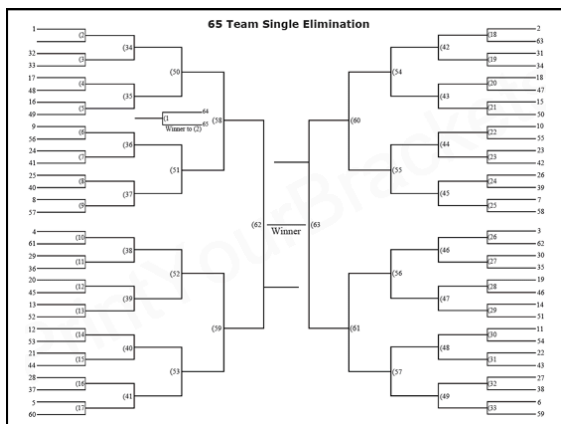


Repairing Automated Repair

Course updates

- Literature review was due today
- Project plan assignment is posted, due April 11
- Homework 3 due this Thursday, 9 AM EDT
- Homework 4 in posted, due April 6



Generalizing

- How many games are there in a 78-team bracket?
- What about an n-team bracket?

Repairing Automated Repair

Cobra effect



What do cobras have to do with automated program repair?

repairing python programs?

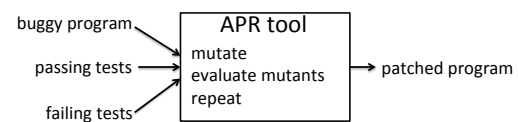
```

1 from random import randint
2
3 def rand_str(n):
4     """Return a string of n random characters"""
5     return ''.join(chr(randint(97, 122)) for _ in range(n))
6
7 def rand_int(n):
8     """Return a random integer between 0 and n-1"""
9     return randint(0, n-1)
10
11 def rand_float():
12     """Return a random float between 0.0 and 1.0"""
13     return randint(0, 1000000000) / 1000000000.0
14
15 def rand_bool():
16     """Return a random boolean"""
17     return randint(0, 1) == 0
18
19 def rand_list(n):
20     """Return a list of n random elements"""
21     return [rand_str(5) if rand_bool() else rand_int(100) for _ in range(n)]
22
23 def rand_dict(n):
24     """Return a dictionary with n random key-value pairs"""
25     return {rand_str(5): rand_int(100) for _ in range(n)}
26
27 def rand_tuple(n):
28     """Return a tuple of n random elements"""
29     return tuple(rand_str(5) if rand_bool() else rand_int(100) for _ in range(n))
30
31 def rand_set(n):
32     """Return a set of n random elements"""
33     return set(rand_str(5) if rand_bool() else rand_int(100) for _ in range(n))
34
35 def rand_file():
36     """Return a file object"""
37     return open(rand_str(5) + '.txt', 'w')
38
39 def rand_dir():
40     """Return a directory object"""
41     return os.mkdir(rand_str(5))
42
43 def rand_module():
44     """Return a module object"""
45     return __import__(rand_str(5))
46
47 def rand_class():
48     """Return a class object"""
49     return type(rand_str(5), (), {})
50
51 def rand_function():
52     """Return a function object"""
53     def f():
54         pass
55     return f
56
57 def rand_variable():
58     """Return a variable object"""
59     return rand_str(5)
60
61 def rand_attribute():
62     """Return an attribute object"""
63     return rand_str(5)
64
65 def rand_method():
66     """Return a method object"""
67     return rand_str(5)
68
69 def rand_property():
70     """Return a property object"""
71     return rand_str(5)
72
73 def rand_exception():
74     """Return an exception object"""
75     return rand_str(5)
76
77 def rand_context_manager():
78     """Return a context manager object"""
79     return rand_str(5)
80
81 def rand_iterator():
82     """Return an iterator object"""
83     return rand_str(5)
84
85 def rand_async_iterator():
86     """Return an async iterator object"""
87     return rand_str(5)
88
89 def rand_coroutine():
90     """Return a coroutine object"""
91     return rand_str(5)
92
93 def rand_task():
94     """Return a task object"""
95     return rand_str(5)
96
97 def rand_future():
98     """Return a future object"""
99     return rand_str(5)
100
101 def rand_event():
102     """Return an event object"""
103     return rand_str(5)
104
105 def rand_lock():
106     """Return a lock object"""
107     return rand_str(5)
108
109 def rand_semaphore():
110     """Return a semaphore object"""
111     return rand_str(5)
112
113 def rand_bounded_semaphore():
114     """Return a bounded semaphore object"""
115     return rand_str(5)
116
117 def rand_condition():
118     """Return a condition object"""
119     return rand_str(5)
120
121 def rand_async_condition():
122     """Return an async condition object"""
123     return rand_str(5)
124
125 def rand_timeout_error():
126     """Return a timeout error object"""
127     return rand_str(5)
128
129 def rand_keyboard_interrupt():
130     """Return a keyboard interrupt object"""
131     return rand_str(5)
132
133 def rand_system_exit():
134     """Return a system exit object"""
135     return rand_str(5)
136
137 def rand_os_error():
138     """Return an os error object"""
139     return rand_str(5)
140
141 def rand_file_error():
142     """Return a file error object"""
143     return rand_str(5)
144
145 def rand_io_error():
146     """Return an io error object"""
147     return rand_str(5)
148
149 def rand_value_error():
150     """Return a value error object"""
151     return rand_str(5)
152
153 def rand_type_error():
154     """Return a type error object"""
155     return rand_str(5)
156
157 def rand_index_error():
158     """Return an index error object"""
159     return rand_str(5)
160
161 def rand_key_error():
162     """Return a key error object"""
163     return rand_str(5)
164
165 def rand_attribute_error():
166     """Return an attribute error object"""
167     return rand_str(5)
168
169 def rand_name_error():
170     """Return a name error object"""
171     return rand_str(5)
172
173 def rand_unbound_local_error():
174     """Return an unbound local error object"""
175     return rand_str(5)
176
177 def rand_module_not_found_error():
178     """Return a module not found error object"""
179     return rand_str(5)
180
181 def rand_import_error():
182     """Return an import error object"""
183     return rand_str(5)
184
185 def rand_syntax_error():
186     """Return a syntax error object"""
187     return rand_str(5)
188
189 def rand_indented_block():
190     """Return an indented block object"""
191     return rand_str(5)
192
193 def rand_def_statement():
194     """Return a def statement object"""
195     return rand_str(5)
196
197 def rand_if_statement():
198     """Return an if statement object"""
199     return rand_str(5)
200
201 def rand_for_statement():
202     """Return a for statement object"""
203     return rand_str(5)
204
205 def rand_while_statement():
206     """Return a while statement object"""
207     return rand_str(5)
208
209 def rand_with_statement():
210     """Return a with statement object"""
211     return rand_str(5)
212
213 def rand_return_statement():
214     """Return a return statement object"""
215     return rand_str(5)
216
217 def rand_continue_statement():
218     """Return a continue statement object"""
219     return rand_str(5)
220
221 def rand_break_statement():
222     """Return a break statement object"""
223     return rand_str(5)
224
225 def rand_pass_statement():
226     """Return a pass statement object"""
227     return rand_str(5)
228
229 def rand_delete_statement():
230     """Return a delete statement object"""
231     return rand_str(5)
232
233 def rand_augmented_assignment():
234     """Return an augmented assignment object"""
235     return rand_str(5)
236
237 def rand_assignment():
238     """Return an assignment object"""
239     return rand_str(5)
240
241 def rand_expression():
242     """Return an expression object"""
243     return rand_str(5)
244
245 def rand_statement():
246     """Return a statement object"""
247     return rand_str(5)
248
249 def rand_block():
250     """Return a block object"""
251     return rand_str(5)
252
253 def rand_suite():
254     """Return a suite object"""
255     return rand_str(5)
256
257 def rand_module():
258     """Return a module object"""
259     return rand_str(5)
260
261 def rand_package():
262     """Return a package object"""
263     return rand_str(5)
264
265 def rand_class():
266     """Return a class object"""
267     return rand_str(5)
268
269 def rand_function():
270     """Return a function object"""
271     return rand_str(5)
272
273 def rand_method():
274     """Return a method object"""
275     return rand_str(5)
276
277 def rand_property():
278     """Return a property object"""
279     return rand_str(5)
280
281 def rand_exception():
282     """Return an exception object"""
283     return rand_str(5)
284
285 def rand_context_manager():
286     """Return a context manager object"""
287     return rand_str(5)
288
289 def rand_iterator():
290     """Return an iterator object"""
291     return rand_str(5)
292
293 def rand_async_iterator():
294     """Return an async iterator object"""
295     return rand_str(5)
296
297 def rand_coroutine():
298     """Return a coroutine object"""
299     return rand_str(5)
300
301 def rand_task():
302     """Return a task object"""
303     return rand_str(5)
304
305 def rand_future():
306     """Return a future object"""
307     return rand_str(5)
308
309 def rand_event():
310     """Return an event object"""
311     return rand_str(5)
312
313 def rand_lock():
314     """Return a lock object"""
315     return rand_str(5)
316
317 def rand_semaphore():
318     """Return a semaphore object"""
319     return rand_str(5)
320
318

```

Automated Program Repair

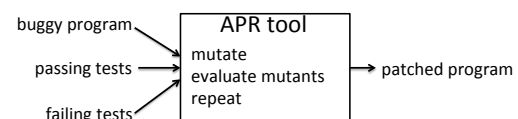
basic idea:



the many repair tools

ClearView [Perkins et al. 2009] GenProg [Weimer et al. 2009]
 Prophet [Long and Rinard 2015] SPR [Long and Rinard 2015]
 TDS [Perelman et al. 2014]
 Par [Kim et al. 2013] AE [Weimer et al. 2013]
 SemFix [Nguyen et al. 2013] AutoFix-E [Wei et al. 2010]
 [Carzaniga et al. 2010] [Carzaniga et al. 2013]
 [Jin et al. 2011] Coker and Hafiz et al. 2013]
 [Debroy and Wong et al. 2010] [Lin and Ernst et al. 2004]
 [Forrest et al. 2009] [Novark et al. 2007] [Demsky et al. 2006]

Potential problem



the patched program may pass all given tests,
 but break other functionality

COMPUTE THE MEDIAN OF THREE NUMBERS

```
int median(int a, int b, int c) {
    int result;
    if ((b<=a && a<=c) ||
        (c<=a && a<=b))
        result = a;
    if ((a<b && b <= c) ||
        (c<=b && b<a))
        result = b;
    if ((a<c && c<b) ||
        (b<c && c<a))
        result = c;
    return result;
}
```

```
int median(int a, int b, int c) {
    int result = 0;
    if ((b<=a && a<=c) ||
        (c<=a && a<=b))
        result = a;
    if ((a<b && b <= c) ||
        (c<=b && b<a))
        result = b;
    if ((a<c && c<b) ||
        (b<c && c<a))
        result = c;
    return result;
}
```

```
int median(int a, int b, int c) {
    int result = 0;
    if ((b<=a && a<=c) ||
        (c<=a && a<=b))
        result = a;
    if ((a<b && b <= c) ||
        (c<=b && b<a))
        result = b;
    if ((a<c && c<b) ||
        (b<c && c<a))
        result = c;
    return result;
}
```

```
int median(int a, int b, int c) {
    int result = 0;
    if ((b<=a && a<=c) ||
        (c<=a && a<=b))
        result = a;
    if ((a<b && b <= c) ||
        (c<=b && b<a))
        result = b;
    if ((a<c && c<b) ||
        (b<c && c<a))
        result = c;
    return result;
}
```

```
int median(int a, int b, int c) {
    int result = 0;
    if ((b<=a && a<=c) ||
        (c<=a && a<=b))
        result = a;
    if ((a<b && b <= c) ||
        (c<=b && b<a))
        result = b;
    if ((a<c && c<b) ||
        (b<c && c<a))
        result = c;
    return result;
}
```

```
int median(int a, int b, int c) {
    int result = 0;
    if ((b<=a && a<=c) ||
        (c<=a && a<=b))
        result = a;
    if ((a<b && b <= c) ||
        (c<=b && b<a))
        result = b;
    if ((a<c && c<b) ||
        (b<c && c<a))
        result = c;
    return result;
}
```

```
int median(int a, int b, int c) {
    int result = 0;
    if ((b<=a && a<=c) ||
        (c<=a && a<=b))
        result = a;
    if ((a<b && b <= c) ||
        (c<=b && b<a))
        result = b;
    if ((a<c && c<b) ||
        (b<c && c<a))
        result = c;
    return result;
}
```

```
int median(int a, int b, int c) {
    int result = 0;
    ((b<=a && a<=c) ||
     (c<=a && a<=b))
    result = a;
    ((a<b && b <= c) ||
     (c<=b && b<a))
    result = b;
    ((a<c && c<b) ||
     (b<c && c<a))
    result = c;
    return result;
}
```

```
int med_broken(int a, int b, int c) {
    int result;
    if ((a==b) || (a==c) ||
        (b<a && a<c) ||
        (c<a && a<b))
        result = a;
    else if ((b==c) || (a<b && b<c) ||
        (c<b && b<a))
        result = b;
    else if (a<c && c<b)
        result = c;
    return result;
}
```

```
int med_broken(int a, int b, int c) {
    int result;
    if ((a==b) || (a==c) ||
        (b<a && a<c) ||
        (c<a && a<b))
        result = a;
    else if ((b==c) || (a<b && b<c) ||
        (c<b && b<a))
        result = b;
    else if (a<c && c<b)
        result = c;
    return result;
}
```

```
int med_broken(int a, int b, int c) {
    int result;
    if ((a==b) || (a==c) ||
        (b<a && a<c) ||
        (c<a && a<b))
        result = a;
    else if ((b==c) || (a<b && b<c) ||
        (c<b && b<a))
        result = b;
    else if (a<c && c<b)
        result = c;
    return result;
}
```

```
int med_broken(int a, int b, int c) {
    int result;
    if ((a==b) || (a==c) ||
        (b<a && a<c) ||
        (c<a && a<b))
        result = a;
    else if ((b==c) || (a<b && b<c) ||
        (c<b && b<a))
        result = b;
    else if (a<c && c<b)
        result = c;
    return result;
}
```

Input	Expected	Pass?
0,0,0	0	✓
2,0,1	1	✗
0,0,1	0	✓
0,1,0	0	✓
0,2,1	1	✓
0,2,3	2	✓

```
int med_broken(int a, int b, int c) {
    int result;
    if ((a==b) || (a==c) ||
        (b<a && a<c) ||
        (c<a && a<b))
        result = a;
    if (b < a)
        result = c;
    else if (b<a) (b==c) || (a<b && b<c) ||
        (c<b && b<a))
        result = b;
    else if (a<c && c<b)
        result = c;
    return result;
}
```

Input	Expected	Pass?
0,0,0	0	✓
2,0,1	1	✗
0,0,1	0	✓
0,1,0	0	✓
0,2,1	1	✓
0,2,3	2	✓

```
int med_broken(int a, int b, int c) {
    int result;
    if ((a==b) || (a==c) ||
        (b<a && a<c) ||
        (c<a && a<b))
        result = a;
    if (b < a)
        result = c;
    if (b<a) (b==c) || (a<b && b<c) ||
        (c<b && b<a))
        result = b;
    if (a<c && c<b)
        result = c;
    return result;
}
```

Input	Expected	Pass?
0,0,0	0	✓
2,0,1	1	✗
0,0,1	0	✓
0,1,0	0	✓
0,2,1	1	✓
0,2,3	2	✓

```
int med_broken(int a, int b, int c) {
    int result;
    if ((a==b) || (a==c) ||
        (b<a && a<c) ||
        (c<a && a<b))
        result = a;
    if (b < a)
        result = c;
    else if (b<a) (b==c) || (a<b && b<c) ||
        (c<b && b<a))
        result = b;
    else if (a<c && c<b)
        result = c;
    return result;
}
```

Input	Expected	Pass?
0,0,0	0	✓
2,0,1	1	✓
0,0,1	0	✓
0,1,0	0	✓
0,2,1	1	✓
0,2,3	2	✓

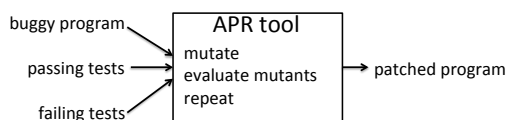
```
int med_broken(int a, int b, int c) {
    int result;
    if ((a==b) || (a==c) ||
        (b<a && a<c) ||
        (c<a && a<b))
        result = a;
    if ((b==c) || (a<b && b<c) ||
        (c<b && b<a))
        result = b;
    if (a<c && c<b)
        result = c;
    return result;
}
```

Input	Expected	Pass?
2,6,8	6	✓
2,8,6	6	✓
6,2,8	6	✓
6,8,2	6	✓
8,2,6	6	✗
8,6,2	6	✓
9,9,9	9	✓

```
int med_broken(int a, int b, int c) {
    int result;
    if ((a==b) || (a==c) ||
        (b<a && a<c) ||
        (c<a && a<b))
        result = a;
    if (b < a)
        result = c;
    else if (b<a) (b==c) || (a<b && b<c) ||
        (c<b && b<a))
        result = b;
    else if (a<c && c<b)
        result = c;
    return result;
}
```

Input	Expected	Pass?
0,0,0	0	✓
2,0,1	1	✓
0,0,1	0	✓
0,1,0	0	✓
0,2,1	1	✓
0,2,3	2	✓
2,6,8	6	✓
2,8,6	6	✓
6,2,8	6	✗
6,8,2	6	✓
8,2,6	6	✓
8,6,2	6	✗
9,9,9	9	✓

Potential solution



Use an independent test suite to measure quality of the patch

Focus of prior evaluations

- Most evaluations are interested in whether tools work
 - produce patches
- Some interest in other factors
 - human acceptance of patches [Durieux et al. 2015] [Fry et al. 2012] [Kim et al. 2013]
 - plausibility [Qi et al. 2015]
 - ...but these don't fully assess functional correctness
- No evaluations test functional correctness of repair outputs independently of repair inputs

What do we need?

- We need bugs with 2 test suites
 - and the test suites need to be good
- Why?
- it's hard enough to find one good test suite, good luck finding programs with two

Make your own!

<http://repairbenchmarks.cs.umass.edu>

998 student-written buggy C programs

- simple (very small)
- have 2 test suites
 - white-box (generated by KLEE)
 - black-box (written by instructor)

Some programs fail some wb tests, others bb tests, others, some of both

RQ1:

What is the base incidence of overfitting?

Give a repair tool the buggy program and the black-box test suite, try to repair it, see what fraction of the white-box tests the patches pass.

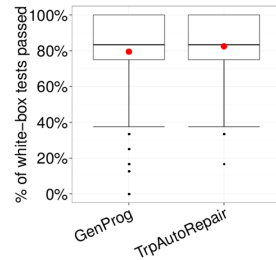
RQ1:

What is the base incidence of overfitting?

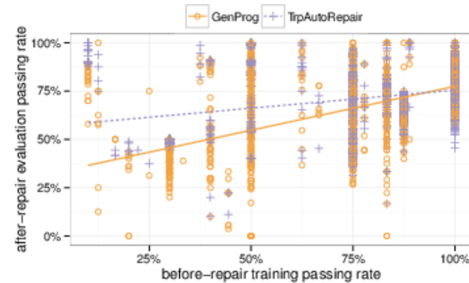
but first, how often can we actually generate patches?

repair tool	patch production %
GenProg	466/778 = 59.9%
TrpAutoRepair	444/778 = 57.1%

RQ1: What is the base incidence of overfitting?

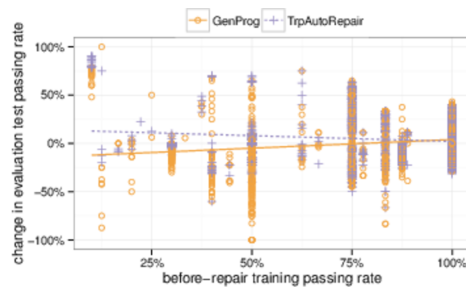


RQ2: What effect do pre-repair test failures have on overfitting?



Programs that fail more tests before repair still fail more tests after repair

RQ2: What effect do pre-repair test failures have on overfitting?

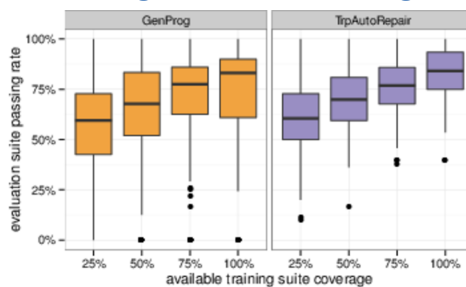


Repair is at best unlikely to improve correctness, at worst likely to worsen it

RQ3: What effect does test suite coverage have on overfitting?

- Randomly sample 25%, 50%, and 75% of passing and failing tests for each buggy program
- Attempt to repair programs
 - with each level of test coverage
- If a repair is found, measure correctness of repair

RQ3: What effect does test suite coverage have on overfitting?

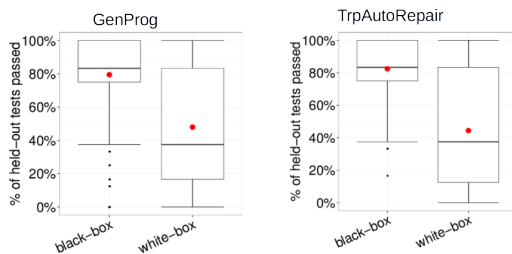


Lower test suite coverage leads to more overfitting

RQ4: What effect does test suite provenance have on overfitting?

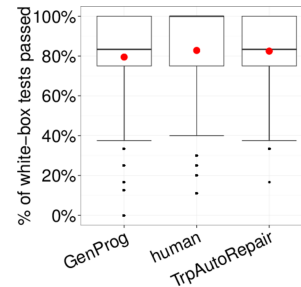
- So far, all experiments have used human-written *black-box* tests to build repairs
- Switch to using KLEE-generated *white-box* tests
- Attempt to repair programs
- If a repair is found, measure correctness of repair
 - this time with *black-box* tests

RQ4: What effect does test suite provenance have on overfitting?



Automatically generated tests produced significantly buggier repairs compared to human-written tests

RQ4: Do tools do better than novices?



Summary

- Overfitting is a real concern
 - median patch for either tool passed only 75% of evaluation suite
- Overfitting is hard to avoid
 - minimization doesn't help on this dataset
 - N-version voting only works in extreme cases
- Program repair is harder for buggier programs, but likely to break more correct programs
- Novice developers don't significantly beat repair tools

So is there no hope?

- SearchRepair, a brand new technique, reduces overfitting to **97.2%**.
- Most SearchRepair repairs pass 100% of the held-out test suite.
(Select few poor repairs drop the overall rate.)

Read more about SearchRepair:

<http://people.cs.umass.edu/~brun/pubs/pubs/Ke15ase.pdf>