A Logical Approach to Total Correctness

Tudor Jebelean, Madalina Erascu

Research Institute for Symbolic Computation,
Johannes Kepler University, Linz, Austria

September 25, 2011

Outline

Outline

Motivation

Why should I trust those verification methods?

Long term goal:

- **Full formalization using mathematical logic,**
- **Computer aided proof of correctness.**

Outline

Syntax: Example

Program computing the GCD of two numbers using substractions

```
Program["G", G[a, b]],
 If[a = 0,
  Return[b]];
 If[b = 0,
  Return[a]];
 If[a ≥ b,
   a := G[a - b, b],
   a := G[a, b - a]]];
 Return[a],
Pre → a ≥ 0 ∧ b ≥ 0,
Post→ IsGCD[y, a, b]
```

Program: list of statements (assignment, conditional, return).

Contains also *terms* and *formulae* from the *object theory*.

Syntax

The meta-level predicate Π checks the syntactical correctness of the program:

- the program contains only valid constructs,
- variables are initialized,
- every program path contains a `Return`.

Definition

1. $\Pi[P] \Longleftrightarrow \Pi[\{\bar{x}\}, P]$
2. $\Pi[V, \langle \texttt{Return}[t] \rangle \frown P] \Longleftrightarrow \mathit{Vars}[t] \subseteq V$
3. $\Pi[V, \langle v := t \rangle \frown P] \Longleftrightarrow \bigwedge \left\{ \begin{array}{l} \mathit{Vars}[t] \subseteq V \\ \Pi[V \cup \{v\}, P] \end{array} \right.$
4. $\Pi[V, \langle \texttt{If}[\varphi, P_T, P_F] \rangle \frown P] \Longleftrightarrow \bigwedge \left\{ \begin{array}{l} \mathit{Vars}[\varphi] \subseteq V \\ \Pi[V, P_T \frown P] \\ \Pi[V, P_F \frown P] \end{array} \right.$
5. $\Pi[V, P] \Longleftrightarrow \mathbb{F}$, in all other cases

The definition of Π is a set of logical formulae!

Semantics: Example (1)

```
Program["G", G[a, b]],
 If[a = 0,
   Return[b]];
  If[b = 0,
   Return[a]];
  If[a ≥ b,
    a := G[a - b, b],
    a := G[a, b - a]]];
  Return[a],
 Pre → a ≥ 0 ∧ b ≥ 0,
 Post→ IsGCD[y, a, b]
```

Semantics

- $(a \geq 0 \wedge b \geq 0) \wedge (a = 0) \Longrightarrow G[a, b] = b$

Semantics: Example (2)

```
Program["G", G[a, b]],
  If[a = 0,
   Return[b]];
  If[b = 0,
   Return[a]];
  If[a ≥ b,
    a := G[a - b, b],
    a := G[a, b - a]]];
  Return[a],
 Pre → a ≥ 0 ∧ b ≥ 0,
 Post→ IsGCD[y, a, b]
```

Semantics

- $(a \geq 0 \wedge b \geq 0) \wedge (a \neq 0) \wedge (b = 0) \Longrightarrow G[a, b] = a$

Semantics: Example (3)

```
Program["G", G[a, b]],
  If[a = 0,
   Return[b]];
  If[b = 0,
   Return[a]];
  If[a ≥ b,
    a := G[a - b, b],
    a := G[a, b - a]]];
 Return[a],
 Pre → a ≥ 0 ∧ b ≥ 0,
 Post→ IsGCD[y, a, b]
```

Semantics

- $(a \geq 0 \wedge b \geq 0) \wedge a \neq 0 \wedge b \neq 0 \wedge$
  $a \geq b \wedge \texttt{True} \wedge a - b \geq 0 \wedge b \geq 0$
  $\Longrightarrow$
  $G[a, b] = G[a - b, b]$

Semantics: Example (4)

```
Program["G", G[a, b]],
  If[a = 0,
   Return[b]];
  If[b = 0,
   Return[a]];
  If[a ≥ b,
    a := G[a - b, b],
    a := G[a, b - a]]];
 Return[a],
 Pre → a ≥ 0 ∧ b ≥ 0,
 Post→ IsGCD[y, a, b]
```

Semantics

- $(a \geq 0 \wedge b \geq 0) \wedge a \neq 0 \wedge b \neq 0 \wedge$
  $a \not> b \wedge \texttt{True} \wedge a \geq 0 \wedge b - a \geq 0$
  $\Longrightarrow$
  $G[a, b] = G[a, b - a]$

Semantics

The meta-level function Σ creates a formula with the shape:

$$\underset{\bar{x}:I_P}{\forall} \bigwedge \left\{ p_i[\bar{x}] \Rightarrow (f[\bar{x}] = g_i[\bar{x}]) \right\}_{i=1}^{n}$$

- This is a logical formula at object level.
- This is the implicit definition of the function implemented by the program.
- This is the functional program equivalent to the imperative one.

Definition

1. $\Sigma[P] = \underset{\bar{x}}{\forall}\left(I_P[\bar{x_0}] \;\Rightarrow\; \Sigma[\{\bar{x} \to \bar{x_0}\}, P]_{\{\bar{x}_0 \leftarrow \bar{x}\}}\right)$
2. $\Sigma[\sigma, \langle \texttt{Return}[t] \rangle \smile P] = (f[x_0] = t\sigma)$
3. $\Sigma[\sigma, \langle v := t \rangle \smile P] = \Sigma[\sigma \circ \{v \to t\sigma\}, P]$
4. $\Sigma[\sigma, \langle \texttt{If}[\varphi, P_T, P_F] \rangle \smile P] = \bigwedge \left\{ \begin{array}{l} \varphi\sigma \Longrightarrow \Sigma[\sigma, P_T \smile P] \\ \neg\varphi\sigma \Longrightarrow \Sigma[\sigma, P_F \smile P] \end{array} \right.$

Partial Correctness: Example (1)

```
Program["G", G[a, b]],
 If[a = 0,
   Return[b]];
  If[b = 0,
   Return[a]];
  If[a ≥ b,
    a := G[a - b, b],
    a := G[a, b - a]]];
  Return[a],
 Pre → a ≥ 0 ∧ b ≥ 0,
 Post→ IsGCD[y, a, b]
```

Verification Conditions

- $(a \geq 0 \wedge b \geq 0) \wedge (a = 0) \Longrightarrow$ $IsGCD[b, a, b]$

Partial Correctness: Example (2)

```
Program["G", G[a, b]],
  If[a = 0,
   Return[b]];
  If[b = 0,
   Return[a]];
  If[a ≥ b,
    a := G[a - b, b],
    a := G[a, b - a]]];
  Return[a],
 Pre → a ≥ 0 ∧ b ≥ 0,
 Post→ IsGCD[y, a, b]
```

Verification Conditions

- $(a \geq 0 \wedge b \geq 0) \wedge (a \neq 0) \wedge (b = 0) \Longrightarrow$ $IsGCD[a, a, b]$

Partial Correctness: Example (3)

```
Program["G", G[a, b]],
  If[a = 0,
   Return[b]];
  If[b = 0,
   Return[a]];
  If[a ≥ b,
    a := G[a - b, b],
    a := G[a, b - a]]];
 Return[a],
 Pre → a ≥ 0 ∧ b ≥ 0,
 Post→ IsGCD[y, a, b]
```

Verification Conditions

- $(a \geq 0 \wedge b \geq 0) \wedge a \neq 0 \wedge b \neq 0 \wedge$
  $a \geq b \wedge \texttt{True} \Longrightarrow a - b \geq 0 \wedge b \geq 0$
- $(a \geq 0 \wedge b \geq 0) \wedge a \neq 0 \wedge b \neq 0 \wedge$
  $a \geq b \wedge \texttt{True} \wedge a - b \geq 0 \wedge b \geq 0 \wedge$
  $IsGCD[x, a - b, b] \Longrightarrow$
  $IsGCD[x, a, b]$

Partial Correctness: Example (4)

```
Program["G", G[a, b]],
  If[a = 0,
   Return[b]];
  If[b = 0,
   Return[a]];
  If[a ≥ b,
    a := G[a - b, b],
    a := G[a, b - a]]];
 Return[a],
 Pre → a ≥ 0 ∧ b ≥ 0,
 Post→ IsGCD[y, a, b]
```

Verification Conditions

- $(a \geq 0 \wedge b \geq 0) \wedge a \neq 0 \wedge b \neq 0 \wedge$
  $a \not> 0 \wedge \texttt{True} \Longrightarrow a \geq b \wedge b - a \geq 0$
- $(a \geq 0 \wedge b \geq 0) \wedge a \neq 0 \wedge b \neq 0 \wedge$
  $a \not> b \wedge \texttt{True} \wedge a \geq 0 \wedge b - a \geq 0 \wedge$
  $\mathit{IsGCD}[x, a, b - a] \Longrightarrow$
  $\mathit{IsGCD}[x, a, b]$

Partial Correctness

The meta-level function Γ generates two kinds of verification conditions:

- coherence (safety) conditions;
- functional conditions.

Definition

1. $\Gamma[P] = \underset{\bar{x}}{\forall}\big(\Gamma[\{\bar{x} \to \bar{x}_0\},\ I_P[\bar{x}_0],\ P]_{\{\bar{x}_0 \leftarrow \bar{x}\}}\big)$
2. $\Gamma[\sigma,\ \Phi,\ \langle \mathtt{Return}[\gamma]\rangle \smile P] = \Big(\Phi \Rightarrow O_P[\bar{x}_0, \gamma\sigma]\Big)$
3. $\Gamma[\sigma,\ \Phi,\ \langle \mathtt{Return}[t_{\tau \leftarrow u[\overline{\gamma}]}]\rangle \smile P] = \Gamma[\sigma,\ \Phi,\ \langle w := u[\overline{\gamma}],\ \mathtt{Return}[t_{\tau \leftarrow w}]\rangle \smile P]$
4. $\Gamma[\sigma,\ \Phi,\ \langle v := \gamma\rangle \smile P] = \Gamma[\sigma \circ \{v \to \gamma\sigma\},\ \Phi,\ P]$

Partial Correctness

The meta-level function Γ generates two kinds of verification conditions:

- coherence (safety) conditions;
- functional conditions.

Definition

5. $\Gamma[\sigma,\ \Phi,\ \langle v := h[\overline{\gamma}]\rangle \smile P] =$
$$\bigwedge \left\{ \begin{array}{l} \Phi \Rightarrow I_h[\overline{\gamma}\sigma] \\ \Gamma[\sigma \circ \{v \to h[\overline{\gamma}\sigma]\},\ \Phi \wedge I_h[\overline{\gamma}\sigma],\ P] \end{array} \right.$$
6. $\Gamma[\sigma,\ \Phi,\ \langle v := g[\overline{\gamma}]\rangle \smile P] =$
$$\bigwedge \left\{ \begin{array}{l} \Phi \Rightarrow I_g[\overline{\gamma}\sigma] \\ \Gamma[\sigma \circ \{v \to c\},\ \Phi \wedge I_g[\overline{\gamma}\sigma] \wedge O_g[\overline{\gamma}\sigma, c],\ P] \end{array} \right.$$ (g could be f)
7. $\Gamma[\sigma,\ \Phi,\ \langle v := t_{\tau \leftarrow u[\overline{\gamma}]}\rangle \smile P] = \Gamma[\sigma,\ \Phi,\ \langle w := u[\overline{\gamma}],\ v := t_{\tau \leftarrow w}\rangle \smile P]$
8. $\Gamma[\sigma,\ \Phi,\ \langle \mathtt{If}[\varphi_{\tau \leftarrow u[\overline{\gamma}]},\ P_T,\ P_F]\rangle \smile P] =$
$\Gamma[\sigma,\ \Phi,\ \langle w := u[\overline{\gamma}],\ \mathtt{If}[\varphi_{\tau \leftarrow w},\ P_T,\ P_F]\rangle \smile P]$
9. $\Gamma[\sigma,\ \Phi,\ \langle \mathtt{If}[\varphi,\ P_T,\ P_F]\rangle \smile P] = \bigwedge \left\{ \begin{array}{l} \Gamma[\sigma,\ \Phi \wedge \varphi\sigma,\ P_T \smile P] \\ \Gamma[\sigma,\ \Phi \wedge \neg\varphi\sigma,\ P_F \smile P] \end{array} \right.$

Termination: Example

```
Program["G", G[a, b]],
 If[a = 0,
  Return[b]];
 If[b = 0,
  Return[a]];
 If[a ≥ b,
   a := G[a - b, b],
   a := G[a, b - a]]];
 Return[a],
Pre → a ≥ 0 ∧ b ≥ 0,
Post→ IsGCD[y, a, b]
```

Termination Condition

$$\left(\forall_{\substack{a,b \\ a\geq 0, b\geq 0}} \wedge \left\{ \begin{array}{l} a = 0 \Rightarrow \pi[a, b] \\ b = 0 \Rightarrow \pi[a, b] \\ (a \neq 0 \wedge b \neq 0 \wedge a \geq b \wedge \pi[a - b, b]) \Rightarrow \pi[a, b] \\ (a \neq 0 \wedge b \neq 0 \wedge a \not\geq b \wedge \pi[a, b - a]) \Rightarrow \pi[a, b] \end{array} \right. \right) \Longrightarrow \left(\forall_{\substack{a,b \\ a\geq 0, b\geq 0}} \pi[a, b] \right)$$

The termination condition is expressed at object level!

Termination

The meta-level function Θ generates one termination condition.

Definition

1. $\Theta[P] = \left(\underset{\bar{x}:I_P}{\forall} \Theta[\{\bar{x} \to \bar{x_0}\},\ \mathbb{T},\ P]_{\{\bar{x}_0 \leftarrow \bar{x}\}}\right) \implies \underset{\bar{x}:I_P}{\forall} \pi[\bar{x}]$

2. $\Theta[\sigma,\ \Phi,\ \langle \texttt{Return}[\gamma]\rangle \smile P] = (\Phi \Rightarrow \pi[\bar{x_0}])$

3. $\Theta[\sigma,\ \Phi,\ \langle v := \gamma\rangle \smile P] = \Theta[\sigma \circ \{v \to \gamma\sigma\},\ \Phi,\ P]$

4. $\Theta[\sigma,\ \Phi,\ \langle v := h[\overline{\gamma}]\rangle \smile P] = \Theta[\sigma \circ \{v \to h[\overline{\gamma}\sigma]\},\ \Phi,\ P]$

5. $\Theta[\sigma,\ \Phi,\ \langle v := f[\overline{\gamma}]\rangle \smile P] =$
$\Theta[\sigma \circ \{v \to y\},\ \Phi \wedge O_P[\overline{\gamma}\sigma, y] \wedge \pi[\overline{\gamma}\sigma],\ P]$

6. $\Theta[\sigma,\ \Phi,\ \langle v := g[\overline{\gamma}]\rangle \smile P] = \Theta[\sigma \circ \{v \to y\},\ \Phi \wedge O_g[\overline{\gamma}\sigma, y],\ P]$

7. $\Theta[\sigma,\ \Phi,\ \langle \texttt{If}[\varphi,\ P_T,\ P_F]\rangle \smile P] =$
$\bigwedge \begin{cases} \Theta[\sigma,\ \Phi \wedge \varphi\sigma,\ P_T \smile P] \\ \Theta[\sigma,\ \Phi \wedge \neg\varphi\sigma,\ P_F \smile P] \end{cases}$

Total Correctness

Total Correctness Formula

$K_P : \underset{\bar{x}}{\forall} I_P[\bar{x}] \Rightarrow O_P[\bar{x}, f[\bar{x}]]$

Total Correctness

$$\left.\begin{array}{l}\text{semantics } \Sigma[P] \\ \text{verification conditions}\end{array}\right\} \models K_P$$

(For a concrete P, this is logical consequence at object level.)
(For the general case, these is logical consequence at first meta level.)

The proof of total correctness applies the induction principle (given by the termination condition) to the partial correctness conditions.
Take $\pi[\bar{x}]$ as $O_P[\bar{x}, f[\bar{x}]]$.

Outline

Conclusion

Existence and Uniqueness

$$\left.\begin{array}{l}\text{semantics } \Sigma[P] \\ \text{coherence conditions} \\ \text{termination condition}\end{array}\right\} \models (\exists_f! \Sigma[P])$$

Mathematics: implicit function definition – needs existence [and uniqueness].

Programming: new function implementation – needs termination.

These are equivalent!