Testing Programs with Symmetry

and why not Java Card applets and APIs?

Arnaud Gotlieb
IRISA / INRIA
Rennes, FRANCE
Outline

- Motivations
- A Generalized definition of symmetry relation
- Symmetric Testing
- First experimental results
- Related and further works
A diagrammatic view of Program Testing

A sequential program

computing

Input test data

Oracle

checking

Outputs

verdict (pass, fail, ?)
Non-testable programs [Weyuker 82]

❔ No (complete and correct) oracle available

Because
- No formal model available
- Only informal and partial specifications
- Expected results too difficult to compute by hand
- ...

Typical examples:
- APIs, third-party libraries (no source code)
- COTS (no source code)
- complex mathematical functions
Testing with symmetry: a very first example

P: a program that implements the gcd of 2 integers

Problem: P(1309, 693) = ?

Symmetry relation: ∀u, v, gcd(u, v) = gcd(v, u)

Hence, if P(1309, 693) ≠ P(693, 1309) then verdict = fail
Outline

Motivations

A Generalized definition of symmetry relation

Symmetric Testing

First experimental results

Related and further works
Background on Group Theory

- Group \((E, o)\) iff \(\exists\) neutral, \(\forall x \exists\) inverse\((x)\), \(o\) associative

- Symmetric Group \(S_n\): set of permutations over \(\{1,\ldots,n\}\)

  if \(x = (x_1,\ldots,x_n)\) \(\theta\).\(x\) denotes \((x_{\theta(1)},\ldots,x_{\theta(n)})\)

  \(S_n\) can be generated by \(\tau = (12)\) and \(\sigma = (12\ldots k)\)

- Group homomorphism from \(S_k\) to \(S_l\)

  \(\varphi : S_k \rightarrow S_l\) such as \(\varphi(\theta \circ \theta') = \varphi(\theta) \circ \varphi(\theta')\)
Symmetry relation

Program $p : D_1 \times \ldots \times D_k \to D_1' \times \ldots \times D_l'$

$\psi_{k,l}$ is a symmetry relation for $p$ over $D_1 \times \ldots \times D_k$ iff:

1) $\forall \theta \in S_k, \exists \eta \in S_l$, such as $\forall x \ p(\theta.x) = \eta.p(x)$

2) $\psi_{k,l} : S_k \to S_l$ is a group homomorphism

Ex: $gcd$ satisfies a $\psi_{2,1}$ symmetry relation over $\mathbb{N} \times \mathbb{N}$
### Symmetry relation: examples

<table>
<thead>
<tr>
<th>Methods from <code>java.util.Collections</code> (12 symmetric methods over 19 distincts methods)</th>
<th>Perm. inputs</th>
<th>Per. outp</th>
<th>Symm relation</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean <code>replaceAll</code>(<code>List A, Object oldVal, Object newVal</code>)</td>
<td>A</td>
<td>A</td>
<td>ψ_{</td>
</tr>
<tr>
<td>Object <code>max</code>(<code>Collection A</code>)</td>
<td>A</td>
<td>Ret</td>
<td>ψ_{</td>
</tr>
<tr>
<td>void <code>copy</code>(<code>List B, List A</code>)</td>
<td>A</td>
<td>B</td>
<td>ψ_{</td>
</tr>
<tr>
<td>void <code>sort</code>(<code>List A</code>)</td>
<td>A</td>
<td>A</td>
<td>ψ_{</td>
</tr>
<tr>
<td>List <code>nCopies</code>(<code>int n, Object O</code>)</td>
<td>O</td>
<td>Ret</td>
<td>ψ_{1,n}</td>
</tr>
</tbody>
</table>
Finding symmetry violations

- The symmetry relation has to be given by the tester:
  in extension  \( \{ (\theta, \eta) \} \forall \theta \in S_k \)

- If \( p(\theta . x) \neq \eta . p(x) \) for any \( x \in D_1 \times \ldots \times D_k \)
  then \text{verdict} = \text{fail}

- Any test data generator can be employed
  (random, pair-wise, boundary-value, ...)

But, how to find all the symmetry violations?
Outline

- Motivations
- A Generalized definition of Symmetry relation
  - Symmetric Testing
- First experimental results
- Related and further works
Local exhaustive testing \cite{WoodMillerNoonan92} tuned for testing symmetry relations

- Tries exhaustively all the input values
  into a restricted finite domain $D$ of $D_1 \times \ldots \times D_k$

→ in Symmetric Test., a Cartesian Product iterator
  Ex: $\{a,b\} \times \{c,d,e\}$ gives $(a,c),(a,d),(a,e),(b,c),(b,d),(b,e)$

- Proves that $p(\theta.x) = \eta.p(x)$ holds $\forall x \in D$
  when both the executions of $p(\theta.x)$ and $p(x)$ terminate
Comparison checks

\[ \forall x \in D, \ \forall \theta \in S_k, \text{ST checks:} \]

\[ \theta \]

\[ x \]

\[ \theta \cdot x \]

\[ p \]

\[ p(x) \]

\[ p \circ \theta (x) \neq \psi_{k, l}(\theta) \circ p (x) \]

\[ \eta = \psi_{k, l}(\theta) \]

\[ \text{but there are } k! \text{ permutations in } S_k \]

\[ \text{needs to know } \psi_{k, l}(\theta) \text{ for all } \theta \in S_k \]
Checking only two permutations:

**Symmetric Testing** requires only to check
\[ \tau = (12) \quad \text{and} \quad \sigma = (12..k) \]

**Proposition:**
\[ \forall \theta \in S_k, \quad p \circ \theta = \psi_{k,l} (\theta) \circ p \iff \begin{cases} \quad p \circ \tau = \psi_{k,l} (\tau) \circ p \\ \quad p \circ \sigma = \psi_{k,l} (\sigma) \circ p \end{cases} \]

**Sketch of proof:**

\[ \iff \] \( \Rightarrow \) trivial

\[ \iff \] \( \Leftarrow \) \[ p \circ \theta = p \circ (\tau \circ \sigma...) = \psi_{k,l}(\tau) \circ p \circ (\sigma...) \]
\[ = (\psi_{k,l}(\tau) \circ \psi_{k,l}(\sigma)...) \circ p \]
\[ = \psi_{k,l}(\theta) \circ p \] (because \( \psi_{k,l} \) is an homomorphism)
A semi-correct procedure for ST

**In:** program \( p \), finite domain \( D \), \( \psi_{k,l}(\tau) \), \( \psi_{k,l}(\sigma) \)

**Out:** a symmetry violation or a proof that \( \psi_{k,l} \) holds over \( D \)

```
while ( D \( \neq \) \( \emptyset \) )
    pick up \( x \) in \( D \) and \( D := D \ \setminus \ \{x\} \)
    let \( r := p(x) \), \( r_{\tau} := p(\tau.x) \), \( r_{\sigma} := p(\sigma.x) \)

    if ( \( r_{\tau} \neq \psi_{k,l}(\tau) \cdot r \) ) then return violation (\( x, r, r_{\tau} \))

    if ( \( r_{\sigma} \neq \psi_{k,l}(\sigma) \cdot r \) ) then return violation (\( x, r, r_{\sigma} \))

return («Q.E.D.»)
```
Limitations of Symmetric Testing

Terminaison not guaranteed, but
# comparison checks is O(d) in place of O(k! . d)
where d = # test data

Impossible to know which inputs among
x, \(\tau \cdot x\), \(\sigma \cdot x\) is responsible of the symmetry violation

Incorrect versions of p may be symmetric too!

But,
No oracle is required, ST is fully automatic
Outline

- Motivations
- A Generalized definition of symmetry relation
- Symmetric Testing
- First experimental results
- Related and further works
Experiments on testing Java methods

Symmetric Testing

- implemented with the Java unit testing tool: *Roast* [Daley, Hoffman, Strooper 2002]

- performed on programs where faults were injected by mutation (37 mutants manually created)

  - Integer `GetMid(Integer i, Integer j, Integer k)`
  - int `Trityp(Integer x, Integer y, Integer z)`
  - Vector `min_nb(Vector V, int n)`

- performed on methods from `java.lang.Collections`

  - void `sort(List A)`
  - void `copy(List A, List B)`
  - bool `replaceAll(List A, Object oldVal, newVal)`
## Experimental results

<table>
<thead>
<tr>
<th>programs</th>
<th>symmetry</th>
<th>Mutation score</th>
<th>Size of D #test data</th>
<th>time used to prove symmetry</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetMid</td>
<td>$\Psi_{3,1}$</td>
<td>$2/2$</td>
<td>$\approx10^6$</td>
<td>9.4 sec</td>
</tr>
<tr>
<td>trityp</td>
<td>$\Psi_{3,1}$</td>
<td>$23/33$</td>
<td>$\approx10^6$</td>
<td>9.6 sec</td>
</tr>
<tr>
<td>sort ° min_nb</td>
<td>$\Psi_{</td>
<td>A</td>
<td>,\text{Ret}}$</td>
<td>$2/2$</td>
</tr>
</tbody>
</table>

### Programs extracted from `java.lang.Collections`

<table>
<thead>
<tr>
<th>programs</th>
<th>symmetry</th>
<th>-</th>
<th>Size of D</th>
<th>time used</th>
</tr>
</thead>
<tbody>
<tr>
<td>sort</td>
<td>$\Psi_{</td>
<td>A</td>
<td>,</td>
<td>A</td>
</tr>
<tr>
<td>copy</td>
<td>$\Psi_{</td>
<td>A</td>
<td>,</td>
<td>B</td>
</tr>
<tr>
<td>replaceAll</td>
<td>$\Psi_{</td>
<td>A</td>
<td>,</td>
<td>A</td>
</tr>
</tbody>
</table>

(CPU time on 1.8GHz Pentium 4 with Sun Standard 1.4.1 JVM)
Outline

- Motivations
- A Generalized definition of symmetry relation
- Symmetric Testing
- First experimental results
- Related and further works
Related work

**Data Diversity**

[Amann, Knight TComp’88]

**Symmetry and Model Checking**

[Emerson, Sistla CAV’93]

[Ip, Dill CHDL’93]

Symmetry is used to prune the exploration of the states space

**Metamorphic Testing**

[Chen, Tse, Zhou COMPSAC’01]

\[ r(x_1, \ldots, x_n) \Rightarrow r_f(p(x_1), \ldots, p(x_n)) \]
Further works

- Reaching the minimum number of comparison checks by finding ad-hoc order of n_tuples generation.

- Expressing symmetry relations in OCL (or in JML) as postconditions requires to define Symmetric Group classes.

- Testing Java Card applets and APIs, where non-trivial symmetric relations may exist:

  Ex: `abstract short javacard.security.Checksum.doFinal(byte inBuff[], ...)`
  which is based on CRC algorithms.
An example with inheritance

abstract class Animal
{
    abstract int m();
}
class B extends A
{
    int m() {return 0;};
}
class C extends A
{
    int m() {return 1;};
}
class Use
{
    int p(A a) { return a.m(); };  
}

\[ p : \{b,c\}^2 \rightarrow \{0,1\}^2 \]
where \( b \) is identified to \((b,c)\)
and \( c \) is identified to \((c,b)\)

\( p \) has to satisfy a
\( \psi_{2,2} \) symmetry relation
because \( p(\tau.(b,c)) = \tau.p((b,c)) \)
and \( p(\tau.(c,b)) = \tau.p((c,b)) \)

Practically, to check whether
\( p(b) = 1 - p(a) \) for example