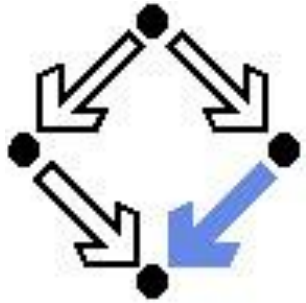
Mathematical Logic



Nikolaj Popov and Tudor Jebelean

Research Institute for Symbolic Computation

popov@risc.uni-linz.ac.at

Introduction to Program Verification

Formal Verification

- Formal verification (from Latin: verus - true) is, in general, the act of proving mathematically the correctness of a program.

- Software testing, in contrast to verification, cannot prove that a program or a system of programs does not contain any defects, neither that it has a certain property, e.g., correctness.

Formal Verification

- Only the process of formal verification can prove that a system does not have a certain defect or does have a certain property.

- For example: Is the following program correct?

$f(x) =$ **`if`** $x=0$ **`then`** 1 **`else`** $x.\ f(x\text{-}1)$

Formal Specification

- Program specification (or formal specification of a program) is the definition of what a program is expected to do.
- Normally, it does not describe, and it should not, how the program is implemented.
- The specification is usually provided by logical formulae describing a relationship between input and output parameters.

Formal Specification

- We will consider specifications which are pairs, containing a precondition (input condition) and a postcondition (output condition).
- The input condition describes what the possible inputs are, e.g., “it is a natural number” or “it is a pair such that the second argument is a real number and the first one is a digit different from 3”.

Formal Specification

- The output condition describes what the relation between the input and the output is.

- For example, for input x and output y, "$y=x!$", "$y=(x+1)(x-1)/2$", "y is the smallest natural number, s.t., $x+y > 25$".

Formal Specification

- The precondition (or input condition) of a program is a condition that must always be true just prior to the execution of that program.
- It is expressed by a predicate on the input of the program.
- If a precondition is violated, the effect of the program becomes undefined and thus may or may not carry out its intended work.

Formal Specification

- For example: the factorial is only defined for integers greater than or equal to zero. So a program that calculates the factorial of an input number would have preconditions that the number be an integer and that it be greater than or equal to zero.

Formal Specification

- A postcondition (or output condition) of a program is a condition that must always be true just after the execution of that program.
- It is expressed by a predicate on the input and the output of the program.
- Remark: We do not consider here informal specifications, which are normally written as comments between the lines of code.

Examples

- Once more the factorial example:
- Prove that the following program is correct with respect to the given specification.

  $f(x)$ = **`if`** x=0 **`then`** 1 **`else`** x. $f(x\text{-}1)$

  Precondition: $I(x) \Leftrightarrow x \in N$

  Postcondition: $O(x,y) \Leftrightarrow y=x!$

Partial and Total Correctness

- A distinction is made between total correctness, which additionally requires that the program terminates, and partial correctness, which simply requires that if an answer is returned (that is, the program terminates) it will be correct.

Partial and Total Correctness

- Partial correctness:

  For any input x obeying the input condition $I(x)$,

  if the program terminates, then the output y obeys the postcondition $O(x,y)$ for the given x.

- $(\forall x : I(x))\ (f(x)\downarrow\ \Rightarrow\ \mathrm{O}(x, f(x)))$

Partial and Total Correctness

- Termination:

  For any input x obeying the input condition $I(x)$, the program terminates.

- $(\forall x : I(x))\ (f(x)\downarrow)$

Partial and Total Correctness

- Total correctness:

  For any input x obeying the input condition $I(x)$,

  the program terminates, and the output y obeys the postcondition $O(x,y)$ for the given x.

- $(\forall x : I(x))\ (f(x)\downarrow\ \wedge\ \mathrm{O}(x, f(x)))$

Examples

- Third time the factorial example:
- Prove that the following program is correct with respect to the given specification.

  $f(x)$ = **`if`** x=0 **`then`** 1 **`else`** x. $f(x$-1)

  Precondition: $I(x) \Leftrightarrow x \in N$

  Postcondition: $O(x,y) \Leftrightarrow y=x!$
- Prove partial correctness
- Prove termination

Examples

- The sum example:
- Prove that the following program is correct with respect to the given specification.

  $f(x)$ = **`if`** $x=0$ **`then`** 0 **`else`** $x+ f(x-1)$

  Precondition: $I(x) \Leftrightarrow x \in N$

  Postcondition: $O(x,y) \Leftrightarrow y=x.(x+1)/2$
- Prove partial correctness
- Prove termination

Examples

- The $3n+1$ program:
- Prove that the following program terminates.

$f(x) =$ **`if`** $x=0 \vee x=1$ **`then`** 0

`else_if` $\text{Even}(x)$ **`then`** $f(x/2)$

`else_if` $\text{Odd}(x)$ **`then`** $f(3x+1)$

Precondition: $I(x) \Leftrightarrow x \in N$

Postcondition: $O(x,y) \Leftrightarrow y=0$