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

Outputs checking

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 \( \text{gcd} \) of 2 integers

Problem: \( P(1309, 693) = ? \)

Symmetry relation: \( \forall u, \forall v, \text{gcd}(u,v) = \text{gcd}(v,u) \)

Hence, if \( P(1309, 693) \neq P(693, 1309) \) then \textbf{verdict = fail}

Outline

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

Background on Group Theory

- Group \((E, o) \iff \exists \text{neutral, } \forall x \exists \text{inverse(x), } o \text{ associative}\)
- Symmetric Group \(S_n: \) set of permutations over \( \{1,..,n\} \)
  - if \( x = (x_1,..,x_n) \) \( \circ \) \( x \) denotes \( (x_{\theta(1)},..,x_{\theta(n)}) \)
  - \( S_n \) can be generated by \( \tau = (12) \) and \( \sigma = (12..k) \)

- Group homomorphism from \( S_k \) to \( S_l \)
  - \( \varphi: S_k \rightarrow S_l \) such as \( \varphi(0 \circ 0') = \varphi(0) \circ \varphi(0') \)

Symmetry relation

Program \( p: D_1 \times \ldots \times D_k \rightarrow D_1' \times \ldots \times D_l' \)

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

1) \( \forall \theta \in S_k, \exists \eta \in S_l \text{ such as } \forall x \ p(\theta \circ x) = \eta \circ p(x) \)
2) \( \psi_{k,l} : S_k \rightarrow S_l \) is a group homomorphism

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

<table>
<thead>
<tr>
<th>Methods from java.util.Collections (12 symmetric methods over 19 distincts methods)</th>
<th>Perm. inputs</th>
<th>Perm. output</th>
<th>Symm. relation</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean <code>replaceAll(List A, Object oldVal, Object newVal)</code></td>
<td>A</td>
<td>A</td>
<td>$\psi_{</td>
</tr>
<tr>
<td>object <code>max(Collection A)</code></td>
<td>A</td>
<td>Ref</td>
<td>$\psi_{</td>
</tr>
<tr>
<td>void <code>copy(List B, List A)</code></td>
<td>A</td>
<td>B</td>
<td>$\psi_{</td>
</tr>
<tr>
<td>void <code>sort(List A)</code></td>
<td>A</td>
<td>A</td>
<td>$\psi_{</td>
</tr>
<tr>
<td>List <code>nCopies(int n, Object O)</code></td>
<td>O</td>
<td>Ref</td>
<td>$\psi_{1,n}$</td>
</tr>
</tbody>
</table>

Finding symmetry violations

- The symmetry relation has to be given by the tester: in extension \( \{(\theta, \eta)\}_{\theta \in \mathcal{S}_k} \)
- If \( p(\theta.x) \neq \eta.p(x) \) for any \( x \in D_1 \times \ldots \times D_k \)
  then verdict = 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 [Wood, Miller, Noonan 92] tuned for testing symmetry relations

- Tries exhaustively all the input values into a restricted finite domain \( \mathcal{D} \) 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 \( \forall x \in \mathcal{D} \) when both the executions of \( p(\theta.x) \) and \( p(x) \) terminate
Comparison checks

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

\[ p(\theta, x) = p(\theta \cdot x) \]

but there are \( k! \) permutations in \( S_k \)

needs to know \( \psi_k(\theta) \) for all \( \theta \in S_k \)

Checking only two permutations:

- **Symmetric Testing** requires only to check \( \tau = (12) \) and \( \sigma = (12..k) \)

- **Proposition**: 
  \[ \forall \theta \in S_k, \quad p \circ \theta = \psi_k(\theta) \circ p \quad \Leftrightarrow \quad p \circ \tau = \psi_k(\tau) \circ p \quad \Leftrightarrow \quad p \circ \sigma = \psi_k(\sigma) \circ p \]

- **Sketch of proof**: 
  \( \Rightarrow \) trivial
  \( \Leftarrow \) \[ p \circ \theta = p \circ (\tau \circ \sigma...) = \psi_k(\tau) \circ p \circ (\sigma...) = (\psi_k(\tau) \circ \psi_k(\sigma)...) \circ p \]

  \( = \psi_k(\theta) \circ p \) (because \( \psi_k \) is an homomorphism)

A semi-correct procedure for ST

**In**: program \( p \), finite domain \( D \), \( \psi_k(\tau), \psi_k(\sigma) \)

**Out**: a symmetry violation or a proof that \( \psi_k \) holds over \( D \)

```plaintext
while ( D ≠ \( \emptyset \) )
  pick up \( x \) in \( D \) and \( D := D \setminus \{x\} \)
  let \( \varepsilon := p(x), \varepsilon_1 := p(\tau \cdot x), \varepsilon_2 := p(\sigma \cdot x) \)
  if ( \( \varepsilon_1 ≠ \psi_k(\tau) \cdot \varepsilon \) ) then return violation (\( x, \varepsilon, \varepsilon_1 \))
  if ( \( \varepsilon_2 ≠ \psi_k(\sigma) \cdot \varepsilon \) ) then return violation (\( x, \varepsilon, \varepsilon_2 \))

return (\( \{Q, E, D\} \})
```

Limitations of Symmetric Testing

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

- Impossible to know which inputs among \( x, \tau, \sigma, \varepsilon \) 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 \( \text{GetMid}(\text{Integer}, \text{Integer}, \text{Integer}) \)
  - List \( \text{Trityp}(\text{Integer}, \text{Integer}, \text{Integer}) \)
  - Vector \( \text{min_nb}(\text{Vector}, \text{int}) \)
- Performed on methods from java.lang.Collections
  - void \( \text{sort}() \)
  - void \( \text{copy}() \)
  - Bool \( \text{replaceAll}() \)

Experimental results

<table>
<thead>
<tr>
<th>Method</th>
<th>Symmetry</th>
<th>Mutation</th>
<th>Size of D</th>
<th>time used</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{GetMid} )</td>
<td>( \psi_{\text{MIN}} )</td>
<td>( \psi_{\text{MAX}} )</td>
<td>( 10^{10} )</td>
<td>9.4 sec</td>
</tr>
<tr>
<td>( \text{Trityp} )</td>
<td>( \psi_{\text{MIN}} )</td>
<td>( \psi_{\text{MAX}} )</td>
<td>( 2 \times 10^{10} )</td>
<td>5.3 sec</td>
</tr>
<tr>
<td>( \text{min_nb} )</td>
<td>( \psi_{\text{MIN}} )</td>
<td>( \psi_{\text{MAX}} )</td>
<td>( 2 \times 10^{10} )</td>
<td>5.3 sec</td>
</tr>
</tbody>
</table>

Programs extracted from java.lang.Collections

<table>
<thead>
<tr>
<th>Method</th>
<th>Symmetry</th>
<th>Mutation</th>
<th>Size of D</th>
<th>time used</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{sort} )</td>
<td>( \psi_{\text{MIN}} )</td>
<td>( \psi_{\text{MAX}} )</td>
<td>( 1/2 * 10^4 )</td>
<td>13.3 sec</td>
</tr>
<tr>
<td>( \text{copy} )</td>
<td>( \psi_{\text{MIN}} )</td>
<td>( \psi_{\text{MAX}} )</td>
<td>( 1/2 * 10^4 )</td>
<td>13.7 sec</td>
</tr>
<tr>
<td>( \text{replaceAll} )</td>
<td>( \psi_{\text{MIN}} )</td>
<td>( \psi_{\text{MAX}} )</td>
<td>( 25 * 10^4 )</td>
<td>4606.6 sec</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**  
  [Ammann, 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]

  \[ r(x_1, \ldots, x_n) = r(p(x_1), \ldots, p(x_n)) \]

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  
(\( \because p(\tau(c,b)) = \tau.p((c,b)) \))

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