Fault-Based Testing

Reading Assignment: Regression testing


Structural Test Data Selection

- Random
- Coverage based
  - Control flow
  - Data flow
- Fault-based
  - Error (fault) seeding
    - e.g., mutation testing
  - Fault constraints
    - E.g., RELAY
- Error-based (Failure-based)
  - domain and computation based
  - use representations created by symbolic execution

Remember: comments on dependence based testing coverage

- for selecting test cases
  - syntactic dependence alone is not adequate
    - the number of syntactic dependencies in a program can be quadratic in the number of statements
    - a given syntactic dependence may be demonstrated by (infinitely) many paths
    - propagation of a fault through a particular path may depend on the selection of input data
      - must use semantic information
  - Dependencies are imprecise due to compound data structures and infeasible paths

Can we do better than this?????

Fault-Based Techniques

- For each statement try to select test data that will expose faults at that statement
- For example, mutation testing monitors effectiveness
  - Fault constraints -- instead of monitoring if the selected test data kills a mutant, determine the necessary and sufficient conditions to guarantee that the fault is revealed if it exists

Can exercise a dependence relationship but not reveal the fault

\[
\begin{align*}
X := Y + Z; \\
X := X * (Z - 1);
\end{align*}
\]
Relay Model

- Selective semantic information + syntactic dependency information
- origination of a fault
- computational transfer of a fault
- propagation of a fault (based on data and control flow)
- Goal: Define necessary and sufficient conditions for detecting certain classes of faults

Overview of Relay Model

- origination
  - introduction of potential failure at smallest (valued) subexpression containing fault
- transfer
  - "movement" of potential failure in program
    - Within the originating statement
      - computational transfer
    - From one statement to the next
      - data dependence transfer
      - control dependence transfer

Example

Correct:

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>C</td>
<td>B</td>
<td>+</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>C</td>
<td>B</td>
<td>*</td>
<td>C</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>X</td>
<td>=</td>
<td>B</td>
<td>*</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>D</td>
<td>&lt;</td>
<td>X</td>
<td>-</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

output Y

Y: = (2 * X) + C

What test data would reveal this fault?

Example

module

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>C</td>
<td>B</td>
<td>+</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>B</td>
<td>*</td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>X</td>
<td>:</td>
<td>B</td>
<td>*</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>D</td>
<td>&lt;</td>
<td>X</td>
<td>-</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

output Y

Y: = (X * * 2) + C

Example

ORIGINATION (NODE 2),
NO COMPUTATIONAL TRANSFER AT NODE 2

Example

ORIGINATION (NODE 2),
NO COMPUTATIONAL TRANSFER AT NODE 2
To Guarantee Detection

Step 1: guarantee introduction of potential failure at statement containing hypothetical fault
- origination condition
- computational transfer conditions at statement
- called original state potential failure condition

Step 2: guarantee transfer of potential failure along information flow to some output
- called transfer set condition

Step 1a
- origination condition
- guarantees introduction of potential failure in smallest subexpression
- $\exp \neq \exp^*$
- defined for fault
- suppose $(b+1)$ instead of $(b+c)$
  $\Rightarrow c = 1$
Step 1b

- computational transfer condition for a statement
  - Given statement ≠ Faulty statement
  - $\text{exp1} <\text{op}> \text{exp2} ≠ \text{exp1} <\text{op}> \text{exp2}′$
  - defined for operator and fault
  - e.g., $d∗(b + c) ≠ d∗(b + 1) ⇒ (c = 1) ∧ (d ≠ 0)$
  - sometimes fault independent
    $d∗(\text{exp}) ≠ d∗(\text{exp})′$
    $⇒ d ≠ 0$

Simple Example

1. input X, Y, Z
2. $A = X + Z$
3. $B = A + X$
4. $C = A + Y$
5. $D = B∗C$
6. output D

Test case: $X=1$, $Y=-3$

Necessary but not sufficient?

1. input X, Y, Z
2. $A = X + Z$
3. $B = A + X$
4. $C = A + Y$
5. $D = B∗C$

Faulty statement

- $A = X + Z$
- $B = A + X$
- $C = A + Y$
- $D = B∗C$

Must consider the interaction among faulty values

1. input X, Y, Z
2. $A = X + Z$
3. $B = A + X$
4. $C = A + Y$
5. $D = B∗C$

Correct output

<table>
<thead>
<tr>
<th>test case</th>
<th>input</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>faulty</td>
<td>1 1 1</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>correct</td>
<td>1 1 1</td>
<td>1 2 1 2</td>
</tr>
<tr>
<td>faulty</td>
<td>1 1 1</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>correct</td>
<td>1 1 1</td>
<td>1 2 1 2</td>
</tr>
</tbody>
</table>

Page 4
Transfer Condition

- Condition that guarantees transfer
  - Must know points of interaction
    - Places where two or more potential failures come together
- Transfer set defines locations of potential interaction
  - Notation: \((U_n, V_m)\) means faulty value for variable \(U\) at node \(n\) transfers to variable \(V\) at node \(m\)
- Transfer route defines chains of transfer set elements that can be combined to form a path

Example

- Transfer Set: \(((A_2, B_3), (B_3, D_5), (A_2, C_4), (C_4, D_5))\)
- Transfer Routes
  1. \((A\text{ at 2 transfers to } B\text{ at 3})\) and \((A\text{ at 2 does not transfer to } C\text{ at 4})\) and \((B\text{ at 3 transfers to } D\text{ at 5})\)
  2. \((A\text{ at 2 does not transfer to } B\text{ at 3})\) and \((A\text{ at 2 transfers to } C\text{ at 4})\) and \((C\text{ at 4 transfers to } D\text{ at 5})\)
  3. \((A\text{ at 2 transfers to } B\text{ at 3})\) and \((A\text{ at 2 transfers to } C\text{ at 4})\) and \((B\text{ at 3 and } C\text{ at 4 transfer to } D\text{ at 5})\)

Construction of Transfer Route

- Different ways to transfer along same set, depending on which portions of chains transfer and which do not
  - A transfer route is a subset of the nodes in a transfer set where transfer does and does not occur
  - A transfer route defines where actual interactions occur

Transfer Routes for Example

1. \((A\text{ transfers to } B\text{ at 3})\) and \((A\text{ does not transfer to } C\text{ at 4})\) and \((B\text{ transfers to } D\text{ at 5})\)
2. \((A\text{ does not transfer to } B\text{ at 3})\) and \((A\text{ transfers to } C\text{ at 4})\) and \((C\text{ transfers to } D\text{ at 5})\)
3. \((A\text{ transfers to } B\text{ at 3})\) and \((A\text{ transfers to } C\text{ at 4})\) and \((B\text{ and } C\text{ transfer to } D\text{ at 5})\)
Condition for First Transfer Route

(A at 2 transfers to B at 3) and
(A at 2 does not transfer to C at 4) and
(B at 3 transfers to D at 5)

- Origination Condition
  \( y \neq z \)

- Transfer Route Conditions:
  \( x \neq 0 \land y = 0 \land a \neq 0 \Rightarrow \) false

Example:

\[ \begin{align*}
A &= 2 \\
B &= 3 \\
C &= 4 \\
D &= 8
\end{align*} \]

Input: \( X, Y, Z \)

\[ \begin{align*}
A &= X + Y \\
B &= A \times X \\
C &= A \times Y \\
D &= B \times C
\end{align*} \]

Output: D

Condition for Second Transfer Route

(A does not transfer to B at 3) and
(A transfers to C at 4) and
(C transfers to D at 5)

- Origination Condition
  \( y \neq z \)

- Transfer Route Conditions:
  \( x = 0 \land y \neq 0 \land b \neq 0 \Rightarrow false \)

Example:

\[ \begin{align*}
A &= 3 \\
B &= 3 \\
C &= 3 \\
D &= 9
\end{align*} \]

Input: \( X, Y, Z \)

\[ \begin{align*}
A &= X + Y \\
B &= A \times X \\
C &= A \times Y \\
D &= B \times C
\end{align*} \]

Output: D

Condition for Third Transfer Route

(A at 2 transfers to B at 3) and
(A at 2 transfers to C at 4) and
(B at 3 and C at 4 transfer to D at 5)

- Origination Condition
  \( y \neq z \)

- Transfer Route Conditions:
  \( x \neq 0 \land y \neq 0 \land b \neq 0 \Rightarrow false \)

Example:

\[ \begin{align*}
A &= 2 \\
B &= 2 \\
C &= 3 \\
D &= 8
\end{align*} \]

Input: \( X, Y, Z \)

\[ \begin{align*}
A &= X + Y \\
B &= A \times X \\
C &= A \times Y \\
D &= B \times C
\end{align*} \]

Output: D

Failure condition

- Original state potential failure condition and transfer condition
- if test data satisfies failure condition (fc) and failure → fault
- if test data satisfies fc and no failure → no fault (equivalent program for this domain)
- if can’t satisfy fc → try another transfer set
- if can’t satisfy fc for all transfer sets → no fault

Relay Fault Based Approach

- recognizes what is needed to transfer to output
- other fault based techniques:
  - do not deal with how to select test data that transfers
  - may recognize need to transfer but provide no guidance in test data selection (assume transfer “usually” occurs)
  - do not consider control dependence
  - none discuss interactions for a single fault/multiple faults -- they assume that there is a single fault or if there is more than one that there is no interaction
Relay Fault Based Approach

• Defines what is needed to reveal a fault at a statement
  • A general procedure that could be applied to any “atomic” fault

• Defines what is needed to propagate erroneous values to output
  • If interaction is not accounted for, then the constraints are neither necessary nor sufficient
  • Assumptions about single faults are now very questionable
  • Can not assume constraints are necessary
    • Other faults might interact with hypothesized faults
    • a very negative result