Antifragile Software and Genetic Improvement

Martin Monperrus
University of Lille & Inria, France

CREST Open Workshop on Genetic Improvement
Jan 2016

```java
try {
    prepare_meringue()
    add_cream()
    make_balls()
    add_chocolate_shavings()
}
catch (MissingChocolateEx e) {
    use_nutella();
}
```

« Anticipated errors »
Expected errors in test suites

Blue and white tests specify anticipated errors
White tests specify resilience

- Test colors
  - Pink
  - Blue
  - White

```java
public String getProperty(String s) {
    String res = null;
    try{
        res = getPropertyFromFile(s);
    }catch(PropertyNotFoundException pnfe){
        return null;
    }
    return res;
}

@Test
public void testAbsentProperty(){
    assertNull(getProperty("@%^ù*µ;?!"));
}
```
Empirical results

![Graph showing empirical results for different software projects.](image_url)
Our intuition: simulating unforeseen errors by perturbing test case execution
Short circuit testing: Exception injection in test suites

White test + throw new X() =
The try-catch blocks that keep the tests passing under exception injection are capable to handle unanticipated errors.

We call them "source-independent".

Empirical assessment of resilience against unforeseen errors
Empirical Evaluation

- shindig-java-gadgets
- commons-lang
- sonar core
- shindig-common
- jbehave core
- joda time
- sonar plugin api
- spojo core

- dependent
- independent
Test driver

```java
try{
    // instrumentation code
    if(Controller.isCurrentTryCatchWithInjection())
        throw new IOException();
}

... //normal try body

... //normal try body

} catch (IOException e) {  // Genetic Improvement
    ... //normal catch body
}
```
Limit cases

Stack of try-catch at runtime

// original
Catch (OutOfMemory) {
    Catch (NullPointerException) {
        Catch (IOException) {
            throw new NullPointerException()
        }
    }
}

// stretched
Catch (OutOfMemory) {
    Catch (NullPointerException) {
        Catch (Exception) {
            throw new NullPointerException()
        }
    }
}
Empirical Evaluation

- shindig-common
- shindig-java-gadgets
- jbehave core
- sonar plugin
- sonar core
- spojo core
- joda time
- commons-codec
- commons-lang

0 5 10 15 20 25 30 35

unstretchable
stretchable

Martin Monperrus, University of Lille & Inria, France
Take-away

“We inject exceptions during test suite execution to assess and improve resilience”

References:

A thing is **antifragile** if it becomes better with stressors, shocks, volatility, noise, mistakes, faults, attacks, failures . . .

(Antifragile, N. N. Taleb, 2013)
A software system is **antifragile** if it becomes better with failures, attacks, failures, misconfigurations, weird usages . . .

(Principles of Antifragile Software, M. Monperrus, 2014)
Short circuit testing is antifragile:
• Better with exception injection
• Does not try to anticipate or predict
hypothesis ← perturbation, measure
while (true) {
    perturb
    if (measure unacceptable) {
        report failure
    }
}
Antifragile software engineering

Core model:

- perturbation model
- perturbation controller
- perturbation cost
- perturbation gain

http://www.monperrus.net/martin/antifragile-software