# Kex at SBFT 2023 Tool Competition

Azat Abdullin

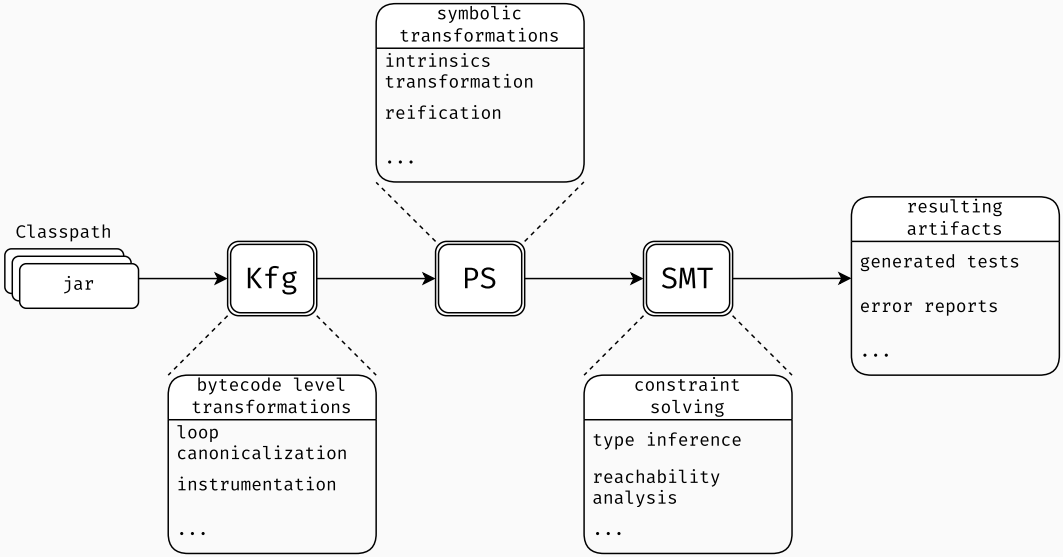
May 14, 2023

# Kex[1]

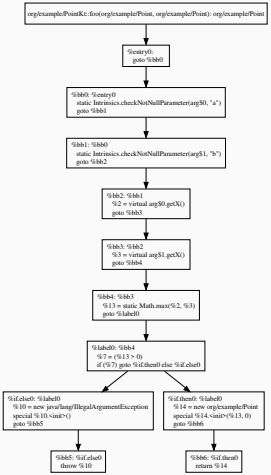
- a platform for analysis of JVM programs
  - mainly focused on automatic test generation
- based on symbolic execution
  - also has a concolic execution engine
- research prototype, under development
- third time participation in SBST/SBFT tool competition

[1] Azat 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

# Kex overiview



# Kfg[2]: CFG for JVM bytecode




- class management
- CFG in SSA form
- bytecode analysis and transformation

[2] https://github.com/vorpal-research/kfg

# Predicate state: IR for symbolic transformations

```
(
  @S kotlin/jvm/internal/Intrinsics.checkNotNullParameter(arg$0, 'a')
  @S kotlin/jvm/internal/Intrinsics.checkNotNullParameter(arg$1, 'b')
  @S term166 = *(arg$0.x)
  @S term355 = *(arg$1.x)
  @S term587 = term166 < term355
  @S term1050 = term355 > 0
  @S term1368 = new java/lang/IllegalArgumentException
  @S throw term1368
) -> (
  @P arg$0 == null = false
  @P arg$0 instanceof org/example/Point = true
  @P arg$1 == null = false
  @P arg$1 instanceof org/example/Point = true
  @P term587 = true
  @P term1050 = false
)
```

- symbolic representation of a program
- SMT-specific transformations

# Constraint solving

- PS allows support of multiple “backend” solvers
  - Z3, Boolector, CVC4, KSMT
- SBFT configuration used KSMT[3]
  - efficient asyncronos API for Z3 solver

[3] https://github.com/UnitTestBot/ksmt

# JUnit test case generation

```
public class PointKt_foo_408172348_throw_java_lang_IllegalArgumentException2 {
    @Rule public Timeout globalTimeout = new Timeout(100, TimeUnit.SECONDS);
    Object term7711;
    Object term7751;

    @Before
    public void setup() throws Throwable {
        term7711 = newInstance(Class.forName("org.example.Point"));
        setIntField(term7711, term7711.getClass(), "x", -2147483648);
        term7751 = newInstance(Class.forName("org.example.Point"));
        setIntField(term7751, term7751.getClass(), "x", -2147483647);
    }

    @Test
    public void test() throws Throwable {
        Class<?> klass = Class.forName("org.example.PointKt");
        Class<?>[] argTypes = new Class<?>[2];
        argTypes[0] = Class.forName("org.example.Point");
        argTypes[1] = Class.forName("org.example.Point");
        Object[] args = new Object[2];
        args[0] = term7711;
        args[1] = term7751;
        callMethod(klass, "foo", argTypes, null, args);
    }
};
```

# Kex-rt[4]: Java standard library approximations

- approximations for standard library of Java 8
- simplifies the semantics of standard library classes
- contains approximations for
  - collections
  - primitive type wrappers
  - string buffers
  - etc.

[4] https://github.com/AbdullinAM/kex-rt

# Kex-symbolic[5]

- traditional symbolic execution engine for automatic test generation
- traverses the CFG of PUT on a basic block level
- uses breadth-first search for path selection
    - proof-of-concept prototype

TODO maybe image

[5] https://github.com/vorpal-research/kex/releases/tag/sbft2023

# Kex-concolic[7]

- traditional concolic engine for automatic test generation
- uses Kfg instrumentation to collect symbolic state during concrete execution
- uses Easy-Random[6] library for initial seed generation
- uses context-guided search for path exploration

TODO maybe image

[6] https://github.com/j-easy/easy-random
[7] https://github.com/vorpal-research/kex/releases/tag/sbft2023

# Results

| | Kex-symbolic | | Kex-concolic | |
|---|---|---|---|---|
| **Metric** | **30 s** | **120 s** | **30 s** | **120 s** |
| Line coverage, % | 53.2 | 59.5 | 57.0 | 65.3 |
| Branch coverage, % | 38.9 | 47.5 | 35.0 | 50.0 |
| Mutation coverage, % | 0.0 | | 0.0 | |
| Test case understandability | 3.95 | | 3.69 | |
| Overall ranking | 4.89 | | 3.69 | |

# Conclusions

TODO: image

# Contact information

email:

- azat.abdullin@jetbrains.com

repository:

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