The Impact of Source Test Case Selection on the Effectiveness of Metamorphic Testing

A.C. Barus, T.Y. Chen, F.-C. Kuo, H. Liu, H.W. Schmidt

huai.liu@rmit.edu.au
Outline

- Motivation
- Background
- Experiment
- Result
- Conclusion
Motivation

- Implementation of metamorphic testing (MT)
  - Metamorphic relations (MRs)
  - Source test cases
  - Follow-up test cases
- Effectiveness of metamorphic testing (MT)
  - Good MRs
  - Good test cases
Motivation

- Generation of source test cases
  - Pure random
  - Special values
Motivation

- Enhancement of random testing
  - Adaptive random testing (ART)
  - Test case diversity
- Source test case generation using ART
Background

- **ART**
  - Contiguous failure regions $\rightarrow$ contiguous non-failure regions
  - Given a test case $t$ does not reveal failure, the input adjacent (similar) to $t$ is very likely to be non-failure-causing
  - An even spread of random test cases will enhance the effectiveness
Background

- Even spread, even distribution, diversity, variant
- Measurement of difference/similarity/distance
- Algorithms for even spreading
Background

- FSCS-ART: One algorithm for even spreading
  - A set of executed test cases
  - A set of candidates
  - The best candidate $\rightarrow$ next test case
Background
Background
Background
Background

- How to select the best candidate
  - max-min:
    - $d_{nn}$: distance to nearest neighbor
    - The largest $d_{nn}$
  - max-sum
    - $d_{sum}$: sum of distances to all executed test cases
    - The largest $d_{sum}$
Background

- Distance measurement
  - Numeric: Euclidean distance, discrepancy, dispersion.
  - Non-numeric: OO distance, model-based
  - More general: category partition based
Background

- Category partition method
  - Identify input parameters and environment variables that affect the execution behaviour
  - Find categories of information that characterize each parameter and variable
  - Partition each category into choices
  - Determine the constraints among choices
  - Construct test frames
  - Generate test case from test frame
Background

- For a given concrete input, we identify its relevant test frame.

- We take two program inputs, determine their categories and choices, and use this information to calculate the distance between them, with a greater distance representing the situation in which the two inputs are more dissimilar.

- Given two program inputs $x$ and $y$, our distance measure is a count of the number of categories in which $x$ and $y$ have different choices.
A transaction processing system handles a large range of monetary quantities (for instance, it may deal with cash and credit transactions, and the transfer of items from a stock inventory).

<table>
<thead>
<tr>
<th>Category</th>
<th>Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit type</td>
<td>Cheque</td>
</tr>
<tr>
<td></td>
<td>Credit</td>
</tr>
<tr>
<td></td>
<td>Inventory item</td>
</tr>
<tr>
<td>Customer type</td>
<td>Business</td>
</tr>
<tr>
<td></td>
<td>Personal</td>
</tr>
<tr>
<td></td>
<td>Government</td>
</tr>
<tr>
<td></td>
<td>Other</td>
</tr>
<tr>
<td>Status</td>
<td>Accepted</td>
</tr>
<tr>
<td></td>
<td>Rejected</td>
</tr>
</tbody>
</table>
Background

Three example inputs

<table>
<thead>
<tr>
<th>Input</th>
<th>Processed Transaction</th>
<th>Category and Choice</th>
</tr>
</thead>
</table>
| $x$   | A cleared cheque payment of $123.45 from Anycorp, a business customer. | Unit type: Cheque  
Customer type: Business  
Status: Accepted |
| $y$   | A credit card payment of $543.21 from Mr. Fred Phisher, a personal customer whose dubious identity leads to the payment being rejected. | Unit type: Credit  
Customer type: Personal  
Status: Rejected |
| $z$   | The dispatch of 12 widgets from stock to Othercorp, a business customer. The order is accepted. | Unit type: Inventory item  
Customer type: Business  
Status: Accepted |
## Background

#### Distance calculation

| Between pair of | DP                                      | DA       | $|DA|$ |
|-----------------|-----------------------------------------|----------|------|
| $(x, y)$        | Unit type:Cheque<br>Unit type:Credt     | Unit type|      |
|                 | Customer type:Business<br>Customer type:Personal | Customer type | 3    |
|                 | Status:Accepted<br>Status:Rejected     | Status   |      |
| $(x, z)$        | Unit type:Cheque<br>Unit type:Inventory Item | Unit type | 1    |
| $(y, z)$        | Unit type:Credt<br>Unit type:Inventory Item | Unit type | 3    |
|                 | Customer type:Personal<br>Customer type:Business | Customer type |      |
|                 | Status:Rejected<br>Status:Accepted     | Status   |      |
Experiment

- **Research question**
  - Can the use of ART in the source test case selection improve the effectiveness of MT?

- **Object programs**
  - grep: regular expression, 6 MRs, 14 categories
  - printtokens/printtokens2: 3 MRs, 28 categories
  - schedule/schedule2: 3 MRs, 34 categories
  - replace: 3 MRs, 24 categories
Experiment

- Techniques under study
  - RT
  - ART with max-min
  - ART with max-sum
  - Aging: a technique for reducing overhead of ART
- Effectiveness metric
  - F-measure
  - Expected number of test cases to detect the first failure
## Result

<table>
<thead>
<tr>
<th>Program</th>
<th>Technique</th>
<th>RT</th>
<th>ART max-min with aging</th>
<th>ART max-sum with aging</th>
</tr>
</thead>
<tbody>
<tr>
<td>grep</td>
<td>RT</td>
<td>N/A</td>
<td>57</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>ART max-min with aging</td>
<td>12</td>
<td>N/A</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>ART max-sum with aging</td>
<td>43</td>
<td>58</td>
<td>N/A</td>
</tr>
<tr>
<td>printtokens</td>
<td>RT</td>
<td>N/A</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>ART max-min with aging</td>
<td>0</td>
<td>N/A</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>ART max-sum with aging</td>
<td>0</td>
<td>0</td>
<td>N/A</td>
</tr>
<tr>
<td>printtokens-2</td>
<td>RT</td>
<td>N/A</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>ART max-min with aging</td>
<td>8</td>
<td>N/A</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>ART max-sum with aging</td>
<td>8</td>
<td>11</td>
<td>N/A</td>
</tr>
<tr>
<td>schedule</td>
<td>RT</td>
<td>N/A</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>ART max-min with aging</td>
<td>7</td>
<td>N/A</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>ART max-sum with aging</td>
<td>7</td>
<td>8</td>
<td>N/A</td>
</tr>
<tr>
<td>schedule-2</td>
<td>RT</td>
<td>N/A</td>
<td>19</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>ART max-min with aging</td>
<td>1</td>
<td>N/A</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>ART max-sum with aging</td>
<td>0</td>
<td>2</td>
<td>N/A</td>
</tr>
<tr>
<td>replace</td>
<td>RT</td>
<td>N/A</td>
<td>16</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>ART max-min with aging</td>
<td>7</td>
<td>N/A</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>ART max-sum with aging</td>
<td>6</td>
<td>13</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Result

- ART max-sum significantly outperformed RT
  - printtokens, printtokens2, schedule2

- ART max-min significantly outperformed RT
  - grep, schedule2, replace

- ART max-sum significantly outperformed ART max-min
  - schedule2

- RT could not significantly outperform ART
Conclusion

- Improvement of MT effectiveness
- ART in source test case selection
- Fewer test cases required to detect the first failure
Thank you