

Verifying program correctness

A program P *satisfies* a (pre,post) contract :

- P transforms a variable v from initial values v_0 and produces a final value $v_f : v_0 \xrightarrow{P} v_f$
- v_0 satisfies pre : $\text{pre}(v_0)$ and v_f satisfies post : $\text{post}(v_0, v_f)$
- $\text{pre}(v_0) \wedge v_0 \xrightarrow{P} v_f \Rightarrow \text{post}(v_0, v_f)$
- \mathbb{D} est le domaine RTE de V

Verifying program correctness

A program P satisfies a (pre,post) contract :

- P transforms a variable v from initial values v_0 and produces a final value $v_f : v_0 \xrightarrow{P} v_f$
- v_0 satisfies pre : $\text{pre}(v_0)$ and v_f satisfies post : $\text{post}(v_0, v_f)$
- $\text{pre}(v_0) \wedge v_0 \xrightarrow{P} v_f \Rightarrow \text{post}(v_0, v_f)$
- \mathbb{D} est le domaine RTE de V

```
requires  $\text{pre}(v_0)$ 
ensures  $\text{post}(v_0, v_f)$ 
variables  $X$ 
  begin
  0 :  $P_0(v_0, v)$ 
  instruction0
  ...
  i :  $P_i(v_0, v)$ 
  ...
  instructionf-1
  f :  $P_f(v_0, v)$ 
  end
```

- $\text{pre}(v_0) \wedge v = v_0 \Rightarrow P_0(v_0, v)$
- $\text{pre}(v_0) \wedge P_f(v_0, v) \Rightarrow \text{post}(v_0, v)$
- For any pair of labels ℓ, ℓ' such that $\ell \longrightarrow \ell'$, one verifies that, pour any values $v, v' \in \text{MEMORY}$
$$\left(\left(\begin{array}{l} \text{pre}(v_0) \wedge P_\ell(v_0, v) \\ \wedge \text{cond}_{\ell, \ell'}(v) \wedge v' = f_{\ell, \ell'}(v) \end{array} \right) \Rightarrow P_{\ell'}(v_0, v') \right),$$

Contracts - Verification Conditions

```
contract P
variables v
requires  $pre(v_0)$ 
ensures  $post(v_0, v_f)$ 
begin
  0 :  $P_0(v_0, v)$ 
  S0
  ...
  i :  $P_i(v_0, v)$ 
  ...
  Sf-1
  f :  $P_f(v_0, v)$ 
end
```

Verification conditions are listed as follows :

```
contract P
variables v
requires  $pre(v_0)$ 
ensures  $post(v_0, v_f)$ 
begin
  0 :  $P_0(v_0, v)$ 
  S0
  ...
  i :  $P_i(v_0, v)$ 
  ...
  Sf-1
  f :  $P_f(v_0, v)$ 
end
```

- (initialisation)
 $pre(v_0) \wedge v = v_0 \Rightarrow P_0(v_0, v)$
- (finalisation)
 $pre(v_0) \wedge P_f(v_0, v) \Rightarrow post(v_0, v)$
- (induction)
For each labels pair ℓ, ℓ'
such that $\ell \longrightarrow \ell'$, one checks that,
for any value $v, v' \in \text{MEMORY}$
$$\left(\begin{array}{l} \left(pre(v_0) \wedge P_\ell(v_0, v) \right) \\ \wedge cond_{\ell, \ell'}(v) \wedge v' = f_{\ell, \ell'}(v) \end{array} \right) \Rightarrow P_{\ell'}(v_0, v')$$

Three kinds of verification conditions should be checked and we justify the method in the full version..


```
MACHINE M
SEES   C0
VARIABLES
  v, pc
INVARIANTS
  typing :  $v \in D$ 
  control :  $pc \in L$ 
  ...
  atl :  $pc = \ell \Rightarrow P_\ell(v_0, v)$ 
  ...
  th1 :  $pre(v_0) \wedge v = v_0 \Rightarrow P_0(v_0, v)$ 
  th2 :  $pre(v_0) \wedge P_f(v_0, v)$ 
         $\Rightarrow post(v_0, v)$ 
  ...
END
  ...
END
```

MACHINE M

SEES C_0

VARIABLES

v, pc

INVARIANTS

typing : $v \in D$

control : $pc \in L$

...

atl : $pc = \ell \Rightarrow P_\ell(v_0, v)$

...

th1 : $pre(v_0) \wedge v = v_0 \Rightarrow P_0(v_0, v)$

th2 : $pre(v_0) \wedge P_f(v_0, v)$
 $\Rightarrow post(v_0, v)$

...

END

...

END

MACHINE M

EVENTS

INITIALISATION

BEGIN

$(pc, v) : | \left(\begin{array}{l} pc' = l_0 \wedge v' = v_0 \\ \wedge pre(v_0) \end{array} \right)$

END

...

$e(\ell, \ell')$

WHEN

$pc = \ell$

$cond_{\ell, \ell'}(v)$

THEN

$pc := \ell'$

$v := f_{\ell, \ell'}(v)$

END

...

END

(Induction Principle (I))

A property $S(z_0, z)$ is a safety for an annotated program P if, and only if, there exists a property $I(z_0, z)$ satisfying :

- 1 $\forall z_0, z \in L \times D. \text{init}(z_0) \wedge z = z_0 \Rightarrow I(z_0, z)$
- 2 $\forall z_0, z, z' \in L \times D. \text{init}(z_0) \wedge I(z_0, z) \wedge (z \xrightarrow{P} z') \Rightarrow I(z_0, z')$
- 3 $\forall z_0, z \in L \times D. \text{init}(z_0) \wedge I(z_0, z) \Rightarrow S(z_0, z)$

(Induction Principle (II))

A property $S(\ell_0, x_0, \ell, x)$ is a safety property for an annotated program P if, and only if, there exists a property $I(\ell_0, x_0, \ell, x)$ satisfying :

- 1 $\forall \ell_0, \ell \in L, x_0 \in D. \ell_0 \in L_0 \wedge \text{pre}(x_0) \wedge x = x_0 \wedge \text{pc} = \ell_0 \Rightarrow J(\ell_0, x_0, \ell, x)$
- 2 $\forall \ell, \ell' \in L, x, x_0 \in D. \ell_0 \in L_0 \wedge \text{pre}(x_0) \wedge J(\ell_0, x_0, \ell, x) \wedge BA(e(\ell, \ell'), \ell, x, \ell', x') \Rightarrow J(\ell_0, x_0, \ell', x')$
- 3 $\forall \ell_0, \ell \in L, x_0, x \in D. \text{pre}(x_0) \wedge \ell_0 \in L_0 \wedge J(\ell_0, x_0, \ell, x) \Rightarrow S(\ell_0, x_0, \ell, x)$

(Induction Principle (II))

A property $S(\ell_0, x_0, \ell, x)$ is a safety property for an annotated program P if, and only if, there exists a property $I(\ell_0, x_0, \ell, x)$ satisfying :

- 1 $\forall \ell_0, \ell \in L, x_0, x \in D. \ell_0 \in L_0 \wedge pre(x_0) \wedge x = x_0 \wedge pc = \ell_0 \Rightarrow J(\ell_0, x_0, \ell, x)$
- 2 $\forall \ell, \ell' \in L, x, x_0 \in D. \ell_0 \in L_0 \wedge pre(x_0) \wedge J(\ell_0, x_0, \ell, x) \wedge BA(e(\ell, \ell'), \cdot)(\ell, x, \ell', x') \Rightarrow J(\ell_0, x_0, \ell', x')$
- 3 $\forall \ell_0, \ell \in L, x_0, x \in D. pre(x_0) \wedge \ell_0 \in L_0 \wedge J(\ell_0, x_0, \ell, x) \Rightarrow S(\ell_0, x_0, \ell, x)$

(Induction Principle (III))

A property $S(x_0, \ell, x)$ is a safety for an annotated program P with one entry point if, and only if, there exists a property $I(x_0, \ell, x)$ satisfying :

- 1 $\forall x_0 \in D. pre(x_0) \wedge x = x_0 \wedge \ell = \ell_0 \Rightarrow J(x_0, \ell, x)$
- 2 $\forall \ell, \ell' \in L, x, x_0 \in D. pre(x_0) \wedge J(x_0, \ell, x) \wedge BA(e(\ell, \ell'), \cdot)(\ell, x, \ell', x') \Rightarrow J(x_0, \ell', x')$
- 3 $\forall \ell \in L, x_0, x \in D. pre(x_0) \wedge J(x_0, \ell, x) \Rightarrow S(x_0, \ell, x)$

A short example

S

contract SIMPLE

variables x

requires $x_0 \in \mathbb{N}$

ensures $x_f = 0$

begin

$\ell_0 : \{0 \leq x \leq x_0 \wedge x_0 \in \mathbb{N}\}$

while $0 < x$ **do**

$\ell_1 : \{0 < x \wedge x \leq x_0 \wedge x_0 \in \mathbb{N}\}$

$x := x - 1;$

od

$\ell_2 : \{x = 0\}$ end

Event *Init*

THEN

$act1 : x := x_0$

$act2 : l := l_0$

Event *el0l1*

WHEN

$grd1 : l = l_0$

$grd2 : 0 < x$

THEN

$act1 : l := l_1$

INVARIANTS

inv1 : $x \in \mathbb{N}$

inv2 : $l \in L$

inv3 : $l = l_0 \Rightarrow$

$0 \leq x \wedge x \leq x_0 \wedge x_0 \in \mathbb{N}$

inv4 : $l = l_1 \Rightarrow$

$0 < x \wedge x \leq x_0 \wedge x_0 \in \mathbb{N}$

inv5 : $l = l_2 \Rightarrow x = 0$

requires : $x_0 \in \mathbb{N} \wedge x = x_0$

$\Rightarrow x = x_0 \wedge x_0 \in \mathbb{N}$

ensures : $x = 0 \wedge x = x_0$

$\Rightarrow x = 0$

Event *el0l2*

WHEN

$grd1 : l = l_0$

$grd2 : \neg(0 < x)$

THEN

$act1 : l := l_2$

Event *el1l0*

WHEN

$grd1 : l = l_1$

THEN

$act1 : l := l_0$

$act2 : x := x - 1$

Partial correctness using Event-B models

For each pair of successive labels ℓ, ℓ' , the three statements are equivalent :

- $P_\ell(v) \wedge \text{cond}_{\ell, \ell'}(v) \wedge v' = f_{\ell, \ell'}(v) \Rightarrow P_{\ell'}(v')$
- $I(c, v) \wedge c = \ell \wedge \text{cond}_{\ell, \ell'}(v) \wedge c' = \ell' \wedge v' = f_{\ell, \ell'}(v) \Rightarrow (c' = \ell' \Rightarrow P_{\ell'}(v'))$
- $I(c, v) \wedge BA(e(\ell, \ell'))(c, v, c', v') \Rightarrow (c' = \ell' \Rightarrow P_{\ell'}(v'))$

L

Let AA an annotated algorithm with precondition $\mathbf{pre}(AA)(v)$ and postcondition $\mathbf{post}(AA)(v_0, v)$. Let the context C and the machine M generated from AA using the construction given previously. We assume that ℓ_0 is the first label and ℓ_e is the last label. We add the following safety properties in the machine M :

- $c = \ell_0 \wedge \mathbf{pre}(AA)(v) \Rightarrow P_{\ell_0}(v)$
- $c = \ell_e \Rightarrow (P_{\ell_e}(v) \Rightarrow \mathbf{post}(AA)(v_0, v))$

If proof obligations are discharged, then the annotated algorithm AA is partially correct with respect to its pre/post specification.

