Signatures

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

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

Terms over Signature

Terms \( t \) are expressions built over a signature \((V, F, \text{af}, R, \text{ar})\)

\[ t ::= v \quad \text{if } v \in V \]
\[ f(t_1, \ldots, t_n) \quad \text{if } f \in F \text{ and } n = \text{af}(f) \]

- 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. \( +, -, \ast, \ldots \))

Structures

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

where

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

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 \( v \in V \)
- \( T_a(f(t_1, \ldots, t_n)) = \sigma(f)(T_a(t_1), \ldots, T_a(t_n)) \)

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
- \( a = \{w \mapsto 3.14, x \mapsto -2.75, y \mapsto 13.9, z \mapsto -25.3\} \)

\[ T_a(add(1, abs(x))) = (T_a(1)) + (T_a(abs(x))) = 1.0 + (T_a(abs(x))) = 1.0 + |a(x)| = 1.0 + |−2.75| = 1.0 + 2.75 = 3.75 \]

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

Interpretation of Formulae

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 \( M_a \) of a formula \( \psi \) assigning a value in \( \{T, F\} \) is defined by structural induction on formulae:
- \( M_a(\text{true}) = T \)
- \( M_a(\text{false}) = F \)
Interpretation of Formulae

Fix structure $S = (G, D, F, φ, R, ρ)$ where $G = (V, F, af, R, ar)$

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

- $M_a(\text{true}) = T$
- $M_a(\text{false}) = F$
- $M_a(\langle t_1, \ldots, t_n \rangle) = T$ if $(T_a(t_1), \ldots, T_a(t_n)) \in ρ(r)$ and $M_a(\langle t_1, \ldots, t_n \rangle) = F$ if $(T_a(t_1), \ldots, T_a(t_n)) \not\in ρ(r)$
- $M_a(⟨⟩) = M_a(ψ)$
- $M_a(¬ψ) = T$ if $M_a(ψ) = F$ and $M_a(¬ψ) = F$ if $M_a(ψ) = T$

Fix structure $S = (G, D, F, φ, R, ρ)$ where $G = (V, F, af, R, ar)$

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

- $M_a(\text{true}) = T$
- $M_a(\text{false}) = F$
- $M_a(\langle t_1, \ldots, t_n \rangle) = T$ if $(T_a(t_1), \ldots, T_a(t_n)) \in ρ(r)$ and $M_a(\langle t_1, \ldots, t_n \rangle) = F$ if $(T_a(t_1), \ldots, T_a(t_n)) \not\in ρ(r)$
- $M_a(⟨⟩) = M_a(ψ)$
- $M_a(¬ψ) = T$ if $M_a(ψ) = F$ and $M_a(¬ψ) = F$ if $M_a(ψ) = T$
- $M_a(ψ_1 \land ψ_2) = T$ if $M_a(ψ_1) = T$ and $M_a(ψ_2) = T$, and $M_a(ψ_1 \land ψ_2) = F$ otherwise
- $M_a(ψ_1 \lor ψ_2) = T$ if $M_a(ψ_1) = T$ or $M_a(ψ_2) = T$, and $M_a(ψ_1 \lor ψ_2) = F$ otherwise
- $M_a(ψ_1 \rightarrow ψ_2) = T$ if $M_a(ψ_1) = F$ or $M_a(ψ_2) = T$, and $M_a(ψ_1 \rightarrow ψ_2) = F$ otherwise
Interpretation of Formulae

Fix structure \( S = (G, D, F, \phi, R, \rho) \) where \( G = (V, F, a_f, R, a_r) \)

Let
\[
a + [v \mapsto d](w) = \begin{cases} 
  d & \text{if } w = v \\
  a(w) & \text{if } w \neq v 
\end{cases}
\]

- \( M_a(\forall v. \psi) = T \) if for every \( d \in D \) we have \( M_a+[v \mapsto d](\psi) = T \), and \( M_a(\forall v. \psi) = F \) otherwise
- \( M_a(\exists v. \psi) = T \) if there exists \( d \in D \) such that \( M_a+[v \mapsto d](\psi) = T \), and \( M_a(\exists v. \psi) = F \) otherwise

Modeling First-order Formulae

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

- \( (S, M) \) 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 \( a \), \( a \models^S \psi_1 \iff a \models^S \psi_2 \)

Examples

- Assignment \( \{x \mapsto 0\} \) satisfies \( \exists y . x < y \) valid in interval \([0, 1]\); assignment \( \{x \mapsto 1\} \) 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

\(|= \exists x. \forall y. (y \leq x) \Rightarrow (\forall y. \exists x. (y \leq x))|

\(|= ((\forall x. \forall y. \psi) \land (\forall x. \psi_2)) \Rightarrow ((\exists x. \psi_1) \land (\exists x. \psi_2))|

Elsa L. Gunter
CS477 Formal Software Dev Methods
February 16, 2018 15 / 33
Free Variables: Terms

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

Free variables of terms defined by structural induction over terms; written
- \( \text{fv}(x) = \{x\} \)
- \( \text{fv}(t_1, \ldots, t_n) = \bigcup_{i=1}^{n} \text{fv}(t_i) \)

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

Free Variables: Formulae

Defined by structural induction on formulae; uses \( \text{fv} \) on terms
- \( \text{fv}((\text{true}) = \text{fv}(\text{false}) = \{\} \)
- \( \text{fv}(t_1, \ldots, t_n) = \bigcup_{i=1}^{n} \text{fv}(t_i) \)
- \( \text{fv}(\phi \land \psi) = \text{fv}(\phi) \cup \text{fv}(\psi) = \text{fv}(\phi \lor \psi) = \text{fv}(\phi \Rightarrow \psi) = \text{fv}(\lnot \phi) = \text{fv}(\exists x. \psi) = \{x\} \)

Variable occurrence at quantifier are binding occurrence
Occurrence that is not free and not binding is a bound occurrence

Example:
- \( \text{fv}(x > 3 \land (\exists y. (\forall z. z \geq (y - x)) \lor (x + y \geq y))) = \{x, z\} \)

Free Variables, Assignments and Interpretation

Theorem
Assume given structure \( S = (G, D, F, \phi, R, \rho) \), term \( t \) over \( G \), and a and \( b \) assignments. If for every \( x \in \text{fv}(t) \) we have \( a(x) = b(x) \) then \( T_a(t) = T_b(a) \).

Theorem
Assume given structure \( S = (G, D, F, \phi, R, \rho) \), formula \( \psi \) over \( G \), and a and \( b \) assignments. If for every \( x \in \text{fv}(\psi) \) we have \( a(x) = b(x) \) then \( M_a(\psi) = M_b(\psi) \).

Substitution in Terms

- Substitution of term \( t \) for variable \( x \) in term \( s \) (written \( s[t/x] \)) gotten by replacing every instance of \( x \) in \( s \) by \( t \)
  - \( x \) called redex; \( t \) called residue
  - Yields instance of \( s \)

Formally defined by structural induction on terms:
- \( x[t/x] = t \)
- \( y[t/x] = y \) for variable \( y \) where \( y \neq x \)
- \( f(t_1, \ldots, t_n)[t/x] = f(t_1[t/x], \ldots, t_n[t/x]) \)

Example: \( (\text{add}(1, \text{abs}(x)))[\text{add}(x, y)/x] = \text{add}(1, \text{abs}(\text{add}(x, y))) \)

Substitution in Formulae: Problems

- Want to define by structural induction, similar to terms
  - Quantifiers must be handled with care
  - Substitution only replaces free occurrences of variable

Example:
- \( (x > 3 \land (\exists y. (\forall z. z \geq (y - x)) \lor (x + y \geq y)))\) yields \( (x > 3 \land (\exists y. (\forall z. z \geq (y - x)) \lor (x + 2 \geq y))) \)

Need to avoid free variable capture

Example Problem:
- \( (x > 3 \land (\exists y. (\forall z. z \geq (y - x)) \lor (x + y \geq y))) \neq \)
  \( (x > 3 \land (\exists y. (\forall z. z \geq (y - x)) \lor (x + y \geq y))) \)
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

Substitution in Formulae

- Defined by structural induction; uses substitution in terms
- Read equations below as saying left is not defined if any expression on right not defined
- true[t/x] = true, false[t/x] = false
- \( r(t_1, \ldots, t_n)[t/x] = r((t_1[t/x], \ldots, t_n[t/x])) \)
- \( (\psi \land \theta)[t/x] = (\psi[t/x]) \land (\theta[t/x]) \)
- \( (\phi \lor \psi)[t/x] = (\phi[t/x]) \lor (\psi[t/x]) \) for \( \phi, \psi \in \{\land, \lor, \land \} \)
- \( (Q \cdot x \cdot \psi)[t/x] = Q \cdot x \cdot \psi \) for \( Q \in \{\forall, \exists\} \)
- \( (Q \cdot y \cdot \psi)[t/x] = Q \cdot y \cdot (\psi[t/x]) \) if \( x \neq y \) and \( y \notin \text{fv}(t) \) for \( Q \in \{\forall, \exists\} \)
- \( (Q \cdot y \cdot \psi)[t/x] \) not defined if \( x \neq y \) and \( y \notin \text{fv}(t) \) for \( Q \in \{\forall, \exists\} \)

Renaming by Swapping: Terms

Define the **swapping** of two variables in a term \( t[x \leftrightarrow y] \) by structural induction on terms:
- \( x[x \leftrightarrow y] = y \) 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:
- \( \text{add}(1, \text{abs}(\text{add}(x, y)))[x \leftrightarrow y] = \text{add}(1, \text{abs}(\text{add}(y, x))) \)
- \( \text{add}(1, \text{abs}(\text{add}(x, y)))[x \leftrightarrow z] = \text{add}(1, \text{abs}(\text{add}(z, y))) \)
Theorem

Proof.

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

- Case: \( t \) variable
  - Subcase: \( t = x \). Then \( T_a(x[x ↔ y]) = T_a(y) = a(y) \) and
    - \( T_a(x) = b(x) = a(x) \)
    - \( T_a(y) = b(y) = a(x) \)
    - \( T_a(t[x ↔ y]) = T_a(t) \)

- Subcase: \( t = y \). Then \( T_a(y[y ↔ x]) = T_a(x) = a(x) \) and
  - \( T_a(y) = b(x) = a(x) \)
  - \( T_a(t[x ↔ y]) = T_a(t) \)

- Case: \( t \) subterm of \( \psi \).
  - Subcase: \( \psi \) \( \psi \)
  - \( \psi \)
  - \( \psi \)
  - \( \psi \)
  - \( \psi \)

- Case: \( t \) subterm of \( \psi \).
  - Subcase: \( \psi \) \( \psi \)
  - \( \psi \)
  - \( \psi \)
  - \( \psi \)

- Case: \( t \) subterm of \( \psi \).
  - Subcase: \( \psi \) \( \psi \)
  - \( \psi \)
  - \( \psi \)
  - \( \psi \)

Examples

\( (x > 3 \land (\exists y. (\forall z. z \geq (y - x)) \lor (z \geq y))) \)

Theorem

Assume given structure \( S = (G, D, F, \phi, R, \rho) \), variables \( x \) and \( y \), formula \( \psi \) over \( G \), and a assignment. If \( x \notin fv(t) \) and \( y \notin fv(t) \) then \( \psi \equiv \psi \)

\( (x > 3 \land (\exists y. (\forall z. z \geq (y - x)) \lor (z \geq y))) \)

\( (x > 3 \land (\exists w. (\forall y. y \geq (w - x)) \lor (y \geq w))) \)

\( (x > 3 \land (\exists w. (\forall y. y \geq (w - x)) \lor (y \geq w))) \)

\( (x > 3 \land (\exists w. (\forall y. y \geq (w - x)) \lor (y \geq w))) \)

\( (x > 3 \land (\exists w. (\forall y. y \geq (w - x)) \lor (y \geq w))) \)

\( (x > 3 \land (\exists w. (\forall y. y \geq (w - x)) \lor (y \geq w))) \)

\( (x > 3 \land (\exists w. (\forall y. y \geq (w - x)) \lor (y \geq w))) \)

\( (x > 3 \land (\exists w. (\forall y. y \geq (w - x)) \lor (y \geq w))) \)

\( (x > 3 \land (\exists w. (\forall y. y \geq (w - x)) \lor (y \geq w))) \)

\( (x > 3 \land (\exists w. (\forall y. y \geq (w - x)) \lor (y \geq w))) \)

\( (x > 3 \land (\exists w. (\forall y. y \geq (w - x)) \lor (y \geq w))) \)
Proof Rules

Will give Sequent version of Natural Deduction rules
All rules from Propositional Logic included

\[ \frac{\Gamma \vdash \psi[t/x]}{\Gamma \vdash \exists x.\psi} \quad \text{Ex I} \]
provided \( \psi \equiv \psi' \)

\[ \frac{\Gamma \vdash \exists x.\psi \quad \Gamma \vdash \{\psi[y/x]\}\vdash \varphi}{\Gamma \vdash \varphi} \quad \text{Ex E} \]
provided \( y \notin \text{fv}(\varphi) \cup (\text{fv}(\psi) \setminus \{x\}) \cup \bigcup_{\psi' \in \Gamma} \text{fv}(\psi') \)

\[ \frac{\Gamma \vdash \psi[y/x]}{\Gamma \vdash \forall x.\psi} \quad \text{All I} \]
provided \( y \notin \{\text{fv}(\psi) \setminus \{x\}) \cup \bigcup_{\psi' \in \Gamma} \text{fv}(\psi') \)

\[ \frac{\Gamma \vdash \forall x.\psi \quad \Gamma \vdash \{\psi'[t/x]\}\vdash \varphi}{\Gamma \vdash \varphi} \quad \text{All E} \]
provided \( \psi \equiv \psi' \)

Example

Show

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

Example

Show

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

Example

Show

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

Example

Show

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

Show

Hyp
{∃x.∀y.x ≤ y; ∀y, z ≤ y} ⊢ (∃x.∀y.x ≤ y) ⇒ (∀x.∃y.y ≤ x)

Elsa L. Gunter
Example of Failure

Let’s try to show 3

\[
\{\forall x. \exists y. y \leq x\} \vdash \forall y. z \leq y
\]

Ex I

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists x. \exists y. x \leq y
\]

Imp I

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

Example of Failure

Let’s try to show 4

\[
\{\forall x. \exists y. y \leq x\} \vdash z \leq x
\]

All I

\[
\{\forall x. \exists y. y \leq x\} \vdash y. z \leq y
\]

Ex I

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists x. \forall y. x \leq y
\]

Imp I

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

Example of Failure

Let’s try to show 5

\[
\{\forall x. \exists y. y \leq x\} \vdash \forall x. \exists y. y \leq x
\]

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists y. y \leq x
\]

All E

\[
\{\forall x. \exists y. y \leq x\} \vdash z \leq x
\]

All I

\[
\{\forall x. \exists y. y \leq x\} \vdash \forall y. z \leq y
\]

Ex I

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists x. \forall y. x \leq y
\]

Imp I

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

Example of Failure

Let’s try to show 6

\[
\{\forall x. \exists y. y \leq x\} \vdash \forall x. \exists y. y \leq x
\]

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists y. y \leq x
\]

All E

\[
\{\forall x. \exists y. y \leq x\} \vdash z \leq x
\]

All I

\[
\{\forall x. \exists y. y \leq x\} \vdash \forall y. z \leq y
\]

Ex I

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists x. \forall y. x \leq y
\]

Imp I

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

Example of Failure

Let’s try to show 7

\[
\{\forall x. \exists y. y \leq x\} \vdash \forall x. \exists y. y \leq x
\]

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists y. y \leq x
\]

All E

\[
\{\forall x. \exists y. y \leq x\} \vdash \forall y. z \leq y
\]

Ex I

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists x. \forall y. x \leq y
\]

Imp I

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

Example of Failure

Let’s try to show 8

\[
\{\forall x. \exists y. y \leq x\} \vdash \forall x. \exists y. y \leq x
\]

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists y. y \leq x
\]

All E

\[
\{\forall x. \exists y. y \leq x\} \vdash z \leq x
\]

All I

\[
\{\forall x. \exists y. y \leq x\} \vdash \forall y. z \leq y
\]

Ex I

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists x. \forall y. x \leq y
\]

Imp I

\[
\{\} \vdash (\forall x. \exists y. y \leq x) \Rightarrow (\exists x. \forall y. x \leq y)
\]
Example of Failure

Let’s try to show

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists x. \forall y. x \leq y
\]

Hyp

\[
\{\forall x. \exists y. y \leq x\} \vdash \forall x. \exists y. y \leq x
\]

Ex E

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists x. \forall y. x \leq y
\]

All E

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

Example of Failure

Let’s try to show

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists x. \forall y. x \leq y
\]

Hyp

\[
\{\forall x. \exists y. y \leq x\} \vdash \forall y. z \leq y
\]

Ex I

\[
\{\forall x. \exists y. y \leq x\} \vdash \exists x. \forall y. x \leq y
\]

Imp I

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

Floyd-Hoare Logic

- Also called Axiomatic Semantics
- Based on formal logic (first order predicate calculus)
- Logical system built from axioms and inference rules
- Mainly suited to simple imperative programming languages
- Ideas applicable quite broadly

Floyd-Hoare Logic

- Used to formally prove a property (post-condition) of the state (the values of the program variables) after the execution of program, assuming another property (pre-condition) of the state holds before execution

Floyd-Hoare Logic

- Goal: Derive statements of form

\[
\{P\} C \{Q\}
\]

- \(P\), \(Q\) logical statements about state, \(P\) precondition, \(Q\) postcondition, \(C\) program
- Example:

\[
\{x = 1\} \ x := x + 1 \ {x = 2}\]

Floyd-Hoare Logic

- Approach: For each type of language statement, give an axiom or inference rule stating how to derive assertions of form

\[
\{P\} C \{Q\}
\]

where \(C\) is a statement of that type
- Compose axioms and inference rules to build proofs for complex programs
Partial vs Total Correctness

- An expression $\{P\} C \{Q\}$ is a partial correctness statement.
- For total correctness must also prove that $C$ terminates (i.e. doesn't run forever).
- Written: $\{P\} C \{Q\}$
- Will only consider partial correctness here.

Simple Imperative Language

- We will give rules for simple imperative language
  
  $\langle \text{command} \rangle ::= \langle \text{variable} \rangle := \langle \text{term} \rangle$
  | $\langle \text{command} \rangle ; \ldots ; \langle \text{command} \rangle$
  | $\text{if} \langle \text{statement} \rangle \text{then} \langle \text{command} \rangle \text{else} \langle \text{command} \rangle$
  | $\text{while} \langle \text{statement} \rangle \text{do} \langle \text{command} \rangle$

- Could add more features, like for-loops.

Substitution

- Notation: $P[e/v]$ (sometimes $P[v \rightarrow e]$)
- Meaning: Replace every $v$ in $P$ by $e$
- Example:
  
  $\{(x + 2)[y - 1/x] = ((y - 1) + 2)\}$

The Assignment Rule

- Example:
  
  $\{x = 2\} x := y \{x = 2\}$
The Assignment Rule

\[ \{ P[e/x] \} x := e \{ P \} \]

Examples:

\[ \{ y = 2 \} x := y \{ x = 2 \} \]

\[ \{ y = 2 \} x := 2 \{ y = x \} \]

\[ \{ x + 1 = n + 1 \} x := x + 1 \{ x = n + 1 \} \]

\[ \{ 2 = 2 \} x := 2 \{ x = 2 \} \]

The Assignment Rule – Your Turn

- What is the weakest precondition of
  \[ x := x + y \{ x + y = wx \} \]?

\[ \{ (x + y) + y = w(x + y) \} \]

\[ x := x + y \]

\[ \{ x + y = wx \} \]

Precondition Strengthening

\[ (P \Rightarrow P') \{ P' \} C \{ Q \} \]

\[ \{ P \} C \{ Q \} \]

Meaning: If we can show that \( P \) implies \( P' \) (i.e. \( P \Rightarrow P' \)) and we can show that \( \{ P \} C \{ Q \} \), then we know that \( \{ P \} C \{ Q \} \).

\* \( P \) is stronger than \( P' \) means \( P \Rightarrow P' \)

Precondition Strengthening

- Examples:
  \[ x = 3 \Rightarrow x < 7 \{ x < 7 \} x := x + 3 \{ x < 10 \} \]
  \[ \{ x = 3 \} x := x + 3 \{ x < 10 \} \]

  \[ True \Rightarrow (2 = 2) \{ 2 = 2 \} x := 2 \{ x = 2 \} \]
  \[ \{ True \} x := 2 \{ x = 2 \} \]

  \[ x = n \Rightarrow x + 1 = n + 1 \{ x + 1 = n + 1 \} x := x + 1 \{ x = n + 1 \} \]
  \[ \{ x = n \} x := x + 1 \{ x = n + 1 \} \]

Which Inferences Are Correct?

\[ \{ x > 0 \land x < 5 \} x := x * x \{ x < 25 \} \]

\[ \{ x = 3 \} x := x * x \{ x < 25 \} \]

\[ \{ x > 0 \land x < 5 \} x := x * x \{ x < 25 \} \]

\[ \{ x * x < 25 \} x := x * x \{ x < 25 \} \]

\[ \{ x > 0 \land x < 5 \} x := x * x \{ x < 25 \} \]
### Which Inferences Are Correct?

\[
\{x > 0 \land x < 5\} \quad \begin{align*}
x &:= x \times x \quad \{x < 25\} \\
\{x = 3\} &:= x \times x \quad \{x < 25\} \\
\{x > 0 \land x < 5\} &:= x \times x \quad \{x < 25\} \\
\{x \times x < 25\} &:= x \times x \quad \{x < 25\} \\
\{x > 0 \land x < 5\} &:= x \times x \quad \{x < 25\}
\end{align*}
\]

**YES**

\[
\{x > 0 \land x < 5\} \quad \begin{align*}
x &:= x \times x \quad \{x < 25\} \\
\{x = 3\} &:= x \times x \quad \{x < 25\} \\
\{x > 0 \land x < 5\} &:= x \times x \quad \{x < 25\} \\
\{x \times x < 25\} &:= x \times x \quad \{x < 25\} \\
\{x > 0 \land x < 5\} &:= x \times x \quad \{x < 25\}
\end{align*}
\]

**YES**

\[
\{x > 0 \land x < 5\} \quad \begin{align*}
x &:= x \times x \quad \{x < 25\} \\
\{x = 3\} &:= x \times x \quad \{x < 25\} \\
\{x > 0 \land x < 5\} &:= x \times x \quad \{x < 25\} \\
\{x \times x < 25\} &:= x \times x \quad \{x < 25\} \\
\{x > 0 \land x < 5\} &:= x \times x \quad \{x < 25\}
\end{align*}
\]

**NO**

\[
\{x > 0 \land x < 5\} \quad \begin{align*}
x &:= x \times x \quad \{x < 25\} \\
\{x = 3\} &:= x \times x \quad \{x < 25\} \\
\{x > 0 \land x < 5\} &:= x \times x \quad \{x < 25\} \\
\{x \times x < 25\} &:= x \times x \quad \{x < 25\} \\
\{x > 0 \land x < 5\} &:= x \times x \quad \{x < 25\}
\end{align*}
\]

**YES**

\[
\{x > 0 \land x < 5\} \quad \begin{align*}
x &:= x \times x \quad \{x < 25\}
\end{align*}
\]

**YES**

\[
\{x > 0 \land x < 5\} \quad \begin{align*}
x &:= x \times x \quad \{x < 25\}
\end{align*}
\]

**NO**

\[
\{x > 0 \land x < 5\} \quad \begin{align*}
x &:= x \times x \quad \{x < 25\}
\end{align*}
\]

**YES**