Dynamic Data-Flow Testing
Mattia Vivanti - University of Lugano
Data flow testing

```java
public class DummyDivision {
    int i;
    public DummyDivision() {
        i=1;
    }
    public void resetI(){
        i=0;
    }
    public int dividePerI(int j){
        return j/i;
    }
}
```
Data flow testing

- Program under test
- Test cases
- Static data flow analysis
- Def-use pairs

- Analyzed by
- Computes
- Targeted by
Data flow testing

Statically computed data flow information does not capture the right information for data flow testing.

Monday, June 16, 14
Data flow testing

- static data flow analysis

- computes

- def-use pairs

- Conservative choices and approximations to address dynamic binding, inter-procedural control flow and scalability

- targeted by

- test cases

Monday, June 16, 14
Frankl and Weiss, “An experimental comparison of the effectiveness of branch testing and data flow testing,” *TSE* 1993

Dynamic Data-Flow Testing

Detect precise data flow information by observing data flow events dynamically.

Incremental computation of test targets by combining dynamic, static analysis and testing.
DReaDs: dynamic reaching definitions analysis

monitors definitions and uses at the memory level

monitors associations between instances to identify nesting of states

public class A {
    private I b;

    public void methOfA() {
        ...
        b = new B("msg");
        ...
    }
    ...
}
DReaDs: dynamic reaching definitions analysis

monitors definitions and uses at the memory level

monitors associations between instances to identify nesting of states

```java
public class A {
    private I b;

    public void methOfA() {
        ...
        b = new B("msg");
        ...
    }
}
```

```
ACTIVE Definitions
{B.s, A.b, A.b.s}

Definition EVENTS
{B.s in B.<init>()
A.b in A.methOfA(),
A.b.s in A.methOfA()}
```
Identification of test targets

Test Case 1

```java
public void setup(){
    this.b = "ciao";
}
```
definitions observed at method exit

```
public void read(){
    print(this.b);
}
```
uses observed in the method

Def: `this.b`  match  Use: `this.b`

Test Case 2

Monday, June 16, 14
Dynamic Data Flow Testing

DReaDs: dynamic reaching definitions analysis

dynamic data flow information for

test targets inference

yet-to-execute definition use pairs for

test case generation

execution traces for

initial test suite
Status of Research

DReaDs: Dynamic Reaching Definitions Analysis for Java

Case study: 5 Java projects, 1531 classes, 88000 eloc.

<table>
<thead>
<tr>
<th></th>
<th># Definitions</th>
<th>% Missed by the other technique</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dreads</td>
<td>169,495</td>
<td>96%</td>
</tr>
<tr>
<td>DaTeC</td>
<td>28,929</td>
<td>23%</td>
</tr>
</tbody>
</table>

Current Work

Dynamic Data Flow Testing

Initial test suite → dynamic data flow information for → test targets inference → yet-to-execute definition use pairs for → test case generation → DReaDs: dynamic reaching definitions analysis → initial test suite

Finalize implementation

Evaluation:

- can we discover a relevant set of definition use pairs using dynamic data flow testing?
- how effective are generated test cases?