First Order Logic vs Propositional Logic

First Order Logic extends Propositional Logic with:
- Non-boolean constants
- Variables
- Functions and relations (or predicates, more generally)
- Quantification of variables

Sample first order formula:
\[ \forall x. \exists y. x < y \land y \leq x + 1 \]

Reference: Peled, Software Reliability Methods, Chapter 3

Terms over Signature

Terms \( t \) are expressions built over a signature \((V, F, af, R, ar)\):
\[ t ::= v \quad v \in V \\
| f (t_1, \ldots, t_n) \quad f \in F \text{ and } n = af(f) \]

- Example: \( \text{add}(1, \text{abs}(x)) \) where \( \text{add}, \text{abs}, 1 \in F; \ x \in V \)
- For constant \( c \) write \( c \) instead of \( c(\ ) \)
- Will write \( s = t \) instead of \( s(= t) \)
  - Similarly for other common infixes (e.g. \( +, -, \ast, \leq, \ldots \))

Signatures

Start with signature:
\[ G = (V, F, af, R, ar) \]

- \( V \) a countably infinite set of variables
- \( F \) finite set of function symbols
- \( af : F \to \mathbb{N} \) gives the arity, the number of arguments for each function
- \( Constant \ c \) is a function symbol of arity \( 0 \) (\( af(c) = 0 \))
- \( R \) finite set of relation symbols
- \( ar : R \to \mathbb{N} \), the arity for each relation symbol
  - Assumes \( \in \in R \) and \( ar(=) = 2 \)

Structures

Meaning of terms starts with a structure:
\[ S = (G, D, F, \phi, R, \rho) \]

where:
- \( G = (V, F, af, R, ar) \) a signature,
- \( D \) and domain on interpretation
- \( F \) set of functions over \( D \): \( F \subseteq \bigcup_{n \geq 0} D^n \to D \)
  - Note: \( F \) can contain elements of \( D \) since \( D = (D^0 \to D) \)
- \( \phi : F \to F \) where if \( \phi(f) \in (D^n \to D) \) then \( n = af(f) \)
- \( R \) set of relations over \( D \): \( R \subseteq \bigcup_{n \geq 1} P(D^n) \)
- \( \rho : R \to R \) where if \( \rho(r) \subseteq D^n \) then \( n = ar(r) \)

Assignments

\( V \) set of variables, \( D \) domain of interpretation

An assignment is a function \( a : V \to D \)

Example:
\[ V = \{ w, x, y, z \} \]
\[ a = \{ w \mapsto 3.14, x \mapsto -2.75, y \mapsto 13.9, z \mapsto -25.3 \} \]

- Assignment is a fixed association of values to variables; not "update-able"
Interpretation of Terms

Fix structure $S = (G, D, F, \phi, R, \rho)$ where $G = (V, F, af, R, ar)$

For given assignment $a : V \rightarrow D$, the interpretation $T_a$ of a term $t$ is defined by structural induction on terms:

- $T_a(v) = a(v)$ for $v \in V$
- $T_a(f(t_1, \ldots, t_n)) = o(f)(T_a(t_1), \ldots, T_a(t_n))$

Example of Interpretation

- $V = \{w, x, y, z\}$, $D = \mathbb{R}$
- 1. add, abs $\in F$, constant 1, and functions (in $F$) for addition and absolute value respectively
- $a = \{w \mapsto 3.14, x \mapsto -2.75, y \mapsto 13.9, z \mapsto -25.3\}$

$T_a(\text{add}(1, \text{abs}(x))) = (T_a(1)) + T_a(\text{abs}(x))$

= $1.0 + |T_a(x)|$

= $1.0 + a(x)$

= $1.0 + |1.25|$

= $1.0 + 2.75$

= $3.75$

First-Order Formulae

First-order formulae built from terms using relations, logical connectives, quantifiers:

$\text{form ::= true} \mid \text{false} \mid \text{r}(t_1, \ldots, t_n) \mid \text{form} \land \text{form} \mid \text{form} \lor \text{form} \mid \text{form} \Rightarrow \text{form} \mid \forall \text{v.form} \mid \exists \text{v.form}$

Note: Scope of quantifiers as far to right as possible

$\forall x (x > y) \land (2 > x)$ same as $\forall x (x > y) \land (2 > x)$

not same as $\forall x (x > y) \land (2 > x)$

Free Variables

Informally: free variables of an expression are variables that have an occurrence in an expression that is not bound. Written $fv(e)$ for expression $e$

Free variables of terms defined by structural induction over terms; written

- $fv(x) = \{x\}$
- $fv(f(t_1, \ldots, t_n)) = \bigcup_{i=1,..,n} fv(t_i)$

Note:

- Free variables of term just variables occurring in term; no bound variables
- No free variables in constants
- Example: $fv(\text{add}(1, \text{abs}(x))) = \{x\}$

Subformulae

A subformula of formula $\psi$ is a formula that occurs in $\psi$

- More rigorous definition by structural induction on formulae
- $\psi$ subformula of $\psi$
- Use proper subformula to exclude $\psi$

Write $\bigwedge_{i=1,..,n} \psi_i$ for $\psi_1 \land \ldots \land \psi_n$

- $\psi_i$ called a conjunct

Write $\bigvee_{i=1,..,n} \psi_i$ for $\psi_1 \lor \ldots \lor \psi_n$

- $\psi_i$ called a disjunct

Free Variables: Formulae

Defined by structural induction on formulae; uses $fv$ on terms

- $fv(\text{true}) = fv(\text{false}) = \{\}\$
- $fv(\text{r}(t_1, \ldots, t_n)) = \bigcup_{i=1,..,n} fv(t_i)$
- $fv(\psi_1 \land \psi_2) = fv(\psi_1 \lor \psi_2) = fv(\psi_1 \Rightarrow \psi_2) = fv(\forall \text{v.form})$

Variable occurrence at quantifier is binding occurrence

Occurrence that is not free and not binding is a bound occurrence

Example:

$fv(x > 3 \land (\exists y. (\forall z. z \geq (y - x)) \lor (z \geq y))) = \{x, z\}$
Interpretation of Formulae

Fix structure $S = (\mathcal{G}, \mathcal{D}, F, \phi, R, \rho)$ where $\mathcal{G} = (V, F, a, R, a)$

For given assignment $a: V \rightarrow D$, the interpretation $M_a$ of a formula $\psi$ assigning a value in $\{T, F\}$ is defined by structural induction on formulae:

$M_a(\text{true}) = T$

$M_a(\text{false}) = F$

$M_a(r(t_1, \ldots, t_n)) = \rho(r)(T_a(t_1), \ldots, T_a(t_n))$

$M_a(\psi \rightarrow \phi) = T$ if $M_a(\psi) = F$ and $M_a(\neg \psi) = F$ if $M_a(\psi) = T$

$M_a(\psi \vee \phi) = T$ if $M_a(\psi) = T$ or $M_a(\phi) = T$

$M_a(\psi \wedge \phi) = T$ if $M_a(\psi) = T$ and $M_a(\phi) = T$

$M_a(\phi) = F$ otherwise
Interpretation of Formulae

Fix structure $S = (G, D, F, \phi, R, \rho)$ where $G = (V, F, af, R, ar)$

For given assignment $a : V \to D$, the interpretation $M_a$ of a formula $\psi$ assigning a value in $\{T, F\}$ is defined by structural induction on formulae:

- $M_a(\text{true}) = T$
- $M_a(\text{false}) = F$
- $M_a(r(t_1, \ldots, t_n)) = \rho(r)(T_a(t_1), \ldots, T_a(t_n))$
- $M_a(\psi) = M_a(\phi)$
- $M_a(\neg \psi) = T$ if $M_a(\psi) = F$ and $M_a(\neg \psi) = F$ if $M_a(\psi) = T$
- $M_a(\psi_1 \land \psi_2) = T$ if $M_a(\psi_1) = T$ and $M_a(\psi_2) = T$, and $M_a(\psi_1 \land \psi_2) = F$ otherwise
- $M_a(\psi_1 \lor \psi_2) = T$ if $M_a(\psi_1) = T$ or $M_a(\psi_2) = T$, and $M_a(\psi_1 \lor \psi_2) = F$ otherwise
- $M_a(\psi_1 \Rightarrow \psi_2) = T$ if $M_a(\psi_1) = F$ or $M_a(\psi_2) = T$, and $M_a(\psi_1 \Rightarrow \psi_2) = F$ otherwise
- $M_a(v \mapsto d) = \{ d \text{ if } w = v \}
\begin{cases} a(w) \text{ if } w \neq v \end{cases}$

Interpretation of Formulae

Fix structure $S = (G, D, F, \phi, R, \rho)$ where $G = (V, F, af, R, ar)$

For given assignment $a : V \to D$, the interpretation $M_a$ of a formula $\psi$ assigning a value in $\{T, F\}$ is defined by structural induction on formulae:

- $M_a(\text{true}) = T$
- $M_a(\text{false}) = F$
- $M_a(r(t_1, \ldots, t_n)) = \rho(r)(T_a(t_1), \ldots, T_a(t_n))$
- $M_a(\psi) = M_a(\phi)$
- $M_a(\neg \psi) = T$ if $M_a(\psi) = F$ and $M_a(\neg \psi) = F$ if $M_a(\psi) = T$
- $M_a(\psi_1 \land \psi_2) = T$ if $M_a(\psi_1) = T$ and $M_a(\psi_2) = T$, and $M_a(\psi_1 \land \psi_2) = F$ otherwise
- $M_a(\psi_1 \lor \psi_2) = T$ if $M_a(\psi_1) = T$ or $M_a(\psi_2) = T$, and $M_a(\psi_1 \lor \psi_2) = F$ otherwise
- $M_a(\psi_1 \Rightarrow \psi_2) = T$ if $M_a(\psi_1) = F$ or $M_a(\psi_2) = T$, and $M_a(\psi_1 \Rightarrow \psi_2) = F$ otherwise
- $M_a(v \mapsto d) = \{ d \text{ if } w = v \}
\begin{cases} a(w) \text{ if } w \neq v \end{cases}$
### Modeling First-order Formulae

Given structure $S = (G, D, F, \phi, R, \rho)$ where $G = (V, F, a, R, a)$

- $(S, M)$ model for first-order language over signature $G$
- Truth of formulae in language over signature $G$ depends on structure $S$
- Assignment $a$ models $\psi$, or $a$ satisfies $\psi$, or $a \models^S \psi$ if $M_a(\psi) = T$
- $\psi$ is valid for $S$ if $a \models^S \psi$ for some $a$
- $S$ is a model of $\psi$, written $\models^S \psi$ if every assignment for $S$ satisfies $\psi$.
- $\psi$ is valid, or a tautology if $\psi$ valid for every mode. Write $\models \psi$
- $\psi_1$ logically equivalent to $\psi_2$ if for all structures $S$ and assignments $a$,

### Examples

- Assignment $\{x \mapsto 0\}$ satisfies $\exists y. x < y$ valid in $[0, 1]$
- $\{x \mapsto 1\}$ doesn’t
- $\forall x. \exists y. x < y$ valid in $\mathbb{N}$ and $\mathbb{R}$, but not interval $[0, 1]$
- $((\exists x. \forall y. (y \leq x)) \Rightarrow (\forall y. \exists x. (y \leq x)))$ tautology
- Why?

### Sample Tautologies

All instances of propositional tautologies

$$\models (\exists y. \forall y. (y \leq x)) \Rightarrow (\forall y. \exists x. (y \leq x))$$

$$\models (((\forall x. \forall y. \psi) \Rightarrow (\forall y. \forall x. \psi)))$$

$$\models (((\forall x. \psi) \Rightarrow (\exists x. \psi)))$$
Substitution in Terms

Example:

All instances of propositional tautologies

\[ \vdash (\exists x. \forall y. (y \leq x)) \Rightarrow (\forall y. \exists x. (y \leq x)) \]

\[ \vdash ((\forall y. (y < x)) \Rightarrow (\forall y. (y < x))) \]

\[ \vdash ((\forall x. (\exists y. (y \leq x))) \Rightarrow (\exists y. (\exists x. (y \leq x)))) \]

\[ \vdash (\forall x. \forall y. (x = y)) \Rightarrow (\exists y. (\exists x. (x = y))) \]

\[ \vdash (\exists x. (\forall y. (y < x))) \Rightarrow (\forall y. (\forall x. (y < x))) \]

\[ (\exists x. (\forall y. (y < x))) \Rightarrow (\forall y. (\forall x. (y < x))) \]

\[ (\exists x. (\forall y. (y < x))) \Rightarrow (\forall y. (\forall x. (y < x))) \]

Syntactic Substitution versus Assignment Update

- When interpreting universal quantification \((\forall x. \psi)\), wanted to check interpretation of every instance of \(\psi\) where \(v\) was replaced by element of semantic domain \(D\)
- How: semantically - interpret \(\psi\) with assignment updated by \(v \mapsto d\) for every \(d \in D\)
- Syntactically?
- Answer: substitution

Substitution in Formulae: Problems

Want to define by structural induction, similar to terms

- Substitution only replaces free occurrences of variable
  - Example: \((x > 3 \land (\exists y. (y < x)))\) \(\Rightarrow (x > 3 \land (\exists y. (y < x)))\)
  - \((x > 3 \land (\exists y. (y < x)))\) \(\Rightarrow (x > 3 \land (\exists y. (y < x)))\)

- Need to avoid free variable capture

Example Problem:

\[ (x > 3 \land (\exists y. (y < x))) \Rightarrow (x > 3 \land (\exists y. (y < x))) \]

\[ (x > 3 \land (\exists y. (y < x))) \Rightarrow (x > 3 \land (\exists y. (y < x))) \]
Substitution in Formulae

Examples

\[(x > 3 \land (\exists y. (\forall z. z \geq (y - x)) \lor (z \geq y)))[x + y/z] \text{ not defined}\]

\[(x > 3 \land (\exists w. (\forall z. z \geq (w - x)) \lor (z \geq w)))[x + y/z] =\]

\[(x > 3 \land (\exists w. (\forall z. z \geq (w - x)) \lor ((x + y) \geq y)))\]

Substitution in Formulae: Two Approaches

- When quantifier would capture free variable of redex, can’t substitute in formula as is
- Solution 1: Make substitution partial function – undefined in this case
- Solution 2: Define equivalence relation based on renaming bound variables; define substitution on equivalence classes
- Will take Solution 1 here
- Still need definition of equivalence up to renaming bound variables

Renaming by Swapping: Terms

Definition of swapping of two variables in a term \(t[x \leftrightarrow y]\) by structural induction on \(t\):

- \(x[x \leftrightarrow y] = y\) and \(y[x \leftrightarrow y] = x\)
- \(z[x \leftrightarrow y] = z\) for \(z\) a variable, \(z \neq x, z \neq y\)
- \(f(t_1,\ldots, t_n)[x \leftrightarrow y] = f(t_1[x \leftrightarrow y],\ldots, t_n[x \leftrightarrow y])\)

Examples:

\[\text{add}(1, \text{add}(x, y))[x \leftrightarrow y] = \text{add}(1, \text{add}(y, x))\]

\[\text{add}(1, \text{add}(x, y))[x \leftrightarrow z] = \text{add}(1, \text{add}(z, y))\]
Renaming by Swapping: Terms

Proof.

By structural induction on terms, it suffices to show theorem for the case where $t$ variable, and case $t = f(t_1, \ldots, t_n)$, assuming result for $t_1, \ldots, t_n$

- Case: $t = x$. Then $T(t[x \leftrightarrow y]) = T(t[y \rightarrow x]) = a(y)$ if $\psi$.
  - $T(t(y \rightarrow x)) = T(t(y))$.
  - Subcase: $t = y$. Then $T(t[x \leftrightarrow y]) = T(t(y[x \leftrightarrow y]))$.
  - Subcase: $t = z$ variable, $z \not= x$ and $z \not= y$. Then $T(t[x \leftrightarrow y]) = T(t(z[x \leftrightarrow y]))$.

- Case: $t = f(t_1, \ldots, t_n)$. Assume $T(t_i[x \leftrightarrow y]) = T(t_i(x))$ for $i = 1, \ldots, n$. Then
  - $T(t[x \leftrightarrow y]) = T(t(f(t_1, \ldots, t_n)[x \leftrightarrow y]))$.
  - $T(t(f(t_1[x \leftrightarrow y], \ldots, t_n[x \leftrightarrow y])))$.
  - $T(t(f(t_1, \ldots, t_n)[x \leftrightarrow y]))$.
  - $T(t_1(f(t_1, \ldots, t_n)[x \leftrightarrow y]))$.
  - $T(t_1(f(t_1, \ldots, t_n)[x \leftrightarrow y]))$.
  - $T(t(k[x \leftrightarrow y]))$.

Renaming by Swapping: Formulae

Define the **swapping** of two variables in a formula $\psi[x \leftrightarrow y]$ by structural induction, using swapping on terms:

- $true[x \leftrightarrow y] = true$, $false[x \leftrightarrow y] = false$.
- $r(t_1, \ldots, t_n)[x \leftrightarrow y] = r(t_1[x \leftrightarrow y], \ldots, t_n[x \leftrightarrow y])$.
- $\forall \in \{\lor, \land\}$ for $Q \in \{\forall, \exists\}$.
- $\exists \in \{\lor, \land\}$ for $Q \in \{\forall, \exists\}$.
- $\forall z, \psi(x, y)[x \leftrightarrow y] = \exists z, \psi(x, y)[x \leftrightarrow y]$.

$\alpha$-equivalence

- $\psi \equiv \psi'$
- If $\psi_1 \equiv \psi_2$ then $\psi_2 \equiv \psi_1$.
- If $\psi_1 \equiv \psi_2$ and $\psi_2 \equiv \psi_3$ then $\psi_1 \equiv \psi_3$.
- If $x \not\in \text{fv}(\psi)$ and $y \not\in \text{fv}(\psi)$ then $\psi \equiv \psi[x \leftrightarrow y]$.
- If $\psi_i \equiv \psi_i'$ for $i = 1, 2$ then
  - $\psi_1 \equiv \psi_2'$
  - $\psi_2 \equiv \psi_1'$
  - $\psi_1 \equiv \psi_1 \equiv \psi_2 \equiv \psi_2 \equiv \psi_1 \equiv \psi_2 \equiv \psi_1 \equiv \psi_2$.
- $Q \in \{\forall, \exists\}$ for $\psi_1 \equiv \psi_2$, $\psi_2 \equiv \psi_3$.

Examples

- $x > 3 \land (\exists y. (\forall z. (x \not= y) \lor (z \not= y)))$.
- $x > 3 \land (\exists y. (\forall z. (x \not= y) \lor (z \not= y)))$.
- $x > 3 \land (\exists y. (\forall z. (x \not= y) \lor (z \not= y)))$.
- $x > 3 \land (\exists y. (\forall z. (x \not= y) \lor (z \not= y)))$.

$\alpha$-equivalence: Example

- $x > 3 \land (\exists y. (\forall z. (x \not= y) \lor (z \not= y)))$.
- $x > 3 \land (\exists y. (\forall z. (x \not= y) \lor (z \not= y)))$.
- $x > 3 \land (\exists y. (\forall z. (x \not= y) \lor (z \not= y)))$.
Proof Rules

Natural Deduction rules:
All rules from Propositional Logic included

\( \Gamma \vdash \psi[t/x] \) \hspace{1cm} \( \Gamma \vdash \exists x.\psi \)

\( \Gamma \vdash \psi[y/x] \quad y \notin (fv(\psi) \setminus \{x\}) \cup \bigcup \psi' \in \Gamma \) \hspace{1cm} \( \Gamma \vdash \forall x.\psi \)

Elsa L. Gunter  
CS477 Formal Software Development Methods  
February 21, 2014