Extracting Inline Tests from Unit Tests

Yuki Liu, Pengyu Nie, Anna Guo, Milos Gligoric, Owolabi Legunsen

Inline Tests

- New granularity of tests for checking individual program statements
  - previous papers: “Inline Tests” in ASE’22, “pytest-inline” in ICSE-DEMO’23

target statement

inline tests

declare

assign

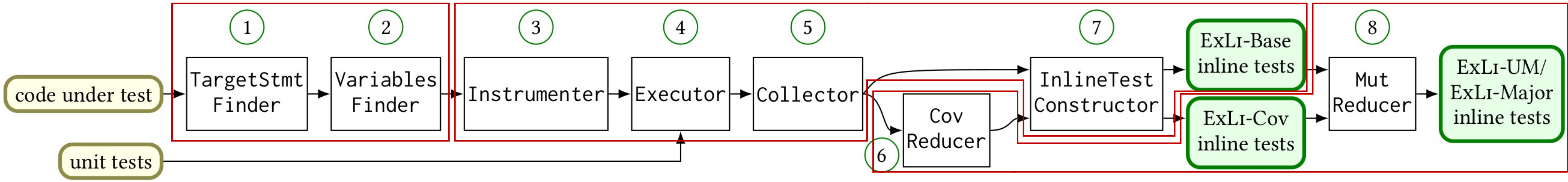
assert

```
public static final String MULTI_VALUE_DELIMITTER = ",";
public static final char EQ = '=';
public static void setAdditionalFields(String spec, GelfMsg gelfMsg) {
 if (null != spec) {
  String[] properties = spec.split(MULTI_VALUE_DELIMITTER);
  for (String field : properties) {
   final int index = field.indexOf(EQ);
   itest().given(field, "profile.requestStart.ms").given(EQ, '=').checkEq(index, -1);
   itest().given(field, " mdcName='long']").given(EQ, '=').checkEq(index, 8);
   if (-1 == index) { continue; }
   ... // add field to gelfMsg
}}}
```

- Insights: we can automatically extract inline tests from unit tests

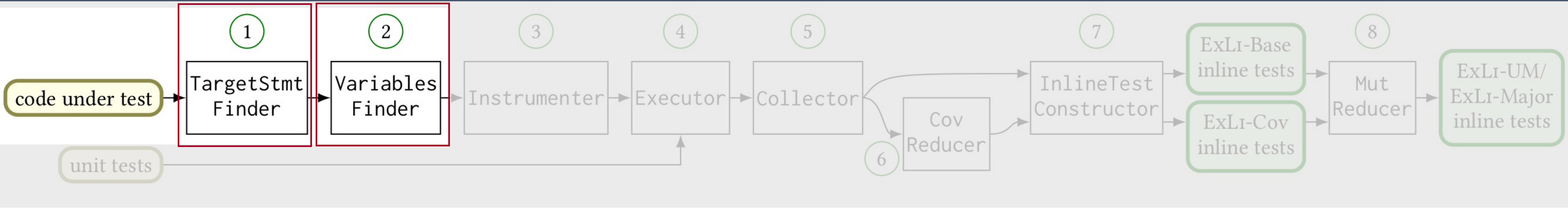
Example from project mp911de/logstash-gelf

ExLi: Extracting Inline Tests from Unit Tests



- Finding and analyzing target statements
- Generating inline tests
- Reducing inline tests using coverage-then-mutants-based algorithm

Finding and Analyzing Target Statements

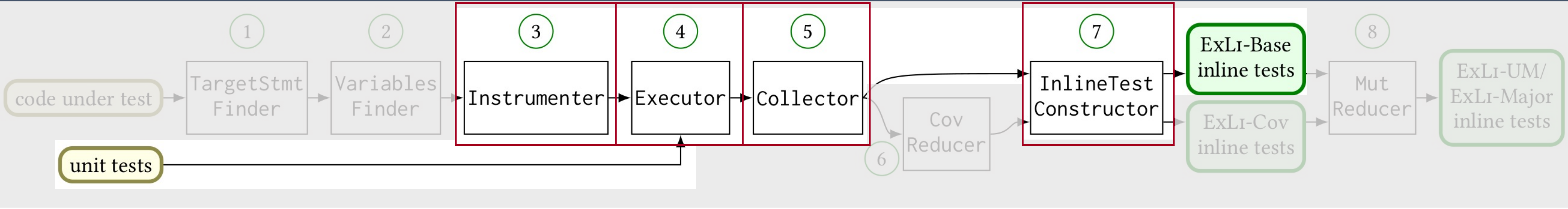


```
public static final String MULTI_VALUE_DELIMITTER = ",";
public static final char EQ = '=';
public static void setAdditionalFields(String spec, GelfMsg gelfMsg) {
 if (null != spec) {
  String[] properties = spec.split(MULTI_VALUE_DELIMITTER);
  for (String field : properties) {
   final int index = field.indexOf(EQ);
   if (-1 == index) { continue; }
   ... // add field to gelfMsg
}}}
```

input	field, EQ
output	index

- Four types of statements
  - regular expression
  - string manipulation
  - bit manipulation
  - Java streams

Generating Inline Tests



```java
public static void setAdditionalFields(String spec, GelfMsg gelfMsg) {
 if (null != spec) {
  String[] properties = spec.split(MULTI_VALUE_DELIMITTER);
  for (String field : properties) {
   try {
    collectCov();
    collectInputs(field, EQ);
    final int index = field.indexOf(EQ);
    collectOutputs(index);
    collectCov();
    if (-1 == index) { continue; }
    ... // add field to gelfMsg
   } finally { collectCov(); }
}}}
```

unit tests ▶

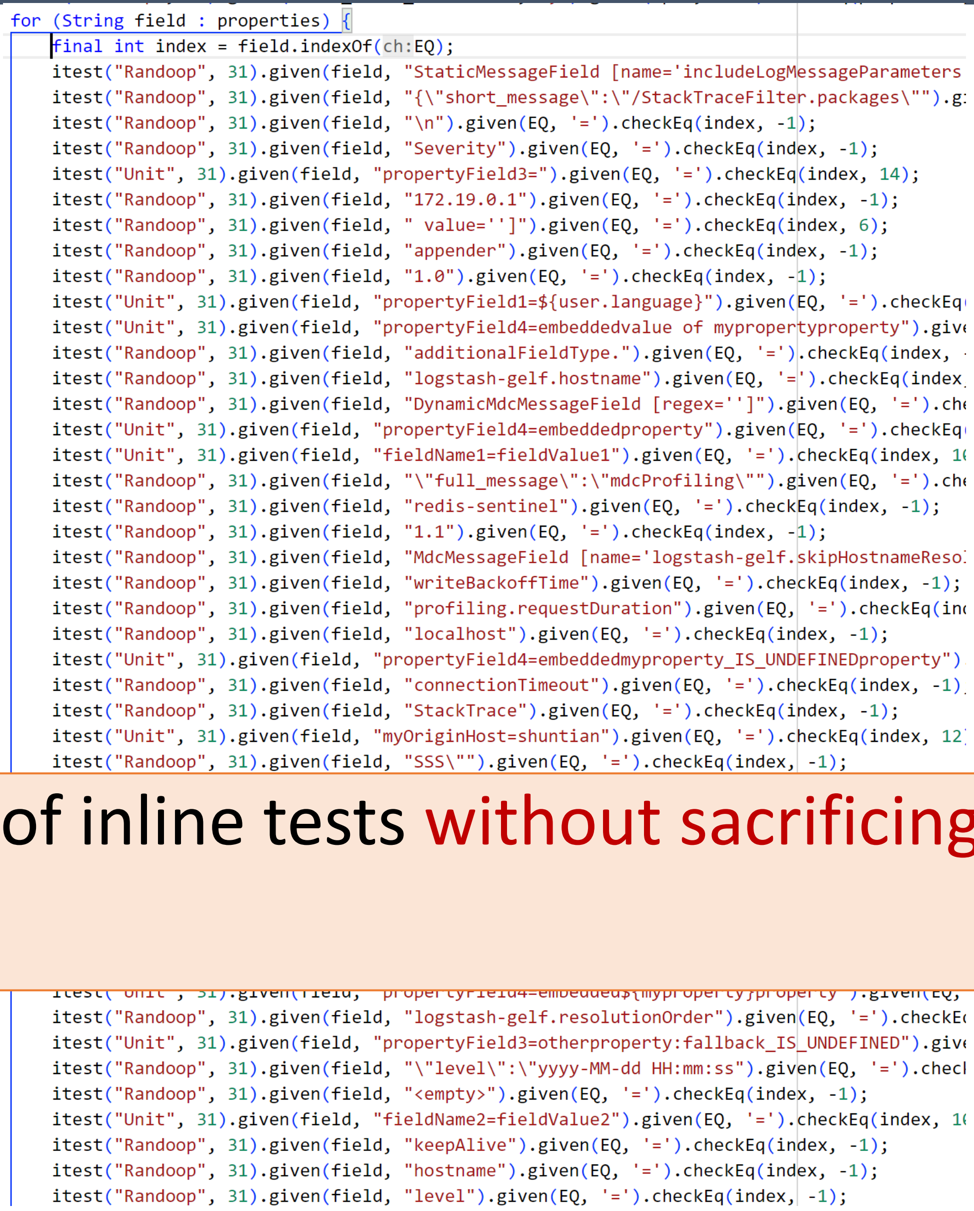
field	EQ	index
"profile.requestStart.ms"	'='	-1
" mdcName='long']"	'='	8
...	...	...

```java
itest().given(field, "profile.requestStart.ms")
       .given(EQ, '=')
       .checkEq(index, -1);
```

Too Many Inline Tests Generated

field	EQ	index
"profile.requestStart.ms"	'='	-1
" mdcName='long']"	'='	8
…	…	…

215 unique sets of values (rows) collected from unit tests

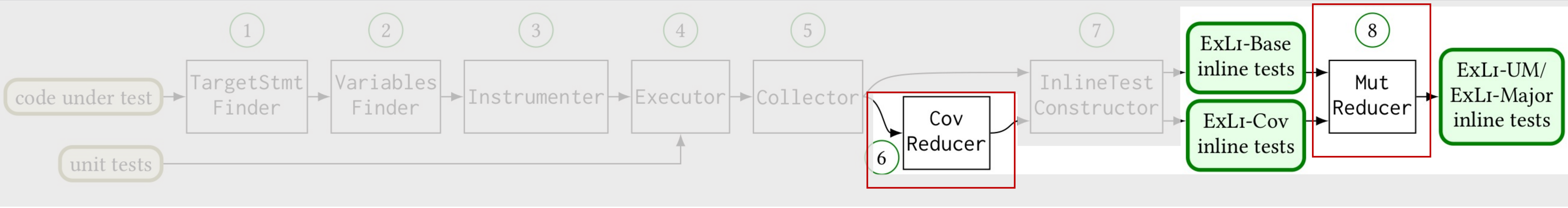


```
for (String field : properties) {
    final int index = field.indexOf(ch:EQ);
    itest("Randoop", 31).given(field, "StaticMessageField [name='includeLogMessageParameters
    itest("Randoop", 31).given(field, "{\"short_message\":\"/StackTraceFilter.packages\"").g
    itest("Randoop", 31).given(field, "\n").given(EQ, '=').checkEq(index, -1);
    itest("Randoop", 31).given(field, "Severity").given(EQ, '=').checkEq(index, -1);
    itest("Unit", 31).given(field, "propertyField3=").given(EQ, '=').checkEq(index, 14);
    itest("Randoop", 31).given(field, "172.19.0.1").given(EQ, '=').checkEq(index, -1);
    itest("Randoop", 31).given(field, " value='']").given(EQ, '=').checkEq(index, 6);
    itest("Randoop", 31).given(field, "appender").given(EQ, '=').checkEq(index, -1);
    itest("Randoop", 31).given(field, "1.0").given(EQ, '=').checkEq(index, -1);
    itest("Unit", 31).given(field, "propertyField1=${user.language}").given(EQ, '=').checkEq
    itest("Unit", 31).given(field, "propertyField4=embeddedvalue of mypropertyproperty").give
    itest("Randoop", 31).given(field, "additionalFieldType.").given(EQ, '=').checkEq(index,
    itest("Randoop", 31).given(field, "logstash-gelf.hostname").given(EQ, '=').checkEq(index
    itest("Randoop", 31).given(field, "DynamicMdcMessageField [regex='']").given(EQ, '=').che
    itest("Unit", 31).given(field, "propertyField4=embeddedproperty").given(EQ, '=').checkEq
    itest("Unit", 31).given(field, "fieldName1=fieldValue1").given(EQ, '=').checkEq(index, 1
    itest("Randoop", 31).given(field, "\"full_message\":\"mdcProfiling\"").given(EQ, '=').che
    itest("Randoop", 31).given(field, "redis-sentinel").given(EQ, '=').checkEq(index, -1);
    itest("Randoop", 31).given(field, "1.1").given(EQ, '=').checkEq(index, -1);
    itest("Randoop", 31).given(field, "MdcMessageField [name='logstash-gelf.skipHostnameReso
    itest("Randoop", 31).given(field, "writeBackoffTime").given(EQ, '=').checkEq(index, -1);
    itest("Randoop", 31).given(field, "profiling.requestDuration").given(EQ, '=').checkEq(in
    itest("Randoop", 31).given(field, "localhost").given(EQ, '=').checkEq(index, -1);
    itest("Unit", 31).given(field, "propertyField4=embeddedmyproperty_IS_UNDEFINEDproperty")
    itest("Randoop", 31).given(field, "connectionTimeout").given(EQ, '=').checkEq(index, -1)
    itest("Randoop", 31).given(field, "StackTrace").given(EQ, '=').checkEq(index, -1);
    itest("Unit", 31).given(field, "myOriginHost=shuntian").given(EQ, '=').checkEq(index, 12
    itest("Randoop", 31).given(field, "SSS\"").given(EQ, '=').checkEq(index, -1);
```

- Next step: we reduce the number of inline tests without sacrificing fault-detection capability

```
    itest("Unit", 31).given(field, "propertyField4=embedded${myproperty}property").given(EQ,
    itest("Randoop", 31).given(field, "logstash-gelf.resolutionOrder").given(EQ, '=').checkE
    itest("Unit", 31).given(field, "propertyField3=otherproperty:fallback_IS_UNDEFINED").give
    itest("Randoop", 31).given(field, "\"level\":\"yyyy-MM-dd HH:mm:ss").given(EQ, '=').chec
    itest("Randoop", 31).given(field, "<empty>").given(EQ, '=').checkEq(index, -1);
    itest("Unit", 31).given(field, "fieldName2=fieldValue2").given(EQ, '=').checkEq(index, 1
    itest("Randoop", 31).given(field, "keepAlive").given(EQ, '=').checkEq(index, -1);
    itest("Randoop", 31).given(field, "hostname").given(EQ, '=').checkEq(index, -1);
    itest("Randoop", 31).given(field, "level").given(EQ, '=').checkEq(index, -1);
```

Coverage-Then-Mutants-Based Reduction



- Reduction by coverage Powered by JACOCO Java Code Coverage
  - target coverage when executing the target statement
  - context coverage after executing the target statement before the end of its containing basic block
- Reduction by mutants universalmutator Major
  - generate mutants for the target statements
  - see paper for more details

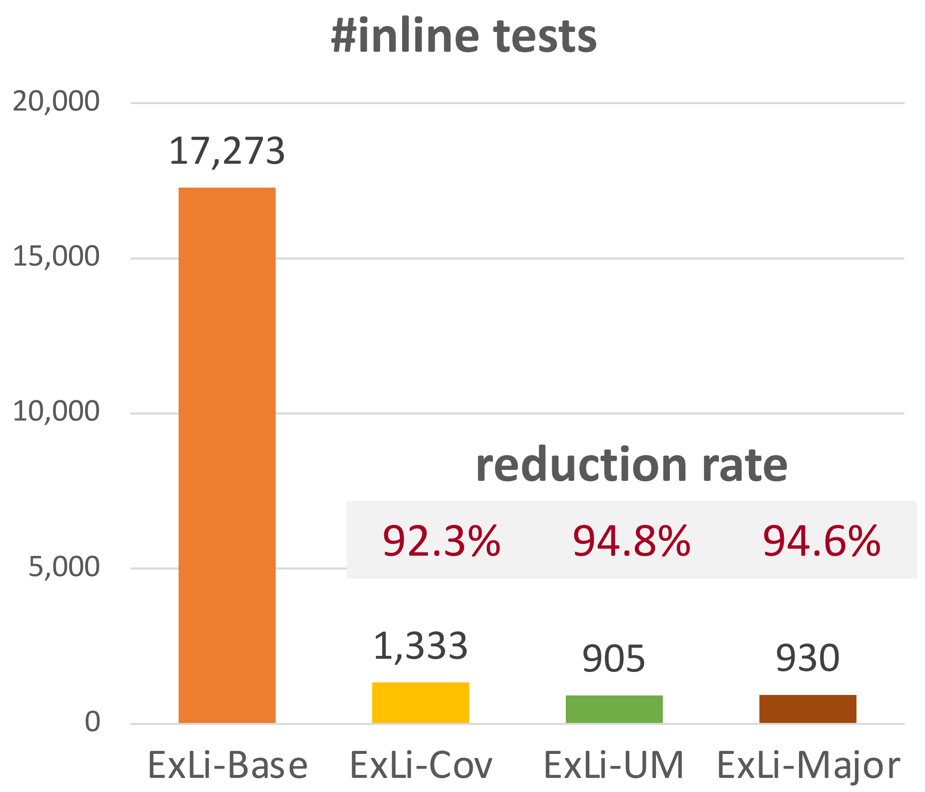
```
... try {
 collectCov();
 collectInputs(field, EQ);
 final int index = field.indexOf(EQ);
 collectOutputs(index);
 collectCov();
 if (-1 == index) { continue; }
 ... // add field to gelfMsg
 } finally { collectCov(); }
} ...
```

```
final int index = null;
```

Evaluation Setup

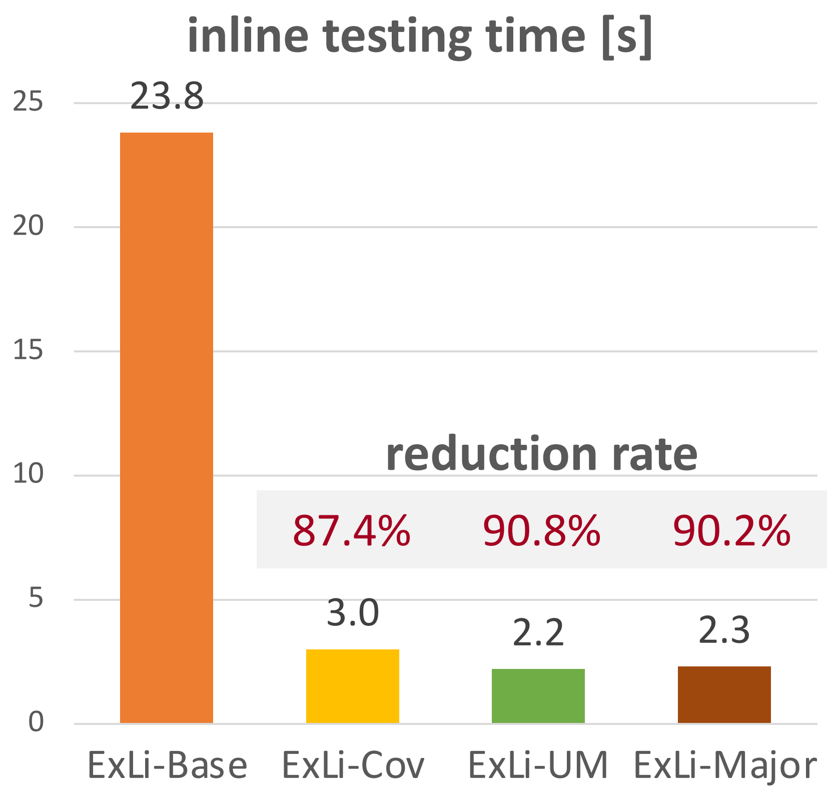
- Dataset: 31 Java projects with 423K LOC
- Extract inline tests from 237K unit tests for 718 target statements
  - 11K developer-written, 215K Randoop-generated, 11K EvoSuite-generated
- Research questions
  - RQ1: how many inline tests does ExLi generate before reduction?
  - RQ2: how many inline tests does ExLi generate after reduction?
  - RQ3: how effective are the generated inline tests in terms of fault-detection capability, compared with unit tests?
  - RQ4: what is the runtime cost of ExLi?

Results: Inline Tests

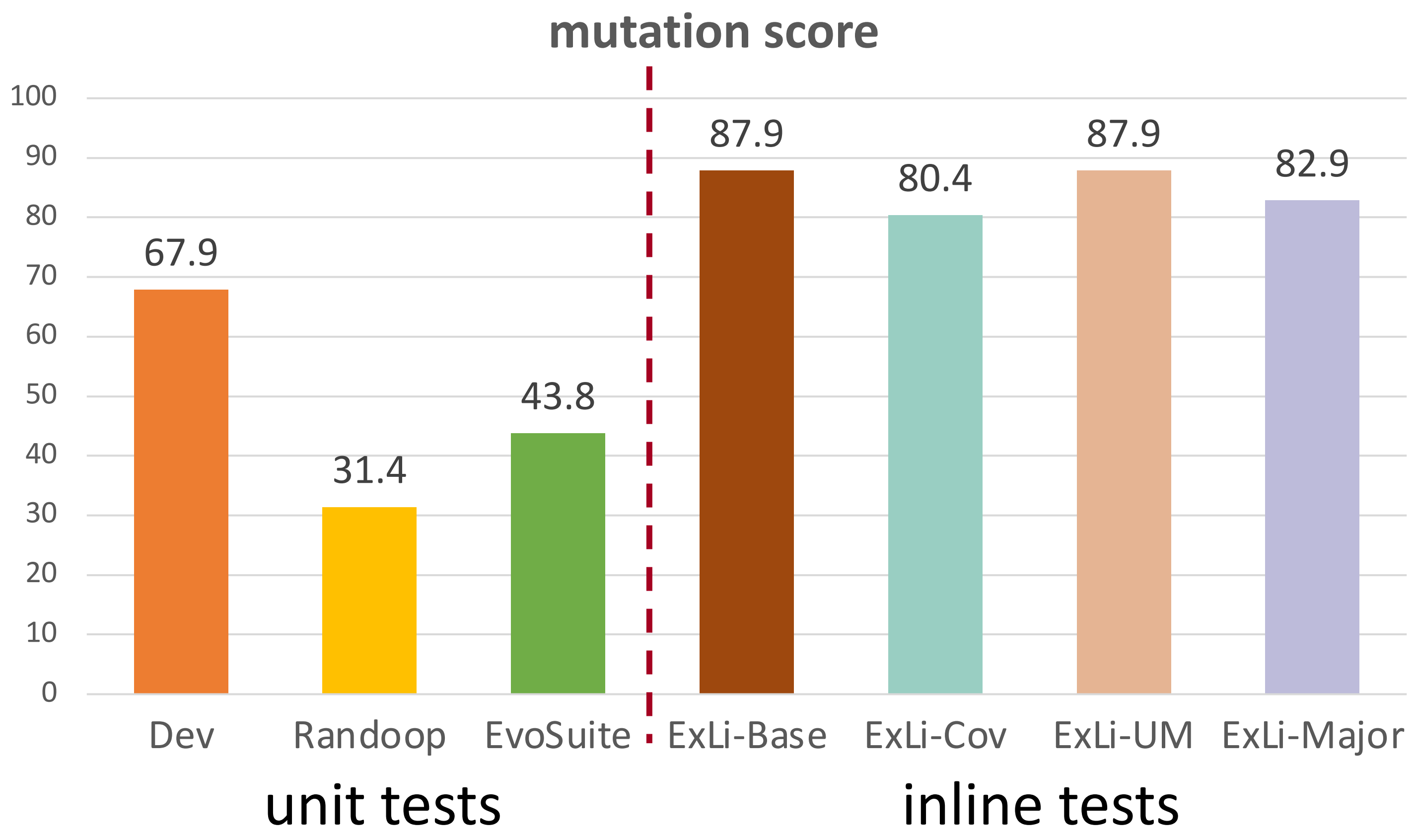


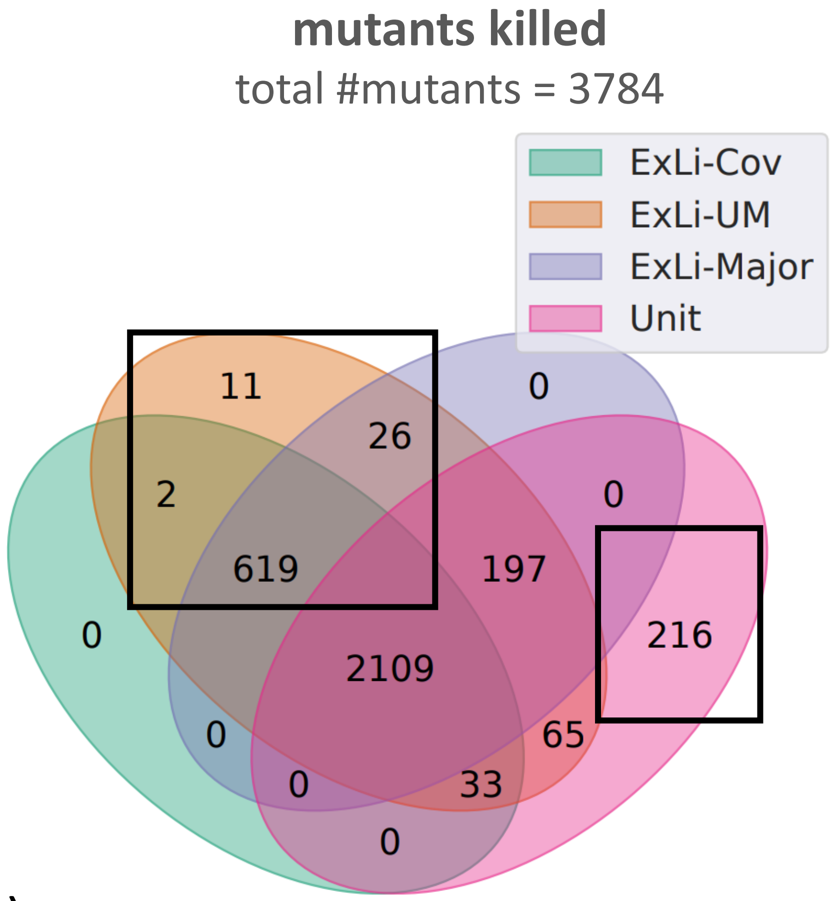
avg	24.1	1.9	1.3	1.3
median	9.0	2.0	1.0	1.0

#inline tests / statement



Results: Mutation Analysis on Target Statements





- killed by inline tests but not unit tests: 658 (20.1%)
- killed by unit tests but not inline tests: 216 (6.6%)
- unit tests and inline tests are complementary for finding faults on target statements

Conclusion

- ExLi extracts inline tests from unit tests
- Coverage-then-mutants-based reduction: 95% reduction rate
- Dataset: 905 inline tests for 718 target statements on 31 Java projects
- Mutation analysis: inline tests kills 20% more mutants on the target statements than the unit tests they were extracted from

https://github.com/EngineeringSoftware/exli

<yuki.liu@utexas.edu>