Programming Language Semantics
Axiomatic Semantics

The Formal Semantics of Programming Languages
Chapter 6
Motivation

• What do we need in order to prove that the program does what it supposed to do?

• Specify the required behavior

• Compare the behavior with the one obtained by the denotational/operational semantics

• Develop a proof system for showing that the program satisfies a requirement

• Mechanically use the proof system to show correctness

• The meaning of a program is a set of verification rules
Plan

• The basic idea
• An assertion language
• Semantics of assertions
• Proof rules
• An example
• Soundness
• Completeness
• Verification conditions
Example Program

\[ S := 0 \]
\[ N := 1 \]

while \( \neg (N = 101) \) do
  \[ S := S + N ; \]
  \[ N := N + 1 \]
\[ N = 101 \]
\[ S = \sum_{1 \leq m \leq 100} m \]
Example Program

\[ S := 0 \]
\[ \{ S = 0 \} \]
\[ N := 1 \]
\[ \{ S = 0 \land N = 1 \} \]

while \( \neg (N = 101) \) do

\[ S := S + N ; \]
\[ N := N + 1 \]

\[ \{ N = 101 \land S = \sum_{1 \leq m \leq 100} m \} \]
Example Program

\[ S := 0 \]
\[ \{ S = 0 \} \]
\[ N := 1 \]
\[ \{ S = 0 \land N = 1 \} \]

while \( \{ 1 \leq N \leq 101 \land S = \sum_{1 \leq m \leq N-1} m \} \land (N = 101) \) do

\[ S := S + N ; \]
\[ \{ 1 \leq N < 101 \land S = \sum_{1 \leq m \leq N} m \} \]

\[ N := N + 1 \]
\[ \{ N = 101 \land S = \sum_{1 \leq m \leq 100} m \} \]
Partial Correctness

• \{P\}S\{Q\}
  – P and Q are assertions
    (extensions of Boolean expressions)
  – S is a statement
  – For all states \(\sigma\) which satisfies P, if the execution of S from state \(\sigma\) terminates in state \(\sigma'\), then \(\sigma'\) satisfies Q

• \{true\}while true do skip\{false\}
Total Correctness

• $[P]S[Q]$
  – $P$ and $Q$ are assertions
    (extensions of Boolean expressions)
  – $S$ is a statement
  – For all states $\sigma$ which satisfies $P$,
    • the execution of $S$ from state $\sigma$ must terminates in a state $\sigma'$
    • $\sigma'$ satisfies $Q$
Formalizing Partial Correctness

\(\sigma \models A\)
- \(A\) is true in \(\sigma\)

\(\{P\} S \{Q\}\)
- \(\forall \sigma, \sigma' \in \Sigma. (\sigma \models P \land <S, \sigma> \rightarrow \sigma') \implies \sigma' \models Q\)
- \(\forall \sigma \in \Sigma. (\sigma \models P \land S [S]\sigma \neq \bot) \implies S [S]\sigma \models Q\)

Convention for all \(A\)
- \(\bot \models A\)

\(\forall \sigma, \sigma' \in \Sigma. \sigma \models P \implies S [S]\sigma \models Q\)
An Assertion Language

• Extend Bexp
• Allow quantifications
  – ∀i: ...
  – ∃i: ...
    • ∃i. k=i×1
• Import well known mathematical concepts
  – n! = n ×(n-1) × ⋅⋅⋅ 2 ×1
Assertion Language

\[ \text{Aexpv} \]

\[ a := n \mid X \mid i \mid a_0 + a_1 \mid a_0 - a_1 \mid a_0 \times a_1 \]

\[ \text{Assn} \]

\[ A := \text{true} \mid \text{false} \mid a_0 = a_1 \mid a_0 \leq a_1 \mid A_0 \land A_1 \mid A_0 \lor A_1 \mid \neg A \mid A_0 \Rightarrow A_1 \mid \forall i. A \mid \exists i. A \]
while \( \neg (M=N) \) do

\hspace{1cm} \text{if } M \leq N

\hspace{2cm} \text{then } N := N - M

\hspace{1cm} \text{else } M := M - N
Free and Bound Variables

• An integer variable is **bound** when it occurs in the scope of a quantifier
• Otherwise it is **free**
• Examples \( \exists i. \; k=i \times L \; (i+100 \leq 77) \land \forall i. \; j+1 = i+3 \)

\[
\begin{align*}
FV(n) &= FV(X) = \emptyset \\
FV(i) &= \{i\} \\
FV(a_0 + a_1) &= FV(a_0 - a_1) = FV(a_0 \times a_1) = FV(a_0) \cup FV(a_1) \\
FV(\text{true}) &= FV(\text{false}) = \emptyset \\
FV(a_0 = a_1) &= FV(a_0 \leq a_1) = FV(a_0) \cup FV(a_1) \\
FV(A_0 \land A_1) &= FV(A_0 \lor A_1) = FV(A_0 \Rightarrow A_1) = FV(A_0) \cup FV(A_1) \\
FV(\neg A) &= FV(A) \\
FV(\forall i. \; A) &= FV(\exists i. \; A) = FV(A) \setminus \{i\}
\end{align*}
\]
Substitution

- Visualization of an assertion $A$
  
- Consider a “pure” arithmetic expression
  $A[a/i]---a---a---$

$n[a/i] = n$ 
$i[a/i] = a$ 
$(a_0 + a_1)[a/i] = a_0[a/i] + a_1/[a/i]$ 
$(a_0 - a_1)[a/i] = a_0[a/i] - a_1[a/i]$ 
$(a_0 \times a_1)[a/i] = a_0[a/i] \times a_1[a/i]$ 

$X[a/i] = X$ 
$j[a/i] = j$
Substitution

• Visualization of an assertion A

• Consider a “pure” arithmetic expression
  \(A[a/i]---a---a---\)

true\([a/i]\) = true
false\([a/i]\) = false

\((a_0 = a_1)[a/i] = (a_0/[a/i] = a_1[a/i])\)
\((a_0 \leq a_1)[a/i] = (a_0/[a/i] \leq a_1[a/i])\)
\((A_0 \land A_1)[a/i] = (A_0[a/i] \land A_1[a/i])\)
\((A_0 \lor A_1)[a/i] = (A_0[a/i] \lor A_1[a/i])\)
\((A_0 \Rightarrow A_1)[a/i] = (A_0[a/i] \Rightarrow A_1[a/i])[a/i]\)
\((\neg A)[a/i] = \neg(A[a/i])\)

\((\forall i. A)[a/i] = \forall i. A\)
\((\exists i. A)[a/i] = \exists i. A\)
\((\forall j. A)[a/i] = (\forall j. A[a/i])\)
\((\exists j. A)[a/i] = (\exists j. A[a/i])\)
Location Substitution

• Visualization of an assertion $A$
  ---$X$---$X$---

• Consider a “pure” arithmetic expression
  $A[a/X]$ ---a---a---
Example Assertions

- \( i \) is a prime number
- \( i \) is the least common multiple of \( j \) and \( k \)
Semantics of Assertions

• An interpretation $I: \text{intvar} \rightarrow \mathbb{N}$

• The meaning of $\text{Aexpv}$
  
  – $\text{Av}[n] I_\sigma = n$
  
  – $\text{Av}[X] I_\sigma = \sigma(X)$
  
  – $\text{Av}[i] I_\sigma = I(i)$
  
  – $\text{Av}[a_0+a1] I_\sigma = \text{Av}[a_0] I_\sigma + \text{Av}[a1] I_\sigma$
  
  – ... 

• For all $a \in \text{Aexp}$ states $\sigma$ and Interpretations $I$
  
  – $\text{A}[a] \sigma = \text{Av}[a] I_\sigma$
Semantics of Assertions (II)

- $I[n/i]$ change i in I to n
- For I and $\sigma \in \Sigma_\bot$, define $\sigma \models^l A$ by structural induction
  - $\sigma \models^l \text{true}$
  - $\sigma \models^l (a_0 = a_1)$ if $A v[a_0] I \sigma = A v[a_1] I \sigma$
  - $\sigma \models^l (A \land B)$ if $\sigma \models^l A$ and $\sigma \models^l B$
  - $\sigma \models^l \neg A$ if not $\sigma \models^l A$
  - $\sigma \models^l A \Rightarrow B$ if (not $\sigma \models^l A$) or $\sigma \models^l B$
  - $\sigma \models^l \forall i.A$ if $\sigma \models^l[n/i] A$ for all $n \in \mathbb{N}$
  - $\bot \models A$
Proposition 6.4

For all $b \in \text{Bexp}$ states $\sigma$ and Interpretations $I$

$B[b] \sigma = \text{true \ iff \ } \sigma \models^I b$

$B[b] \sigma = \text{false \ iff \ } \neg \sigma \models^I b$
Partial Correctness Assertions

- \{P\}c\{Q\}
  - \(P, Q \in \text{Assn} \text{ and } c \in \text{Com}\)

- For a state \(\sigma \in \Sigma_\bot\) and interpretation \(I\)
  - \(\sigma \models^I \{P\}c\{Q\}\) if \((\sigma \models^I P \Rightarrow C [\llbracket c \rrbracket] \sigma \models^I Q)\)

- Validity
  - When \(\forall \sigma \in \Sigma_\bot, \sigma \models^I \{P\}c\{Q\}\) we write
    - \(\models^I \{P\}c\{Q\}\)
  - When \(\forall \sigma \in \Sigma_\bot, \text{ and } I \sigma \models^I \{P\}c\{Q\}\) we write
    - \(\models \{P\}c\{Q\}\)
    - \(\{P\}c\{Q\}\) is valid
The extension of an assertion

\[ A^I = \{ \sigma \in \Sigma \perp \mid \sigma \models^I A \} \]
The extension of assertions

Suppose that $\models (P \Rightarrow Q)$

Then for any interpretation $I$
\[
\forall \sigma \in \Sigma_\perp. \sigma \models^I P \Rightarrow \sigma \models^I Q
\]

$P^I \subseteq Q^I$
The extension of assertions

Suppose that $\models \{P\}c\{Q\}$

Then for any interpretation $I$
$\forall \sigma \in \Sigma_{\bot} \cdot \sigma \models^I P \Rightarrow C[c]\sigma \models^I Q$

$C[c]P^I \subseteq Q^I$
Hoare Proof Rules for Partial Correctness

\{A\} \text{skip} \ {A} \\
\{B[a/X]\} \ X:=a \ {B} \\
\{P\} S_0 \ {C} \ {C} \ S_1 \ {Q} \\
\{P\} S_0;S_1\{Q\} \\
\{P \wedge b\} \ S_0 \ {Q} \ {P \wedge \neg b} \ S_1 \ {Q} \\
\{P\} \ \text{if} \ b \ \text{then} \ S_0 \ \text{else} \ S_1 \ {Q} \\
\{I \wedge b\} \ S \ {I} \\
\{I\} \ \text{while} \ b \ \text{do} \ S \{I \wedge \neg b\} \\
\models P \Rightarrow P' \ {P'} \ S \ {Q'} \ \models Q' \Rightarrow Q \\
\{P\} \ S \ {Q}
Example

\{ X = n \land n \geq 0 \} 

Y := 1;
\{ X = n \land Y = 1 \land n \geq 0 \} 

while X > 0 do

Y := X \times Y;

X := X - 1

\{ Y = n! \}
Example

\{X = n \land n \geq 0\}

\text{Y := 1;}

\{X = n \land Y=1 \land n \geq 0\}

\text{while X > 0 do}
\{X \geq 0 \land n \geq 0 \land Y=n!/X!\}
\{X > 0 \land n \geq 0 \land Y=n!/X!\}

\text{Y := X \times Y;}
\{X > 0 \land n \geq 0 \land Y=n!/X-1!\}
\{X > 0 \land n \geq 0 \land Y=n!/X!\}
\{X = X - 1\}
\{X > 0 \land n \geq 0 \land Y=n!/X!\}
\{Y = n! \}
Example Formal

\[ \{ X = n \land n \geq 0 \} \ Y := 1 \ \{ X = n \land Y = 1 \land n \geq 0 \} \]

\[ \{ X = n \land n \geq 0 \} \ Y := 1 \ \{ X \geq 0 \land n \geq 0 \land Y = n!/X! \} \]

\[ \{ X > 0 \land n \geq 0 \land Y = n!/X! \} \ Y := X \times Y; \ \{ X > 0 \land n \geq 0 \land Y = n!/(X-1)! \} \]

\[ \{ X > 0 \land n \geq 0 \land Y = n!/(X-1)! \} \ X := X-1; \ \{ X \geq 0 \land n \geq 0 \land Y = n!/X! \} \]

\[ \{ X > 0 \land n \geq 0 \land Y = n!/X! \} \ Y := X \times Y; \ X := X-1 \ \{ X \geq 0 \land n \geq 0 \land Y = n!/X! \} \]

\[ \{ X \geq 0 \land n \geq 0 \land Y = n!/X! \land X > 0 \} \ Y := X \times Y; \ X := X-1 \ \{ X \geq 0 \land n \geq 0 \land Y = n!/X! \} \]

\[ \{ X \geq 0 \land n \geq 0 \land Y = n!/X! \} \ \text{while} \ X > 0 \ \text{do} \ Y := X \times Y; \ X := X-1 \ \{ X \geq 0 \land n \geq 0 \land Y = n!/X! \land \neg X > 0 \} \]

\[ \{ X \geq 0 \land n \geq 0 \land Y = n!/X! \} \ \text{while} \ X > 0 \ \text{do} \ Y := X \times Y; \ X := X-1 \ \{ Y = n! \} \]

\[ \{ X = n \land n \geq 0 \} \ Y := 1; \ \text{while} \ X > 0 \ \text{do} \ Y := X \times Y; \ X := X-1 \ \{ Y = n! \} \]
Soundness

• Every theorem obtained by the rule system is valid
  – \( \vdash \{P\} \implies \{Q\} \implies \vdash \{P\} \implies \{Q\} \)

• The system can be implemented (HOL, LCF, Coq)
  – Requires user assistance

• Proof of soundness
  – Every rule preserves validity (Theorem 6.1)
Soundness of skip axiom

\[ \models \{A\} \text{skip} \{A\} \]
Soundness of the assignment axiom

\[ \models \{ B[a/X] \} \ X := a \ \{ B \} \]
Soundness of the sequential composition rule

• Assume that
  \[\models \{P\} S_0 \{C\}\]
  and
  \[\models \{C\} S_1 \{Q\}\]
• Show that
  \[\models \{P\} S_0 ; S_1 \{Q\}\]
Soundness of the conditional rule

• Assume that
  \[ \models \{P \land b\} S_0 \{Q\} \]
  and
  \[ \models \{P \land \neg b\} S_1 \{Q\} \]

• Show that
  \[ \models \{P\} \text{ if } b \text{ then } S_0 \text{ else } S_1 \{Q\} \]
Soundness of the while rule

- Assume that
  \[ \vdash \{ I \land b \} S \{ I \} \]
- Show that
  \[ \models \{ I \} \text{ while } b \text{ do } S \{ I \land \neg b \} \]
Soundness of the consequence rule

• Assume that
  \[ \vdash \{P'\} S \{Q'\} \]
and
  \[ \vdash P \Rightarrow P' \]
and
  \[ \vdash Q' \Rightarrow Q \]

• Show that
  \[ \vdash \{P\} S \{Q\} \]
(Ideal) Completeness

• Every valid theorem can be proved by the rule system
• For every P and Q such that $\models \{ P \} \text{ S } \{ Q \}$ there exists a proof such $\vdash \{ P \} \text{ S } \{ Q \}$
• But what about Gödel’s incompleteness?
  $\models \{ \text{true} \} \text{ skip } \{ Q \}$
• What does $\models \{ \text{true} \} \text{ c } \{ \text{false} \}$ mean?
Relative Completeness (Chapter 7)

- Assume that every math theorem can be proved
\[ \vdash \{P\} S \{Q\} \text{ implies } \models \{P\} S \{Q\} \]
Relative completeness of composition rule

• Prove that \( \{P\} S_0;S_1\{Q\} \)

• Does there exist an assertion \( I \) such that
  \[ \vdash \{P\} S_0 \{C\} \]
  and
  \[ \vdash \{I\} S_1 \{Q\} \]
Weakest (Liberal) Precondition

• $wp(S, Q)$ – the weakest condition such that every terminating computation of $S$ results in a state satisfying $Q$

• $\llbracket wp^I(S, Q) \rrbracket = \{ \sigma \in \Sigma^\perp \mid S[\llbracket S \rrbracket] \sigma = I Q \}$

• [Can employ predicate transformer semantics to formally define the meaning (Chapter 7.5)]

• Prove that $\{P\} S_0; S_1 \{Q\}$ by proving $\models \{P\} S_0 \{I\}$ and $\models \{I\} S_1 \{Q\}$ where $I = wp(S_1, Q)$

• $\models \{P\} S \{Q\}$ iff for all $I \llbracket P \rrbracket \subseteq \llbracket wp^I(S, Q) \rrbracket$

• $\models \{P\} S \{Q\}$ iff for $P \Rightarrow wp(S, Q)$
Some WP rules

- \( \wp(\text{skip}, Q) = Q \)
- \( \wp(X := a, Q) = Q[a/X] \)
- \( \wp(S_0; S_1, Q) = \wp(S_0, \wp(S_1, Q)) \)
- \( \wp(\text{if } b \text{ then } S_0 \text{ else } S_1, Q) = b \land \wp(S_0, Q) \lor \neg b \land \wp(S_1, Q) \)
- \( \wp(S, \text{false}) = \)
• For every command $S$ and assertion $B$
  – there exists an assertion $A$, such that
    $A = \text{wp}(S, B)$ (Theorem 7.5)
  – $\vdash \{\text{wp}(S, B)\} S \{B\}$ (Lemma 7.6)
• Theorem 7.7: The proof system is relatively complete
  – $\models \{P\} S \{Q\}$ implies $\vdash \{P\} S \{Q\}$
Verification Conditions

• Generate assertions that describe the partial correctness of the program
• Use automatic theorem provers to show partial correctness
• Existing tools ESC/Java, Spec#
Verification condition for annotated commands

\[ S ::= \text{skip} | X := a | S; (X:=a) | S_0 ; \{D\} S_1 | \text{if } b \text{ then } S_0 \text{ else } S_1 | \text{while } b \{D\} \text{ do } S \]

\[ \text{vc}({\{P\} \text{ skip } \{Q\}}) = \{P\Rightarrow Q\} \]
\[ \text{vc}({\{P\} X:= a \{Q\}}) = \{P \Rightarrow Q[a/X]\} \]
\[ \text{vc}({\{P\} S ; \ X:=a \{Q\}}) = \text{vc}({\{P\} S \{Q[a/X]\}}) \]
\[ \text{vc}({\{P\} S_0; \{D\} S_1 \{Q\}}) = \text{vc}({\{P\} S_0 \{D\}}) \cup \text{vc}({\{D\} S_1 \{Q\}}) \]
\[ \text{vc}({\{P\} \text{ if } b \text{ then } S_0 \text{ else } S_1 \{Q\}}) = \text{vc}({\{P \land b\} S_0 \{Q\}}) \cup \text{vc}({\{P \land \neg b\} S_1 \{Q\}}) \]
\[ \text{vc}({\{P\} \text{ while } b \{D\} \text{ do } c \{Q\}}) = \text{vc}({\{D \land b\} c \{D\}}) \cup \{P\Rightarrow D\} \cup \{D\land \neg b \Rightarrow Q\} \]
Summary

• Axiomatic semantics provides an abstract semantics
• Can be used to explain programming
• Extensions
  – Procedures
  – Concurrency
  – Events
  – Rely/Guarantee
  – Heaps
• Can be automated
• More effort is required to make it practical