Trusted Components

Prof. Dr. Bertrand Meyer

Lecture 2: Axiomatic semantics
Reading assignment for next week

1. Ariane paper and response (see course page)

2. Axiomatic semantics chapter in *Introduction to the Theory of Programming Languages* (also accessible from course page)
Axiomatic semantics


Purpose:
- Describe the effect of programs through a theory of the underlying programming language, allowing proofs
What is a theory?

(Think of any mathematical example, e.g. elementary arithmetic)

A theory is a mathematical framework for proving properties about a certain object domain.

Such properties are called theorems.

Components of a theory:

- **Grammar** (e.g. BNF), defines well-formed formulae (WFF)
- **Axioms**: formulae asserted to be theorems
- **Inference rules**: ways to prove new theorems from previously obtained theorems
Notation

Let $f$ be a well-formed formula

Then

$\vdash f$

expresses that $f$ is a theorem
An inference rule is written

\[ f_1, f_2, \ldots, f_n \]

\[ \underline{f_0} \]

It expresses that if \( f_1, f_2, \ldots, f_n \) are theorems, we may infer \( f_0 \) as another theorem.
"Modus Ponens" (common to many theories):

\[ p, \ p \Rightarrow q \]

\[ \rightarrow \]

\[ q \]
How to obtain theorems

Theorems are obtained from the axioms by zero or more* applications of the inference rules.

*Finite of course
Proof techniques

Proof by contradiction

Deduce a contradiction from $\neg f$

Conditional proof

Prove $e \Rightarrow f$ by assuming $e$ and inferring $f$
Caution: use $e$ only within the scope of the conditional proof!

(See book chapter)
Example: a simple theory of integers

Grammar: Well-Formed Formulae are boolean expressions
- \( i_1 = i_2 \)
- \( i_1 < i_2 \)
- \( \neg b_1 \)
- \( b_1 \Rightarrow b_2 \)

where \( b_1 \) and \( b_2 \) are boolean expressions, \( i_1 \) and \( i_2 \) integer expressions

An integer expression is one of
- \( 0 \)
- A variable \( n \)
- \( f' \) where \( f \) is an integer expression (represents “successor”)


An axiom and axiom schema

\[ \vdash 0 < 0' \]

\[ \vdash f < g \Rightarrow f' < g' \]
An inference rule

\[
P(0), \quad P(f) \Rightarrow P(f')
\]

\[
P(f)
\]
The theories of interest

Grammar: a well-formed formula is a “Hoare triple”

\{P\} \ A \ \{Q\}

Informal meaning: \( A \), started in any state satisfying \( P \), will terminate in a state satisfying \( Q \)
Partial vs total correctness

\{P\} \quad A \quad \{Q\}

Total correctness:

- $A$, started in any state satisfying $P$, will terminate in a state satisfying $Q$

Partial correctness:

- $A$, started in any state satisfying $P$, will, if it *terminates*, yield a state satisfying $Q$
Axiomatic semantics

“Hoare semantics” or “Hoare logic”: a theory describing the partial correctness of programs, plus termination rules.
What is an assertion?

Predicate (boolean-valued function) on the set of computation states

True: Function that yields True for all states
False: Function that yields False for all states

P implies Q: means ∀ s: State, P(s) ⇒ Q(s)
and so on for other boolean operators
Another view of assertions

We may equivalently view an assertion $P$ as a subset of the set of states (the subset where the assertion yields True):

- **True**: Full *State* set
- **False**: Empty subset
- **implies**: subset (inclusion) relation
- **and**: intersection
- **or**: union
Assume we want to prove, on integers

\[ \{x > 0\} \land \{y \geq 0\} \] \hspace{2cm} [1]

but have actually proved

\[ \{x > 0\} \land \{y = z^2\} \] \hspace{2cm} [2]

We need properties from other theories, e.g. arithmetic
The mark [EM] will denote results from other theories, taken (in this discussion) without proof

Example:

\[ y = z^2 \quad \text{implies} \quad y \geq 0 \]  

[EM]
Rule of consequence

\{P\} A \{Q\}, \quad P' \text{ implies } P, \quad Q \text{ implies } Q'

\{P'\} A \{Q'\}
Rule of conjunction

\{P\} \land \{Q\}, \quad \{P\} \land \{R\}

\{P\} \land \{Q \text{ and } R\}
Axiomatic semantics for a programming language

Example language: Graal (from *Introduction to the theory of Programming Languages*)

Scheme: give an axiom or inference rule for every language construct
Skip
Abort

{False} abort {P}
Sequential composition

\[
\{P\} A \{Q\}, \quad \{Q\} B \{R\}
\]

\[
\{P\} \quad A ; B \quad \{R\}
\]
Assignment axiom (schema)

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

\(P [e/x]\) is the expression obtained from \(P\) by replacing (substituting) every occurrence of \(x\) by \(e\).
Substitution

\[ x [x/x] = \]
\[ x [y/x] = \]
\[ x [x/y] = \]
\[ x [z/y] = \]
\[ 3 \times x + 1 [y/x] = \]
Applying the assignment axiom