Take nothing on its looks; take everything on evidence. There's no better rule.

Charles Dickens, "Great Expectations."

Repairing Bugs in Conditional Expressions

Favio DeMarco

Universidad de Buenos Aires - INRIA

August 28th, 2013

Context
So far, the Universe is winning.

> *Bug fixing continues to be a mostly manual, time consuming, and therefore expensive activity in software development.*

Hoang Duong Thien Nguyen et al, "SemFix: Program Repair via Semantic Analysis"

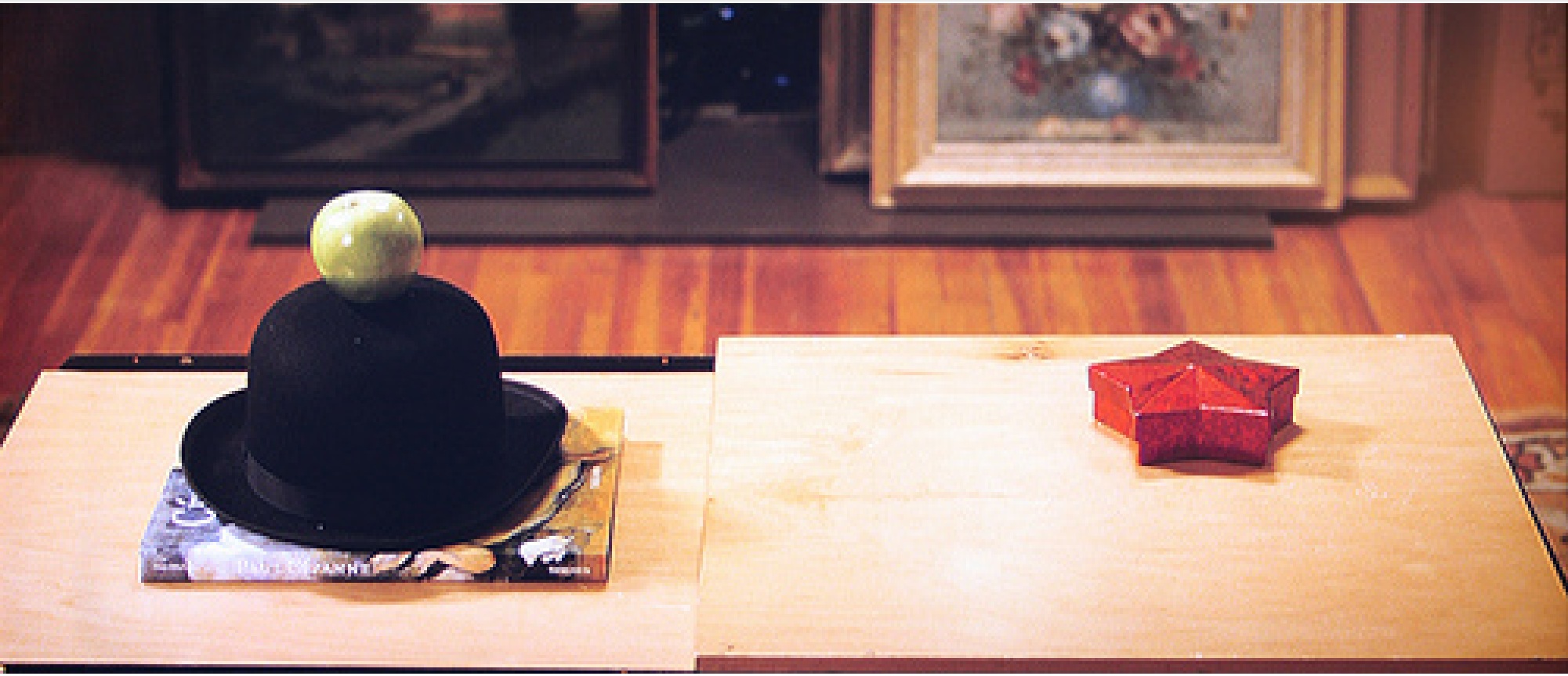
Case study
Commons Math - MathUtils class

```
public static int gcd(int u, int v) {
  if (u * v == 0) {
    return (Math.abs(u) + Math.abs(v));
  }
...
```

```
assertEquals(3 * (1<<15)
        , gcd(3 * (1<<20), 9 * (1<<15)));
```

```
public static int gcd(int u, int v) {
    if ((u == 0) || (v == 0)) {
        return (Math.abs(u) + Math.abs(v));
    }
    // ...
}
```

State of the art



State of the art

SemFix: Program Repair via Semantic Analysis

- Statement ranking (Tarantula) $\rightarrow$
- Symbolic Execution (KLEE) $\rightarrow$
- Repair Constraint $\rightarrow$ Program Synthesis (SOLFP[1] -paper-)

[1] Synthesis of Loop-free Programs

State of the art

GenProg: A Generic Method for Automatic Software Repair
Claire Le Goues, ThanhVu Nguyen, Stephanie Forrest, Westley Weimer

State of the art

ClearView, AutoFix-E, Gopinath et al, Pachika.

Problems

Unwillingness to share code.

Quality is free, but only to those who are willing to pay heavily for it.

Tom DeMarco, Peopleware

Limitations

Resources (time, code monkeys, knowledge, tools, etc.).

- Statement ranking (GZoltar) $\rightarrow$
- Ad hoc code manipulation and values capturing (OGCBPS[2] -paper-) $\rightarrow$
- Repair Constraint $\rightarrow$
- Program Synthesis (SOLFP[3] -paper-)

[2] Oracle-Guided Component-Based Program Synthesis
[3] Synthesis of Loop-free Programs

Experimental methodology

Seeded and wild bugs.

Evaluation / Validation

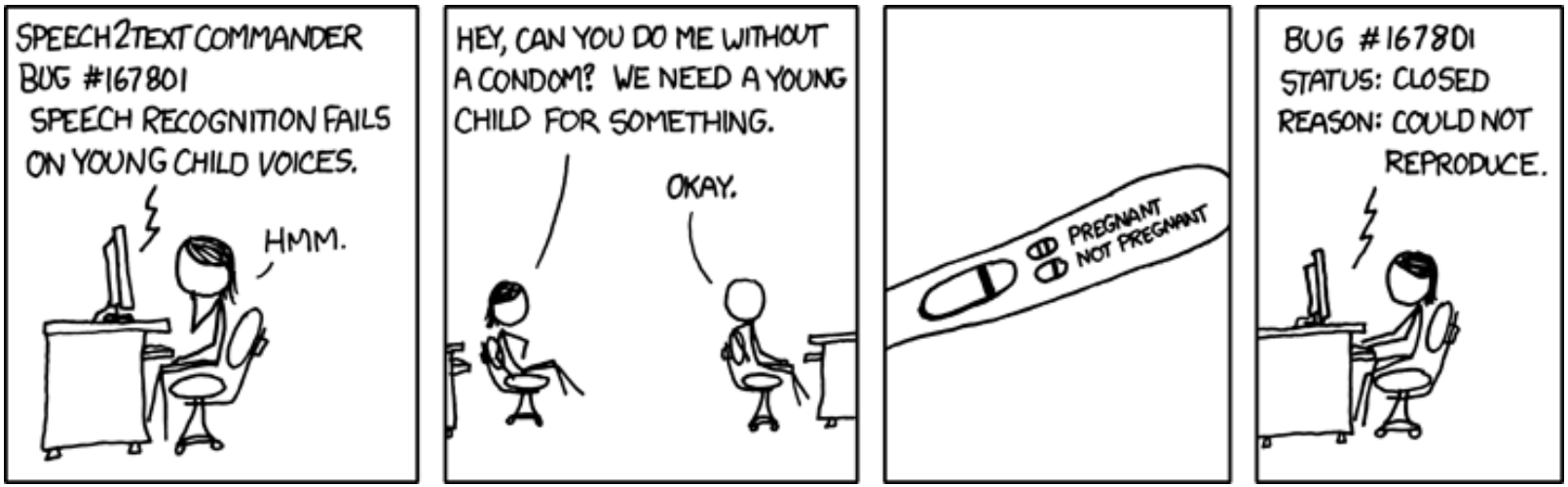
Generated patches vs. reality.

Perspectives

Conclusion

Contribution

Can't and shouldn't.



Case study
Commons Math - MathUtils class

```
411: public static int gcd(int u, int v) {
412:     if (u * v == 0) {
413:         return (Math.abs(u) + Math.abs(v));
414:     }
...
```

```
assertEquals(3 * (1<<15)
        , gcd(3 * (1<<20), 9 * (1<<15)));
```

Case study
Statement ranking (GZoltar)

```
MathUtils:413 Suspiciousness 0.2357022603955158 7
MathUtils:431 Suspiciousness 0.154303349962091 9

...

MathUtils:460 Suspiciousness 0.1132277034144595 6
MathUtils:412 Suspiciousness 0.1118033988749894 8
```

Case study
Ad hoc code manipulation and values capturing (OGCBPS -paper-)

```
411: public static int gcd(int u, int v) {
412:   if (true) {
413:     return (Math.abs(u) + Math.abs(v));
414:   }
...
```

$$(u = 0 \wedge v = 0 \Rightarrow output = true) \wedge$$

$$(u = 0 \wedge v = 55 \Rightarrow output = true) \wedge$$

$$\ldots$$

$$(u = 77 \wedge v = 55 \Rightarrow output = false)$$

```
public static int gcd(int u, int v) {
    if ((u == 0) || (v == 0)) {
        return (Math.abs(u) + Math.abs(v));
    }
    // ...
}
```