CS @ ILLINOIS
COMPUTER SCIENCE

Repairing Test Dependence

Wing Lam
University of Illinois at Urbana-Champaign
winglam2@illinois.edu

RESEARCH PROBLEM

- Test dependence causes inconsistent test result when tests are run in different execution orders.
- There are two serious consequences.
  - False positives: Test fails even when the software is correct.
  - False negatives: Test passes even when a bug exists.

Test code containing test dependence.

```
static int x = 0;
void testXDefaultVal() {
    assertEquals(0, x);
}
void testChangingXVal() {
    x = 1 + x;
    assertEquals(1, x);
}
```

(Depends on)

APPROACH

- Automatically applies various refactorings to the code under test and test code as they are compiled.
- Most common root cause of test dependence is side-effecting access to shared global variables.
- Our prototype addresses side-effecting tests in two phases.
  1. Determine the initial value of every global variable.
  2. For every test, reassign all occurrences of global variables whose value is potentially read during the test's execution.

Repaired test code.

```
static int x = 0;
void testXDefaultVal() {
    x = 0;
    assertEquals(0, x);
}
void testChangingXVal() {
    x = 0;
    x = 1 + x;
    assertEquals(1, x);
}
```

RESULTS

- Subject programs were chosen from our previous study.
- Had no knowledge of whether the programs actually contained dependent tests.
- Measure the number of dependent tests exposed by applying test prioritization with and without our prototype.
- With a lower-bound number from a previous study, our prototype eliminates
  - 10.8% of human-written dependent tests
  - 12.5% of the automatically-generated dependent tests

Subject programs used in our evaluation.

Program	LOC CUT	LOC Tests	# Tests Human	# Tests Auto	Version
Crystal	4573	1302	78	3198	1.0.20111015
JFreechart	92255	49942	2234	2438	1.0.15
Joda-Time	27183	51492	3875	2234	b609d7d66d
Synoptic	5317	2758	118	2467	d5ea6fb3157e
XML Security	18255	3807	108	665	1.0.4

Dependent tests exposed by applying four test prioritization algorithms.

Program	Number of dependent tests: Human tests Original program	Human tests Repaired program	Auto tests Original program	Auto tests Repaired program
Crystal	5	2	51	43
JFreechart	3	3	5	5
Joda-Time	1	0	224	157
Synoptic	0	0	2	2
XML Security	4	4	78	77
Total	13	9	360	284

RELATED & FUTURE WORK

Related work

- Tests could be required to run in a given order.
  - Prevents the use of test selection and prioritization.
- Execute each test in a separate virtual machine.
  - Significantly increases test execution time.
- Other related work only detects test dependence.
  - Our approach *repairs* test dependence.
  - Repaired tests yield consistent results on a *single, standard* JVM.

Future work

- Our prototype fixes test dependence for only variables that are of primitive or String types.
  - Improve the prototype to support variables of any type.
- Expand our approach to handle additional causes of test dependence.
  - Such as access to a database or file system and concurrent programs.
- By covering additional causes of test dependence, our approach can repair all cases of test dependence.

This work is supported by the US National Science Foundation under grant no. CNS-1434582.