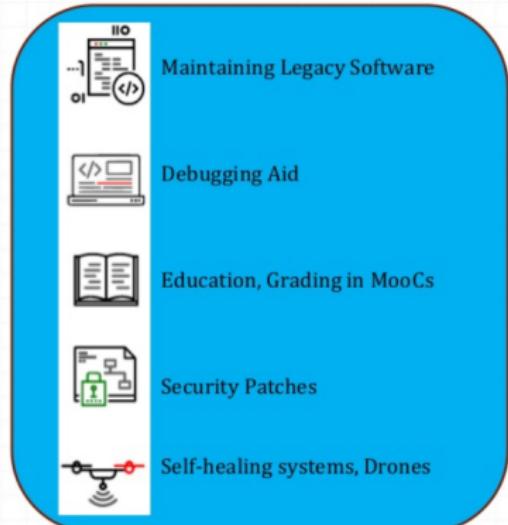
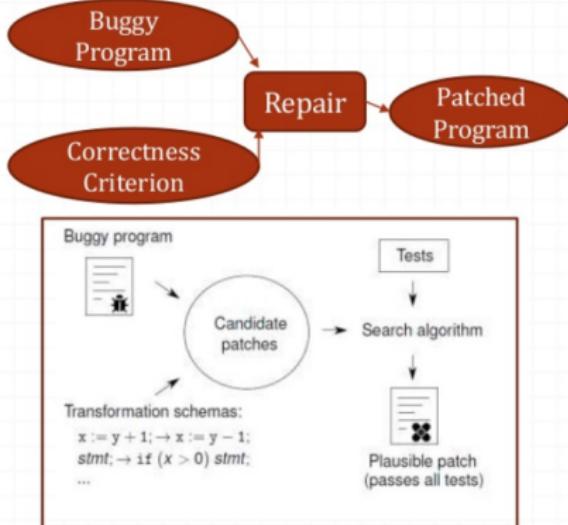


Program Repair

selected slides from Prof Abhik Roychoudhury's keynote talk

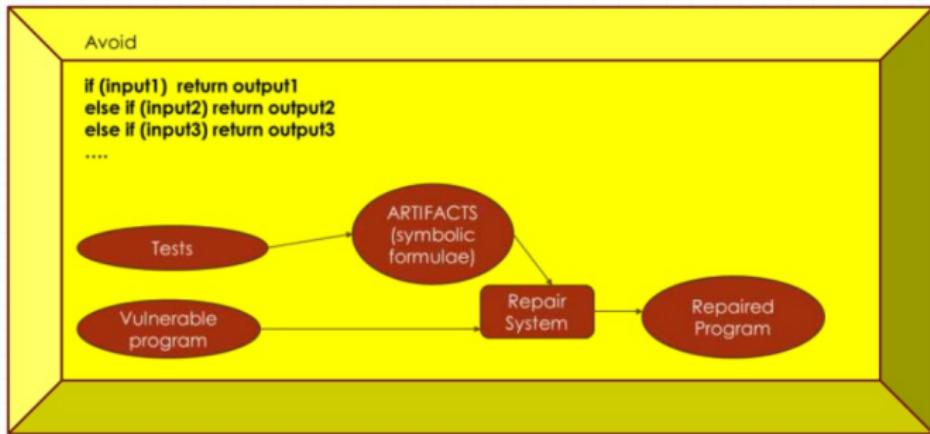
March 12, 2020

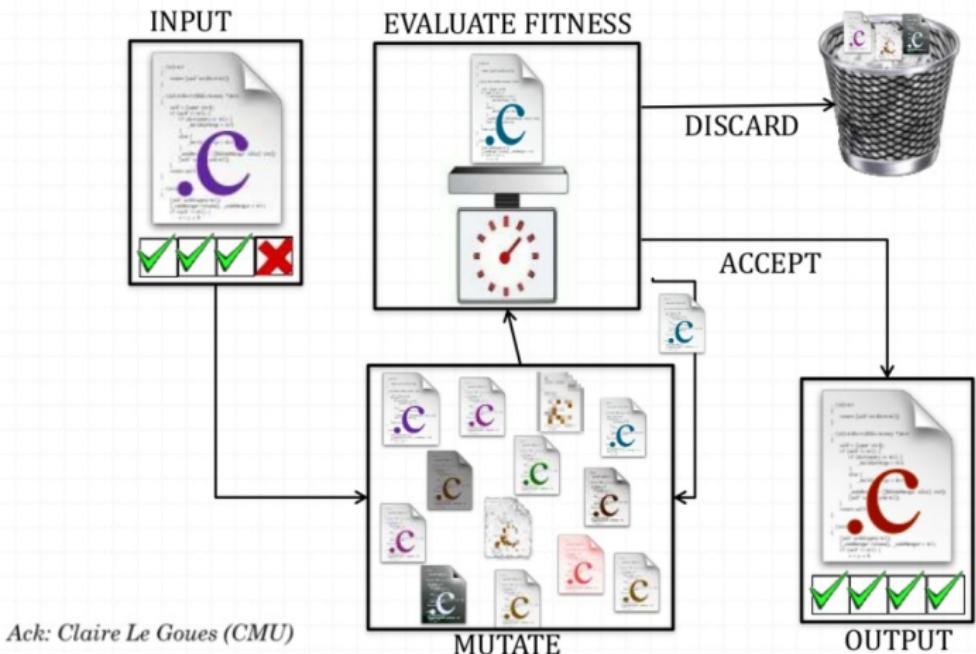
Repair - Why?



Troubles with repair

- Weak description of intended behavior / *correctness criterion* e.g. tests
 - Possibility to use "Bugs as deviant behavior" philosophy
- Weak applicability of repair techniques e.g. only overflow errors
- *Large search space* of candidate patches for general-purpose repair tools.
- Patch suggestions and *Interactive Repair*





```

1 int triangle(int a, int b, int c){
2     if (a <= 0 || b <= 0 || c <= 0)
3         return INVALID;
4     if (a == b && b == c)
5         return EQUILATERAL;
6     if (a == b || b != c) // bug!
7         return ISOSCELES;
8 return SCALENE;
9 }
```

Correct fix
 $(a == b \parallel b == c \parallel a == c)$

Traverse all *mutations* of line 6, and check

Hard to generate correct fix since $a == c$ never appears elsewhere in the program.

OR

Generate the constraint

$$f[2,2,3] \wedge f[2,3,2] \wedge f[3,2,2] \wedge \neg f[2,3,4]$$

and get the solution

$$f[a,b,c] = (a == b \parallel b == c \parallel a == c)$$

Test id	a	b	c	oracle	Pass
1	-1	-1	-1	INVALID	pass
2	1	1	1	EQUILATERAL	pass
3	2	2	3	ISOSCELES	pass
4	2	3	2	ISOSCELES	fail
5	3	2	2	ISOSCELES	fail
6	2	3	4	SCALENES	fail

Comparison

Syntactic Program Repair

```
Syntax-based Schematic
for e in Search-space{
    Validate e against Tests
}
```

1. Where to fix, which line?
2. Generate patches in the candidate line
3. Validate the candidate patches against correctness criterion.

Semantic Program Repair

```
Semantics-based Schematic
for t in Tests {
    generate repair constraint  $\Psi_t$ 
}
Synthesize e from  $\bigwedge_t \Psi_t$ 
```

1. Where to fix, which line(s)?
2. What values should be returned by those lines, e.g. `<inp ==1, ret== 0>`
3. What are the expressions which will return such values?

The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. This weakness allows stealing the information protected, under normal conditions, by the SSL/TLS encryption used to secure the Internet. SSL/TLS provides communication security and privacy over the Internet for applications such as web, email, instant messaging (IM) and some virtual private networks (VPNs).

--- Source: heartbleed.com

```

1 if ( hbtpe == TLS1 HB REQUEST) {
2 ...
3 memcpy (bp , pl , payload );
4 ...
5 }

```

(a) The buggy part of the Heartbleed-vulnerable OpenSSL.

```

1 if ( hbtpe == TLS1 HB REQUEST
2   && payload + 18 < s->s3->rrec.length) {
3 ...
4 }

```

(b) A fix generated automatically



```

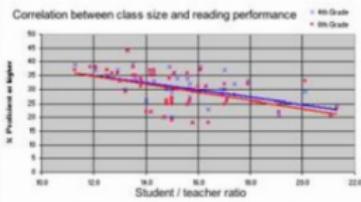
1 if(1 + 2 + payload + 16 > s->s3->rrec.length)
2 return 0;
3 ...
4 if ( hbtpe == TLS1_HB_REQUEST) {
5 ...
6 }
7 else if ( hbtpe == TLS1_HB_RESPONSE) {
8 ...
9 }
10 return 0;

```

(c) The developer-provided repair



Application in education

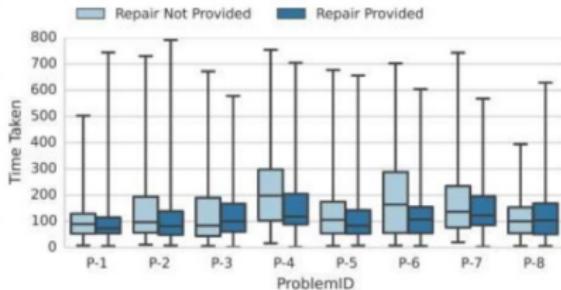


Use program repair in **intelligent tutoring systems** to give the students' individual attention.

Detailed Study in IIT-Kanpur, India by Prof. Amey Karkare and team (FSE17)



Application in Education



43 buggy student submissions from dataset
Across **8** unique problems

37 TA graders volunteered for study
Each TA gets all **43** submissions to grade
With repair hints for half the submissions

Example

```

1 int is_upward( int inhibit, int up_sep, int down_sep){
2     int bias;
3     if (inhibit)
4         bias = down_sep; // bias= up_sep + 100
5     else bias = up_sep ;
6     if (bias > down_sep)
7         return 1;
8     else return 0;
9 }
```

inhibit	up_sep	down_sep	Observed output	Expected Output	Result
1	0	100	0	0	pass
1	11	110	0	1	fail
0	100	50	1	1	pass
1	-20	60	0	1	fail
0	0	10	0	0	pass

What it should have been

Inhibit	<code>up_sep ==</code>	<code>down_sep ==</code>
<code>= 1</code>	<code>11</code>	<code>110</code>

```

1 int is_upward( int inhibit, int up_sep, int
2     down_sep){
3     int bias;
4     if (inhibit)
5         bias = f(inhibit, up_sep, down_sep)
6     else bias = up_sep ;
7     if (bias > down_sep)
8         return 1;
9     else return 0;
}
  
```

$$f(1, 11, 110) > 110$$



Symbolic Execution



Fix the suspect

0 Accumulated constraints

- 0 $f(1,11,110) > 110 \wedge$
- 0 $f(1,0,100) \leq 100 \wedge$
- 0 ...

0 Find a f satisfying this constraint

- 0 By fixing the set of operators appearing in f

0 Candidate methods

- 0 Search over the space of expressions
- 0 Program synthesis with fixed set of operators
 - 0 Can also be achieved by second-order constraint solving



0 Generated fix

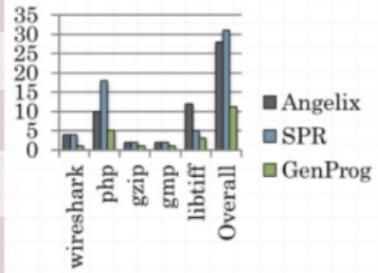
0 `f(inhibit,up_sep,down_sep) = up_sep + 100`

28

(Sample) Results

Subject	LoC	Defect	Fixed Expressions
wireshark	2814K	Libtiff-4a24508-cc79c2b	2
php	1046K	Libtiff-829d8c4-036d7bb	2
gzip	491K	CoreUtils-00743a1f-ec48bead	3
gmp	145K	CoreUtils-1dd8a331-d461bfd2	2
libtiff	77K	CoreUtils-c5ccf29b-a04ddb8d	3

Defect	Fixed Expressions
Libtiff-4a24508-cc79c2b	2
Libtiff-829d8c4-036d7bb	2
CoreUtils-00743a1f-ec48bead	3
CoreUtils-1dd8a331-d461bfd2	2
CoreUtils-c5ccf29b-a04ddb8d	3



	#Fixes	Del	Del, Per
Angelix	28	5	18%
SPR	31	13	42%

SemGraft Results

	Program	Commit	Bug	Angelix	SemGraft
GNU Coreutils as reference	sed	c35545a	Handle empty match	Correct	Correct
	seq	f7d1c59	Wrong output	Correct	Correct
	sed	7666fa1	Wrong output	Incorrect	Correct
	sort	d1ed3e6	Wrong output	Incorrect	Correct
	seq	d86d20b	Don't accepts 0	Incorrect	Correct
	sed	3a9365e	Handle s///	Incorrect	Correct
Linux Busybox as reference	Program	Commit	Bug	Angelix	SemGraft
	mkdir	f7d1c59	Segmentation fault	Incorrect	Correct
	mkfifo	cdb1682	Segmentation fault	Incorrect	Correct
	mknod	cdb1682	Segmentation fault	Incorrect	Correct
	copy	f3653f0	Failed to copy a file	Correct	Correct
	md5sum	739cf4e	Segmentation fault	Correct	Correct
	cut	6f374d7	Wrong output	Incorrect	Correct