Automated Compatibility Testing Method for Software Logic by Using Symbolic Execution

Keiji Uetsuki
FeliCa Networks, Inc.
Agenda

- Background
- Current Compatibility Testing Process
- Proposed Testing Process
- Application Experiment
- Discussion to apply to commercial software
- Conclusion
Software is everywhere in today’s world.

Context where software is working is changing because world (our society) is changing.

Software must be updated to keep matching context with our real world.

Longer the software lifecycle, more changing demand.
Trend of Software Development

- Scratch
- Maintenance
- Expansion
- Rebuild

Modification 64%

Scratch Development Down Trend

Agile software development process

Development | Iteration | Feedback

FuncA | FuncA | FuncB

FuncC | FuncC | FuncB

FuncB | FuncB | FuncA

FuncA | FuncA | FuncA

FuncD
Issues on Software modification

- Short time of development
  - Modification tends to be treated as easier than scratch

- Lack of knowledge
  - No document, no person in charge

- Compatibility assurance
  - Our interest
Objective of our study

- Resolve compatibility testing issues by applying Symbolic Execution technique to the testing process

- Scope
  - Software logical behavior
    - there are also some more perspectives such as performance but out of scope
  - Software modification does not change the user interface
    - Test cases of the original software can be re-used
Compatibility testing process

Testing process for original software

All outputs and oracles shall be matched
Compatibility testing process

Testing process for modified software

- Test Case For original
- Test Case For modified
- Target Software
- Oracle
- Compare
- Output

All outputs and oracles shall be matched
Test coverage

Can we find all bugs?

Original S/W

Porting, Refactoring

New S/W

R

M

A

Removed

Changed

Added
Test coverage

Can we find all bugs?

Original S/W

Porting, Refactoring

New S/W

COVERED BY TEST CASE

COVERED BY TEST CASE

Removed

Added

Changed
Test coverage

Can we find all bugs?

Original S/W

Porting, Refactoring

New S/W

In fact: weak test coverage or no test case!

Removed

Added

Changed
Proposed testing process

Generate TC from Original Apply it to Modified

Original Software

Symbolic Exec Tool

Output \( R_{o1} \)

Compare

Test Input \( TC_n \)

Output \( R_{n1} \)

Modified Software

Symbolic Exec Tool

Test Input \( TC_o \)

Compare

Output \( R_{o2} \)

Output \( R_{n2} \)
Proposed testing process

Generate TC from Modified 
Apply it to Original

Original Software

Modified Software

Symbolic Exec Tool

Test Input TC₀

Test Input TCₙ

Output R₀₁

Compare

Output R₀₂

Compare

Output Rₙ₁

Symbolic Exec Tool

Output Rₙ₂
Test Coverage

Original S/W

Porting, Refactoring

New S/W

TCn to Original Software

Removed

Added

Changed

M A R

Original S/W

TCo to Modified Software
Application Experiment

- **Experiment 1**
  - Apply this method to the sample application in case of specification modification
  - To verify if we can find three types of modification: Add, Change, Remove

- **Experiment 2**
  - Apply this method to the sample application in case that the application has bugs
  - To verify if we can find three types of bugs
If age of a customer is less than or equal to 3, Free (0% of the normal fee)
If Wednesday, 90% of the normal fee
If age of a customer is greater than or equal to 60, 60% of the normal fee
If sex of a customer is female and her age is greater than or equal to 50, 65% of the normal fee
If the memorial day, 80% of the normal fee
If a customer is local citizen, 50% of the normal fee
If after 3 P.M., 70% of the normal fee
If age of a customer is less than or equal to 12, 40% of the normal fee

Note that greater discount rate is applied when multiple conditions are met
Modified specification

- If age of a customer is less than or equal to 3, Free (0% of the normal fee)
- If Wednesday, 90% of the normal fee
- If age of a customer is greater than or equal to 60, 60% of the normal fee
- If sex of a customer is female and her age is greater than or equal to 55, 65% of the normal fee
- **If the memorial day, 80% of the normal fee**
- If a customer is local citizen, 50% of the normal fee
- If after 3 P.M., 70% of the normal fee
- If age of a customer is less than or equal to 12, 40% of the normal fee
- **If January or February, 67% of the normal fee**
- Note that greater discount rate is applied when multiple conditions are met
Flow chart of the program

**Original Specification**

- Age $\leq 3$? (T) → 0%
- Age $\leq 12$? (T) → 40%
- Citizen? (T) → 50%
- Age $\geq 60$? (T) → 60%
- Age $\geq 50$? (T) → Female? (T) → 65%
- After pm3? (T) → 70%
- Memorial day? (T) → 80%
- Wednesday? (T) → 90%
- F → 100%

**Modified Specification**

- Age $\leq 3$? (T) → 0%
- Age $\leq 12$? (T) → 40%
- Citizen? (T) → 50%
- Age $\geq 60$? (T) → 60%
- Age $\geq 55$? (T) → Female? (T) → 65%
- January? (T) → 67%
- February? (T) → After pm3? (T) → 70%
- After pm3? (T) → Wednesday? (T) → 90%
- F → 100%

Memorial day? is removed
Tool demonstration

- Symbolic Execution Tool for Java: SPF (Symbolic Path Finder)

- See here for detail of SPF:
  - http://babelfish.arc.nasa.gov/trac/jpf/wiki/projects/jpf-symbc/doc
## Test result from the original specification

<table>
<thead>
<tr>
<th>#</th>
<th>Sex</th>
<th>Age</th>
<th>Day of week</th>
<th>local citizen?</th>
<th>Month</th>
<th>memorial day?</th>
<th>Time (hour)</th>
<th>Output (discount rate %)</th>
<th>Output from Modified Spec.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Male</td>
<td>0</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Male</td>
<td>4</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>3</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>Yes</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>Male</td>
<td>60</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>5</td>
<td>Female</td>
<td>50</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>65</td>
<td>100</td>
</tr>
<tr>
<td>6</td>
<td>Female</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>7</td>
<td>Female</td>
<td>13</td>
<td>Monday</td>
<td>Yes</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>80</td>
<td>100</td>
</tr>
<tr>
<td>8</td>
<td>Female</td>
<td>13</td>
<td>Wednesday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>9</td>
<td>Female</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>10</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>15</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>11</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>Yes</td>
<td>0</td>
<td>80</td>
<td>100</td>
</tr>
<tr>
<td>12</td>
<td>Male</td>
<td>13</td>
<td>Wednesday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>13</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

Because the condition for age of customer is changed.

Because the condition “Memorial day?” is removed.
### Test result from the modified specification

<table>
<thead>
<tr>
<th>#</th>
<th>Sex</th>
<th>Age</th>
<th>Day of week</th>
<th>local citizen?</th>
<th>Month</th>
<th>memorial day?</th>
<th>Time (hour)</th>
<th>Output (discount rate %)</th>
<th>Output from Original Spec.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Male</td>
<td>0</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Male</td>
<td>4</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>3</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>Yes</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>Male</td>
<td>60</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>5</td>
<td>Female</td>
<td>55</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>65</td>
<td>65</td>
</tr>
<tr>
<td>6</td>
<td>Female</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>February</td>
<td>No</td>
<td>15</td>
<td>67</td>
<td>100</td>
</tr>
<tr>
<td>7</td>
<td>Female</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>15</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>8</td>
<td>Female</td>
<td>13</td>
<td>Wednesday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>9</td>
<td>Female</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>January</td>
<td>No</td>
<td>0</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>10</td>
<td>Female</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>67</td>
<td>100</td>
</tr>
<tr>
<td>11</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>67</td>
<td>100</td>
</tr>
<tr>
<td>12</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>January</td>
<td>No</td>
<td>15</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>13</td>
<td>Male</td>
<td>13</td>
<td>Wednesday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>14</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>15</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>January</td>
<td>No</td>
<td>0</td>
<td>67</td>
<td>100</td>
</tr>
</tbody>
</table>

Because the condition for Month is newly added.
Bug implementation

Original Specification

Bug implemented program

Change

Citizen? is removed

Add
## Test result from the original program

<table>
<thead>
<tr>
<th>#</th>
<th>Sex</th>
<th>Age</th>
<th>Day of week</th>
<th>local citizen?</th>
<th>Month</th>
<th>memorial day?</th>
<th>Time (hour)</th>
<th>Output (discount rate %)</th>
<th>Output from Bug impl.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Male</td>
<td>0</td>
<td></td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>No</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Male</td>
<td>4</td>
<td></td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>No</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>3</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>Yes</td>
<td>December</td>
<td>No</td>
<td>No</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>4</td>
<td>Male</td>
<td>60</td>
<td></td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>No</td>
<td>60</td>
<td>100</td>
</tr>
<tr>
<td>5</td>
<td>Female</td>
<td>50</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>65</td>
<td>100</td>
</tr>
<tr>
<td>6</td>
<td>Female</td>
<td>13</td>
<td></td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>7</td>
<td>Female</td>
<td>13</td>
<td></td>
<td>Yes</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>80</td>
<td>100</td>
</tr>
<tr>
<td>8</td>
<td>Female</td>
<td>13</td>
<td></td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>9</td>
<td>Female</td>
<td>13</td>
<td></td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>10</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>15</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>11</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>Yes</td>
<td>0</td>
<td>80</td>
<td>100</td>
</tr>
<tr>
<td>12</td>
<td>Male</td>
<td>13</td>
<td>Wednesday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>13</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

Because the condition “Citizen?” is **removed**

Because inequality sign of the condition for age is **changed**
### Test result from the bug implemented program

<table>
<thead>
<tr>
<th>#</th>
<th>Sex</th>
<th>Age</th>
<th>Day of week</th>
<th>local citizen?</th>
<th>Month</th>
<th>memorial day?</th>
<th>Time (hour)</th>
<th>Output (discount rate %)</th>
<th>Output from Original Spec.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Male</td>
<td>0</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Male</td>
<td>4</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>3</td>
<td>Male</td>
<td>61</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>4</td>
<td>Female</td>
<td>55</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>65</td>
<td>65</td>
</tr>
<tr>
<td>5</td>
<td>Female</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>February</td>
<td>No</td>
<td>0</td>
<td>67</td>
<td>100</td>
</tr>
<tr>
<td>6</td>
<td>Female</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>15</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>7</td>
<td>Female</td>
<td>13</td>
<td>Wednesday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>8</td>
<td>Female</td>
<td>13</td>
<td>Tuesday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>9</td>
<td>Female</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>10</td>
<td>Female</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>January</td>
<td>No</td>
<td>0</td>
<td>67</td>
<td>100</td>
</tr>
<tr>
<td>11</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>November</td>
<td>No</td>
<td>0</td>
<td>67</td>
<td>100</td>
</tr>
<tr>
<td>12</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>15</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>13</td>
<td>Male</td>
<td>13</td>
<td>Wednesday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>14</td>
<td>Male</td>
<td>13</td>
<td>Tuesday</td>
<td>No</td>
<td>December</td>
<td>No</td>
<td>0</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>15</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>January</td>
<td>No</td>
<td>0</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>16</td>
<td>Male</td>
<td>13</td>
<td>Monday</td>
<td>No</td>
<td>January</td>
<td>No</td>
<td>0</td>
<td>67</td>
<td>100</td>
</tr>
</tbody>
</table>

Because the condition “Tuesday?” is newly added
Discussion

- Is there any issue when we apply the method to a commercial software?
Issues to be solved

- Tool dependency
- Scalability
- Education
ISSUE: Tool dependency

- Our method uses Symbolic execution tool to generate test cases

- Some restrictions on the tools
  - Supported Language: C, C++, Java, JavaScript, etc.
  - Supported Input Variables: bit array, integer, floating point, etc.
ISSUE: Scalability

- Sample program is okay, but commercial program is much bigger

- Issues on scalability has been studied

  1) Path explosion
     - Too many path to be treated

  2) Checking differences of test cases
     - Too many differences to be checked by hand
Solution for path explosion

- Path cutting technique
  - Add constraints about coverage level
  - This study introduces condition coverage instead of full path coverage

Source: Enhancing Symbolic Execution to Test the Compatibility of Re-engineered Industrial Software
Solution for path explosion

- Variable grouping
  - If functions of the software can be treated independently, the input variables can be divided

Source: Enhancing Symbolic Execution to Test the Compatibility of Re-engineered Industrial Software
Bigger modification is made, more differences may be made.

Therefore we must stop big-bang testing, instead, frequently testing at small modification.

Since the testing process is automated, we can do it.
Symbolic Execution technique and tools are now open

However it is still not easy to understand and use

Due to the tool installation, restrictions, no user community in Japan

Now we are creating the user community!
Conclusion

- We proposed a new software testing method for logic compatibility verification.

- By using this method compatibility of logical behavior was exhaustively verified and also full path coverage was achieved.

- From the experimental results, it was verified that the method could detect all the three types of specification changes and bugs.

- Our next step is to apply the method to many types of real software to clarify its limitation or restriction which depends on the types.