First order Logic

Klaus v. Gleissenthall

Logic:
- Propositional
- First order
- Theories

Programs to Logic:
- Hoare Logic
- VCGen

Automation
- Horn clauses
- Predicate Abstraction

Security
- Information Flow Control
- Side-Channels

Where are we?

- Logic as the language of computation
- We can now ask and answer questions in propositional logic
- But, it's too restricted to encode many important problems about programs
- Now: more expressive logics
  - We start with first order logic

First Order Logic

Propositional logic: $(p \land r)$

First Order Logic: $\forall x. \exists y. \exists z.\ x+y > 0 \land 0<x*z$

New concepts:

- Quantifiers $\exists$, $\forall$
- Functions +,*
- Relations >,<
- Constants 0

First Order Logic: Syntax

First order language: $L(C, F, R)$

- C : set of constants
- F : set of function symbols
- R : set of relations

Basic Terms: constant $a,b,c,\ldots \in C$ variable $x, y, z, x_1, x_2, \ldots$

Composite Terms: $f(t_1, \ldots, t_k)$ where $f \in F$ and $t_1, \ldots, t_k$ are (basic/composite) terms

Example: mary, x , sister(mary), price(x ,bol), age(mother(y)), …

First Order Logic: Syntax

First order language: $L(C, F, R)$

- C : set of constants
- F : set of function symbols
- R : set of relations

Formula: F, F_1, F_2

- $\top$, $\bot$
- <u>Atomic predicate:</u> $p(t_1, \ldots, t_k)$, and $p \in R$ with arity k, $t_1, \ldots, t_k$ are terms
- $\neg F$, $F_1 \wedge F_2$, $F_1 \vee F_2$
- $\forall x.\ F$, $\exists x.\ F$, for some variable x

Atomic predicates are the propositional variables of FOL

First Order Logic: Syntax

Quiz: Which of the following are syntactically correct first order formulas?

- f(x)
- p(x)
- p(f(x))
- p(p(x))
- p(f(f(x)))

Quantifiers and Scoping

<u>Scope</u>: For a quantifier $\forall x.\ F$ (or $\exists x.\ F$) F is the called <u>scope</u> of the quantifier

An occurrence of a variable is called <u>bound</u>, if it's in the scope of a quantifier

An occurrence of a variable is called <u>free</u>, if it's not in the scope of any quantifier

Quiz:

$$\forall y.((\forall x\ .p(x\ )) \rightarrow q(x\ ,\ y))$$

- Is y bound or free?
- Is the first occurrence of x bound or free?
- What about the second?

Closed, Open, and Ground Formulas

- A formula with no free variables is called a <u>closed</u> formula, or <u>sentence</u>

- A formula with free variables is called <u>open</u>

Quiz: Is the formula $\forall y.((\forall x.p(x)) \rightarrow (\exists x.q(x, y)))$ closed or open?

- A formula is called <u>ground</u> if it does not contain any variables

<u>Example:</u> $p(a, f(b)) \rightarrow q(c)$ is ground

Quiz:

- Is $\forall x.p(x)$ ground?

FOL Example: Fermat's Last Theorem

Fermat's Last Theorem:

- No three positive integers x,y,z satisfy the equation x^n + y^n = z^n , for any integer n greater than 2.
- Assuming we are talking about integers, how do we express this theorem in FOL using function constant ^ and relation constants >, =?

First Order Logic: Semantics

- For propositional logic, semantic concepts were quite simple
- FOL is a bit more involved
- To give a semantics to FOL, we need to first fix a universe of discourse
- The universe of discourse is a non-empty set of objects we want to say something about
- Can be finite, countably infinite, uncountably infinite; but can't be empty.

Examples: $\mathbb{N}$, $\mathbb{R}$, {□,◍}, students in this class, types of fruit sold at AH

First Order Logic: Semantics

- An interpretation I is a mapping from C, F, R to objects in universe U
- I maps $c \in C$ to U, i.e., $I(c) \in U$
- I maps $f \in F$ to $I(f) \in U^k \rightarrow U$ (i.e., a function over U), where k is the arity of f
- I maps $p \in R$ to $I(p) \in U^k$, where k is the arity of p

Note: A first order interpretation does not talk about variables, only constants

First Order Logic: Semantics

Example: Consider the first order language containing object constants $\{a, b, c\}$, unary function constant f, and ternary relation constant r.

- Let's fix the following universe of discourse $U \triangleq \{1,2,3\}$
- A possible interpretation I is:

$$I(a) \triangleq 1, \ I(b) \triangleq 2, I(c) \triangleq 2$$

$$I(f) \triangleq \{1 \rightarrow 2, 2 \rightarrow 3, 3 \rightarrow 3\}$$

$$I(r) \triangleq \{\langle 1,2,1 \rangle, \langle 2,2,1 \rangle\}$$

Structures and Variable Assignments

A structure $S = \langle U, I \rangle$ for a first order language consists of a universe of discourse U and an interpretation I

A variable assignment σ to a FOL formula F in a structure $S = \langle U, I \rangle$ is a mapping from variables in F to an element in U

Example: given $U=\{1,2,3\}$ a possible assignment for x is $\sigma(x)=2$

Semantics: Evaluating Terms

We define how to evaluate a term t under interpretation I and assignment σ, written $\langle I, \sigma \rangle(t)$.

Constant: $\langle I, \sigma \rangle(a) \triangleq I(a)$

Variable: $\langle I, \sigma \rangle(x) \triangleq \sigma(x)$

Function Term: $\langle I, \sigma \rangle(f(t_1, \ldots, t_k)) \triangleq I(f)(v_1, \ldots, v_k)$, where $v_1 \triangleq \langle I, \sigma \rangle(t_1), \ldots, v_k \triangleq \langle I, \sigma \rangle(t_k)$

Semantics: Evaluating Terms

Quiz: Under σ and I, what do these terms evaluate to?

Example:

Let $U \triangleq \{1,2\}$ and $\sigma \triangleq \{x \rightarrow 2, y \rightarrow 1\}$

$I(a) \triangleq 1, \; I(b) \triangleq 2$

$I(f) \triangleq \{\langle 1, 1\rangle \rightarrow 2, \langle 1, 2\rangle \rightarrow 2, \langle 2, 1\rangle \rightarrow 1, \langle 2, 2\rangle \rightarrow 1\}$

$f(a, y) =$

$f(x, b) =$

$f(f(x, b), f(a, y)) =$

Semantics: Evaluating Formulas

We define evaluation of formula F under structure $S = \langle U, I \rangle$ and variable assignment σ.

- If F evaluates to <u>true</u> under U , I, σ, we write $U,I,\sigma \models F$
- If F evaluates to <u>false</u> under U ,I, σ, we write $U,I,\sigma \not\models F$
- Let's define the semantics of $\models$, by induction

Semantics: Evaluating Formulas

Base Case 1:

- $U,I,\sigma \vDash \top$
- $U,I,\sigma \nvDash \bot$

Base Case 2:

- $U,I,\sigma \vDash p(t_1, \ldots, t_k)$ iff $\langle v_1, \ldots, v_k \rangle \in I(p)$ where, $v_1 \triangleq \langle I, \sigma \rangle(t_1), \ldots, v_k \triangleq \langle I, \sigma \rangle(t_k)$

Semantics: Evaluating Formulas

Quiz:

- Consider constants a, b and unary function f , and binary relation p
- Universe U={□,◍} and interpretation I:

I(a) = □ I(b) = ◍ I(f) ={□ → ◍, ◍ → □} I(p) ={⟨◍,□⟩, ⟨◍, ◍⟩}

- Consider variable assignment σ : {x →□}

Under, U,I, σ, what do the following formulas evaluate to?

- p(f(b), f(x)) = p(f(x), f(b)) = p(a,f(x)) =

Semantics: Evaluating Formulas

Boolean connectives:

- $U,I,\sigma \models \neg F$ iff $U,I,\sigma \not\models F$
- $U,I,\sigma \models F_1 \wedge F_2$ iff $U,I,\sigma \models F_1$ and $U,I,\sigma \models F_2$
- $U,I,\sigma \models F_1 \vee F_2$ iff $U,I,\sigma \models F_1$ or $U,I,\sigma \models F_2$

Semantics: Variant of Variable Assignment

- What's still missing? Quantifiers!
- First, let's define an x -variant of a variable assignment.
- An **x-variant** of assignment σ, written $\sigma[x \mapsto c]$, is the assignment that agrees with σ for assignments to all variables except x and assigns x to c.

Example: $\sigma \triangleq \{x \rightarrow 1, y \rightarrow 2\}$, what is $\sigma[x \mapsto 3]$?

Semantics: Evaluating Quantifiers

Universal Quantifier:

- $U,I,\sigma \models \forall x.\ F$ iff for all $v \in U$, $U,I,\sigma[x \mapsto v] \models F$
- $U,I,\sigma \models \exists x.\ F$ iff there exists $v \in U$ such that $U,I,\sigma[x \mapsto v] \models F$

Semantics: Evaluating Quantifiers

Example: • Universe U={□,◍}, assignment σ : {x →□} and interpretation I:

I(a) = □ I(b) = ◍ I(f) ={□ → ◍, ◍ → □} I(p) ={⟨◍,□⟩, ⟨◍, ◍⟩}

Quiz: Under U,I,σ, what do the following formulas evaluate to:

$\forall x.p(x, a) =$

$\forall x.p(b, x) =$

$\exists x .p(a, x) =$

$\forall x . (p(a, x) \rightarrow p(b, x)) =$

$\exists x.(p(f(x), f(x)) \rightarrow p(x , x)) =$

Satisfiability and Validity

- A first-order formula F is <u>satisfiable</u> iff there exists a structure S and variable assignment σ such that $S,\sigma \models F$
- F is <u>unsatisfiable</u> otherwise
- Structure S is a <u>model</u> of F written $S \models F$, iff for all variable assignments σ, $S,\sigma \models F$
- Formula F is <u>valid</u> written $\models F$, iff for all structures S, $S \models F$

Satisfiability and Validity

Quiz:

- Is the formula $\forall x. \exists y. p(x , y)$ satisfiable?

- Is the formula $\forall x . (p(x,x) \rightarrow \exists y. p(x,y))$ valid?

Satisfiability and Validity

Quiz:

- Is the formula $(\exists x\ .p(x)) \rightarrow p(x)$ sat, unsat, or valid
- Is the formula $(\forall x\ .p(x)) \rightarrow p(x)$ sat, unsat, or valid?
- What about $(\forall x\ .(p(x) \rightarrow q(x))) \rightarrow (\exists x\ .(p(x) \wedge q(x)))$?

Satisfiability and Validity

- Recall: A structure S is a model of a formula if for all σ, $S,\sigma \models F$

Quiz:

- Consider a formula F such that $S,\sigma \models F$. Is S a model F?
- Consider a sentence F such that $S,\sigma \models F$. Is S a model F?
- Consider a ground formula F such that $S,\sigma \models F$. Is S a model F?

Semantic arguments

- We have seen what it means for a formula F to be valid, but how to <u>prove</u> validity?
- We extend the semantic argument method from PL to FOL
- Recall: In propositional logic, satisfiability and validity are duals

F is valid iff ¬F is unsatisfiable

- Since this duality also holds in FOL, we focus on validity

Semantic arguments

- Recall: Semantic argument method is a proof by <u>contradiction</u>
- Basic Idea: Assume that F is not valid, i.e., there exist S, σ such that $S, \sigma \not\models F$
- Then, apply proof rules
- If we can derive a contradiction on <u>every branch</u> of the proof, F is valid

Semantic arguments: New Rules

- All rules from propositional logic, but we need new rules for quantifiers

<u>univ I</u> $$\frac{U,I,\sigma \models \forall x.F}{U,I,\sigma[x \mapsto v] \models F} \quad \text{(for } \textbf{any } v \in U\text{)}$$

- For example, suppose $U,I,\sigma \models \forall x\ .hates(jack, x)$

- Using the above rule, we can conclude $U,I,\sigma[x \mapsto I(jack)] \models hates(jack, x)$

Semantic arguments: New Rules

<u>univ II</u> $\dfrac{U,I,\sigma \not\models \forall x.F}{U,I,\sigma[x \mapsto v] \not\models F}$ (for a **fresh** $v \in U$)

- By fresh, we mean not previously used in the proof
- Why do we need this restriction?

Semantic arguments: New Rules

$$\underline{\text{exist I}} \quad \frac{U,I,\sigma \models \exists x.F}{U,I,\sigma[x \mapsto v] \models F} \quad \text{(for a } \textbf{fresh } v \in U\text{)}$$

- Again fresh, means not previously used in the proof

$$\underline{\text{exist II}} \quad \frac{U,I,\sigma \not\models \exists x.F}{U,I,\sigma[x \mapsto v] \not\models F} \quad \text{(for } \textbf{any } v \in U\text{)}$$

- If U ,I, σ do not entail ∃x .F, this means there does not exist any object for which F holds
- Thus, no matter what object x maps to, it still won't entail F

Semantic arguments: New Rules

- Finally, we need a rule for deriving contradictions

$$\text{\underline{contr}}\quad \frac{\begin{array}{l} U,I,\sigma[\ldots] \vDash p(s_1, \ldots, s_k) \\ U,I,\sigma[\ldots] \nvDash p(t_1, \ldots, t_k) \\ \langle I, \sigma\rangle(t_1) = \langle I, \sigma\rangle(s_1), \ldots, \ \langle I, \sigma\rangle(t_k) = \langle I, \sigma\rangle(s_k) \end{array}}{U,I,\sigma \vDash \bot}$$

- Example: Suppose we have $S,\{x \rightarrow a\} \vDash p(x)$ and $S,\{y \rightarrow a\} \nvDash p(y)$
- Then, we can derive $\bot$

Semantic arguments: Examples

<u>Example:</u> $F = (\forall x\, .p(x)) \rightarrow (\forall y.p(y))$

- Start: assume there exist S,σ such that $S,\sigma \not\models F$

Semantic arguments: Examples

Example: $F = (\forall y \,.\, (p(y) \lor q(y))) \rightarrow (\exists x \,.p(x) \lor \forall x \,.q(x))$

- Prove that the formula is valid

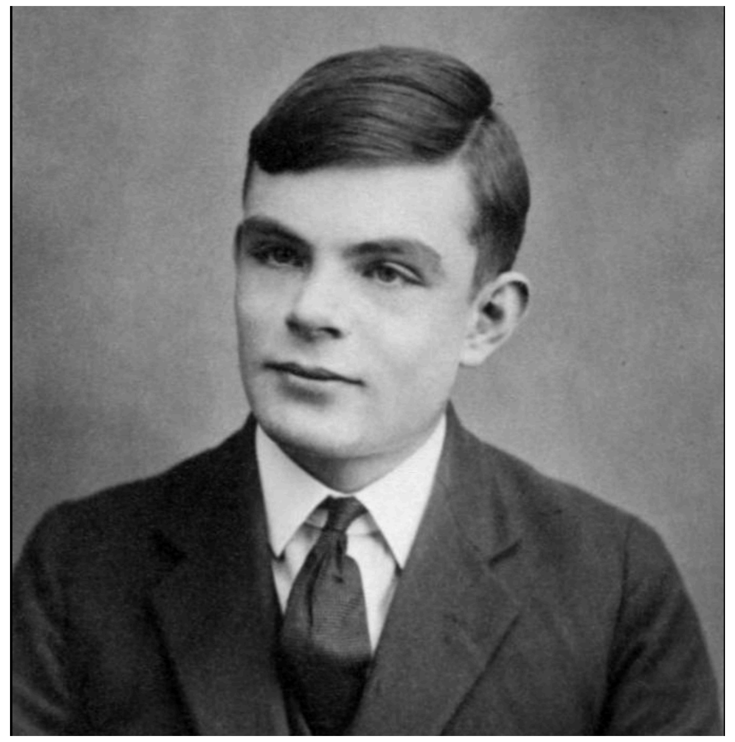
Semantic arguments: Examples

- To try at home:
  - $p(a) \rightarrow \exists x.\ p(x)$
  - $(\forall x.\ p(x)) \leftrightarrow (\neg\exists x.\ \neg p(x))$
  - $(\forall x\ .(p(x) \wedge q(x))) \rightarrow (\forall x\ .p(x)) \wedge (\forall x\ .q(x))$
  - $\exists x\ (P(x) \vee Q(x)) \Leftrightarrow \exists x\ P(x) \vee \exists x\ Q(x)$

Soundness and Completeness

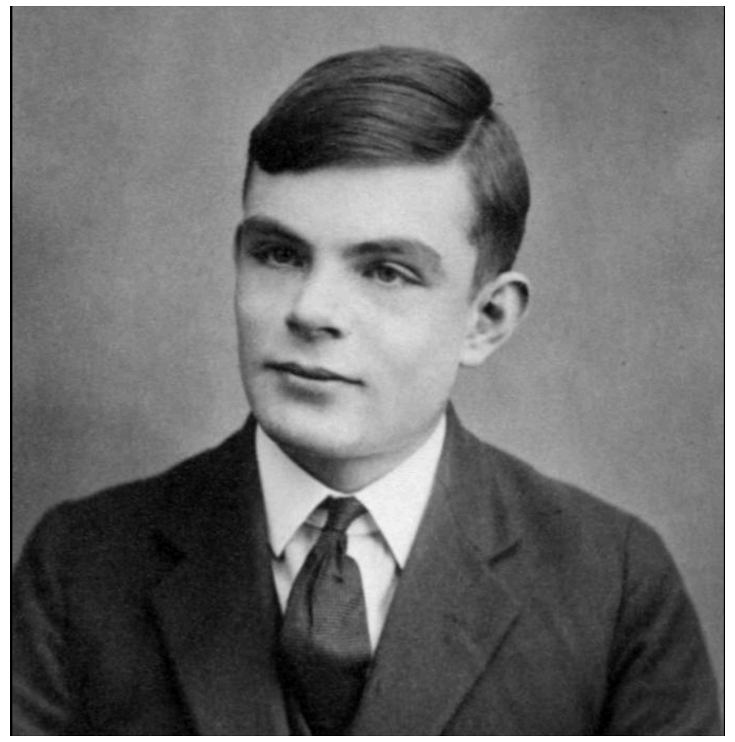
- The proof rules we used are <u>sound</u> and <u>complete</u>.
- <u>Soundness</u>: If every branch of semantic argument proof derives a contradiction, then F is indeed valid.
- Translation: The proof system does not reach wrong conclusions
- <u>Completeness:</u> If formula F is valid, then there exists a finite-length proof in which every branch derives $\perp$
- Translation: There are no valid first-order formulas which we cannot prove to be valid using our proof rules.

Undecidability of FOL




- Really important result by Church and Turing
- It is <u>undecidable</u> whether a first-order formula is valid
- Review: A problem is decidable iff there exists a procedure P such that for any input:
  - P <u>halts</u> and says “yes" if the answer is positive
  - P <u>halts</u> and says “no" if the answer is negative
- Can’t we just use the proof system? What’s the hard part?

Semidecidability of FOL



- First order logic is <u>semidecidable</u>
- A problem is semidecidable iff there exists a procedure P such that for any input:
    - P <u>halts</u> and says “yes" if the answer is positive
    - P <u>may not halt</u> if the answer is negative (if it halts, it says “no")
- How could we build such an algorithm from the proof system?
- No algorithm is guaranteed to terminate, if the formula is not valid

Where are we?

- Logic as the language of computation
- We’ve seen first-order logic
- *Very* expressive (see Fermat’s last theorem)
  - In fact, we can use it to encode all of mathematics via ZF set theory
- But undecidable makes decision procedures unpredictable
  - We don’t know if they will terminate!
- Next first-order theories
- Focus on decidable fragments of FOL that allow encoding interesting questions about programs

Proof Rules

<u>neg</u>

$$\frac{S,\sigma \models \neg F}{S,\sigma \not\models F} \qquad \frac{S,\sigma \not\models \neg F}{S,\sigma \models F}$$

<u>conj</u>

$$\frac{S,\sigma \models F_1 \land F_2}{S,\sigma \models F_1 \quad S,\sigma \models F_2} \qquad \frac{S,\sigma \not\models F_1 \land F_2}{S,\sigma \not\models F_1 \ \underline{\text{or}}\ S,\sigma \not\models F_2}$$

<u>disj</u>

$$\frac{S,\sigma \models F_1 \lor F_2}{S,\sigma \models F_1 \ \underline{\text{or}}\ S,\sigma \models F_2} \qquad \frac{S,\sigma \not\models F_1 \lor F_2}{S,\sigma \not\models F_1 \quad S,\sigma \not\models F_2}$$

<u>imp</u>

$$\frac{S,\sigma \models F_1 \rightarrow F_2}{S,\sigma \models \neg F_1 \ \underline{\text{or}}\ S,\sigma \models F_2} \qquad \frac{S,\sigma \not\models F_1 \rightarrow F_2}{S,\sigma \models F_1 \quad S,\sigma \not\models F_2}$$

<u>univ</u>

$$\frac{U,I,\sigma \models \forall x.F}{U,I,\sigma[x \mapsto v] \models F}\ (\textbf{any}\ v \in U) \qquad \frac{U,I,\sigma \not\models \forall x.F}{U,I,\sigma[x \mapsto v] \not\models F}\ (\textbf{fresh}\ v \in U)$$

<u>exists</u>

$$\frac{U,I,\sigma \models \exists x.F}{U,I,\sigma[x \mapsto v] \models F}\ (\textbf{fresh}\ v \in U) \qquad \frac{U,I,\sigma \not\models \exists x.F}{U,I,\sigma[x \mapsto v] \not\models F}\ (\textbf{any}\ v \in U)$$

<u>contr</u>