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 → computing → Outputs

Oracle
↓
Input test data → 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) = ?

<u>Symmetry relation:</u> $\forall u, \forall v,$ *gcd*(u,v) = *gcd*(v,u)

Hence, if P(1309, 693) $\neq$ 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,..,n}
  if $x = (x_1,..,x_n)$ $\theta.x$ denotes $(x_{\theta(1)}, .., x_{\theta(n)})$

  S_n can be generated by τ **= (12)** and σ **= (12..k)**
- **Group homomorphism** from S_k to S_l
  $\varphi : S_k \rightarrow S_l$ such as $\varphi(\theta \text{ o } \theta') = \varphi(\theta) \text{ o } \varphi(\theta')$

Symmetry relation [Gotlieb ISSRE 03]

Program p : $D_1 \times ... \times D_k \rightarrow D_1' \times ... \times D_l'$

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

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

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

<u>Ex:</u> *gcd* satisfies a $\psi_{2,1}$ symmetry relation over $\mathbb{N} \times \mathbb{N}$

Symmetry relation : examples

Methods from `java.util.Collections` (12 symmetric methods over 19 distincts methods)	Perm. inputs	Per. outp	Symm relation
`boolean` **`replaceAll`**`(List A, Object oldVal,Object newVal)`	A	A	$\Psi_{\lvert A\rvert,\lvert A\rvert}$
`Object` **`max`**`(Collection A)`	A	Ret	$\Psi_{\lvert A\rvert,1}$
`void` **`copy`**`(List B, List A)`	A	B	$\Psi_{\lvert A\rvert,\lvert B\rvert}$
`void` **`sort`**`(List A)`	A	A	$\Psi_{\lvert A\rvert,\lvert A\rvert}$
`List` **`nCopies`**`(int n,Object O)`	O	Ret	$\Psi_{1,n}$

Finding symmetry violations

- The symmetry relation has to be given by the tester: in extension $\{ (\theta, \eta) \}_{\forall\theta\in Sk}$
- **If $p(\theta.x) \neq \eta.p(x)$ for any $x \in D_1 \times .. \times D_k$** 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

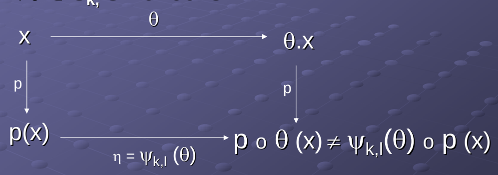
- Tries exhaustively all the input values into a restricted finite domain `D` of $D_1 \times .. \times D_k$

→ in Symmetric Test., **a Cartesian Product iterator**
Ex: {a,b} x {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, $\forall\theta \in S_k$, ST checks:



- **but there are k! permutations in S_k**
- **needs to know $\psi_{k,l}(\theta)$ for all $\theta \in S_k$**

Checking only two permutations:

- **Symmetric Testing** requires only to check
  τ = (12) and σ = (12..k)
- Proposition :

$\forall\theta \in S_k$, $p \circ \theta = \psi_{k,l}(\theta) \circ p \Leftrightarrow \begin{cases} p \circ \tau = \psi_{k,l}(\tau) \circ p \\ p \circ \sigma = \psi_{k,l}(\sigma) \circ p \end{cases}$

- Sketch of proof:

($\Rightarrow$) trivial

($\Leftarrow$) $p \circ \theta = p \circ (\tau \circ \sigma \dots) = \psi_{k,l}(\tau) \circ p \circ (\sigma \dots)$
$= (\psi_{k,l}(\tau) \circ \psi_{k,l}(\sigma) \dots) \circ p$
$= \psi_{k,l}(\theta) \circ p$ (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 ≠ ∅ )
   pick up x in D and D := D \ {x}
   let r:= p(x), rτ := p(τ.x), rσ := p(σ.x)

  if( rτ ≠ ψk,l(τ) . r) then return violation (x,r,rτ)

  if( rσ ≠ ψk,l(σ) . r) then return violation(x,r,rσ)

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, τ.x, σ.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

programs	symmetry	Mutation score	Size of D #test data	time used to prove symmetry
`GetMid`	$\psi_{3,1}$	2/2	$\approx 10^6$	9.4 sec
`trityp`	$\psi_{3,1}$	23/33	$\approx 10^6$	9.6 sec
`sort ° min_nb`	$\psi_{\lvert A\rvert,Ret}$	2/2	$\approx 2*10^6$	53.3 sec

Programs extracted from `java.lang.Collections`

`sort`	$\psi_{\lvert A\rvert,\lvert A\rvert}$	--	$\approx 1/2 * 10^6$	13.1 sec
`copy`	$\psi_{\lvert A\rvert,\lvert B\rvert}$	--	$\approx 1/2 * 10^6$	13.7 sec
`replaceAll`	$\psi_{\lvert A\rvert,\lvert A\rvert}$	--	$\approx 25 * 10^6$	4606.6 sec

(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 Ckecking* [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]

  $r(x_1,..,x_n) \Rightarrow r_f(p(x_1),..,p(x_n))$

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