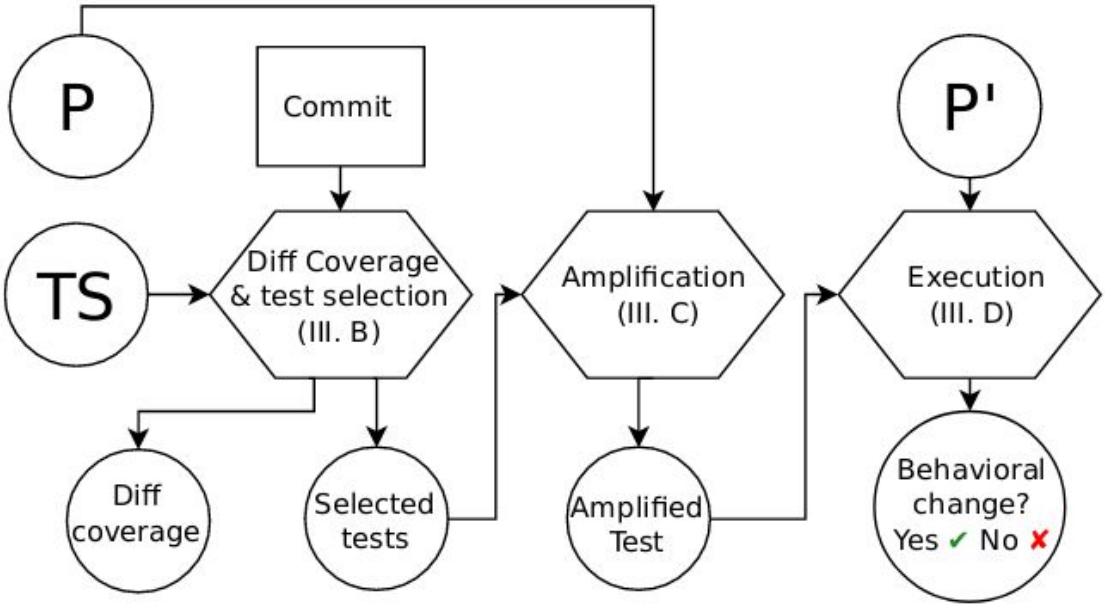
An Approach and Benchmark to Detect Behavioral Changes of Commits in Continuous Integration

Motivation

```
&& Objects.equals(getDataFormat(), ((FilterStreamType) object).getDataFormat());
&& Objects.equals(getDataFormat(), ((FilterStreamType) object).getDataFormat())
&& Objects.equals(getVersion(), ((FilterStreamType) object).getVersion());
```

Commit 7e79f77 on XWiki-Commons that changes the behavior without a test.

Review


P
Commit
P'
TS
Diff Coverage
& test selection
(III. B)
Amplification
(III. C)
Execution
(III. D)
Diff
coverage
Selected
tests
Amplified
Test
Behavioral
change?
Yes ✔ No ✘

Algorithm 1 AAMPL: Assertion amplification algorithm.

Require: Program P
Require: Test Suite TS
Ensure: An Amplified Test Suite ATS

```
1: ATS ← ∅
2: for Test in TS do
3:     NoAssertTest ← removeAssertions(Test)
4:     InstrTest ← instrument(NoAssertTest)
5:     execute(InstrTest)
6:     AmplTest ← NoAssertTest.clone()
7:     for Observ in InstrTest.observations() do
8:         Assert ← generateAssertion(Observ)
9:         AmplTest ← AmplTest.add(Assert)
10:    end for
11:    ATS.add(select(AmplTest))
12:    ATS.add(AmplTest)
13: end for
14: return ATS
```

Definition 5. *A* method execution *is a sextuple* $e = (m, S_{args}, S_{entry}, S_{exit}, S_{args'}, r)$ *where* m, S_{args}, S_{entry}, S_{exit}, $S_{args'}$, *and* r *are the method name (including the signature), the argument-object states at the method entry, the receiver-object state at the method entry, the receiver-object state at the method exit, the argument-object states at the method exit, and the method return value, respectively.*

Algorithm 1 AAMPL: Assertion amplification algorithm.

Require: Program P
Require: Test Suite TS
Ensure: An Amplified Test Suite ATS

```
1: ATS ← ∅
2: for Test in TS do
3:     NoAssertTest ← removeAssertions(Test)
4:     InstrTest ← instrument(NoAssertTest)
5:     execute(InstrTest)
6:     AmplTest ← NoAssertTest.clone()
7:     for Observ in InstrTest.observations() do
8:         Assert ← generateAssertion(Observ)
9:         AmplTest ← AmplTest.add(Assert)
10:    end for
11:    ATS.add(select(AmplTest))
12:    ATS.add(AmplTest)
13: end for
14: return ATS
```

```
testCaseGeneration(classUnderTest: Class)
1    targetsToCover ← targets(classUnderTest)
2    curPopulation ← generateRandomPopulation(popSize)
3    while targetsToCover ≠ ∅ and
                    executionTime() < maxExecutionTime
4        t ← selectTarget(targetsToCover), attempts ← 0
5        while not covered(t) and attempts < maxAttempts
6            execute test cases in curPopulation
7            update targetsToCover
8            if covered(t) break
9            compute fitness[t] for test cases in curPopulation
10           extract newPopulation from curPopulation
                    according to fitness[t]
11           mutate newPopulation
12           curPopulation ← newPopulation
13           attempts ← attempts + 1
14       end while
15   end while
```

```
<chromosome>  ::=  <actions> @ <values>
<actions>     ::=  <action> {: <actions>}?
<action>      ::=  $id = constructor ( {<parameters>}?  )
                |  $id = class # null
                |  $id . method ( {<parameters>}?  )
<parameters>  ::=  <parameter> {, <parameters>}?
<parameter>   ::=  builtin-type {<generator>}?
                |  $id
<generator>   ::=  [ low ; up ]
                |  [ genClass ]
<values>      ::=  <value> {, <values>}?
<value>       ::=  integer
                |  real
                |  boolean
                |  string
```

Figure 2: Syntax of chromosomes.

A simple example:

```
$a=A():$b=B():$b.f():$a.m(int,$b) @1,5
$a=A(int,int):$b=B():$b.g():$a.m(int,$b) @0,3,4

$a=A():$b=B():$b.f():$b.g():$a.m(int,$b)@1,4
$a=A(int,int):$b=B():$a.m(int,$b) @0,3,5
```

An example which requires the insertion and the removal of a constructor invocation:

```
$a=A():$b=B(int):[ $c=C(int) ]: $b.h($c):$b.f():$a.m(int,$b)
@1,[ 4 ],5
$a=A(int,int):$b=B():$a.m(int,$b) @0,3,6

$a=A():$b=B(int): $a.m(int,$b)@1,6
$a=A(int,int):$b=B():[ $c=C() ]: $b.h($c):$b.f():$a.m(int,$b)
@0,3,5
```

Algorithm 2 SBAMPL: Search based amplification algorithm

Require: Program P
Require: Program P'
Require: Test Suite TS
Require: Iterations number Nb
Ensure: An Amplified Test Suite ATS

```
ATS ← ∅
TMPTests ← ∅
for Test in TS do
    TMPTests ← Test
    for i ← 0, i < Nb do
        TransformedTests ← transform(TMPTests)
        AmplifiedTests ← aampl(TransformedTests)
        ATS.add(select(AmplifiedTests))
        TMPTests ← AmplifiedTests
    end for
end for
return ATS
```

Test transformations

- on numbers: ±1, replace values with Integer.MAX_VALUE or Integer.MIN_VALUE
- on booleans: negate the value
- on string literals: use another existing string literal, random replacements, null
- on methods: remove or duplicate the call

Benchmark

Considered projects properties:

1. publicly-available
2. written in Java
3. use continuous integration.

Commits

Considered:

1. the commit modifies Java files (most behavioral changes are source code changes)
2. the commit provides or modifies a manually written test that detects a behavioral change
3. The changes of the commit must be covered by the pre-commit test suite

Dropped:

1. Changes in documentation, annotation etc (xwiki-commons#6d0808f)
2. No existing tests execute the changes (xwiki-commons#bdc33d7)
3. Majority of changes is new contents (xwiki-commons#c745c10)
4. The targeted module is only tests (xwiki-commons#b0dc5bc)

Commits

project	LOC	start date	end date	#total commits	#discarded commits	#selected commits
commons-io	59607	11/19/2015	9/29/2018	385	375	10
commons-lang	77410	10/23/2017	10/9/2018	227	217	10
Gson	49766	11/26/2016	10/9/2018	159	149	10
jsoup	20088	12/21/2017	10/10/2018	50	40	10
XWiki-Commons	87289	10/30/2017	9/29/2018	689	679	10

		RQ1					RQ3	RQ2			
	id	date	#Test	#Modified Tests	+ / -	Diff Coverage	#Selected Tests	#AAMPL Tests	Time	#SBAMPL Tests	Time
commons-io	c6b8a38	6/12/18	1348	2	104 / 3	100.0	3	-	21.0s	-	324.0s
	2736b6f	12/21/17	1343	2	164 / 1	85.45	8	-	25.0s	-	0.0s
	a4705cc	4/29/18	1328	1	37 / 0	100.0	2	-	3.0s	-	23.0s
	f00d97a	5/2/17	1316	10	244 / 25	93.33	2	✓ (2)	14.0s	✓ (178)	160.0s
	3378280	4/25/17	1309	2	5 / 5	100.0	1	✓ (1)	13.0s	✓ (114)	103.0s
	703228a	12/2/16	1309	1	6 / 0	100.0	8	-	23.0s	-	0.0s
	a7bd568	9/24/16	1163	1	91 / 83	100.0	8	-	23.0s	-	0.0s
	81210eb	6/2/16	1160	1	10 / 2	100.0	1	-	6.0s	✓ (2)	46.0s
	57f493a	11/19/15	1153	1	15 / 1	100.0	8	-	6.0s	-	97.0s
	5d072ef	9/10/15	1125	12	74 / 34	83.33	25	-	2.0s	-	718.0s
	average						6.6	1.5	14.12s	98.0	147.3s
xwiki-commons	ced2635	8/13/18	1081	1	21 / 14	50.0	5	-	0.0s	-	54.23m
	10841b1	8/1/18	1061	1	107 / 19	42.86	51	-	756.0s	✓ (7)	824.73m
	848c984	7/6/18	1074	1	154 / 111	13.33	1	-	2.0s	-	2.0s
	d3101ae	1/18/18	1062	2	71 / 9	60.0	4	✓ (1)	25.0s	✓ (11)	124.37m
	a0e8b77	1/18/18	1062	2	51 / 8	85.71	4	✓ (1)	25.0s	✓ (20)	121.48m
	78ff099	12/19/17	1061	1	16 / 0	85.71	2	-	2.0s	-	82.0s
	1b79714	11/13/17	1060	1	20 / 5	100.0	22	-	22.0s	-	80.35m
	6dc9059	10/31/17	1060	1	4 / 14	62.5	22	-	22.0s	-	78.58m
	7b9ed5a	10/31/17	1060	2	21 / 3	100.0	15	-	18.0s	-	31.43m
	8f82e34	10/30/17	1059	1	137 / 115	100.0	26	-	35.0s	✓ (1)	189.0s
	average						15.2	1.0	91.08s	9.75	131.98m

```java
@Test
public void testLANG1396() {
    assertEquals("{\"Let's \\\"quote\\\" this\":\"value\"}",new ToStringBuilder(base).append("Let's \"quote\" this","value").toString());
}
```

```
ToStringBuilder o_testNestingPerson_add33752__20 =
    new ToStringBuilder(nestP).append("pid", nestP.pid).append("per/on",
Assert.assertEquals("{\"pid\":\"#1@Jane\",\"per/on\":{\"name\":\"Jane Doe
```

Fig. 5 Test generated by DCI that detects the behavioral change of 3FADFDD from commons-lang.

```
public void testNumberAsStringDeserialization() {
  Number value = gson.fromJson("\"18\"", Number.class);
  assertEquals(18, value.intValue());
}
```

Fig. 11 Provided test by the developer for 44CAD04 of Gson.

```
    json = "dhs";
    Assert.assertEquals("dhs", json);
    actual = this.gson.fromJson(json, Number.class);
    actual.longValue();
    org.junit.Assert.fail("testNumberDeserialization
} catch (JsonSyntaxException expected) {
}
```

Fig. 10 Test generated by DCI that detects the behavioral change of commit 44CAD04 in Gson.

```
public void booleanAttributeOutput() {
    Document doc = Jsoup.parse("<img src=foo noshade='' nohref async=async autofocus=false>");
    Element img = doc.selectFirst("img");

    assertEquals("<img src=\"foo\" noshade nohref async autofocus=\"false\">", img.outerHtml());
}
```

Fig. 13 Provided test by the developer for 3676B13 of Jsoup.

```
Attribute o_parsesBooleanAttributes_add8698__15 = attributes.get(1);
Assert.assertEquals("boolean=\"\"", ((BooleanAttribute) (o_parsesBooleanAttributes_add8
```

Fig. 12 Test generated by DCI_{SBAMPL} that detects the behavioral change of 3676B13 of Jsoup.

Research questions

- RQ1: What are the characteristics of commits with behavioral changes in the context of continuous integration?
- RQ2: To what extent are DCI_{AAMPL} and DCI_{SBAMPL} able to produce amplified test methods that detect the behavioral changes?
- RQ3: How do human and generated tests that detect behavioral changes differ?