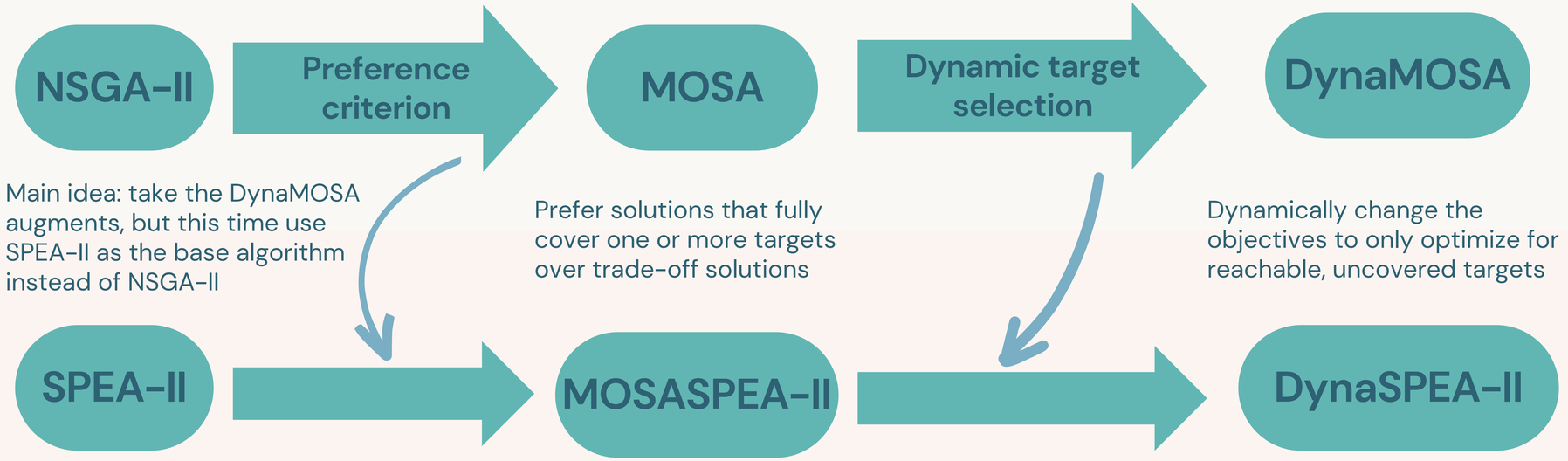
Investigating the performance of SPEA-II on automatic test case generation

Author: Erwin Li - C.R.E.Li@student.tudelft.nl
Supervisors: Annibale Panichella,
Mitchell Olsthoorn, Dimitri stallenberg

Approach



Introduction

Manual software testing is time consuming and tedious, so automatic testing has become an active research field

Genetic algorithms have been shown to be effective in automatic test case generation [1]

Current state-of-the-art tool for JavaScript is Syntest, which implements the DynaMOSA algorithm [2]

DynaMOSA is based on NSGA-II and augmented with domain knowledge. But what if we use a different genetic algorithm as the base?

SPEA-II is a genetic algorithm, that has been successfully applied to solve multi-objective problems. It has performed better than NSGA-II in some problems with higher-dimensional objective spaces. [3]

Study design

RQ1: How effective are the additions of DynaMOSA features to SPEA-II?

RQ2: How does DynaSPEA-II perform compared to DynaMOSA on branch coverage?

Implement all SPEA variants in the Syntest framework

Run these algorithms on a benchmark of 36 diverse classes

Measure branch coverage for each algorithm.

Compare results via statistical analysis and see if an algorithm is statistically superior

Results

Overall stats

Median branch coverage
- SPEA-II: 0.394
- DynaSPEA-II 0.405
- DynaMOSA: 0.399

SPEA-II VS DynaSPEA-II

- DynaSPEA-II had higher coverage in 13.9% of classes
- Average increase of 4.92%
- Increase ranges from 2.2% to 16.7%

DynaSPEA-II VS DynaMOSA

- No statistically significant difference was found
- Algorithms have equal performance

Conclusion

Main findings

- The DynaMOSA enhancements are effective in increasing branch coverage in SPEA-II
- DynaSPEAII performs equally to DynaMOSA

Future work

- Increase benchmark size and diversity
- Conduct hyperparameter experiments
- Conduct experiments in other programming language

References

[1] M Moein Almasi, Hadi Hemmati, Gordon Fraser, Andrea Arcuri, and Janis Benefelds. 2017. An industrial evaluation of unit test generation: Finding real faults in a financial application. In 2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP). IEEE, 263–272.
[2] Annibale Panichella, Fitsum Meshesha Kifetew, and Paolo Tonella. Automated test case generation as a many-objective optimisation problem with dynamic selection of the targets. IEEE Transactions on Software Engineering, 44(2):122–158, 2017
[3] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. 2001. SPEA2: Improving the strength Pareto evolutionary algorithm. TIK-report 103 (2001).