Logic and Mechanized Reasoning

First-Order Logic

Marijn J.H. Heule

Carnegie Mellon University

Introduction

Syntax

Using First-Order Logic

Semantics

Normal Forms

15311 students are the best!

How to encode this in propositional logic?

Introduction: Motivation

15311 students are the best!

How to encode this in propositional logic?

In first-order logic: $\forall x.\ 15311(x) \to \neg \exists y.\ \mathit{Better}(y, x)$

Introduction: Examples

$$\forall x. \exists y.\ R(x, y)$$

Important changes compared to propositional logic:

- Variables range over objects instead of Boolean values
- Relations are Boolean and the new literals
- First-order logic includes quantifiers to bound variables

Introduction: Examples

$$\forall x. \exists y.\ R(x, y)$$

Important changes compared to propositional logic:

- Variables range over objects instead of Boolean values
- Relations are Boolean and the new literals
- First-order logic includes quantifiers to bound variables

Many possible models:

- $(\mathbb{Z}, <)$

Introduction: Examples

$$\forall x.\exists y.\ R(x,y)$$

Important changes compared to propositional logic:

- Variables range over objects instead of Boolean values
- Relations are Boolean and the new literals
- First-order logic includes quantifiers to bound variables

Many possible models:

- $(\mathbb{Z}, <)$
- $(\mathbb{N}, <)$

Introduction: Examples

$$\forall x. \exists y.\ R(x, y)$$

Important changes compared to propositional logic:

- Variables range over objects instead of Boolean values
- Relations are Boolean and the new literals
- First-order logic includes quantifiers to bound variables

Many possible models:

- $(\mathbb{Z}, <)$
- $(\mathbb{N}, <)$
- $(\mathbb{N}, >)$

Introduction: Examples

$$\forall x. \exists y.\ R(x, y)$$

Important changes compared to propositional logic:

- Variables range over objects instead of Boolean values
- Relations are Boolean and the new literals
- First-order logic includes quantifiers to bound variables

Many possible models:

- $(\mathbb{Z}, <)$
- $(\mathbb{N}, <)$
- $(\mathbb{N}, >)$
- $(\{\text{People}\}, \mathit{Loves})$

Introduction: Quantifiers

The quantifier $\forall x$: something holds for all choices of x.

The quantifier $\exists x$: something holds for some choice of x.

The quantifiers do not commute:

- $\forall x. \exists y.\ x \neq y$
- $\exists y. \forall x.\ x \neq y$

Introduction: Quantifiers

The quantifier $\forall x$: something holds for all choices of x.

The quantifier $\exists x$: something holds for some choice of x.

The quantifiers do not commute:

- $\forall x . \exists y .\ x \neq y$
  For all objects there exist a different object
- $\exists y . \forall x .\ x \neq y$

Introduction: Quantifiers

The quantifier $\forall x$: something holds for all choices of x.

The quantifier $\exists x$: something holds for some choice of x.

The quantifiers do not commute:

- $\forall x . \exists y .\ x \neq y$
  For all objects there exist a different object
- $\exists y . \forall x .\ x \neq y$
  There exists an object that differs from all other objects (including itself)

Introduction: Terms and Formulas

Syntax and semantics are similar to propositional logic

Two additional categories of expression:

- Terms name things in the intended interpretation
- Formulas say things about those objects

We use recursive definitions to specify how to evaluate them for a given interpretation

Introduction: Propositional vs First-Order Logic

Propositional logic is decidable

- Assign truth values to finitely many variables
- Various decision procedures, e.g. truth table

First-order logic is undecidable

- Some satisfiable formulas require infinitely many objects
- A statement is true in all models if and only if it is provable
- Provability is equivalent to the halting problem

Introduction: Decision Procedures

Decidable fragments:

- Equational reasoning
- Linear arithmetic on the real numbers
- Efficiently implemented in SMT solvers
- Strong tools: Z3 and CVC5

First-order theorem proving:

- Searching for proofs from axioms
- Potentially infinite runtime if no proof exists
- Strong tool: Vampire

Introduction

Syntax

Using First-Order Logic

Semantics

Normal Forms

Syntax: Language

- Functions map objects onto an object
  - We use lowercase for functions, e.g. f, g, and h
  - Functions can have arbitrary arity, e.g. $f(x, y)$
  - 0-arity functions are constants, e.g. a, b, and c
  - We use $x + y$ as shorthand for $+(x, y)$
- Relations can be either true or false
  - We use uppercase for relations, e.g. P, Q, and R
  - Relations can have arbitrary arity, e.g. $R(x, y)$
  - 0-arity relations are similar to Boolean variables
  - Special relation $=$ whether two objects are equal
  - We use $x \neq y$ as shorthand for $\neg(x = y)$
  - We use $x \leq y$ as shorthand for $\leq (x, y)$

Syntax: Set of Terms

The set of terms of the language L is generated inductively:

- Each variable $x, y, z, \ldots$ is a term.
- Each constant symbol of L is a term.
- If f is any n-ary function symbol of L and $t_1, t_2, \ldots, t_n$ are terms of L, then $f(t_1, t_2, \ldots, t_n)$ is a term.

Syntax: Quantifiers and Renaming

The quantifiers $\forall$ and $\exists$ bound variables

Variables that are not bounded are free

$$\exists z.\ x < z \wedge z < y$$

Closed variable z is in between free variables x and y

Syntax: Quantifiers and Renaming

The quantifiers $\forall$ and $\exists$ bound variables

Variables that are not bounded are free

$$\exists z.\ x < z \land z < y$$

Closed variable z is in between free variables x and y

This is the same as $\exists w.\ x < w \land w < y$

Bound variables can be renamed

Syntax: Quantifiers and Renaming

The quantifiers $\forall$ and $\exists$ bound variables

Variables that are not bounded are free

$$\exists z.\ x < z \wedge z < y$$

Closed variable z is in between free variables x and y

This is the same as $\exists w.\ x < w \wedge w < y$

Bound variables can be renamed

A formula without free variables is called a sentence

Syntax: Set of Formulas

The set of formulas of the language L is generated inductively:

- If R is any n-ary relation symbol of L and $t_1, t_2, \ldots, t_n$ are terms of L, then $R(t_1, t_2, \ldots, t_n)$ is a formula.
- If s and t are terms, then $s = t$ is a formula.
- $\top$ and $\bot$ are formulas.
- If A and B are formulas, so are $\neg A$, $A \wedge B$, $A \vee B$, $A \to B$, and $A \leftrightarrow B$.
- If A is a formula and x is a variable, then $\forall x.\ A$ and $\exists x.\ A$.

Syntax: Substitution

Recall substitution in propositional logic
Substitution in first-order logic is similar

- $s[t/x]$ substitutes term t for variable x in term s
- $A[t/x]$ substitutes term t for variable x in formula A

Syntax: Substitution

Recall substitution in propositional logic
Substitution in first-order logic is similar

- $s[t/x]$ substitutes term t for variable x in term s
- $A[t/x]$ substitutes term t for variable x in formula A

Simultaneous substitution replaces multiple variables at once

Given a substitution σ and a term t, substitution is defined as

$$\begin{aligned} \sigma\, x &= \sigma(x) \\ \sigma f(t_1, \ldots, t_n) &= f(\sigma\, t_1, \ldots, \sigma\, t_n) \end{aligned}$$

Syntax: Substitution

Recall substitution in propositional logic
Substitution in first-order logic is similar

- $s[t/x]$ substitutes term t for variable x in term s
- $A[t/x]$ substitutes term t for variable x in formula A

Simultaneous substitution replaces multiple variables at once

Given a substitution σ and a term t, substitution is defined as

$$\begin{aligned} \sigma\, x &= \sigma(x) \\ \sigma f(t_1, \ldots, t_n) &= f(\sigma\, t_1, \ldots, \sigma\, t_n) \end{aligned}$$

Substitution σA is similar, though it may require renaming

Introduction

Syntax

Using First-Order Logic

Semantics

Normal Forms

Using FOL: Quantifier Examples

Bill and Ann are married and all their children are smart

How to express this in first-order logic?

Using FOL: Quantifier Examples

Bill and Ann are married and all their children are smart

How to express this in first-order logic?

$$Married(Bill, Ann) \wedge \forall x.\ Child(x, Bill) \wedge Child(x, Ann) \rightarrow Smart(x)$$

Using FOL: Quantifier Examples

Bill and Ann are married and all their children are smart

How to express this in first-order logic?

$$Married(Bill, Ann) \land \forall x.\ Child(x, Bill) \land Child(x, Ann) \rightarrow Smart(x)$$

First-order logic allows expressing many other things:

- $Married(x, y)$ is symmetric, $Married(x, y) \leftrightarrow Married(y, x)$
- $Child(x, y)$ is asymmetric, $\neg Child(x, y) \lor \neg Child(y, x)$

Using FOL: Quantifier Examples

- $\exists x.\ Dog(x) \land Blue(x)$
- $\exists x.\ Dog(x) \rightarrow Blue(x)$
- $\forall x.\ Dog(x) \land Blue(x)$
- $\forall x.\ Dog(x) \rightarrow Blue(x)$

Using FOL: Integer Examples

$$\begin{aligned} even(x) &\equiv \exists y.\ x = 2 \cdot y \\ odd(x) &\equiv \exists y.\ x = 2 \cdot y + 1 \\ div(x, y) &\equiv \exists z.\ y = x \cdot z. \end{aligned}$$

Every integer is even or odd, but not both.

A integer is even if and only if it is divisible by two.

If some integer, x, is even, then so is x^2.

A integer x is even if and only if $x + 1$ is odd.

If x divides y and y divides z, then x divides z.

Using FOL: Integer Examples

$$\begin{aligned} even(x) &\equiv \exists y.\; x = 2 \cdot y \\ odd(x) &\equiv \exists y.\; x = 2 \cdot y + 1 \\ div(x,y) &\equiv \exists z.\; y = x \cdot z. \end{aligned}$$

Every integer is even or odd, but not both.

- $\forall x.\; (even(x) \vee odd(x)) \wedge \neg(even(x) \wedge odd(x))$

A integer is even if and only if it is divisible by two.

If some integer, x, is even, then so is x^2.

A integer x is even if and only if $x + 1$ is odd.

If x divides y and y divides z, then x divides z.

Using FOL: Integer Examples

$$\begin{aligned} even(x) &\equiv \exists y.\ x = 2 \cdot y \\ odd(x) &\equiv \exists y.\ x = 2 \cdot y + 1 \\ div(x,y) &\equiv \exists z.\ y = x \cdot z. \end{aligned}$$

Every integer is even or odd, but not both.

- $\forall x.\ (even(x) \vee odd(x)) \wedge \neg(even(x) \wedge odd(x))$

A integer is even if and only if it is divisible by two.

- $\forall x.\ even(x) \leftrightarrow div(2,x)$

If some integer, x, is even, then so is x^2.

A integer x is even if and only if $x+1$ is odd.

If x divides y and y divides z, then x divides z.

Using FOL: Integer Examples

$$\begin{aligned} even(x) &\equiv \exists y.\ x = 2 \cdot y \\ odd(x) &\equiv \exists y.\ x = 2 \cdot y + 1 \\ div(x, y) &\equiv \exists z.\ y = x \cdot z. \end{aligned}$$

Every integer is even or odd, but not both.

- $\forall x.\ (even(x) \vee odd(x)) \wedge \neg(even(x) \wedge odd(x))$

A integer is even if and only if it is divisible by two.

- $\forall x.\ even(x) \leftrightarrow div(2, x)$

If some integer, x, is even, then so is x^2.

- $\forall x.\ even(x) \rightarrow even(x^2)$

A integer x is even if and only if $x + 1$ is odd.

If x divides y and y divides z, then x divides z.

Using FOL: Integer Examples

$$
\begin{aligned}
even(x) &\equiv \exists y.\ x = 2 \cdot y \\
odd(x) &\equiv \exists y.\ x = 2 \cdot y + 1 \\
div(x, y) &\equiv \exists z.\ y = x \cdot z.
\end{aligned}
$$

Every integer is even or odd, but not both.

- $\forall x.\ (even(x) \vee odd(x)) \wedge \neg(even(x) \wedge odd(x))$

A integer is even if and only if it is divisible by two.

- $\forall x.\ even(x) \leftrightarrow div(2, x)$

If some integer, x, is even, then so is x^2.

- $\forall x.\ even(x) \to even(x^2)$

A integer x is even if and only if $x + 1$ is odd.

- $\forall x.\ even(x) \leftrightarrow odd(x + 1)$

If x divides y and y divides z, then x divides z.

Using FOL: Integer Examples

$$\begin{aligned} even(x) &\equiv \exists y.\ x = 2 \cdot y \\ odd(x) &\equiv \exists y.\ x = 2 \cdot y + 1 \\ div(x,y) &\equiv \exists z.\ y = x \cdot z. \end{aligned}$$

Every integer is even or odd, but not both.

- $\forall x.\ (even(x) \vee odd(x)) \wedge \neg(even(x) \wedge odd(x))$

A integer is even if and only if it is divisible by two.

- $\forall x.\ even(x) \leftrightarrow div(2,x)$

If some integer, x, is even, then so is x^2.

- $\forall x.\ even(x) \rightarrow even(x^2)$

A integer x is even if and only if $x+1$ is odd.

- $\forall x.\ even(x) \leftrightarrow odd(x+1)$

If x divides y and y divides z, then x divides z.

- $\forall x.\ \forall y.\ \forall z.\ div(x,y) \wedge div(y,z) \rightarrow div(x,z)$

Using FOL: Relativization

Quantifiers always range over the entire universe

Propositional connectives can restrict the domain of a quantifier

There is an even number between 1 and 3

- $\exists x.\ even(x) \wedge 1 < x \wedge x < 3$

Every even number greater than 1 is greater than 3

- $\forall x.\ even(x) \wedge x > 1 \to x > 3$

Using FOL: Many-Sorted FOL

First-order logic formulas can have multiple sorts of variables

Example

Consider a geometry problem with multiple sorts:

- points: p, q, r, ...
- lines: L, M, N, ...
- circles: α, β, γ, ...
- $On(p, L)$ denoting point p lies on line L

We will focus on first-order logic with a single sort

Introduction

Syntax

Using First-Order Logic

Semantics

Normal Forms

Semantics: Model

A model $\mathfrak{M}$ for a language consists of

- A set of objects, $|\mathfrak{M}|$, called the universe of $\mathfrak{M}$.
- For each function symbol f in the language, a function $f^{\mathfrak{M}}$ from the universe of $\mathfrak{M}$ to itself, with the corresponding arity.
- For each relation symbol R in the language, a relation $R^{\mathfrak{M}}$ on the universe of $\mathfrak{M}$, with the corresponding arity.

Semantics: Model

A model $\mathfrak{M}$ for a language consists of

- A set of objects, $|\mathfrak{M}|$, called the universe of $\mathfrak{M}$.
- For each function symbol f in the language, a function $f^{\mathfrak{M}}$ from the universe of $\mathfrak{M}$ to itself, with the corresponding arity.
- For each relation symbol R in the language, a relation $R^{\mathfrak{M}}$ on the universe of $\mathfrak{M}$, with the corresponding arity.

Example

$$\forall x, y.\ (f(x) \neq f(f(x)) \wedge (R(x,y) \leftrightarrow x \neq y)$$

Semantics: Model

A model $\mathfrak{M}$ for a language consists of

- A set of objects, $|\mathfrak{M}|$, called the universe of $\mathfrak{M}$.
- For each function symbol f in the language, a function $f^{\mathfrak{M}}$ from the universe of $\mathfrak{M}$ to itself, with the corresponding arity.
- For each relation symbol R in the language, a relation $R^{\mathfrak{M}}$ on the universe of $\mathfrak{M}$, with the corresponding arity.

Example

$$\forall x, y.\ (f(x) \neq f(f(x)) \wedge (R(x, y) \leftrightarrow x \neq y)$$

- Set of objects: $\{\star, \circ\}$
- $f^{\mathfrak{M}}(\star) = \circ$, $f^{\mathfrak{M}}(\circ) = \star$
- $R^{\mathfrak{M}}(\star, \star) = R^{\mathfrak{M}}(\circ, \circ) = \bot$, $R^{\mathfrak{M}}(\star, \circ) = R^{\mathfrak{M}}(\circ, \star) = \top$

Semantics: Finite and Infinite Models

Example

$$\forall x, y.\ (f(x) \neq f(f(x)) \wedge (R(x,y) \leftrightarrow x \neq y)$$

- Set of objects: $\{\star, \circ\}$
- $f^{\mathfrak{M}}(\star) = \circ, f^{\mathfrak{M}}(\circ) = \star$
- $R^{\mathfrak{M}}(\star,\star) = R^{\mathfrak{M}}(\circ,\circ) = \bot$, $R^{\mathfrak{M}}(\star,\circ) = R^{\mathfrak{M}}(\circ,\star) = \top$

What about the formula

$$\forall x.\ (f(x) \neq c) \wedge (f(x) \neq f(y) \vee x = y)$$

Semantics: Assignment

Let σ be an assignment of elements of $|\mathfrak{M}|$ to free variables.
Then every term t has a value $[\![t]\!]_{\mathfrak{M},\sigma}$ in $|\mathfrak{M}|$ defined recursively:

- $[\![x]\!]_{\mathfrak{M},\sigma} = \sigma(x)$
- For every n-ary function symbol f and every tuple of terms $t_1, \ldots, t_n$, $[\![f(t_1, \ldots, t_n)]\!]_{\mathfrak{M},\sigma} = f^{\mathfrak{M}}([\![t_1]\!]_{\mathfrak{M},\sigma}, \ldots, [\![t_n]\!]_{\mathfrak{M},\sigma})$

Semantics: Evaluation

- $\mathfrak{M} \models_\sigma t = t'$ iff $[\![t]\!]_{\mathfrak{M},\sigma} = [\![t']\!]_{\mathfrak{M},\sigma}$.
- $\mathfrak{M} \models_\sigma R(t_0, \ldots, t_{n-1})$ iff $R^{\mathfrak{M}}([\![t_0]\!]_{\mathfrak{M},\sigma}, \ldots, [\![t_{n-1}]\!]_{\mathfrak{M},\sigma})$.
- $\mathfrak{M} \models_\sigma \bot$ is always false.
- $\mathfrak{M} \models_\sigma \top$ is always true.
- $\mathfrak{M} \models_\sigma A \wedge B$ iff $\mathfrak{M} \models_\sigma A$ and $\mathfrak{M} \models_\sigma B$.
- $\mathfrak{M} \models_\sigma A \vee B$ iff $\mathfrak{M} \models_\sigma A$ or $\mathfrak{M} \models_\sigma B$.
- $\mathfrak{M} \models_\sigma A \to B$ iff $\mathfrak{M} \not\models_\sigma A$ or $\mathfrak{M} \models_\sigma B$.
- $\mathfrak{M} \models_\sigma A \leftrightarrow B$ iff $\mathfrak{M} \models_\sigma A$ and $\mathfrak{M} \models_\sigma B$ either both hold or both don't hold.
- $\mathfrak{M} \models_\sigma \forall x.\ A$ iff for every $a \in |\mathfrak{M}|$, $\mathfrak{M} \models_{\sigma[x \mapsto a]} A$.
- $\mathfrak{M} \models_\sigma \exists x.\ A$ iff for some $a \in |\mathfrak{M}|$, $\mathfrak{M} \models_{\sigma[x \mapsto a]} A$.

Semantics: Satisfiable, Unsatisfiable, and Valid

- A formula A is satisfiable if and only if there exists a model $\mathfrak{M}$ and assignment σ, such that $\mathfrak{M} \models_{\sigma} A$.
- A formula A unsatisfiable if it is not satisfiable.
- A formula A is valid if it is satisfied by every model.

Semantics: Satisfiable, Unsatisfiable, and Valid

- A formula A is satisfiable if and only if there exists a model $\mathfrak{M}$ and assignment σ, such that $\mathfrak{M} \models_\sigma A$.
- A formula A unsatisfiable if it is not satisfiable.
- A formula A is valid if it is satisfied by every model.

Example

Which one(s) of the formulas is satisfiable/unsatisfiable/valid?

- $\exists x.\ R(x) \wedge \neg R(x)$
- $\forall x.\ x \neq x$
- $\forall x.\ R(x) \vee \neg R(x)$

Normal Forms: De Morgan Laws for Quantifiers

Recall De Morgan laws for propositional logic:

$$\begin{aligned} \neg(A \wedge B) &\equiv \neg A \vee \neg B \\ \neg(A \vee B) &\equiv \neg A \wedge \neg B \end{aligned}$$

Additionally, we have De Morgan laws for quantifiers:

$$\neg \forall x.\ A \equiv \exists x.\ \neg A$$

$$\neg \exists x.\ A \equiv \forall x.\ \neg A$$

These rules allow you to move negations inward

Normal Forms: Bring Quantifiers to the Front

These rules allow you to move the quantifiers to the front:

$$\begin{aligned}
(\forall x.\ A) \vee B &\leftrightarrow \forall x.\ A \vee B \\
(\forall x.\ A) \wedge B &\leftrightarrow \forall x.\ A \wedge B \\
(\exists x.\ A) \vee B &\leftrightarrow \exists x.\ A \vee B \\
(\exists x.\ A) \wedge B &\leftrightarrow \exists x.\ A \wedge B
\end{aligned}$$

Some renaming might be required

Normal Forms: Bring Quantifiers to the Front

These rules allow you to move the quantifiers to the front:

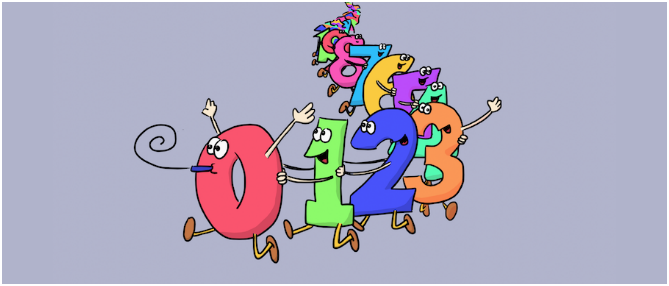
$$\begin{array}{lcl} (\forall x.\ A) \vee B & \leftrightarrow & \forall x.\ A \vee B \\ (\forall x.\ A) \wedge B & \leftrightarrow & \forall x.\ A \wedge B \\ (\exists x.\ A) \vee B & \leftrightarrow & \exists x.\ A \vee B \\ (\exists x.\ A) \wedge B & \leftrightarrow & \exists x.\ A \wedge B \end{array}$$

Some renaming might be required

In practice it is better to apply Skolemization to get rid of quantifiers (covered in a future lecture)

One More Thing

Natural Number Game



Go to `https://adam.math.hhu.de`

Click on "Natural Number Game" to start