Kex at the 2022 SBST Tool Competition

Azat Abdullin

May 9, 2022

Overview

Kex[1]

- a platform for analysis of JVM programs with main focus on automatic test generation
- based on symbolic execution
- uses an approach called Reanimator to produce JUnit 4 test cases
- research prototype, under development

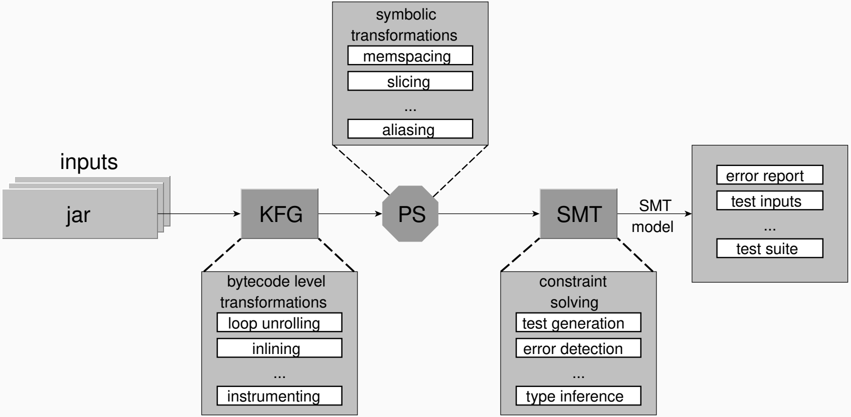
Kex-reflection[2]

- similar to Kex
- uses Java Reflection API to produce test cases

[1] https://github.com/vorpal-research/kex/tree/sbst2022

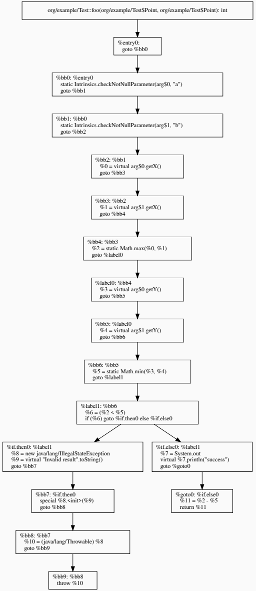
[2] https://github.com/vorpal-research/kex/tree/sbst2022-reflection

Kex overview[3]



[3] Azat M Abdullin and Vladimir Itsykson. 2022. Kex: A platform for analysis of JVM programs. Information and Control Systems 1 (2022), 30–43. http://www.ius.ru/index.php/ius/article/view/15201

Kfg: CFG for JVM bytecode



```
org/example/Test::foo(org/example/Test$Point, org/example/Test$Point): int

%entry0:
  goto %bb0

%bb0: %entry0
  static Intrinsics.checkNotNullParameter(arg$0, "a")
  goto %bb1

%bb1: %bb0
  static Intrinsics.checkNotNullParameter(arg$1, "b")
  goto %bb2

%bb2: %bb1
  %0 = virtual arg$0.getX()
  goto %bb3

%bb3: %bb2
  %1 = virtual arg$1.getX()
  goto %bb4

%bb4: %bb3
  %2 = static Math.max(%0, %1)
  goto %label0

%label0: %bb4
  %3 = virtual arg$0.getY()
  goto %bb5

%bb5: %label0
  %4 = virtual arg$1.getY()
  goto %bb6

%bb6: %bb5
  %5 = static Math.min(%3, %4)
  goto %label1

%label1: %bb6
  %6 = (%2 < %5)
  if (%6) goto %if.then0 else %if.else0

%if.then0: %label1
  %8 = new java/lang/IllegalStateException
  %9 = virtual "Invalid result".toString()
  goto %bb7

%if.else0: %label1
  %7 = System.out
  virtual %7.println("success")
  goto %goto0

%bb7: %if.then0
  special %8.<init>(%9)
  goto %bb8

%goto0: %if.else0
  %11 = %2 - %5
  return %11

%bb8: %bb7
  %10 = (java/lang/Throwable) %8
  goto %bb9

%bb9: %bb8
  throw %10
```

- instrumentation
- loop canonicalization
- loop unrolling
- etc.

Predicate state: IR for SMT-specific transformations

```
(
  @S kotlin/jvm/internal/Intrinsics.class.checkNotNullParameter(arg$0, 'a')
  @S kotlin/jvm/internal/Intrinsics.class.checkNotNullParameter(arg$1, 'b')
  @S %0 = arg$0.getX()
  @S %1 = arg$1.getX()
  @S %2 = java/lang/Math.class.max(%0, %1)
  @S %3 = arg$0.getY()
  @S %4 = arg$1.getY()
  @S %5 = java/lang/Math.class.min(%3, %4)
  @S %6 = %2 < %5
) -> (
  @P %6 = false
  @S %7 = *(java/lang/System.class.out)
  @S %7.println('success')
  @S %11 = %2 - %5
  @S <retval> = %11
)
```

- slicing
- memory spacing
- inlining
- reification

Kex-reflection: generating test cases from test inputs

Generated test inputs

```
instance:
term757 = org/example/Test {}

args:
term758 = org/example/Point {
    (x, int) = 0
    (y, int) = 2
}

term759 = org/example/Point {
    (x, int) = -2147483647
    (y, int) = 1
}
```

Test case

```
@Before
public void setup() throws Throwable {
  term1 = newInstance(Class.forName("example.Test"));
  term2 = newInstance(Class.forName("example.Point"));
  setIntField(term2, term2.getClass(), "x", 0);
  setIntField(term2, term2.getClass(), "y", 2);
  term5 = newInstance(Class.forName("example.Point"));
  setIntField(term5, term5.getClass(), "x", -2147483647);
  setIntField(term5, term5.getClass(), "y", 1);
}
@Test
public void test() throws Throwable {
  Class<?> klass = Class.forName("example.Test");
  Class<?>[] argTypes = {Class.forName("example.Point"),
    Class.forName("example.Point")};
  Object[] args = {term2, term5};
  callMethod(klass, "foo", argTypes, term1, args);
}
```

Reanimator: generating test cases from test inputs

Generated test inputs

```
instance:
term757 = org/example/Test {}

args:
term758 = org/example/Point {
    (x, int) = 0
    (y, int) = 2
}

term759 = org/example/Point {
    (x, int) = -2147483647
    (y, int) = 1
}
```

Test case

```
@Test
public void testbb8() throws Throwable {
  Test term757 = new Test();
  Point term758 = new Point(2);
  Point term759 = new Point(1);
  term759.setValue(-2147483647);
  term757.foo(term758, term759);
}
```

Kex weaknesses

- symbolic execution
    - resource intensive
- test generation
    - Reanimator
        - not always able to generate test case from inputs
        - resource intensive
    - reflection
        - produces tests that are hard to understand

Kex at the SBST 2021 competition

- score of 44.21 and fifth place
- failed to produce test cases for 5 out of 6 projects
- competition has revealed a lot of technical issues

Results

- **Kex** scored 80.94 and ranked seventh
- **Kex-reflection** scored 133.18 and ranked fifth

	Kex		**Kex-reflection**	
Project	**30 s**	**120 s**	**30 s**	**120 s**
FASTJSON	12.91	14.16	21.39	22.18
GUAVA	31.06	35.33	24.25	25.19
SEATA	22.06	29.53	18.03	22.88
SPOON	23.06	25.14	27.94	30.52
Overall average	21.55	25.13	22.79	24.96
3 project average[a]	25.39	30.00	23.41	26.20

[a] excluding the benchmarks of FastJSON project

Conclusion

- fixed the major technical issues and successfully performed on all of the projects
- improved the results in comparison with the previous year
- the tools still need to be improved to be competitive with the other participants

Contact information

azat.aam@gmail.com

https://github.com/vorpal-research/kex

POLYTECH
Peter the Great
St. Petersburg Polytechnic
University