A Critical Review of "Automatic Patch Generation Learned from Human-Written Patches":

Essay on the Problem Statement and the Evaluation of Automatic Software Repair

Martin Monperrus
University of Lille & Inria, France

ICSE'2014, Hyderabad, India

Automatic software repair is the process of fixing bugs automatically.

Offline

Weimer, Le Goues et al., GenProg, (ICSE'2009)

Source code

+

Failing test cases

+

AST manipulation

Online

Locasto, Sidoroglou, Keromytis, (NDSS'2006)

Binary code

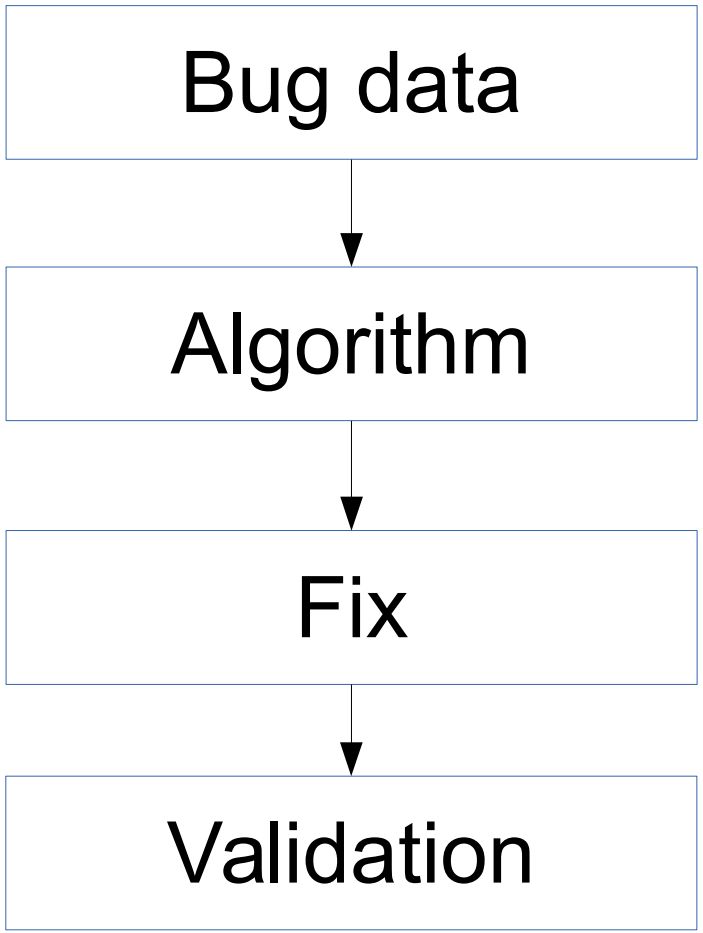
+

Crash

+

Error-return insertion

Diversity of repair problem statements



Bug oracle

- Automated? (Genprog, ...)
- Unstructured? (Liu et al., ICST'13)
- Manual? (Carzaniga et al., FSE'10)

- Is the bug fixed? Bug oracle
- Is there a regression? Regression oracle

Classical repair: Test-suite based program repair

Problem statement: Bug oracle = Test suite = Regression oracle

Instances: GenProg, Debroy & Wong (ICST'2010), SemFix (Nguyen et al., ICSE'2013), PAR (Kim, Nam, Jaewoo and Kim, ICSE'2013)

Open questions:

- How common is it to have a failing test case? (Böhme & Roychoudhury, ISSTA'2014)
- Can we trust test suites as regression oracles?

How to evaluate an automatic repair approach?

Technical Contribution: A new drug for headache.

Evaluation: Apply it to random patients of a physician's list

BIAS

Comparison:

- New drug from top lab A (influenza)
- New drug from top lab B (back pain)

Evaluation: Apply them to 10000 random patients

(incl. 2000 influenza, 1000 back pains, 7000 others)

Result:

- 1000 healed with A
- 900 healed with B

Conclusion:

~~A is better~~

Efficiency of A: 1000/2000=50%
Efficiency of B: 900/1000 = 90%

Comparison:

- New repair technique from top lab A (defect class?)
- New repair technique from top lab B (defect class?)

Evaluation: Apply it to 100+ real bugs of large-scale software (inclusion criteria?)

Bias in “Automatic Patch Generation Learned from Human-Written Patches”

We need explicit defect classes

Examples:
Nguyen, Qi, Roychoudhury, Chandra (ICSE'13): wrong arithmetic assignments, buggy if conditions (Semfix)

Gopinath et al. (ICSE'14): selection condition of database statements

We need explicit defect classes

Two conditions for conclusive comparison:

- Clear and same defect class
- Sound dataset composition

Example: a dataset of buggy if conditions

Acknowledgements (before concluding)

- Dongsun Kim, Jaechang Nam, Jaewoo Song, and Sunghun Kim, the authors of PAR at ICSE'13
- Matias Martinez, Benoit Cornu, Raphael Marvie, Favio DeMarco, Tegawendé F. Bissyandé, Jifeng Xuan, Lionel Seinturier, Benoit Baudry, Jean-Marc Jézéquel, Yves Le Traon, Friedrich Steimann, and Bertrand Meyer
- The Software Engineering community and the ICSE crowd

Conclusion

"We need explicit and well-defined defect classes

for automatic software repair"

Future work:

- Improving the repairability of a particular defect class
- Identifying new defect classes for automatic software repair
- Identify new bug oracles (Liu et al., ICSE-NIER'14)