Mutation-based spreadsheet debugging

Birgit Hofer and Franz Wotawa, Institute for Software Technology, {bhofer,wotawa}@ist.tugraz.at
25.11.2013
Problem statement

• Given a spreadsheet and indications about its misbehavior, i.e., expected cells’ values.

• Search for the root causes and potential corrections using mutation operators.

• Ideas based on previous work on program debugging based on mutations / genetic programming:
Motivating example

- Example from economy (Reinhart and Rogoff, Roosevelt Institute)!
- Fault in spreadsheet lead to wrong conclusions

<table>
<thead>
<tr>
<th>Country</th>
<th>Coverage</th>
<th>30 or less</th>
<th>30 to 60</th>
<th>60 to 90</th>
<th>90 or above</th>
<th>30 or less</th>
</tr>
</thead>
<tbody>
<tr>
<td>US</td>
<td>1946-2009</td>
<td>3.4</td>
<td>3.3</td>
<td>n.a.</td>
<td>-2.0</td>
<td>n.a.</td>
</tr>
<tr>
<td>UK</td>
<td>1946-2009</td>
<td>2.4</td>
<td>2.5</td>
<td>2.4</td>
<td>n.a.</td>
<td>n.a.</td>
</tr>
<tr>
<td>Sweden</td>
<td>1946-2009</td>
<td>3.6</td>
<td>2.9</td>
<td>2.7</td>
<td>n.a.</td>
<td>6.3</td>
</tr>
<tr>
<td>Spain</td>
<td>1946-2009</td>
<td>1.5</td>
<td>3.4</td>
<td>4.2</td>
<td>n.a.</td>
<td>9.9</td>
</tr>
<tr>
<td>Portugal</td>
<td>1952-2009</td>
<td>4.8</td>
<td>2.6</td>
<td>0.3</td>
<td>n.a.</td>
<td>7.9</td>
</tr>
<tr>
<td>New Zealand</td>
<td>1948-2009</td>
<td>2.5</td>
<td>2.9</td>
<td>3.5</td>
<td>-7.9</td>
<td>2.6</td>
</tr>
<tr>
<td>Netherlands</td>
<td>1956-2009</td>
<td>4.1</td>
<td>2.7</td>
<td>1.1</td>
<td>n.a.</td>
<td>6.4</td>
</tr>
<tr>
<td>Norway</td>
<td>1947-2009</td>
<td>3.4</td>
<td>5.1</td>
<td>n.a.</td>
<td>n.a.</td>
<td>5.4</td>
</tr>
<tr>
<td>Japan</td>
<td>1946-2009</td>
<td>7.0</td>
<td>4.0</td>
<td>1.0</td>
<td>0.7</td>
<td>7.0</td>
</tr>
<tr>
<td>Italy</td>
<td>1951-2009</td>
<td>5.4</td>
<td>2.1</td>
<td>1.6</td>
<td>1.0</td>
<td>5.6</td>
</tr>
<tr>
<td>Ireland</td>
<td>1948-2009</td>
<td>4.4</td>
<td>4.5</td>
<td>4.0</td>
<td>2.4</td>
<td>2.9</td>
</tr>
<tr>
<td>Greece</td>
<td>1970-2009</td>
<td>4.0</td>
<td>0.3</td>
<td>2.7</td>
<td>2.9</td>
<td>13.3</td>
</tr>
<tr>
<td>Germany</td>
<td>1946-2009</td>
<td>3.9</td>
<td>0.9</td>
<td>n.a.</td>
<td>n.a.</td>
<td>3.2</td>
</tr>
<tr>
<td>France</td>
<td>1946-2022</td>
<td>4.9</td>
<td>2.7</td>
<td>3.0</td>
<td>n.a.</td>
<td>5.2</td>
</tr>
<tr>
<td>Finland</td>
<td>1946-2023</td>
<td>3.8</td>
<td>2.4</td>
<td>5.5</td>
<td>n.a.</td>
<td>7.0</td>
</tr>
<tr>
<td>Denmark</td>
<td>1946-2024</td>
<td>3.5</td>
<td>1.7</td>
<td>2.4</td>
<td>n.a.</td>
<td>5.6</td>
</tr>
<tr>
<td>Canada</td>
<td>1946-2025</td>
<td>1.9</td>
<td>3.6</td>
<td>4.1</td>
<td>n.a.</td>
<td>2.2</td>
</tr>
<tr>
<td>Belgium</td>
<td>1946-2026</td>
<td>n.a.</td>
<td>4.2</td>
<td>3.1</td>
<td>2.6</td>
<td>n.a.</td>
</tr>
<tr>
<td>Austria</td>
<td>1946-2027</td>
<td>5.2</td>
<td>3.3</td>
<td>-3.6</td>
<td>n.a.</td>
<td>5.7</td>
</tr>
<tr>
<td>Australia</td>
<td>1946-2028</td>
<td>3.2</td>
<td>4.9</td>
<td>4.0</td>
<td>n.a.</td>
<td>5.9</td>
</tr>
</tbody>
</table>

Birgit Hofer and Franz Wotawa, Institute for Software Technology, {bhofer,wotawa}@ist.tugraz.at
Past work

• GoalDebug [Abraham and Erwig; 2007] – Ranked list of repair suggestions

• Model-based approaches [Jannach and Engler; ] and [Hofer et al.; 2013]

• Trace-based approaches [Ruthruff et al.; 2003] – Similar to Tarantula,...
Past work [Hofer et al.; 2013]

• Have a look at the following simple example

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Inputs</td>
<td>Outputs</td>
<td>Expected Outputs</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>=A3*A5</td>
<td>=B3+B4</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>=A4*A6</td>
<td>=B4+B5</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>=A7*A5</td>
<td>=B6+B5</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>=SUMME(A3:A7)</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Inputs</td>
<td>Outputs</td>
<td>Expected Outputs</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>6</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>6</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>6</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Birgit Hofer and Franz Wotawa, Institute for Software Technology, {bhofer,wotawa}@ist.tugraz.at
Cones / Slicing / SFL / etc.

Each cone is a slice and considered as a “trace” (for SFL)
Model-based approaches

Convert formulas to constraints and add special predicates \( AB(C) \):

\[
\begin{array}{c|c|c}
\text{A} & \text{B} & \text{C} \\
\hline
2 & =A3*A5 & =B3+B4 \\
3 & =A4*A6 & =B4+B5 \\
5 & =A7*A5 & =B6+B5 \\
6 & =\text{SUMME}(A3:A7) & \\
\end{array}
\]

\( AB(B2) \) or \( (B2 = A3*A5) \)

\( AB(B3) \) or \( (B3 = A4*A6) \)

....

Choose \( AB(C) \) such that no contradiction occurs anymore!
Obtained results

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Avg</th>
<th>Min</th>
<th>Max</th>
<th>Std.dev</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of formulas</td>
<td>225.0</td>
<td>6</td>
<td>4,170</td>
<td>384.9</td>
<td>104.5</td>
</tr>
<tr>
<td>Number of incorrect output cells</td>
<td>1.7</td>
<td>1</td>
<td>22</td>
<td>1.9</td>
<td>1</td>
</tr>
<tr>
<td>Number of correct output cells</td>
<td>64.9</td>
<td>0</td>
<td>2,962</td>
<td>162.5</td>
<td>24</td>
</tr>
</tbody>
</table>

Table 1. Characteristics of the created mutants.

Fig. 1. Comparison of the SFL, SENDYS, the Union and Intersection of the cones in terms of the amount of formula cells that must be investigated.

Birgit Hofer and Franz Wotawa, Institute for Software Technology, {bhofer,wotawa}@ist.tugraz.at
25.11.2013
Obained results (cont)

<table>
<thead>
<tr>
<th>Technique</th>
<th>$\bar{\phi}$ absolute ranking</th>
<th>$\bar{\phi}$ relative ranking</th>
<th>$\bar{\phi}$ computation time (in ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Union (cones of faulty output)</td>
<td>34.8</td>
<td>29.3%</td>
<td>14.0</td>
</tr>
<tr>
<td>Intersection (cones of faulty output)</td>
<td>33.6</td>
<td>27.5%</td>
<td>13.9</td>
</tr>
<tr>
<td>SFL</td>
<td>32.9</td>
<td>27.1%</td>
<td>15.0</td>
</tr>
<tr>
<td>SENDYS</td>
<td>31.9</td>
<td>27.0%</td>
<td>63.9</td>
</tr>
<tr>
<td>ConBug</td>
<td>33.9</td>
<td>27.9%</td>
<td>631.7</td>
</tr>
</tbody>
</table>

Table 3. Average ranking and computation time of union, intersection, SFL, SENDYS, and ConBug. The column '$\bar{\phi}$ relative ranking' shows the average ranking of the statement normalized to the number of formulas per spreadsheet. This evaluation comprises 227 spreadsheets.

Research question: ConBug “better” than SENDYS when used for imperative languages. Why is this not the case in the spreadsheet domain?
Mutation-based diagnosis
Short description

3 parts:
1. use cones for focusing on relevant statements
2. generate mutations for relevant statements
3. check correctness (thus eliminating candidates)

ad 1. cones:

\[ \text{CONE}(c) = c \cup \bigcup_{c' \in \text{FORMULA}(c)} \text{CONE}(c') \]
Mutations

ad 2. mutations:

▪ Constants
  ▪ Change Boolean values
  ▪ Permutate digits
  ▪ Increase or decrease number by 1 – Use a random number instead
  ▪ Change the sign

▪ References
  ▪ Randomly increase/decrease the borders of areas
  ▪ Randomly change a single reference • Formulas
  ▪ Replace a binary operator (e.g., ‘+’) with another binary operator (e.g., ‘-’)
  ▪ Replace a formula (e.g., ‘SUM’) with another formula which can process the same arguments (e.g., ‘AVG’)
  ▪ Remove parts of a formula (e.g., ‘A3+A4*3’→‘A3*3’)
  ▪ Relocate parts of a formula (e.g., ‘if(A1 > 1; A2; A3)’→‘if (A1 > 1; A3; A2)’)

Birgit Hofer and Franz Wotawa, Institute for Software Technology, {bhofer,wotawa}@ist.tugraz.at
25.11.2013
Mutations (cont)

- Crossover not considered

- Fitness function (ako ad 3. checking):

\[
\text{FITNESS}(m) = - (\omega_{\text{initial}} \cdot |M \cap O| + \omega_{\text{new}} \cdot |M \setminus O|)
\]

Output cells of mutants with wrong values

Output cells with wrong values

Throw away all mutants where

\[
\text{FITNESS}(m) < -\omega_{\text{initial}} \cdot |O|
\]
The mutation-based diagnosis algorithm

- computes new mutants until results is found, or
- limit is reached!

```
Algorithm 1 Evolutionary algorithm

1 procedure EVOLUTION(s, o)  > s: Spreadsheet, o: set of (cell, expectedOutput) pairs
2    cones ← {}             
3  for (cell, expectedOutput) ∈ o do
4    if value(cell) ≠ expectedOutput then
5      Cones ← Cones ∪ CONE(cell)
6    end if
7  end for
8  i ← 0  > Generation counter
9  P ← {m}  > Population
10  while i < maxGenerations do
11    G ← {}  > New generation
12    j ← 0
13    while j < populationSize do
14      Randomly select c from cones
15      Randomly select m₁ from P
16      m₂ ← MUTATE(m₁, c)
17      if FITNESS(m₂) = 0 then
18        return m₂  > Solution found
19      end if
20      if FITNESS(m₂) ≥ FITNESS(m) then
21        G ← G ∪ {m₂}  > Improved mutant
22      end if
23      j = j + 1
24    end while
25    P ← P ∪ G
26    i = i + 1
27  end while
28  return FAIL
29 end procedure
```
Empirical results

Preliminaries:

- EUSES spreadsheet corpus
- Remove some spreadsheets (no input cells, too small,..). Approx. 500 spreadsheets remained (with 6..4,000 formulas, and on avg. 225).
- Both weights set to 1
- 16 trails per spreadsheet

Obtained results:

- In about 55 % of the cases the fault has been located and corrected (at least once)
- Computation time ranged between 2 ms and 40 minutes (avg. 16.3 seconds)
Empirical results (cont)

approx. 24% of cases the fault could always be found

Success rate
Conclusions

• Presented a mutation-based approach for fault localization and correction for spreadsheets

• Solution = Mutation that explains faulty behavior

• Not always successful (like the other mutation-based approaches)
Thank you for your attention!

Birgit Hofer and Franz Wotawa, Institute for Software Technology, 
{bhofer,wotawa}@ist.tugraz.at

25.11.2013