mathsf{app}(Xs, Ys) = \mathsf{cons}(\mathsf{head}(Xs), \mathsf{app}(\mathsf{tail}(Xs), Ys))$$

and the language $\mathcal{L}$:

$$\mathsf{nil},\ \ \mathsf{head}(Xs),\ \ \mathsf{tail}(Xs),\ \ \mathsf{cons}(X, Xs),\ \ =,\ \text{ and }\ \mathsf{App}(Xs, Ys, Zs)$$

then list concatenation is expressible as follows:

$$\forall x\ \mathsf{App}(\mathsf{nil}, x, x) \land$$
$$\land\ \forall x \forall y \forall z\, (x \neq \mathsf{nil} \land \mathsf{App}(\mathsf{tail}(x), y, z) \to \mathsf{App}(x, y, \mathsf{cons}(\mathsf{head}(x), z)))$$

Example

define

$$I_n := \forall x_1 \dots \forall x_{n-1} \exists y (x_1 \neq y \land \cdots \land x_{n-1} \neq y)$$

if $\mathcal{I} \models I_n$, then $\mathcal{I}$ has at least n elements

Definition

let F be a formula such that x occurs in F

- x is bound if it occurs inside the scope of a quantifier
- otherwise x is free
- a formula without free variables is called closed or a sentence

Example

consider $\forall x(\mathsf{P}(x) \land \mathsf{Q}(x, y))$; then x is bound and y is free

Notation

let F be a formula, x a free variable in F, t a term

- we sometimes write $F(x)$ instead of F to indicate x
- $F(t)$ denotes the replacement of x by t
- $F(t)$ is an instance of $F(x)$

Definition

F is called unsatisfiable
if $\neg\ \exists$ interpretation that is a model of F — $\neg\,\mathsf{Sat}(F)$

Definition

F and G are logically equivalent if $F \models G$ and $G \models F$ — $F \equiv G$

Lemma

$\forall$ formulas F and all sets of formulas $\mathcal{G}$: $\mathcal{G} \models F$ iff $\neg\,\mathsf{Sat}(\mathcal{G} \cup \{\neg F\})$

Lemma

1. *let $\mathcal{I}_1 = (\mathcal{A}_1, \ell_1)$ and $\mathcal{I}_2 = (\mathcal{A}_2, \ell_2)$ be interpretations*
2. *the universes of $\mathcal{I}_1$, $\mathcal{I}_2$ coincide*
3. *$\mathcal{I}_1$ and $\mathcal{I}_2$ coincide on the constants and variables occurring in F*

then $\mathcal{I}_1 \models F$ iff $\mathcal{I}_2 \models F$

Toy Example: Logic as Modelling Language

Argument ①

1. a mother or father of a person is an ancestor of that person
2. an ancestor of an ancestor of a person is an ancestor of a person
3. Sarah is the mother of Isaac, Isaac is the father of Jacob
4. Thus, Sarah is an ancestor of Jacob

Argument ②

1. a square or cube of a number is a power of that number
2. a power of a power of a number is a power of that number
3. 64 is the cube of 4, four is the square of 2
4. Thus, 64 is a power of 2

A Bit of History

Fact

the idea of automated reasoning is (very) old

Gottfried Leibniz (1646–1716) proposed the idea of

- lingua characteristica
  a universal language, able to express all concepts
- calculus ratiocinator
  a machine to "compute" whether a given argument is sound

we already know that a 'calculus ratiocinator' cannot exist

Undecidability of First-Order Logic

Theorem

1. *the decision problem for the consequence relation is undecidable*
2. *the set of valid first-order formulas is not recursive*

Proof Ideas.

- encoding of TMs as first-order formulas
- reduction from Post correspondence problem ∎

Theorem

the set of valid first-order formulas (over a countable language) is recursive enumerable

Proof.

- the set of all formulas (over a countable language) is countable
- completeness yields that one can enumerate all valid formulas ∎

Outline of the Lecture

Propositional Logic

short reminder of propositional logic, soundness and completeness theorem, natural deduction, propositional resolution

First Order Logic

introduction, syntax, semantics, Löwenheim-Skolem, compactness, model existence theorem, natural deduction, completeness, normalisation

Properties of First Order Logic

Craig's Interpolation Theorem, Robinson's Joint Consistency Theorem, Herbrand's Theorem

Limits and Extensions of First Order Logic

Intuitionistic Logic, Curry-Howard Isomorphism, Limits, Second-Order Logic

Definition

$\mathcal{A}$, $\mathcal{B}$ two structures over the same language; assume $\exists$ bijection $m\colon A \to B$ such that

1. $\forall$ individual constant c: $m(c^{\mathcal{A}}) = c^{\mathcal{B}}$
2. $\forall$ function constant f, $\forall$ $a_1, \ldots, a_n \in A$:
$$m(f^{\mathcal{A}}(a_1, \ldots, a_n)) = f^{\mathcal{B}}(m(a_1), \ldots, m(a_n)) \quad \text{and}$$
3. $\forall$ predicate constant P, $\forall$ $a_1, \ldots, a_n \in A$:
$$P^{\mathcal{A}}(a_1, \ldots, a_n) \iff P^{\mathcal{B}}(m(a_1), \ldots, m(a_n))$$

then m is called an isomorphism between $\mathcal{A}$ and $\mathcal{B}$ denoted $m\colon \mathcal{A} \cong \mathcal{B}$

Lemma

let A, B be sets; $m\colon A \to B$ be a bijection; if $\mathcal{A}$ is a structure with domain A, then $\exists$ structure $\mathcal{B}$ with $\mathcal{A} \cong \mathcal{B}$

Isomorphism Theorem

let $\mathcal{A}$, $\mathcal{B}$ be structures such that $m\colon \mathcal{A} \cong \mathcal{B}$, then for all sentences F: $\mathcal{A} \models F$ iff $\mathcal{B} \models F$

Proof.

1. let $\mathcal{I} = (\mathcal{A}, \ell)$, define $\ell^m = m \circ \ell$, set $\mathcal{J} = (\mathcal{B}, \ell^m)$
2. $\forall$ terms t: $m(t^{\mathcal{I}}) = t^{\mathcal{J}}$ (follows by induction on t)
3. $\forall$ formulas F: $\mathcal{I} \models F \iff \mathcal{J} \models F$
   - base case $F = (s = t)$
   $$\mathcal{I} \models s = t \iff s^{\mathcal{I}} = t^{\mathcal{I}} \iff m(s^{\mathcal{I}}) = m(t^{\mathcal{I}}) \iff \mathcal{J} \models s = t$$
   - step case $F = \exists x G$
   $$\begin{aligned} \mathcal{I} \models \exists x G &\iff \text{there exists } a \in A,\ \mathcal{I}\{x \mapsto a\} \models G \\ &\iff \text{there exists } a \in A,\ \mathcal{J}\{x \mapsto m(a)\} \models G \\ &\iff \text{there exists } b \in B,\ \mathcal{J}\{x \mapsto b\} \models G \\ &\iff \mathcal{J} \models \exists x G \end{aligned}$$

∎

Corollary

1. *$\forall$ formula F that has a finite model has a model in the domain $\{0, 1, 2, \ldots, n\}$*
2. *$\forall$ formula F that has a countable infinite model has a model whose domain is $\mathbb{N}$*

Proof.

combination of both lemmas ∎

Example

consider $\mathcal{L} = \{\fallingdotseq\}$ and
$$\begin{aligned} E &:\iff \forall x\, x \fallingdotseq x \land \forall x \forall y\, (x \fallingdotseq y \land y \fallingdotseq x) \land \\ &\qquad \forall x \forall y \forall z\, ((x \fallingdotseq y \land y \fallingdotseq z) \to x \fallingdotseq z) \\ F &:\iff \forall x \forall y\, x \fallingdotseq y \end{aligned}$$

if $\mathcal{M}$ and $\mathbb{N}$ are countable infinite and $\mathcal{M} \models E \land F$, $\mathcal{N} \models E \land F$, then $\mathcal{M} \cong \mathcal{N}$

Compactness and Löwenheim-Skolem

Theorem (Compactness Theorem)

if every finite subset of a set of formulas $\mathcal{G}$ has a model, then $\mathcal{G}$ has a model

Theorem (Löwenheim-Skolem Theorem)

if a set of formulas $\mathcal{G}$ has a model, then $\mathcal{G}$ has a countable model

Corollary

if a set of formulas $\mathcal{G}$ has arbitrarily large finite models, then it has a countable infinite model

Proof Idea.

employ compactness to show that $\mathcal{G}$ has an infinite model and Löwenheim-Skolem to show that this model is countable ∎

Corollary

1. *any satisfiable set of formulas $\mathcal{G}$ has a model whose domain is either the set of natural numbers $< n$ or $\mathbb{N}$*
2. *if $\mathcal{G}$ is a satisfiable set of formulas, no function symbols, no identity in language, then $\mathcal{G}$ has a model whose domain is $\mathbb{N}$*

Proof (of second item).

1. suppose $\mathcal{G}$ has a model $\mathcal{I}$ with domain $\{0, \ldots, n-1\}$
2. define $f \colon \mathbb{N} \to \{0, 1, \ldots, n-1\}$ as:
$$f(m) = \min\{m, n-1\}$$
3. define $\mathcal{J}$ with domain $\mathbb{N}$ and look-up table ℓ
   - $f(c^{\mathcal{J}}) = c^{\mathcal{I}}$
   - $\forall$ predicate constants P, $\forall\ n_1, \ldots, n_k$
     $(n_1, \ldots, n_k) \in P^{\mathcal{J}}$ iff $(f(n_1), \ldots, f(n_k)) \in P^{\mathcal{I}}$
4. f is a surjective homomorphism, the proof of the isomorphism lemma holds ∎

Proof Plan for Completeness

first-order logic features the following three theorems

1. (soundness and) completeness
2. compactness
3. Löwenheim-Skolem

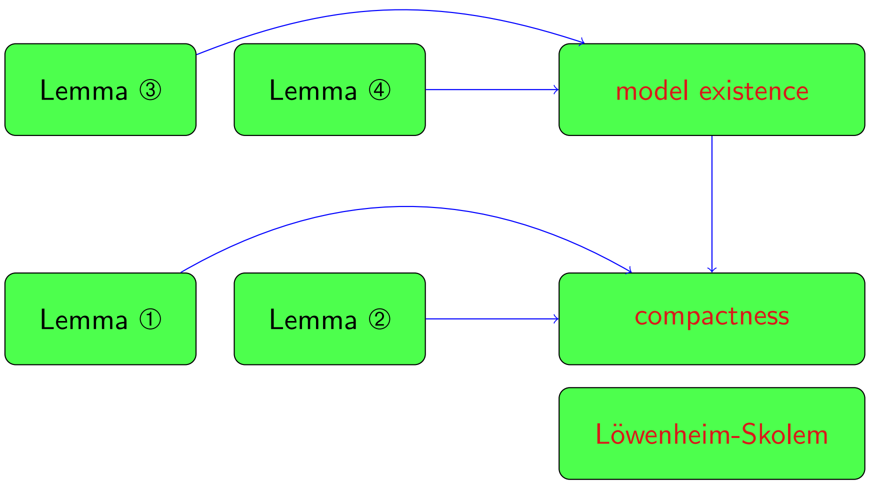
Observations

⊥ is not derivable

- any proof of completeness is indirect:
  suppose $\exists$ a consistent set $\mathcal{G}$, then $\mathcal{G}$ is satisfiable
- to show $\mathcal{G}$ is satisfiable one constructs a countable model $\mathcal{M}$
- Löwenheim-Skolem and compactness follow
- the central piece of work is the construction of $\mathcal{M}$; this is independent on the proof system

in proof, we restrict the logical symbols to $\neg$, $\vee$, $\exists$, and $=$

Howto Prove Compactness and Löwenheim-Skolem



Lemma ①

let S be the set of satisfiable sets of formulas; pick $\mathcal{G} \in S$

1. *if $\mathcal{G}_0 \subseteq \mathcal{G}$, then $\mathcal{G}_0 \in S$*
2. *no formula F and $\neg F$ in $\mathcal{G}$*
3. *if $\neg\neg F \in \mathcal{G}$, then $\mathcal{G} \cup \{F\} \in S$*
4. *if $(E \vee F) \in \mathcal{G}$, then $\mathcal{G} \cup \{E\} \in S$ or $\mathcal{G} \cup \{F\} \in S$*
5. *if $\neg(E \vee F) \in \mathcal{G}$, then $\mathcal{G} \cup \{\neg E\} \in S$ and $\mathcal{G} \cup \{\neg F\} \in S$*
6. *if $\exists x F(x) \in \mathcal{G}$, the constant c doesn't occur in $\mathcal{G}$, then $\mathcal{G} \cup \{F(c)\} \in S$*
7. *if $\neg\exists x F(x) \in \mathcal{G}$, then $\forall$ terms t, $\mathcal{G} \cup \{\neg F(t)\} \in S$*
8. *for any term t, $\mathcal{G} \cup \{t = t\} \in S$*
9. *if $\{F(s), s = t\} \subseteq \mathcal{G}$, then $\mathcal{G} \cup \{F(t)\} \in S$*

Definition

we call the properties (of S) in the lemma satisfaction properties

Lemma ②

1. *assume S is a set of formula sets and S has the satisfaction properties*
2. *let S^* be the set of all formula sets $\mathcal{G}$ such that $\forall$ finite $\mathcal{G}_0 \subseteq \mathcal{G}$, $\mathcal{G}_0 \in S$*
3. *then S^* has the satisfaction properties*

Proof.

we treat the case of disjunction

- assume $\mathcal{G} \in S^*$, $(E \vee F) \in \mathcal{G}$, $\mathcal{G} \cup \{E\} \notin S^*$ and $\mathcal{G} \cup \{F\} \notin S^*$
- $\forall$ finite $\mathcal{G}_0 \subseteq \mathcal{G}$, $\mathcal{G}_0 \in S$,
- $\exists$ finite $\mathcal{G}_1 \subseteq \mathcal{G} \cup \{E\}$, $\mathcal{G}_1 \notin S$, $\exists$ finite $\mathcal{G}_2 \subseteq \mathcal{G} \cup \{F\}$, $\mathcal{G}_2 \notin S$
- wlog $\mathcal{G}_1 = \mathcal{G}_1' \cup \{E\}$, $\mathcal{G}_2 = \mathcal{G}_2' \cup \{F\}$, and $\mathcal{G}_1', \mathcal{G}_2' \subseteq \mathcal{G}$ finite
- $\mathcal{G}_1' \cup \mathcal{G}_2' \cup \{(E \vee F)\} \subseteq \mathcal{G}$, hence $\mathcal{G}_1' \cup \mathcal{G}_2' \cup \{(E \vee F)\} \in S$
- hence $\mathcal{G}_1' \cup \mathcal{G}_2' \cup \{E\} \in S$ or $\mathcal{G}_1' \cup \mathcal{G}_2' \cup \{F\} \in S$
- contradiction ∎

Compactness and Löwenheim-Skolem Theorem

$\mathcal{L}$ base language; $\mathcal{L}^+ \supseteq \mathcal{L}$ infinitely many new individual constants

Theorem (Model Existence Theorem)

1. *if S^* is a set of formula sets of $\mathcal{L}^+$ having the satisfaction properties, then $\forall$ formula sets $\mathcal{G} \in S^*$ of $\mathcal{L}$, $\exists\, \mathcal{M}$, $\mathcal{M} \models \mathcal{G}$*
2. *$\forall$ elements m of $\mathcal{M}$: m denotes term in $\mathcal{L}^+$*

Compactness Theorem

if every finite subset of a set of formulas $\mathcal{G}$ has a model, then $\mathcal{G}$ has a model

Remark

the statement and the proof of the compactness theorem do not refer to provability; compactness is extensible to non-enumerable language

Proof (of compactness).

- consider the set S of satisfiable formula sets (over $\mathcal{L}$) (as in Lemma ①)
- consider the set S^* of all formulas set $\mathcal{G}$, $\forall\, \mathcal{G}_0 \subseteq \mathcal{G}$, $\mathcal{G}_0$ finite, $\mathcal{G}_0 \in S$ (as in Lemma ②)
- Lemma ① yields that S admits the satisfaction properties
- Lemma ② yields that S^* admits the satisfaction properties
- by assumption $\mathcal{G}$ is in S^*
- by model existence $\mathcal{G}$ has a model $\mathcal{M}$ ∎

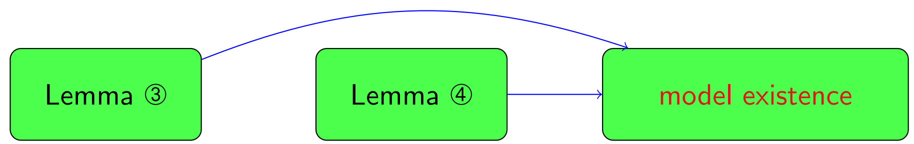
Theorem (Löwenheim-Skolem Theorem)

if a set of formulas $\mathcal{G}$ has a model, then $\mathcal{G}$ has a countable model

Proof.

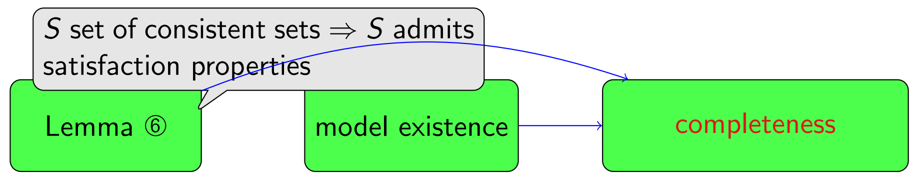
the model $\mathcal{M}$ constructed is countable ∎

How to Prove Completeness



Definition

for any formal system; if $\neg\, \exists$ proof of $\perp$ from a formula set $\mathcal{G}$, we say $\mathcal{G}$ is consistent



Later We Exploit the Proof