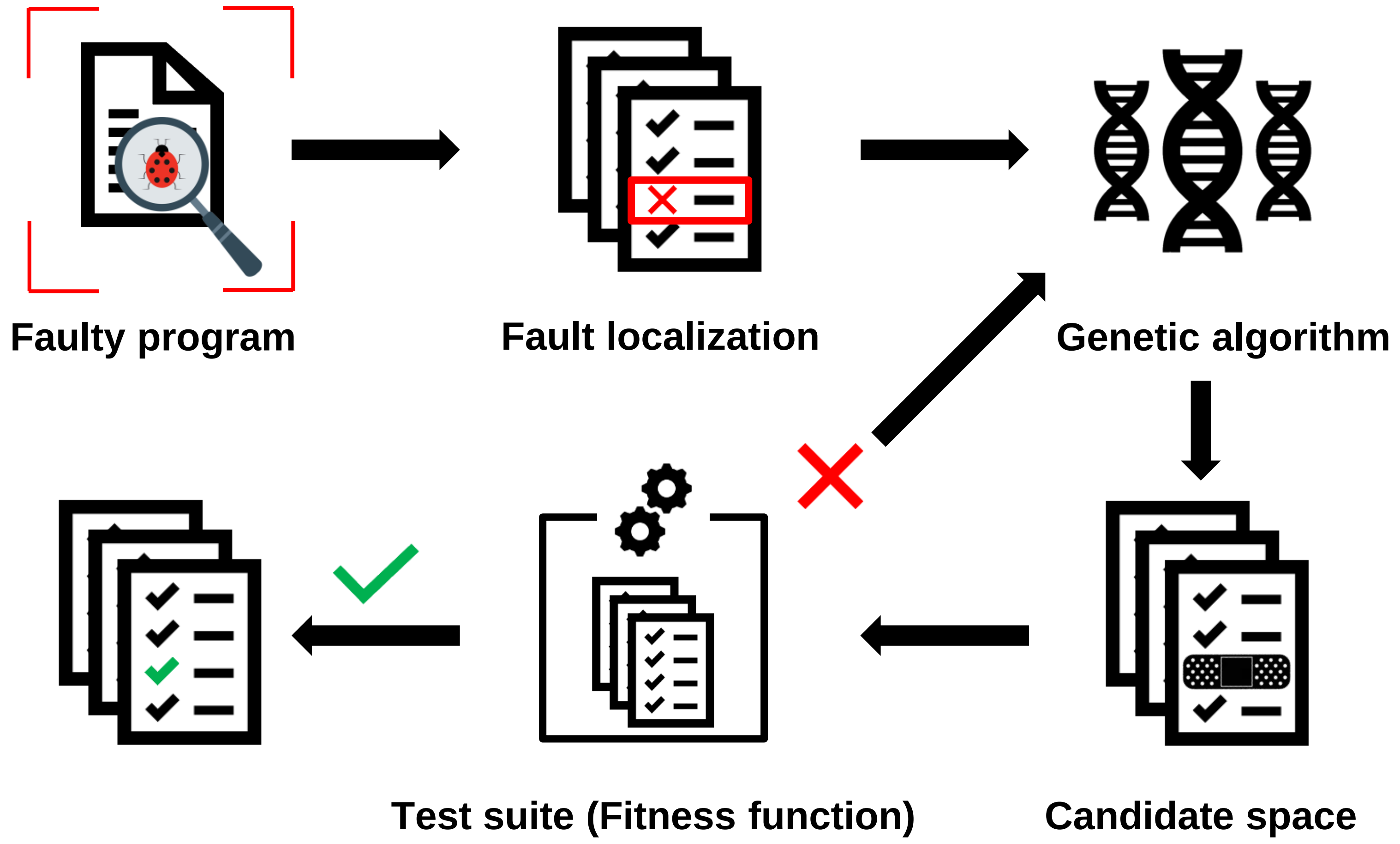
Automated Patching Using Genetic Programming

TEAM 8

Peerapon Akkapusit, Zuzana Jelčicová, Zelalem Mihret, Peter Muschick

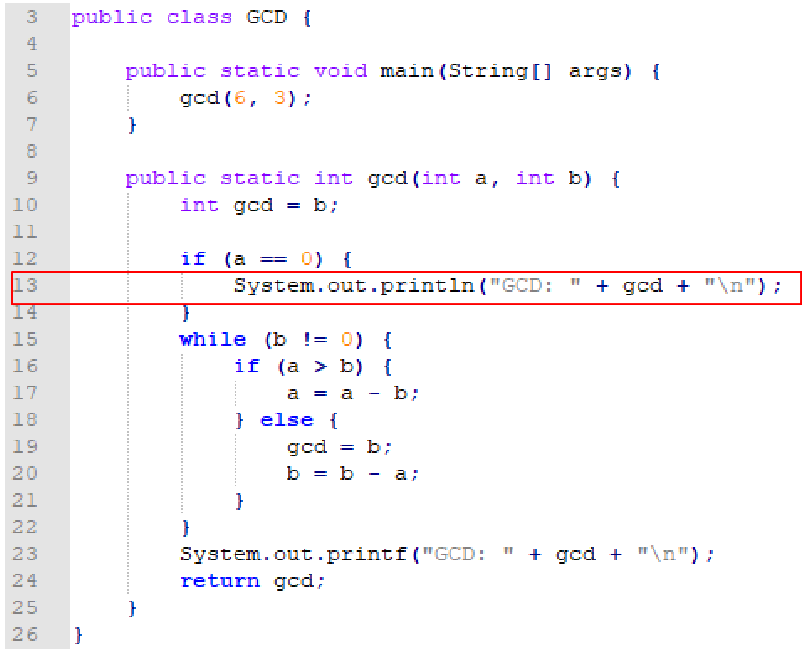
Agenda

- Project topic recap
- One lifecycle GA example (GCD)
- Experiments
- Results
- Discussion

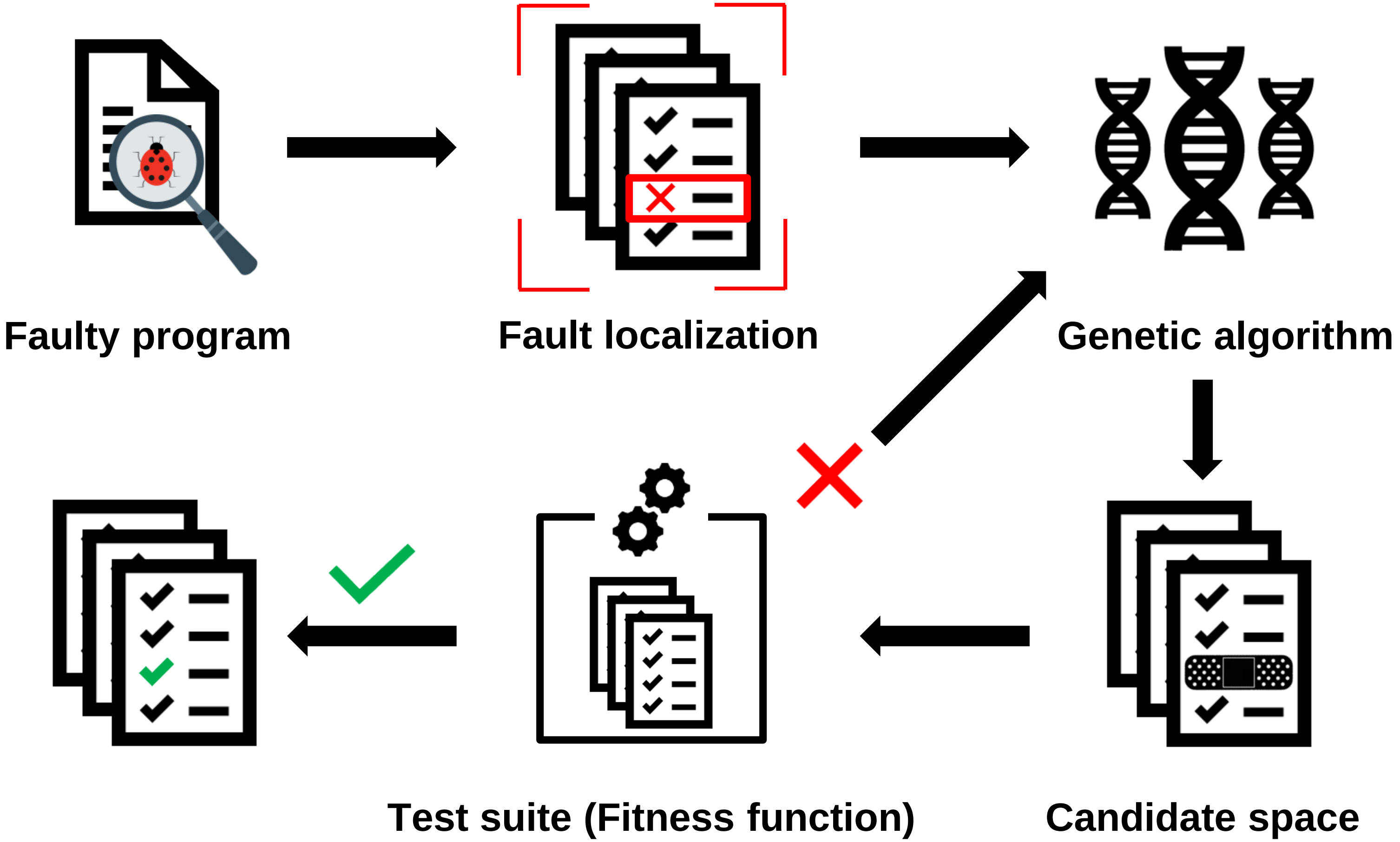

Faulty program
Fault localization
Genetic algorithm
Test suite (Fitness function)
Candidate space

Faulty program - Greatest Common Divisor (GCD)

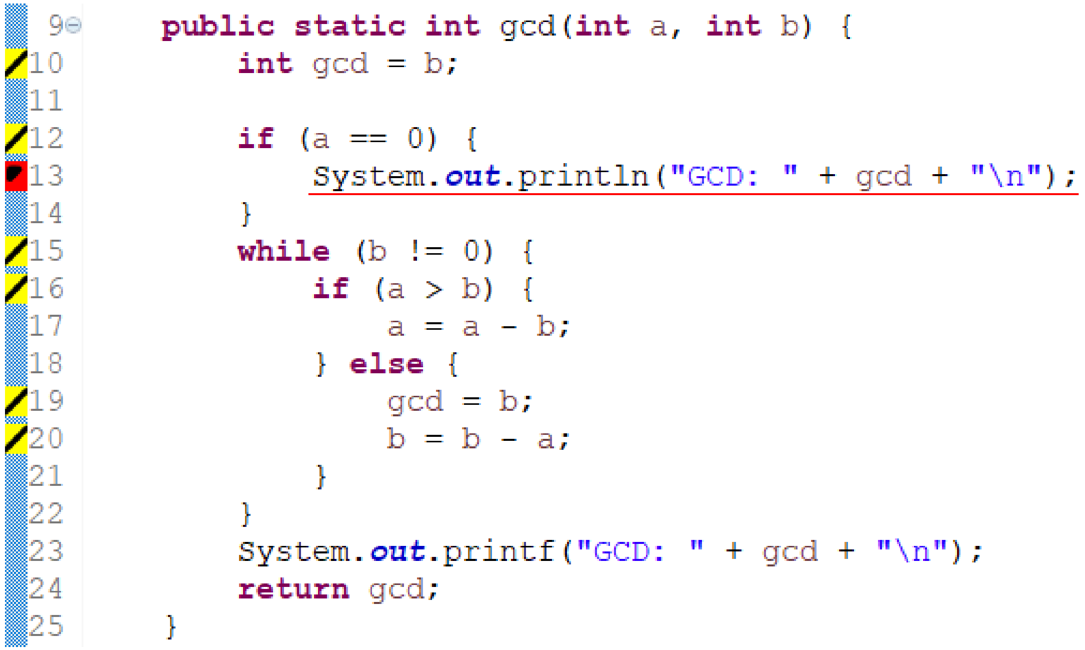
Example input:
gcd(0, 20)



```
public class GCD {

    public static void main(String[] args) {
        gcd(6, 3);
    }

    public static int gcd(int a, int b) {
        int gcd = b;

        if (a == 0) {
            System.out.println("GCD: " + gcd + "\n");
        }
        while (b != 0) {
            if (a > b) {
                a = a - b;
            } else {
                gcd = b;
                b = b - a;
            }
        }
        System.out.printf("GCD: " + gcd + "\n");
        return gcd;
    }
}
```

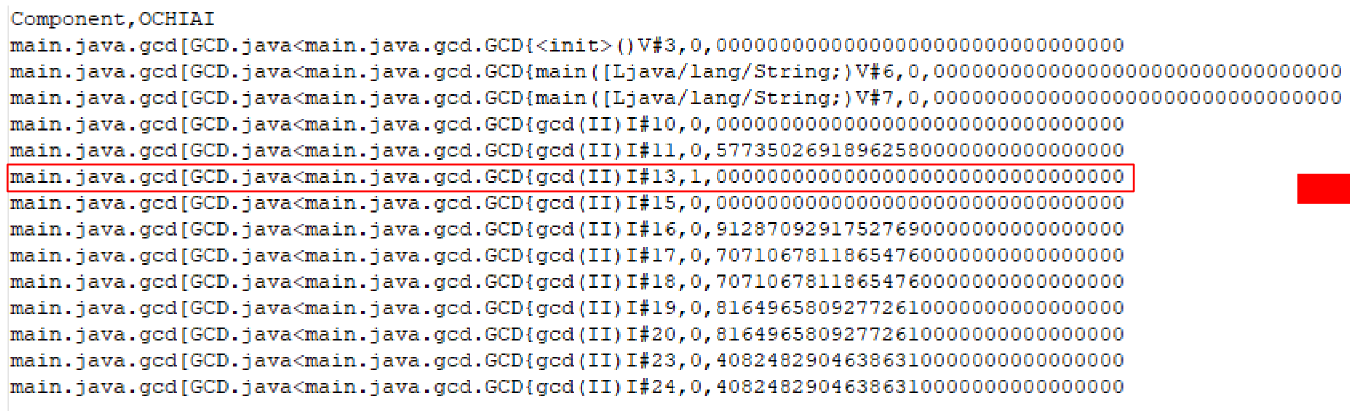

Faulty program
Fault localization
Genetic algorithm
Test suite (Fitness function)
Candidate space

Fault Localization - GZoltar

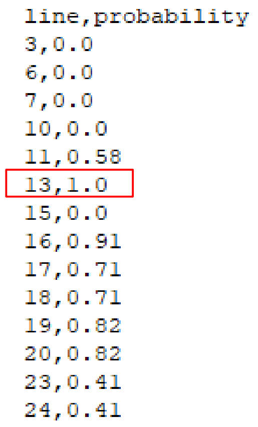


```
    public static int gcd(int a, int b) {
        int gcd = b;

        if (a == 0) {
            System.out.println("GCD: " + gcd + "\n");
        }
        while (b != 0) {
            if (a > b) {
                a = a - b;
            } else {
                gcd = b;
                b = b - a;
            }
        }
        System.out.printf("GCD: " + gcd + "\n");
        return gcd;
    }
```

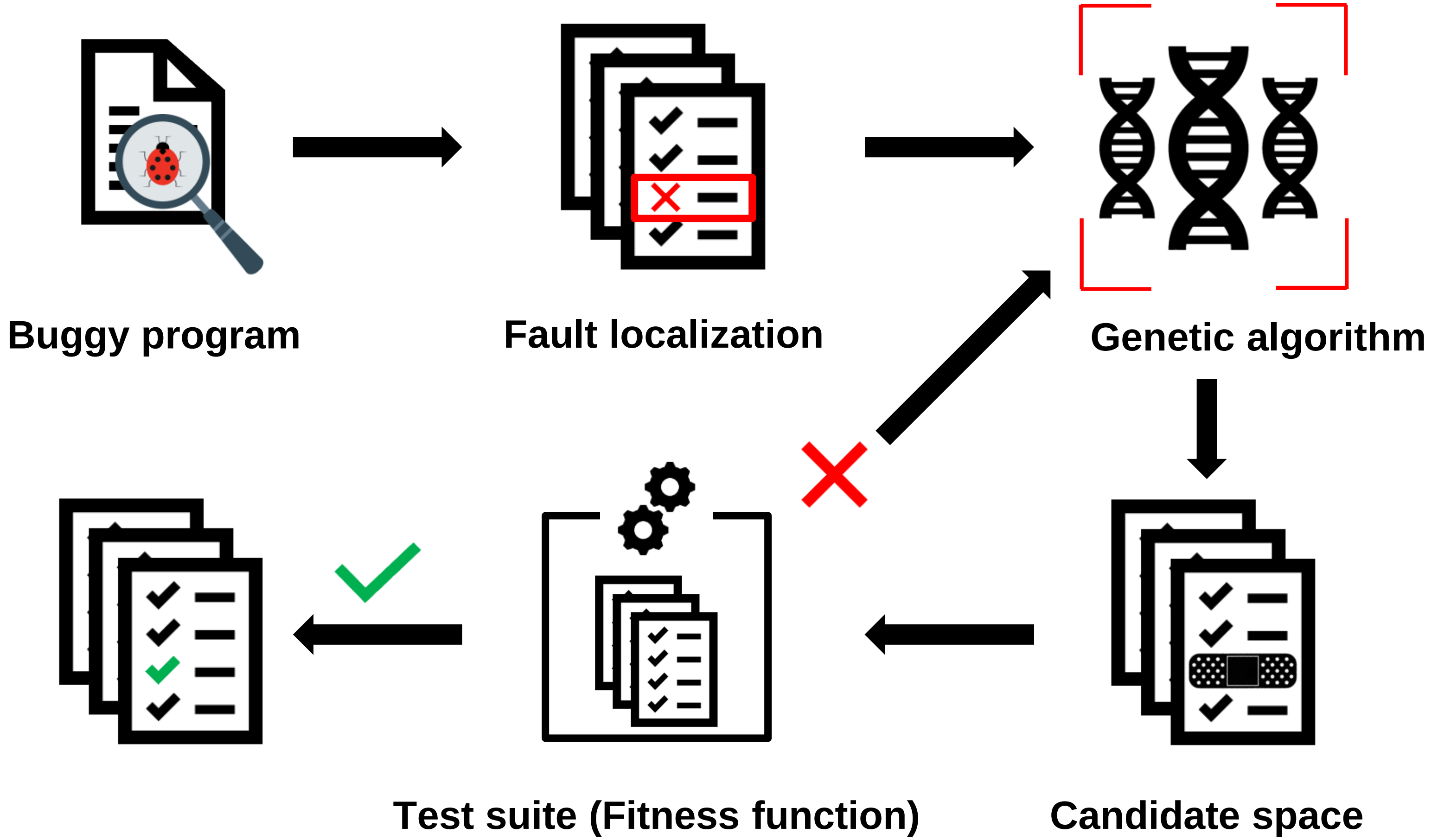
Fault Localization - GZoltar



```
Component,OCHIAI
main.java.gcd[GCD.java<main.java.gcd.GCD{<init>()V#3,0,00000000000000000000000000000000
main.java.gcd[GCD.java<main.java.gcd.GCD{main([Ljava/lang/String;)V#6,0,00000000000000000000000000000000
main.java.gcd[GCD.java<main.java.gcd.GCD{main([Ljava/lang/String;)V#7,0,00000000000000000000000000000000
main.java.gcd[GCD.java<main.java.gcd.GCD{gcd(II)I#10,0,00000000000000000000000000000000
main.java.gcd[GCD.java<main.java.gcd.GCD{gcd(II)I#11,0,57735026918962580000000000000000
main.java.gcd[GCD.java<main.java.gcd.GCD{gcd(II)I#13,1,00000000000000000000000000000000
main.java.gcd[GCD.java<main.java.gcd.GCD{gcd(II)I#15,0,00000000000000000000000000000000
main.java.gcd[GCD.java<main.java.gcd.GCD{gcd(II)I#16,0,91287092917527690000000000000000
main.java.gcd[GCD.java<main.java.gcd.GCD{gcd(II)I#17,0,70710678118654760000000000000000
main.java.gcd[GCD.java<main.java.gcd.GCD{gcd(II)I#18,0,70710678118654760000000000000000
main.java.gcd[GCD.java<main.java.gcd.GCD{gcd(II)I#19,0,81649658092772610000000000000000
main.java.gcd[GCD.java<main.java.gcd.GCD{gcd(II)I#20,0,81649658092772610000000000000000
main.java.gcd[GCD.java<main.java.gcd.GCD{gcd(II)I#23,0,40824829046386310000000000000000
main.java.gcd[GCD.java<main.java.gcd.GCD{gcd(II)I#24,0,40824829046386310000000000000000
```

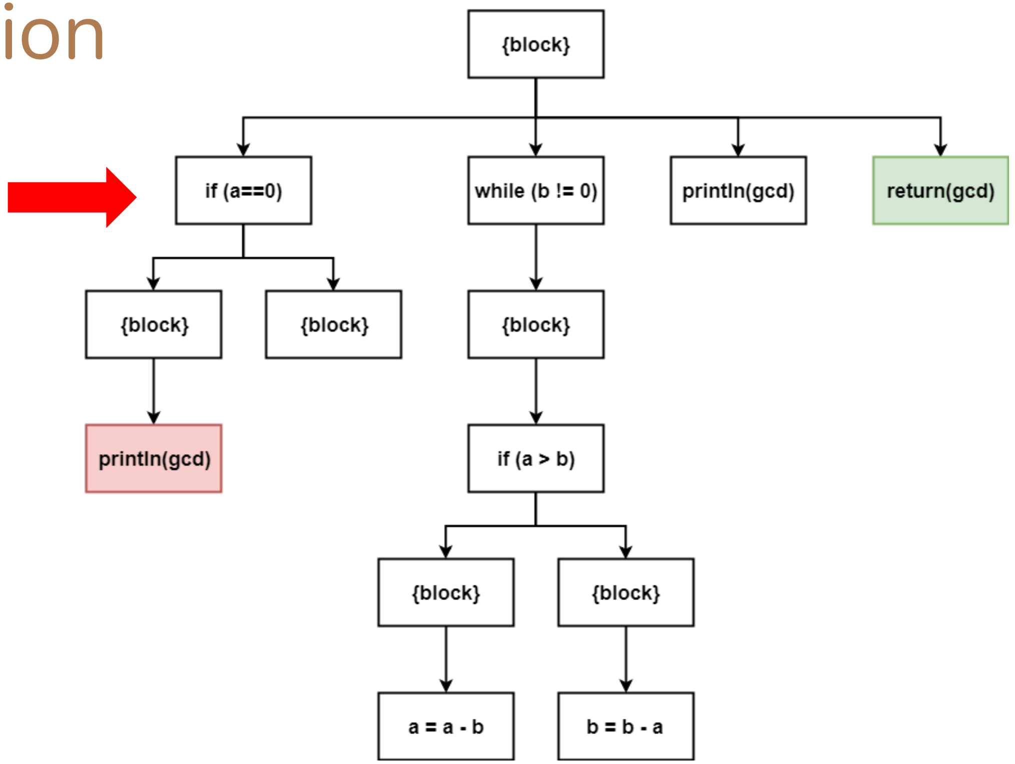


```
line,probability
3,0.0
6,0.0
7,0.0
10,0.0
11,0.58
13,1.0
15,0.0
16,0.91
17,0.71
18,0.71
19,0.82
20,0.82
23,0.41
24,0.41
```


Buggy program
Fault localization
Genetic algorithm
Test suite (Fitness function)
Candidate space

AST Representation

```
private static final List<String> ALLOWED_STATEMENTS =
new ArrayList<>(Arrays.asList("if", "while", "for", "do",
        "break", "continue", "return", "switch", "assert",
        "empty_stmt", "expr_stmt", "else"));
```

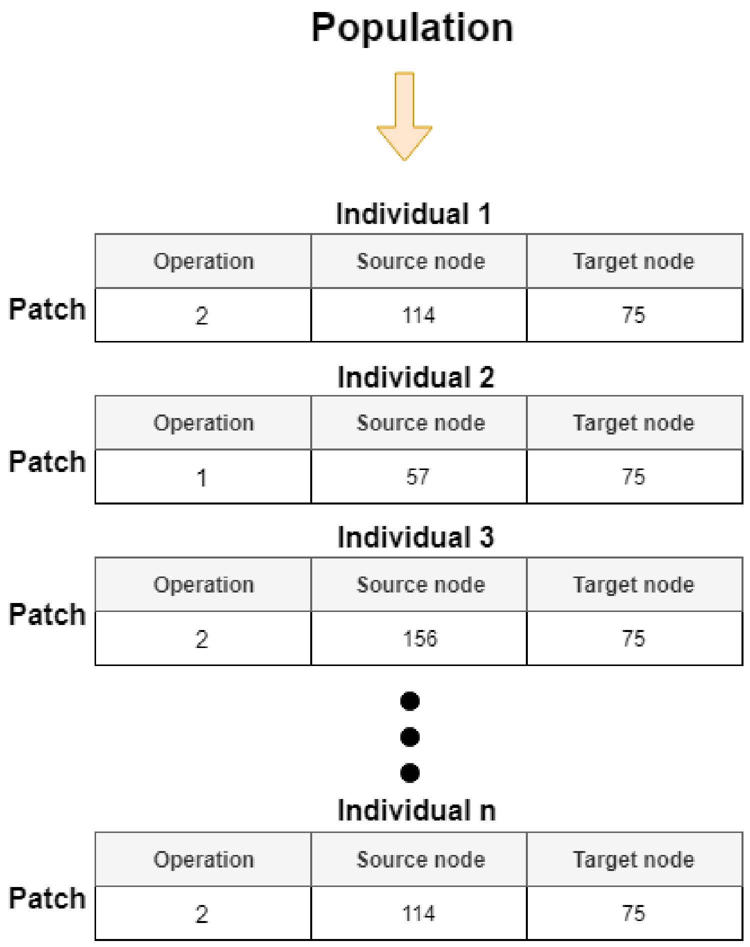


```
Fault ID : 75
Fault Line : 13
Fault Content:
System.out.println("GCD: " + gcd + "\n");
```

```
Candidate ID : 156
Candidate Line : 24
Candidate Content:
return gcd;
```

Population representation

Delete:	**0**
Replace:	**1**
Insert:	**2**



Population

Individual 1

	Operation	Source node	Target node
Patch	2	114	75

Individual 2

	Operation	Source node	Target node
Patch	1	57	75

Individual 3

	Operation	Source node	Target node
Patch	2	156	75

●
●
●

Individual n

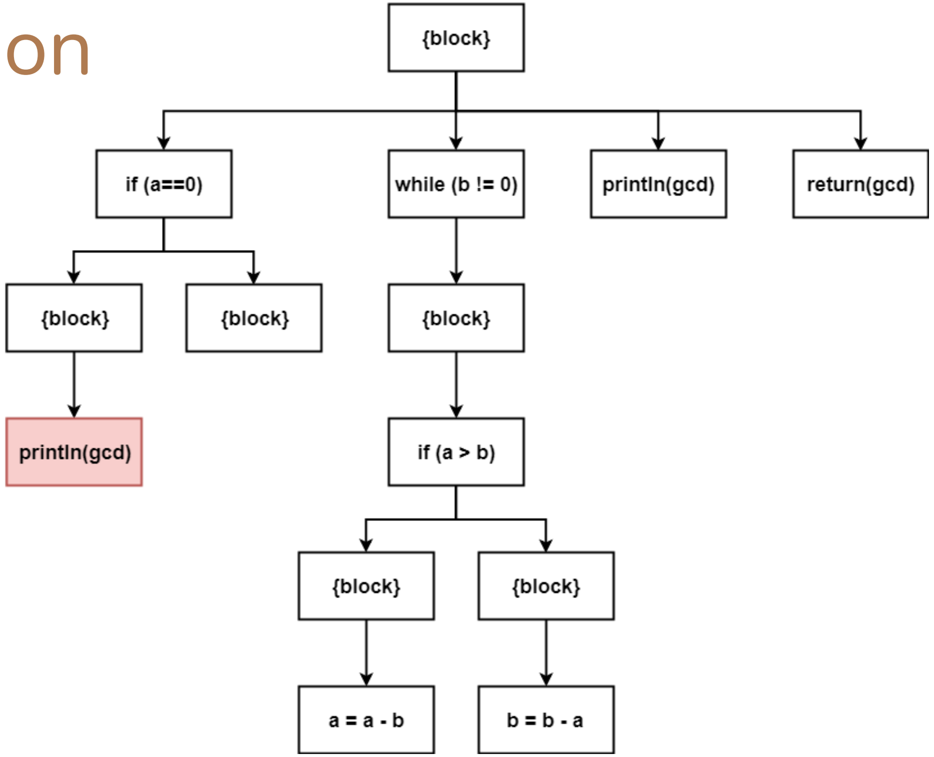
	Operation	Source node	Target node
Patch	2	114	75

AST Representation

- Available operations
  - **Delete**
  - Insert
  - Replace

```
Fault ID : 75
Fault Line : 13
Fault Content:
System.out.println("GCD: " + gcd + "\n");
```

Patch: **0** **-1**
75

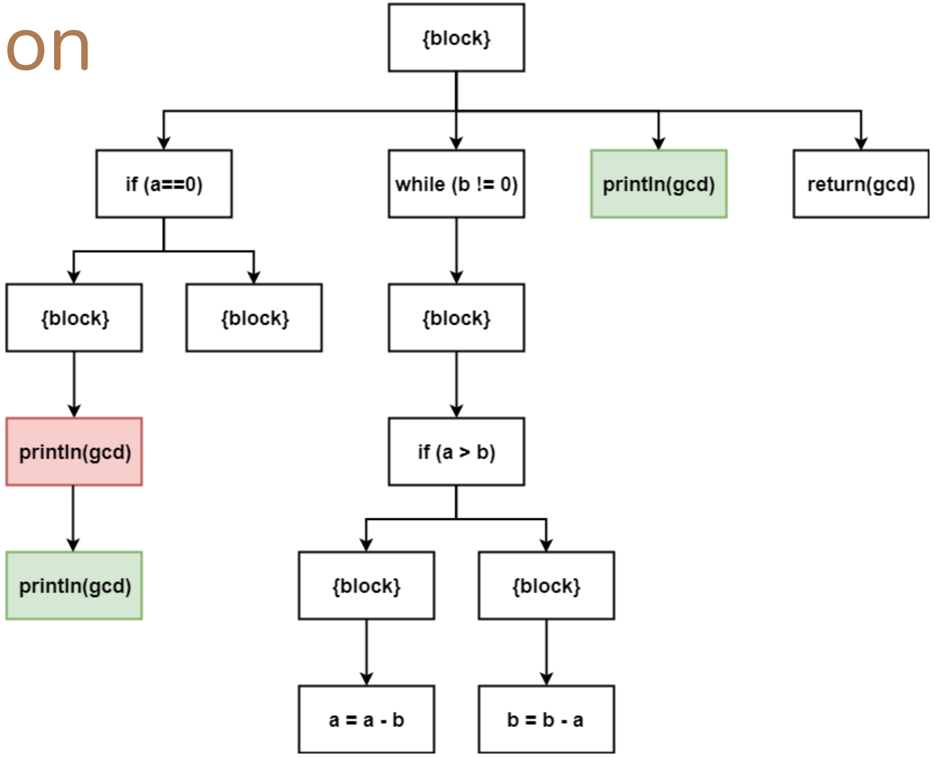


AST Representation

- Available operations
  - Delete
  - **Insert**
  - Replace

```
Candidate ID : 138
Candidate Line : 23
Candidate Content:
System.out.printf("GCD: " + gcd + "\n");
```

Patch: 2 138
75

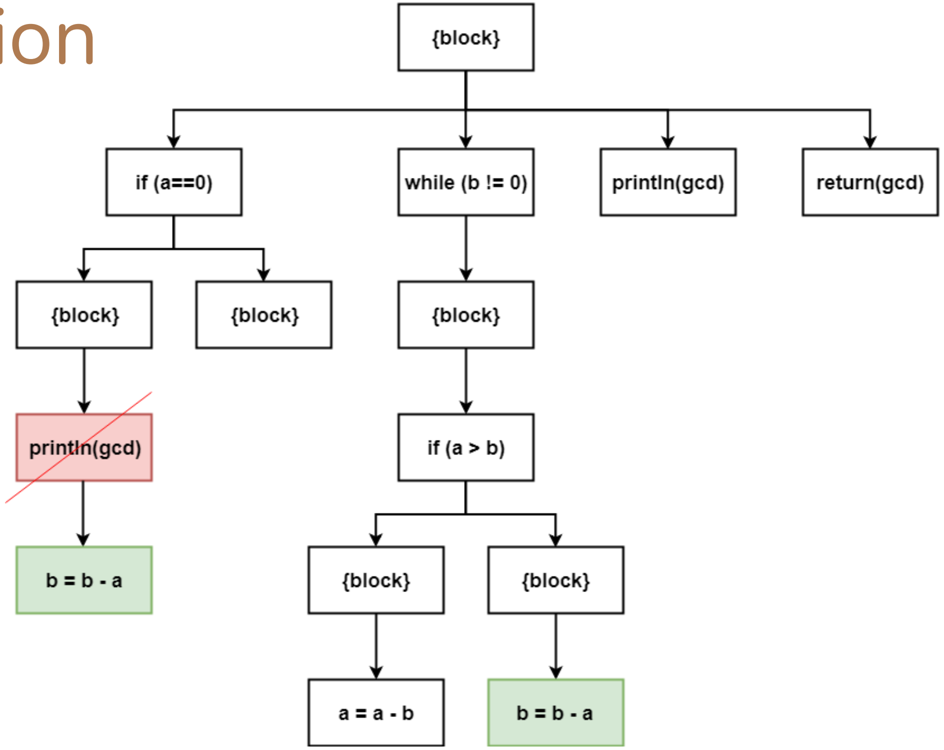


AST Representation

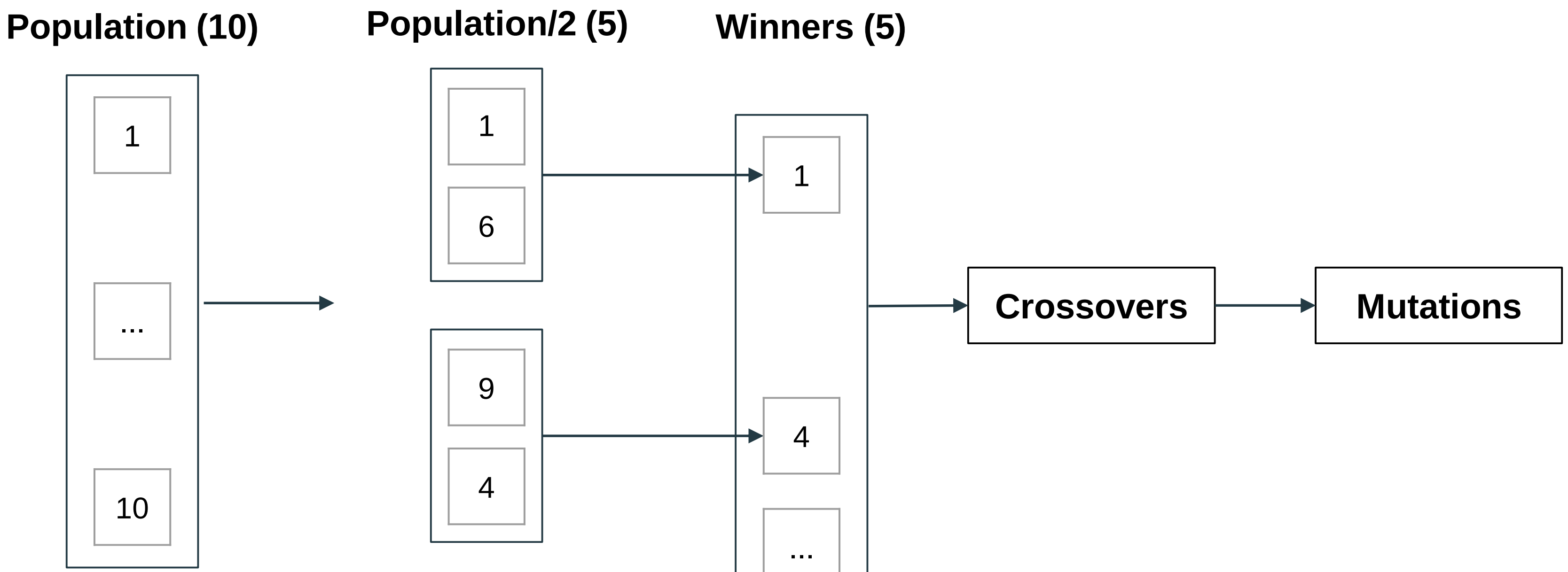
- Available operations
  - Delete
  - Insert
  - **Replace**

```
Candidate ID : 128
Candidate Line : 20
Candidate Content:
b = b - a;
```

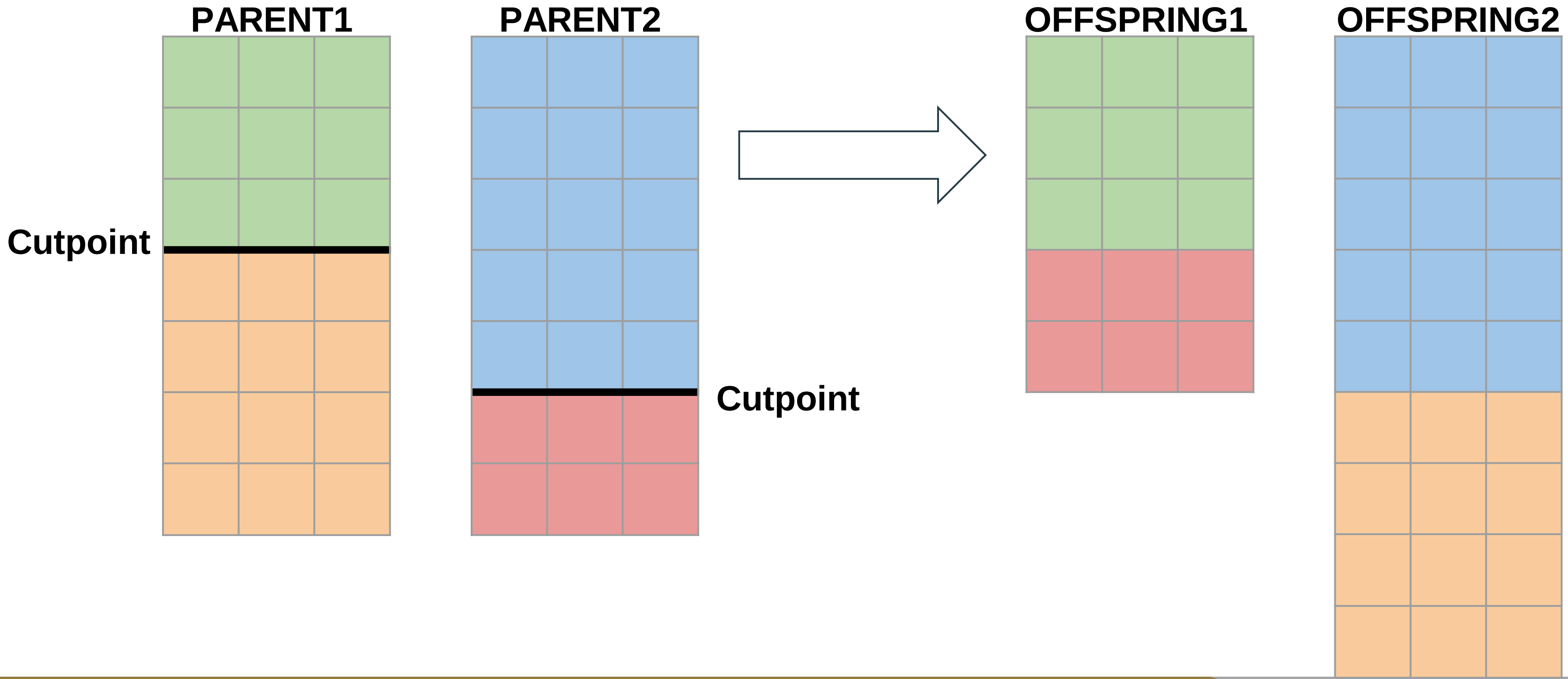
Patch: 1 75 128



Tournament Selection

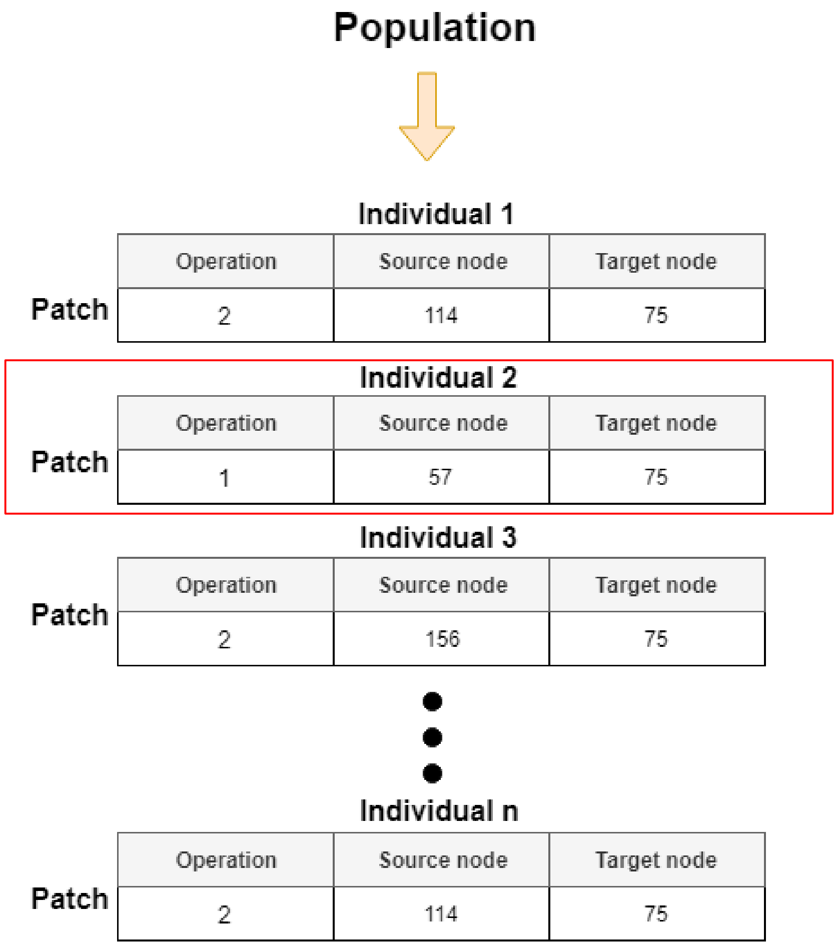


Crossover


PARENT1
PARENT2
OFFSPRING1
OFFSPRING2
Cutpoint
Cutpoint

Mutation

Delete:	**0**
Replace:	**1**
Insert:	**2**



Individual 1

Patch

Operation	Source node	Target node
2	114	75

Individual 2

Patch

Operation	Source node	Target node
1	57	75

Individual 3

Patch

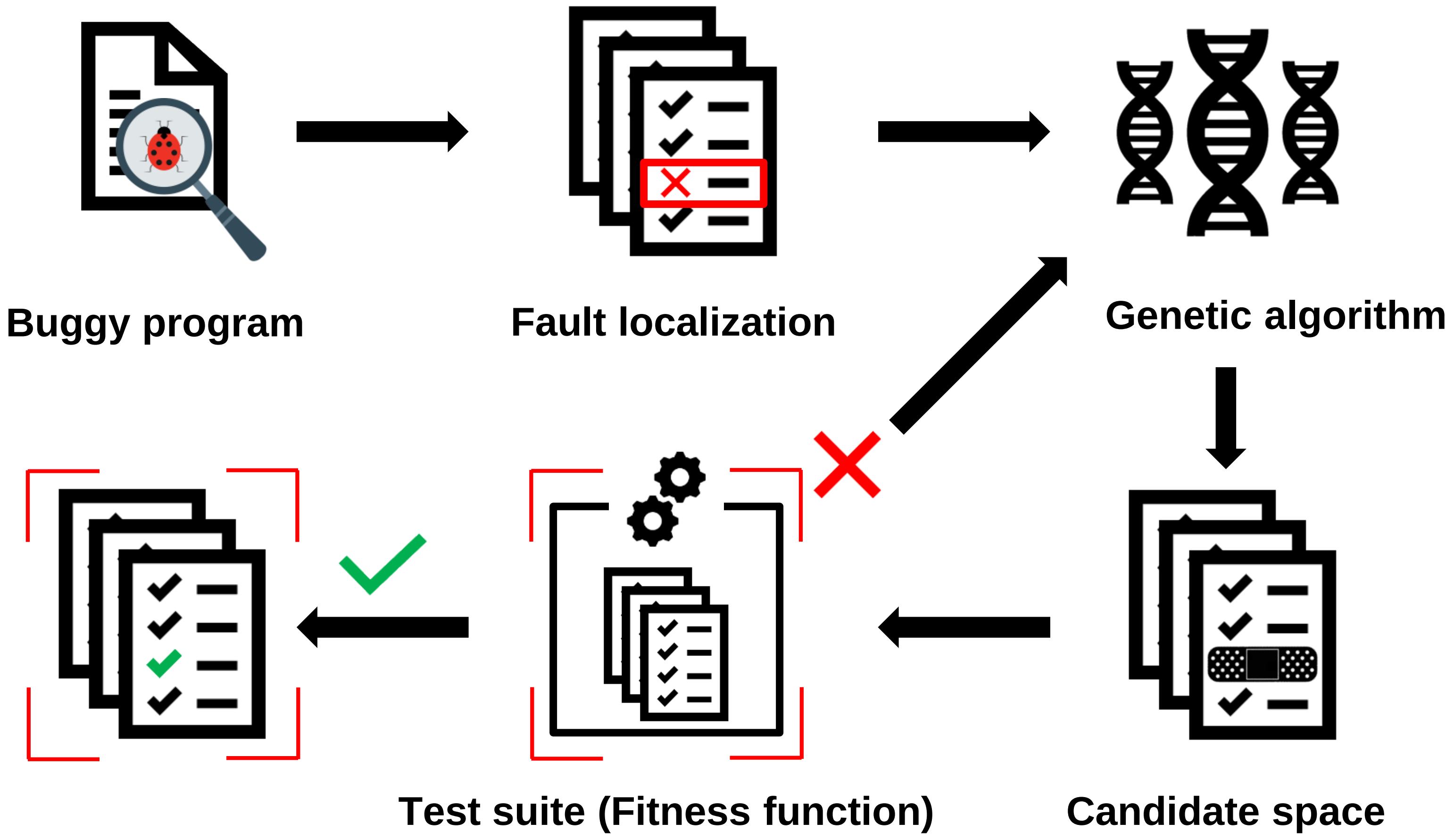
Operation	Source node	Target node
2	156	75

●
●
●

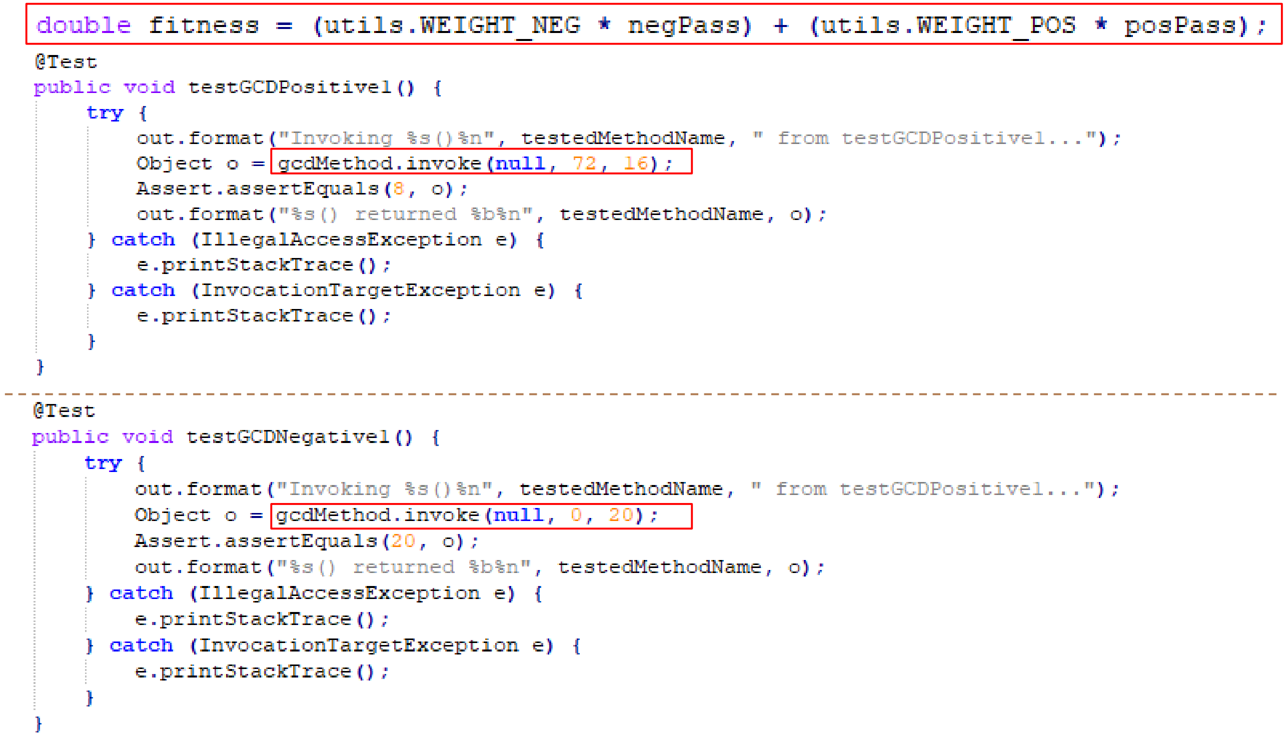
Individual n

Patch

Operation	Source node	Target node
2	114	75


Buggy program
Fault localization
Genetic algorithm
Test suite (Fitness function)
Candidate space

Test Suite & Fitness Evaluation



```java
double fitness = (utils.WEIGHT_NEG * negPass) + (utils.WEIGHT_POS * posPass);
```

```java
@Test
public void testGCDPositivel() {
    try {
        out.format("Invoking %s()%n", testedMethodName, " from testGCDPositivel...");
        Object o = gcdMethod.invoke(null, 72, 16);
        Assert.assertEquals(8, o);
        out.format("%s() returned %b%n", testedMethodName, o);
    } catch (IllegalAccessException e) {
        e.printStackTrace();
    } catch (InvocationTargetException e) {
        e.printStackTrace();
    }
}
```

```java
@Test
public void testGCDNegativel() {
    try {
        out.format("Invoking %s()%n", testedMethodName, " from testGCDPositivel...");
        Object o = gcdMethod.invoke(null, 0, 20);
        Assert.assertEquals(20, o);
        out.format("%s() returned %b%n", testedMethodName, o);
    } catch (IllegalAccessException e) {
        e.printStackTrace();
    } catch (InvocationTargetException e) {
        e.printStackTrace();
    }
}
```

Experiments

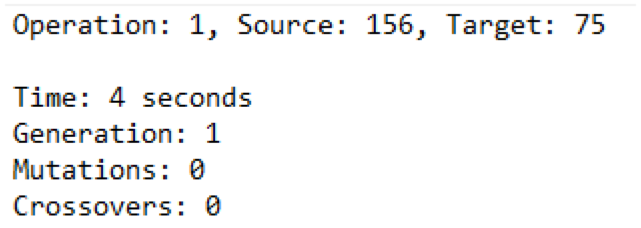
Population size: 10

Time limit: 90 mins

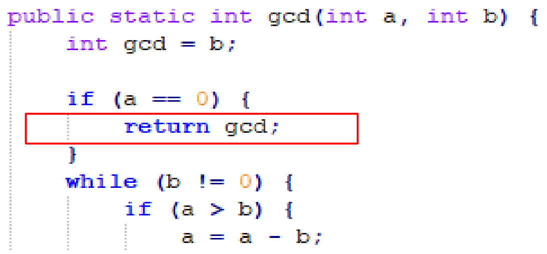
Test cases: ~1-6 positive, ~1-6 negative

Test	LOC	Time(sec)	Success
GCD	26	4-600	100%
Triangle	35	24	100%

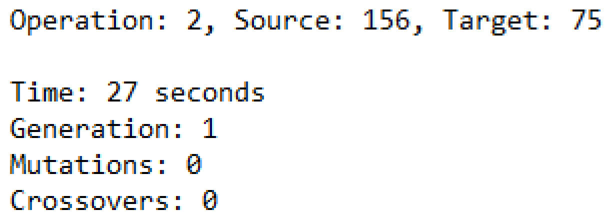
Results - GCD example



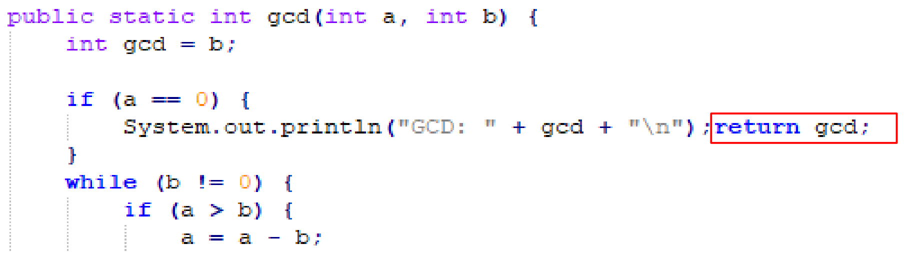
```
Operation: 1, Source: 156, Target: 75

Time: 4 seconds
Generation: 1
Mutations: 0
Crossovers: 0
```



```
public static int gcd(int a, int b) {
    int gcd = b;

    if (a == 0) {
        return gcd;
    }
    while (b != 0) {
        if (a > b) {
            a = a - b;
```



```
Operation: 2, Source: 156, Target: 75

Time: 27 seconds
Generation: 1
Mutations: 0
Crossovers: 0
```



```
public static int gcd(int a, int b) {
    int gcd = b;

    if (a == 0) {
        System.out.println("GCD: " + gcd + "\n");return gcd;
    }
    while (b != 0) {
        if (a > b) {
            a = a - b;
```

Discussion

- Pre-processing & post-processing
- Benchmark tests
- Testing time (more computational power)

Future work

- More experiments (tests, parameter tuning)
- Extend solution space
    - More node types as candidates
    - Outside the given source code
- Multiple bugs
- Context patching

Thank you for your attention!

Q/A