Verifier WLP Definitions

Jenna Wise, Johannes Bader, Jonathan Aldrich, Éric Tanter

November 12, 2018

1 Weakest liberal precondition calculus definitions over self-framed non-gradual formulas

$$\text{WLP}(\mathit{skip}, \widehat{\phi}) = \widehat{\phi}$$

$$\text{WLP}(s_1; s_2, \widehat{\phi}) = \text{WLP}(s_1, \text{WLP}(s_2, \widehat{\phi}))$$

$$\text{WLP}(T\ x := e, \widehat{\phi}) = \max_{\Rightarrow} \left\{ \widehat{\phi}' \mid \widehat{\phi}' \Rightarrow \widehat{\phi}[e/x] \quad \wedge \quad \widehat{\phi}' \Rightarrow \text{acc}(e) \right\}$$

$$\text{WLP}(\mathit{if}\ (x \odot y)\ \{s_1\}\ \mathit{else}\ \{s_2\}, \widehat{\phi}) =$$

$$\text{WLP}(x.f := y, \widehat{\phi}) = \text{acc}(x.f) * \max_{\Rightarrow} \left\{ \widehat{\phi}' \mid \widehat{\phi}' * \text{acc}(x.f) * (x.f = y) \Rightarrow \widehat{\phi} \ \wedge\ \widehat{\phi}' * \text{acc}(x.f) \in \textsc{SatFormula} \right\}$$

$$\text{WLP}(x := \mathit{new}\ C, \widehat{\phi}) = \max_{\Rightarrow} \left\{ \widehat{\phi}' \mid \widehat{\phi}' * (x \neq \mathit{null}) * \overline{\text{acc}(x.f_i)} \Rightarrow \widehat{\phi} \right\} \quad \text{where fields}(C) = \overline{T_i\ f_i}$$

$$\text{WLP}(y := z.m(\overline{x}), \widehat{\phi}) = \mathit{undefined}$$

$$\text{WLP}(y := z.m_C(\overline{x}), \widehat{\phi}) = \max_{\Rightarrow} \Big\{ \widehat{\phi}' \mid y \notin \text{FV}(\widehat{\phi}') \quad \wedge \quad \widehat{\phi}' \Rightarrow (z \neq \mathit{null}) * \text{pre}(C, m) \left[z/\mathit{this}, \overline{x_i/\text{params}(C, m)_i} \right] \quad \wedge \quad \widehat{\phi}' * \text{post}(C, m) \left[z/\mathit{this}, \overline{x_i/\text{old}(\text{params}(C, m)_i)}, y/\mathit{result} \right] \Rightarrow \widehat{\phi} \Big\}$$

$$\text{WLP}(\mathit{assert}\ \phi_a, \widehat{\phi}) = \max_{\Rightarrow} \left\{ \widehat{\phi}' \mid \widehat{\phi}' \Rightarrow \widehat{\phi} \quad \wedge \quad \widehat{\phi}' \Rightarrow \phi_a \right\}$$

$$\text{WLP}(\mathit{release}\ \phi_a, \widehat{\phi}) =$$

$$\text{WLP}(\mathit{hold}\ \phi_a\ \{s\}, \widehat{\phi}) =$$

Note:

Dynamic method calls. Dynamic method calls are left undefined, because we are not verifying programs with dynamic dispatch at this time (all method calls should be static method calls). They are included in the grammar for future implementation.

If & Release & hold. Definitions coming soon.

Predicates in the logic. Although the grammar allows for abstract predicate families, we do not support them yet. Therefore, we assume formulas look like:

$$\phi ::= \text{true} \mid e \odot e \mid acc(e.f) \mid \phi * \phi$$

2 Helpful function definitions

TBD

3 Algorithmic WLP calculus definitions over self-framed non-gradual formulas

$\text{WLP}(skip, \widehat{\phi}) = \widehat{\phi}$

$\text{WLP}(s_1; s_2, \widehat{\phi}) = \text{WLP}(s_1, \text{WLP}(s_2, \widehat{\phi}))$

$$\text{WLP}(T\ x := e, \widehat{\phi}) = \begin{cases} \widehat{\phi}[e/x] & if\ \widehat{\phi}[e/x] \Rightarrow acc(e) \\ acc(e) * \widehat{\phi}[e/x] & otherwise \end{cases}$$

Check that $\text{WLP}(T\ x := e, \widehat{\phi}) * x = e \Rightarrow \widehat{\phi}$ and that $\text{WLP}(T\ x := e, \widehat{\phi})$ is satisfiable.

$\text{WLP}(if\ (x \odot y)\ \{s_1\}\ else\ \{s_2\}, \widehat{\phi}) =$

$$\text{WLP}(x.f := y, \widehat{\phi}) = \begin{cases} \widehat{\phi}[y/x.f] & if\ \widehat{\phi}[y/x.f] \Rightarrow acc(x.f) \\ acc(x.f) * \widehat{\phi}[y/x.f] & otherwise \end{cases}$$

Check that $\text{WLP}(x.f := y, \widehat{\phi}) * x.f = y \Rightarrow \widehat{\phi}$ and that $\text{WLP}(x.f := y, \widehat{\phi})$ is satisfiable.

Important cases to consider:

$\widehat{\phi} = acc(x.f) * x.f = p * x.f = q * a = b$

$\widehat{\phi} = acc(x.f) * acc(x.f.f) * x = y$

$$\text{WLP}(x := new\ C, \widehat{\phi}) = \begin{cases} \widehat{\phi} \div x & if\ (\widehat{\phi} \div x) * x \neq null * \overline{x \neq e_i} * \overline{acc(x.f_i)} \Rightarrow \widehat{\phi} \\ undefined & otherwise \end{cases}$$

where $\text{fields}(C) = \overline{T_i\ f_i}$, $\widehat{\phi} \div x$ means to transitively expand (in-)equalities ($\odot$) and then removing conjunctive terms containing x, and $\overline{x \neq e_i}$ is a conjunction in $\widehat{\phi}$.

Check $\text{WLP}(x := new\ C, \widehat{\phi})$ is satisfiable.

Important cases to consider:

$\widehat{\phi} = x \neq null * acc(x.f)$
$\widehat{\phi} = x \neq null * acc(x.f) * x.f = 1 * x.f = y$ — should fail, bad postcondition
$\widehat{\phi} = x \neq null * acc(x.f) * x = y * x = z$ — should fail, bad postcondition
$\widehat{\phi} = x \neq null * acc(x.f) * x = y * y = z$ — should fail, bad postcondition
$\widehat{\phi} = x \neq null * acc(x.f) * x \neq y * y = z$
$\widehat{\phi} = x \neq null * acc(x.f) * acc(x.f.f) * x.f.f \neq y$ — should fail, bad postcondition
$\widehat{\phi} = x \neq null * acc(y.f) * x = y$ — should fail, bad postcondition
$\widehat{\phi} = x \neq null * acc(x.f) * y > x.f * x.f > z * r \geq x.f * x.f \geq s$ — should fail, bad postcondition

Note:

$x := new\ C$ creates a fresh object and assigns it to x without setting default values to the object's fields; therefore, postconditions cannot say anything about the value of x other than it does not equal other values (no aliasing with x) and they cannot say anything about the values of the fields of x.

WLP$(y := z.m(\overline{x}), \widehat{\phi}) = undefined$

WLP$(y := z.m_C(\overline{x}), \widehat{\phi}) =$

WLP$(assert\ \phi_a, \widehat{\phi}) =$

WLP$(release\ \phi_a, \widehat{\phi}) =$

WLP$(hold\ \phi_a\ \{s\}, \widehat{\phi}) =$