Formal Methods for Java

Lecture 25: Proving a JML-Program with KeY

Jochen Hoenicke

Software Engineering
Albert-Ludwigs-University Freiburg

January 29, 2013

The KeY-Project

- Theorem Prover
- Developed at University of Karlsruhe
- http://www.key-project.org/.

- Theory specialized for Java(Card).
- Can generate proof-obligations from JML specification.
- Underlying theory: Sequent Calculus + Dynamic Logic

Case Study: Euklid's Algorithm

Java code to compute gcd of non-negative numbers:

```
public static int gcd(int a, int b) {
    while (a != 0 && b != 0) {
        if (a > b)
            a = a - b;
        else
            b = b - a;
    }
    return (a > b) ? a : b;
}
```

Lets prove it with KeY-System.

Specification

We first need a specification.

Definition (GCD)

Let a and b be natural numbers. A number d is the greatest common divisor (GCD) of a and b iff

1. $d|a$ and $d|b$
2. If $c|a$ and $c|b$, then $c|d$.

$d|a$ means d divides a.
$d|a :\Leftrightarrow \exists q. d * q = a$

The specifation can be converted to JML:

```
/*@
  @ requires a >= 0 && b >= 0;
  @ ensures \result >= 0;
  @ ensures (\exists int q; \result*q == a) &&
  @        (\exists int q; \result*q == b) &&
  @  (\forall int c;
  @       (\exists int q; c*q == a) && (\exists int q; c*q == b);
  @       (\exists int q; c*q == \result));
  @*/
public static int gcd(int a, int b)
```

The rule while_invariant_with_variant_dec

The rule while_invariant_with_variant_dec takes an invariant *inv*, a modifies set $\{m_1, \ldots, m_k\}$ and a variant v. The following cases must be proven.

- Initially Valid: $\Longrightarrow inv \land v \geq 0$
- Body Preserves Invariant:

$$\Longrightarrow \{m_1 := x_1 \| \ldots \| m_k := x_k\}(inv \land [\{b = COND;\}]b = \textbf{true} \\ \to \langle BODY \rangle inv$$

- Use Case:

$$\Longrightarrow \{m_1 := x_1 \| \ldots \| m_k := x_k\}(inv \land [\{b = COND;\}]b = \textbf{false} \\ \to \langle \ldots \rangle \phi$$

- Termination:

$$\Longrightarrow \{m_1 := x_1 \| \ldots \| m_k := x_k\}(inv \land v \geq 0 \land [\{b = COND;\}]b = \textbf{true} \\ \to \{old := v\}\langle BODY \rangle v \leq old \land v \geq 0$$

Loop-Invariant

What is the loop invariant?

The algorithm changes *a* and *b*, but the gcd of *a* and *b* should stay the same.

In fact the set of common divisors of *a* and *b* never changes.
This suggests the following invariant:

$$\forall d.(d | \texttt{\textbackslash old}(a) \land d | \texttt{\textbackslash old}(b) \leftrightarrow d|a \land d|b)$$

In JML this can be specified as:

```
/*@ loop_invariant a >= 0 && b >= 0 &&
  @     (\forall int d; true;
  @     (\exists int q; \old(a) == q*d)
  @      && (\exists int q; \old(b) == q*d)
  @ <==>(\exists int q;  a == q*d) && (\exists int q; b == q*d)
  @     );
  @ assignable a, b;
  @ decreases a+b;
  @*/
```