Bugs related to overly strong conditions found in the JRRT's implementations

Melina Mongiovi
Advisor: Rohit Gheyi

Federal University of Campina Grande, Brazil
melina@copin.ufcg.edu.br

1 Bugs

Listing 1.1. Original version

```
public class A {
  public B f = null;
  protected long m(int b) {
    return 0;
  }
}
public class B extends A {
  protected long m(int b) {
    return 1;
  }
  public long k() {
    return new B().m(2);
  }
}
```

Listing 1.2. Behavioral Preserving target's version after removing a subset of overly strong conditions.

```
public class A {
  public B f = null;
}
public class B extends A {
  public long m(int b) {
    return 1;
  }
 public long m(A a, int b) {
    return 0;
  }
  public long k() {
    return new B().m(2);
  }
}
```

Fig. 1. Moving method A.m(int) to class B using JRRT (03/feb/2013). The engine reports the following warning message: "Overriding has changed".

Listing 1.3. Original version

```
public class A {
  public C f = null;
  protected long m(long b) {
    return 1;
  }
  protected long m(int b) {
    return 0;
  }
}
public class B extends A {
  public long k() {
    return m(2);
  }
}
public class C {}
```

Listing 1.4. Behavioral Preserving target's version after removing a subset of overly strong conditions.

```
abstract public class A {
  public C f = null;
  protected long m(long b) {
    return 1;
  }
  abstract protected long m(int b);
}
abstract public class B extends A {
  public long k() {
    return m(2);
  }
}
public class C {
  protected long m(int b) {
    return 0;
  }
}
```

Fig. 2. Moving method A.m(int) to class C using JRRT (03/feb/2013). The engine reports the following warning message: "Method not accessible".

Listing 1.5. Original version

```
public class A {
  protected long m(int b) {
    return 1;
  }
  public long k() {
    return new A().m(2);
  }
}
public class B extends A {
  public A f = null;
  protected long m(int b) {
    return 0;
  }
}
```

Listing 1.6. Behavioral Preserving target's version after removing a subset of overly strong conditions.

```
public class A {
  protected long m(int b) {
    return 1;
  }
  public long k() {
    return new A().m(2);
  }
 public long m(A a, int b) {
    return 0;
  }
}
public class B extends A {
  public A f = null;
}
```

Fig. 3. Moving method B.m(int) to class A using JRRT (03/feb/2013). The engine reports the following warning message: "Method is used".

Listing 1.7. Original version

```
public class A {
  protected long m(int b) {
    return 1;
  }
}
public class B extends A {
  public C f = null;
  public long k() {
    return new A().m(2);
  }
  public long m(int b) {
    return 0;
  }
}
public class C extends A {}
```

Listing 1.8. Behavioral Preserving target's version after removing a subset of overly strong conditions.

```
public class A {
  protected long m(int b) {
    return 1;
  }
}
public class B extends A {
  public C f = null;
  public long k() {
    return new A().m(2);
  }
}
public class C extends A {
  public long m(int b) {
    return 0;
  }
}
```

Fig. 4. Moving method B.m(int) to class C using JRRT (03/feb/2013). The engine reports the following warning message: "Cannot inline ambiguous method call".

Listing 1.9. Original version

```
public class A {
  private int f = 10;
}
public class B extends A {
  protected int f = 11;
  public long m() {
    return f;
  }
}
public class C extends A {}
```

Listing 1.10. Behavioral Preserving target's version after removing a subset of overly strong conditions.

```
public class A {}
public class B extends A {
  protected int f = 11;
  public long m() {
    return f;
  }
}
public class C extends A {
  private int f = 10;
}
```

Fig. 5. Pushing down field A.f using JRRT (03/feb/2013). The engine reports the following warning message: "Can only push down to exactly one subclass".

Listing 1.11. Original version

```
public class A {
  public long k(int a) {
    return 1;
  }
  public long m() {
    return new A().k(2)
  }
}
public class B extends A {
  long n(int a) {
    return 0;
  }
}

public class C extends A {}
```

Listing 1.12. Behavioral Preserving target's version after removing a subset of overly strong conditions.

```
public class A {
  public long k(int a) {
    return 1;
  }
  public long m() {
    return new A().k(2)
  }
}
public class B extends A {
  long k(int a) {
    return 0;
  }
}
```

Fig. 6. Renaming method B.n(int) using JRRT (03/feb/2013). The engine reports the following warning message: "Cannot introduce new method".

Listing 1.13. Original version

```
package p0;
public class A {
  public long k(int a) {
    return 1;
  }
  public long m() {
    return new A().k(2);
  }
  public long test() {
    return m();
  }
}
package p1;
import p0.*;
public class B extends A {
  public long k() {
    return 0;
  }
}
package p0;
public class B extends A {}
```

Listing 1.14. Behavioral Preserving target's version after removing a subset of overly strong conditions.

```
package p0;
public class A {
  public long k(int a) {
    return 1;
  }
  public long m() {
    return new A().k(2);
  }
  public long test() {
    return m();
  }
}
package p1;
import p0.*;
public class B extends A {
  public long k() {
    return 0;
  }
  public long m() {
    return new A().k(2);
 }
}
```

Fig. 7. Pushing down method A.m() using JRRT (03/feb/2013). The engine reports the following warning message: "Or of refactorings: no refactoring succeeded".