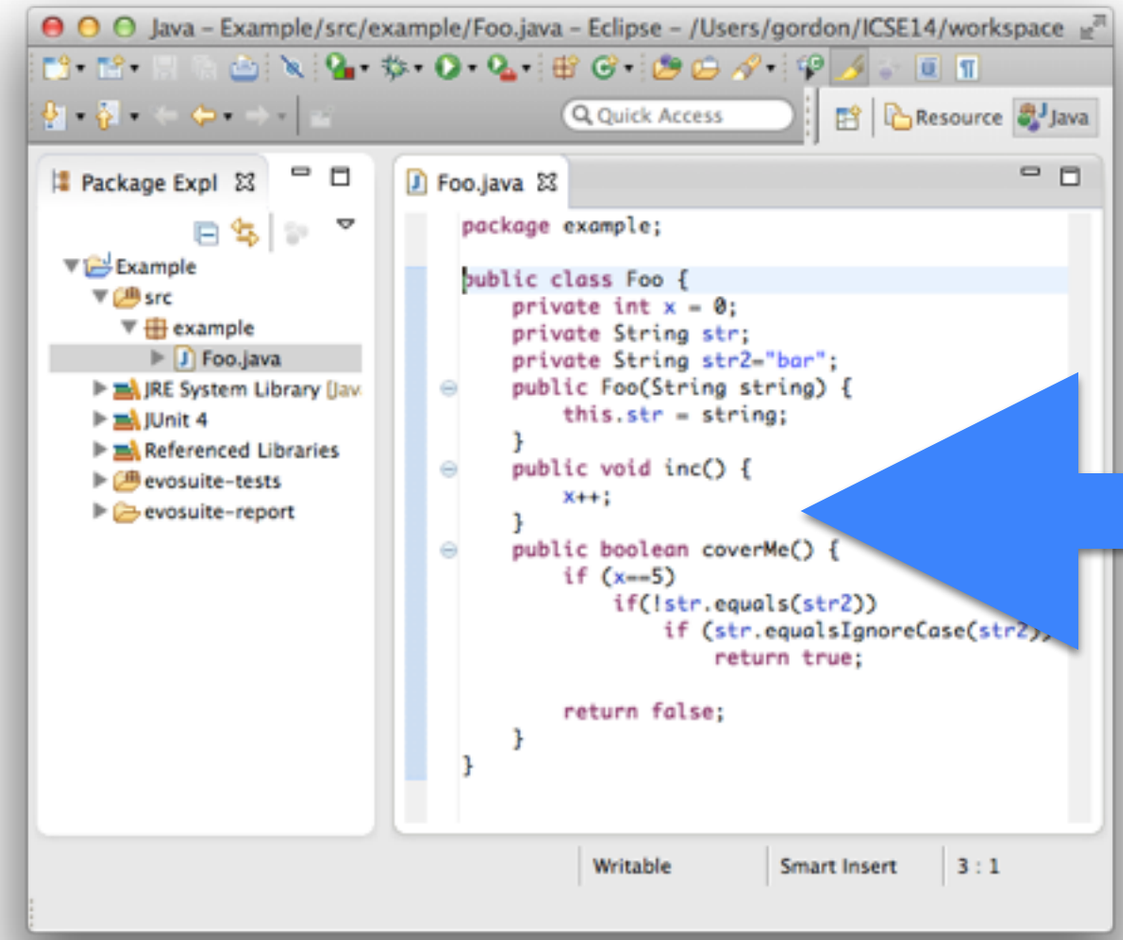
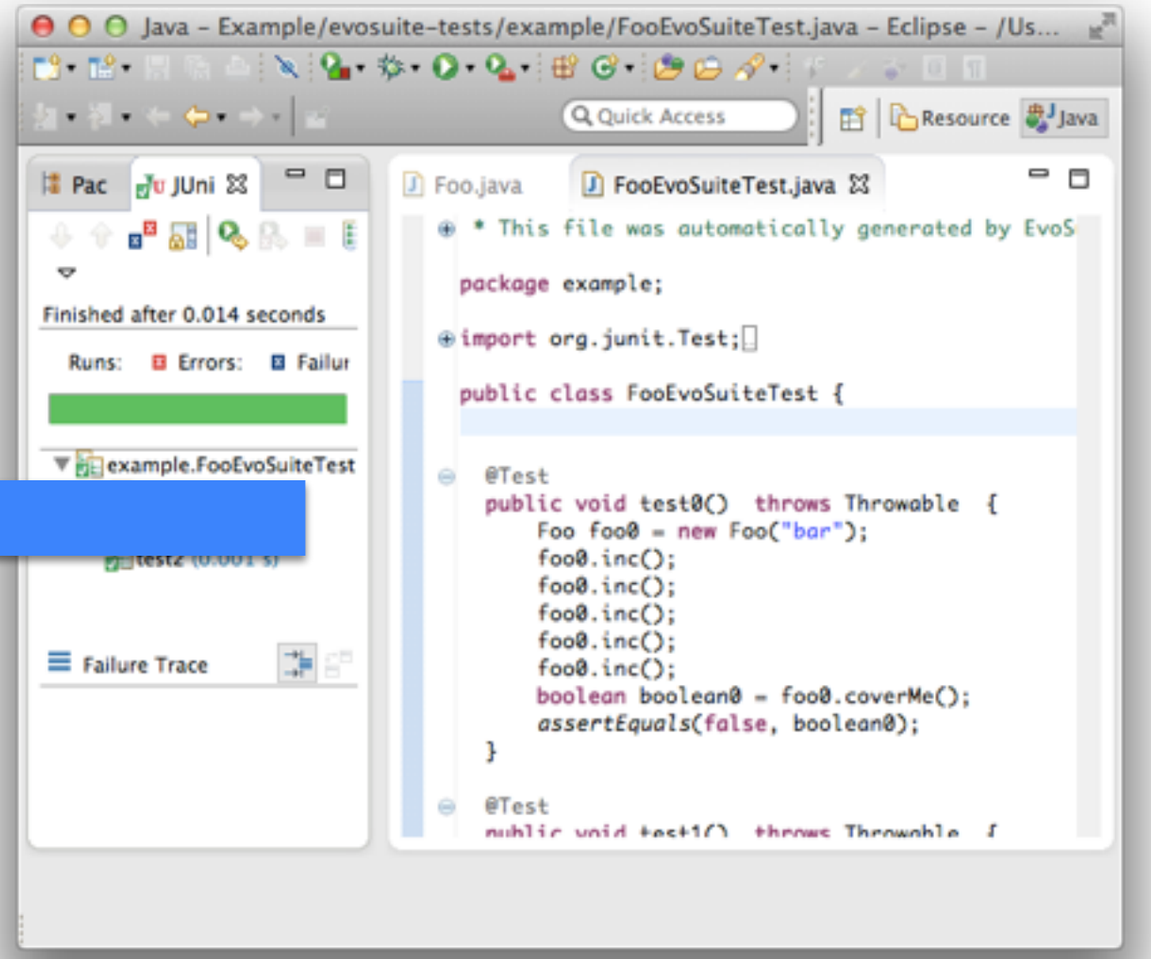
Search-Based Software Engineers Need Tools

Gordon Fraser, University of Sheffield

EVOSUITE

Contents

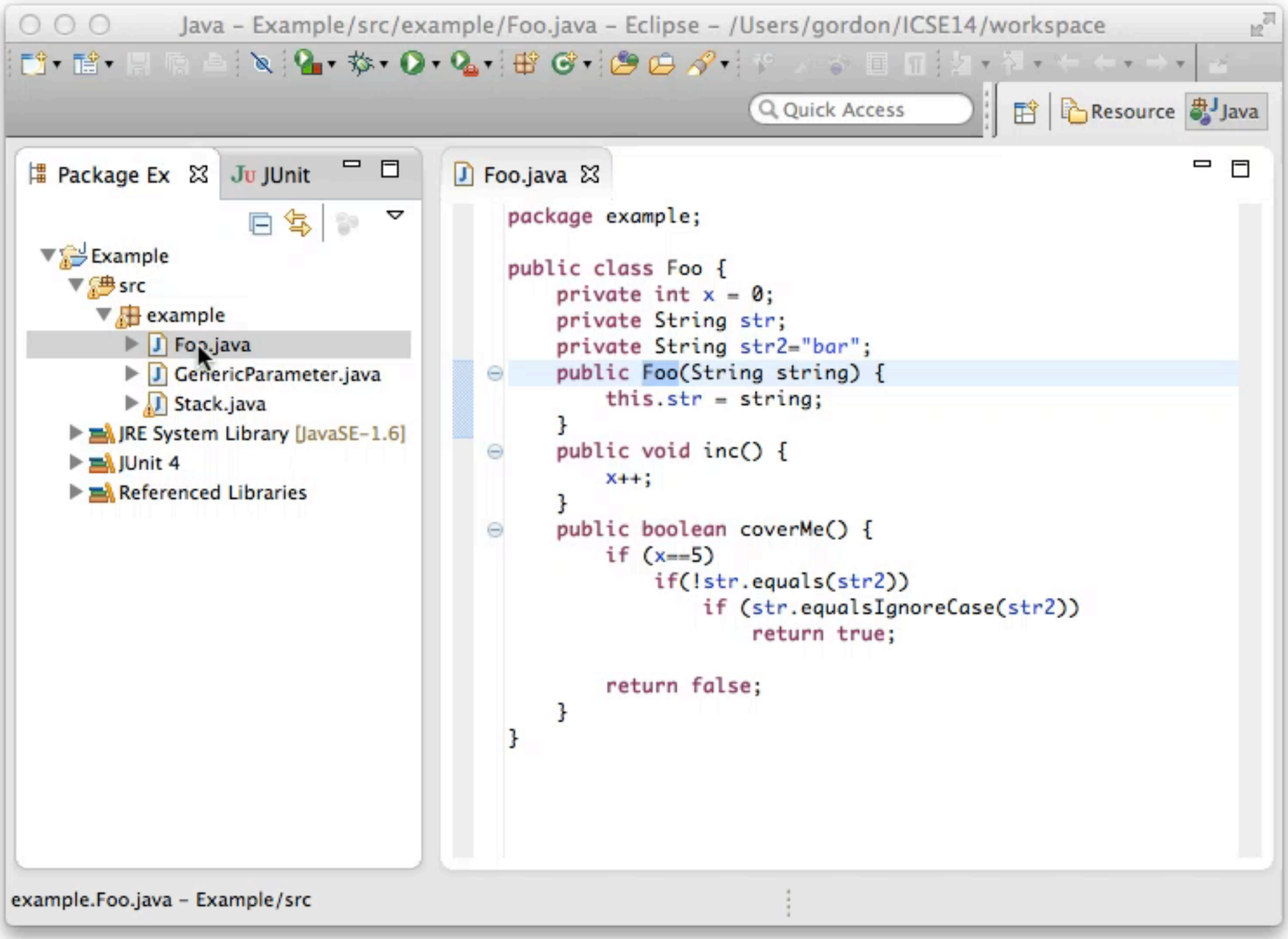


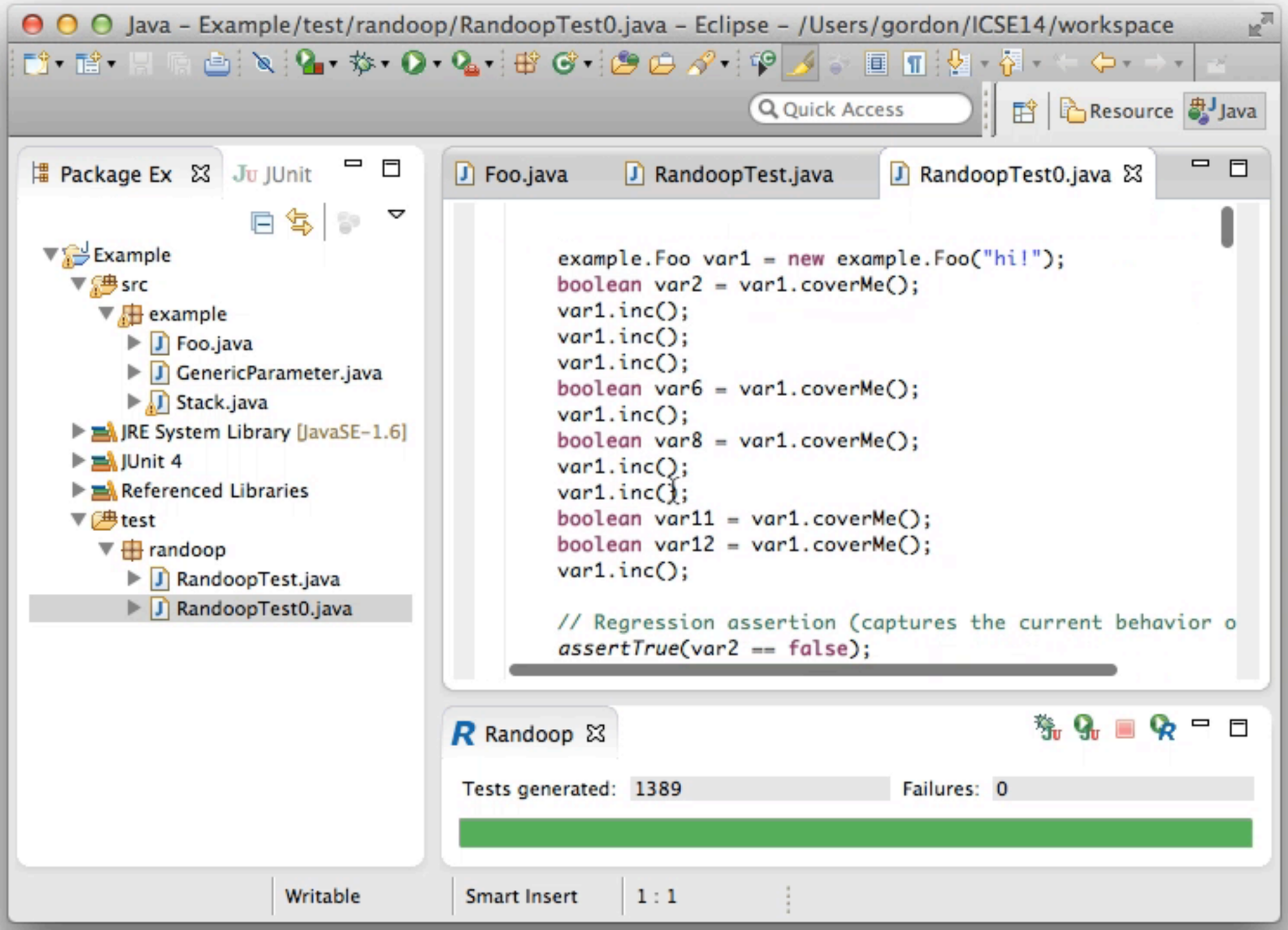


Source code

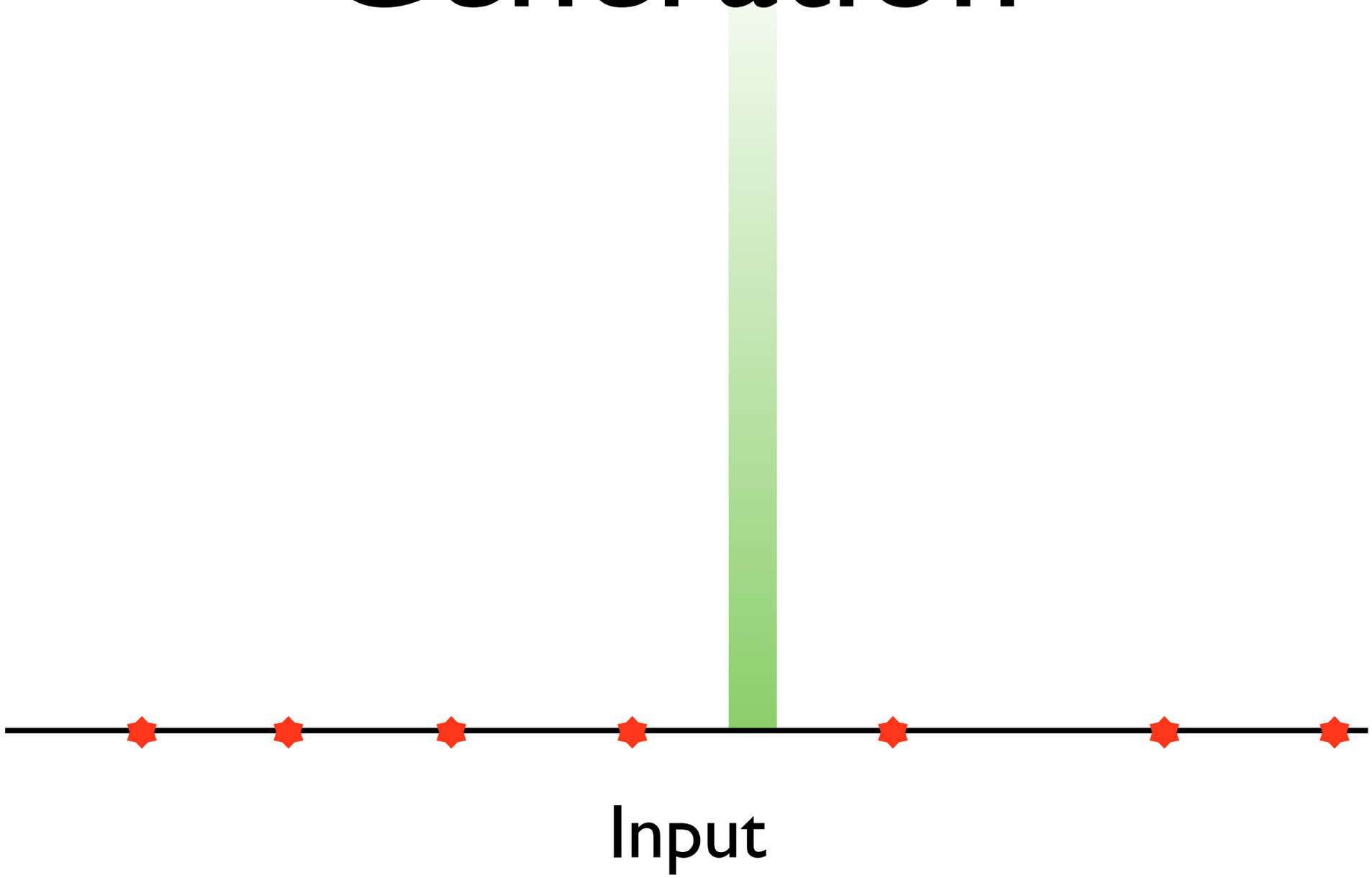
Tests

Automated test generation


Java – Example/src/example/Foo.java – Eclipse – /Users/gordon/ICSE14/workspace
Quick Access
Resource
Java
Package Ex
JUnit
Example
src
example
Foo.java
GenericParameter.java
Stack.java
JRE System Library [JavaSE-1.6]
JUnit 4
Referenced Libraries
Foo.java
package example;
public class Foo {
private int x = 0;
private String str;
private String str2="bar";
public Foo(String string) {
this.str = string;
}
public void inc() {
x++;
}
public boolean coverMe() {
if (x==5)
if(!str.equals(str2))
if (str.equalsIgnoreCase(str2))
return true;
return false;
}
}
example.Foo.java – Example/src


Java – Example/test/randoop/RandoopTest0.java – Eclipse – /Users/gordon/ICSE14/workspace
Quick Access
Resource
Java
Package Ex
JUnit
Example
src
example
Foo.java
GenericParameter.java
Stack.java
JRE System Library [JavaSE-1.6]
JUnit 4
Referenced Libraries
test
randoop
RandoopTest.java
RandoopTest0.java
Foo.java
RandoopTest.java
RandoopTest0.java
example.Foo var1 = new example.Foo("hi!");
boolean var2 = var1.coverMe();
var1.inc();
var1.inc();
var1.inc();
boolean var6 = var1.coverMe();
var1.inc();
boolean var8 = var1.coverMe();
var1.inc();
var1.inc();
boolean var11 = var1.coverMe();
boolean var12 = var1.coverMe();
var1.inc();
// Regression assertion (captures the current behavior o
assertTrue(var2 == false);
Randoop
Tests generated: 1389
Failures: 0
Writable
Smart Insert
1 : 1

Random Test Data Generation



Pex for fun – from Microsc ×

pexforfun.com

My Duels▼ | Settings▼ | Sign In

Curious? Learn More!

Pex for fun

Random Puzzle | Learn | APCS | New | 1,525,714 clicked 'Ask Pex!' | C# | Visual Basic | F#

The code is a puzzle. Do you understand what the code does? Click **Ask Pex!** to find out.

```
using System;

public class Foo {
        private int x = 0;
        private String str;
        private String str2="bar";
        public Foo(String str){
                this.str = str;
        }
        public void inc() {
                x++;
        }
        public bool coverMe() {
                if (x==5)
                        if(!str.Equals(str2))
                                if (str.Equals(str2
```

Ask Pex!

Permalink

Click Here!

Community Live Feed Publications About

Microsoft Research RiSE

for Windows Phone

Pex for fun – from Microso

pexforfun.com

Curious? Learn More!

Pex for fun

Random Puzzle | Learn | APCS | New | 1,525,714 clicked 'Ask Pex!' | C# | Visual Basic | F#

The code is a puzzle. Do you understand what the code does? Click **Ask Pex!** to find out.

```
                this.str = str;
        }
        public void inc() {
                x++;
        }
        public bool coverMe() {
                if (x==5)
                        if(!str.Equals(str2))
                                if (str.Equals(str2,
                                    StringComparison.OrdinalIgnoreCase))
                                        return true;

                return false;
        }
}
```

Ask Pex!

Click Here!

Permalink

Community | Live Feed | Publications | About

Microsoft Research

RiSE

for Windows Phone

Puzzle history (show Permalinks): initial;

Pex for fun – from Microso

pexforfun.com

Pex for fun

Random Puzzle | Learn | APCS | New | 1,525,714 clicked 'Ask Pex!' | C# | Visual Basic | F#

The code is a puzzle. Do you understand what the code does? Click **Ask Pex!** to find out.

```
        public bool coverMe() {
                if (x==5)
                        if(!str.Equals(str2))
                                if (str.Equals(str2,
                                    StringComparison.OrdinalIgnoreCase))
                                        return true;

                return false;
        }

        public static bool Puzzle(Foo foo) {
          return foo.coverMe();
        }

}
```

Ask Pex! *Done. 2 interesting inputs found.* How does Pex work? Permalink

	foo	result	Output/Exception	Error Message
✗	null		NullReferenceException	Object reference not set to an instance of an object.
✓	new Foo{}	false		

Coding Duel Name: Turn This Puzzle Into A Coding Duel Help

Pex and Moles

Generating vs Checking

Conventional Software Testing Research

Write a method to construct test cases

Search-Based Testing

Write a method
to determine how good a test case is

Generating vs Checking

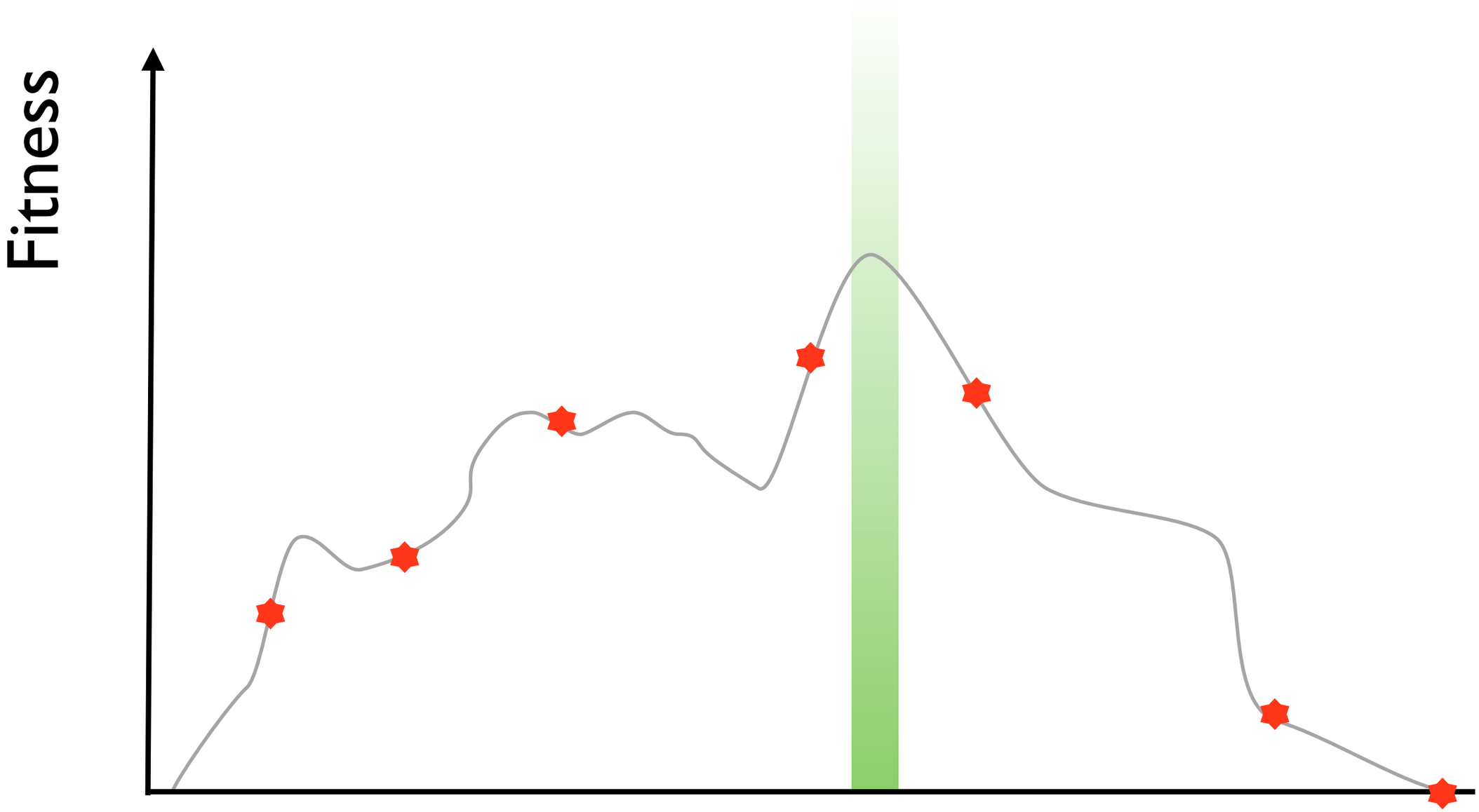
Conventional Software Testing Research

Write a method to construct test cases

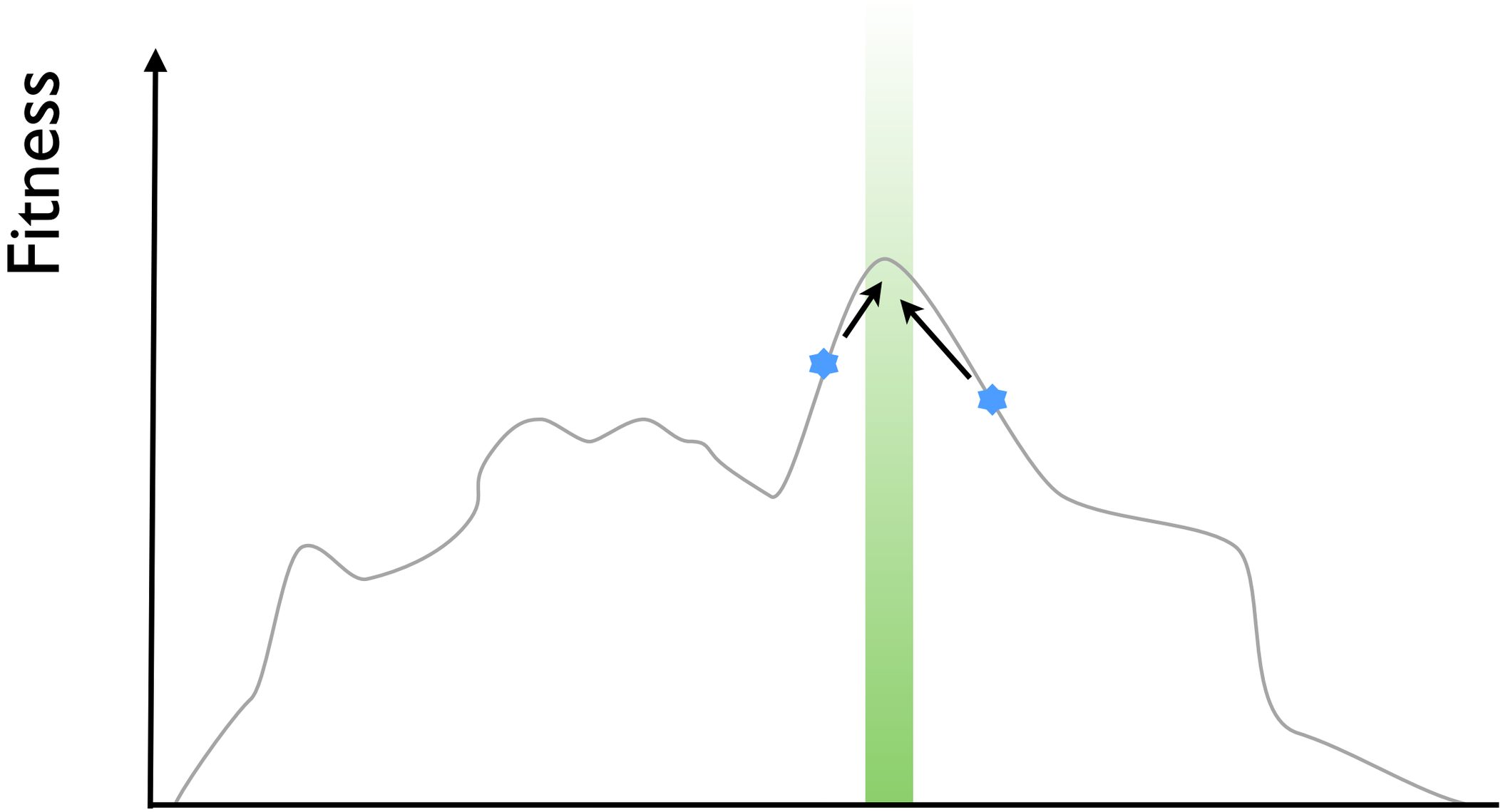
Search-Based Testing

Write a fitness function
to determine how good a test case is

Fitness-guided search

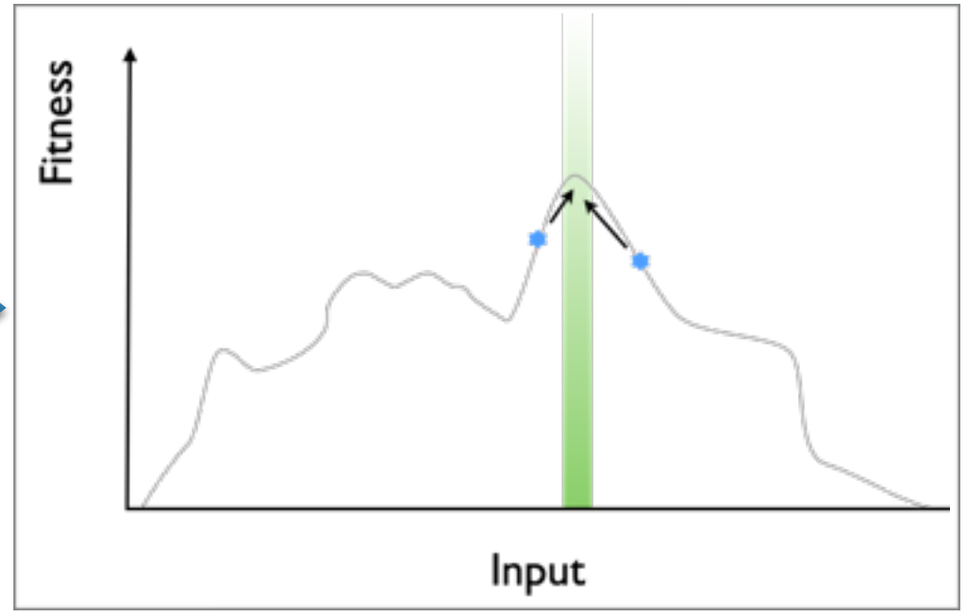


Fitness-guided search



Components of an SBST Tool

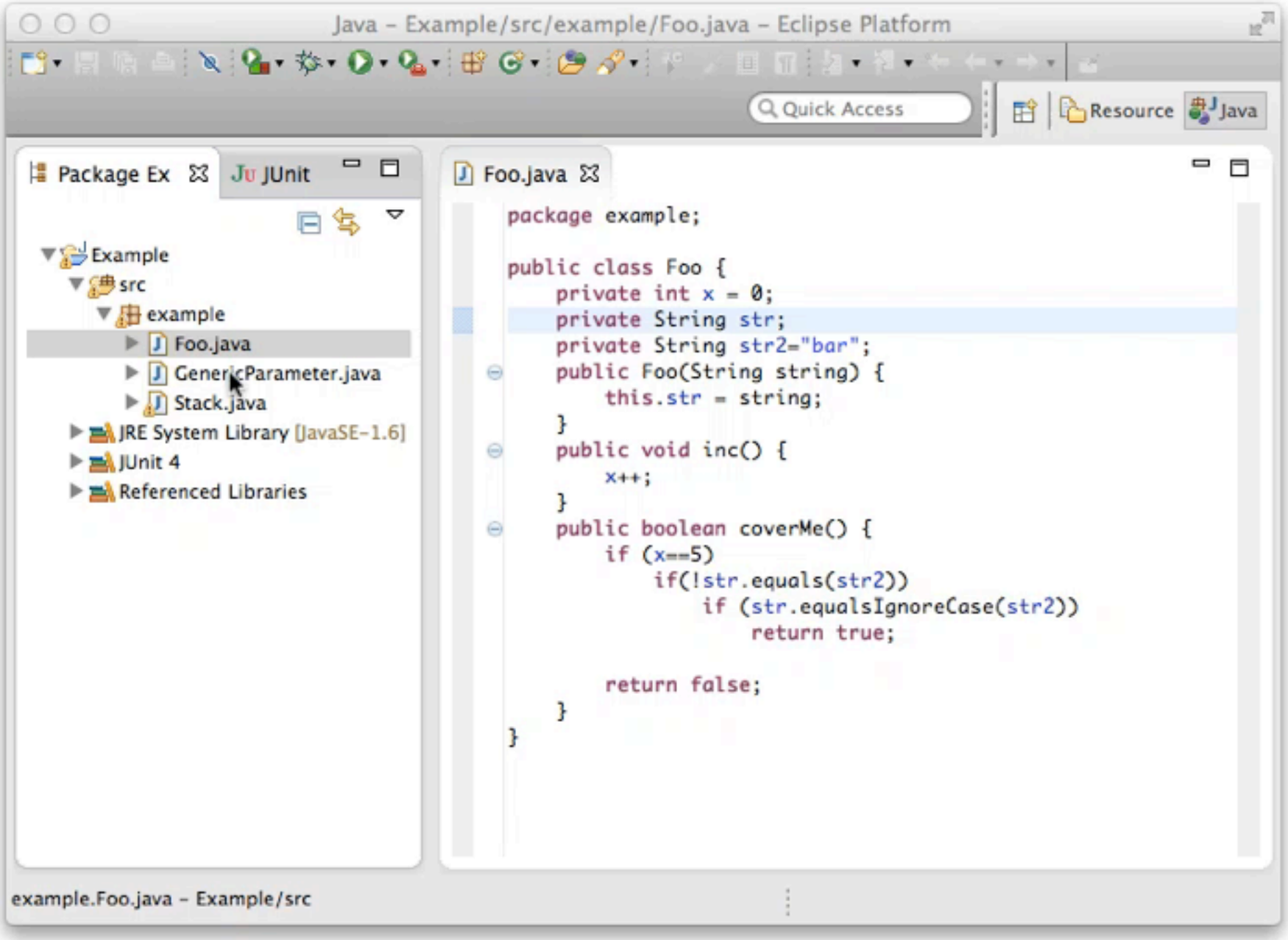
```
def testMe(x, y):
    if x == 2 * (y + 1):
        return True
    else:
        return False
```



- Search Algorithm
- Representation
- Fitness Function

Components of an SBST Tool

Search Algorithm	Meta-heuristic algorithm
Representation	Encoding of the problem solution
Search Operators	Modifications of encoded solutions
Fitness Function	Measure how good a candidate solution is
Test Execution	Execute tests
Instrumentation	Collect data/traces for fitness calculation during execution


Java – Example/src/example/Foo.java – Eclipse Platform
Quick Access
Resource
Java
Package Ex
JUnit
Example
src
example
Foo.java
GenericParameter.java
Stack.java
JRE System Library [JavaSE-1.6]
JUnit 4
Referenced Libraries
Foo.java
package example;
public class Foo {
private int x = 0;
private String str;
private String str2="bar";
public Foo(String string) {
this.str = string;
}
public void inc() {
x++;
}
public boolean coverMe() {
if (x==5)
if(!str.equals(str2))
if (str.equalsIgnoreCase(str2))
return true;
return false;
}
}
example.Foo.java - Example/src

Coverage Progress

83.64 %

Address Book

New contact

New category

First name | Last name | E-mail | Phone | Mobile

All

Create a category

Category name:

eO"l" already exists

Abbrechen | OK

First name

Last name

Phone

Mobile

Notes

Second e-mail

URL

Apply

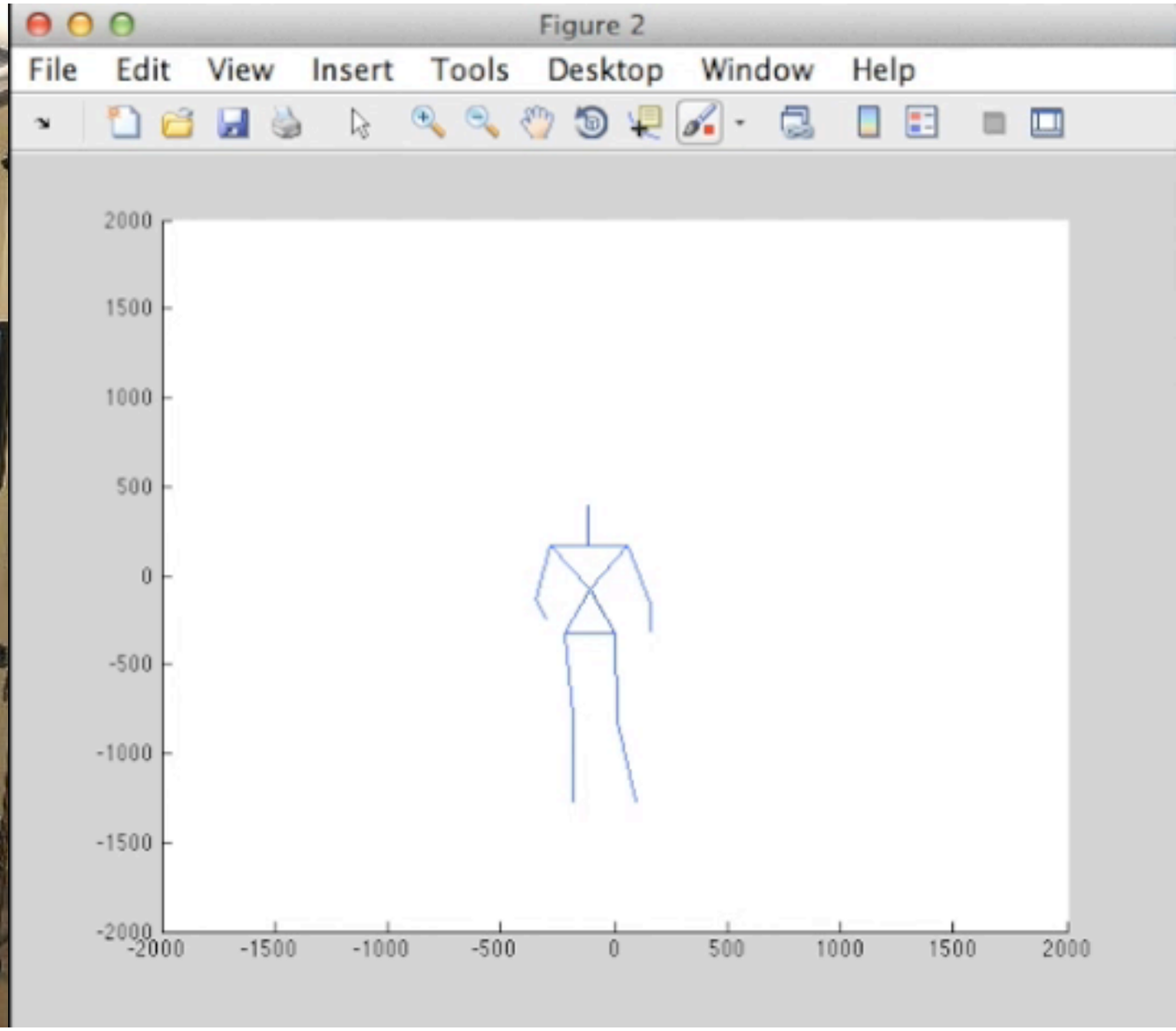


Figure 2
File Edit View Insert Tools Desktop Window Help
2000
1500
1000
500
0
-500
-1000
-1500
-2000
-2000 -1500 -1000 -500 0 500 1000 1500 2000

Contents

```
def testMe(x, y):
    if x == 2 * (y + 1):
        return True
    else:
        return False
```

Components of an SBST Tool

Search Algorithm

Hill-climbing

Representation

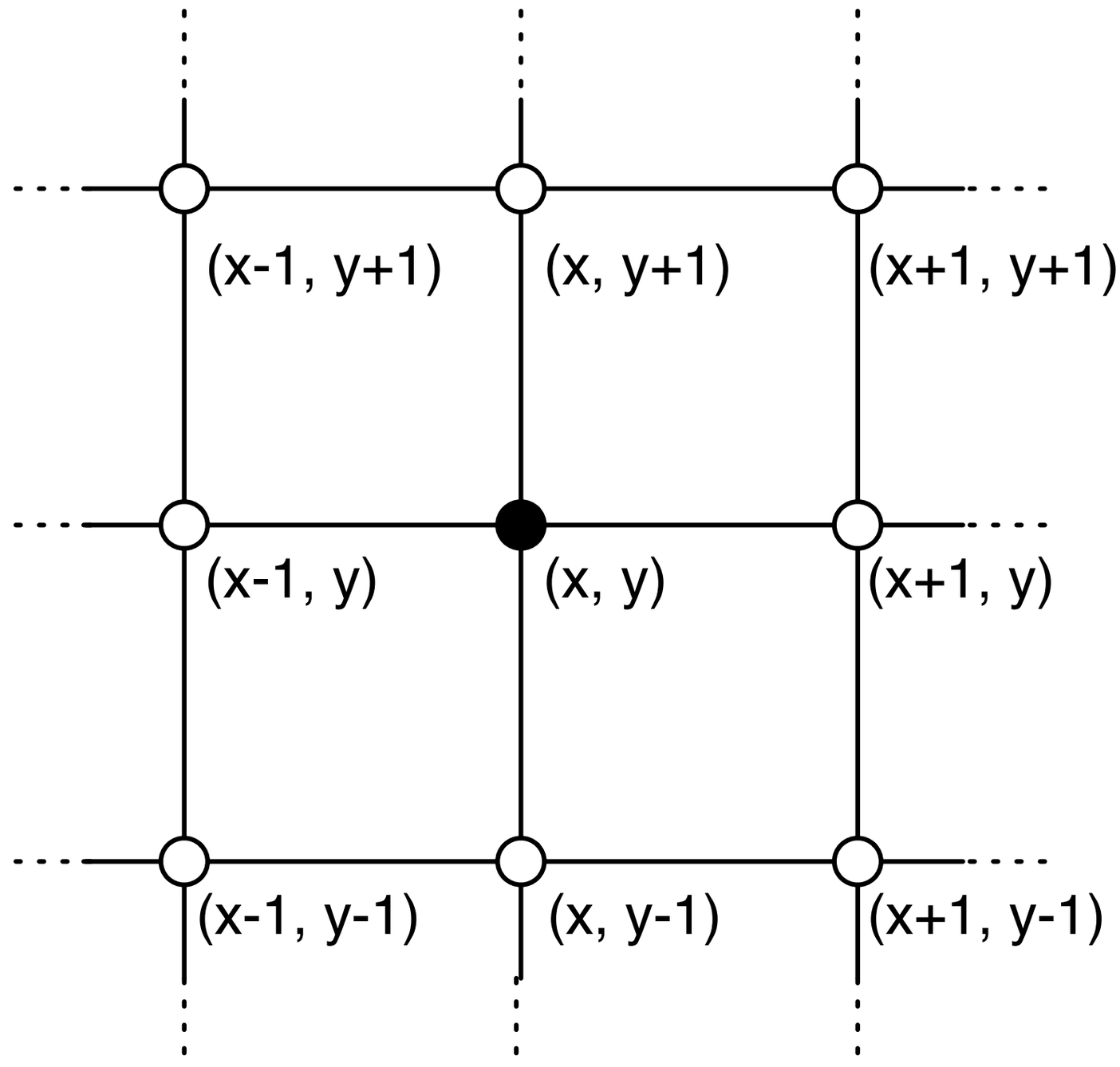
Search Operators

Fitness Function

Test Execution

Instrumentation

```
def testMe(x, y):
    if x == 2 * (y + 1):
        return True
    else:
        return False
```



Components of an SBST Tool

Search Algorithm — Hill-climbing

Representation — Tuple (x, y)

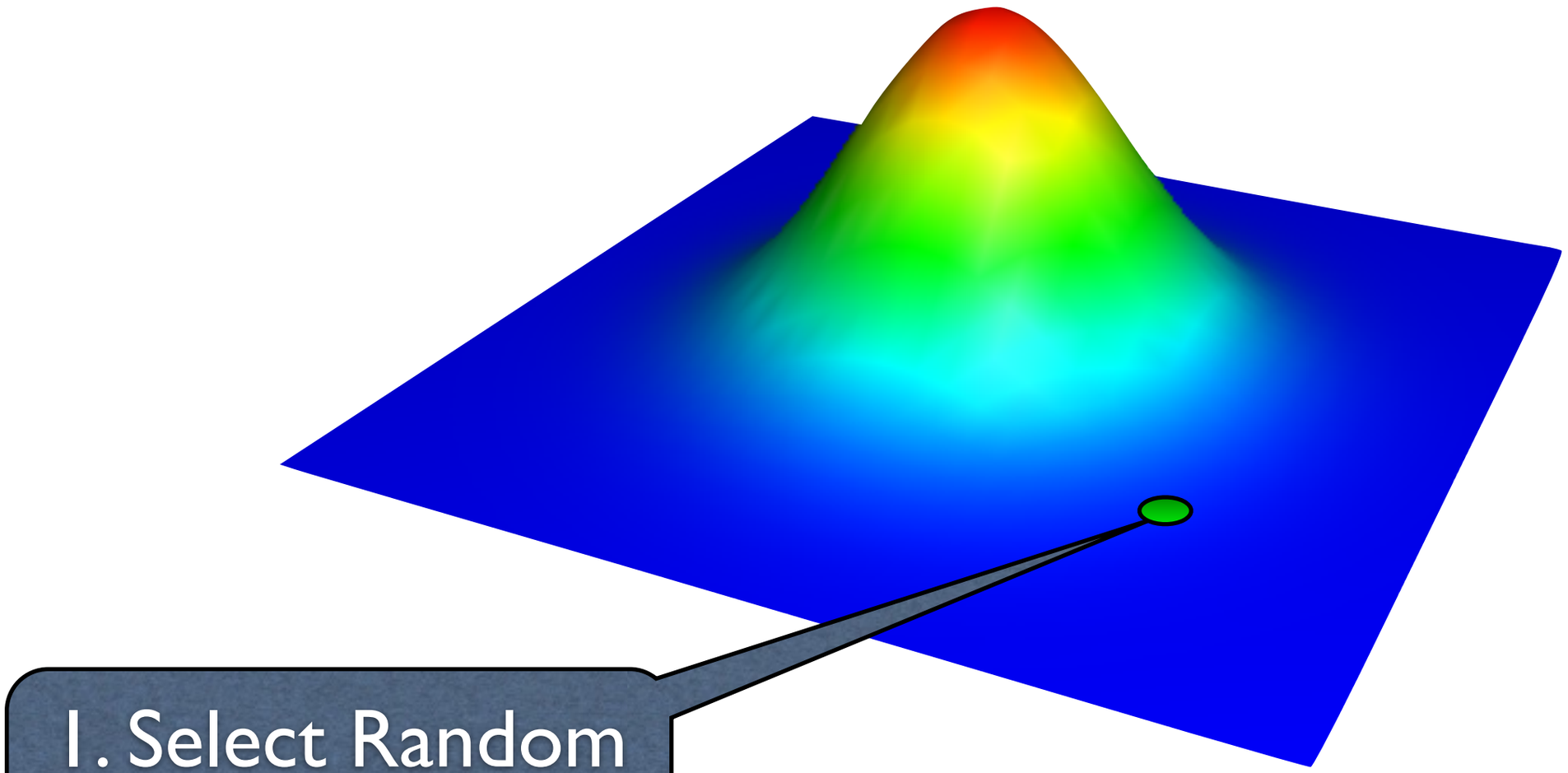
Search Operators — Neighbourhood of (x, y)

Fitness Function

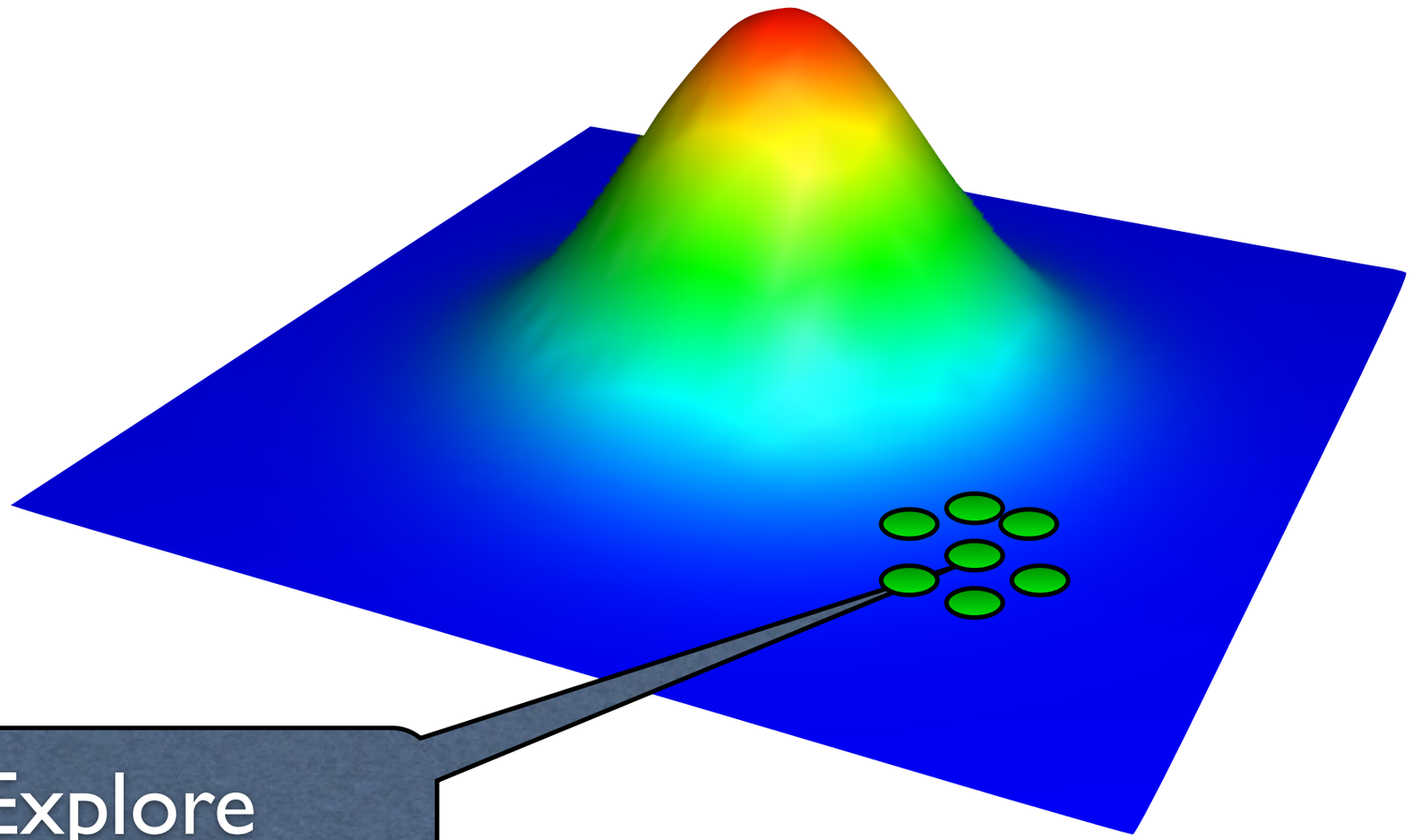
Test Execution

Instrumentation

Hill Climbing

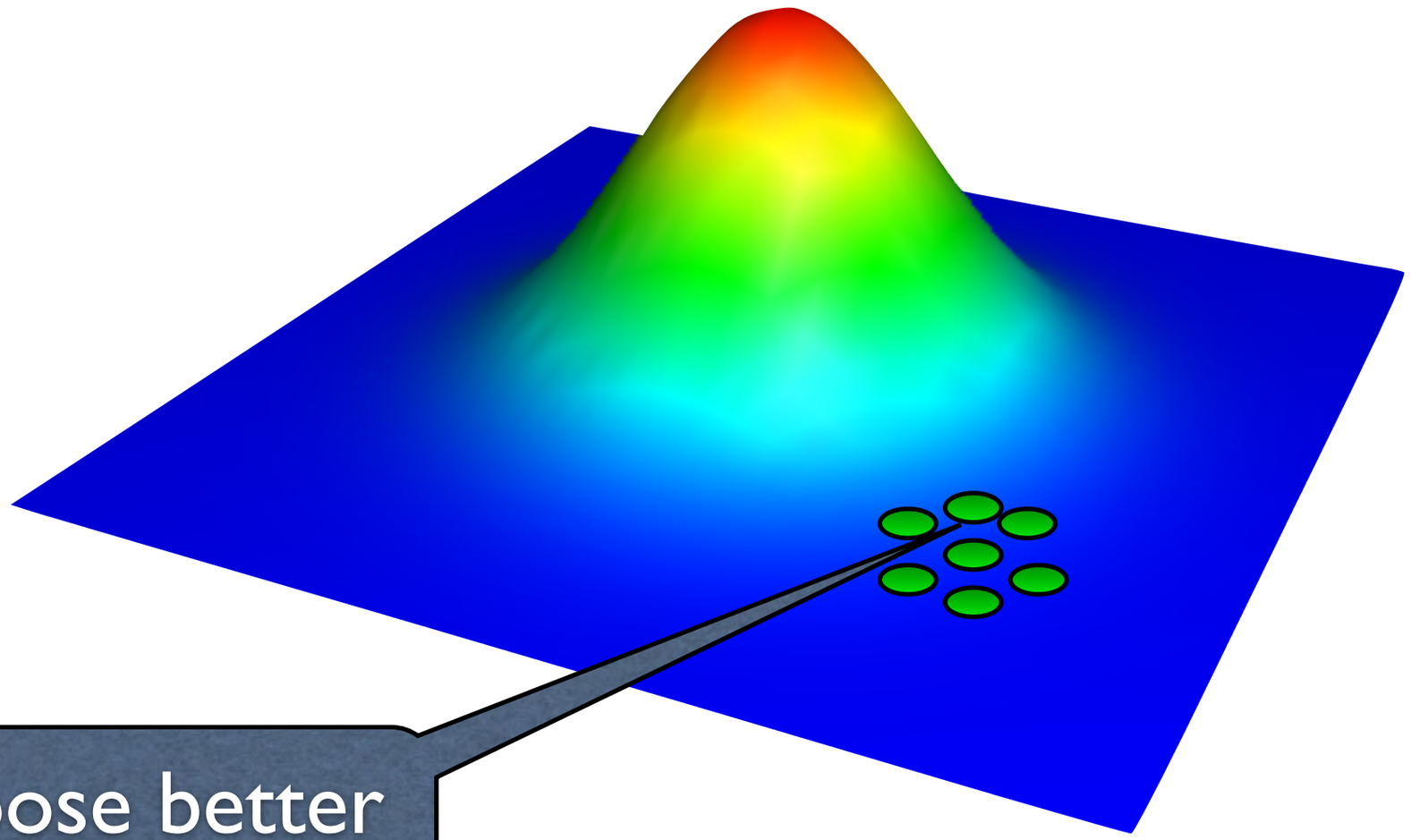


Hill Climbing



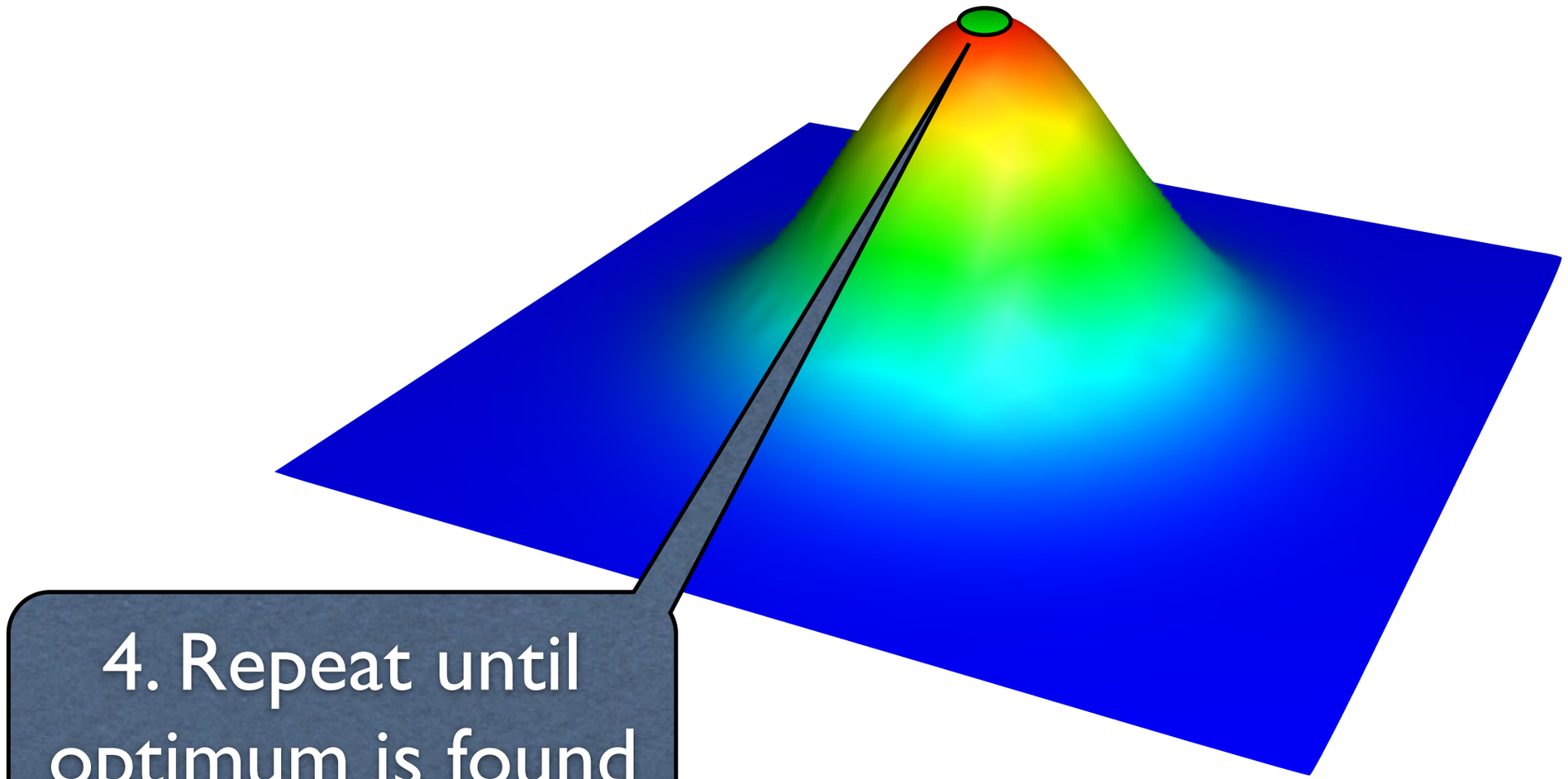
2. Explore Neighbourhood

Hill Climbing



3. Choose better neighbour

Hill Climbing



Components of an SBST Tool

Search Algorithm — Hill-climbing

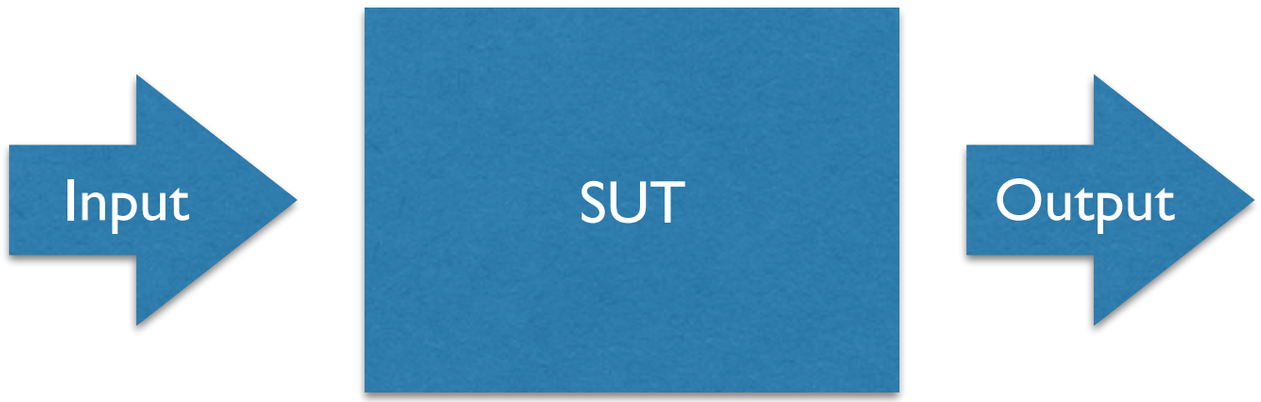
Representation — Tuple (x, y)

Search Operators — Neighbourhood of (x, y)

Fitness Function

Test Execution

Instrumentation


Input
SUT
Output

Test Data

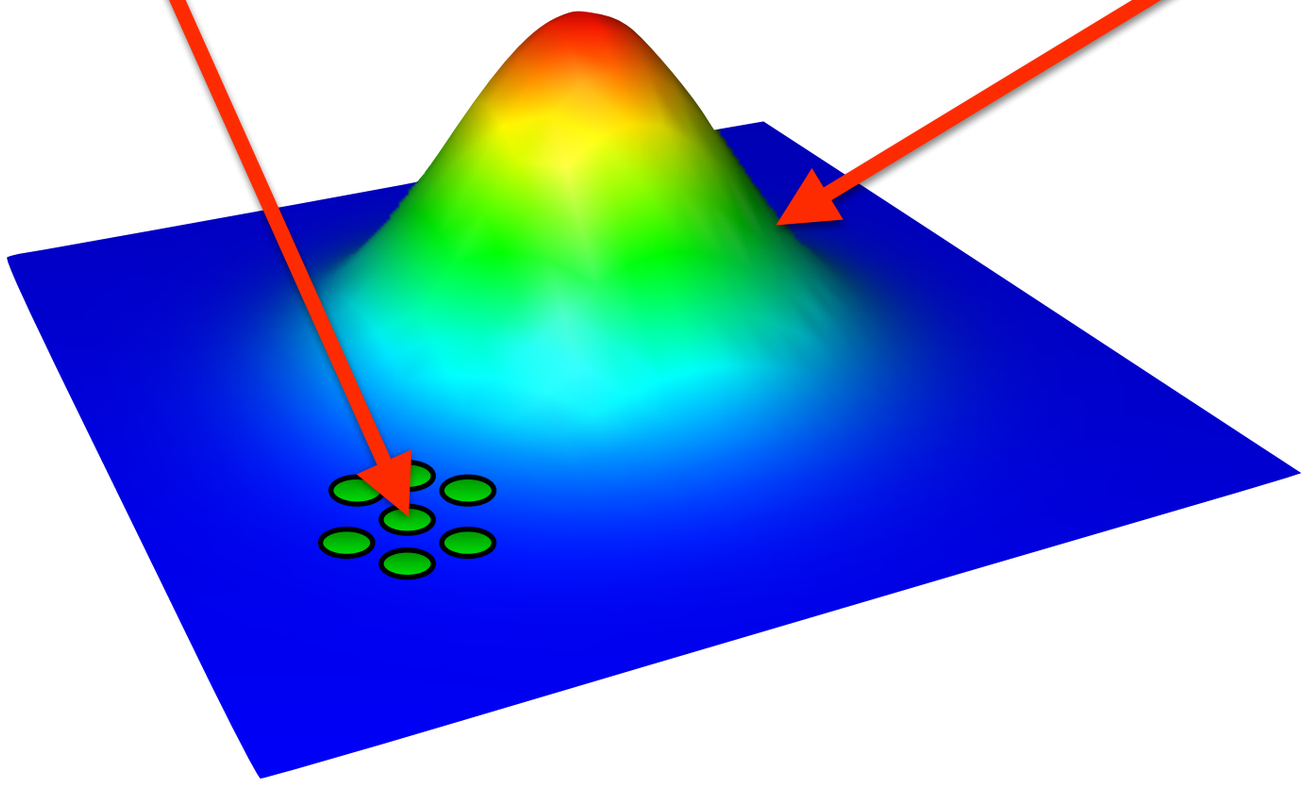
Input

Instrumented SUT

Output

Trace

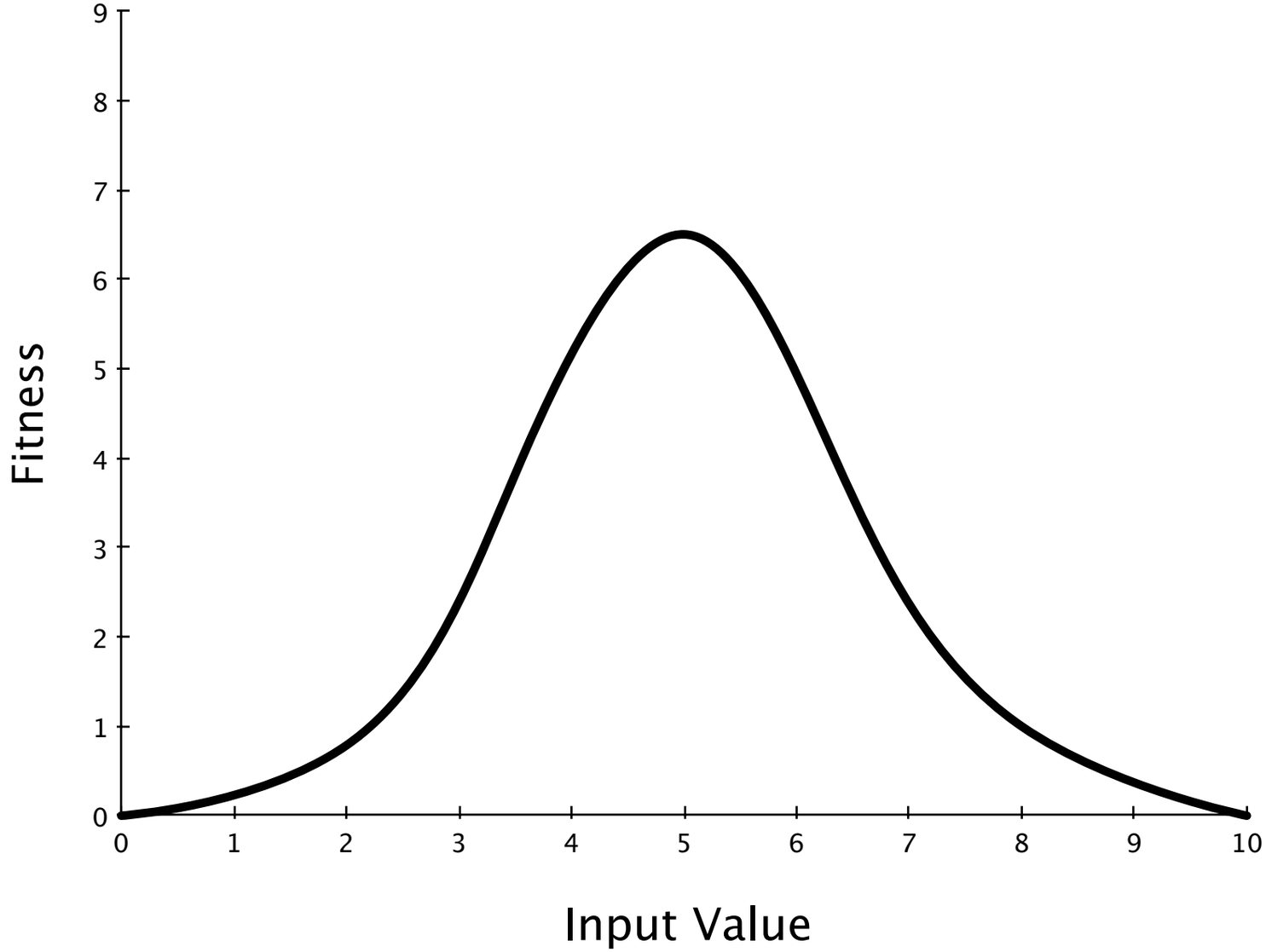
Fitness

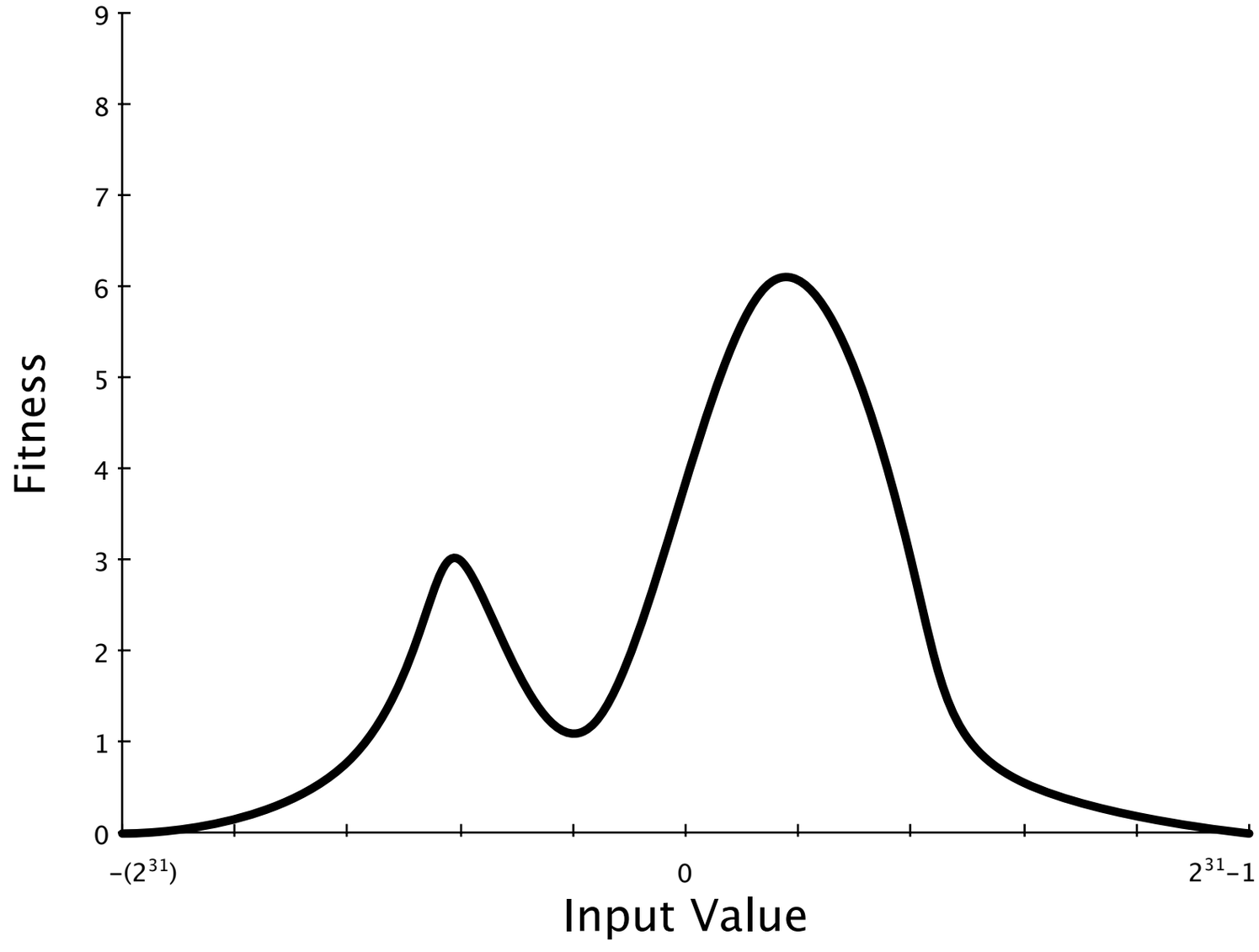


```
def testMe(x, y):
    if x == 2 * (y + 1):
        return True
    else:
        return False
```

Components of an SBST Tool

Search Algorithm	Hill-climbing
Representation	Tuple (x, y)
Search Operators	Neighbourhood of (x, y)
Fitness Function	Branch distance
Test Execution	Call method
Instrumentation	Global variable


Fitness
Input Value
0
1
2
3
4
5
6
7
8
9
10


Fitness
9
8
7
6
5
4
3
2
1
0
-(2^31)
0
2^31-1
Input Value

```
def testMe(x, y):
    if x == 2 * y and y > 1:
        return True
    else:
        return False
```

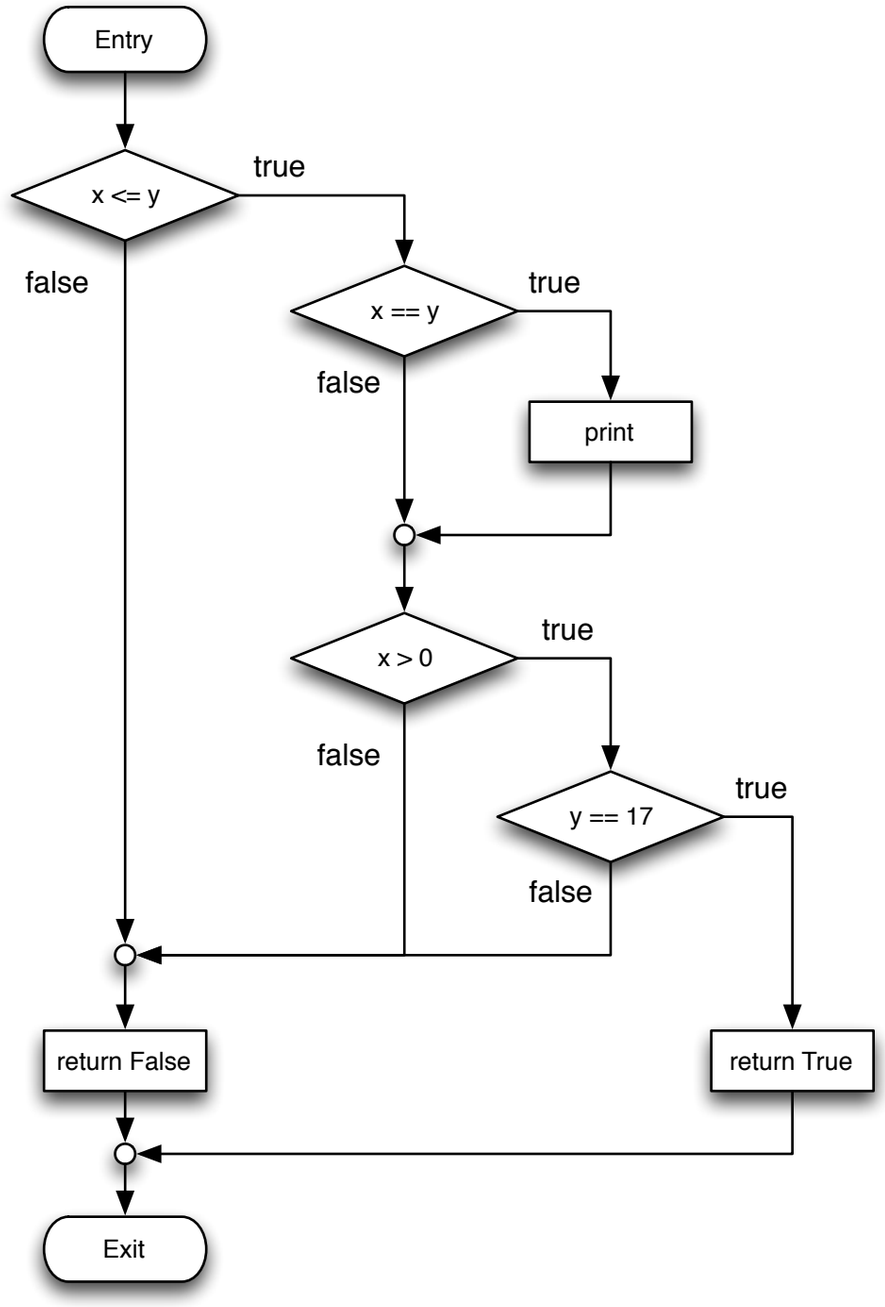
Branch Distance

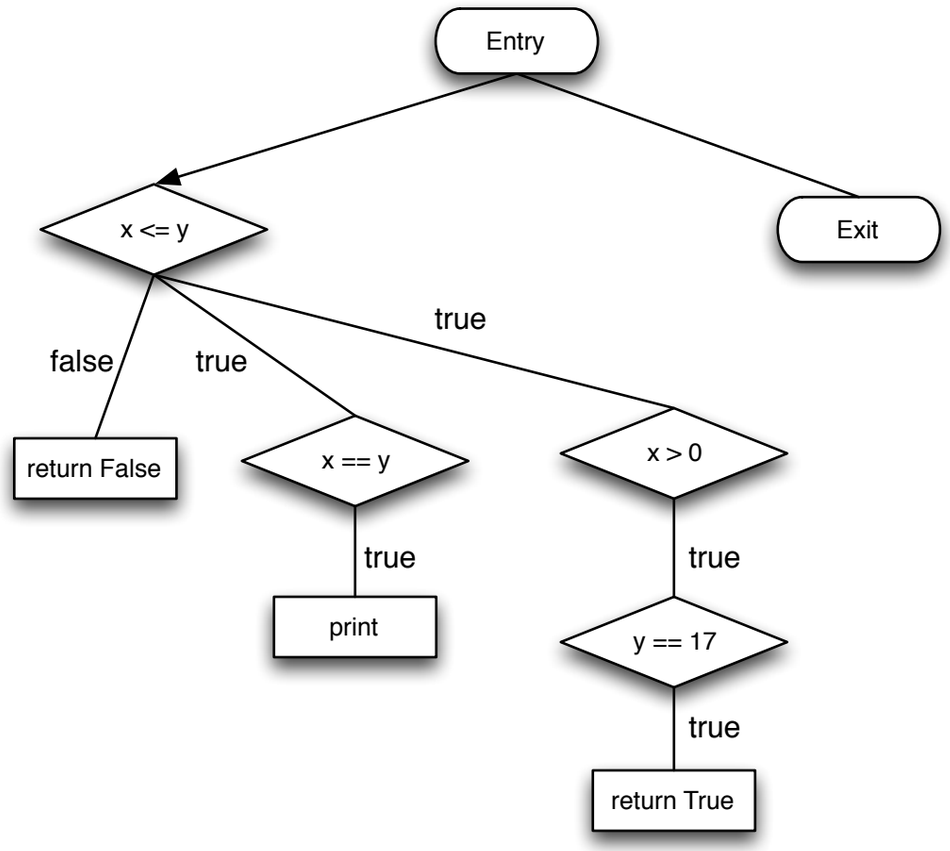
Expression	Distance True	Distance False
x == y	|x - y|	1
x != y	1	|x - y|
x > y	y - x + 1	x - y
x >= y	y - x	x - y + 1
x < y	x - y+ 1	x - y
x <= y	x - y	x - y + 1

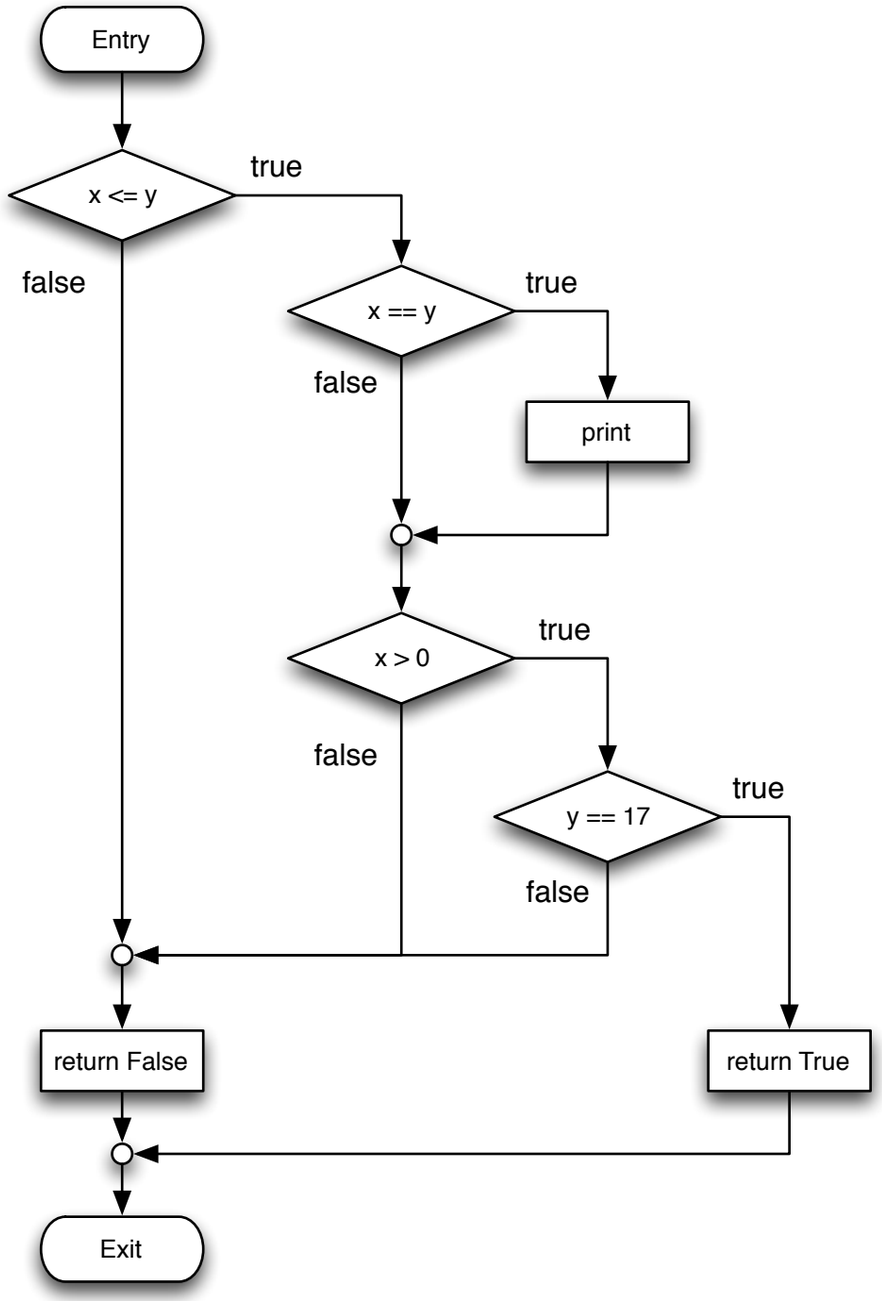
```
def testMe(x, y):
    if x == 2 * y and y > 1:
        return True
    else:
        return False
```

```
def testMe(x, y):
    if x <= y:
        if x == y:
            print("Some output")
        if x > 0:
            if y == 17:
                # Target Branch
                return True
return False
```

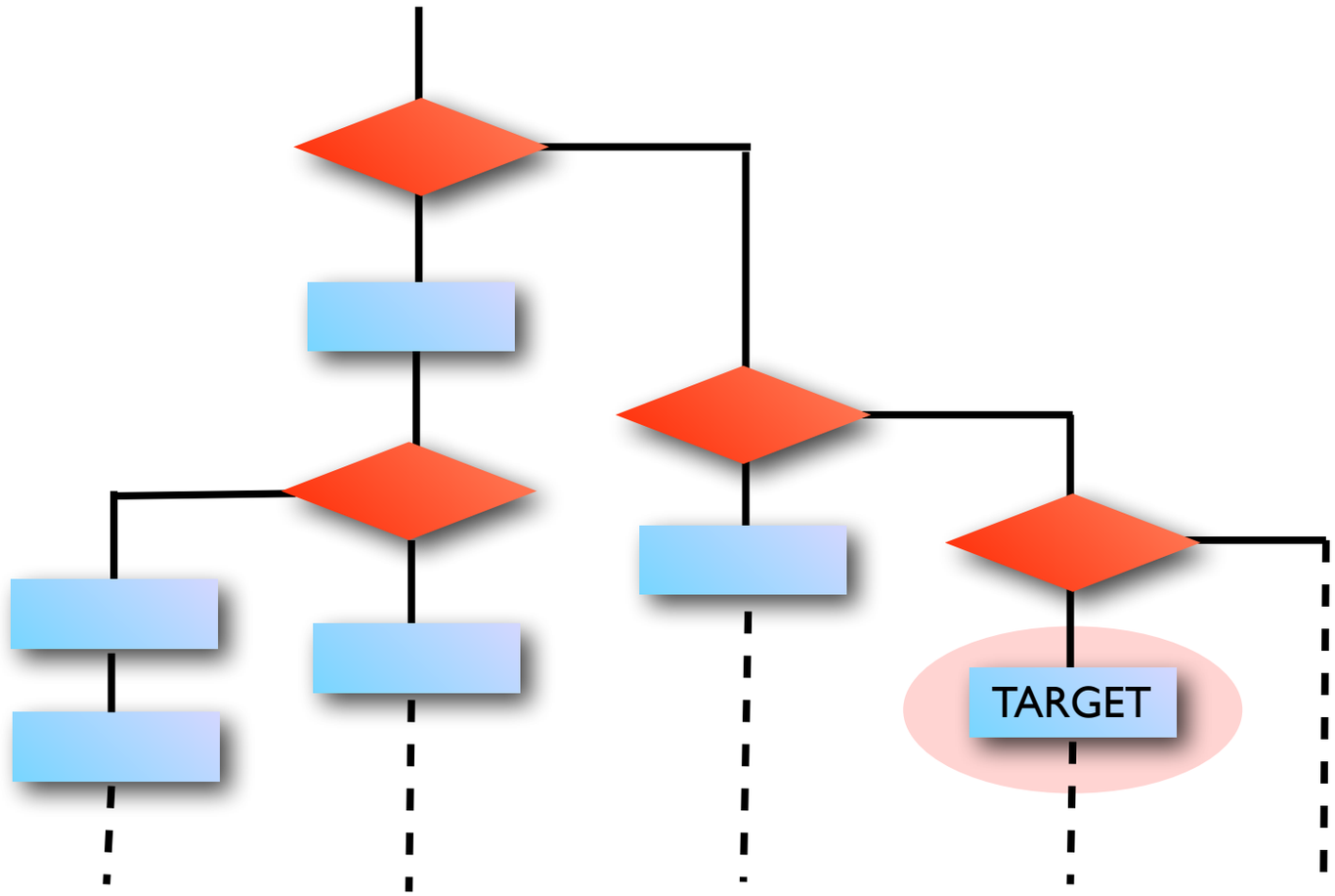
```
def testMe(x, y):
    if x <= y:
        if x == y:
            print("Some output")
    if x > 0:
        if y == 17:
            # Target Branch
            return True
return False
```



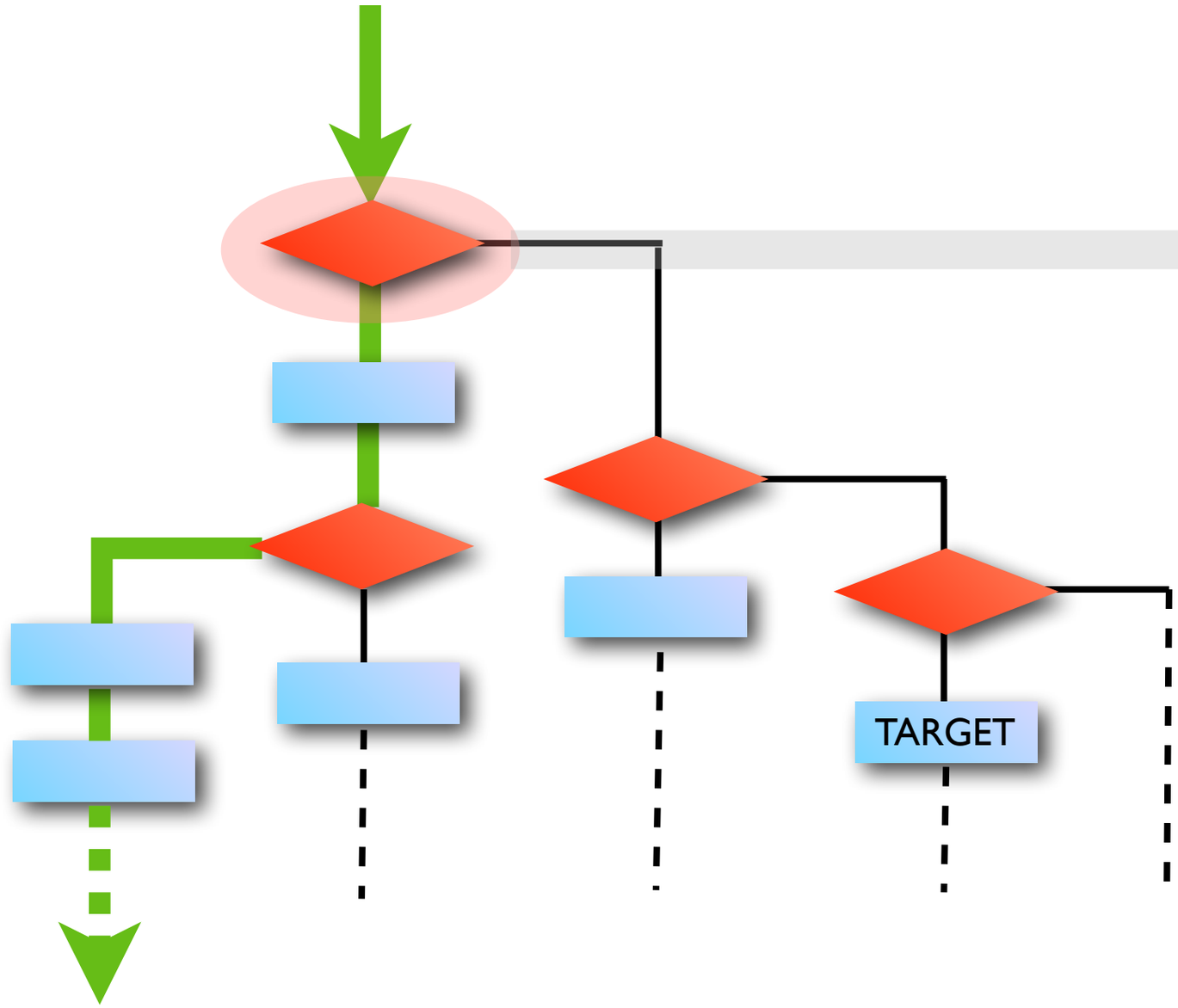

Entry
Exit
x <= y
false
true
true
return False
x == y
true
print
x > 0
true
y == 17
true
return True


Entry
x <= y
true
false
x == y
true
false
print
true
x > 0
true
false
y == 17
true
false
return False
return True
Exit

Covering a structure

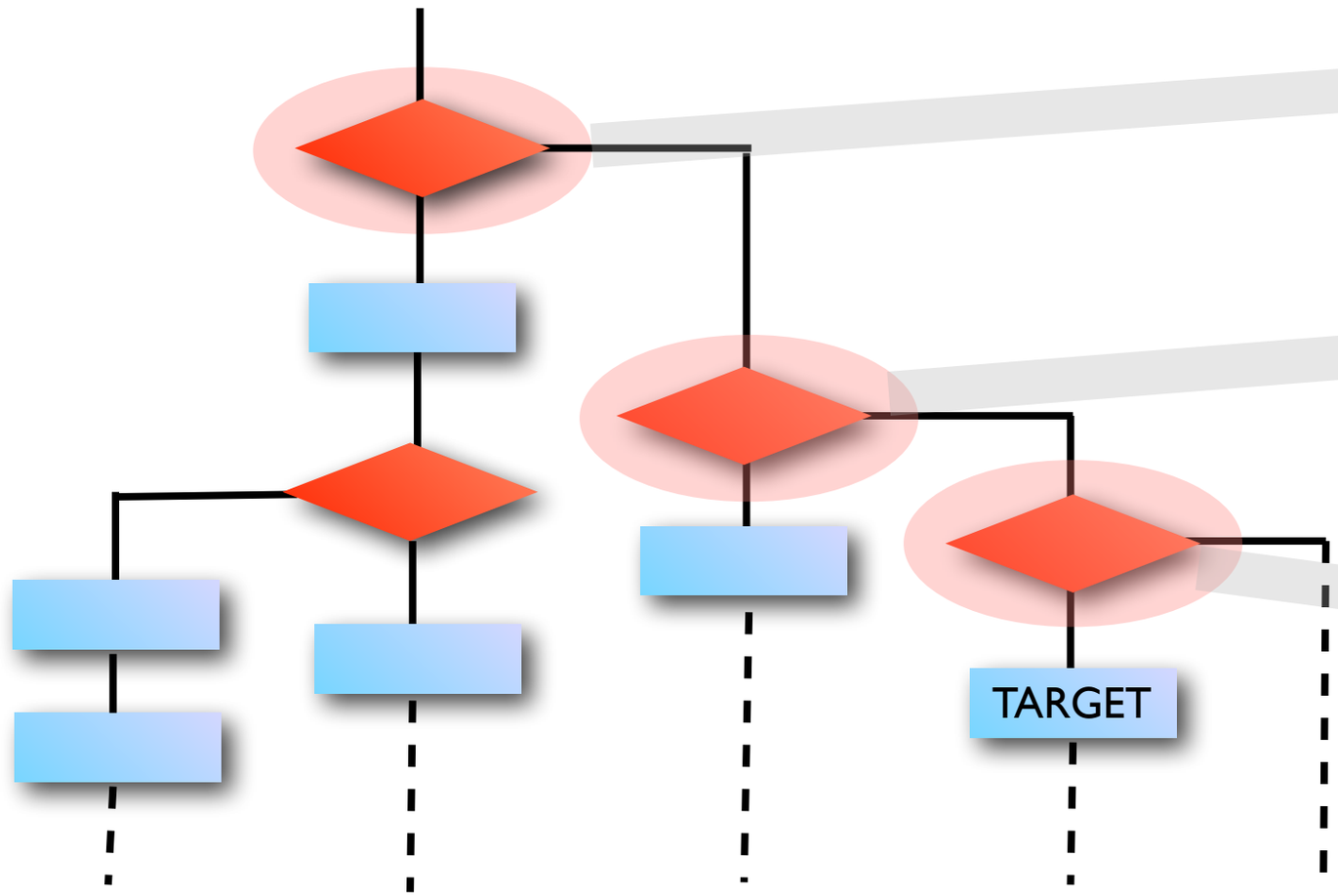


Fitness evaluation



The test data executes the ‘wrong’ path

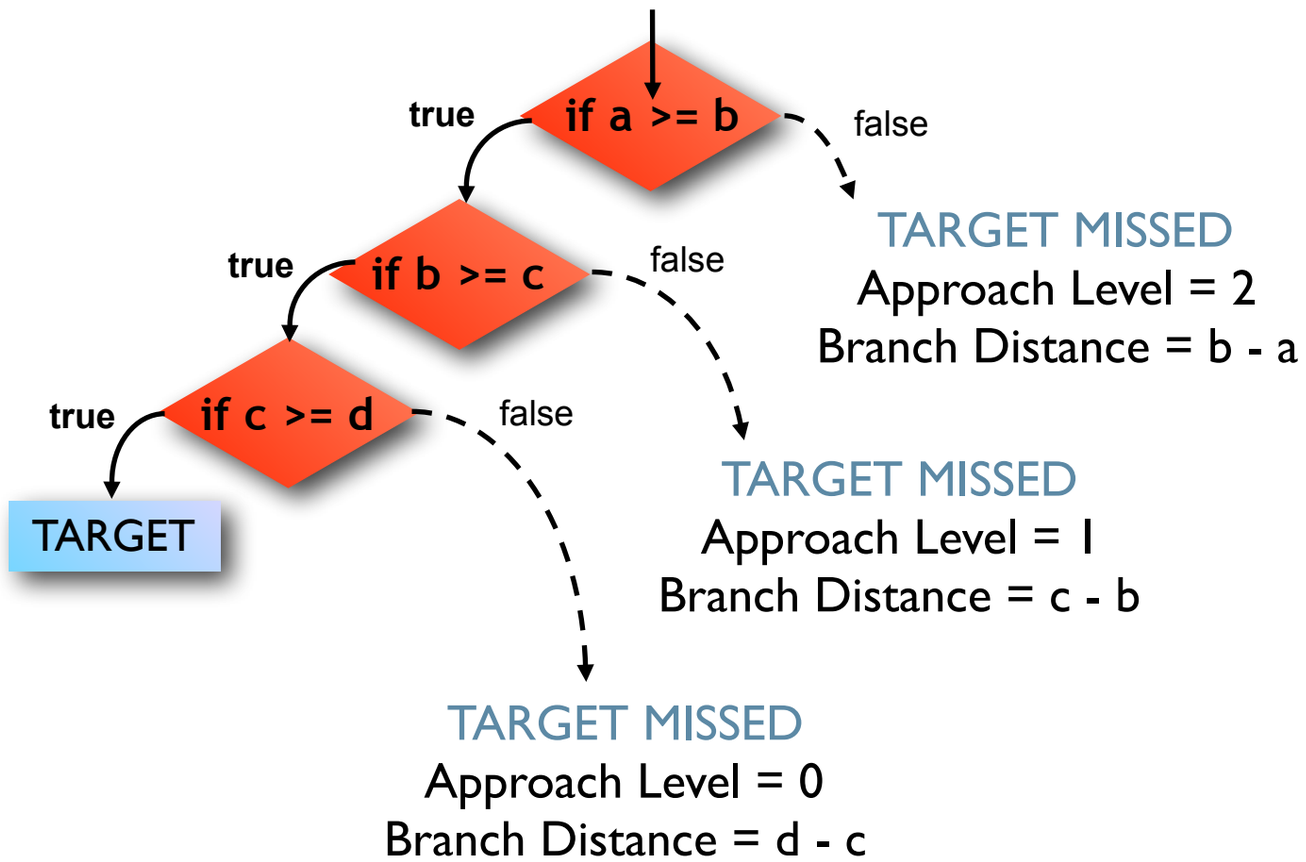
Approach Level



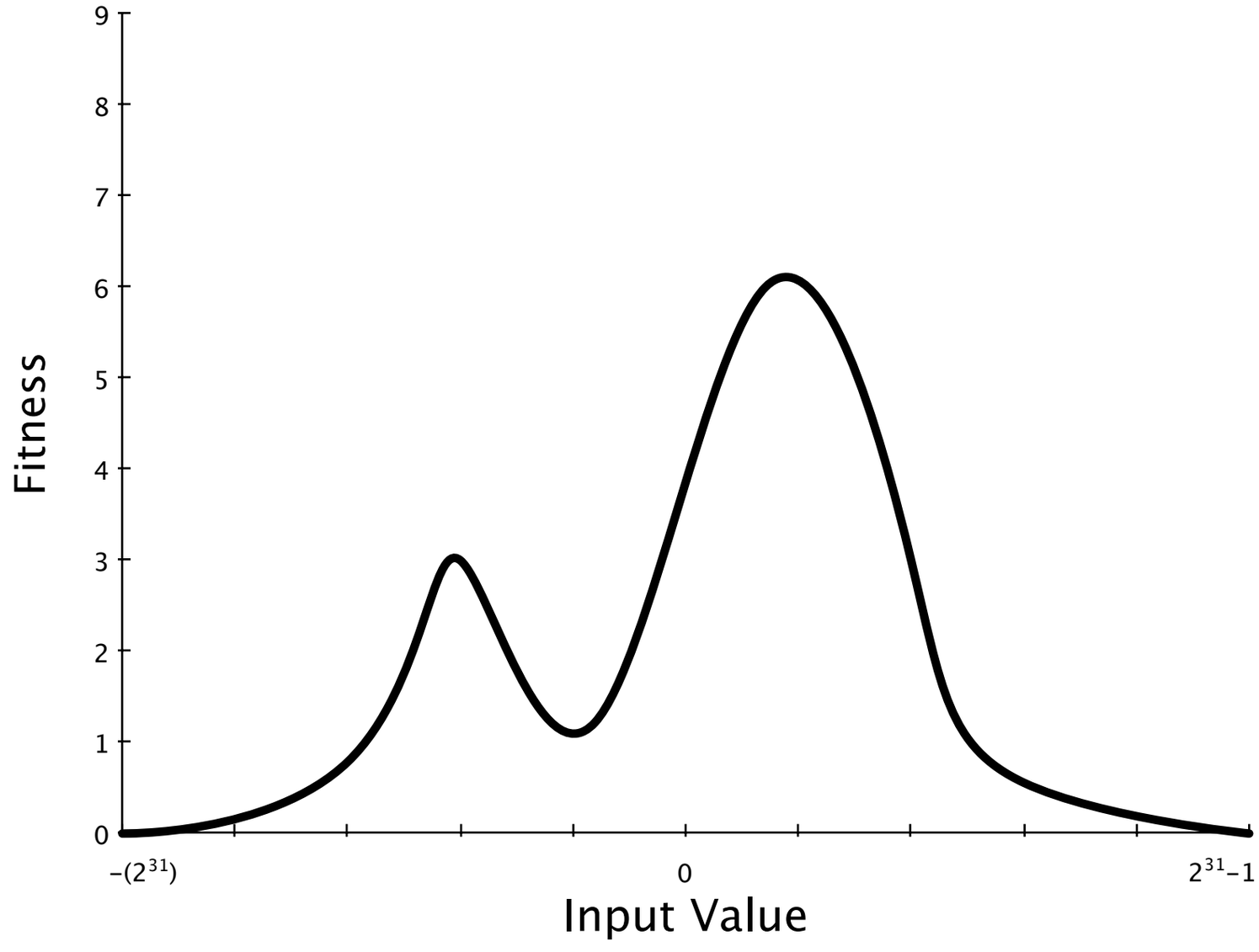
Putting it all together

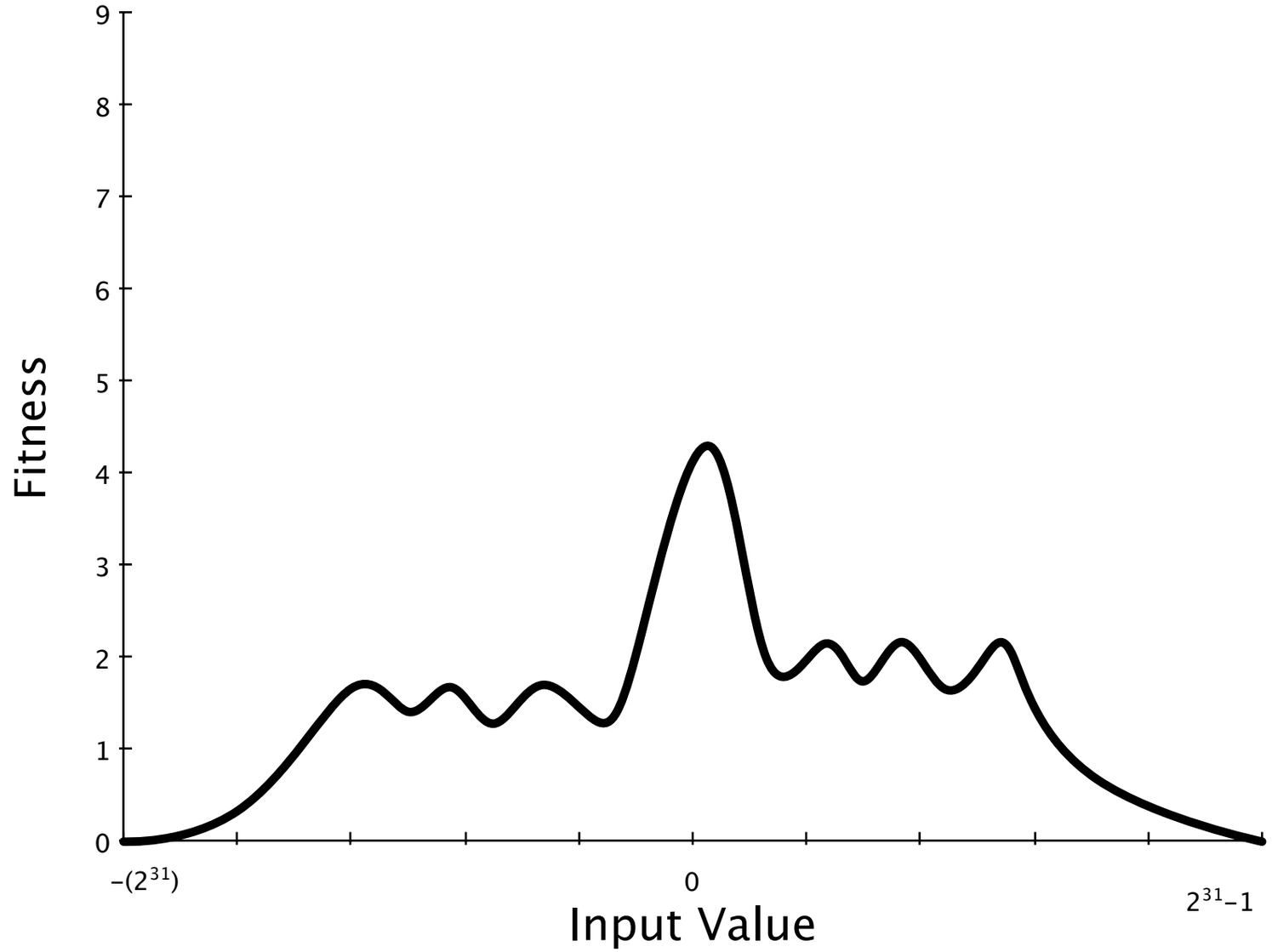
Fitness = approach Level + *normalised* branch distance

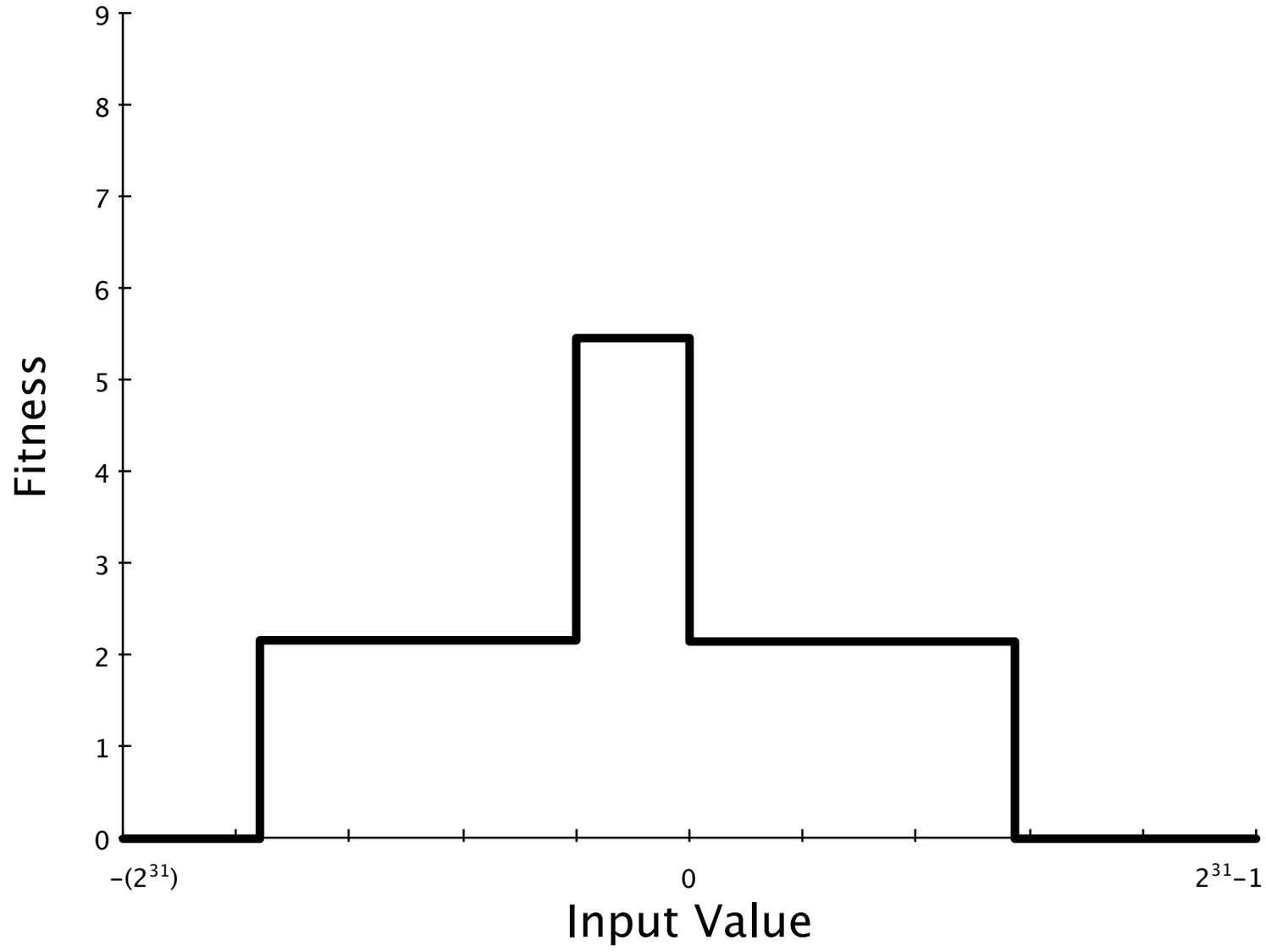
```
void f1(int a, int b, int c, int d)
{
    if (a > b)
    {
        if (b > c)
        {
            if (c > d)
            {
                // target
. . .
```



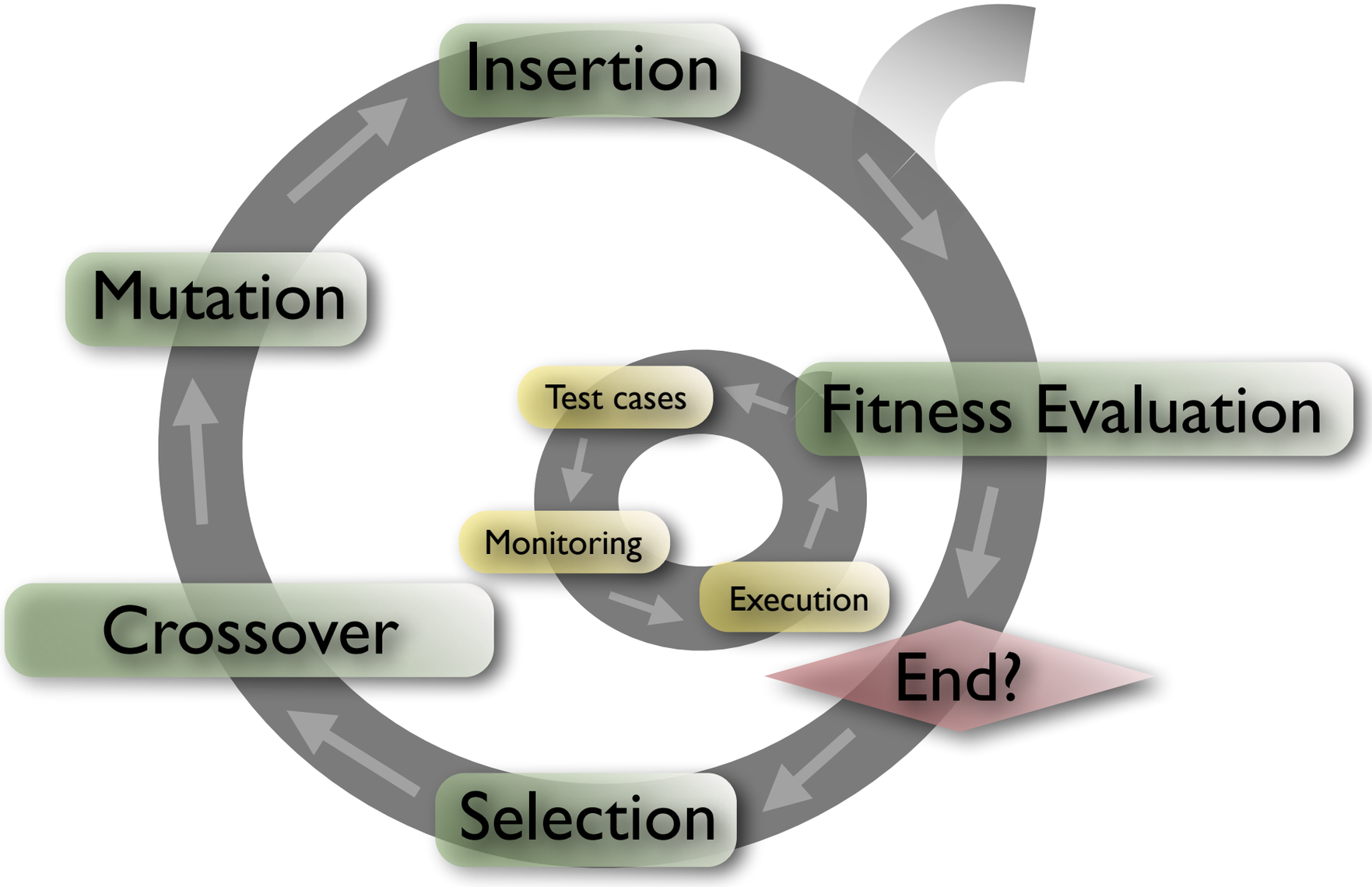
normalised branch distance between 0 and 1
indicates how close approach level is to being penetrated


Fitness
9
8
7
6
5
4
3
2
1
0
-(2^31)
0
2^31-1
Input Value


Fitness
9
8
7
6
5
4
3
2
1
0
-(2^31)
0
2^31-1
Input Value


Fitness
9
8
7
6
5
4
3
2
1
0
-(2^31)
0
2^31-1
Input Value

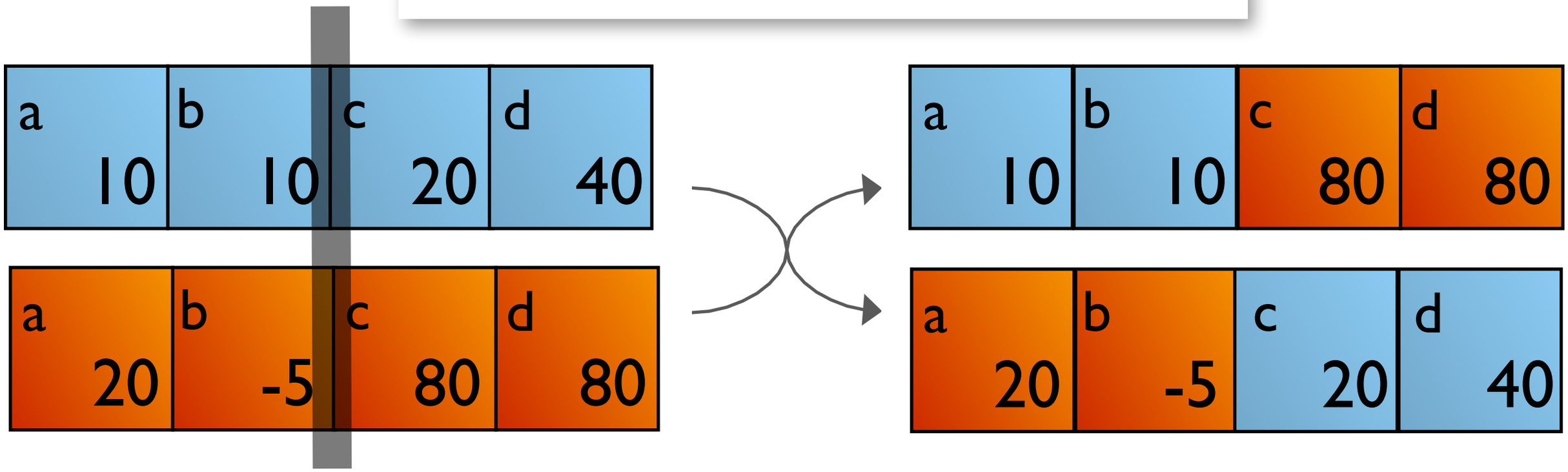
Evolutionary Testing



```
void test_me(int a, int b, int c, int d) {

   if (a == b) {

       if (c == d) {

           // branch we want to execute

       }
   }

   ...
```

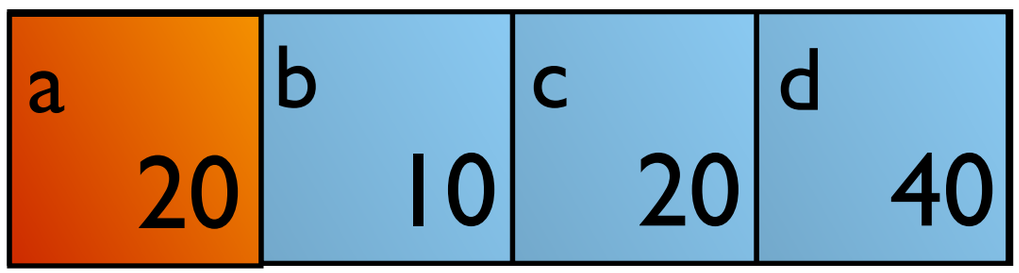
Crossover

```
void test_me(int a, int b, int c, int d) {

   if (a == b) {

       if (c == d) {

           // branch we want to execute

       }
   }

   ...
```



Mutation

```
void test_me(int a, int b, int c, int d) {

   if (a == b) {

       if (c == d) {

           // branch we want to execute

       }
   }

   ...
```



Selection

- Selective pressure:
  The higher, the more likely the fittest are chosen
- Stagnation:
  Selective pressure too small
- Premature convergence:
  Selective pressure too high
- Standard algorithms:
  Rank selection, tournament selection, roulette wheel selection

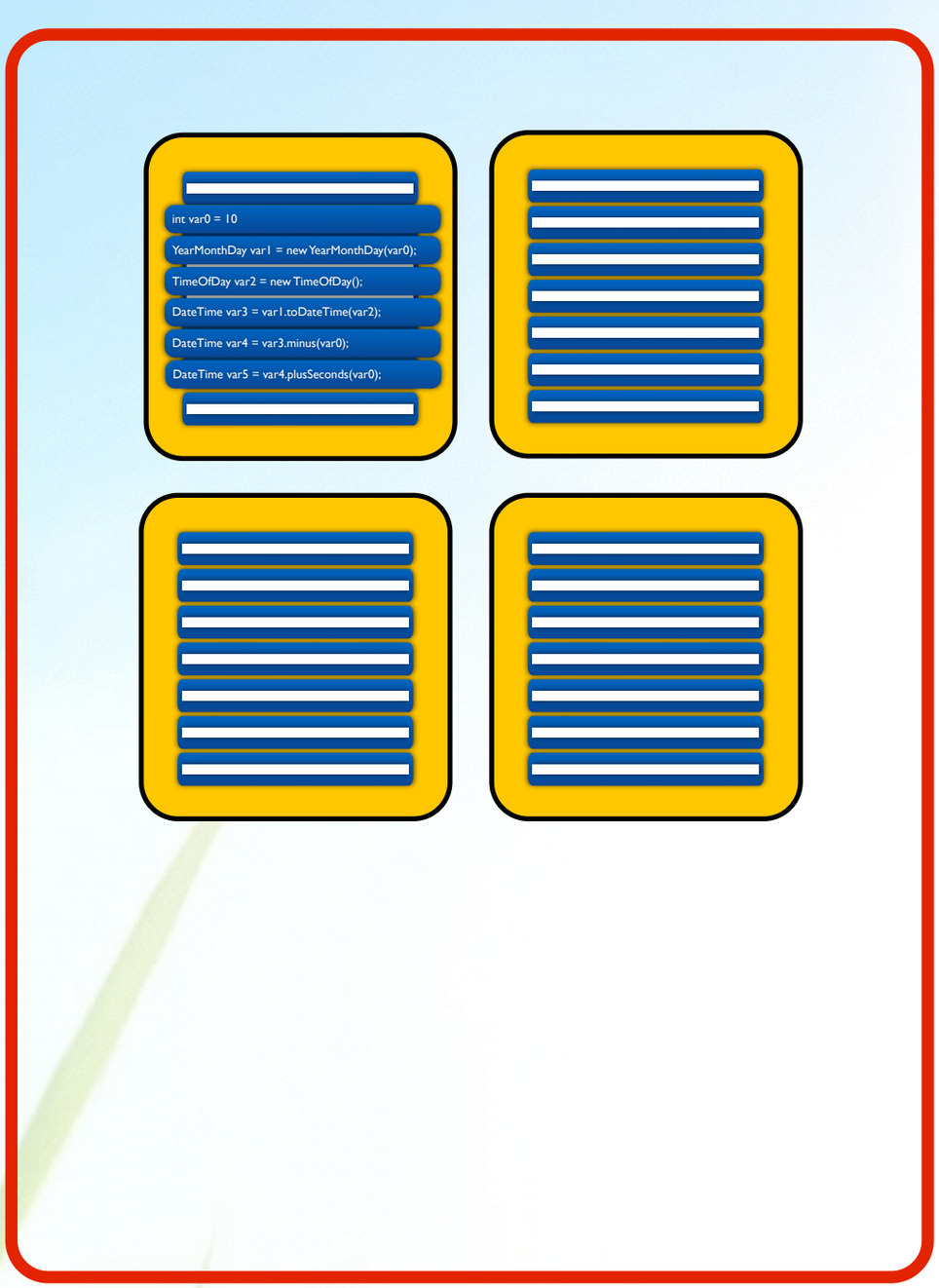
Contents

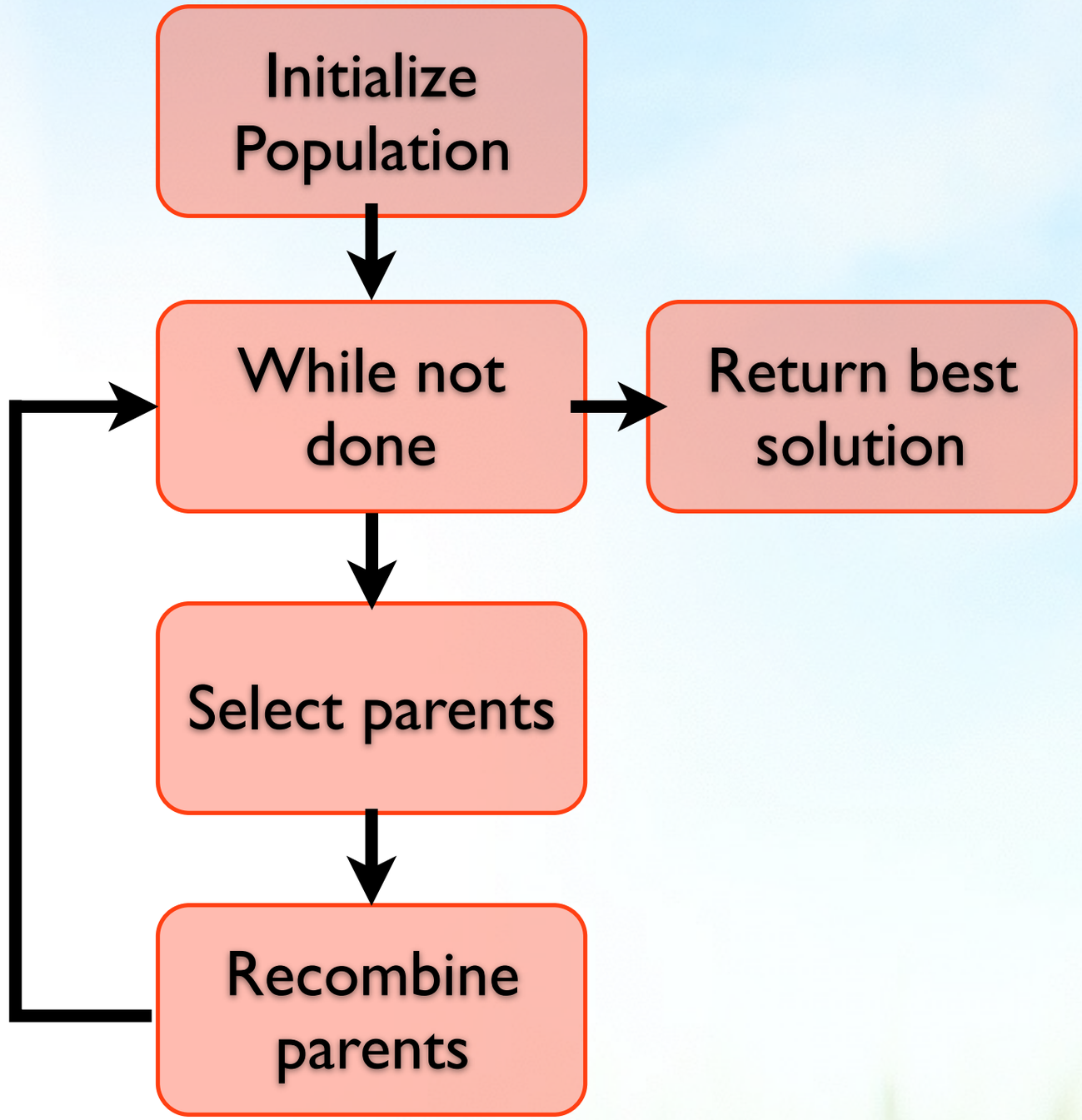
```
@Test
public void test()
{
   int x = 2;
   int y = 2;
   int result = x + y;
   assertEquals(4, result);
}
```

EVOSUITE

```
@Test
public void test()
{
    int var0 = 10
    YearMonthDay var1 = new YearMonthDay(var0);
    TimeOfDay var2 = new TimeOfDay();
    DateTime var3 = var1.toDateTime(var2);
    DateTime var4 = var3.minus(var0);
    DateTime var5 = var4.plusSeconds(var0);
}
```

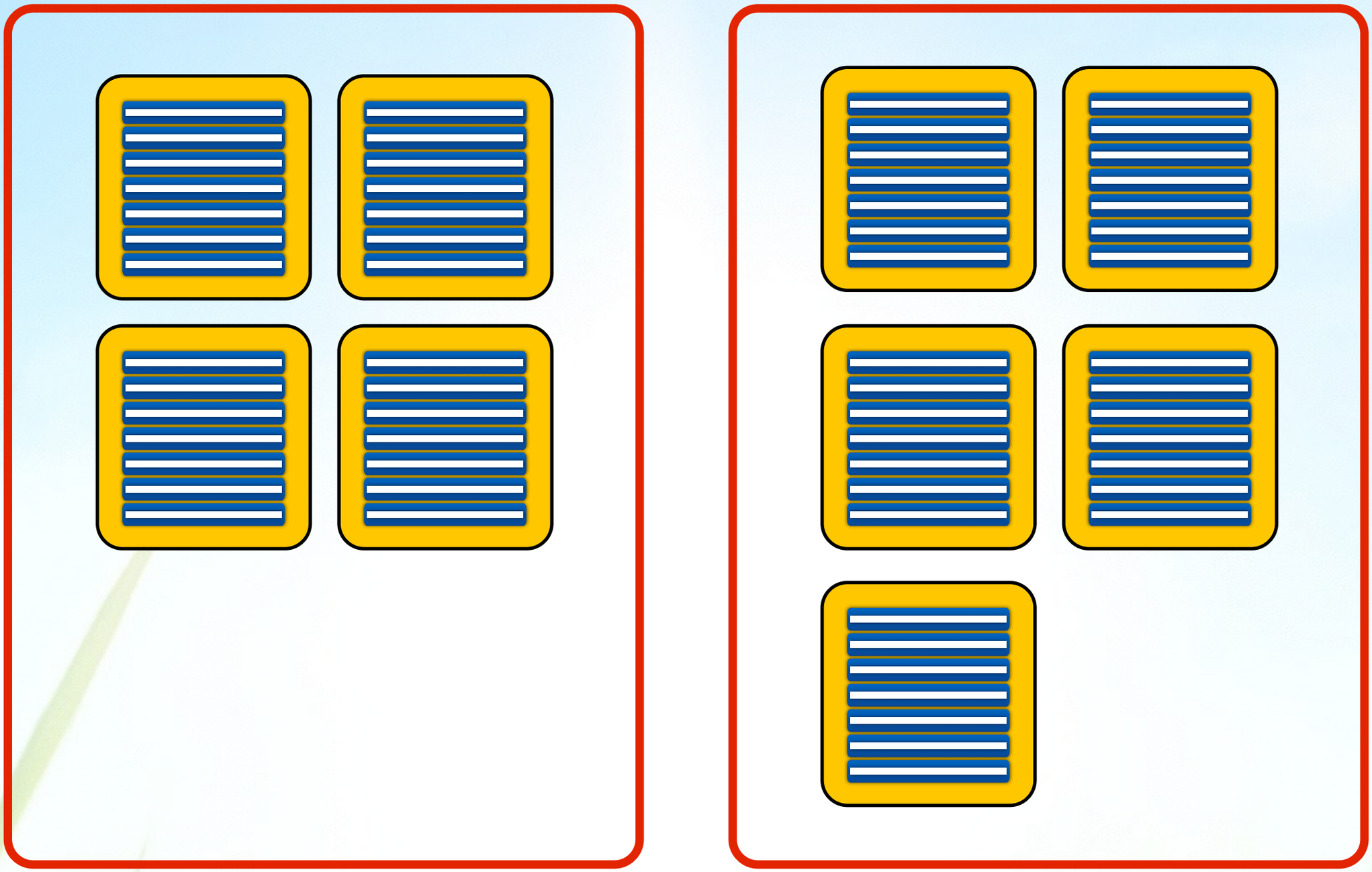
Test Suite Generation



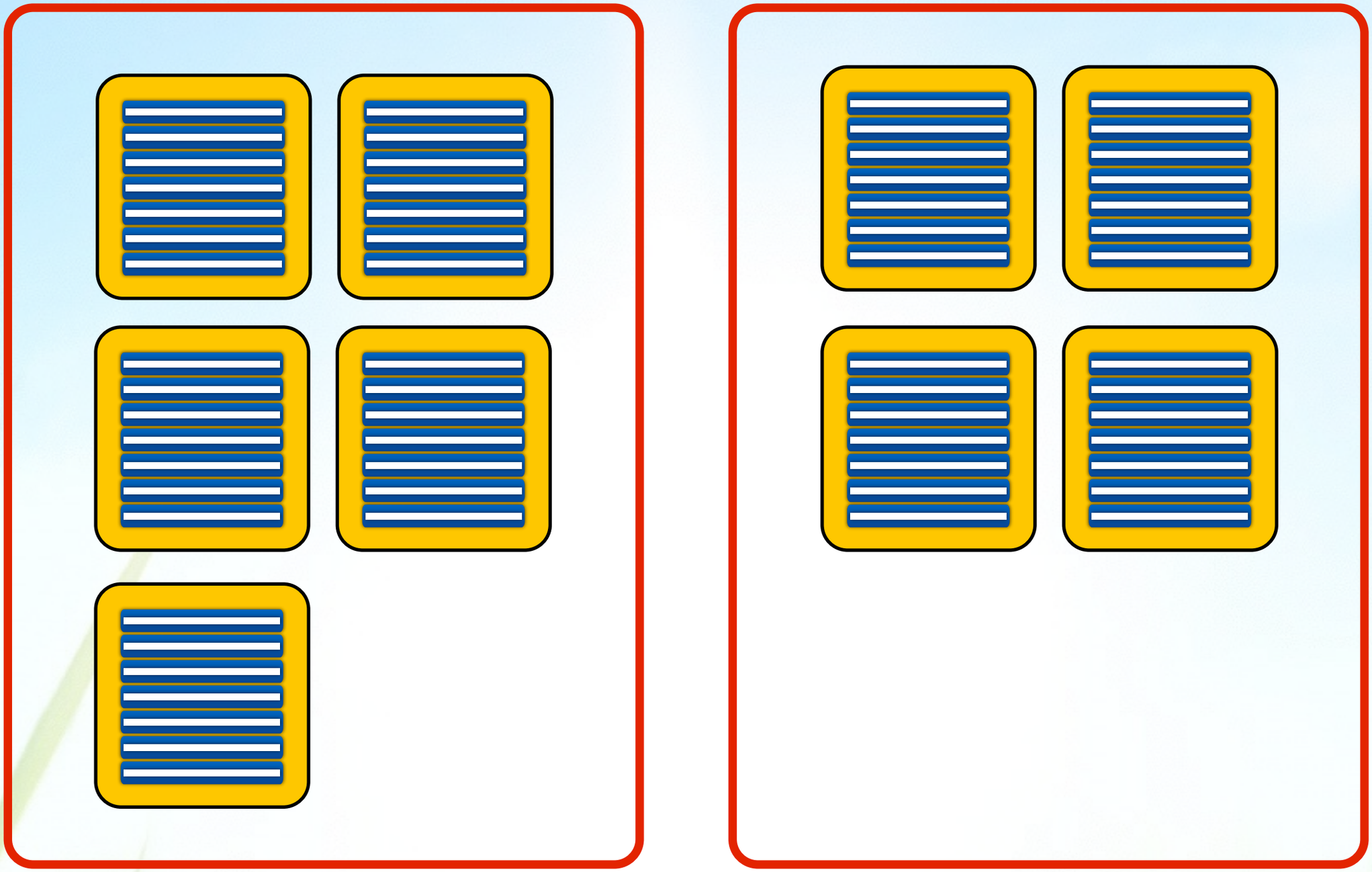


EVOSUITE

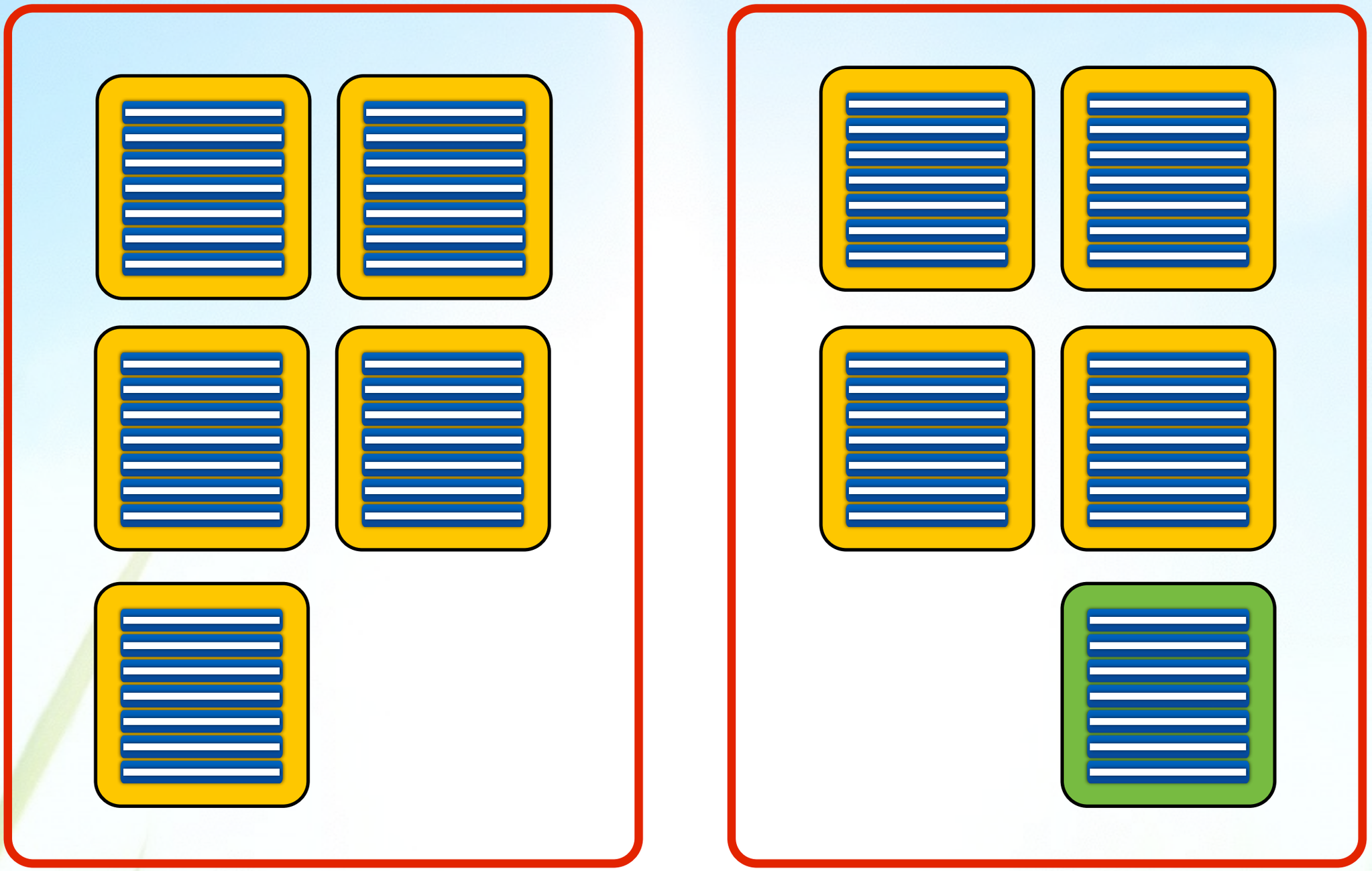
Test Suite Generation



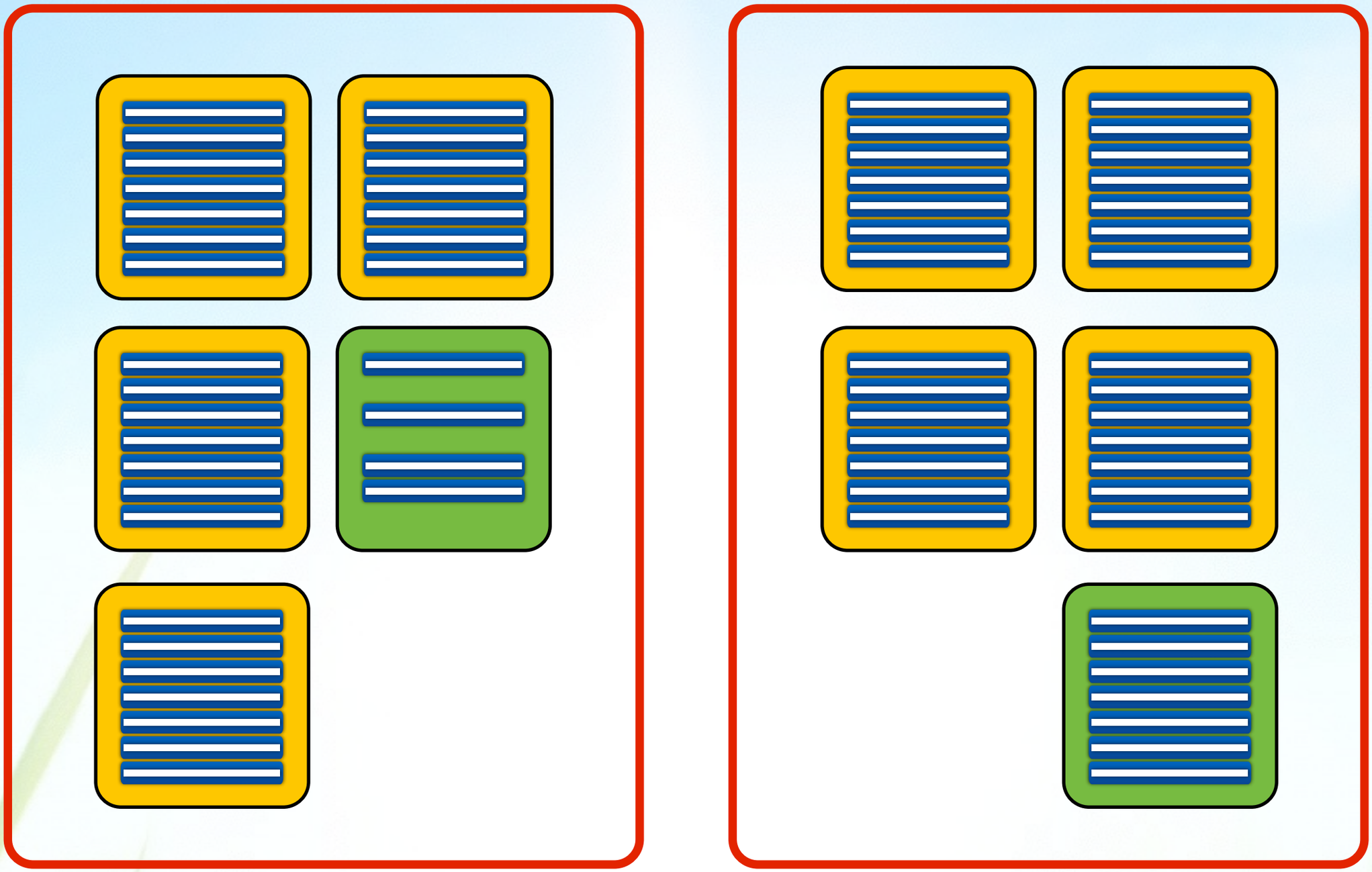
Crossover



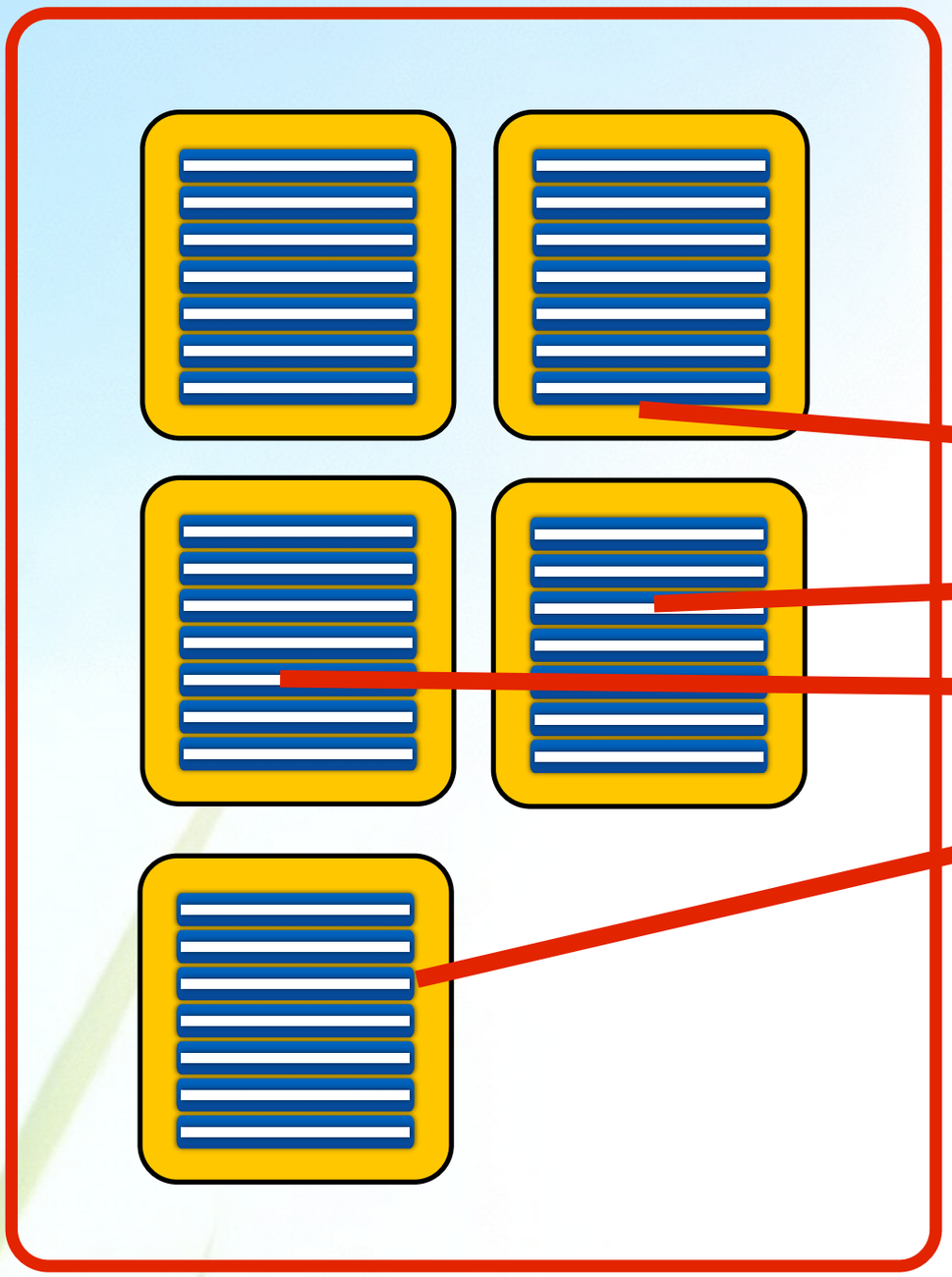
Mutation



Mutation



Fitness



```
public int gcd(int x, int y) {
  int tmp;
  while (y != 0) {
    tmp = x % y;
    x = y;
    y = tmp;
  }
  return x;
}
```

Components of an SBST Tool

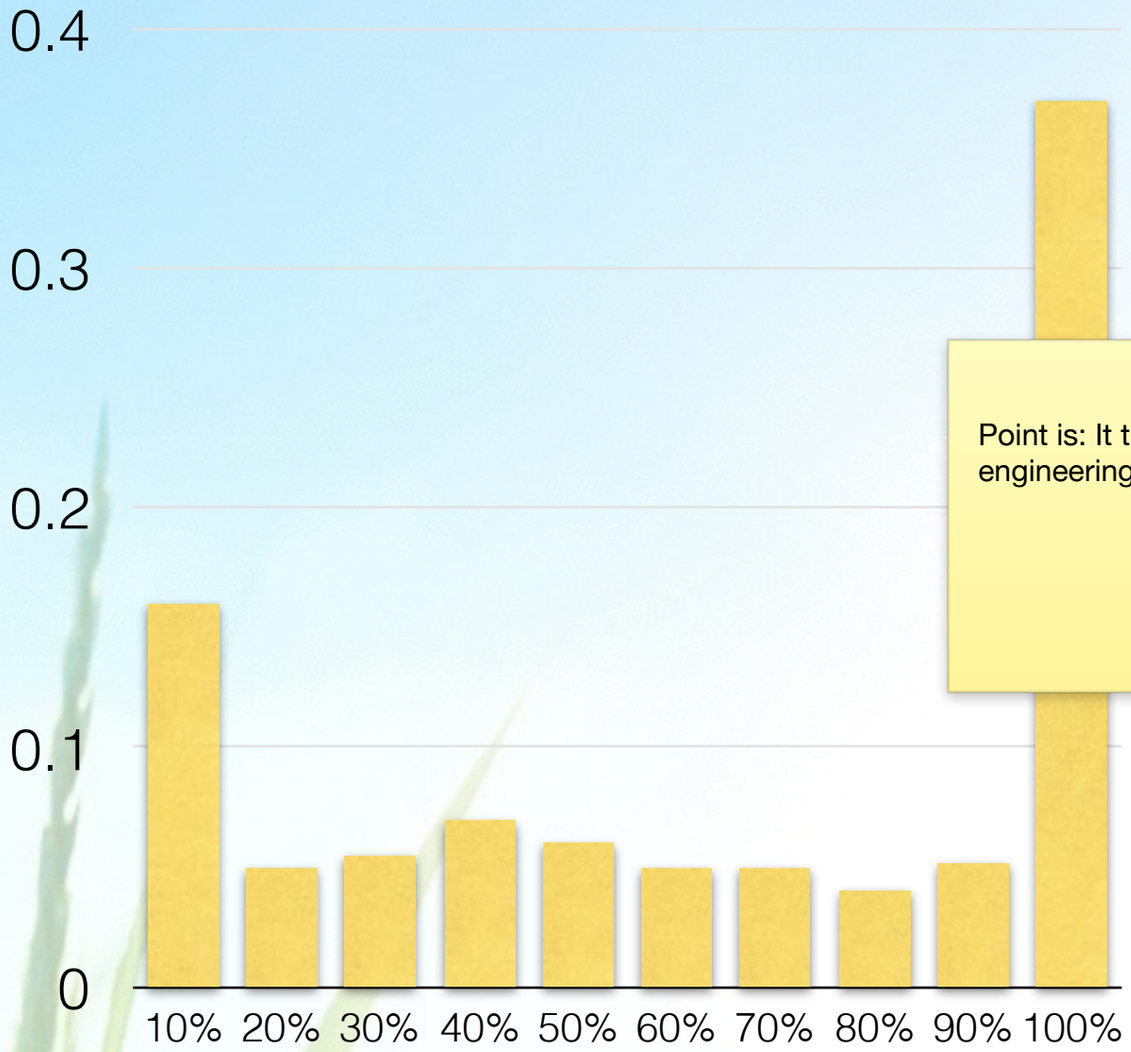
Search Algorithm	Genetic Algorithm (+Archive, Seeding, Local Search, DSE)
Representation	Sets of sequences of Java statements
Search Operators	Standard GA operators implemented for test suites
Fitness Function	Sum of branch distances (and others)
Test Execution	Java reflection
Instrumentation	Java bytecode instrumentation

EvoSuite

`http://www.evosuite.org/downloads`

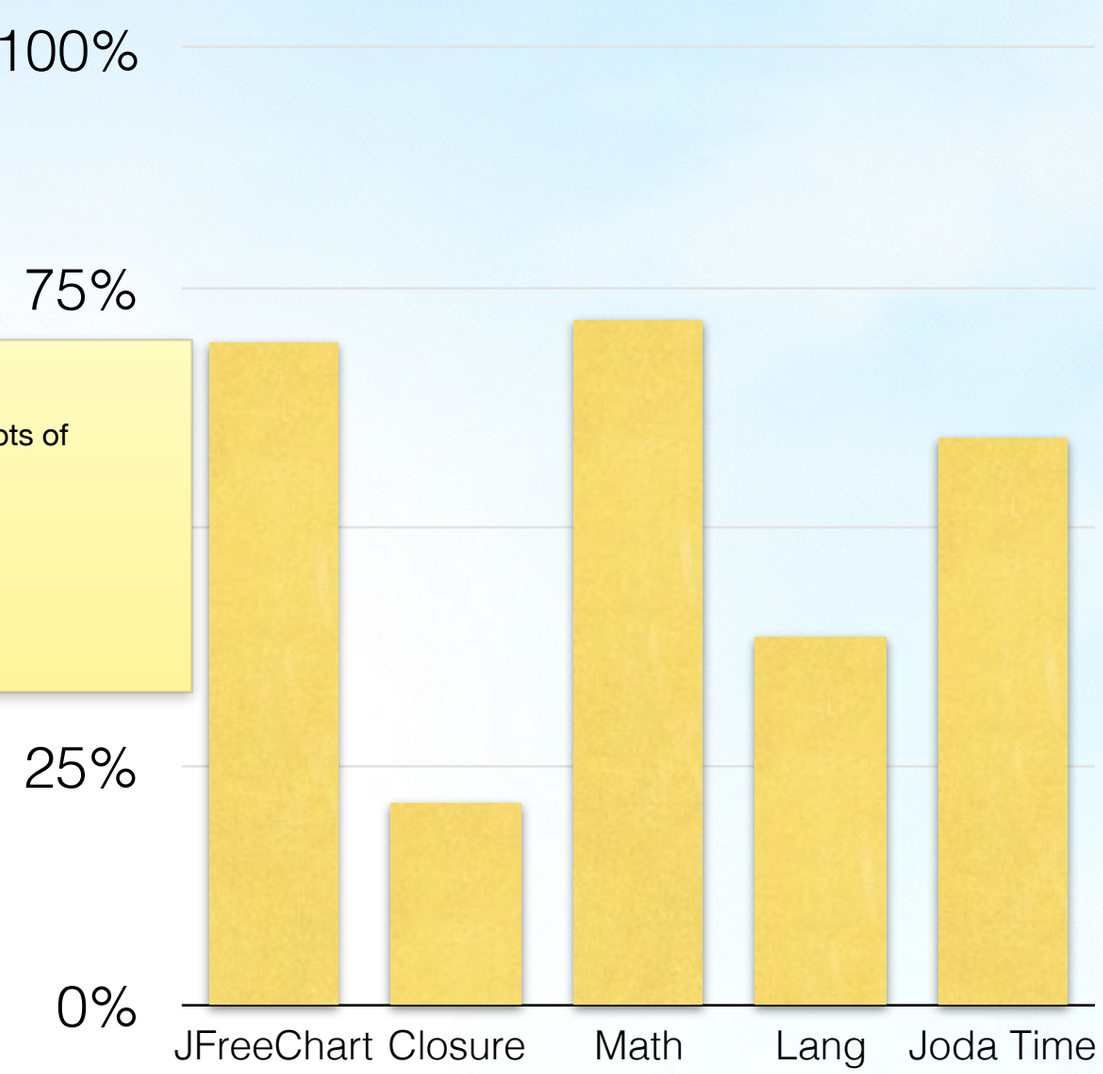
- Jar release - for command line usage
- Maven plugin
- IntelliJ plugin
- Eclipse plugin
- Jenkins plugin

Does it work?



SF110: 23,886 Classes
6,628,619 LOC

G. Fraser, A. Arcuri. "A Large Scale Evaluation of Automated Unit Test Generation with EvoSuite" TOSEM 24(2), 2014.

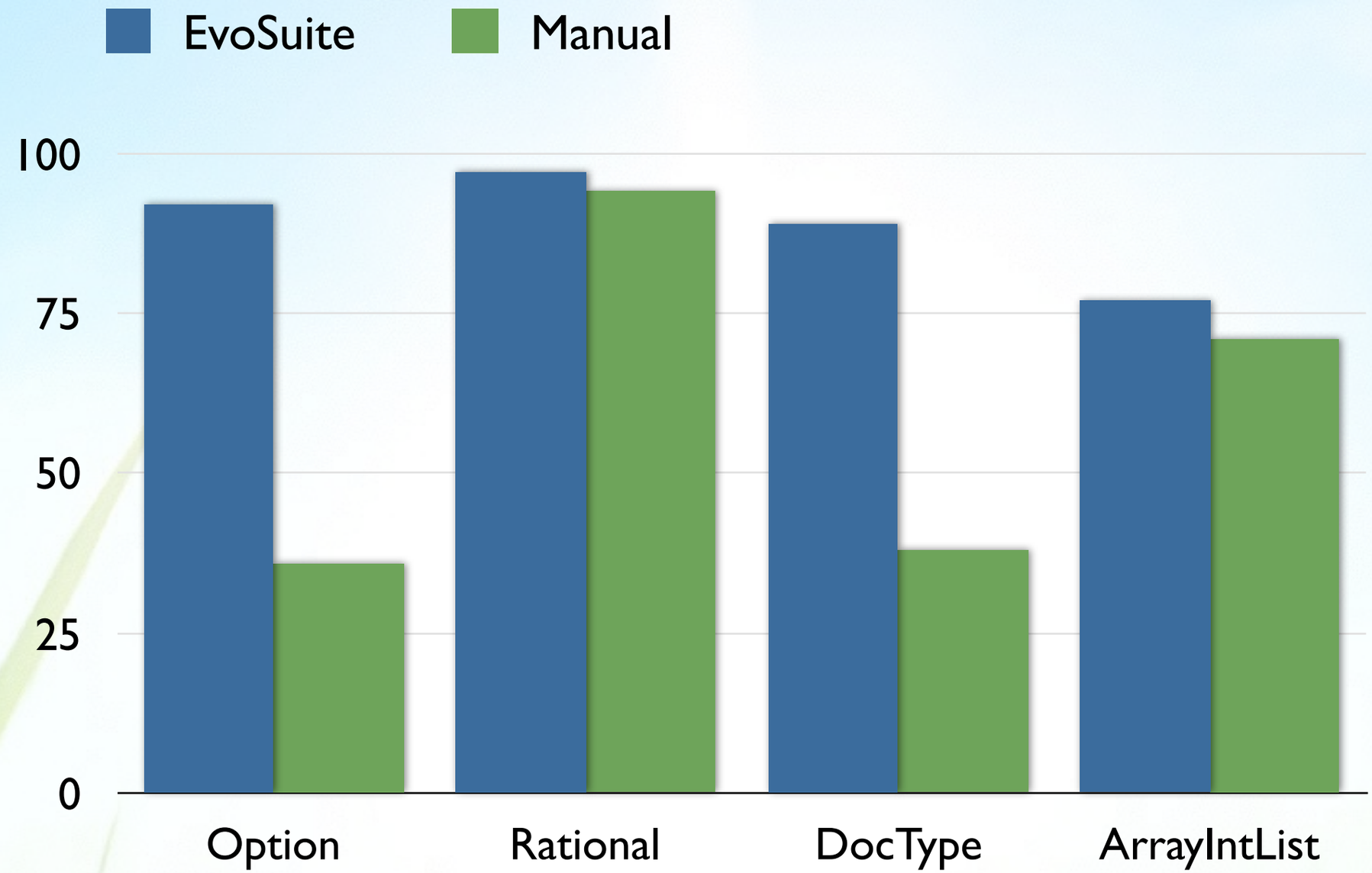


Defects4J: 357 real bugs

Shamshiri et al. "Do Automatically Generated Unit Tests Find Real Faults? An Empirical Study of Effectiveness and Challenges" ASE, 2015

Point is: It takes a tool and lots of engineering to do this.

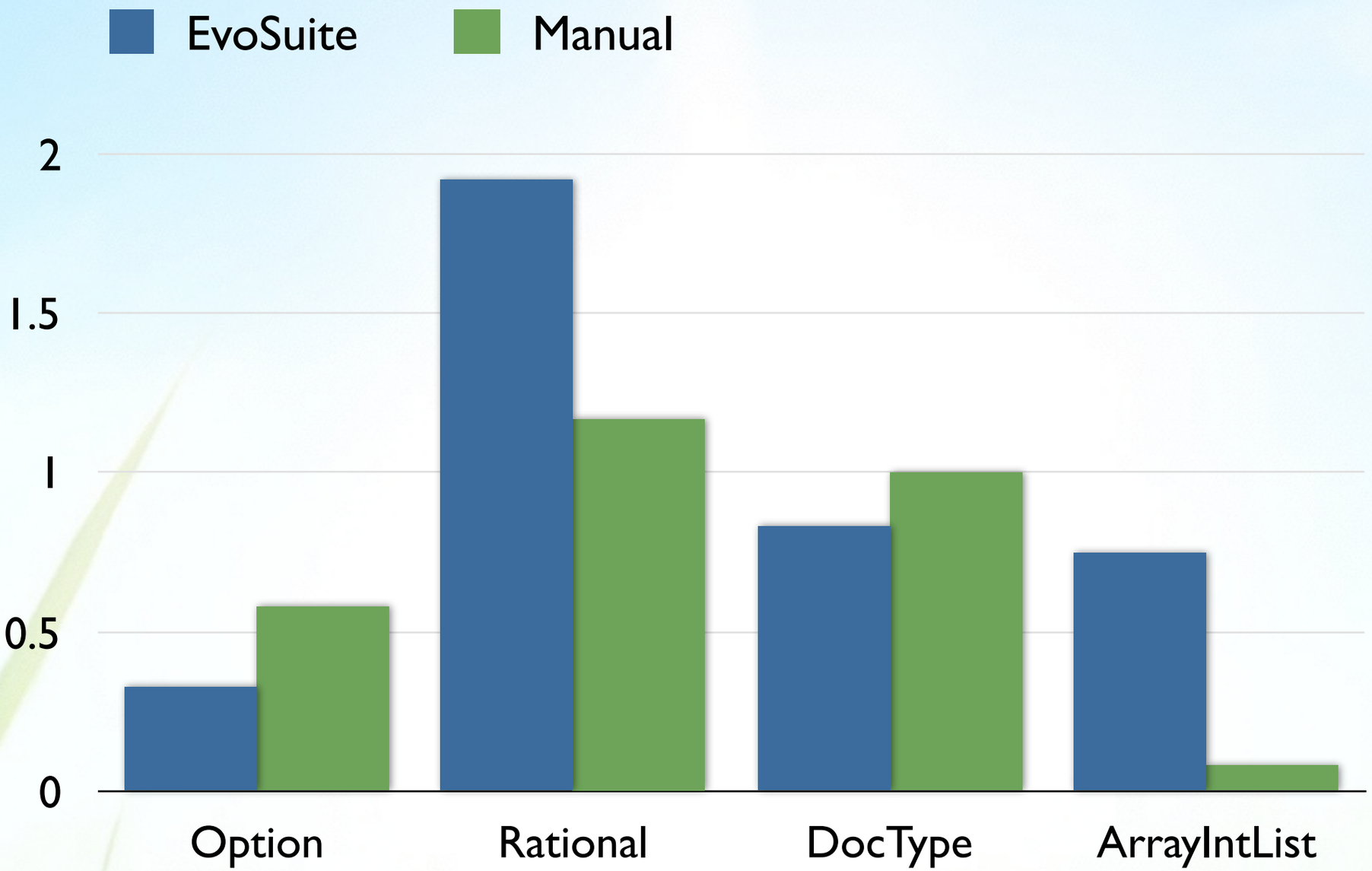
Coverage


EvoSuite
Manual
100
75
50
25
0
Option
Rational
DocType
ArrayIntList

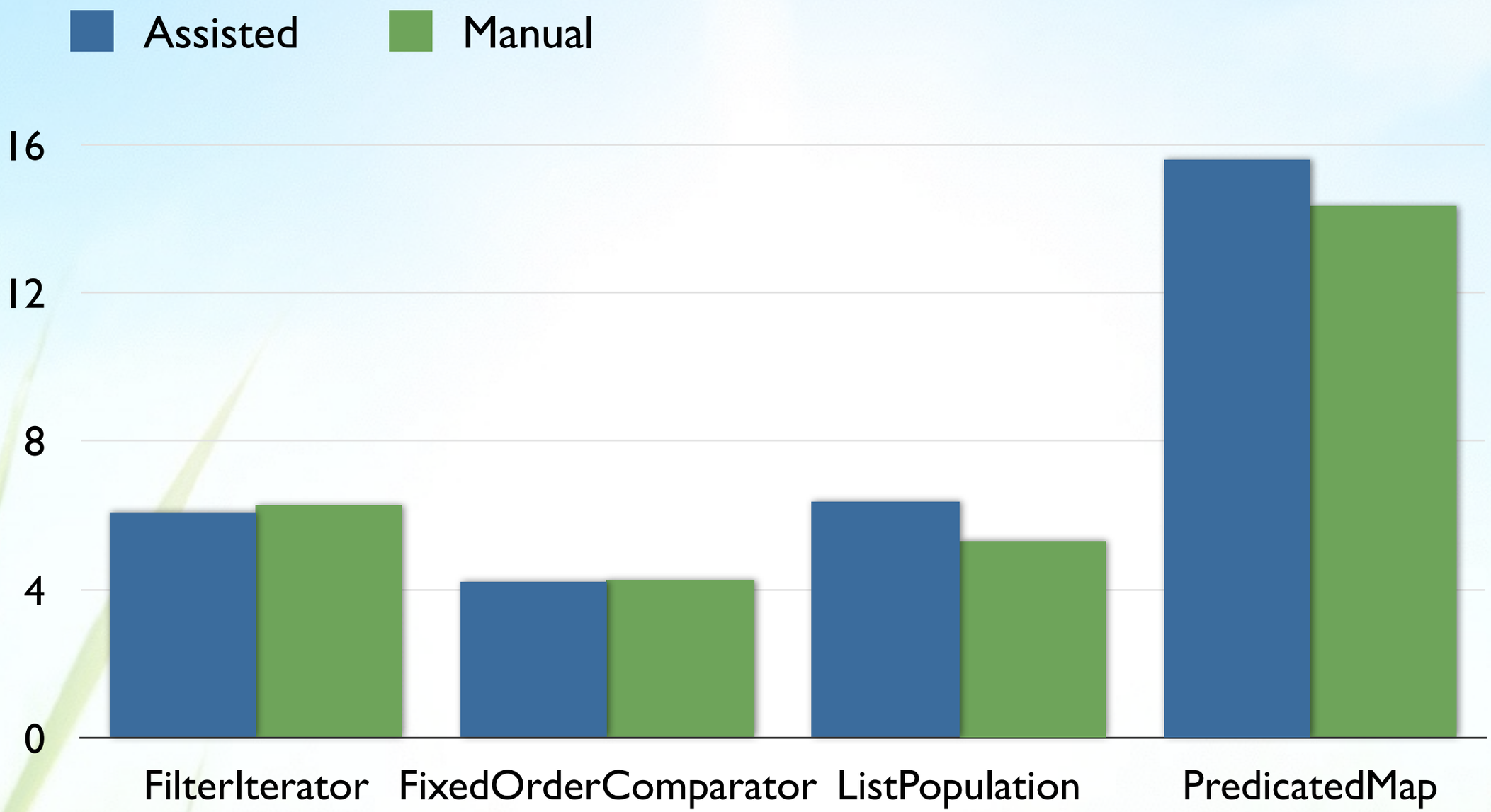
EVOSUITE

Time Spent on Testing

Fault Detection

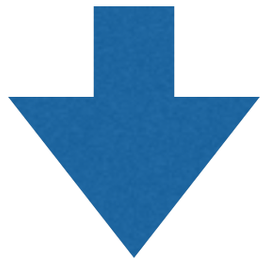


Faults Prevention



Method Names

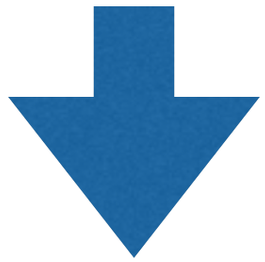
```
@Test(timeout = 4000)
public void test3()  throws Throwable  {
    StringExample stringExample0 = new StringExample();
    boolean boolean0 = stringExample0.foo("");
    assertFalse(boolean0);
}
```



```
@Test(timeout = 4000)
public void testFooReturningFalse()  throws Throwable  {
    StringExample stringExample0 = new StringExample();
    boolean boolean0 = stringExample0.foo("");
    assertFalse(boolean0);
}
```

Variable Names

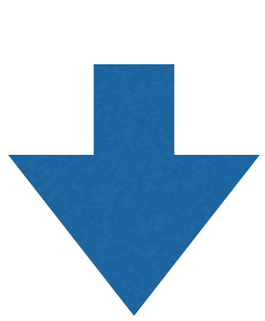
```
@Test(timeout = 4000)
public void testFooReturningFalse()  throws Throwable  {
    StringExample stringExample0 = new StringExample();
    boolean boolean0 = stringExample0.foo("");
    assertFalse(boolean0);
}
```



```
@Test(timeout = 4000)
public void testFooReturningFalse()  throws Throwable  {
    StringExample invokesFoo = new StringExample();
    boolean resultFromFoo = invokesFoo.foo("");
    assertFalse(resultFromFoo);
}
```

Variable Names

```
public class Foo {
   public void foo() {
      StringExample sx = new StringExample();
      boolean bar = sx.foo("");
   }
}
```



```
@Test(timeout = 4000)
public void testFooReturningFalse()  throws Throwable  {
    StringExample sx = new StringExample();
    boolean bar = sx.foo("");
    assertFalse(bar);
}
```

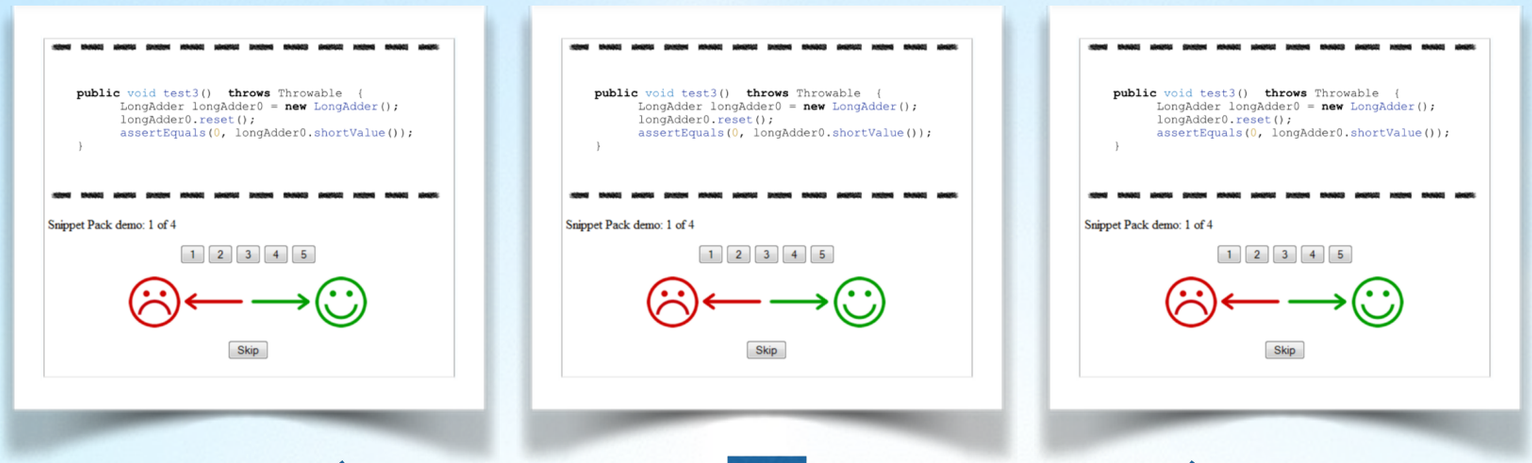
EVOSUITE

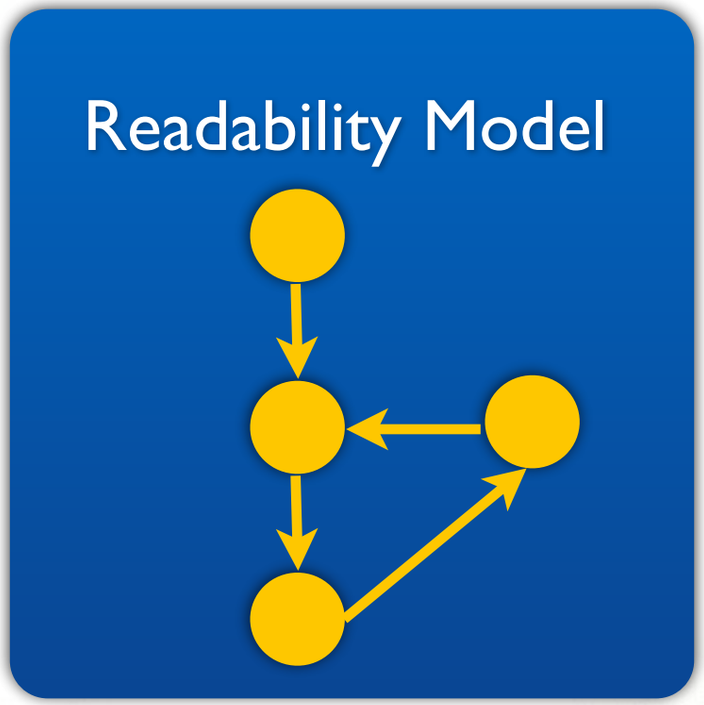
```
public void test3()  throws Throwable  {
        LongAdder longAdder0 = new LongAdder();
        longAdder0.reset();
        assertEquals(0, longAdder0.shortValue());
}
```

Snippet Pack demo: 1 of 4

1 2 3 4 5

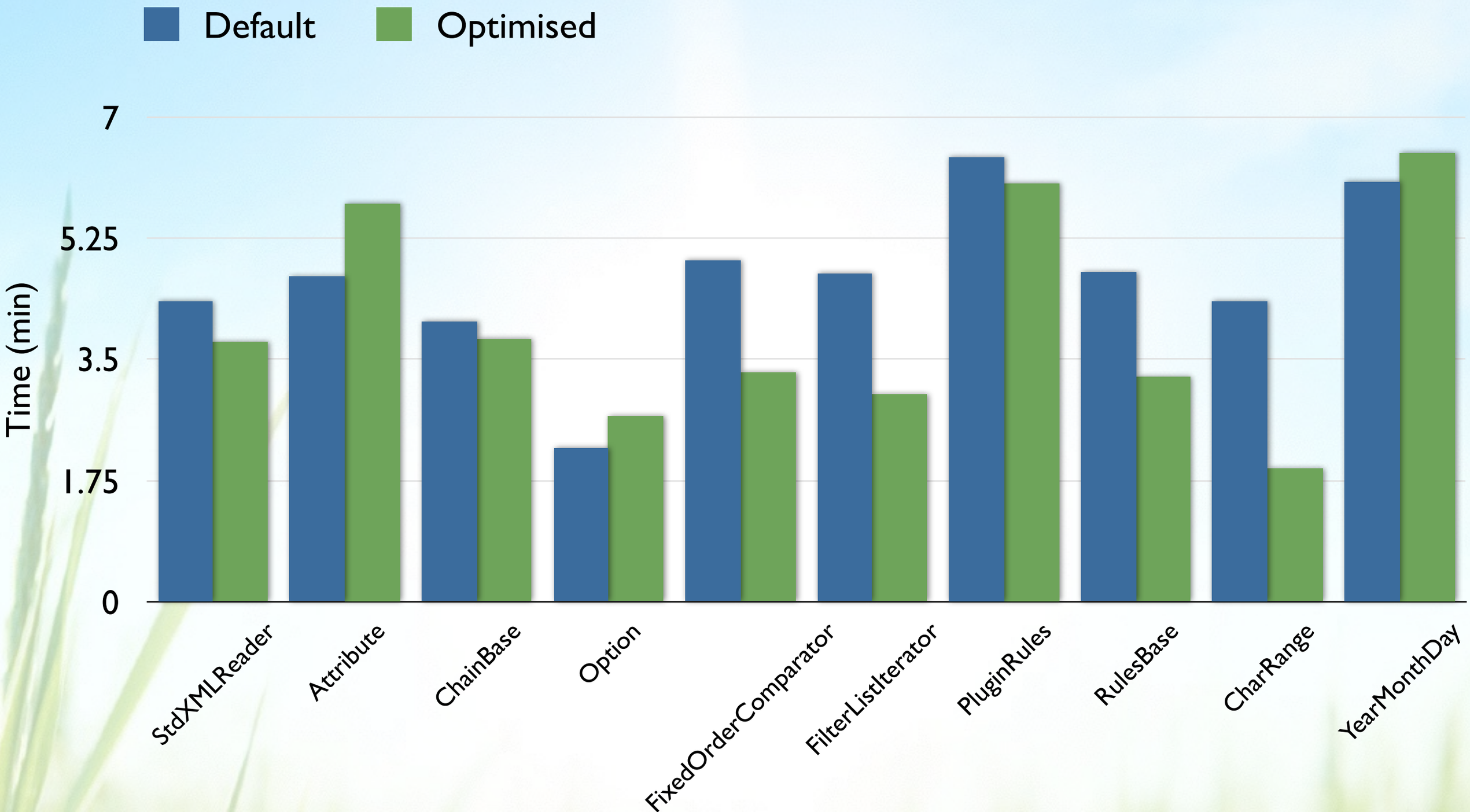
Skip


public void test3() throws Throwable {
LongAdder longAdder0 = new LongAdder();
longAdder0.reset();
assertEquals(0, longAdder0.shortValue());
}
Snippet Pack demo: 1 of 4
1 2 3 4 5
Skip


Readability Model

EVOSUITE

Time Spent Understanding



Contents

EVOSUITE

I . Java

…is a weird language
and never ceases to surprise me

My personal enemy: Java Generics

Bytecode over sourcecode - yes!

2. Corner Cases

The more corner cases you cover

...the more can go wrong

...the more new corner cases you will find

...the slower EvoSuite becomes

2. Corner Cases

- Constant Seeding: +5%
- Virtual FS: +1.4%
- Mocking +4.7%
- JEE support: +3%
- DSE: +1.2%

EVOSUITE

3. Developers

…some really care only about coverage

…others don’t care about coverage:

"I wouldn’t normally in real life be aiming for 100% coverage. I’d probably end up with fewer tests without this tool but I couldn’t tell you if they would be all the right tests.”

…do not want their tests to be generated

…hate ugly tests

…don’t like waiting

Talk to them!

3. Developers

```
public class Example {

    private Example() {}

    // …
}
```

4. Testing

Testing randomised algorithms is difficult

Make the implementation deterministic

Always use LinkedHashSet over HashSet, LinkedHashMap over HashMap

Java reflection is not deterministic

Avoid static state (e.g. singletons)

EVOSUITE

4. Testing

EvoSuite uses one central random number generator

Any change will affect something at a completely different part of the program

Change seeds frequently during testing to find flaky tests

EVOSUITE

5. Documentation

I don’t comment my code

Students struggle

I spend more time explaining things than it would take me to implement them

6. Tool Comparisons

Reviewers want to see them

I don't like doing them

It's impossible to make them fair

Contact tool authors

Report bugs

Make your own tools *usable*

7. Open Source

"The source code will be released under an open source library (most likely GPL2) at a later point, as soon as a number of refactorings are completed." — FSE'11 tool paper appendix

Public GitHub repo: 2015

It will never be clean enough, just release it!

8. Licensing

License matters

Google will not touch GPL

BSD, MIT - do you want others to become rich with your idea?

Gnu Lesser Public License, Apache

9. Tool Papers

The first one will be cited

The rest no one will cite

It shouldn't be this way

EVOSUITE

10. Tool Building

Building a quick prototype is easy

…and will give you a paper

Building a real tool is difficult

…but lets you identify many new problems

…lets you talk to developers

…lets other people build on your work

…will give you lots of citations and papers

EVOSUITE

10. Tool Building

Build

Search-Based Software Engineers Need Tools!

Build

lems

rk

…will give you lots of citations and papers

EVOSUITE

www.evosuite.org